# Challenges Faced

* Mage did not identify Kaggle Install
  + I had installed CONDA and had activated a CONDA Virtual Environment.
  + In the Mage Data Loader Pipeline, I was not able to import CONDA at all.
  + Realized that the issue was because Mage was not able to identify the CONDA install from the run time.
  + When I reinstalled the PIP Requirements in the non-Virtual Environment Path, Mage Data Loader worked.
* Terraform, File Provisioner and Remote Exec. Terraform Provider passing the Service Account JSON as user input
  + I had to spend invariably long amount of time when I used Terraform to copy the file to the provisioned VM and run few commands from the GCP Shell.
  + Several examples online do not talk about the connection required in each provisioner.
  + Any problems in the ‘inline’ array of commands in Remote-Exec do not throw clear error but the terraform apply fails.
  + Terraform google provider requires credentials. To ensure the sensitivity of the credentials file, I had to spend lot of time to input that as a file.

provider "google" {

project = var.projectId

credentials = file("${var.gcpkey}")

#region = var.region

zone = var.zone

}

* Error 403: Insufficient regional quota to satisfy request: resource 'SSD\_TOTAL\_GB' exceeded.
  + Changing the type of boot-disk from PD-Balanced to PD-Standard helped solve the problem.
* When trying to execute the Mage Docker on a GCP VM, I built the image of mage\_spark. Unfortunately the docker build was struck for a long time. Mage\_Spark Dockerfile from Mage site recommended

RUN ${PIP} install --no-cache-dir --upgrade pip

COPY requirements.txt /tmp/requirements.txt

RUN python3 -m pip install -r /tmp/requirements.txt

# FAQs

## Docker not installable on Ubuntu

On few versions of Ubuntu, snap command can be used to install Docker.

sudo snap install docker

## Terraform google provider requires credentials. To ensure the sensitivity of the credentials file, I had to spend lot of time to input that as a file.

provider "google" {

project = var.projectId

credentials = file("${var.gcpkey}")

#region = var.region

zone = var.zone

}

## Error 403: Insufficient regional quota to satisfy request: resource 'SSD\_TOTAL\_GB' exceeded.

Changing the type of boot-disk from PD-Balanced to PD-Standard helped solve the problem.

## When trying to execute the Mage Docker on a GCP VM, I built the image of mage\_spark. Unfortunately the docker build was struck for a long time. Mage\_Spark Dockerfile from Mage site recommended

RUN ${PIP} install --no-cache-dir --upgrade pip

COPY requirements.txt /tmp/requirements.txt

RUN python3 -m pip install -r /tmp/requirements.txt

## Basic Commands

### Docker Commands

                     # Create a Docker Image from a base image

Docker run -it ubuntu bash

#List docker images

Docker images list

#List  Running containers

Docker ps -a

#List with full container ids

Docker ps -a --no-trunc

#Add onto existing image to create new image

Docker commit -a <User\_Name> -m "Message" container\_id New\_Image\_Name

# Create a Docker Image with an entrypoint from a base image

Docker run -it --entry\_point=bash python:3.11

#Attach to a stopped container

Docker start -ai <Container\_Name>

#Attach to a running container

docker exec -it <Container\_ID> bash

#copying from host to container

Docker cp <SRC\_PATH/file> <containerid>:<dest\_path>

#copying from container to host

Docker cp <containerid>:<Srct\_path> <Dest Path on host/file>

#Create an image from a docker file

Docker build -t <Image\_Name> <Location of Dockerfile>

#DockerFile Options and best practices

<https://devopscube.com/build-docker-image/>

#Docker delete all images forcefully

docker rmi -f $(docker images -aq)

#Docker delete all containers forcefully

docker rm -f $(docker ps -qa)

#docker compose creation

<https://www.composerize.com/>

### GCP Commands

1.     Create SSH Keys

2.     Added to the Settings of Compute Engine VM Instance

3.     SSH-ed into the VM Instance with a config similar to following

Host my-website.com

HostName my-website.com

User my-user

IdentityFile ~/.ssh/id\_rsa

4.     Installed Anaconda by installing the sh file through bash <Anaconda.sh>

5.     Install Docker after

a.     Sudo apt-get update

b.     Sudo apt-get docker

6.     To run Docker without SUDO permissions

a.    <https://github.com/sindresorhus/guides/blob/main/docker-without-sudo.md>

7.     Google cloud remote copy

a.     gcloud compute scp LOCAL\_FILE\_PATHVM\_NAME:REMOTE\_DIR

Install GCP Cloud SDK on Docker Machine

<https://stackoverflow.com/questions/23247943/trouble-installing-google-cloud-sdk-in-ubuntu>

sudo apt-get install apt-transport-https ca-certificates gnupg && echo "deb [signed-by=/usr/share/keyrings/cloud.google.gpg]<https://packages.cloud.google.com/apt> cloud-sdk main"| sudo tee -a /etc/apt/sources.list.d/google-cloud-sdk.list&& curl<https://packages.cloud.google.com/apt/doc/apt-key.gpg> | sudo apt-key --keyring /usr/share/keyrings/cloud.google.gpg add - && sudo apt-get update && sudo apt-get install google-cloud-sdk && sudo apt-get install google-cloud-sdk-app-engine-java && sudo apt-get install google-cloud-sdk-app-engine-python && gcloud init

### Anaconda Commands

### 

                     #Activate environment

Conda Activate <environment\_name>

#DeActivate environment

Conda DeActivate <environment\_name>

#Start iterm without conda environment

conda config --set auto\_activate\_base false

# Using Conda forge as default (Community driven packaging recipes and solutions)

<https://conda-forge.org/docs/user/introduction.html>

conda --version

conda update conda

conda config --add channels conda-forge

conda config --set channel\_priority strict

#Using Libmamba as Solver

conda install pgcli  --solver=libmamba

### 

### Linux/MAC Commands

**Starting and Stopping Services on Linux**

●  sudo systemctl start postgresql

●  sudo systemctl stop postgresql

**Starting and Stopping Services on MAC**

●      launchctl start postgresql

●      launchctl stop postgresql

**Identifying processes listening to a Port across MAC/Linux**

sudo lsof -i -P -n | grep LISTEN

$ sudo netstat -tulpn | grep LISTEN

$ sudo ss -tulpn | grep LISTEN

$ sudo lsof -i:22 ## see a specific port such as 22 ##

$ sudo nmap -sTU -O IP-address-Here

**Installing a package on Debian**

 sudo apt install <packagename>

**Listing all package on Debian**

Dpkg -l | grep <packagename>

**UnInstalling a package on Debian**

Sudo apt remove <packagename>

Sudo apt autoclean  && sudo apt autoremove

**List all Processes on Debian/Ubuntu**

Ps -aux

apt-get update && apt-get install procps

apt-get install iproute2 for ss -tulpn

**#Postgres Install**

sudo sh -c 'echo "deb<https://apt.postgresql.org/pub/repos/apt> $(lsb\_release -cs)-pgdg main" > /etc/apt/sources.list.d/pgdg.list'

wget --quiet -O -<https://www.postgresql.org/media/keys/ACCC4CF8.asc> | sudo apt-key add -

sudo apt-get update

sudo apt-get -y install postgresql

**#Changing Postgresql port to 5432**

 - sudo service postgresql stop - sed -e 's/^port.\*/port = 5432/' /etc/postgresql/10/main/postgresql.conf > postgresql.conf

- sudo chown postgres postgresql.conf

- sudo mv postgresql.conf /etc/postgresql/10/main

- sudo systemctl restart postgresql