2023 지방기능경기대회 채점기준

1. 채점상의 유의사항	직 종 명	클라우드컴퓨팅

※ 다음 사항을 유의하여 채점하시오.

1) AWS의 지역은 ap-northeast-2을 사용합니다.
2) 웹페이지 접근은 크롬이나 파이어폭스를 이용합니다.
3) 웹페이지에서 언어에 따라 문구가 다르게 보일 수 있습니다.
4) shell에서의 명령어의 출력은 버전에 따라 조금 다를 수 있습니다.
5) 채점 진행 전 환경 셋업을 위해 다음 사항을 확인해야 합니다.
   - Bastion에 SSH로 접근 가능한지 확인합니다.
   - Bastion에서 curl, jq, awscli가 설치되었는지 확인합니다.
   - Bastion에서 IAM Role이 맵핑되어 awscli로 AWS 모든 리소스에 접근 가능한지 확인합니다.
   - aws sts get-caller-identity 명령을 통해 선수의 계정이 아닌 다른 계정에 접근하고 있는지 확인합니다. 만약, 다른 계정이라면 부정행위를 의심할 수 있습니다.
6) 문제지와 채점지에 있는 <> 는 변수입니다. 해당 부분을 변경해 입력합니다.
7) 채점은 문항 순서대로 진행해야 합니다.
8) 삭제된 내용은 되돌릴 수 없음으로 유의하여 채점을 진행합니다.
9) 이의신청까지 종료된 이후 선수가 생성한 클라우드 리소스를 삭제합니다.
10) 부분 점수가 있는 문항은 채점 항목에 부분 점수가 적혀져 있습니다.
11) 부분 점수가 따로 없는 문항은 전체 다 맞아야 점수로 인정됩니다.
12) 채점 전 채점환경 구성을 위해 ~/.aws/config 에 아래 내용이 추가되도록 합니다.

```
aws configure
/////
[default]
region = ap-northeast-2
output = json
/////
```

2. 채점기준표

1) 주요항목별 배점	직 종 명	클라우드컴퓨팅

과제 번호	일련 번호	주요항목	배점	채점방법		채점시기		비고
				독립	합의	경기 진행중	경기 종료후	
제2과제	1	API Gateway	9		○		○	
	2	EC2	3		○		○	
	3	S3	2		○		○	
	4	Kinesis Data Streams	2		○		○	
	5	Kinesis Data Firehose	5		○		○	
	6	Glue 크롤러	5		○		○	
	7	Glue Studio 작업	7		○		○	
	8	Glue 워크플로	7		○		○	
합 계			40					

2) 채점방법 및 기준

과제 번호	일련 번호	주요항목	일련 번호	세부항목(채점방법)	배점
제2과제	1	API Gateway	1	API Gateway 생성 확인	1
			2	리소스 확인	2
			3	메소드 확인	2
			4	배포 스테이지 확인	2
			5	Kinesis 전달 확인	2
	2	EC2	1	인스턴스 확인	1.5
			2	Public IP 확인	1.5
	3	S3	1	S3 버킷 생성 확인	1
			2	파일 업로드 확인	1
	4	Kinesis Data Streams	1	Data Stream 생성 확인	1
			2	샤드 개수 확인	1
	5	Kinesis Data Firehose	1	Delivery Stream 생성 확인	1
			2	소스 확인	2
			3	저장소 설정 확인	2
	6	Glue 크롤러	1	크롤러 생성 확인	1
			2	데이터 카탈로그 생성 확인	2
			3	테이블 구조 확인	2
	7	Glue Studio 작업	1	작업 생성 확인	1
			2	데이터 변환 및 저장 확인	2
			3	데이터 카탈로그 업데이트 확인	2
			4	테이블 구조 확인	2
	8	Glue 워크플로	1	워크플로 생성 확인	1
			2	Glue 크롤러 실행 확인	3
			3	Glue Studio 작업 실행 확인	3
	총점				40

3) 채점 내용

순번	채점 항목
1-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "wsi-api"가 출력되는지 확인합니다. (0.5점) aws apigateway get-rest-apis --query "items[?name=='wsi-api'].name"
1-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력합니다. api=$(aws apigateway get-rest-apis --query "items[?name=='wsi-api'].id" --output text) 3) 아래 명령어 입력 후 "/api"가 출력되는지 확인합니다. aws apigateway get-resources --rest-api-id $api --query "items[?path=='/api'].path"
1-3	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력합니다. api=$(aws apigateway get-rest-apis --query "items[?name=='wsi-api'].id" --output text) resource=$(aws apigateway get-resources --rest-api-id $api --query "items[?path=='/api'].id" --output text) 3) 아래 명령어 입력 후 "arn:aws:apigateway:ap-northeast-2:kinesis:action/PutRecord"가 출력되는지 확인합니다. aws apigateway get-method --rest-api-id $api --resource-id $resource --http-method POST ₩ --query "methodIntegration.uri"
1-4	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력합니다. api=$(aws apigateway get-rest-apis --query "items[?name=='wsi-api'].id" --output text) 3) 아래 명령어 입력 후 "prod"가 출력되는지 확인합니다. aws apigateway get-stages --rest-api-id $api --query "item[?stageName=='prod'].stageName"

순번	채점 항목
1-5	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력합니다. aws s3 rm --recursive s3://**<선수가 생성한 S3 버킷>**/data/raw/ api=$(aws apigateway get-rest-apis --query "items[?name=='wsi-api'].id" --output text) uri=https://$api.execute-api.ap-northeast-2.amazonaws.com/prod/api echo "{₩"testkey₩":₩"testvalue₩"}" > post curl -X POST -H "Content-Type: application/json" -d @post $uri 3) 5분 정도 지난 후 아래 명령어를 입력합니다. aws s3 mv --recursive s3://**<선수가 생성한 S3 버킷>**/data/raw/ ./raw 4) 아래 명령어 입력 후 {"testkey":"testvalue"}가 출력되는지 확인합니다. cat raw/*/*/*/*/*
2-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력하여 **"i-"**로 시작하는 문구를 받아오는지 확인합니다. aws ec2 describe-instances --filter Name=tag:Name,Values=wsi-bastion-ec2 ₩ --query "Reservations[].Instances[].InstanceId"
2-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력해 나오는 IP를 기록합니다. aws ec2 describe-instances --filter Name=tag:Name,Values=wsi-bastion-ec2 ₩ --query "Reservations[].Instances[].PublicIpAddress" 3) 아래 명령어 입력 후 출력되는 IP리스트 중에 2)번에서 출력된 IP가 있는지 확인합니다. aws ec2 describe-addresses --query "Addresses[].PublicIp"
3-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력하여 "wsi-<비번호>-<4자리 임의 영문>-etl" 형식의 문구가 출력되는지 확인합니다. aws s3api list-buckets --query "Buckets[].Name"
3-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력하여 titles.json이 출력에 포함되는지 확인합니다. aws s3 ls s3://**<선수가 생성한 S3 버킷>**/data/ref/

순번	채점 항목
4-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "wsi-data-stream"가 출력되는지 확인합니다. aws kinesis describe-stream --stream-name wsi-data-stream ₩ --query "StreamDescription.StreamName"
4-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "1"이 출력되는지 확인합니다. aws kinesis list-shards --stream-name wsi-data-stream --query "Shard[].ShardId" | wc -l
5-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "wsi-delivery-stream"가 출력되는지 확인합니다. aws firehose describe-delivery-stream --delivery-stream-name wsi-delivery-stream ₩ --query "DeliveryStreamDescription.DeliveryStreamName"
5-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "KinesisStreamAsSource"가 출력되는지 확인합니다. aws firehose describe-delivery-stream --delivery-stream-name wsi-delivery-stream ₩ --query "DeliveryStreamDescription.DeliveryStreamType"
5-3	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "arn:aws:s3:::**<선수가 생성한 S3 버킷>**"가 출력되는지 확인합니다. aws firehose describe-delivery-stream --delivery-stream-name wsi-delivery-stream ₩ --query "DeliveryStreamDescription.Destinations[].S3DestinationDescription.BucketARN" 3) 아래 명령어 입력 후 "/data/raw/"가 출력되는지 확인합니다. aws firehose describe-delivery-stream --delivery-stream-name wsi-delivery-stream ₩ --query "DeliveryStreamDescription.Destinations[].S3DestinationDescription.Prefix"
6-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "wsi-glue-crawler"가 출력되는지 확인합니다. aws glue get-crawler --name wsi-glue-crawler --query "Crawler.Name"
6-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "wsi-glue-database"가 출력되는지 확인합니다. aws glue get-database --name wsi-glue-database --query "Database.Name" 3) 아래 명령어 입력 후 "ref"가 출력되는지 확인합니다. aws glue get-table --database-name wsi-glue-database --name ref --query "Table.Name" 3) 아래 명령어 입력 후 "raw"가 출력되는지 확인합니다. aws glue get-table --database-name wsi-glue-database --name raw --query "Table.Name"

순번	채점 항목
6-3	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "title_id", "title"이 출력되는지 확인합니다. aws glue get-table --database-name wsi-glue-database --name ref ₩ --query "Table.StorageDescriptor.Columns[].Name" 3) 아래 명령어 입력 후 "uuid", "device_ts", "device_id", "title_id", "device_type"이 출력에 포함되는지 확인합니다. aws glue get-table --database-name wsi-glue-database --name raw ₩ --query "Table.StorageDescriptor.Columns[].Name"
7-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "wsi-glue-job"이 출력되는지 확인합니다. aws glue get-job --job-name wsi-glue-job --query "Job.Name"
7-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 채점용으로 지급된 SampleLogData.json 파일을 Bastion 서버 /home/ec2-user/ 에 복사합니다. 3) 아래 명령어를 입력합니다. aws s3 rm --recursive s3://**<선수가 생성한 S3 버킷>**/result/ aws s3 rm --recursive s3://**<선수가 생성한 S3 버킷>**/data/raw/ aws s3 cp ./SampleLogData.json s3://**<선수가 생성한 S3 버킷>**/data/raw/2022/ aws glue start-job-run --job-name wsi-glue-job 4) 시간 경과 후 아래 명령어를 입력합니다. aws s3 cp --recursive s3://**<선수가 생성한 S3 버킷>**/result/ ./result 5) 아래 명령어 입력 후 2990 ~ 3010 사이의 값이 출력되는지 확인합니다. cat ./result/* | wc -l 6) 아래 명령어 입력 후 device_id, device_ts, device_type, title, title_id, uuid가 출력되는지 확인합니다. cat ./result/* | jq -r 'keys[]' | sort -u
7-3	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "result"가 출력되는지 확인합니다. aws glue get-table --database-name wsi-glue-database --name result --query "Table.Name"
7-4	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "title_id", "title", "uuid", "device_ts", "device_id", "device_type"이 출력에 포함되는지 확인합니다. aws glue get-table --database-name wsi-glue-database --name result ₩ --query "Table.StorageDescriptor.Columns[].Name"

순번	채점 항목
8-1	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력 후 "wsi-glue-workflow"가 출력되는지 확인합니다. aws glue get-workflow --name wsi-glue-workflow --query "Workflow.Name"
8-2	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어 입력합니다. aws glue delete-table --database-name wsi-glue-database --name ref aws glue delete-table --database-name wsi-glue-database --name raw aws glue start-workflow-run --name wsi-glue-workflow 시간 경과 후 채점을 진행합니다. 3) 아래 명령어 입력 후 "title_id", "title"이 출력되는지 확인합니다. aws glue get-table --database-name wsi-glue-database --name ref ₩ --query "Table.StorageDescriptor.Columns[].Name" 4) 아래 명령어 입력 후 "uuid", "device_ts", "device_id", "title_id", "device_type"이 출력에 포함되는지 확인합니다. aws glue get-table --database-name wsi-glue-database --name raw ₩ --query "Table.StorageDescriptor.Columns[].Name"
8-3	1) SSH를 통해 Bastion 서버에 접근합니다. 2) 아래 명령어를 입력합니다. aws glue delete-table --database-name wsi-glue-database --name result aws glue start-workflow-run --name wsi-glue-workflow 시간 경과 후 채점을 진행합니다. 3) 아래 명령어 입력 후 "title_id", "title", "uuid", "device_ts", "device_id", "device_type"이 출력에 포함되는지 확인합니다. aws glue get-table --database-name wsi-glue-database --name result ₩ --query "Table.StorageDescriptor.Columns[].Name"