```
# ML1020 - Assignment 2 (DAG for wordcount)

"""Example Airflow DAG that creates a Cloud Dataproc cluster, runs the Hadoop
wordcount example, and deletes the cluster.

This DAG relies on three Airflow variables
https://airflow.apache.org/docs/apache-airflow/stable/concepts/variables.html
* gcp_project - Google Cloud Project to use for the Cloud Dataproc cluster.
* gce_zone - Google Compute Engine zone where Cloud Dataproc cluster should be
  created.
* gcs_bucket - Google Cloud Storage bucket to use for result of Hadoop job.
  See https://cloud.google.com/storage/docs/creating-buckets for creating a
  bucket.
"""

import datetime
import os

from airflow import models
from airflow.providers.google.cloud.operators import dataproc
from airflow.utils import trigger_rule

# Output file for Cloud Dataproc job.
# If you are running Airflow in more than one time zone
# see https://airflow.apache.org/docs/apache-airflow/stable/timezone.html
# for best practices
output_file = os.path.join(
    models.Variable.get('gcs_bucket'), 'wordcount',
    datetime.datetime.now().strftime('%Y%m%d-%H%M%S')) + os.sep
```

```
# Path to Hadoop wordcount example available on every Dataproc cluster.
WORDCOUNT_JAR = (
    'file:///usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar'
)
# Arguments to pass to Cloud Dataproc job.
#input_file = 'gs://pub/shakespeare/rose.txt'
input_file = 'gs://ml1020-bucket/asn1/*.txt'
wordcount_args = ['wordcount', input_file, output_file]

HADOOP_JOB = {
    "reference": {"project_id": models.Variable.get('gcp_project')},
    "placement": {"cluster_name": 'composer-hadoop-tutorial-cluster-{{ ds_nodash }}'},
    "hadoop_job": {
        "main_jar_file_uri": WORDCOUNT_JAR,
        "args": wordcount_args,
    },
}

CLUSTER_CONFIG = {
    "master_config": {
        "num_instances": 1,
        "machine_type_uri": "n1-standard-2"
    },
    "worker_config": {
        "num_instances": 2,
        "machine_type_uri": "n1-standard-2"
    },
}

yesterday = datetime.datetime.combine(
    datetime.datetime.today() - datetime.timedelta(1),
    datetime.datetime.min.time())

default_dag_args = {
    # Setting start date as yesterday starts the DAG immediately when it is
    # detected in the Cloud Storage bucket.
    'start_date': yesterday,
    # To email on failure or retry set 'email' arg to your email and enable
    # emailing here.
    'email_on_failure': False,
    'email_on_retry': False,
    # If a task fails, retry it once after waiting at least 5 minutes
    'retries': 1,
    'retry_delay': datetime.timedelta(minutes=5),
    'project_id': models.Variable.get('gcp_project'),
    'location': models.Variable.get('gce_region'),

}


with models.DAG(
        'wordcount_assignment',
        # Continue to run DAG once per day
        schedule_interval=datetime.timedelta(days=1),
        default_args=default_dag_args) as dag:

    # Create a Cloud Dataproc cluster.
    create_dataproc_cluster = dataproc.DataprocCreateClusterOperator(
        task_id='create_dataproc_cluster',
        # Give the cluster a unique name by appending the date scheduled.
        # See https://airflow.apache.org/docs/apache-airflow/stable/macros-ref.html
        cluster_name='composer-hadoop-tutorial-cluster-{{ ds_nodash }}',
        cluster_config=CLUSTER_CONFIG,
        region=models.Variable.get('gce_region'))

```

```python
    # Run the Hadoop wordcount example installed on the Cloud Dataproc cluster
    # master node.
    run_wordcount = dataproc.DataprocSubmitJobOperator(
        task_id='run_dataproc_hadoop',
        job=HADOOP_JOB)

    # Delete Cloud Dataproc cluster.
    delete_dataproc_cluster = dataproc.DataprocDeleteClusterOperator(
        task_id='delete_dataproc_cluster',
        cluster_name='composer-hadoop-tutorial-cluster-{{ ds_nodash }}',
        region=models.Variable.get('gce_region'),
        # Setting trigger_rule to ALL_DONE causes the cluster to be deleted
        # even if the Dataproc job fails.
        trigger_rule=trigger_rule.TriggerRule.ALL_DONE)

    # Define DAG dependencies.
    create_dataproc_cluster >> run_wordcount >> delete_dataproc_cluster
```