**GCP – HOL -Session 2**

* Create and manage [Google Compute Engine](https://cloud.google.com/shell/docs/examples#managing_compute_engine_resources) instances
* Create and access [Google Cloud SQL](https://cloud.google.com/shell/docs/examples#managing_cloud_sql_data) databases
* Manage [Google Cloud Storage](https://cloud.google.com/shell/docs/examples#managing_cloud_storage_data) data
* Interact with hosted or remote Git repositories, including [Google Cloud Source Repositories](https://cloud.google.com/tools/cloud-repositories/docs/)
* Build and deploy [Google App Engine](https://cloud.google.com/shell/docs/examples#running_and_deploying_a_python_app_engine_application) applications

The following examples show how to perform some common tasks using Cloud Shell.

Managing Compute Engine resources

You can manage [Compute Engine](https://cloud.google.com/compute/docs/) resources like virtual machine instances using the gcloud command-line tool in a Cloud Shell session.

For example:

1. [Launch Cloud Shell](https://cloud.google.com/shell/docs/quickstart) in the [GCP Console](https://console.cloud.google.com/).
2. Enter the following at the Cloud Shell command prompt:

# List all compute instances in the project  
gcloud compute instances list  
  
# Grep the serial console output from all instances in the project  
# for a specific pattern in the output.  
gcloud compute instances list | \  
awk 'NR > 1 { print "--zone " $2 " " $1 }' | \  
xargs -L1 gcloud compute instances get-serial-port-output | \  
grep BREAK-IN

Managing Cloud SQL data

You can manage [Cloud SQL data](https://cloud.google.com/sql/docs/) using gcloud and the database administration client in a Cloud Shell session.

For example, for a Cloud SQL for MySQL instance, enter the following at the Cloud Shell command prompt:

# Set the project of interest  
gcloud config set project hello-world-314  
  
# Create a Cloud SQL instance  
gcloud sql instances create my-instance  
  
# Assign the instance an IPv4 address, because Compute Engine  
# does not yet support IPv6 addresses.  
gcloud sql instances patch --assign-ip my-instance  
  
# Set the root user password for the instance.  Replace the [PASSWORD]  
# placeholder with the actual password you want to use.  
gcloud sql users set-password root --host % --instance my-instance --password [PASSWORD]  
  
# Connect to the instance.  
gcloud sql connect my-instance --user=root

The gcloud command and mysql client return the following:

Whitelisting your IP for incoming connection for 5 minutes...done.  
Connecting to database with SQL user [root]. Enter password:  
  
Welcome to the MySQL monitor.  Commands end with ; or \g.  
Your MySQL connection id is 101436  
Server version: 5.7.14-google-log (Google)  
Copyright (c) 2000, 2017, Oracle and/or its affiliates. All rights reserved.  
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
mysql>

Managing Cloud Storage data

You can also use the gsutil tool in Cloud Shell to manage [Cloud Storage](https://cloud.google.com/storage/docs/) resources. This includes creating and deleting buckets and objects, copying and moving storage data, and managing bucket and object ACLs. You can also use gsutil to transfer data in and out of your Cloud Shell instance.

For example, enter the following at the Cloud Shell command prompt:

# Create a Cloud Storage bucket

gsutil mb gs://my-bucket-555

# Upload some data to the Cloud Storage bucket you created

gsutil cp test.dat gs://my-bucket-555

Managing Container Engine clusters

You can also create and manage [Container Engine](https://cloud.google.com/container-engine/docs/) clusters from the Cloud Shell command line.

For example, enter the following at the Cloud Shell command prompt:

gcloud config set compute/zone us-central1-a  
gcloud container clusters create my-cluster

After the cluster is created, you can initialize the kubectl tool and use it to manage the cluster:

gcloud container clusters get-credentials my-cluster  
kubectl get nodes  
...  
kubectl cluster-info  
...

Running and deploying a Python App Engine application

Do the following in a Cloud Shell session:

1. Create a new directory for the application:

mkdir helloworld && cd helloworld

1. Create app.yaml and helloworld.py with content from the Google App Engine [Python “Hello World” example](https://cloud.google.com/appengine/docs/python/gettingstartedpython27/helloworld).
2. Start a deployment instance of the application server:

dev\_appserver.py ./

1. Click on the Web Preview icon Server icon in the Cloud Shell toolbar and choose port 8080. A tab in your browser opens and connects to the server you just started.
2. Deploy your Hello World server to the production App Engine environment:

gcloud app deploy ./app.yaml

After the application is deployed, you can visit it by opening the URL http://<project-id>.appspot.com in your web browser.

Running and deploying a Java App Engine application

Do the following:

1. In a Cloud Shell session, follow the steps from the Google App Engine [Java tutorial](https://cloud.google.com/appengine/docs/java/gettingstarted/creating).
2. Start a deployment instance of the application server:

mvn appengine:devserver -Dappengine.address=0.0.0.0

This makes sure that the server is accessible through the Cloud Shell HTTP proxy.

1. Click on the Web Preview icon Server icon in the Cloud Shell toolbar and choose port 8080. A tab in your browser opens and connects to the server you just started.