Lab3. Run Apache Spark Application on AWS Glue ETL

1. WS Glue Studio의 Job 수정하기
   1)[Lab. Using AWS Glue Workflows]에서 생성한 Job 상세페이지로 이동
   2)[Script] 탭에서 다음의 코드를 16 라인에 붙여넣기
      -즉, job이 초기화 후 --> job.init(args['JOB_NAME'], args)

```
      #공통칼럼
      new_cols_expr = ['VendorID as vendor_id', 'passenger_count','trip_distance','RatecodeID as ratecode_id',
      'store_and_fwd_flag',
                  'PULocationID as pu_location_id','DOLocationID as do_location_id','payment_type','fare_amount','extra',
                  'mta_tax','tip_amount','tolls_amount','improvement_surcharge','total_amount','congestion_surcharge']

      #yellow_df용 칼럼
      yellow_cols = ['tpep_pickup_datetime as pickup_datetime', 'tpep_dropoff_datetime as dropoff_datetime']
      yellow_cols.extend(new_cols_expr)

      #green_df용 칼럼
      green_cols = ['lpep_pickup_datetime as pickup_datetime','lpep_dropoff_datetime as dropoff_datetime']
      green_cols.extend(new_cols_expr)


      input_path = 's3a://henry-datalake-bucket/input'
      output_path = 's3a://henry-datalake-bucket/output'

      count = 0
      for ym in ['2022-01', '2022-02', '2022-03', '2022-04', '2022-05', '2022-06', '2022-07', '2022-08', '2022-09', '2022-10',
      '2022-11']:
         green_df = spark.read.parquet(f'{input_path}/green_tripdata_{ym}.parquet') \
            .selectExpr(green_cols).withColumn('taxi_type', f.lit('GREEN'))
         yellow_df = spark.read.parquet(f'{input_path}/yellow_tripdata_{ym}.parquet') \
            .selectExpr(yellow_cols).withColumn('taxi_type', f.lit('YELLOW'))
         union_df = green_df.union(yellow_df)
         union_df.repartition(1).write.option('header', 'true').mode('overwrite').csv(f'{output_path}/ym={ym}/')
         count += 1
      print(f'{count} Files Successfully Saved')
```

   3)[Script] 탭 코드의 마지막 라인은 다음과 같다.

```
      ...
      job.commit()
```

   4){계정}-glue-job 페이지 상단의 [Save] 버튼 클릭

2. S3 "output" folder 삭제
   1)[Amazon S3]의 "{계정}-datalake-bucket" 상세페이지로 이동
   2)"output" folder 삭제

3. Job 실행
   1)다시 {계정}-glue-job 페이지로 돌아와서 페이지 상단의 [Run] 버튼 클릭하여 실행
   2)[Runs] 탭으로 이동하여 [Run status]가 Running임을 확인
   3)Error 발생 : Error의 메시지는 다음과 같다.

```
      An error occurred while calling o92.parquet. s3a://henry-datalake-bucket/input/green_tripdata_2022-01.parquet:
      getFileStatus on s3a://henry-datalake-bucket/input/green_tripdata_2022-01.parquet:
      com.amazonaws.services.s3.model.AmazonS3Exception: Forbidden (Service: Amazon S3; Status Code: 403; Error Code:
      403 Forbidden; Request ID: EBGAJZPV53P8SFNV; S3 Extended Request ID:
      POVMjMspjAaW9/jZ4UHQzOaavhitane6agEuXTSb7CdmCmcDq9Ig4Maor/LMVbPlmGBQC0PLjs0=; Proxy: null), S3 Extended
      Request ID: POVMjMspjAaW9/jZ4UHQzOaavhitane6agEuXTSb7CdmCmcDq9Ig4Maor/LMVbPlmGBQC0PLjs0=:403 Forbidden
```

   4)에러의 이유는 현재 {계정}-glue-job의 [Job details] 탭에서 확인해보면, [IAM Role]이 {계정}-glue-role인데, 이 권한은 S3 Bucket에 대한 권한이 없기 때문이다. 해결하는 방법은 Access Key와 Secret Access Key를 Code에 넣거나, {계정}-glue-role에 S3 Bucket에 접근할 권한을 추가하면 된다.

4. {계정}-glue-role에 새 권한 추가
   1)[IAM]의 [역할]로 이동하여, 현재 Job에 설정되어 있는 역할인 {계정}-glue-role을 검색한다.
   2)이 Role의 [권한 정책] 섹션에서 보면 "AWSGlueServiceRole" 정책 하나만 가지고 있는 것을 확인할 수 있다.
   3)이 Role에 권한을 추가하기 위해, [권한 추가] > [인라인 정책 생성]을 클릭한다.
   4)[정책 생성] 페이지에서,
      -[JSON] 탭으로 이동하여 다음 코드를 "Statement"에 리스트로 추가

```
      {
         "Version": "2012-10-17",
         "Statement": [
            {
               "Sid": "AllowS3GetAndPut",
               "Effect": "Allow",
               "Action": ["s3:GetObject", "s3:PutObject", "s3:DeleteObject"],
               "Resource": ["arn:aws:s3:::{계정}-datalake-bucket*"]
               <---Bucket의 [속성] 탭에 있는 [Amazon 리소스 이름(ARN)] 복사해서 넣을 것
               <---ARN 제일 뒤에 반드시 *을 넣을 것
            }
         ]
      }
```

5)페이지 하단의 [정책 검토] 버튼 클릭
6)[정책 검토] 페이지에서
-이름 : {계정}-glue-s3-access-policy
-[정책 생성] 버튼 클릭

7)다시 {계정}-glue-role의 상세 페이지로 돌아오면 [권한 정책] 섹션에서 방금 추가한 [정책 이름] 확인할 것

5. {계정}-glue-job 코드 수정 후, 다시 실행
1)다시 {계정}-glue-job 상세 페이지로 이동하여 [Run] 버튼을 클릭하여 실행한다.
2)[Runs] 탭으로 이동하여 현재 [Run status]가 Running임을 확인한다.
3)페이지 아래의 상세 페이지에 보면 [Cloudwatch logs] 항목이 있는데, 여기서 "Output logs"를 클릭하여 확인한다.
4)현재 Job의 [Requested number of workers]의 값이 3개로 설정했기 때문에 [CloudWatch]의 [로그 스트림]도 3개임을 확인할 수 있다.
5)3개의 [로그 스트림]중에서 surfix가 없는 [로그 스트림]을 클릭한다.
6)다음과 같은 오류를 확인할 수 있다.
...main WARN JNDI lookup class is not available because this JRE does not support JNDI. JNDI string lookups will not be available, continuing configuration. java.lang.ClassNotFoundException:
...
...

7)Error를 보다 정확하게 보기 위해 {계정}-glue-job의 [Runs] 탭으로 이동한다.
8)오류의 원인은 다음과 같다.
NameError: name 'f' is not defined

9)물론 페이지 아래의 상세 내용 중에 [Cloudwatch logs] 항목 중 [Error logs]의 창을 열어서 확인하는 방법도 있다.
10)동일하게 [CloudWatch] 페이지로 이동하고, 동일하게 [로그 스트림]이 3개이며, surpix가 없는 [로그 스트림]의 링크를 클릭할 수 있다.
11)목록에서 "ERROR [main]"을 찾으면 된다.
12)여기에서 확인한 오류는 다음과 같다.
...ERROR [main] glue.ProcessLauncher (Logging.scala:logError(73)): Error from Python:Traceback (most recent call last):
File "/tmp/henry-glue-job.py", line 37, in <module>
.selectExpr(green_cols).withColumn('taxi_type', f.lit('GREEN'))
NameError: name 'f' is not defined

13)'f'라는 단어가 정의안되어 있다는 오류이다.
14)즉, py파일에서 코드를 복사할 때 'f'에 대한 import 구문을 넣지 않았기 때문에 다음과 같은 코드를 7라인에 넣는다.
import pyspark.sql.functions as f

15)페이지 상단의 [Save] 버튼을 클릭하여 저장한 후, [Run] 버튼 실행

6. 결과 확인
1)[Runs] 탭 페이지 하단의 내용 중에 [Cloudwatch logs] 항목 > [Output logs] 링크를 클릭한다.
2)[CloudWatch]의 [로그 스트림] 섹션을 보면 3개의 [로그 스트림] 있고, 이 중에서 surpix가 없는 [로그 스트림] 링크를 클릭한다.
3)만일 {계정}-glue-job의 [Runs] 탭에서 [Run status]가 "Succeeded"이면, [CloudWatch]의 [로그 스트림]에는 다음과 같은 Log가 기록됨을 확인할 수 있다.
11 Files Successfully Saved

4)[Amazon S3]의 {계정}-datalake-bucket 으로 이동해서 "output" 폴더가 있는 것을 확인하고, 각 하위 폴더가 생성되어 있으며, 그 폴더 하위에 csv 파일이 있음을 확인할 수 있다.

7. 버전 관리하기
1)본인의 GitHub 페이지 로그인
2)새 Repository 생성
-Repository name : {계정}-glue-job
-Public
-[Add a README file] check
-[Create Repository] 클릭
3)[Personal Access Tokens] 생성
-Token 값 복사하여 별도의 텍스트에디터에 저장
-ex)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

4)다시 AWS Console로 돌아와서, {계정}-glue-job의 [Version Control] 탭으로 이동
5)[Git configuration] 항목에서,
-[Git service] : "GitHub"
-[Personal access token] : 방금 생성한 token 값
-[Repository owner] : GitHub 계정명

6)[Repository configuration] 항목에서,
-[Repository] : 목록에서 방금 생성한 GitHub Repository 선택
-[Branch] : main

7)Job 페이지 상단의 [Save] 버튼 클릭
8)Job 페이지 상단의 [Actions] > [Version control] > [Push to repository] 클릭
9)[Push to repository] 팝업창에서, [Confirm] 버튼 클릭
10)Push에 성공하면 다음과 같은 메시지가 나옴.
Latest commit
Latest available commit {CommitID} is from GitHub on {Repository name}/main.

11)GitHub 페이지로 이동하여 {계정}-glue-jog.json과 {계정}-glue-job.py의 파일 확인
12){계정}-glue-job.py의 파일의 내용은 Job의 [Scripts]의 코드와 같음을 확인할 수 있다.