## @type: DataSource

## @args: [database = "capstone", table\_name = "project190", redshift\_tmp\_dir = args["TempDir"], transformation\_ctx = "<transformation\_ctx>"]

## @return: <output>

## @inputs: []

import sys

from awsglue.transforms import \*

from awsglue.utils import getResolvedOptions

from pyspark.context import SparkContext

from awsglue.context import GlueContext

from awsglue.job import Job

## @params: [JOB\_NAME]

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

sc = SparkContext()

glueContext = GlueContext(sc)

spark = glueContext.spark\_session

job = Job(glueContext)

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

## @type: DataSource

## @args: [database = "capstone", table\_name = "project190", transformation\_ctx = "datasource0"]

## @return: datasource0

## @inputs: []

datasource0 = glueContext.create\_dynamic\_frame.from\_catalog(database = "capstone", table\_name = "project190", transformation\_ctx = "datasource0")

## @type: ApplyMapping

## @args: [mapping = [("invoiceno", "string", "invoiceno", "string"), ("stockcode", "string", "stockcode", "string"), ("description", "string", "description", "string"), ("quantity", "long", "quantity", "long"), ("invoicedate", "string", "invoicedate", "string"), ("unitprice", "double", "unitprice", "double"), ("customerid", "long", "customerid", "long"), ("country", "string", "country", "string")], transformation\_ctx = "applymapping1"]

## @return: applymapping1

## @inputs: [frame = datasource0]

applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [("invoiceno", "string", "invoiceno", "string"), ("stockcode", "string", "stockcode", "string"), ("description", "string", "description", "string"), ("quantity", "long", "quantity", "long"), ("invoicedate", "string", "invoicedate", "string"), ("unitprice", "double", "unitprice", "double"), ("customerid", "long", "customerid", "long"), ("country", "string", "country", "string")], transformation\_ctx = "applymapping1")

## @type: ResolveChoice

## @args: [choice = "make\_struct", transformation\_ctx = "resolvechoice2"]

## @return: resolvechoice2

## @inputs: [frame = applymapping1]

resolvechoice2 = ResolveChoice.apply(frame = applymapping1, choice = "make\_struct", transformation\_ctx = "resolvechoice2")

## @type: DropNullFields

## @args: [transformation\_ctx = "dropnullfields3"]

## @return: dropnullfields3

## @inputs: [frame = resolvechoice2]

dropnullfields3 = DropNullFields.apply(frame = resolvechoice2, transformation\_ctx = "dropnullfields3")

## @type: DataSink

## @args: [connection\_type = "s3", connection\_options = {"path": "s3://projectparquet1901"}, format = "parquet", transformation\_ctx = "datasink4"]

## @return: datasink4

## @inputs: [frame = dropnullfields3]

datasink4 = glueContext.write\_dynamic\_frame.from\_options(frame = dropnullfields3, connection\_type = "s3", connection\_options = {"path": "s3://projectparquet1901"}, format = "parquet", transformation\_ctx = "datasink4")

job.commit()