Setup spark1

In [ ]:

```
def setupSpark():
  # Spark needs to run with Java 8 ...
  !pip install -q findspark
  !apt-get install openjdk-8-jdk-headless > /dev/null
  !echo 2 | update-alternatives --config java > /dev/null
  !java -version
  import os, findspark
  os.environ['JAVA_HOME'] = '/usr/lib/jvm/java-8-openjdk-amd64'
  # !echo JAVA_HOME=$JAVA_HOME
  !pip install -q pyspark
  findspark.init(spark_home='/usr/local/lib/python3.7/dist-packages/pyspark')
  !pyspark --version

setupSpark()

from pyspark import SparkContext
from pyspark.sql import SparkSession

spark = SparkSession\
        .builder\
        .master('local[*]')\
        .getOrCreate()
sc = spark.sparkContext
```

Connect to Google Cloud

You need to set the `PROJECT_ID` variable.

In [ ]:

```
PROJECT_ID = 'cloud-computing-project-309514'
BUCKET_URI = 'gs://bdcc_open_images_dataset'
from google.colab import auth
auth.authenticate_user()
!gcloud config set project {PROJECT_ID}
```

Get necessary data

This will fetch files that contain the same data as in the BigQuery tables we use for the project.

In [ ]:

```
!gsutil cp {BUCKET_URI}/data/classes.csv .
!gsutil cp {BUCKET_URI}/data/image-labels.csv .
!head classes.csv
!head image-labels.csv
```

Initialize data frames

In [ ]:

```
classes = spark.read.csv('classes.csv',inferSchema=True,header=True)
classes.cache()
classes.createOrReplaceTempView('classes')
classes.printSchema()
classes.show()

image_labels = spark.read.csv('image-labels.csv',inferSchema=True,header=True)
image_labels.cache()
image_labels.createOrReplaceTempView('image_labels')
image_labels.printSchema()
image_labels.show()
```

Define the classes for your model.

Change `CLASSES` to the image classes you want.

See the project description for instructions.

In [ ]:

```
CLASSES =[
          ('Squirrel',),
          ('Flag',),
          ('Coin',),
          ('Ball',),
          ('Falcon',),
          ('Glove',),
          ('Goat',),
          ('Taco',),
          ('Computer monitor',),
          ('Knife',)
]
```

In [ ]:

```
class_labels = spark.createDataFrame(data=CLASSES,schema=['Description'])
class_labels.cache()
class_labels.createOrReplaceTempView('class_labels')
class_labels.printSchema()
class_labels.show()
```

Define the data set you want using Spark

Now it's up to you.

In [ ]:

```
import pandas as pd
dataToCSV = []
listCSVToImagems = []

for classAux in CLASSES:
  query = spark.sql('''
   SELECT * FROM image_labels
   JOIN classes USING(Label)
   WHERE Description = '{0}'
   LIMIT 100
  '''.format(classAux[0]))
  numbersOfMLControl = 0
  for row in query.rdd.collect():

    if numbersOfMLControl<80:
      typetoCSV = "TRAIN"
      uritoCSV = "gs://projectbucket10/images/" + row.ImageId + ".jpg"
      classToCSV = row.Description
    if numbersOfMLControl>=80 and numbersOfMLControl<90:
      typetoCSV = "VALIDATION"
      uritoCSV = "gs://projectbucket10/images/" + row.ImageId + ".jpg"
      classToCSV = row.Description
    if numbersOfMLControl>=90:
      typetoCSV = "TEST"
      uritoCSV = "gs://projectbucket10/images/" + row.ImageId + ".jpg"
      classToCSV = row.Description

    numbersOfMLControl = numbersOfMLControl+1

    csvLine = []
    csvLine.append(typetoCSV)
    csvLine.append(uritoCSV)
    csvLine.append(classToCSV)
    dataToCSV.append(csvLine)

    uritoList = "bdcc_open_images_dataset/images/" + row.ImageId + ".jpg"
    listCSVToImagems.append(uritoList)

  numbersOfMLControl = 0

# Create the pandas DataFrame
df = pd.DataFrame(dataToCSV)

# print dataframe.
# df

csv = df.to_csv(index=False, header=False)
csv

file = open('csvClasses.csv', mode='w')
file.write(csv)
file.close()
```

Put the data in a convenient bucket

Now upload the CSV file describing the file and **only** the necessary images to the bucket you'll use with AutoML.

Note: the bucket must be created using a **Regional** location setting. Choose **us-central1** for example.

In [ ]:

```
MY_AUTOML_BUCKET='projectbucket10'

!gsutil -m cp -R /content/csvClasses.csv gs://{MY_AUTOML_BUCKET}
```

In [ ]:

```
MY_AUTOML_IMAGES = 'projectbucket10/images/'

for img in listCSVToImagems:
  !gsutil -m cp -R gs://{img} gs://{MY_AUTOML_IMAGES}
```