Operate your Oracle Cloud Infrastructure from the command line

Nelson Calero

HrOUG - Croatia Oracle UG
October 16th, 2019

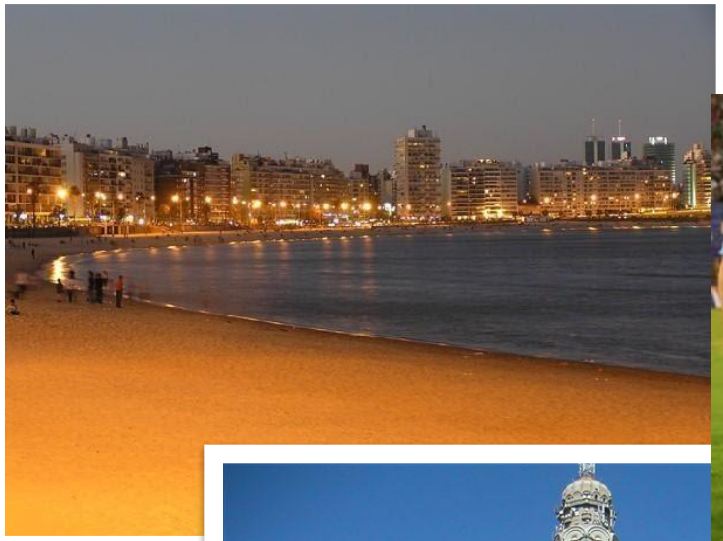
Pythian

Today's topics

- Oracle cloud concepts (quick review)
- CLI tools (focus on OCI)
- Examples
- Automation

Intended audience: DBAs interested in Cloud and DevOps





About me

- Principal Consultant at Pythian – several roles since 2014
- Working with Oracle tools and Linux environments since 1996
- DBA Oracle (2001) & MySQL (2005)
- Co-founder and President of the Oracle user Group of Uruguay (2009)
- LAOUC Director of events (2013)

- Computer Engineer (1998)
- Oracle ACE (2014), Oracle ACE Director (2017)
- Oracle Certified Professional 10g/11g/12c, OCE, Cloud DB & Infra
- Amazon Solutions Architect – Associate (2016)
- Google Cloud Architect (2017), Google Cloud Data Engineer (2017)
- Oracle University Instructor (2011)
- Blogger and speaker: Oracle Open World, Collaborate, OTN Tour, Regional conferences

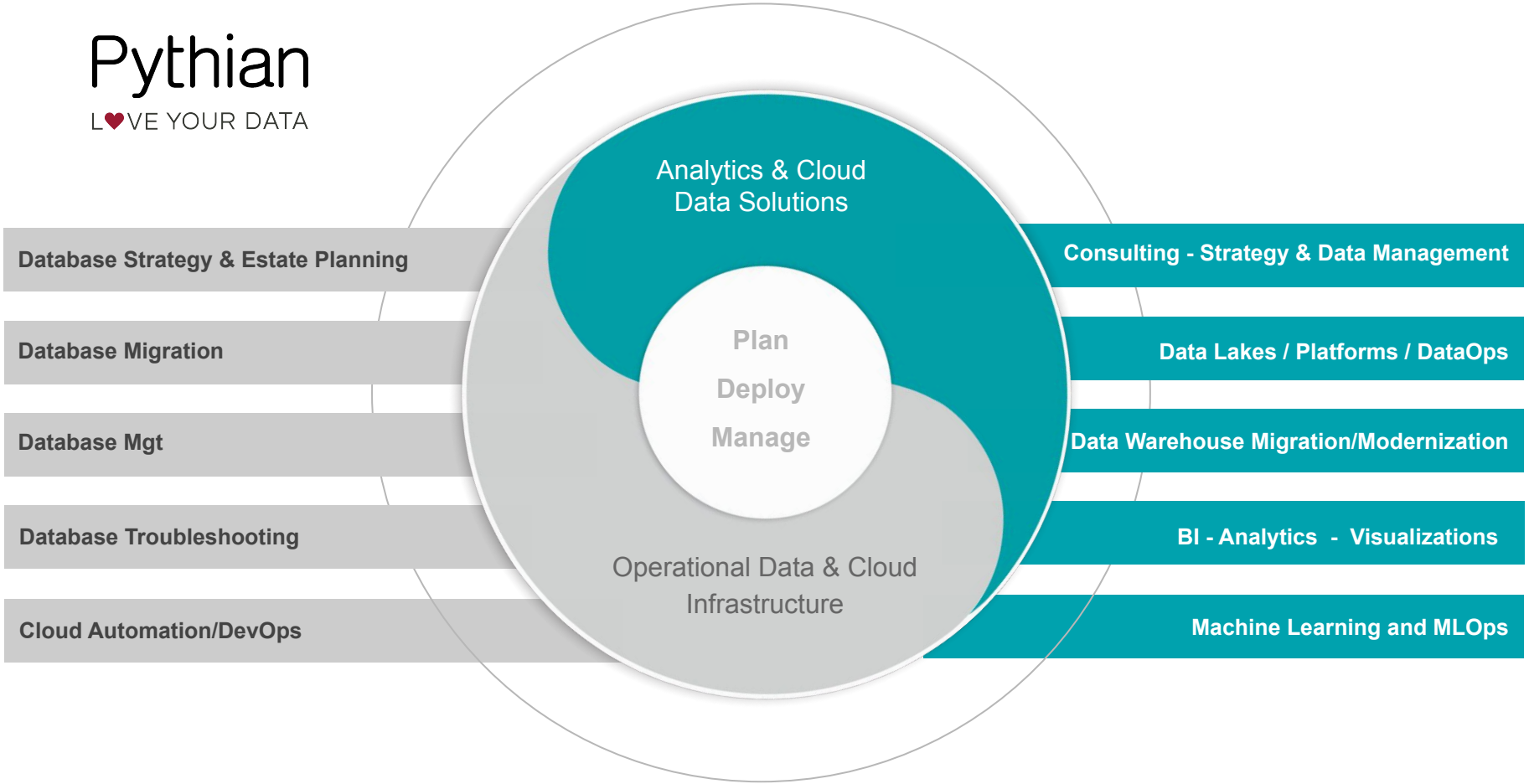

http://www.linkedin.com/in/ncalero


@ncalerouy

Pythian
love your data

Pythian

LOVE YOUR DATA



500+ Technical Experts Helping Peers Globally

ORACLE ACE Program

ORACLE ACE Director

ORACLE ACE

ORACLE ACE Associate

---

3 Membership Tiers

- Oracle ACE Director
- Oracle ACE
- Oracle ACE Associate

bit.ly/OracleACEProgram

Connect:

oracle-ace_ww@oracle.com

Facebook.com/oracleaces

@oracleace

Oracle Groundbreakers

Oracle Cloud Infrastructure

New Free Tier

oracle.com/gbtour

Always Free

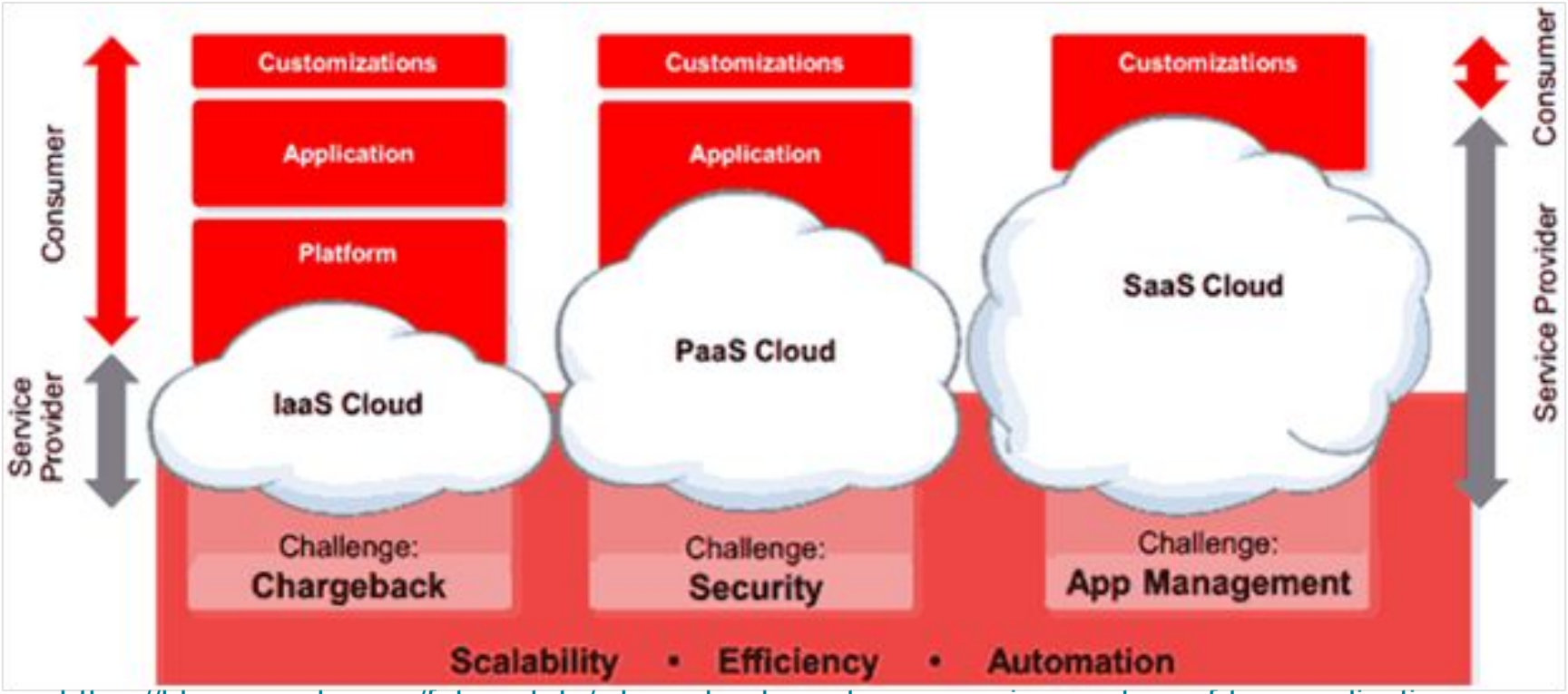
Services you can use for unlimited time

+

30-Day Free Trial

Free credits you can use for more services

Oracle Cloud offering



https://blogs.oracle.com/futurestate/when-cloud-meets-on-premise:-a-story-of-two-applications

Oracle Cloud - IaaS

OCI (new gen. - 2017)

- Audit
- Compute
- Container Engine for Kubernetes
- Container Registry
- Load Balancing
- Object Storage
- Oracle Cloud Infrastructure Database
- Networking
- Ravello
- Wercker

OCI Classic (v1 - 2014)

- Compute Classic
- Container Classic
- Container Registry Classic
- Dedicated Compute Classic - SPARC Model 300
- DIVA (media assets)
- Load Balancing Classic
- Messaging
- Storage Classic
- Storage Software Appliance

https://docs.cloud.oracle.com/iaas/Content/home.htm
https://docs.oracle.com/en/cloud/iaas/index.html

Oracle Cloud - IaaS

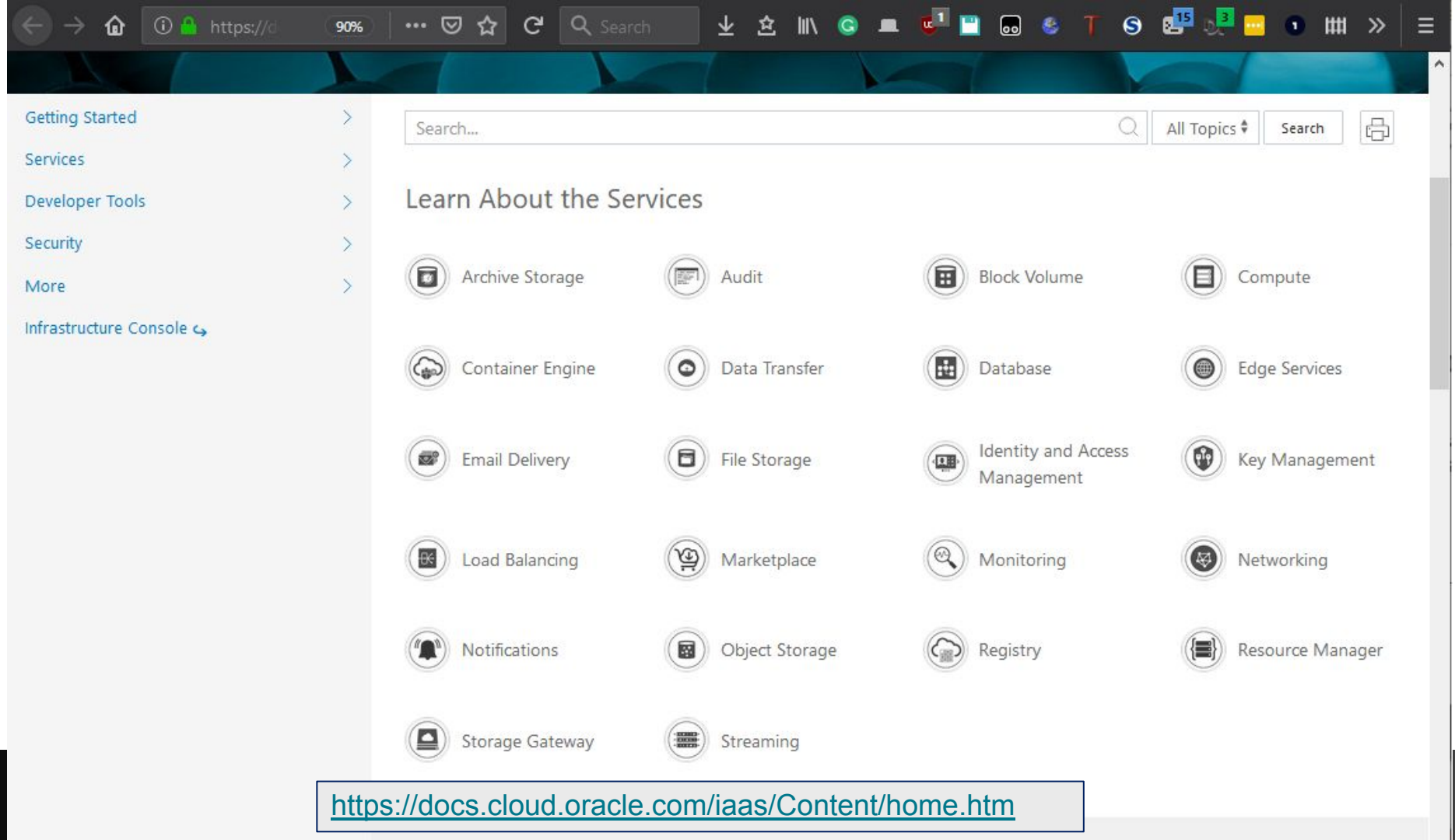
OCI (new gen. - 2017)

- Audit
- Compute
- Container Engine for Kubernetes
- Container Registry
- Load Balancing
- Object Storage
- Oracle Cloud Infrastructure Database
- Networking
- Ravello
- Wercker

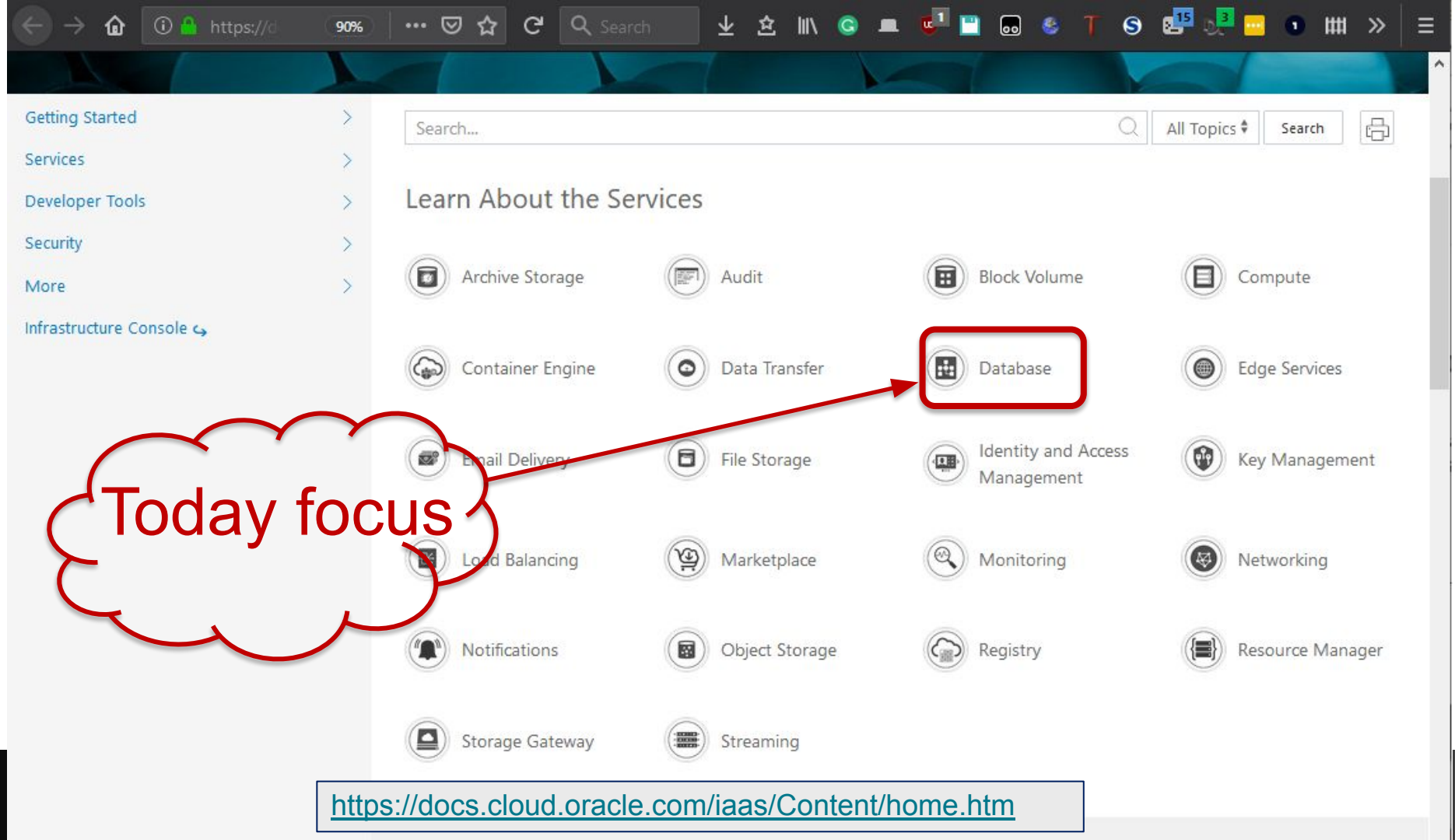
OCI Classic (v1 - 2014)

- Compute Classic
- Container Classic
- Container Registry Classic
- Dedicated Compute Classic - SPARC Model 300
- DIVA (media assets)
- Load Balancing Classic
- Messaging
- Storage Classic
- Storage Software Appliance

https://docs.cloud.oracle.com/iaas/Content/home.htm
https://docs.oracle.com/en/cloud/iaas/index.html



https://docs.cloud.oracle.com/iaas/Content/home.htm



https://docs.cloud.oracle.com/iaas/Content/home.htm

OCI Regions

https://cloud.oracle.com/regions

Instance shapes

OCI Classic

General Purpose Shapes		
Shape	OCPUs	RAM(GB)
OC3	1	7.5
OC4	2	15
OC5	4	30
OC6	8	60
OC7	16	120
OC8	24	180
OC9	32	240

High I/O Shapes			
Shape	OCPUs	RAM(GB)	SSD(GB)
OCIO1M	1	15	400
OCIO2M	2	30	800
OCIO3M	4	60	1600
OCIO4M	8	120	3200
OCIO5M	16	240	6400

High Memory Shapes		
Shape	OCPUs	RAM(GB)
OC1M	1	15
OC2M	2	30
OC3M	4	60
OC4M	8	120
OC5M	16	240
OC8M	24	360
OC9M	32	480

GPU Shapes				
Shape	OCPUs	RAM(GB)	NVMe(GB)	CUDA Cores
OCSG1-M60	3	60	375	2048
OCSG2-M60	6	120	750	4096
OCSG3-M60	12	240	1500	8192
OCSG1-K80	3	60	375	2496
OCSG2-K80	6	120	750	4992
OCSG3-K80	12	240	1500	9984

Not in all regions

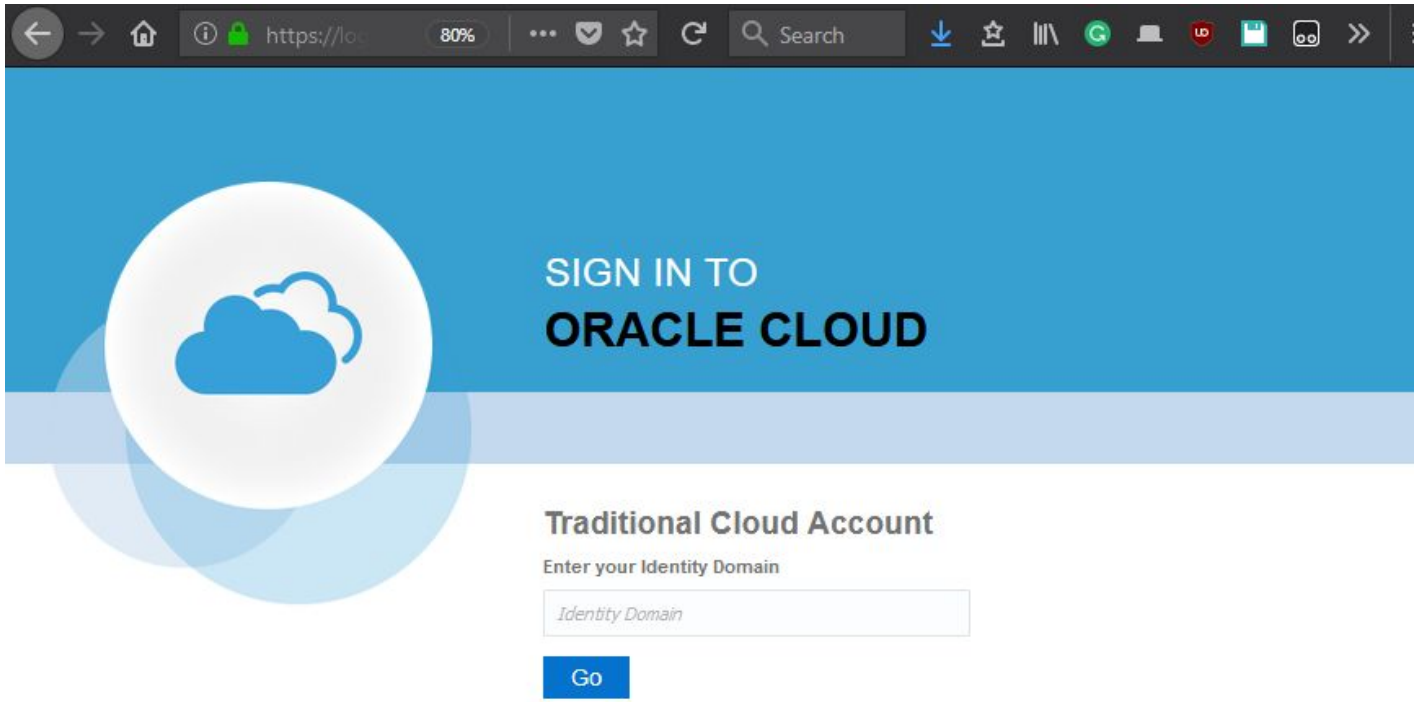
OCI

Compute - Bare Metal Instances				
Instance	Shape	OCPUs	RAM(GB)	Storage
Standard	BM.Standard1.36	36	256	1 PB Max
Standard	BM.Standard2.52	52	768	1 PB Max
Dense I/O	BM.DenseIO1.36	36	512	1 PB Max
Dense I/O	BM.DenseIO2.52	52	768	1 PB Max

Compute - GPU Instances						
Instance	Shape	OCPUs	RAM(GB)	Storage	GPU	Network
Pascal VM GPU	VM.GPU2.1	12	104	1 PB Max	1x P100	1x 25 Gbps
Pascal Bare Metal GPU	BM.GPU2.2	28	192	1 PB Max	2x P100	2x 25 Gbps
Volta VM GPU	VM.GPU3.1	6	90		1x V100	400 Mbps
Volta VM GPU	VM.GPU3.2	12	180		2x V100	800 Mbps
Volta VM GPU	VM.GPU3.4	24	360		4x V100	1x 25 Gbps
Volta BM GPU	BM.GPU3.8	52	768	1Pb Max	8x V100	2x 25 Gbps

Compute - Virtual Machine Instances					
Instance	Shape	OCPUs	RAM(GB)	Storage	Local Disk
Standard	VM.Standard1.1	1	7	1Pb Max	
Standard	VM.Standard2.1	1	15	1Pb Max	
Standard	VM.Standard1.2	2	14	1Pb Max	
Standard	...				
Standard	VM.Standard2.16	16	240	1Pb Max	
Standard	VM.Standard2.24	24	320	1Pb Max	
DenseIO	VM.DenseIO1.4	4	60	1Pb Max	3.2 TB
DenseIO	...				
DenseIO	VM.DenseIO2.24	24	320	1Pb Max	25.6 TB

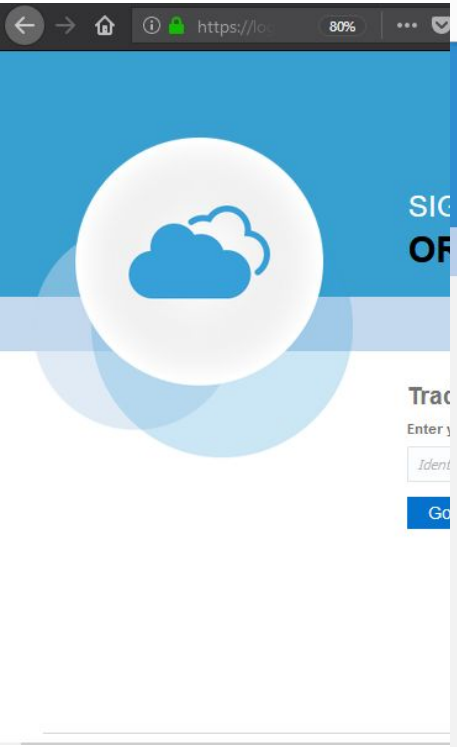
Connecting to the Web console



http://cloud.oracle.com

ORACLE

Pythian
love your data



https://login.us-ashburn-1.oraclecloud.com/

Search

Most Visited Getting Started

ORACLE Cloud Infrastructure

SIGN IN

Signing in to cloud tenant:
iougcloudtrial
Change tenant

Single Sign-On (SSO)

We have detected that your tenancy has been federated to another Identity Provider.

Select your Identity Provider below.

IDENTITY PROVIDER
oracleidentitycloudservice

Continue

or

Oracle Cloud Infrastructure

The login is uncommon for federated accounts. If you have questions, please review the FAQ or contact your tenancy administrator.

USER NAME
nelson.calero@gmail.com

PASSWORD
••••••••••

Sign In Forgot password?

console.us-ashburn-1.oraclecloud.com

ORACLE Cloud

US East (Ashburn)

- Core Infrastructure
  - Compute
  - Block Storage
  - Object Storage
  - File Storage
  - Networking
- Database
  - Bare Metal, VM, and Exadata
  - Autonomous Data Warehouse
  - Autonomous Transaction Processing
  - Data Safe
- Solutions and Platform
  - Analytics
  - Resource Manager
  - Email Delivery
  - Application Integration
  - Monitoring
  - Developer Services
  - Marketplace
- More Oracle Cloud Services

2-6 mins

3-5 mins

AUTONOMOUS TRANSACTION PROCESSING

Create a database

3-5 mins

AUTONOMOUS DATA WAREHOUSE

Create a data warehouse

1-3 mins

ork

2-6 mins

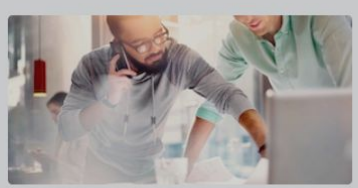
OBJECT STORAGE

Store data

2-3 mins

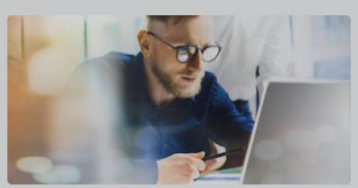
NETWORKING SOLUTIONS

Create an IPsec VPN connection



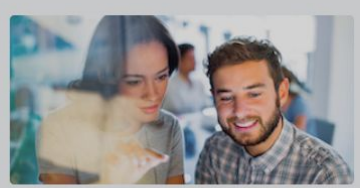
Evaluate the PeopleSoft Validated Solution Architecture

Meet your business and technical goals for PeopleSoft in the cloud.



Migrate custom applications onto Oracle Cloud

Learn the benefits and validated solution for moving custom applications that use Oracle Database to the cloud.



Get a Quick Start on popular data management technologies

From Couchbase to Confluent: deploy complex software quickly on powerful cloud infrastructure.

All systems operational

View health dashboard

Action Center

User Management

Add a user to your tenancy

Billing

Trial credits used

Trial days elapsed 423 / 730

Analyze costs

Upgrade your account

What's New

Announcing Free Tier: Use Always Free resources free of charge for the life of your account

Sep 16, 2019

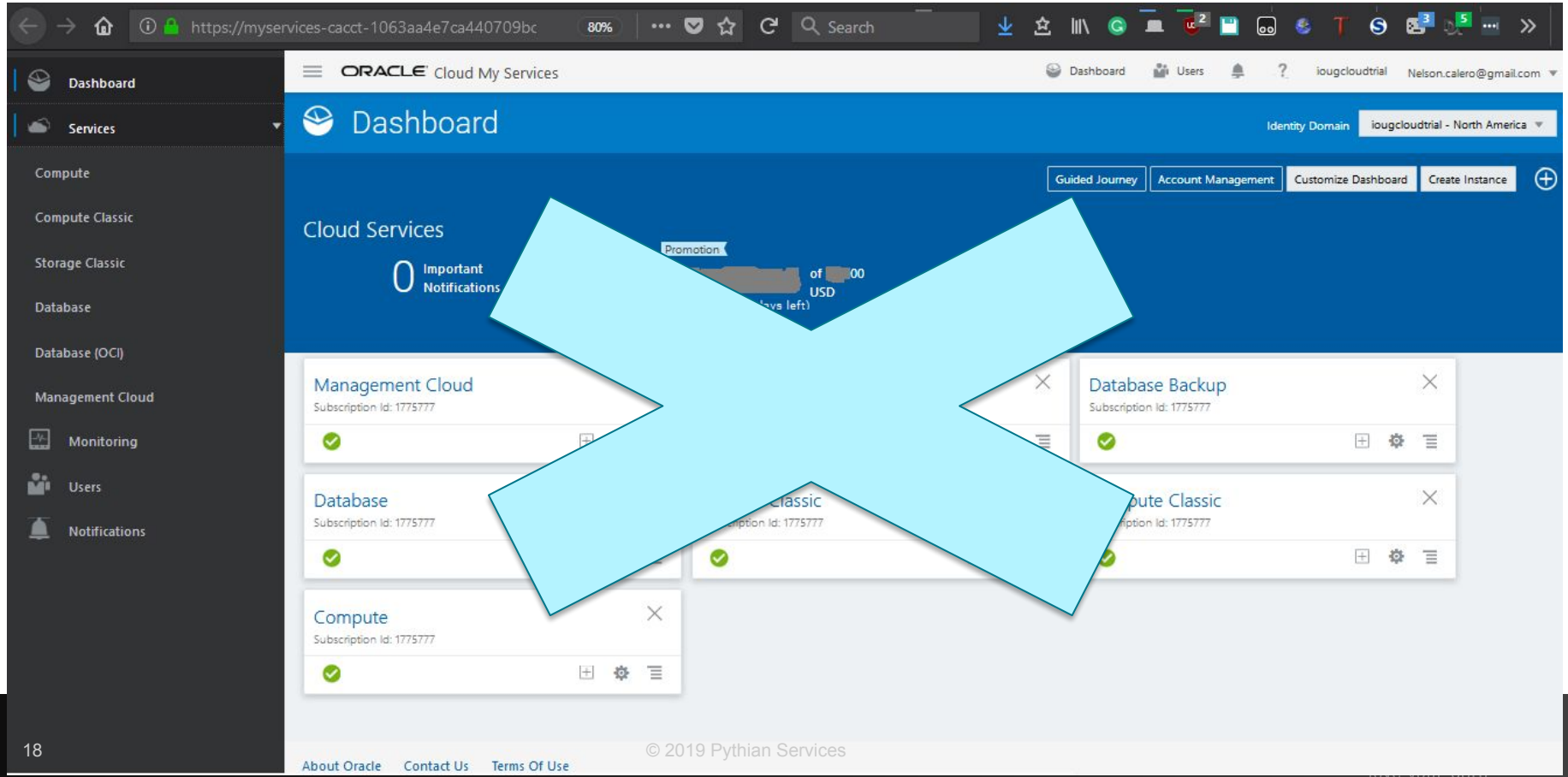
Console experience enhancements unite IaaS, PaaS, and SaaS

Sep 9, 2019

Oracle Functions is now generally available and recommended for event-driven workloads

Jul 31, 2019

Oracle Cloud dashboard – includes Classic

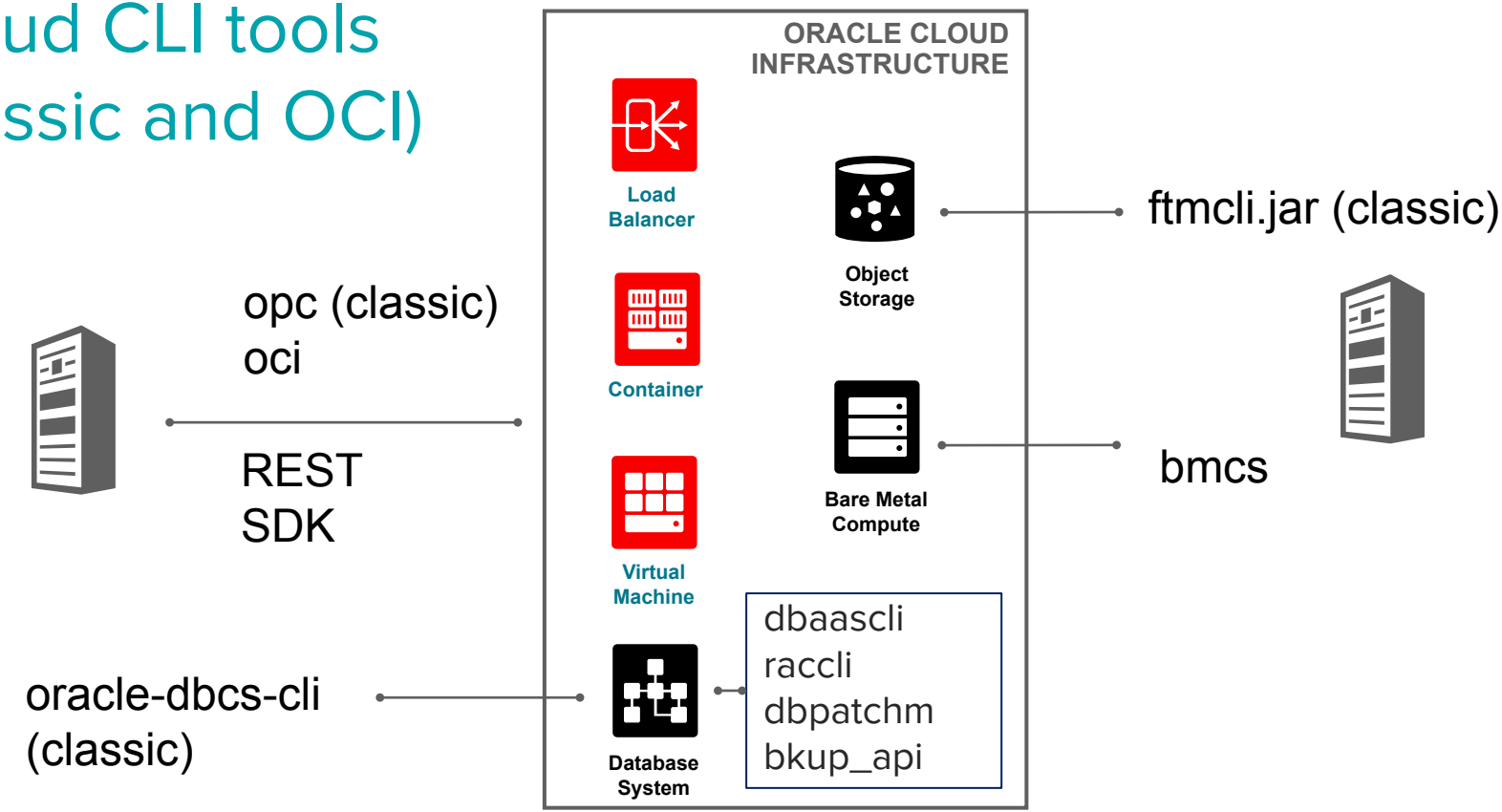


Other ways to access the Oracle Cloud

- REST API - https://docs.cloud.oracle.com/iaas/api/
- SDK - Java, Python, Ruby, Go
- CLI tool – one for all services (not the case for OCI-C)
- Specific for particular services:
  https://docs.cloud.oracle.com/iaas/Content/API/Concepts/devopstools.htm
  - HDFS Connector for Object storage Storage Gateway (NFS)
  - Terraform provider
  - Ansible Modules
  - Chef Knife Plug-in
  - Compute Jenkis Plug-in
  - Grafana Plug-in

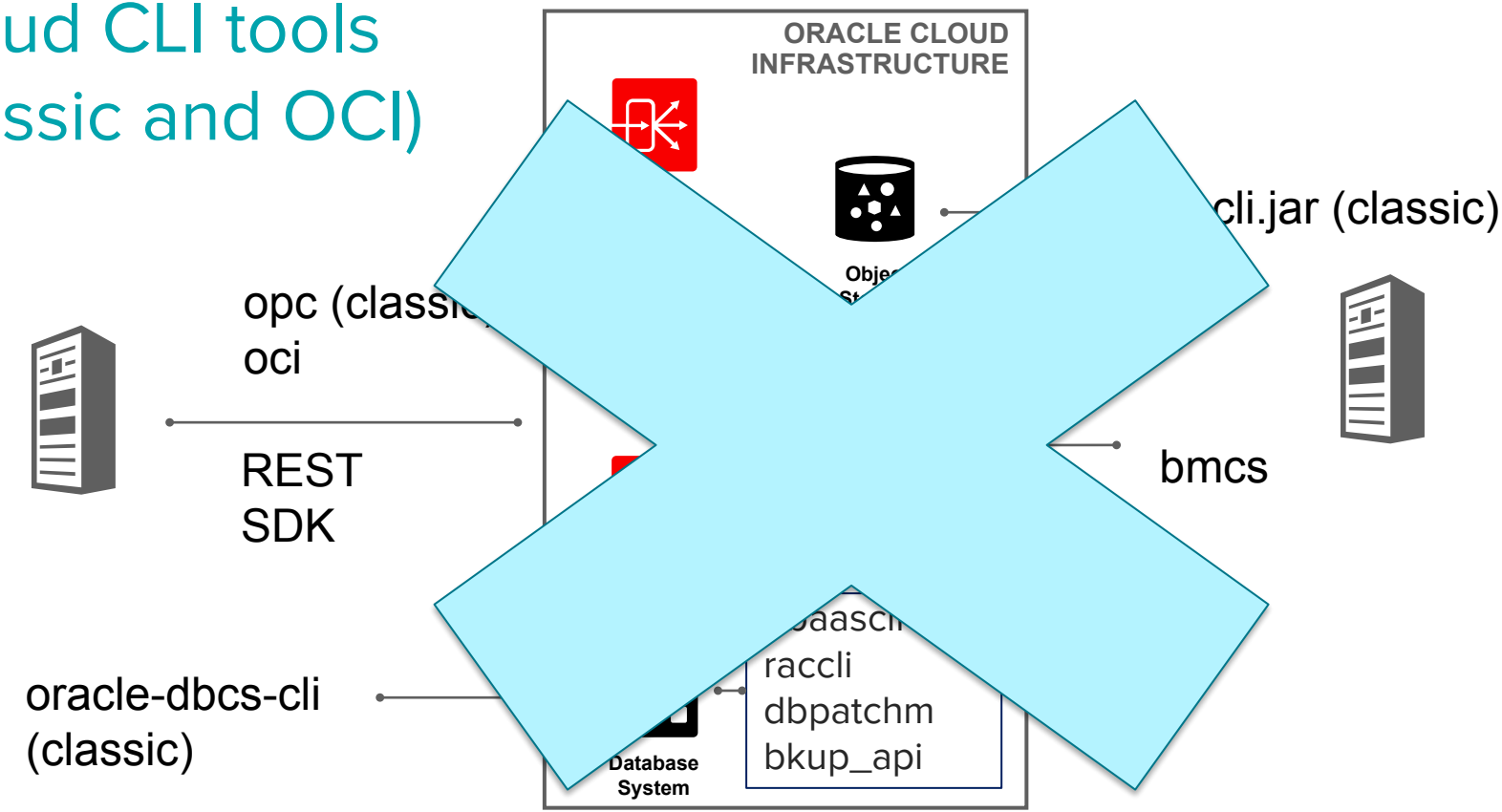
Note: OCI Object storage has an Amazon S3 compatible API -
https://docs.cloud.oracle.com/iaas/Content/Object/Tasks/s3compatibleapi.htm

Cloud CLI tools (classic and OCI)



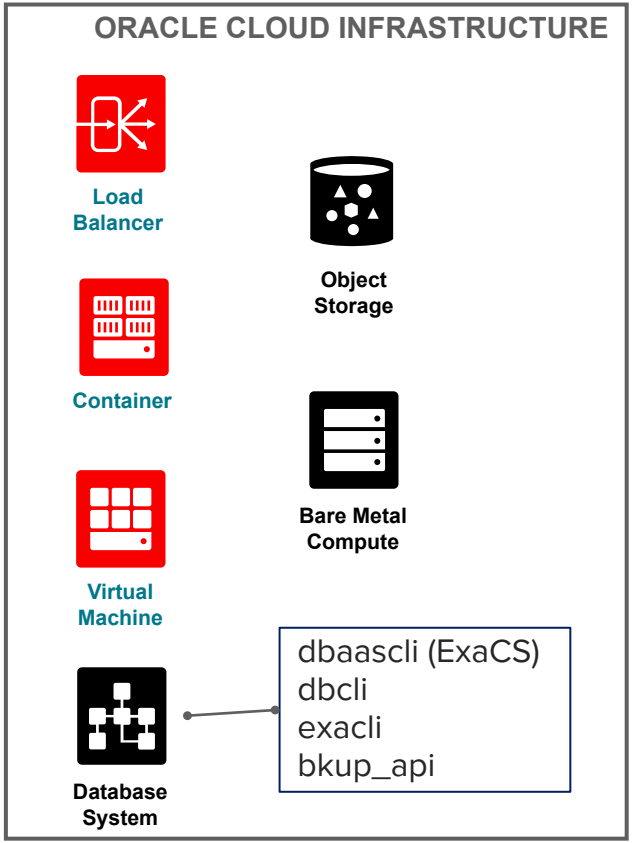
Download tools from OTN: http://www.oracle.com/technetwork/topics/cloud/downloads/index.html

Cloud CLI tools (classic and OCI)



Download tools from OTN: http://www.oracle.com/technetwork/topics/cloud/downloads/index.html

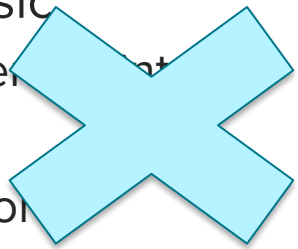
OCI CLI tools



Required account details for CLI

OCI Classic

- REST e
- user
- passwo
- identity domain



OCI

- tenancy OCID
- compartment OCID
- user OCID
- public key fingerprint
- region

OCI: https://docs.cloud.oracle.com/iaas/Content/API/Concepts/apisigningkey.htm

OCI-C: https://docs.oracle.com/en/cloud/iaas/compute-iaas-cloud/stopc/preparing-use-cli.html

OCI CLI installation in unix

```
$ bash -c "$(curl -L https://raw.githubusercontent.com/oracle/oci-cli/master/scripts/install/install.sh)"
```

- dependencies could require update of other packages (ex: openssl in in OEL 6.9)
- It should work in Windows having admin privileges

Follows an example of installing it under Ubuntu 16.04.3 LTS (WLS in Windows 10)

https://docs.cloud.oracle.com/iaas/Content/API/SDKDocs/cliinstall.htm

OCI CLI on Windows bash – installation 1/2

```
$ bash -c "$(curl -L https://raw.githubusercontent.com/oracle/oci- cli/master/scripts/install/install.sh)"
…
Downloading Oracle Cloud Infrastructure CLI install script …
######################################################################## 100.0%
Running install script.
python3 /tmp/oci_cli_install_tmp_QomF  < /dev/tty
-- Verifying Python version.
-- Python version 3.5.2 okay.
-- Verifying native dependencies.
-- Executing: 'dpkg -s libssl-dev libffi-dev python3-dev build-essential'
-- One or more of the following native dependencies are not currently installed and may be required.
"sudo apt-get install -y libssl-dev libffi-dev python3-dev build-essential"

===> Missing native dependencies. Continue and install the following dependencies: libssl-dev,
libffi-dev, python3-dev, build-essential? (Y/n):  Y
[sudo] password for ncalero:
Hit:1 http://archive.ubuntu.com/ubuntu xenial InRelease
…
```

OCI CLI on Windows bash – installation 2/2

```
…
Successfully installed PyYAML-3.13 arrow-0.10.0 asn1crypto-0.24.0 certifi-2019.3.9 cffi-1.12.2
click-6.7 configparser-3.5.0 cryptography-2.4.2 cx-Oracle-7.0.0 httpsig-cffi-15.0.0 idna-2.6
jmespath-0.9.3 oci-2.2.3 oci-cli-2.5.4 pyOpenSSL-18.0.0 pycparser-2.19 python-dateutil-2.7.3
pytz-2016.10 retrying-1.3.3 six-1.11.0 terminaltables-3.1.0

===> Modify profile to update your $PATH and enable shell/tab completion now? (Y/n): y

===> Enter a path to an rc file to update (leave blank to use '/home/ncalero/.bashrc'):
-- Backed up '/home/ncalero/.bashrc' to '/home/ncalero/.bashrc.backup'
-- Tab completion set up complete.
-- If tab completion is not activated, verify that '/home/ncalero/.bashrc' is sourced by your shell.
--
-- ** Run `exec -l $SHELL` to restart your shell. **
--
-- Installation successful.
-- Run the CLI with /home/ncalero/bin/oci --help
ncalero@H6RMYZ1:~$
ncalero@H6RMYZ1:~$
```

OCI CLI setup

1) Generate RSA public key

```
$ openssl genrsa -out OCI_api_key.pem 2048
$ chmod go-rwx OCI_api_key.pem
$ openssl rsa -pubout -in OCI_api_key.pem -out  OCI_api_key-pub.pem
```

2) Upload public key using OCI web console

It should be a local user in your tenant

It will report the fingerprint of the public key uploaded

```
Fingerprint: 8c:c7:11:87:b8:aa:d5:a7:5c:d4:9f:ff:48:48:5d:e3
```

3) Setup OCI CLI

Create $HOME/.oci/config file, manually or using the wizard

```
$ oci setup config
```

OCI CLI setup

```
ncalero@H6RMYZ1:~/ioug$ cat ~/.oci/config
[DEFAULT]
user=ocid1.user.oc1..aaaaaaaaxotqff7cznh76yknkyqau2l2ysw72l65l4tb3ojto7rhfiyhrscq
fingerprint=a4:c2:1d:57:91:ba:a8:da:41:26:fd:08:f8:65:46:56
key_file=/home/ncalero/ioug/OCI_api_key.pem
tenancy=ocid1.tenancy.oc1..aaaaaaaalqia3yg7kmfvj7fvmgt4j4kokziowmjkjmymyo62bwkw5hodl4pa
region=us-ashburn-1
```

Make sure to use the proper values:

```
ncalero@H6RMYZ1:~$ oci db system list --compartment-id
ocid1.compartment.oc1..aaaaaaaanesmhx3kiiuxohpynihp4owu7tidov5wfqehmoubc5je6s76hlsa
ServiceError:
{
    "code": "NotAuthenticated",
    "message": "The required information to complete authentication was not provided or was
incorrect.",
    "opc-request-id":
"C6A15A6E74B741C89704721C843E002A/00990538D26560F13A71509835BE3DF7/7CAC362441627E09812C81DCDD4491D5
",
    "status": 401
```

OCI CLI examples (1/2)

List Database Systems

```
ncalero@H6RMYZ1:~/ioug$ oci db system list -c $COMPID --output table --query 'data
[*].{ClusterName:"cluster-name", shape:shape}'
+-------------+----------------+
| ClusterName | shape          |
+-------------+----------------+
| None        | VM.Standard1.1 |
| None        | VM.Standard1.1 |
+-------------+----------------+
```

List Compute instances

```
ncalero@H6RMYZ1:~/ioug$ oci compute instance list -c $COMPID --output table --query 'data
[*].{AD:"availability-domain", Name:"display-name", shape:"shape", region:region}'
+---------------------+----------+--------+----------------+
| AD                  | Name     | region | shape          |
+---------------------+----------+--------+----------------+
| HDGG:US-ASHBURN-AD-1 | oem13c   | iad    | VM.DenseIO1.4  |
| HDGG:US-ASHBURN-AD-1 | ora122vm | iad    | VM.Standard2.1 |
+---------------------+----------+--------+----------------+
```

OCI CLI examples (2/2)

Filtering results using --query "data[expr] ...". http://jmespath.org/proposals/filter-expressions.html

```
ncalero@H6RMYZ1:~/ioug$ oci network subnet list -c $COMPID --vcn-id $VCNID --output table
--query "data [?contains(\"display-name\",'ashb')].{Name:\"display-name\",
AD:\"availability-domain\", CIDR:\"cidr-block\"}"
+----------------------+-------------+-------------------------------------+
| AD                   | CIDR        | Name                                |
+----------------------+-------------+-------------------------------------+
| HDGG:US-ASHBURN-AD-3 | 10.0.2.0/24 | Public Subnet HDGG:US-ASHBURN-AD-3 |
| HDGG:US-ASHBURN-AD-2 | 10.0.1.0/24 | Public Subnet HDGG:US-ASHBURN-AD-2 |
| HDGG:US-ASHBURN-AD-1 | 10.0.0.0/24 | Public Subnet HDGG:US-ASHBURN-AD-1 |
+----------------------+-------------+-------------------------------------+
```

Getting a JSON template to use for creation commands

```
ncalero@H6RMYZ1:~/ioug$ $ oci db system launch --generate-full-command-json-input
{
  "adminPassword": "string",
  "availabilityDomain": "string",
  "backupSubnetId": "string",
  "characterSet": "string",
…
```

Options to create OCI Instances and DBs

- Using web console
- Using API / CLI / SDK
  - No wizard, each component used must be already created
- Terraform provider - OCI
  Creates components if they don't exists
- Ansible modules
- CLI examples provided by Oracle:
  https://github.com/oracle/oci-cli/blob/master/scripts/examples

Example: creating a Database System with CLI

We need to use several components from OCI:

- Compartment
- Subnet
  - route table, security list, security rules

We need to specify:

- AD, faultDomains
- Type (Exadata, BM, VM)
- Shape
- licenseModel
- nodeCount
- version
- hostname

- dbname
- characterSet
- dbWorkload (OLTP/DW)
- dbVersion
- pdbName
- databaseEdition
- ssh keys

Example: creating a Database System with CLI

```
$ oci db system launch --from-json file://db-ioug-ad1-2n.json
```

It returns a long JSON including ID when it completes

```
Action completed. Waiting until the resource has entered state: PROVISIONING
{
  "data": {
    "availability-domain": "HDGG:US-ASHBURN-AD-1",
    ...
    "id": "ocid1.dbsystem.oc1.iad.abuwcljt2ws772ch6xlw2zadecuvlgven4hslmupiyx6y3add3na5kra6xgq",
    ...
```

Or returns error if it cannot be created

```
ncalero@H6RMYZ1:~/ioug$ oci db system launch --from-json file://db-ioug.json
ServiceError:
{
    "code": "LimitExceeded",
    "message": "You have reached your service limit of 8 Virtual Machine CPU Cores in this Availability Domain.
Please try launching the instance in a different Availability Domain or Region, or try using a different shape. If
you have reached all Service limits, please contact Oracle support to request a limit increase.",
    "opc-request-id": "9360A40AFADE4A4EB9A316AF0EB57015/0D5F255C1E15084A13B3D/ABD24B413BE4BA2B9414E9108CF02",
    "status": 400
}
```

Example: creating a Database System with CLI

```
$ oci db system launch --from-json file://db-ioug-ad1-2n.json
```

It returns a long JSON including ID when it completes

```
Action completed. Waiting until the resource has entered state: PROVISIONING
{
  "data": {
    "availability-domain": "HDGG:US-ASHBURN-AD-1",
    ...
    "id": "ocid1.dbsystem.oc1.iad.abuwcljt2ws772ch6xlw2zadecuvlgven4hslmupiyx6y3add3na5kra6xgq",
    ...
```

Or returns error if it cannot be created

```
ncalero@H6RMYZ1:~/ioug$ oci db system launch --from-json file://db-ioug.json
ServiceError:
{
    "code": "InvalidParameter",
    "message": "backupSubnetId : is not supported with current shape.",
    "opc-request-id": "2A1EC170EFDC4B8385A92FA8CED88A07/A6255B952C6932028509361640F6…",
    "status": 400
}
```

Example: creating a Database System with CLI

```
$ oci db system launch --from-json file://db-ioug-ad1-2n.json
```

It returns a long JSON including ID when it completes

```
Action completed. Waiting until the resource has entered state: PROVISIONING
{
  "data": {
    "availability-domain": "HDGG:US-ASHBURN-AD-1",
    ...
    "id": "ocid1.dbsystem.oc1.iad.abuwcljt2ws772ch6xlw2zadecuvlgven4hslmupiyx6y3add3na5kra6xgq",
    ...
```

Or returns error if it cannot be created

```
ncalero@H6PMYZ1: /ioug$ oci db system launch --from-json file://db-ioug.json
Servic
{
    "c
    "m
Please
you ha
    "o
    "s
}
```

```
ServiceError:
{
    "code": "InvalidParameter",
    "message": "Request is rejected as port 22 is not enabled in the security list for subnet
ocid1.subnet.oc1.iad.aaaaaaaahzydgowtl3tim7usrixkwhlbbzexowa",
    "opc-request-id": "7DE3DBB7637A4E9887B6913C40D3A227C33329E82/25664AE40C8E17A36F709A042656496B",
    "status": 400
}
```

Example: creating a Database System with CLI

Fixing previously reported problems:

1) `Request is rejected as port 22 is not enabled in the security list for subnet`

- The **default** security list for the subnet should have an ingress rule of type TCP for the subnet CIDR **for all ports** as source and target port 22. MOS note 2433870.1 helps.

```
$ oci network security-list get --security-list-id $SECLISTID
```

2) `InvalidParameter, backupSubnetId : is not supported with current shape`

- Double check parameters required for the DB we want to create.

  Note “oci db system get“ reports some attributes not needed to launch

3) `LimitExceeded, You have reached your service limit of 8 Virtual Machine CPU Cores in this Availability Domain`

- Validate the DBs already created in all your compartments

```
$ oci iam compartment list --output table --query 'data [*].{id:id}'
```

Example: creating a Database System with CLI

```
ncalero@H6RMYZ1:~/ioug$ cat db-ioug-ad1-2n.json
{
  "adminPassword": "xxx",
  "characterSet": "WE8MSWIN1252",
  "dbWorkload": "OLTP",
  "dbName": "CIOUG",
  "dbVersion": "12.1.0.2.181016",
  "ncharacterSet": "AL16UTF16",
  "pdbName": "PIOUG",
  "waitForState": "PROVISIONING",
  "waitIntervalSeconds": 30,
  "version": "12.2.0.1.181016",
  "availabilityDomain": "HDGG:US-ASHBURN-AD-1",
  "clusterName": "IOUG",
  "compartmentId": "ocid1.compartment.oc1..aaaaaaaazh5fxaasdfa",
  "cpuCoreCount": 8,
  "dataStoragePercentage": 80,
  "databaseEdition": "ENTERPRISE_EDITION_EXTREME_PERFORMANCE",
  "diskRedundancy": "HIGH",
```

```
"diskRedundancy": "HIGH",
  "displayName": "IOUG",
  "domain": "sub1234.vcn1234.oraclevcn.com",
  "faultDomains": [
        "FAULT-DOMAIN-1"
  ],
  "freeformTags": {},
  "hostname": "iougdb0",
  "initialDataStorageSizeInGb": 256,
  "licenseModel": "LICENSE_INCLUDED",
  "nodeCount": 2,
  "shape": "VM.Standard1.2",
  "sparseDiskgroup": null,
  "sshAuthorizedKeysFile": "./ssh-keys-ioug.txt",
  "subnetId": "ocid1.subnet.oc1.iad.aaaaaaaabcd",
  "timeZone": "UTC"
}
```

Example: creating a Database System with CLI

```
[opc@iougdb0 ~]$ sudo su -
[root@iougdb0 ~]# dbcli describe-component
System Version
---------------
18.3.3.0.0

Component                                Installed Version    Available Version
---------------------------------------- -------------------- --------------------
GI                                        12.2.0.1.181016       12.2.0.1.190115
DB                                        12.1.0.2.181016       12.1.0.2.190115

[root@iougdb0 ~]# dbcli describe-latestpatch

componentType   availableVersion
--------------- --------------------
gi              12.2.0.1.190115
gi              12.1.0.2.190115
gi              18.5.0.0.190115
db              11.2.0.4.190115
db              12.2.0.1.190115
db              12.1.0.2.190115
db              18.5.0.0.190115
```

OCI: creating a Compute Instance

We need to use several components from OCI:

- Compartment
- Virtual Cloud Network
  - Subnet, Internet GW, route table, security list, security rules
- OS Image
- Shape

We need to provide:

- ssh keys

Example: creating an OCI Instance with Terraform

- Download Terraform binary (using v0.11.3 today)
  - https://www.terraform.io/downloads.html
- OCI-provider – plugin in v3.0 automatically installed
  - https://github.com/oracle/terraform-provider-oci/releases
- Create .tf HCL Project (based on examples):
  - https://github.com/oracle/terraform-provider-oci/tree/master/docs/examples
- Configure your account Access (env-vars.ps1)
  - Tenancy and user OCIDs, fingerprint, private key, region
- Terraform Plan
- Terraform Apply
- Step by step guide: https://community.oracle.com/docs/DOC-1019936

```
ncalero@H6RMYZ1:/mnt/d/GitHub/terraform-OCI/$  ls
block.tf      compute.tf      env-vars.ps1  outputs.tf   userdata
datasources.tf  network.tf    provider.tf  remote-exec.tf  variables.tf

ncalero@H6RMYZ1:/mnt/d/GitHub/terraform-OCI$  cat /mnt/c/users/calero/AppData/Roaming/terraform.rc
providers { oci = "d:\\soft\\terraform-provider-oci_v2.1.0.exe"  }
```

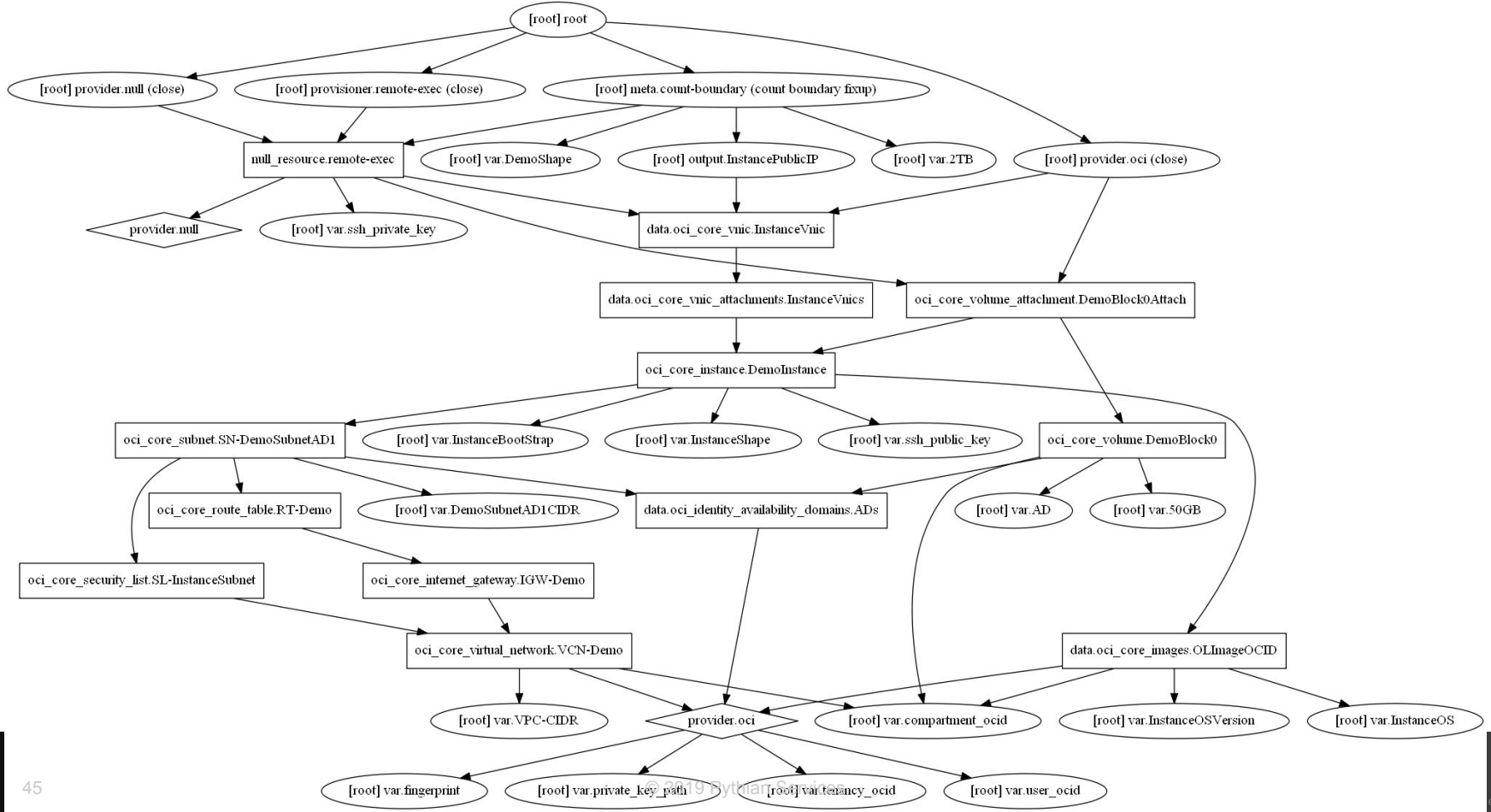
```
PS D:\GitHub\terraform-OCI>  cat ./env-vars.ps1
$env:TF_VAR_tenancy_ocid="ocid1.tenancy.oc1..aaaaaaaaw5hodl4pajmymyo62bwkj4kokziowmjkvj7fvmgt4lqia3yg7km
"
$env:TF_VAR_user_ocid="ocid1.user.oc1..aaaaaaaaiyhrscq7rhfiyhrscql4tb3ojto2ysw72l656yknkyqau2lxotqf"
$env:TF_VAR_compartment_ocid="ocid1.compartment.oc1..
aaaaaaaakdevdha7q7yjdheridcy63sgv2syefv7x46u67jk4hj"
$env:TF_VAR_region="us-ashburn-1"

$env:TF_VAR_fingerprint="db:a3:b8:59:ce:05:ff:78:c0:fb:aa:ca:d7:09:77:ad"
$env:TF_VAR_private_key_path="C:\Users\calero\.oci\oci_api_key.pem"

### Public/private keys used on the instances
$env:TF_VAR_ssh_public_key = Get-Content C:\Users\calero\.ssh\id_rsa.pub -Raw
$env:TF_VAR_ssh_private_key = Get-Content C:\Users\calero\.ssh\id_rsa -Raw
```

```
ncalero@H6RMYZ1:/mnt/d/GitHub/terraform-OCI/demo$  cat provider.tf
provider "oci" {
  tenancy_ocid     = "${var.tenancy_ocid}"
  user_ocid        = "${var.user_ocid}"
  fingerprint      = "${var.fingerprint}"
  private_key_path = "${var.private_key_path}"
  region           = "${var.region}"
}

ncalero@H6RMYZ1:/mnt/d/GitHub/terraform-OCI$  cat compute.tf
resource "oci_core_instance" "DemoInstance" {
  availability_domain =
"${lookup(data.oci_identity_availability_domains.ADs.availability_domains[0],"name")}"
  compartment_id      = "${var.compartment_ocid}"
  display_name        = "Demo-Instance"
  image               = "${lookup(data.oci_core_images.OLImageOCID.images[0], "id")}"
  shape               = "${var.InstanceShape}"
  subnet_id           = "${oci_core_subnet.SN-DemoSubnetAD1.id}"

  metadata {
    ssh_authorized_keys = "${var.ssh_public_key}"
    user_data           = "${base64encode(file(var.InstanceBootStrap))}"
  }
}
```

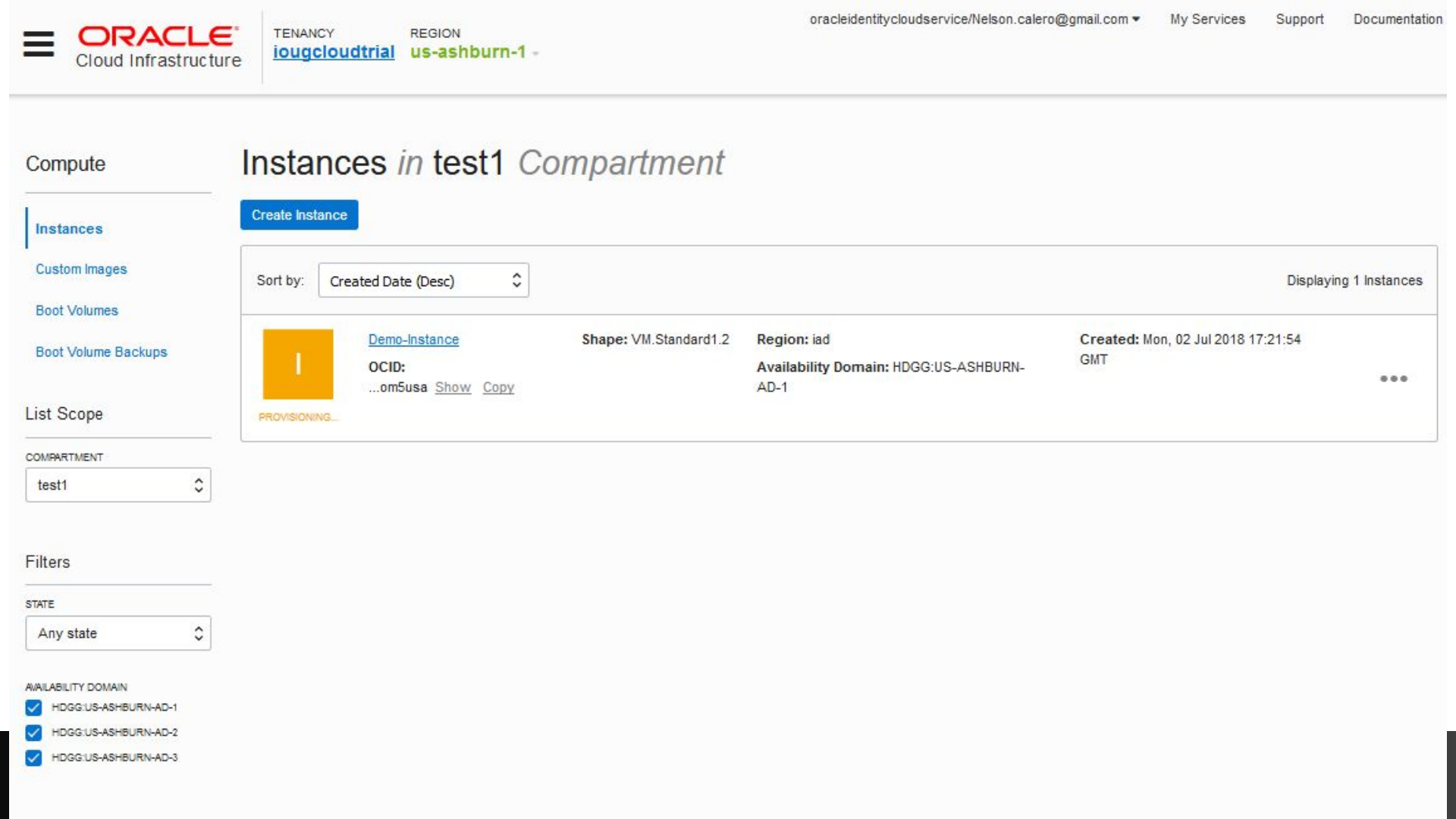
```
PS D:\GitHub\terraform-OCI\demo> d:\soft\terraform init

Initializing provider plugins...
- Checking for available provider plugins on https://releases.hashicorp.com...
- Downloading plugin for provider "null" (1.0.0)...

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, it is recommended to add version = "..." constraints to the
corresponding provider blocks in configuration, with the constraint strings
suggested below.

* provider.null: version = "~> 1.0"

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS D:\GitHub\terraform-OCI\demo>
```

```
PS D:\GitHub\terraform-OCI\demo> d:\soft\terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.

data.oci_identity_availability_domains.ADs: Refreshing state...
data.oci_core_images.OLImageOCID: Refreshing state...

------------------------------------------------------------------------

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create
  ~ update in-place
 <= read (data resources)

Terraform will perform the following actions:

  ~ data.oci_core_vnic.InstanceVnic
      id:                                   "" => <computed>
      availability_domain:                  "" => <computed>
      compartment_id:                       "" => <computed>
      display_name:                         "" => <computed>
      hostname_label:                       "" => <computed>
      is_primary:                           "" => <computed>
      mac_address:                          "" => <computed>
      private_ip_address:                   "" => <computed>
      public_ip_address:                    "" => <computed>
      skip_source_dest_check:               "" => <computed>
      state:                                "" => <computed>
      subnet_id:                            "" => <computed>
      time_created:                         "" => <computed>
      vnic_id:                              "" => "${lookup(data.oci_core_vnic_attachments.InstanceVnics.vnic_
attachments[0],\"vnic_id\")}"

 <= data.oci_core_vnic_attachments.InstanceVnics
      id:                                   <computed>
      availability_domain:                  "HDGG:US-ASHBURN-AD-1"
```

```
PS D:\GitHub\terraform-OCI>  d:\soft\terraform graph | dot -Tpng -o demo-graph.png
```



```
PS D:\GitHub\terraform-OCI\demo>  d:\soft\terraform apply
Terraform will perform the following actions:
…
Plan: 9 to add, 1 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value:  yes

oci_core_volume.DemoBlock0: Creating...
…
oci_core_instance.DemoInstance:  Creation complete after 1m9s (ID:
ocid1.instance.oc1.iad.abuwcljtxf33kdni...toxpdtljex6zq5gl2vsccmgffvlfcsswom5usa)
…
Apply complete! Resources: 9 added, 0 changed, 0 destroyed.

Outputs:

InstancePublicIP = [
    129.213.48.178
]
```


ORACLE
Cloud Infrastructure
TENANCY
iougcloudtrial
REGION
us-ashburn-1
oracleidentitycloudservice/Nelson.calero@gmail.com
My Services
Support
Documentation
Compute
Instances
Custom Images
Boot Volumes
Boot Volume Backups
List Scope
COMPARTMENT
test1
Filters
STATE
Any state
AVAILABILITY DOMAIN
HDGG:US-ASHBURN-AD-1
HDGG:US-ASHBURN-AD-2
HDGG:US-ASHBURN-AD-3
Instances in test1 Compartment
Create Instance
Sort by:
Created Date (Desc)
Displaying 1 Instances
Demo-Instance
OCID:
...om5usa Show Copy
PROVISIONING...
Shape: VM.Standard1.2
Region: iad
Availability Domain: HDGG:US-ASHBURN-AD-1
Created: Mon, 02 Jul 2018 17:21:54 GMT

Backup considerations:

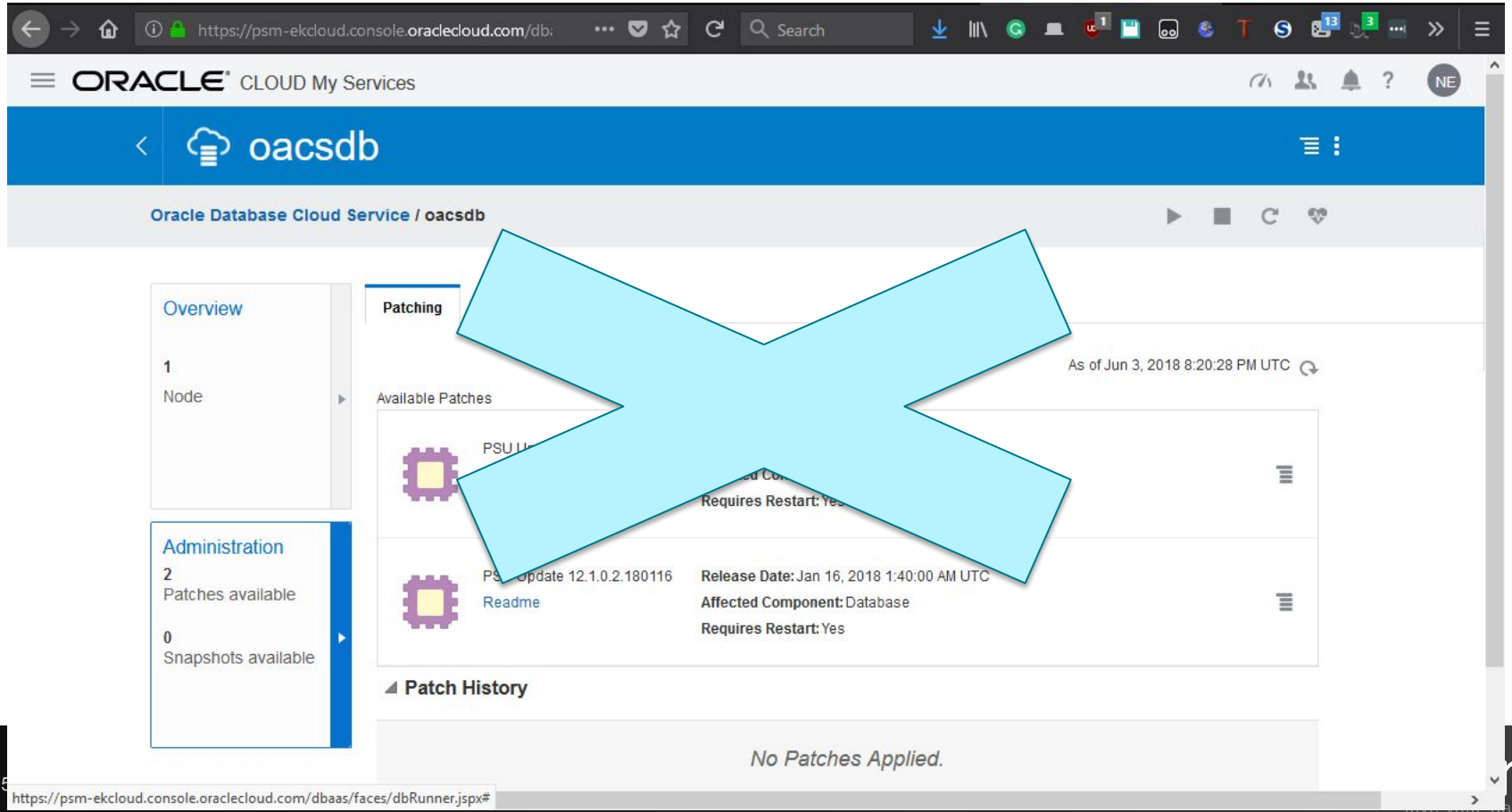
- "Backups that were configured using the Console may become unusable if you make changes using these commands. For backups configured using the Console, use these commands with support guidance only."

```
dbcli getstatus-backup
dbcli getstatus-backup -in CTEST2 -t Archivelog

dbcli create-rmanbackupreport -w summary -rn test --dbname CTEST2
dbcli list-jobs
dbcli describe-job -i 4340156e-9e72-48ce-9e6a-f1d34b343603

ls -lrt /opt/oracle/dcs/log/test2/rman/bckup/CTEST2_* | tail
```

Automation approach

- Not the way you are used to as DBA: PaaS have built-in automation. DBaaS example:
  - Backup
  - Recovery
  - Patch
  - Failover
  - Standby creation (VM DB Systems only, same VCN for prim/stb, web console)
- Extra tools provided by Oracle
  - OCI Terraform provider
  - Ansible modules: https://github.com/oracle/oci-ansible-modules
  - Chef Knife Plugin
  - EBS Cloud Admin Tool

Built-in automation: DBCS patching (classic)



Built-in automation: DBCS patching

- Different in ExaCs and VM/BM DB Systems
- Tools automatically deployed to server when created
  - manually update to new version before patching
- PSU can be specified when creating the DB
- Web console shows System (GI) and DB patches in different pages

- CLI Example: apply the DBBP bundle patch to a database home: https://docs.cloud.oracle.com/iaas/Content/Database/References/dbacli.htm#updatedbhome

DBCS versions available (August vs April 2019)

```
calero@H6RMYZ1:~$ oci db version list -c $COMPID --db-system-shape "VM.Standard2.4" --all --output table
+-----------------------------+--------------+-----------------+
| is-latest-for-major-version | supports-pdb | version         |
+-----------------------------+--------------+-----------------+
| True                        | False        | 11.2.0.4        |
| False                       | False        | 11.2.0.4.181016 |
| False                       | False        |11.2.0.4.190115  |
| False                       | False        |11.2.0.4.190416  |
| True                        | True         | 12.1.0.2        |
| False                       | True         | 12.1.0.2.181016 |
| False                       | True         |12.1.0.2.190115  |
| False                       | True         |12.1.0.2.190416  |
| True                        | True         | 12.2.0.1        |
| False                       | True         | 12.2.0.1.181016 |
| False                       | True         |12.2.0.1.190115  |
| False                       | True         |12.2.0.1.190416  |
| True                        | True         | 18.0.0.0        |
| False                       | True         | 18.4.0.0        |
| False                       | True         |18.5.0.0         |
| False                       | True         |18.6.0.0         |
| True                        | True         |19.0.0.0         |
| False                       | True         |19.3.0.0.190416  |
+-----------------------------+--------------+-----------------+
```

```
+-----------------+
| version         |
+-----------------+
| 11.2.0.4        |
| 11.2.0.4.180417 |
| 11.2.0.4.180717 |
| 11.2.0.4.181016 |
| 12.1.0.2        |
| 12.1.0.2.180417 |
| 12.1.0.2.180717 |
| 12.1.0.2.181016 |
| 12.2.0.1        |
| 12.2.0.1.180417 |
| 12.2.0.1.180717 |
| 12.2.0.1.181016 |
| 18.0.0.0        |
| 18.2.0.0        |
| 18.3.0.0        |
| 18.4.0.0        |
+-----------------+
```

DBCS patching from CLI – VM and BM

```
# patching
cliadm update-dbcli
dbcli list-jobs
dbcli describe-component
dbcli describe-latestpatch

# patch GI
dbcli update-server
dbcli describe-job -i xxx

# patch DB
dbcli list-dbhomes
dbcli update-dbhome -i xxx2 (from list-dbhomes)
dbcli describe-job -i xxx3 (from update)

# one-off
Use opatch as usual
```


https://docs.cloud.oracle.com/iaas/Content/Database/Tasks/patchingDB.htm

DBCS patching from CLI - Exadata CS

Note exadbcpatchmulti utility has been incorporated into dbaascli

```
# list available patches
dbaascli patch db list --oh exacs-node1:/u02/app/oracle/product/18.0.0.0/dbhome_1

# Validate patch prereqs
dbaascli patch db prereq --patchid 12345678 --dbnames MYDB

# Apply a patch
# NOTE: you run the SQL part only on the last node
dbaascli patch db apply --patchid 12345678 --dbnames MYDB --run_datasql 1

# rollback a patch
dbaascli patch db switchback --patchid 12345678 --dbnames MYDB --run_datasql 1 \
  --instance1 exacs-node1:/u02/app/oracle/product/18.0.0.0/dbhome_1
```


https://docs.cloud.oracle.com/iaas/Content/Database/Tasks/exapatching.htm

Built-in automation: OCI DB backups

- BM and VM:
  - Managed by Oracle: OCI to Object storage
  - unmanaged: RMAN or *dbcli* to local storage (BM only)

  https://docs.cloud.oracle.com/iaas/Content/Database/Tasks/backingup.htm

- Exadata
  - Managed by Oracle: Object storage and local FRA disk

  https://docs.cloud.oracle.com/iaas/Content/Database/Tasks/exabackingup.htm
  - unmanaged: RMAN or *dbcli* to local storage

- logs: /opt/oracle/dcs/log/<nodename>/rman/bkup/<db_unique_name>

What's next?

- Explore the APIs/CLI for all Oracle cloud services
- Automate infrastructure changes with CI/CD pipelines
- Take advantage of Oracle Management Cloud
  - more than just monitoring: https://cloud.oracle.com/management
- Explore hybrid cloud deployments to expand your on-premises capacity for specific use cases

THANK YOU

Questions?

calero@pythian.com

@ncalerouy

http://www.linkedin.com/in/ncalero

Pythian
love your data®

More terraform examples

Creating a kubernetes cluster

https://github.com/oracle/terraform-kubernetes-installer

Same account configuration as previous OCI instance example, but also:

- Policy at root compartment
  - allow service OKE to manage all-resources in tenancy
- A group to which a policy grants appropriate permissions
- command line tool kubectl

https://docs.cloud.oracle.com/iaas/Content/ContEng/Concepts/contengprerequisites.htm

NOTE: Only cluster setup – need an application to consume it

References - documentation

Oracle Cloud tools: http://www.oracle.com/technetwork/topics/cloud/downloads/index.html

OCI CLI examples: https://github.com/oracle/oci-cli/tree/master/scripts/examples

Oracle Cloud API: https://docs.cloud.oracle.com/iaas/api/#/

Oracle Cloud Compute shapes:
https://docs.cloud.oracle.com/iaas/Content/Compute/References/computeshapes.htm
https://cloud.oracle.com/en_US/compute-classic/pricing

EBS cloud admin tool
https://blogs.oracle.com/ebsandoraclecloud/june-2018-ebs-cloud-admin-tool-updates-available

Using Terraform with OCI: https://community.oracle.com/docs/DOC-1019936

OCI Examples used in this presentation:
https://github.com/ncalero-uy/conferences/blob/master/oci-cli-demo-oow19.txt