Below is a sample script that uses the CData JDBC driver with the PySpark and AWSGlue modules to extract Smartsheet data and write it to an S3 bucket in CSV format. Make any necessary changes to the script to suit your needs and save the job.

import sys

from awsglue.transforms import \*

from awsglue.utils import getResolvedOptions

from pyspark.context import SparkContext

from awsglue.context import GlueContext

from awsglue.dynamicframe import DynamicFrame

from awsglue.job import Job

args = getResolvedOptions(sys.argv, ['JOB\_NAME'])

sparkContext = SparkContext()

glueContext = GlueContext(sparkContext)

sparkSession = glueContext.spark\_session

##Use the CData JDBC driver to read Smartsheet data from the Sheet\_Event\_Plan\_Budget table into a DataFrame

##Note the populated JDBC URL and driver class name

source\_df = sparkSession.read.format("jdbc").option("url","jdbc:smartsheet:RTK=5246...;OAuthClientId=MyOauthClientId;OAuthClientSecret=MyOAuthClientSecret;CallbackURL=http://localhost:33333;").option("dbtable","Sheet\_Event\_Plan\_Budget").option("driver","cdata.jdbc.smartsheet.SmartsheetDriver").load()

glueJob = Job(glueContext)

glueJob.init(args['JOB\_NAME'], args)

##Convert DataFrames to AWS Glue's DynamicFrames Object

dynamic\_dframe = DynamicFrame.fromDF(source\_df, glueContext, "dynamic\_df")

##Write the DynamicFrame as a file in CSV format to a folder in an S3 bucket.

##It is possible to write to any Amazon data store (SQL Server, Redshift, etc) by using any previously defined connections.

retDatasink4 = glueContext.write\_dynamic\_frame.from\_options(frame = dynamic\_dframe, connection\_type = "s3", connection\_options = {"path": "s3://mybucket/outfiles"}, format = "csv", transformation\_ctx = "datasink4")

glueJob.commit()