http://bit.ly/aws-analytics-hol

AWS Analytics

Hands-on Guidebook

1st June 2019

김태현 (kimtaehy@amazon.com), Analytics Specialist SA

Corrections or feedback on the lab guide, please email me at: kimtaehy@amazon.com

Table of Contents

1. Overview

Lab 개요

Amazon Athena 는 표준 SQL 을 사용해 Amazon S3 에 저장된 데이터를 간편하게 분석할 수 있는 대화식 쿼리 서비스이고 AWS Glue 는 데이터 분석을 위해 손쉽게 데이터를 준비하고 로드할 수 있게 지원하는 완전관리형 ETL(추출, 변환 및 로드) 서비스입니다. 이 워크샵에서는 Amazon Athena 를 통해 S3 에 저장된 데이터를 분석하고 AWS Glue 를 통해 ETL 및 머신러닝을 실습하면서 Amazon Athena 및 AWS Glue 의 기능을 알아보도록 하겠습니다.

목표

- AWS Glue Crawler를 사용하여 S3에 저장된 데이터를 Catalog에 등록합니다.
- Amazon Athena를 이용해 Catalog에 등록된 데이터를 조회하고 분석합니다.
- AWS Glue의 Dev endpoint 및 SageMaker notebook을 생성하고 AWS Glue ETL 코드를 작성하고 실행합니다.
- AWS Glue에서 JDBC 연결을 통해 데이터를 RDS로 읽고 쓰기를 수행합니다.
- AWS Glue SageMaker notebook에서 머신러닝 모델을 만들고 학습 및 테스트를 수행합니다.

2. Prerequisite

실습을 위해서는 AWS IAM, S3, Amazon Glue, AWS Glue 자원을 생성할 수 있는 권한을 가진 계정이 필요합니다. 이번 실습은 Seoul(ap-northeast-2) region 에서 실행되고 Browser 는 최신 버전의 Chrome 또는 Firefox 를 사용하셔야 합니다.

※ 주의 사항:
- Notebook 안의 Cell 에서 코드 실행 후 결과 값이 나오는 데는 수 초에서 수 분이 걸립니다.
- 실습 완료 후에는 아래 가이드에 따라 생성된 자원을 꼭 종료/삭제해 주세요.

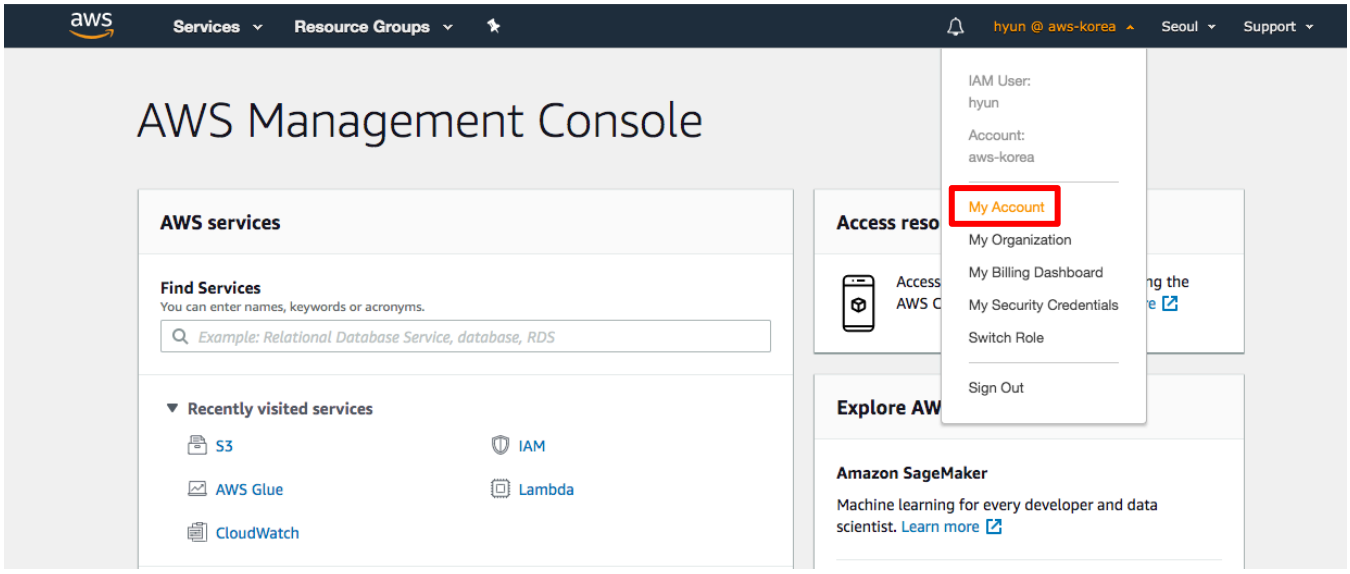
Amazon Web Service account 생성

AWS 계정이 없는 경우 다음 link 를 참고해서 계정을 생성하세요.

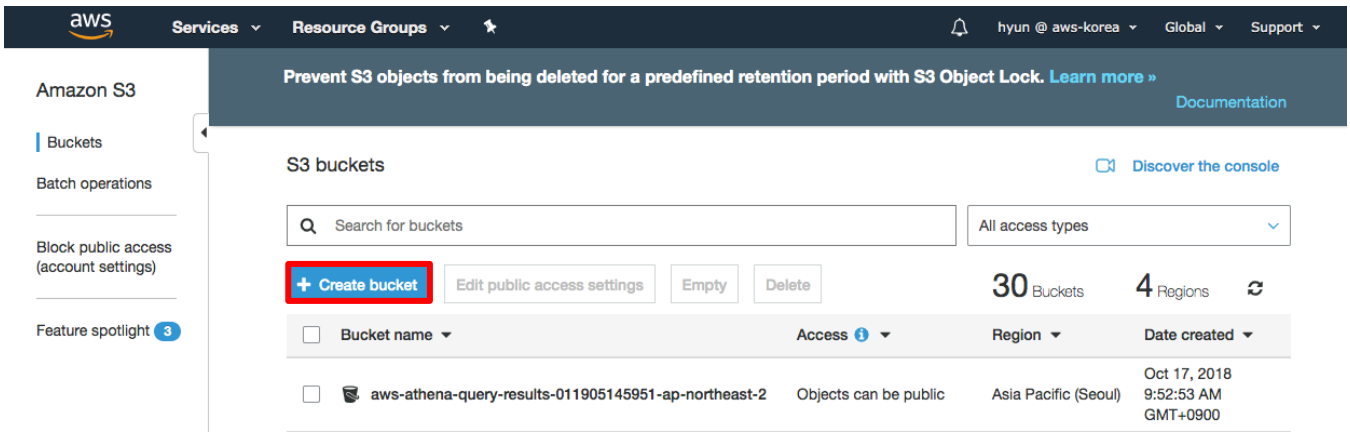
http://bit.ly/create-aws-account-id

S3 버킷 생성

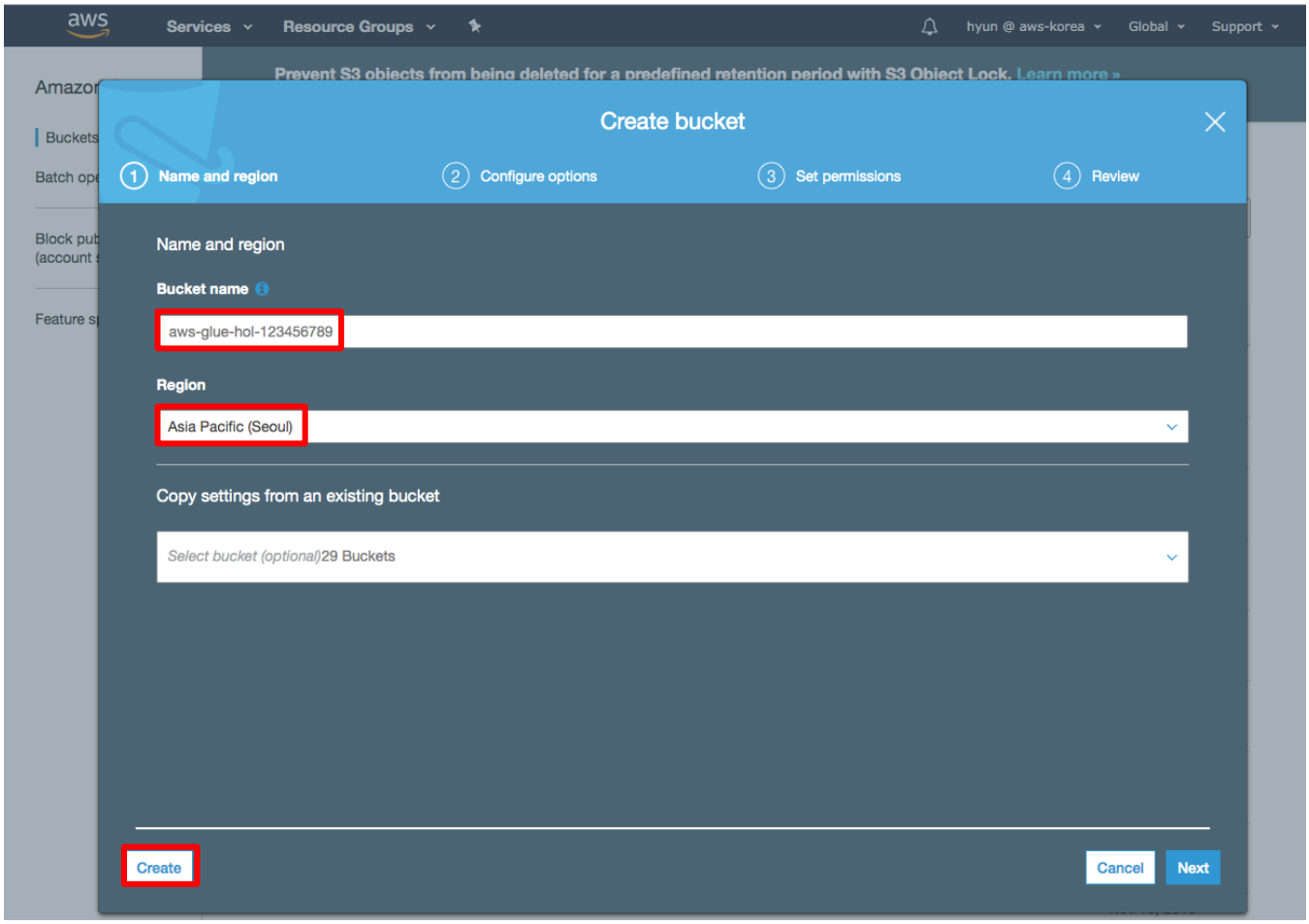
2-1. S3 버킷 이름은 중복된 이름을 가질 수 없으므로, 콘솔에서 My Account 메뉴를 클릭하여 Account Id 정보를 확인하고 복사 또는 메모합니다.



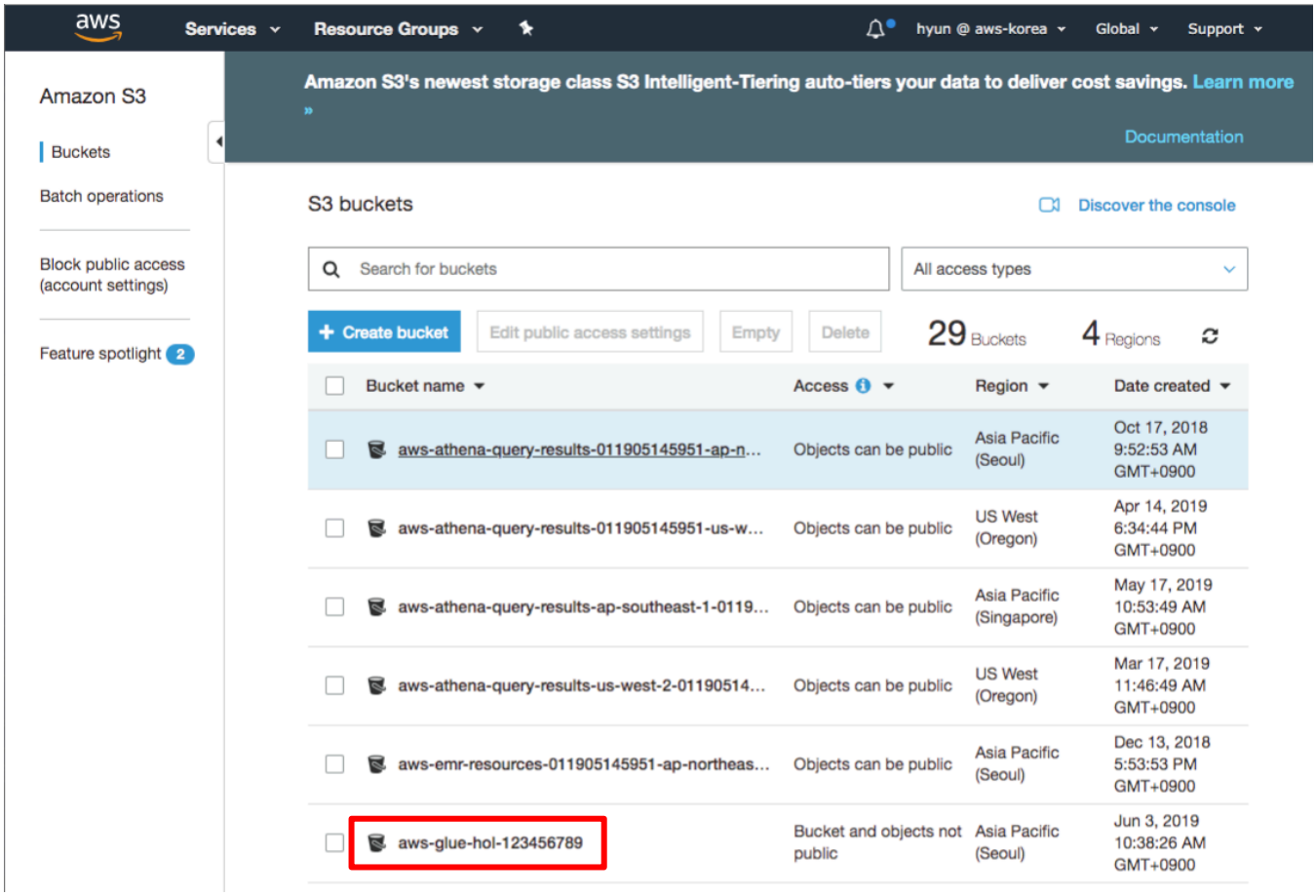
2-2. S3 버킷 생성을 위해 S3 콘솔로 이동 후 Create bucket 버튼을 클릭합니다.



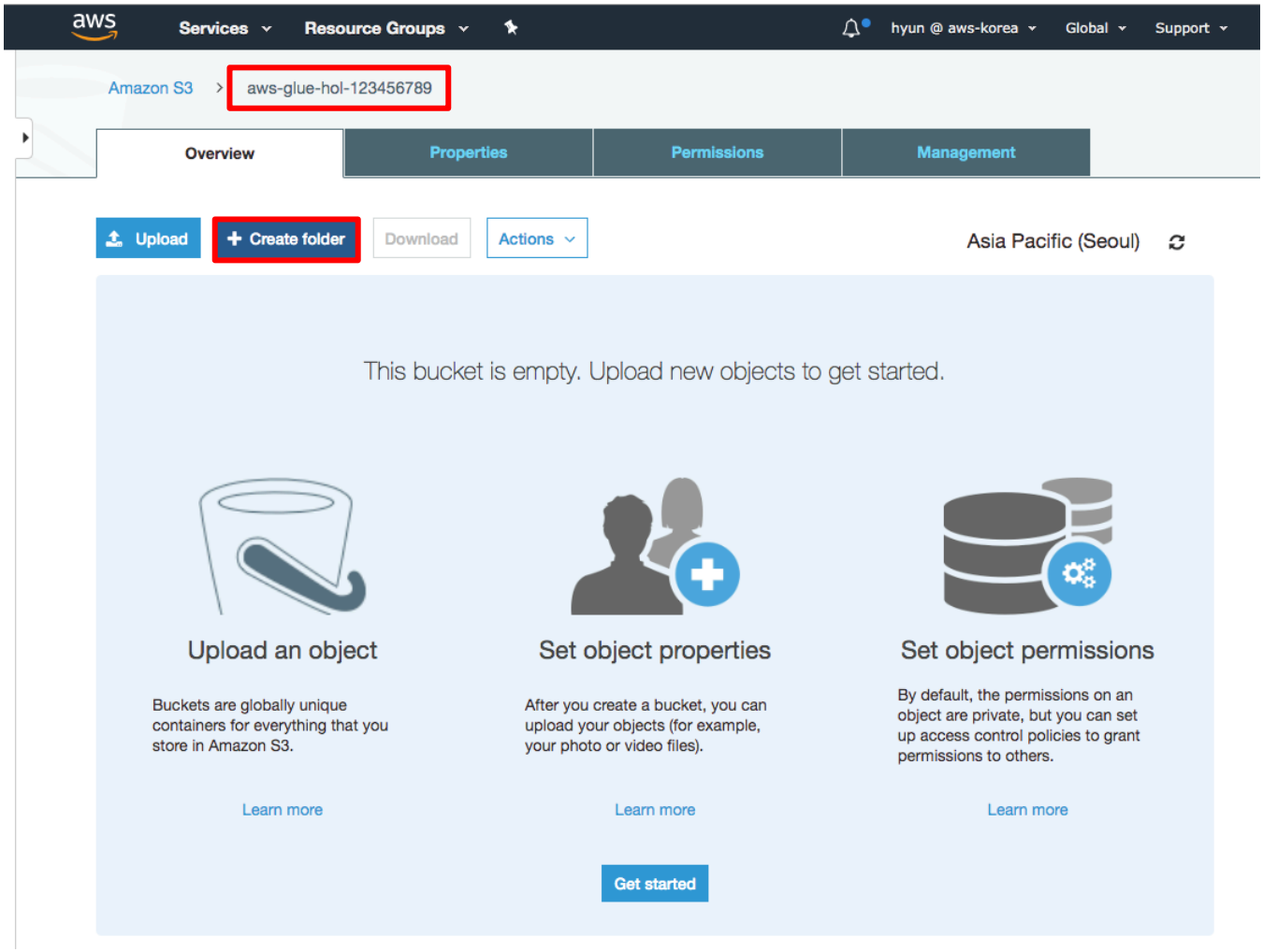
2-3. Bucket name 에 aws-glue-hol-[account-id]를 입력합니다. Region 은 Asia Pacific (Seoul)로 선택된 것을 확인 후, Create 버튼을 클릭하여 S3 버킷을 생성합니다.



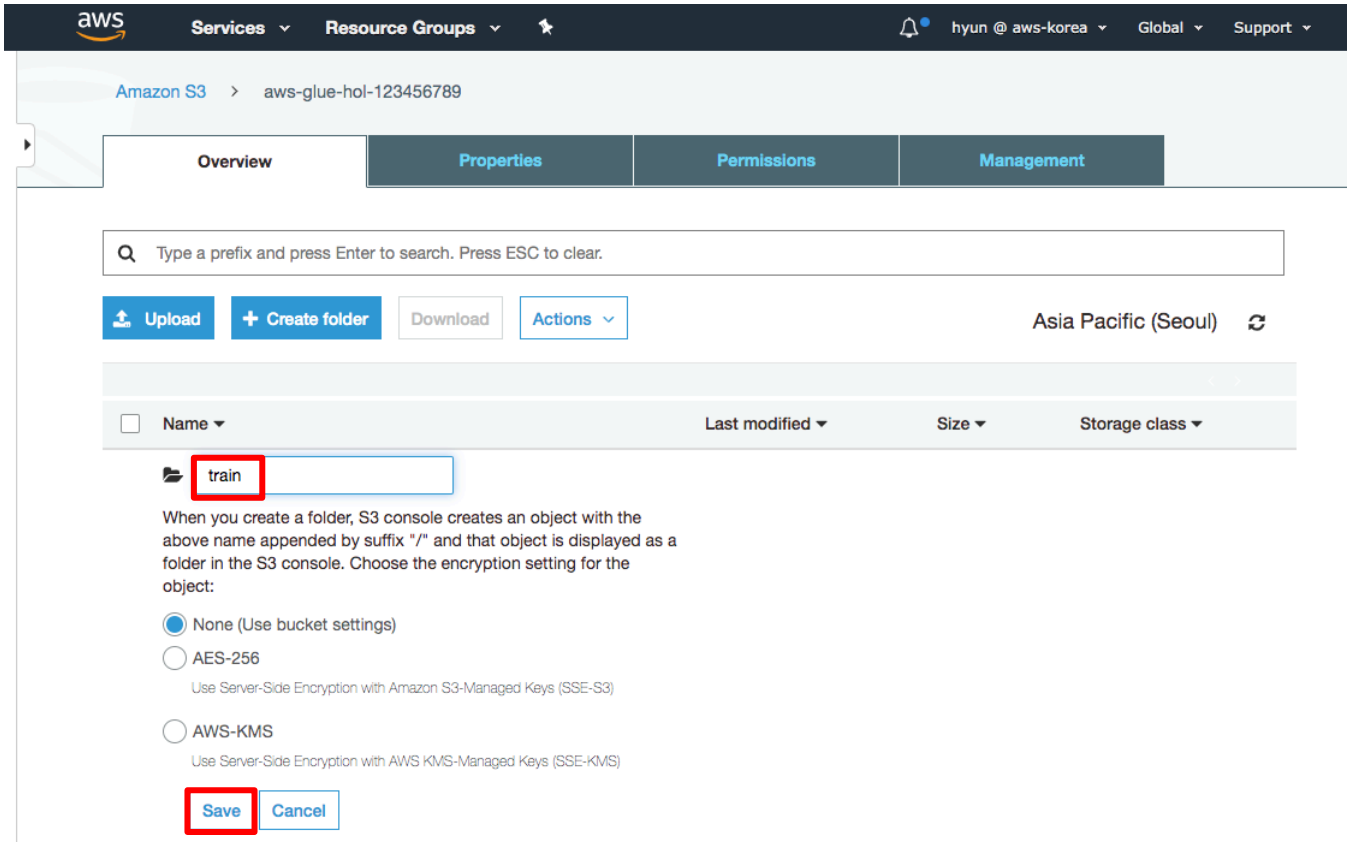
2-4. 생성한 Bucket name 을 클릭하여 Bucket 으로 이동합니다.



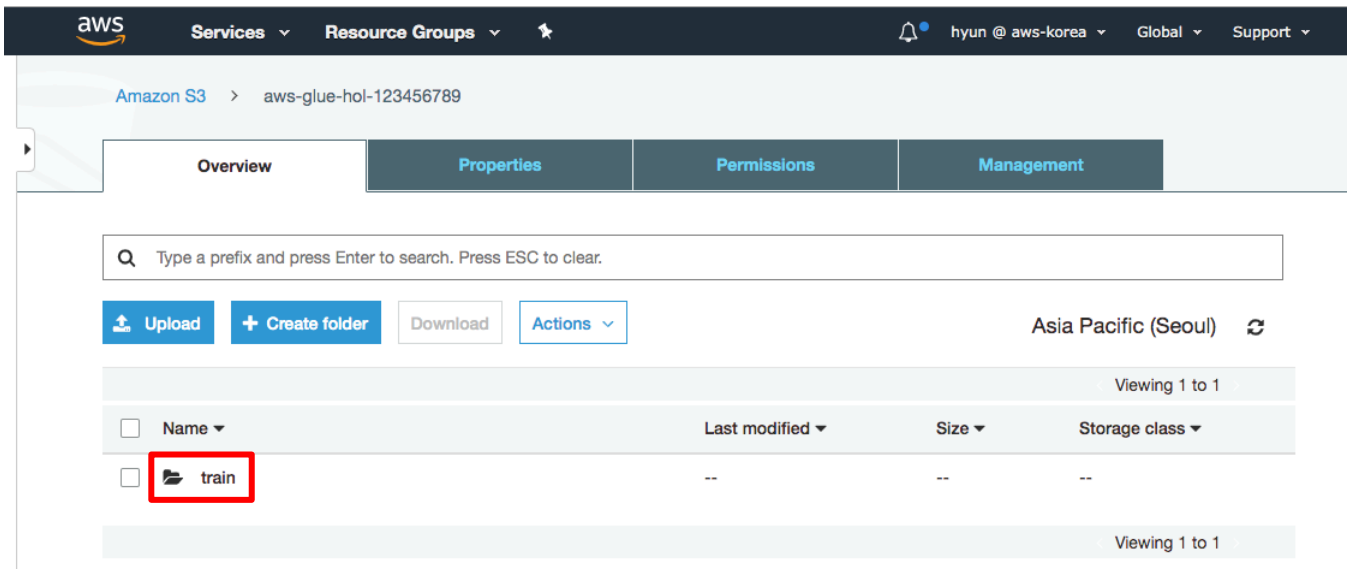
2-5. train 폴더를 생성하기 위해 Create folder 버튼을 클릭합니다.



2-6. New folder 부분에 train 을 입력하고 Save 버튼을 클릭합니다.



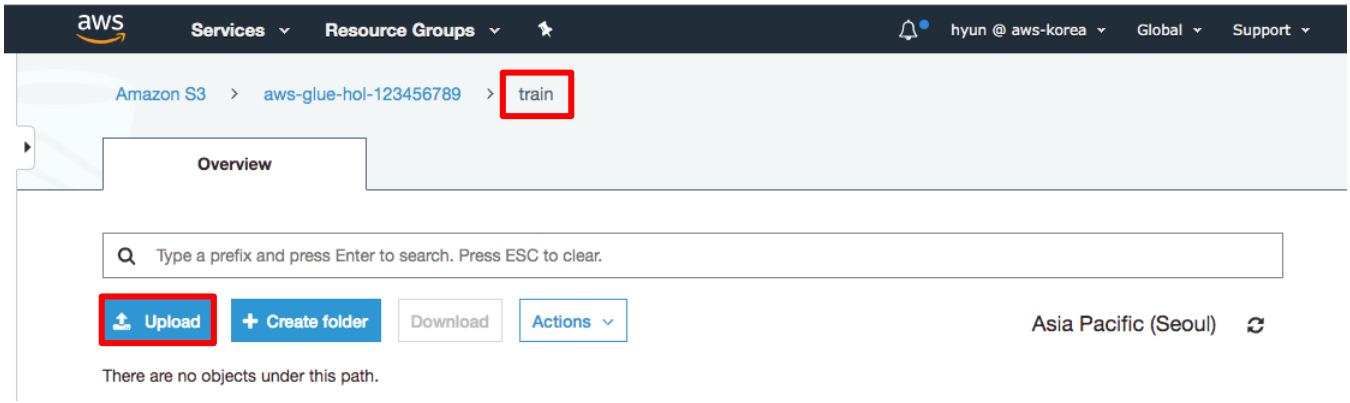
2-7. 아래와 같이 aws-glue-hol-[account-id] 버킷에 총 1 개의 폴더가 생성된 것을 확인할 수 있습니다.



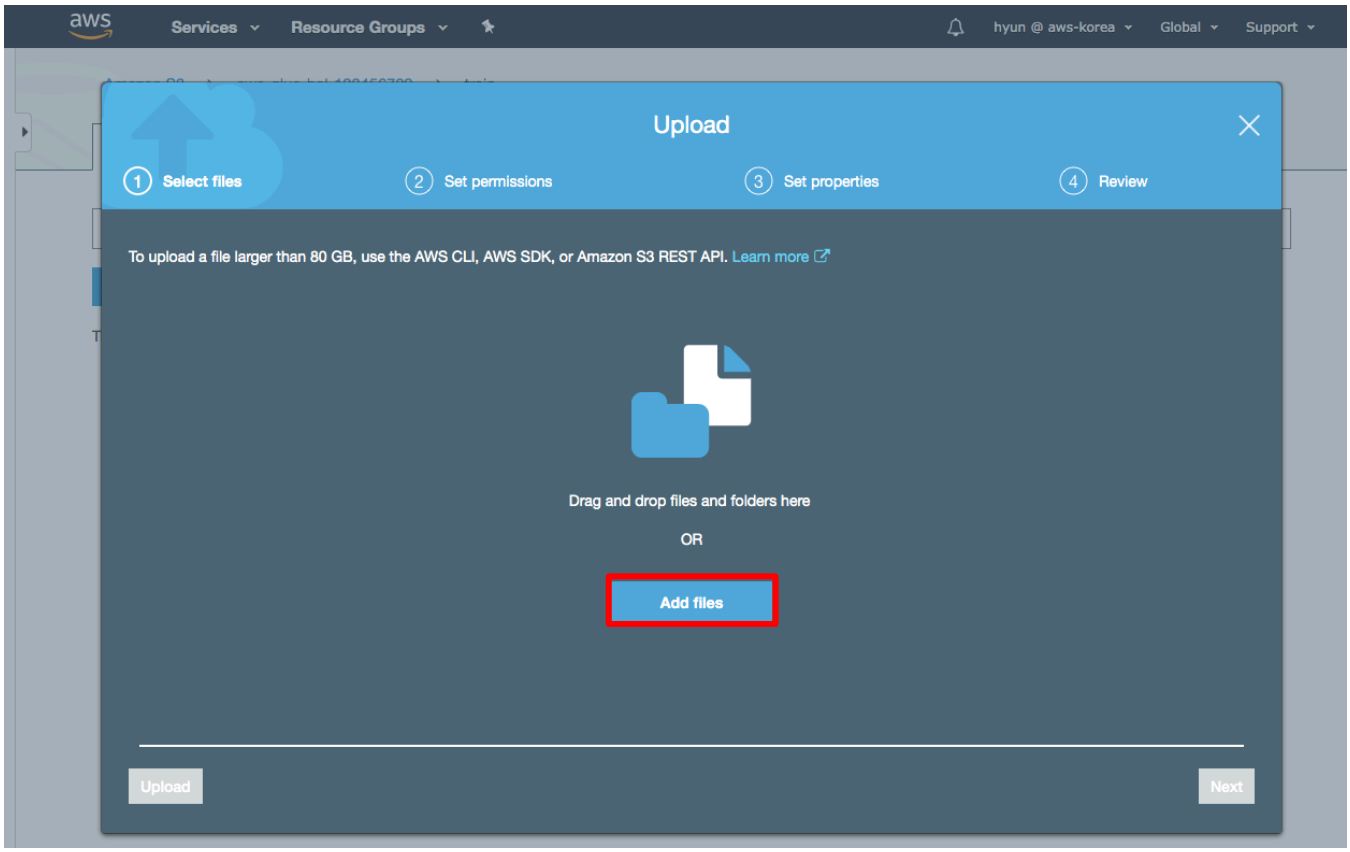
파일 업로드

2-8. 아래 link 의 hol-titanic-train 파일을 다운로드 받습니다. 다운로드 받은 파일을 S3 train 폴더에 Upload 하기 위해 train 폴더 Name 을 클릭하여 이동한 후 Upload 버튼을 클릭합니다.

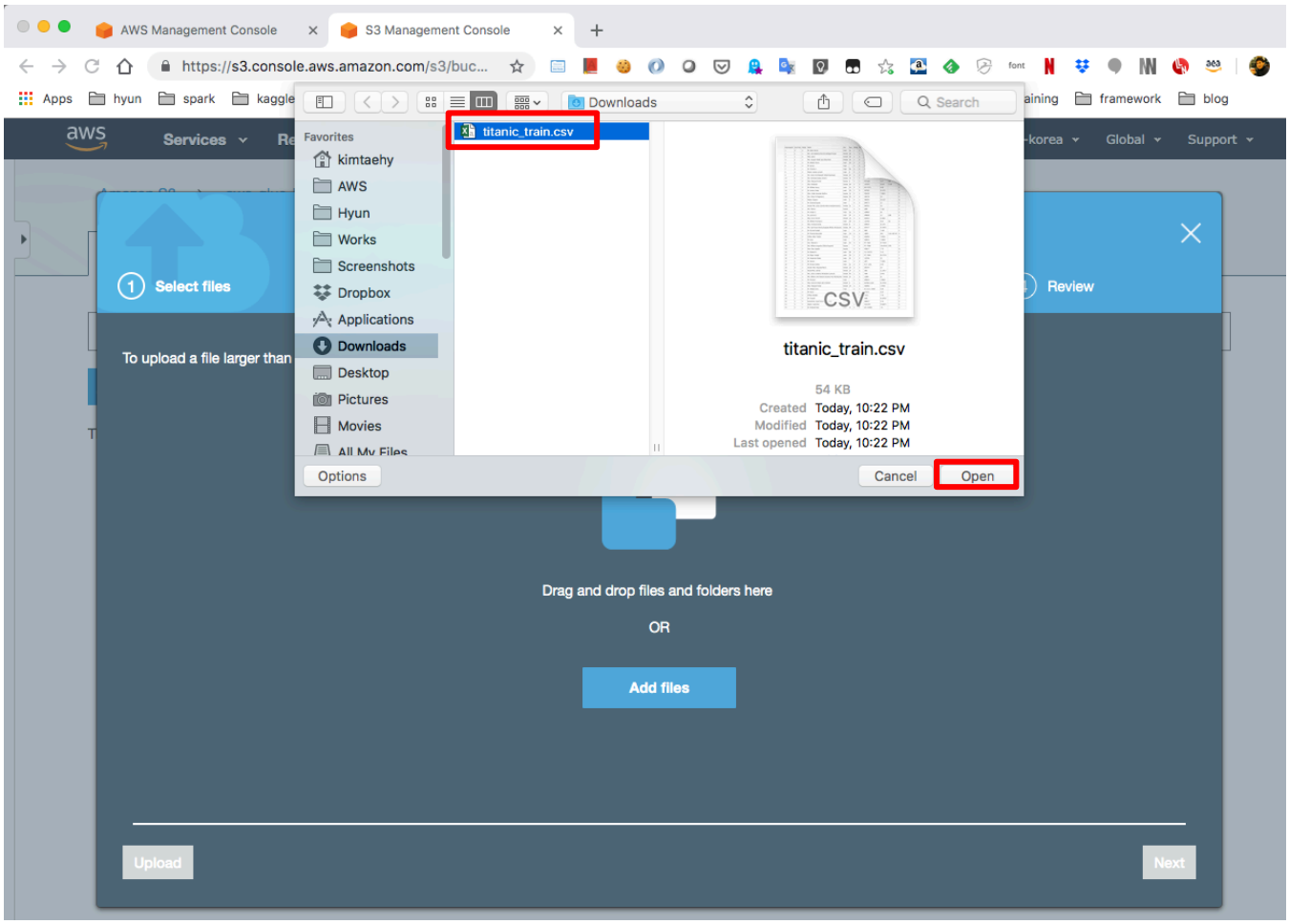
http://bit.ly/hol-titanic-train



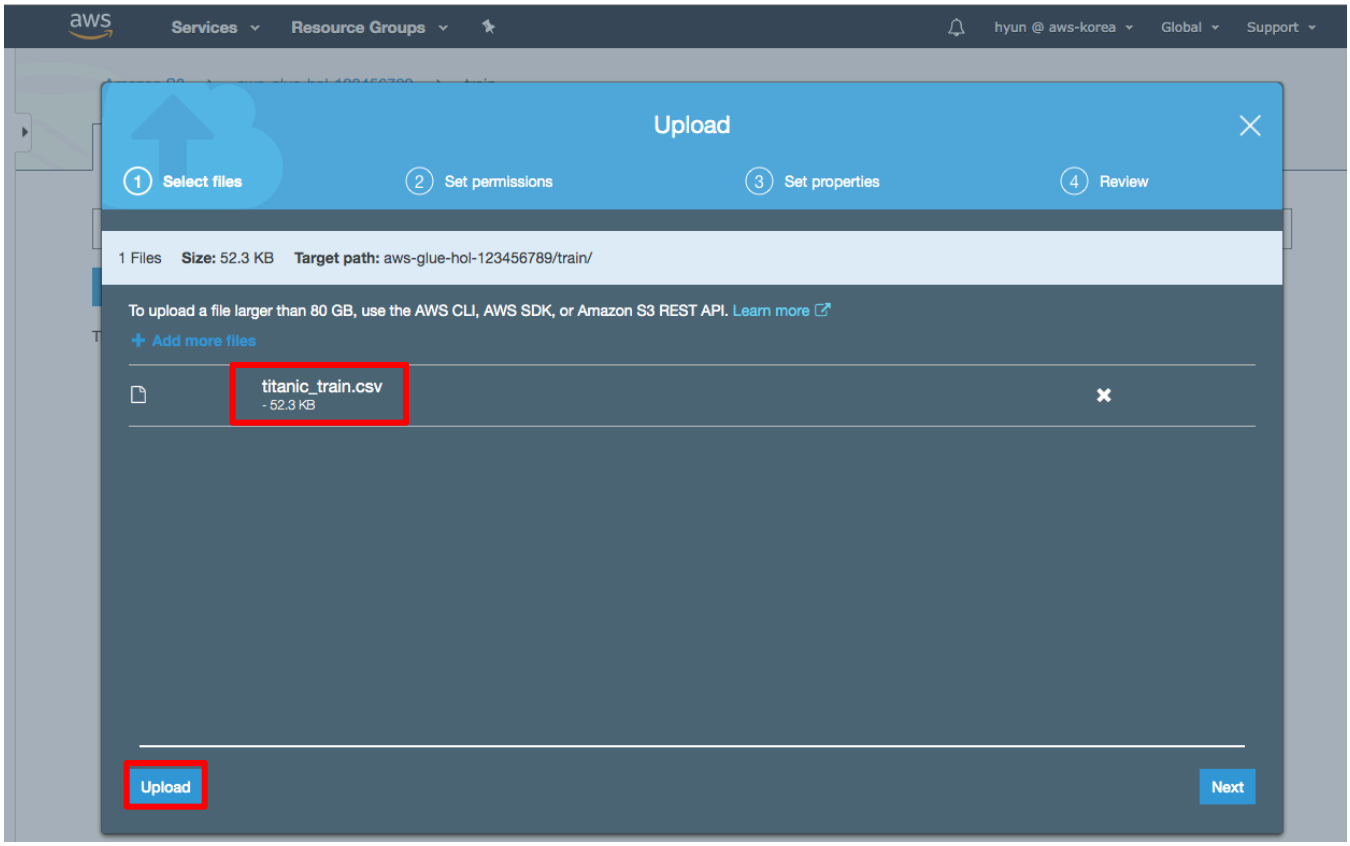
2-9. Add files 를 클릭합니다.

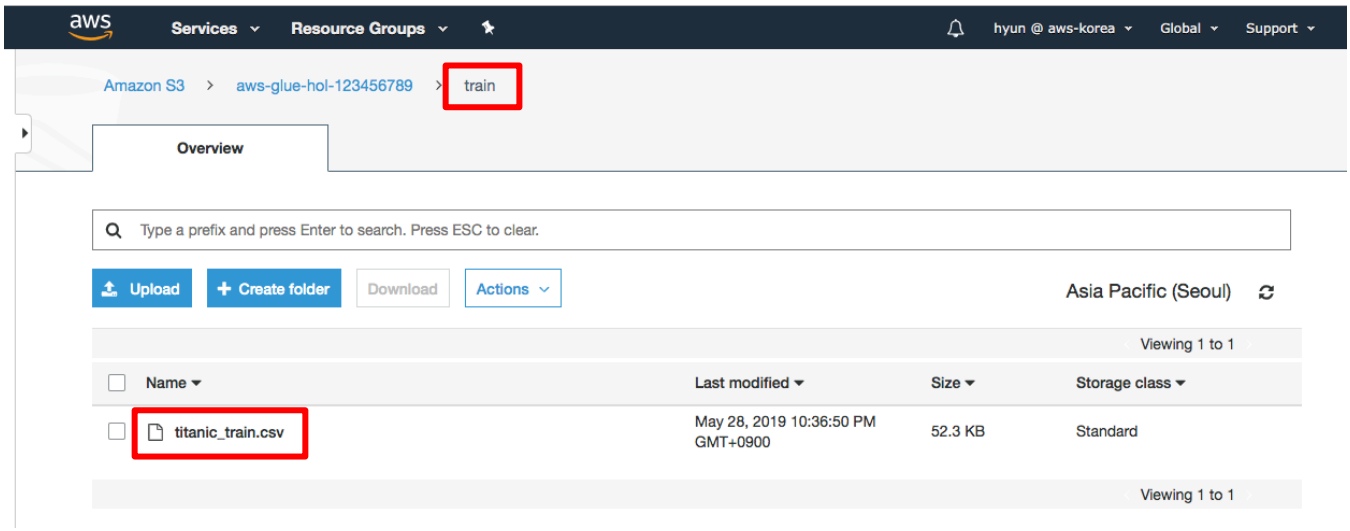


2-10. 다운로드 받은 titanic_train.csv 파일을 선택합니다.
(본 Document 는 Mac 에서 작성되어 Windows 나 Linux 의 경우 다른 형태의 파일 탐색창이 팝업됩니다.)

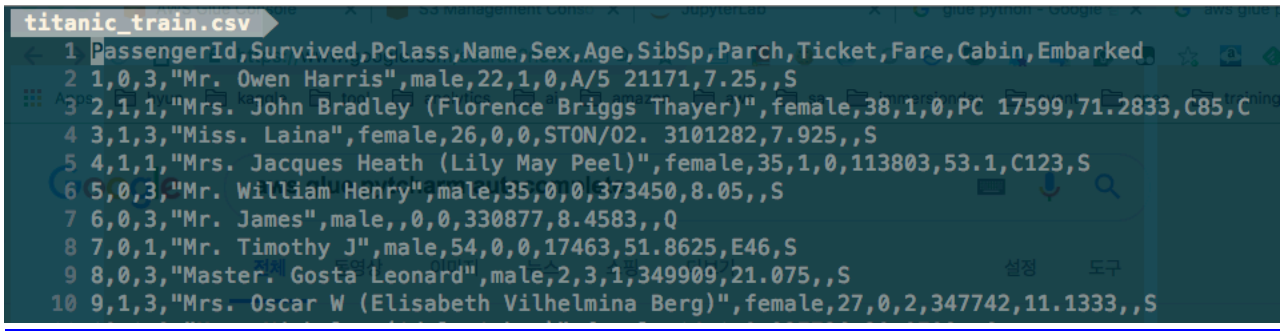


2-11. 파일 선택이 완료되면 Upload 버튼을 클릭하고 파일이 정상적으로 업로드 된 것을 확인합니다.





2-12. 참고로 다운로드 받은 titanic_train.csv 파일을 살펴보면 다음과 같이 구성되어있는 것을 확인할 수 있습니다.



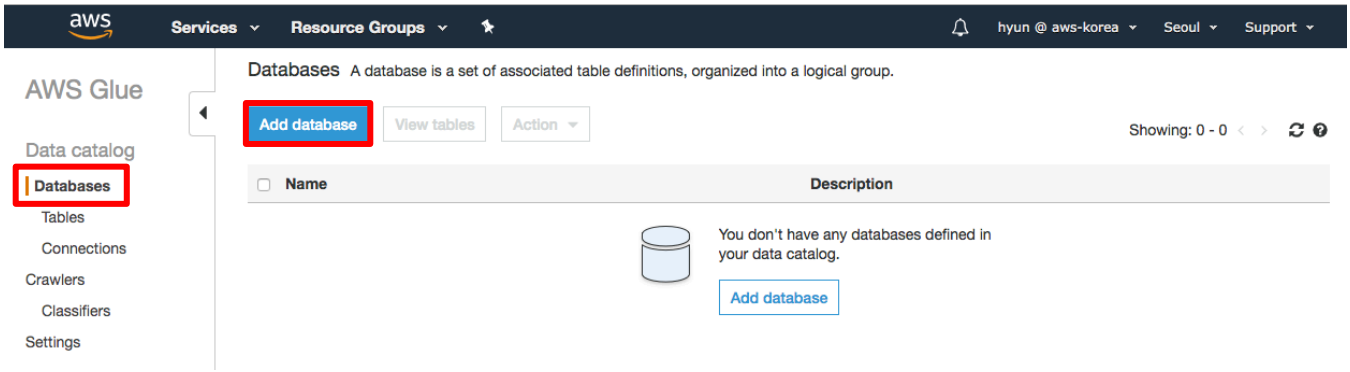
3. Create Glue Catalog and Crawling

본 단계에서는 AWS Glue 를 통해 데이터 Catalog 를 생성할 수 있도록 Crawling 작업을 구성합니다.

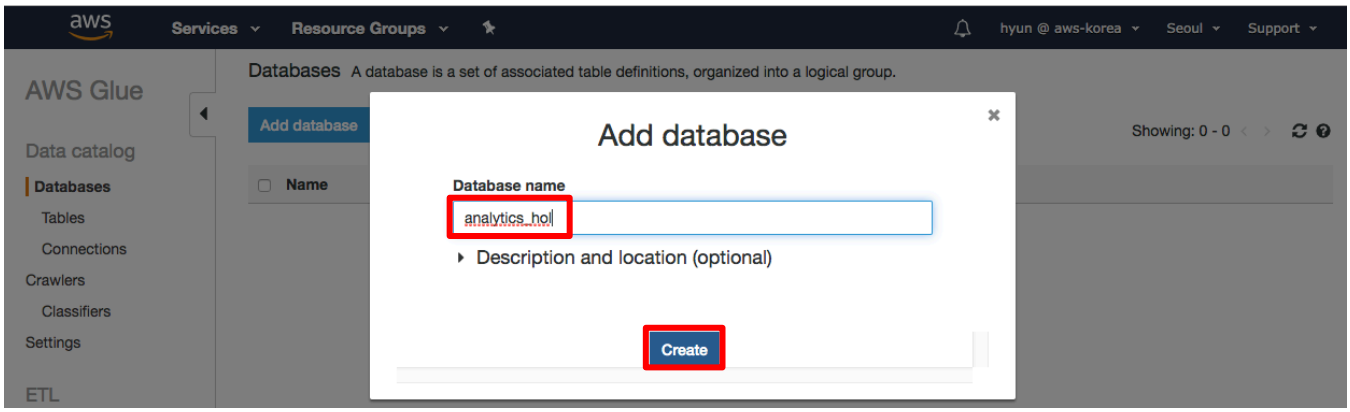
Data Catalog databases 생성

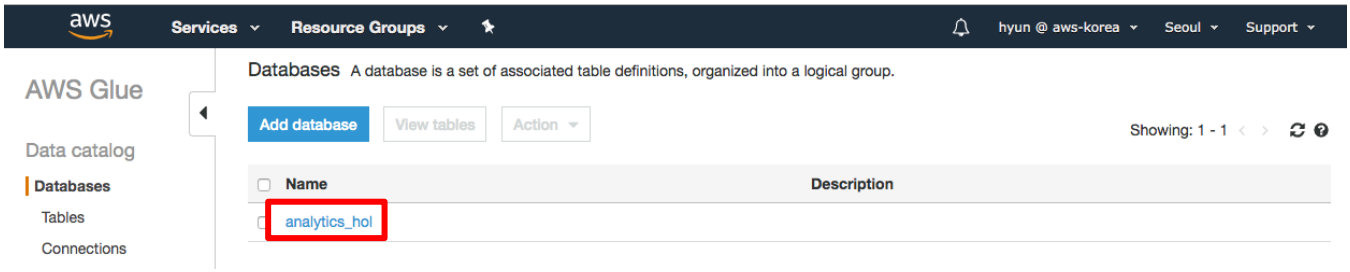
3-1. 서비스에서 AWS Glue 를 선택하여 Glue 콘솔로 이동합니다. 이전에 Glue 를 사용한 적이 없는 경우 초기 안내화면에서 Get stated 버튼을 클릭합니다.

3-2. Glue 콘솔에서 Databases 를 선택한 후 Add database 버튼을 클릭합니다.



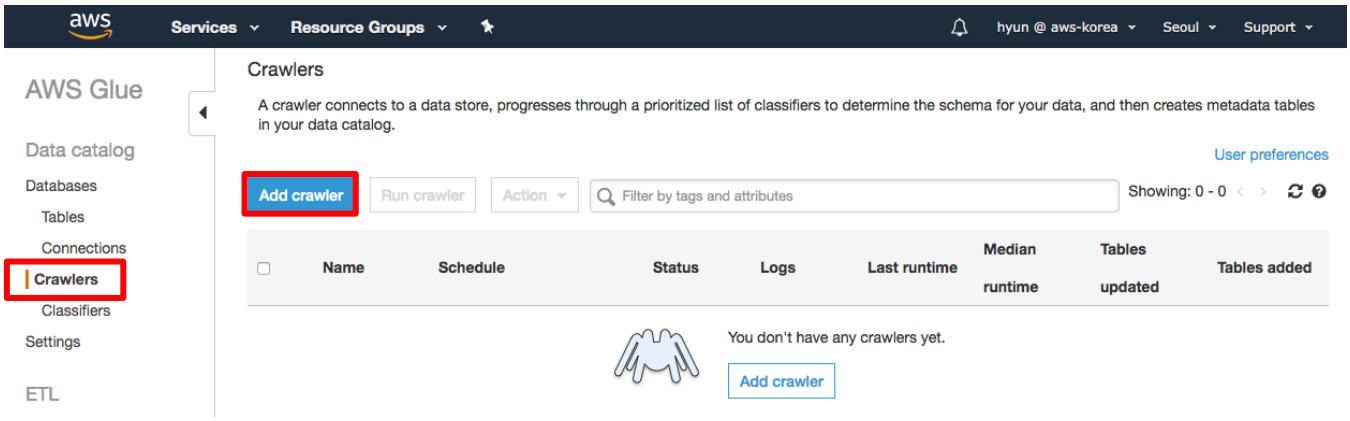
3-3. Database name 에 analytics_hol 를 입력한 후 Create 버튼을 클릭합니다.



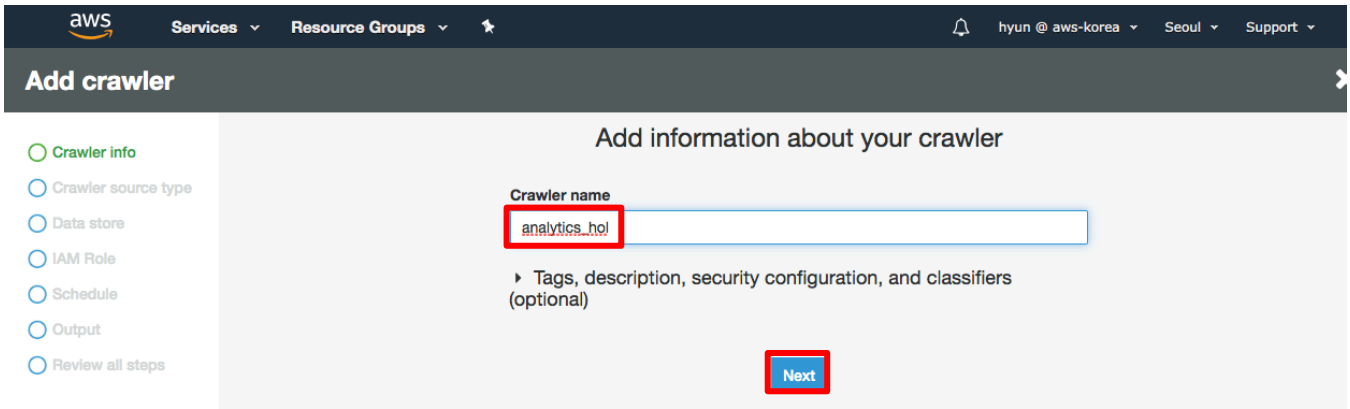


Crawler 및 Glue Catalog 생성

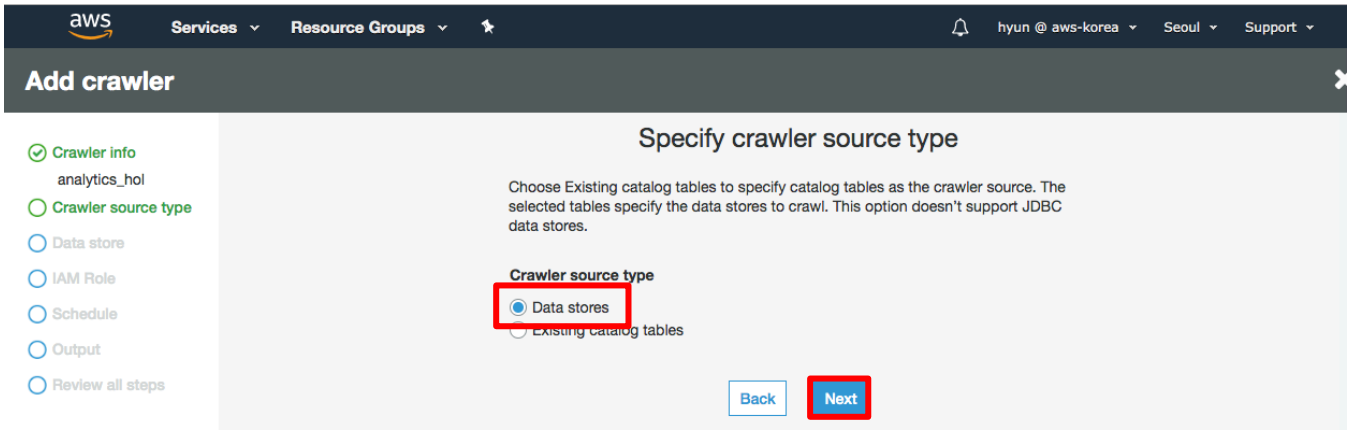
3-4. Glue 콘솔에서 Crawler 를 클릭해서 이동한 후 Add Crawler 버튼을 클릭합니다.



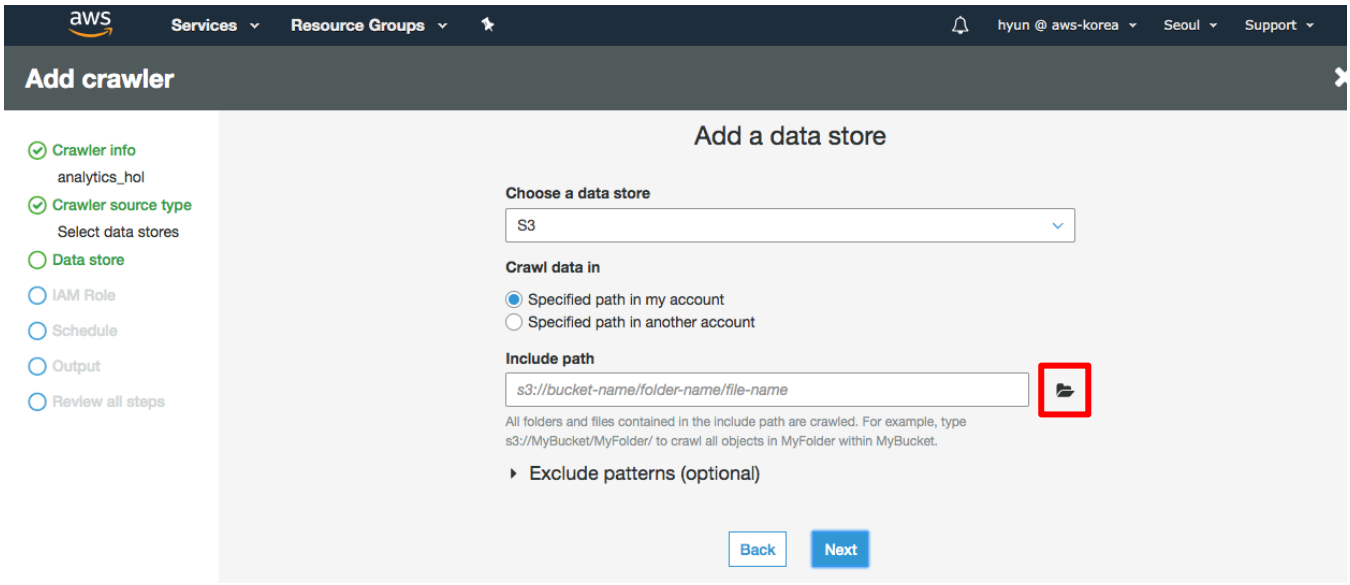
3-5. Crawler info 설정에서 Crawler name 을 analytics_hol 로 입력하고 Next 를 클릭합니다.



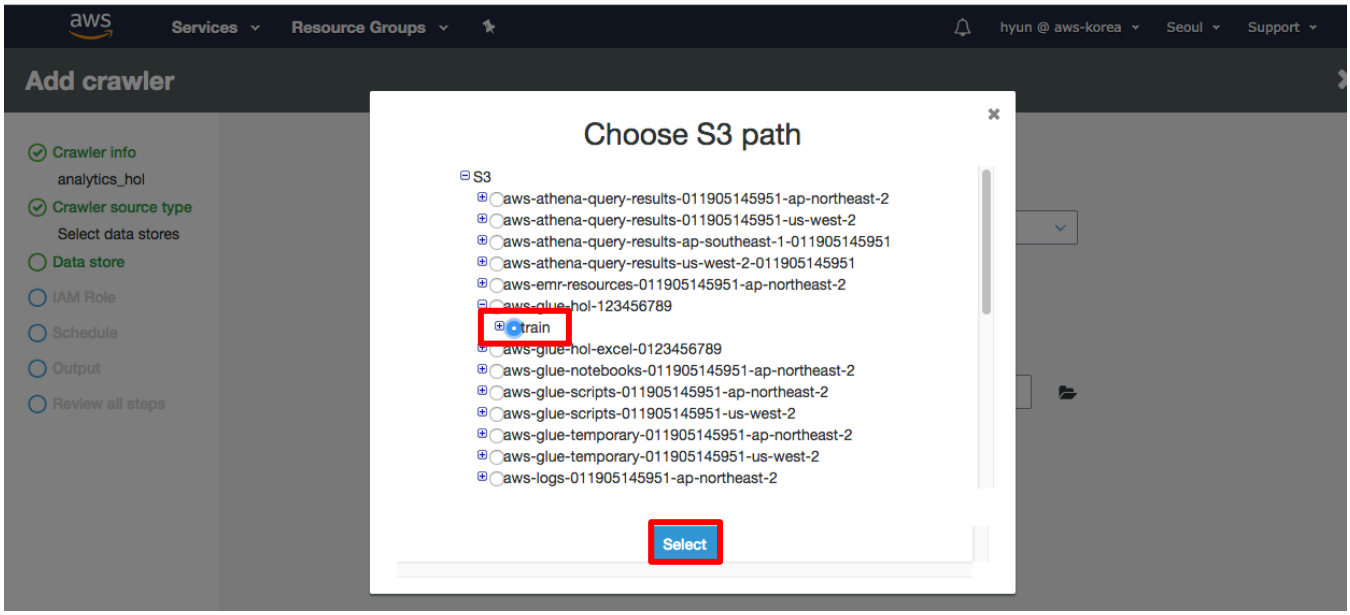
3-6. Crawler source type 설정에서 Data stores 를 선택하고 Next 를 클릭합니다.



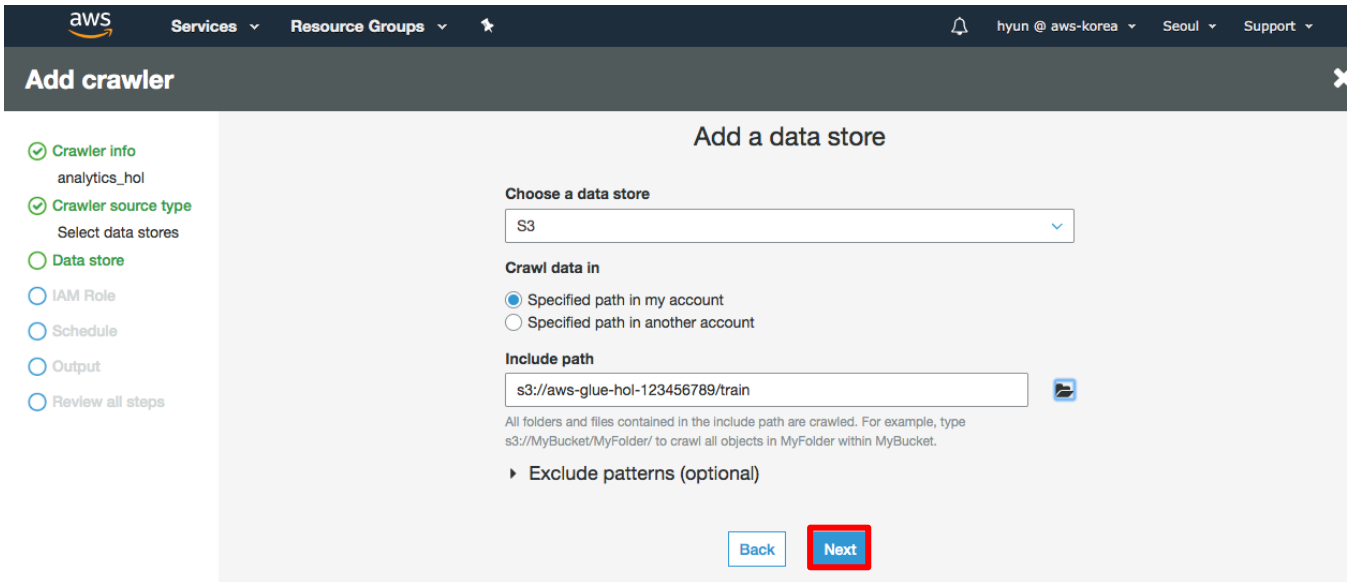
3-7. Data store 설정에서 Include path 오른쪽의 폴더 아이콘을 클릭합니다.



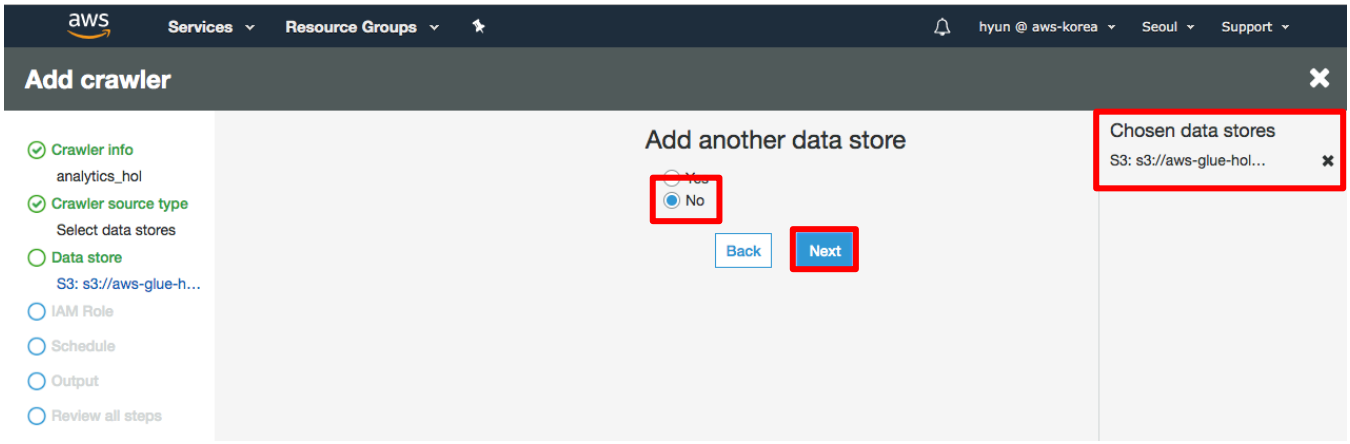
3-8. 팝업 화면에서 aws-glue-hol-[accout-id]/train 폴더를 선택하고 Select 버튼을 클릭합니다. (파일이 아닌 디렉토리를 선택합니다.)



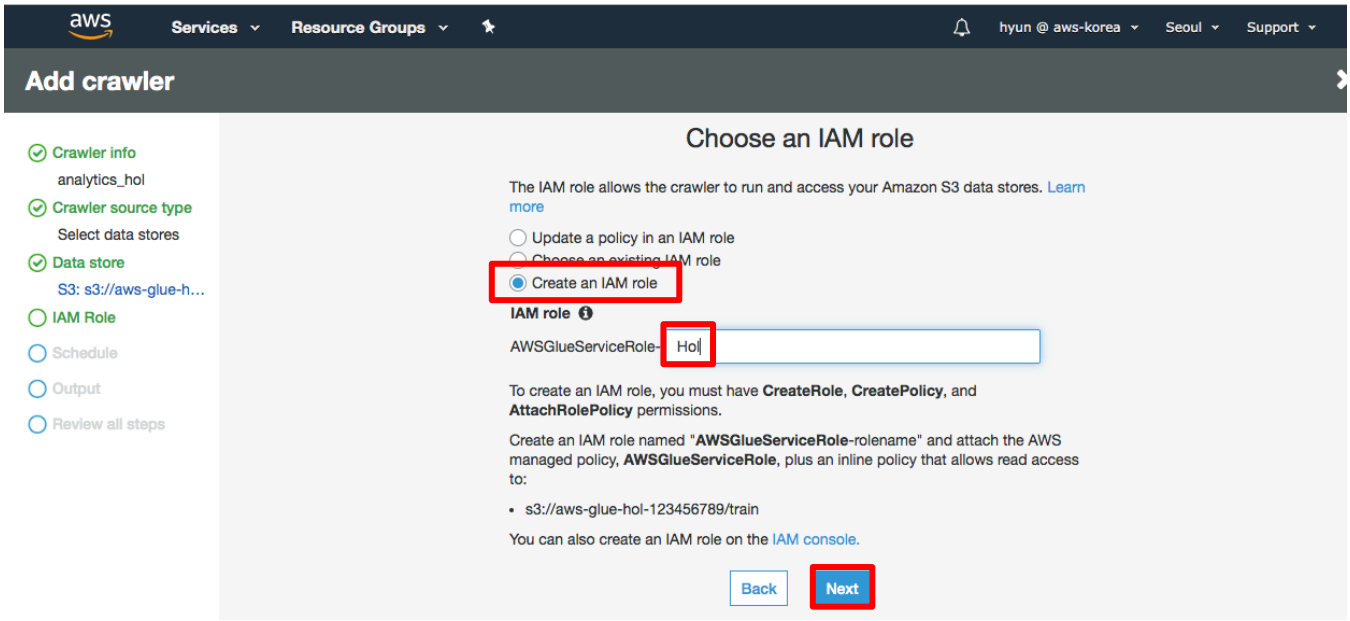
3-9. 나머지 설정을 Default 로 두고 Next 를 클릭합니다.



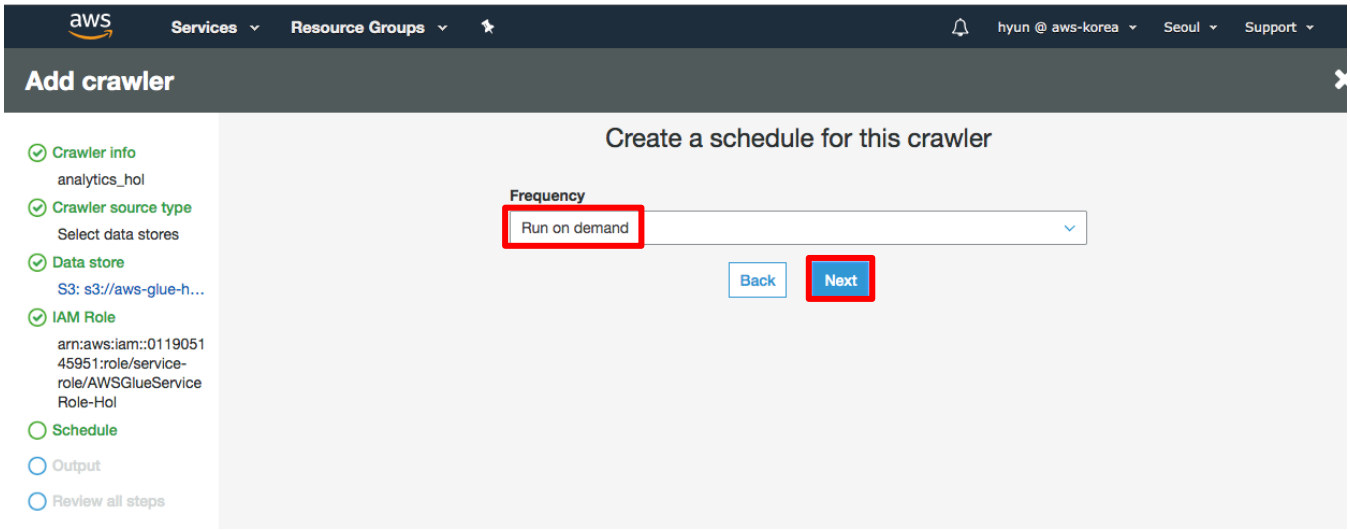
3-10. Add another data store 화면에서 No 를 선택된 상태로 Next 를 클릭합니다. 오른쪽 Chosen data stores 에 앞서 선택한 파일이 표시되는 것을 확인할 수 있습니다.



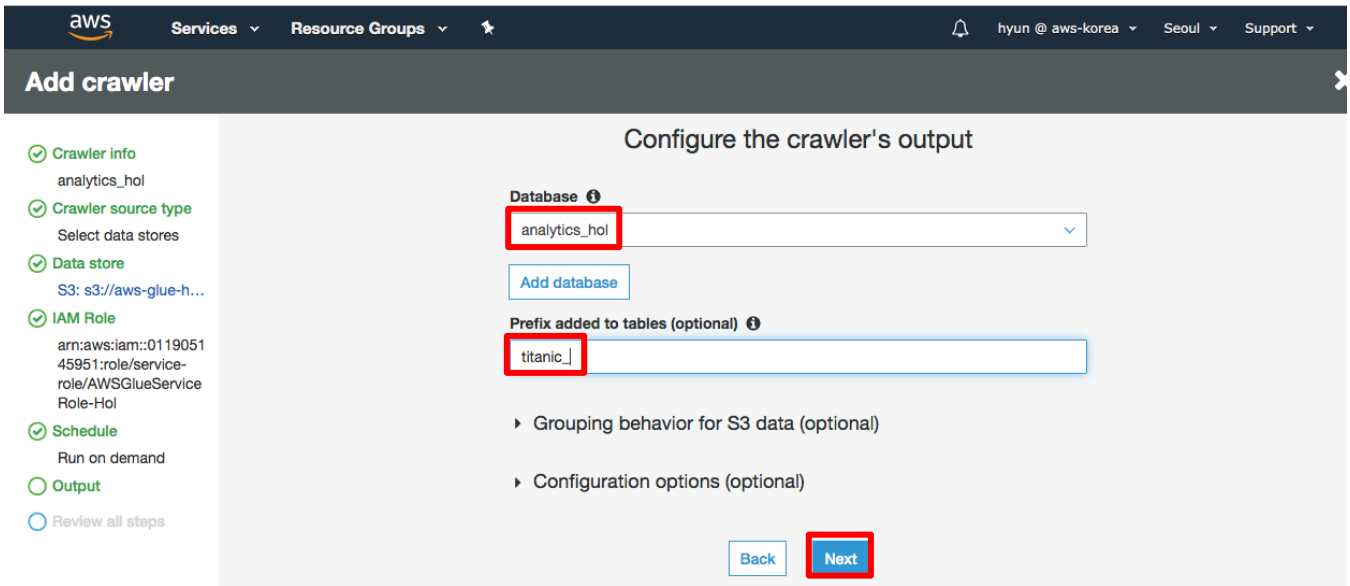
3-11. IAM Role 설정에서 Create an IAM role 을 선택합니다. IAM role 에 Hol 을 입력하고 Next 버튼을 클릭합니다.



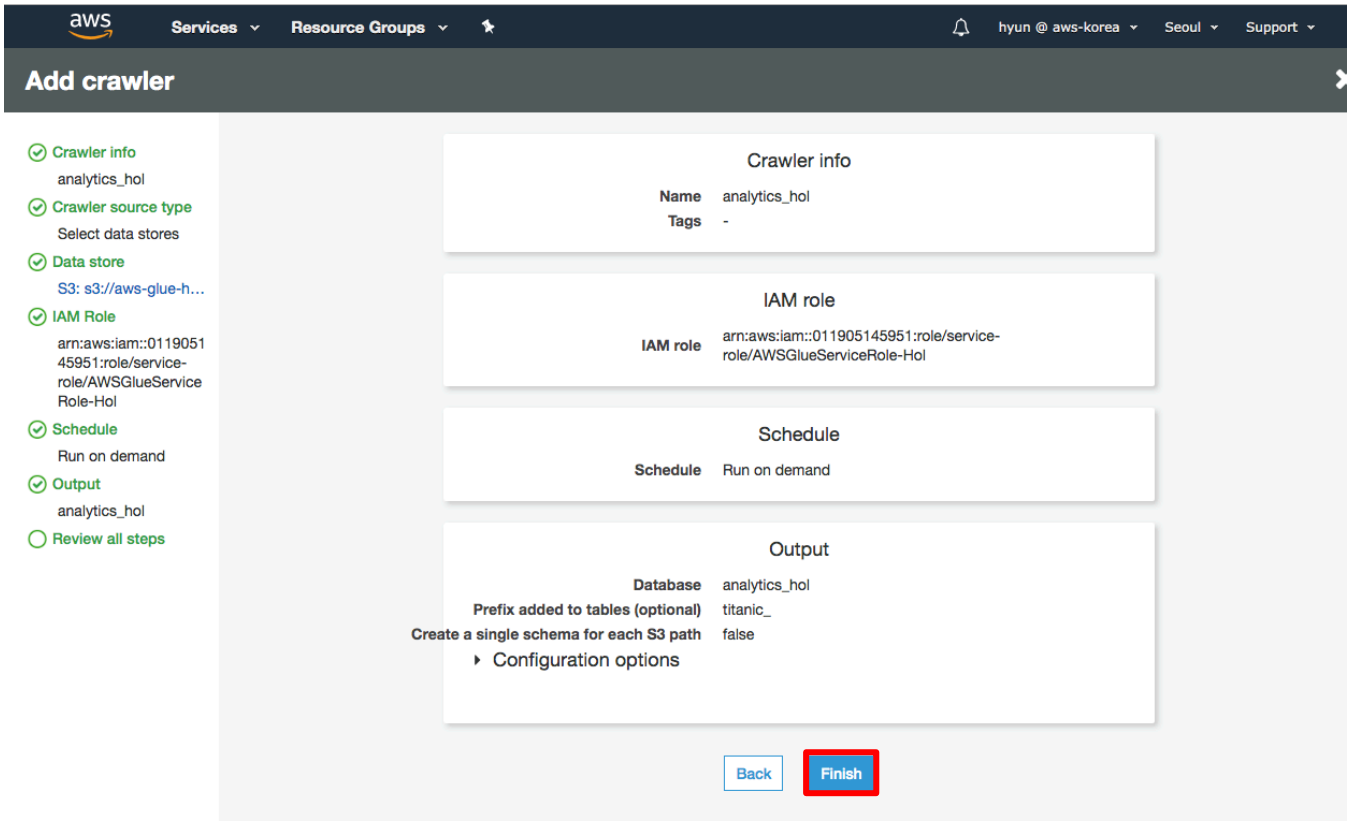
3-12. Schedule 설정은 Run on demand 상태로 두고 Next 를 클릭합니다.

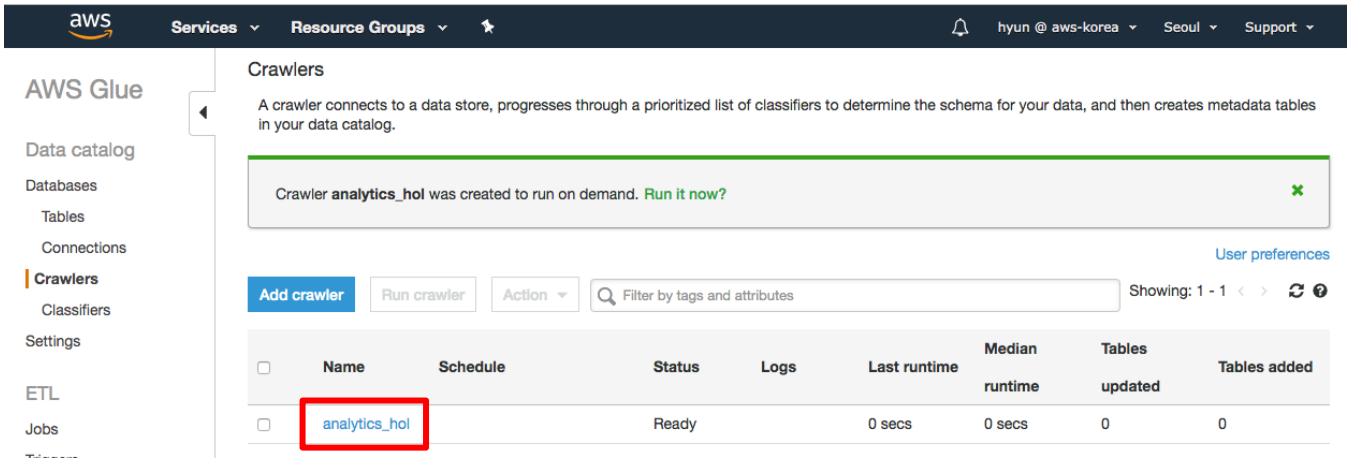


3-13. Output 설정에서는 Output Database 를 analytics_hol 로 선택합니다. 생성할 table 에 prefix 를 붙이기 위해 Prefix added to tables[optinal]에 titanic_ 를 입력하고 Next 버튼을 클릭합니다.

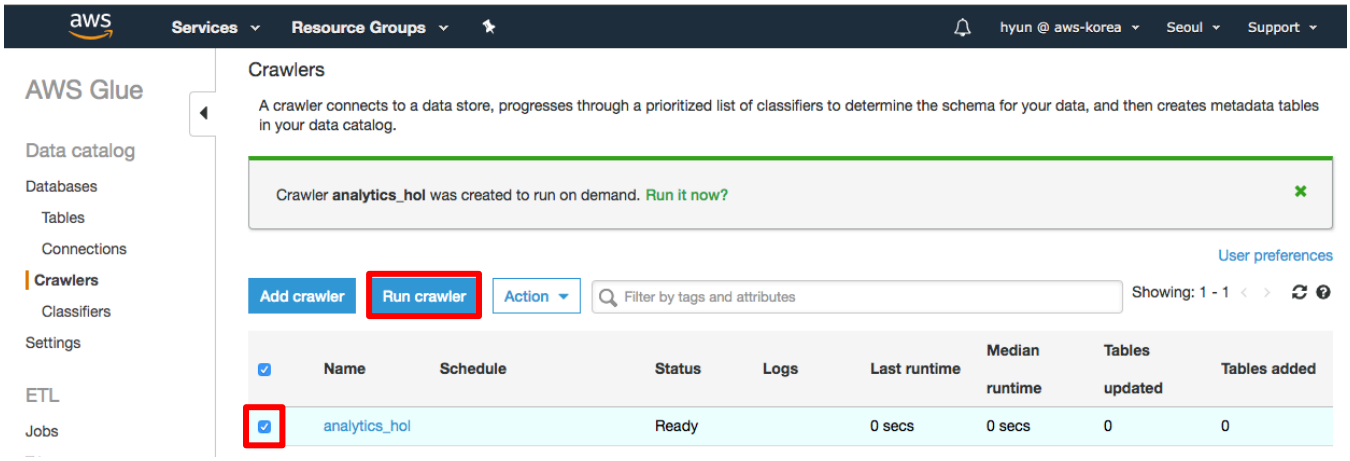


3-14. 내용을 Review 한 후 Finish 버튼을 클릭하면 첫번째 Crawler Job 이 생성됩니다.

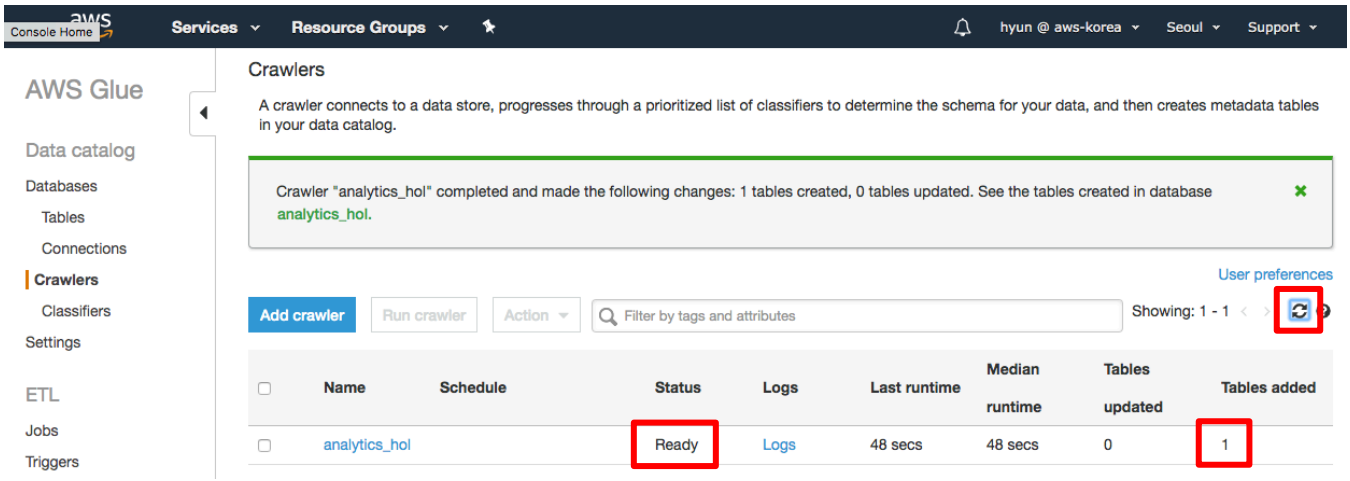




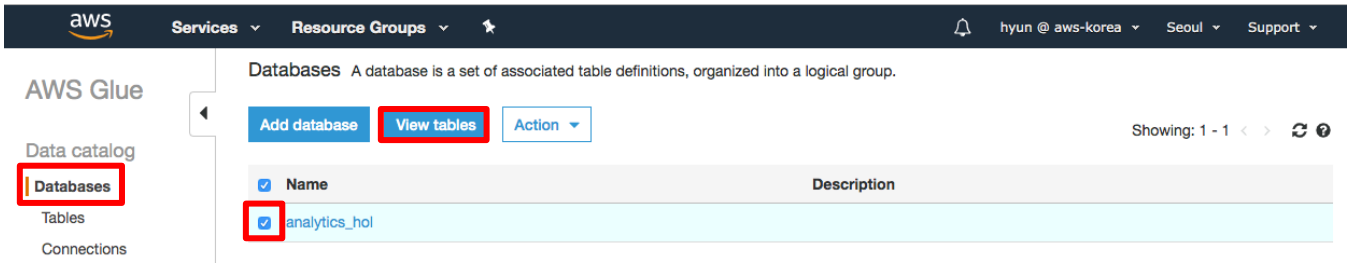
3-15. Job 체크박스를 선택한 후 Run crawler 를 클릭하여 생성한 Crawler 를 실행합니다.



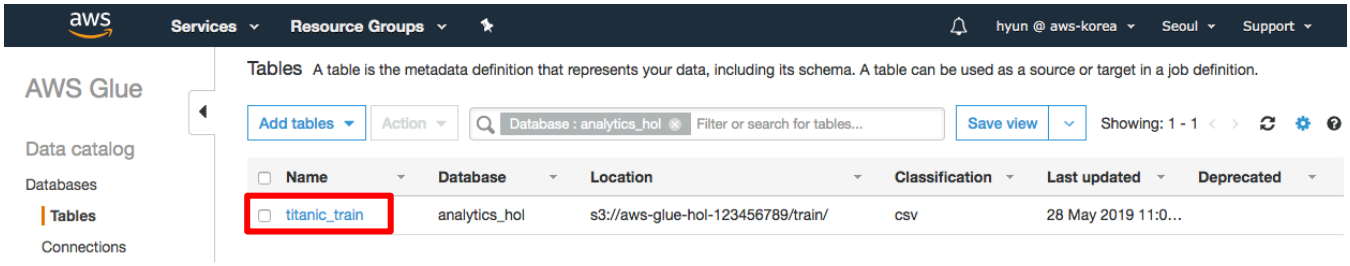
3-16. Crawler Job 이 완료되면 Ready 상태가 되고, 맨 오른쪽 컬럼에서 1 개의 Table 이 생성된 것을 확인할 수 있습니다. (Refresh 버튼으로 상태 변경을 확인할 수 있습니다.)



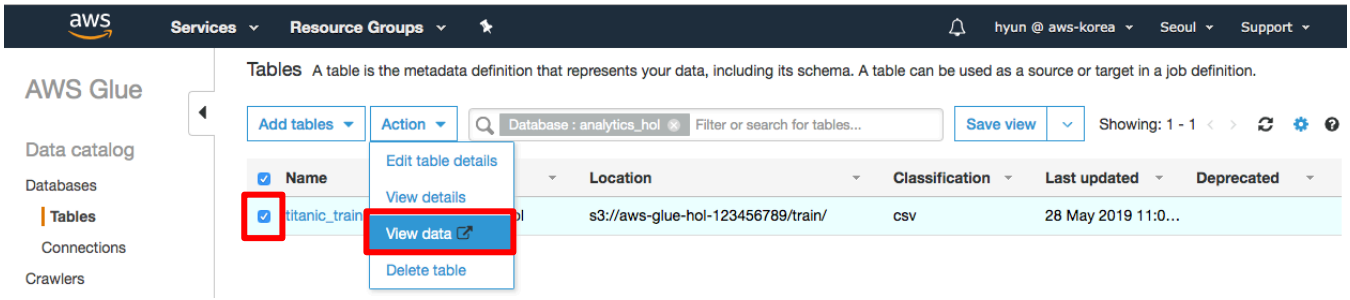
3-17. Databases 메뉴로 이동합니다. analytics_hol database 를 선택하고 View tables 버튼을 클릭합니다.

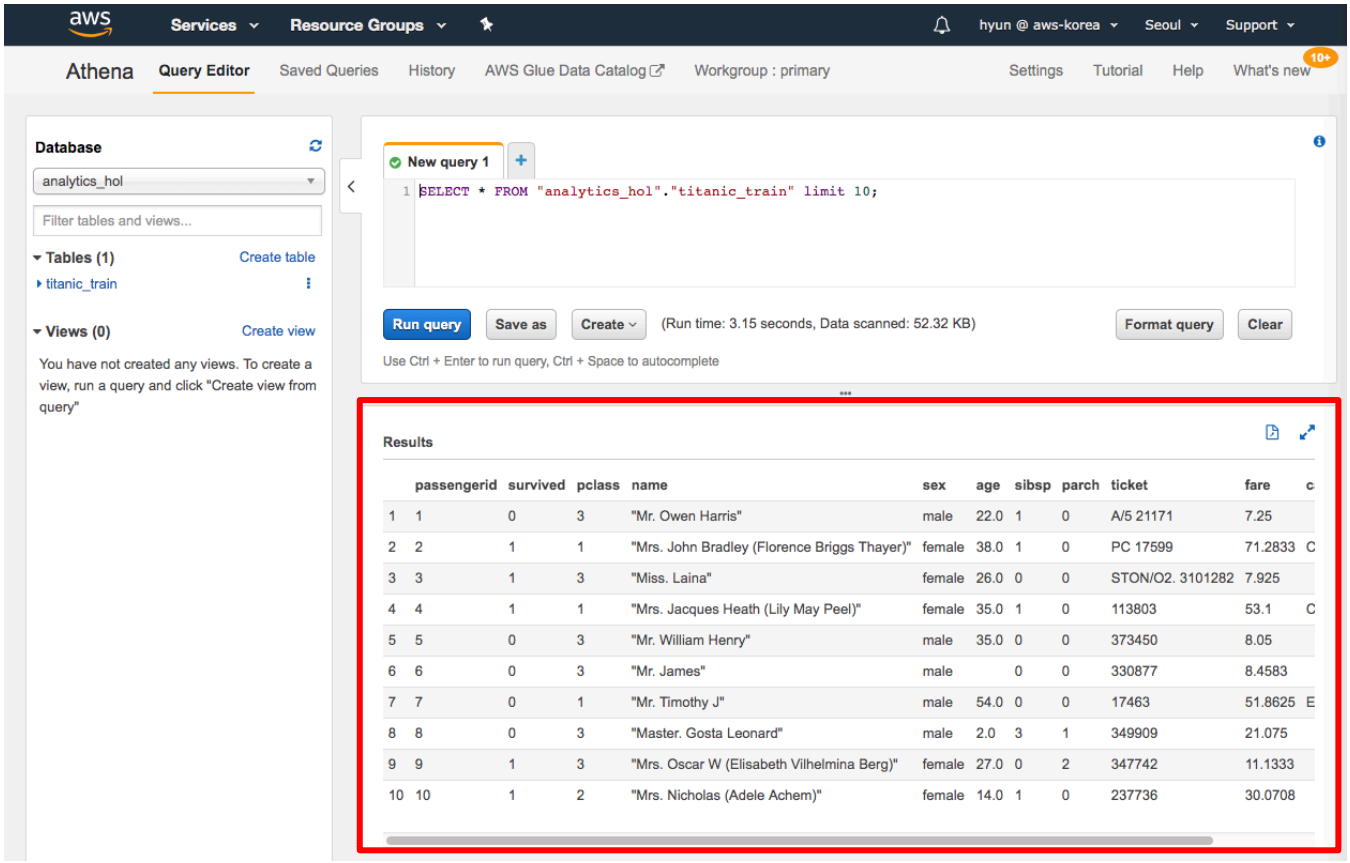


3-18. titanic_train 테이블이 생성된 것을 확인합니다.



3-19. titanic_train 테이블을 선택하고 Action 에서 View data 메뉴를 선택하면 Amazon Athena 에서 Sample 데이터가 표시되는 것을 확인합니다. Athena 를 처음 실행하는 경우 Get Started 버튼을 클릭하면 Sample 데이터가 표시됩니다.





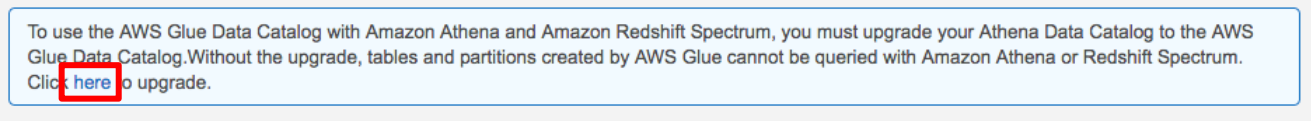
	passengerid	survived	pclass	name	sex	age	sibsp	parch	ticket	fare	c
1	1	0	3	"Mr. Owen Harris"	male	22.0	1	0	A/5 21171	7.25	
2	2	1	1	"Mrs. John Bradley (Florence Briggs Thayer)"	female	38.0	1	0	PC 17599	71.2833	C
3	3	1	3	"Miss. Laina"	female	26.0	0	0	STON/O2. 3101282	7.925	
4	4	1	1	"Mrs. Jacques Heath (Lily May Peel)"	female	35.0	1	0	113803	53.1	C
5	5	0	3	"Mr. William Henry"	male	35.0	0	0	373450	8.05	
6	6	0	3	"Mr. James"	male		0	0	330877	8.4583	
7	7	0	1	"Mr. Timothy J"	male	54.0	0	0	17463	51.8625	E
8	8	0	3	"Master. Gosta Leonard"	male	2.0	3	1	349909	21.075	
9	9	1	3	"Mrs. Oscar W (Elisabeth Vilhelmina Berg)"	female	27.0	0	2	347742	11.1333	
10	10	1	2	"Mrs. Nicholas (Adele Achem)"	female	14.0	1	0	237736	30.0708	

4. Exploring Data Analysis with Athena

2 장, 3 장에서 준비된 데이터는 Kaggle 에서 입문 tutorial 로 사용되는 titanic competition dataset 입니다. 이번 장에서는 Amazon Athena 와 Glue Catalog 를 이용하여 해당 데이터를 분석하는 작업을 진행하겠습니다.

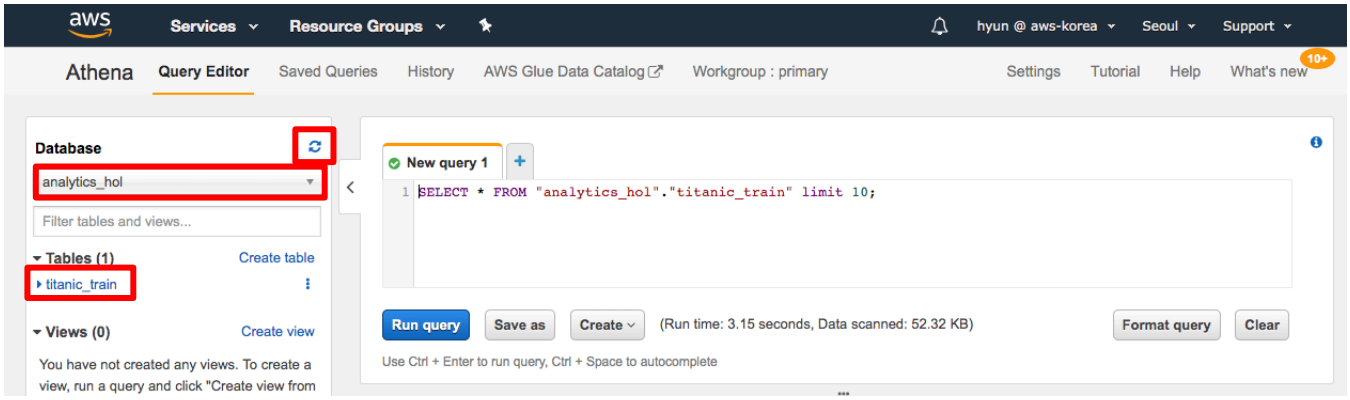
GLUE 카탈로그로 업그레이드

4-1. AWS 콘솔에서 Athena 서비스를 선택합니다. Database 에 Glue 카탈로그가 보이는지 확인합니다. 리전에 따라 Athena 에서 Glue 카탈로그를 이용하기 위해서는 카탈로그 통합 작업이 필요할 수 있습니다. 아래 메시지가 오픈되는 경우 Click here to upgrade 에서 here 링크를 클릭합니다. Database 에 카탈로그가 보이는 경우는 이미 통합이 완료된 것이므로 다음 단계로 넘어갑니다. 카탈로그 통합과 관련한 자세한 내용은 다음 링크를 참고할 수 있습니다.
https://docs.aws.amazon.com/athena/latest/ug/glue-athena.html



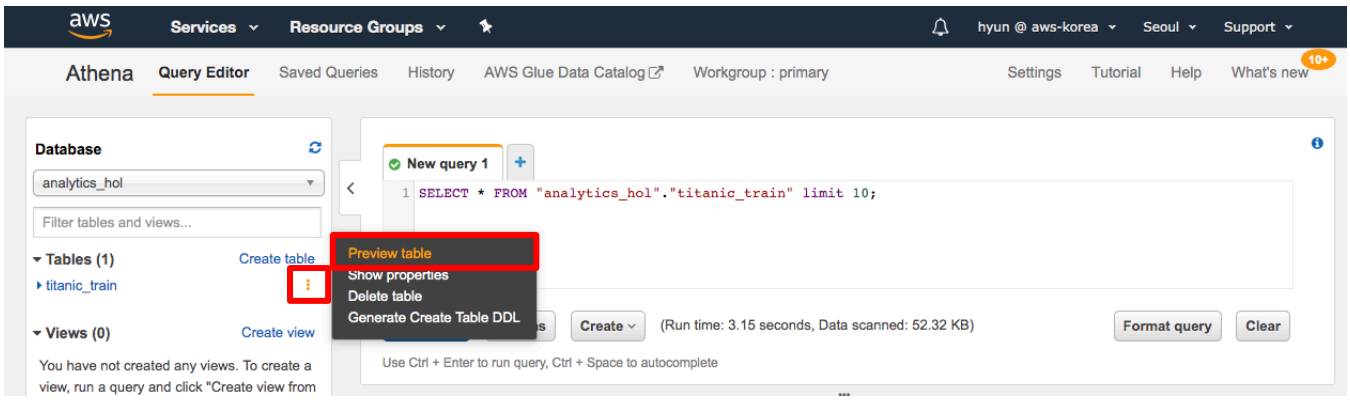
IAM 정책 업데이트에 대한 안내 페이지가 오픈됩니다. 만약 기존에 Athena 와 Redshift Spectrum 을 사용하면서 사용자 정의 정책을 정의한 경우 카탈로그 업데이트에 따라 권한 추가가 필요하다는 안내입니다. 기존에 Athena 와 Redshift Spectrum 을 사용하지 않았을 경우에는 바로 업그레이드를 실행할 수 있습니다. 파란색 Upgrade 버튼을 클릭합니다.

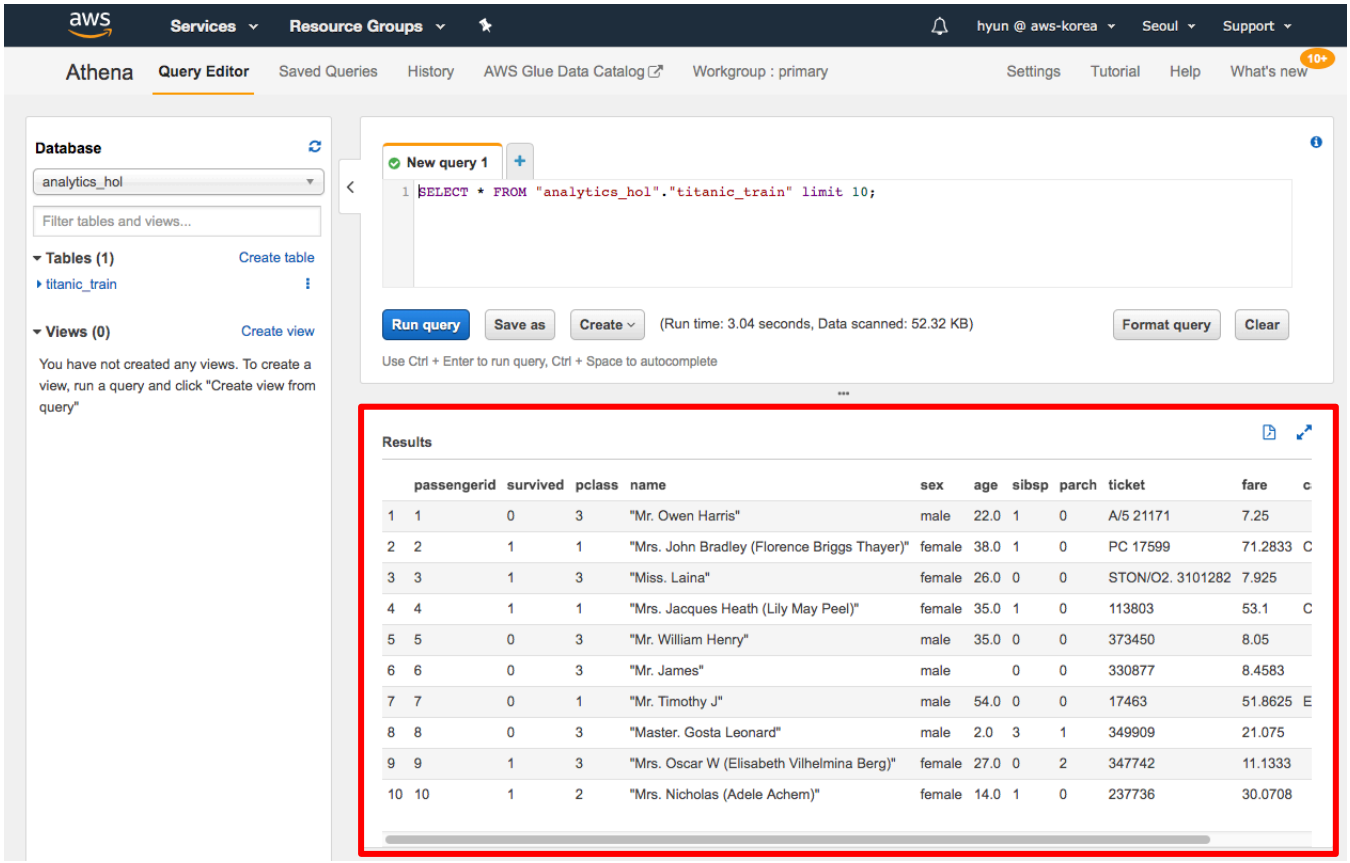
업그레이드가 완료되면 왼쪽 Database 네비게이션에서 analytics_hol 데이터베이스를 선택할 수 있고 Crawling 작업을 통해 생성한 titanic_train 테이블을 볼 수 있습니다. 데이터베이스나 테이블이 보이지 않을 경우 refresh 버튼을 클릭하세요.



S3 data query with Athena

4-2. 테이블명 옆 ⋮을 클릭하고 Preview table 을 클릭하면 샘플 쿼리가 생성되고 실행됩니다. 이제 S3 에 저장된 데이터를 sql 문을 통해 조회할 수 있습니다.



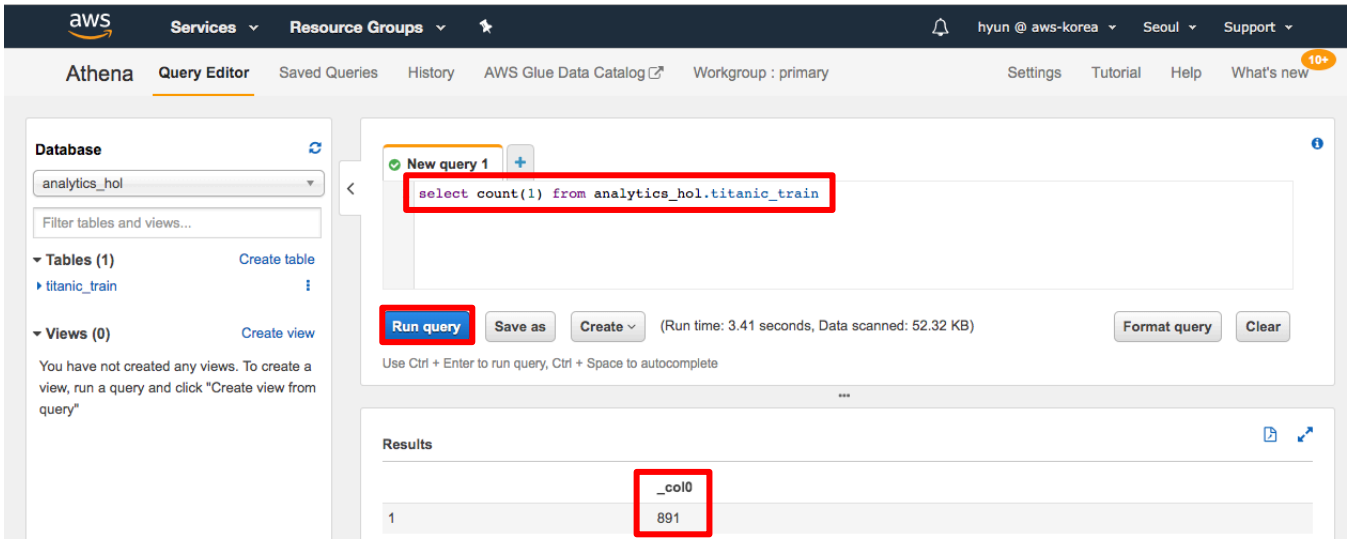


4-3. 샘플 데이터를 살펴보면 데이터는 다음과 같은 Column 들로 구성되어 있는 것을 확인할 수 있습니다.

- passengerid: 승객 id
- survived: 생존여부 (0=No, 1=Yes)
- pclass: 티켓등급 (1=1st, 2=2nd, 3rd)
- name: 이름
- sex: 성별 (male, female)
- age: 나이
- slibsp: 함께 탑승한 형제자매, 배우자의 수
- parch: 함께 탑승한 부모, 자식의 수
- ticket: 티켓번호
- fare: 운임
- cabin: 객실 번호
- embarked: 탑승항구 (C=Cherbourg, Q=Queenstown, S=Southampton)

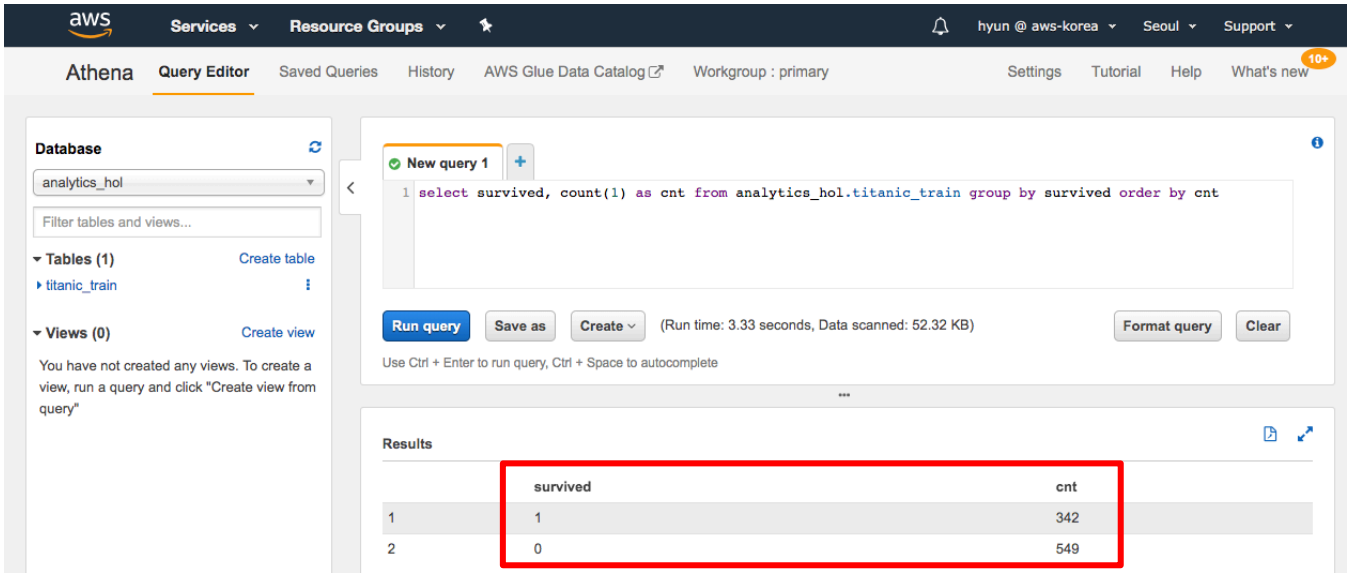
4-4. 전체 데이터 수 확인

select count(1) from analytics_hol.titanic_train



4.5. 탑승객 생존율 확인

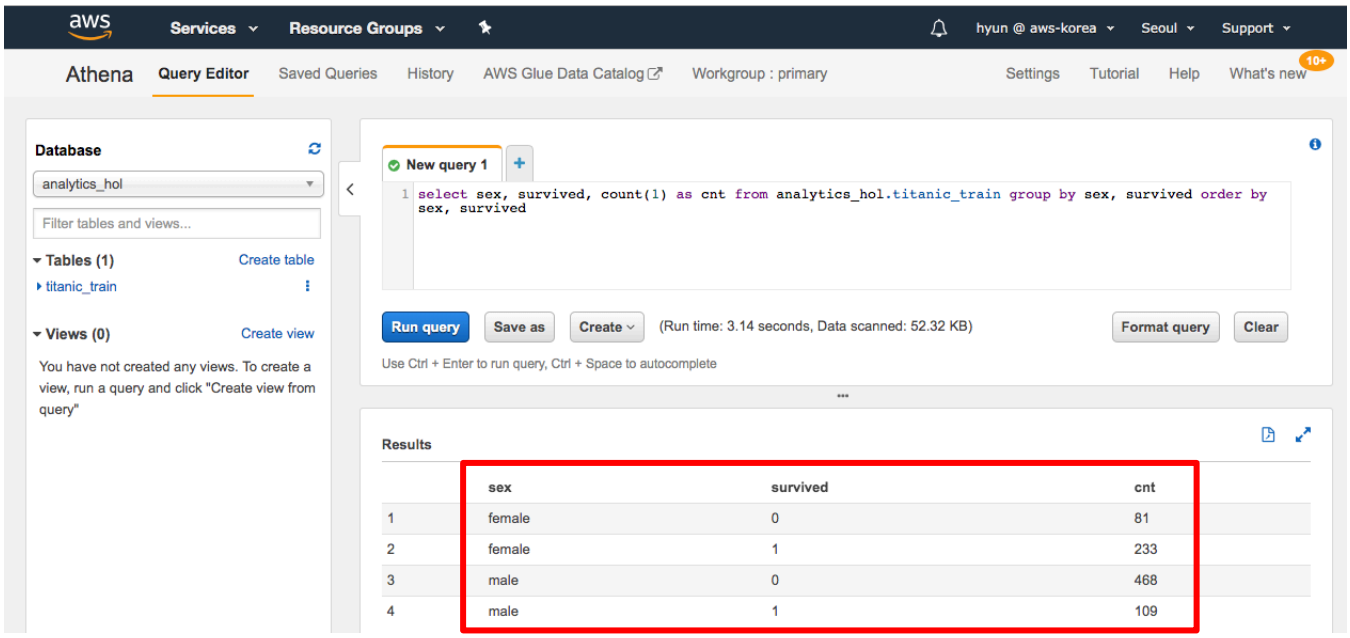
select survived, count(1) as cnt from analytics_hol.titanic_train group by survived order by cnt



891 명의 승객 중 342 명만 생존했음을 알 수 있습니다.

4.6. 성별에 따른 생존율 확인

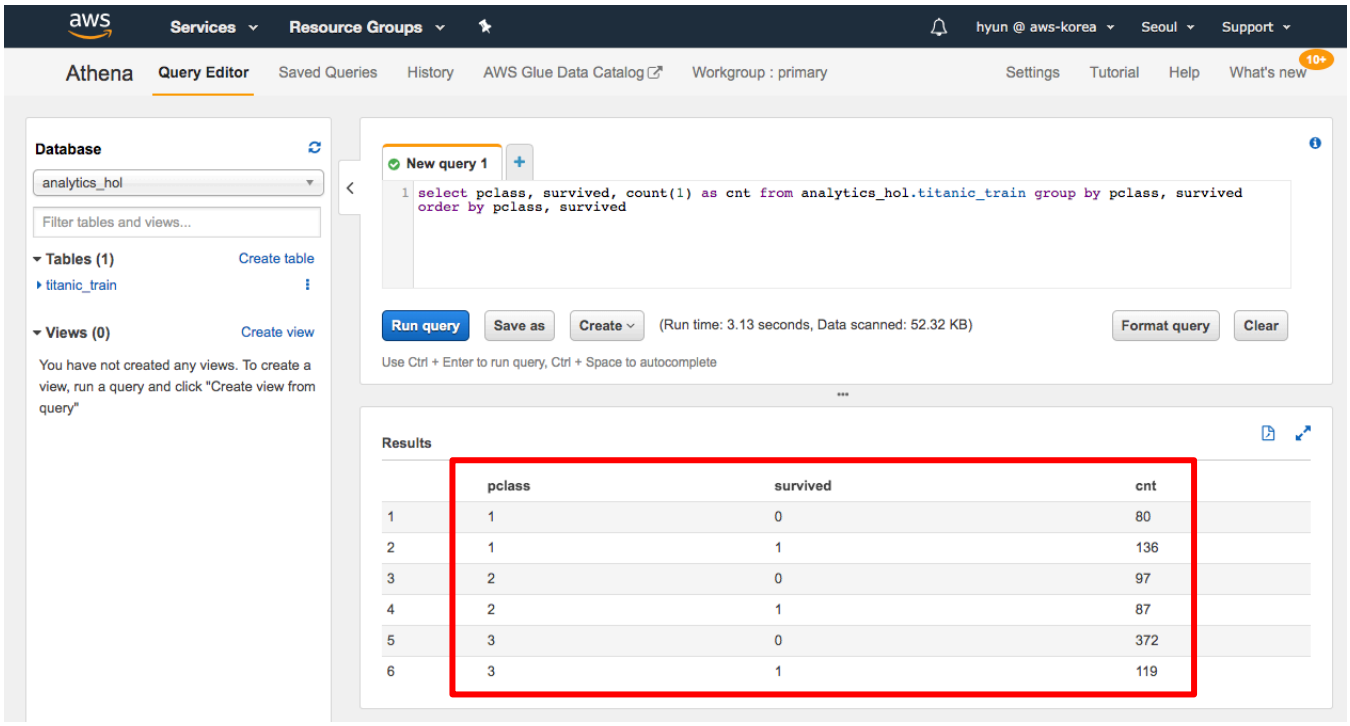
select sex, survived, count(1) as cnt from analytics_hol.titanic_train group by sex, survived order by sex, survived



남성보다 여성의 생존율이 높음을 알 수 있습니다.

4.7. 티켓등급에 따른 생존율 확인

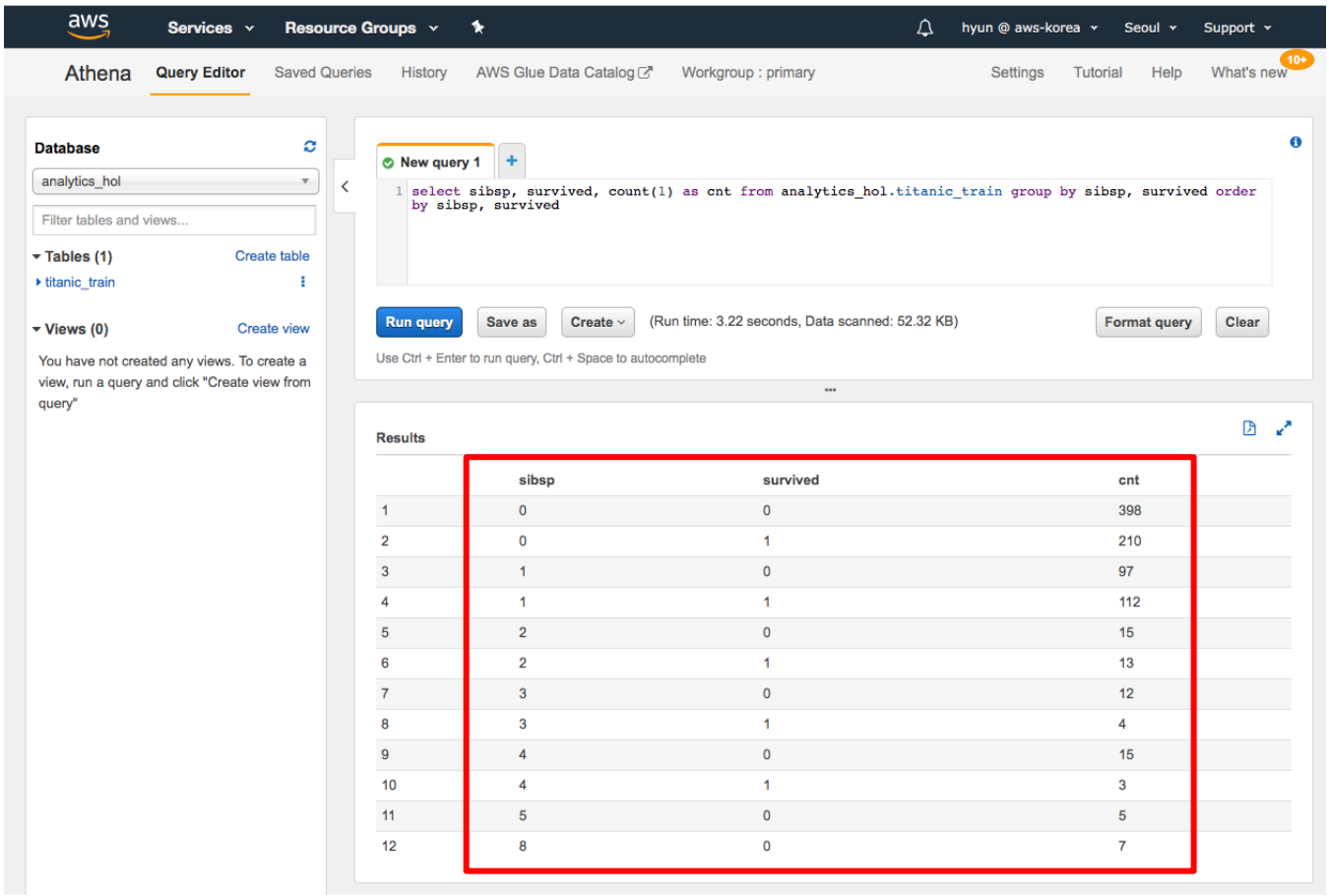
select pclass, survived, count(1) as cnt from analytics_hol.titanic_train group by pclass, survived order by pclass, survived



1 등석의 생존율은 높고, 3 등석의 생존율은 낮은 것을 알 수 있습니다.

4.8. 함께 탑승한 형제자매나 배우자에 따른 생존율 확인

select sibsp, survived, count(1) as cnt from analytics_hol.titanic_train group by sibsp, survived order by sibsp, survived

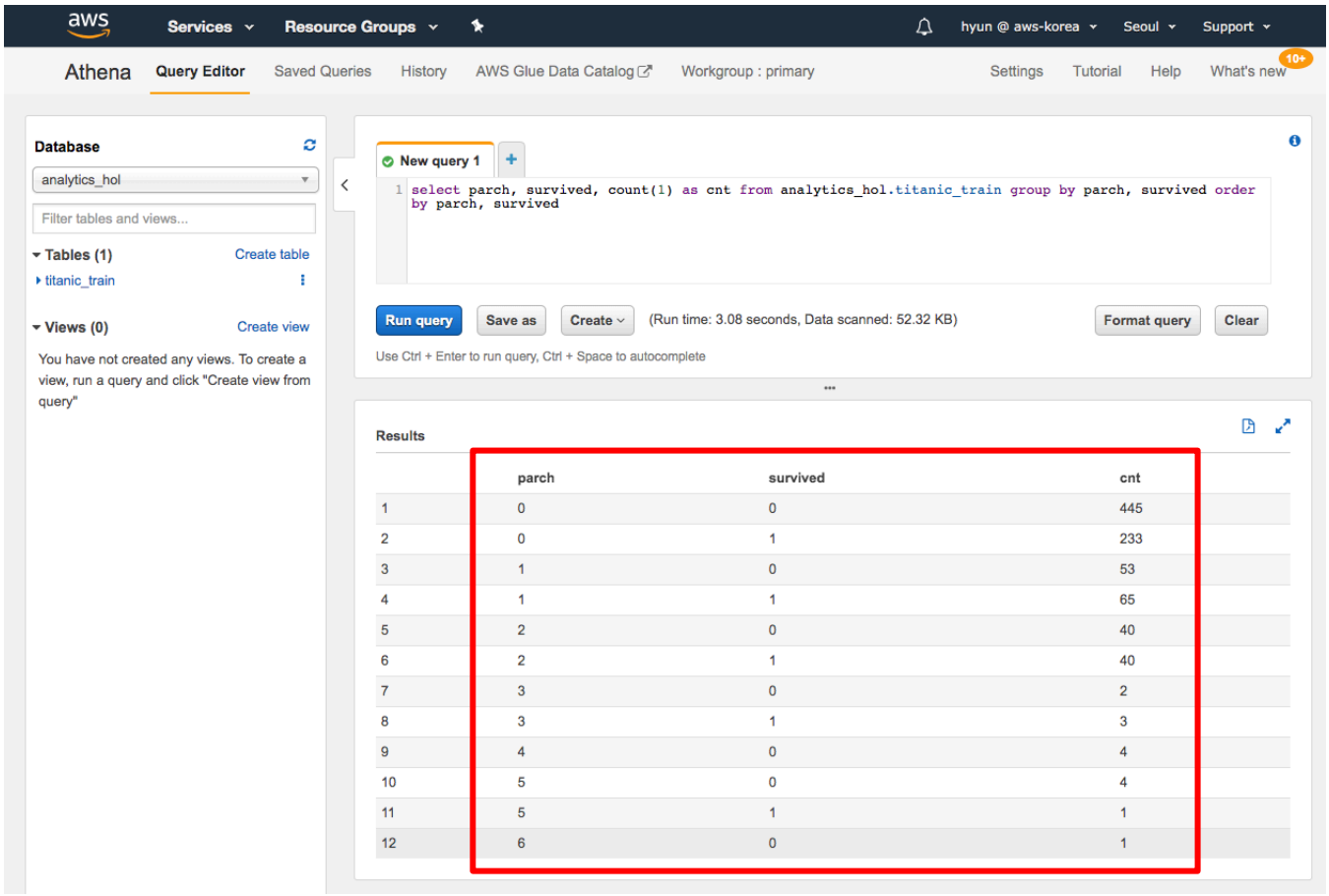


	sibsp	survived	cnt
1	0	0	398
2	0	1	210
3	1	0	97
4	1	1	112
5	2	0	15
6	2	1	13
7	3	0	12
8	3	1	4
9	4	0	15
10	4	1	3
11	5	0	5
12	8	0	7

동승한 형제자매나 배우자가 없는 경우 생존율이 낮은 것을 알 수 있습니다.

4.9. 함께 탑승한 부모님이나 자녀에 따른 생존율 확인

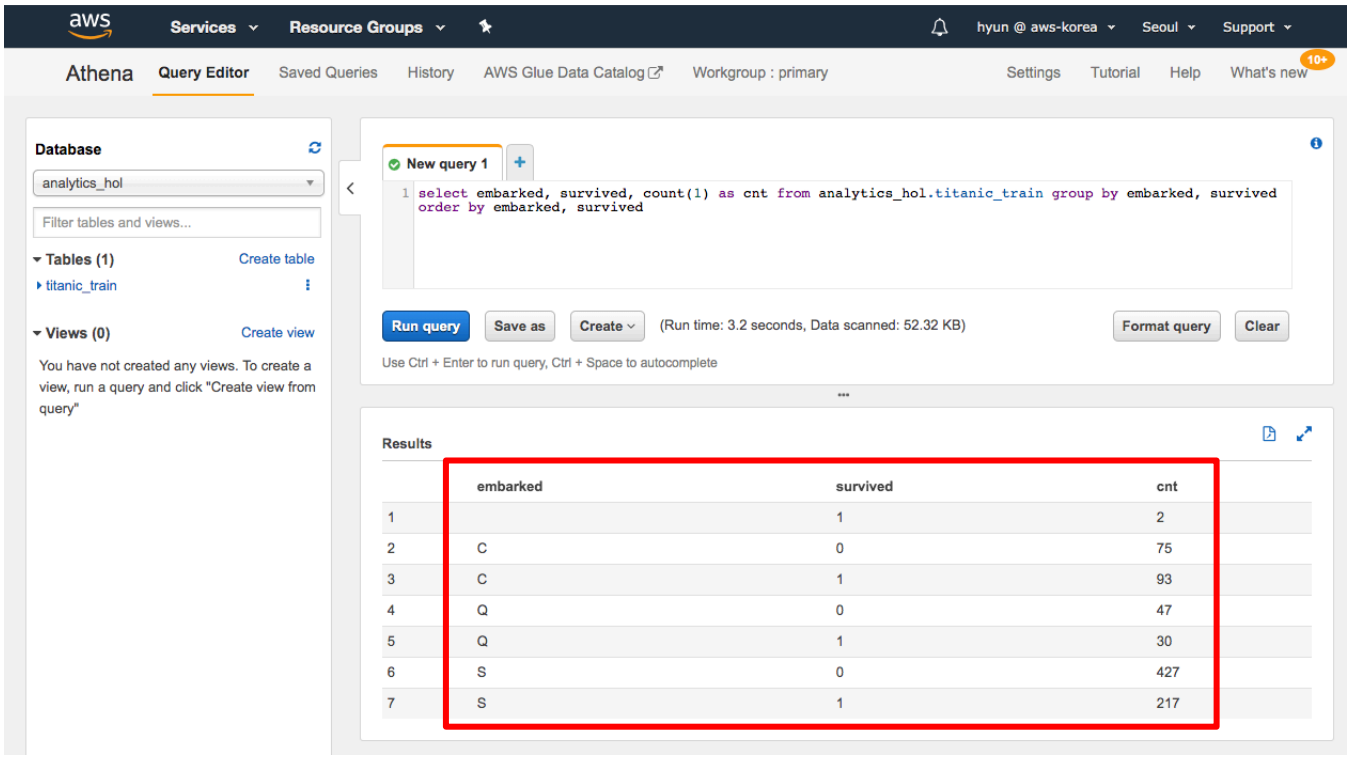
select parch, survived, count(1) as cnt from analytics_hol.titanic_train group by parch, survived order by parch, survived



동승한 부모나 자녀가 없는 경우 생존율이 낮은 것을 알 수 있습니다.

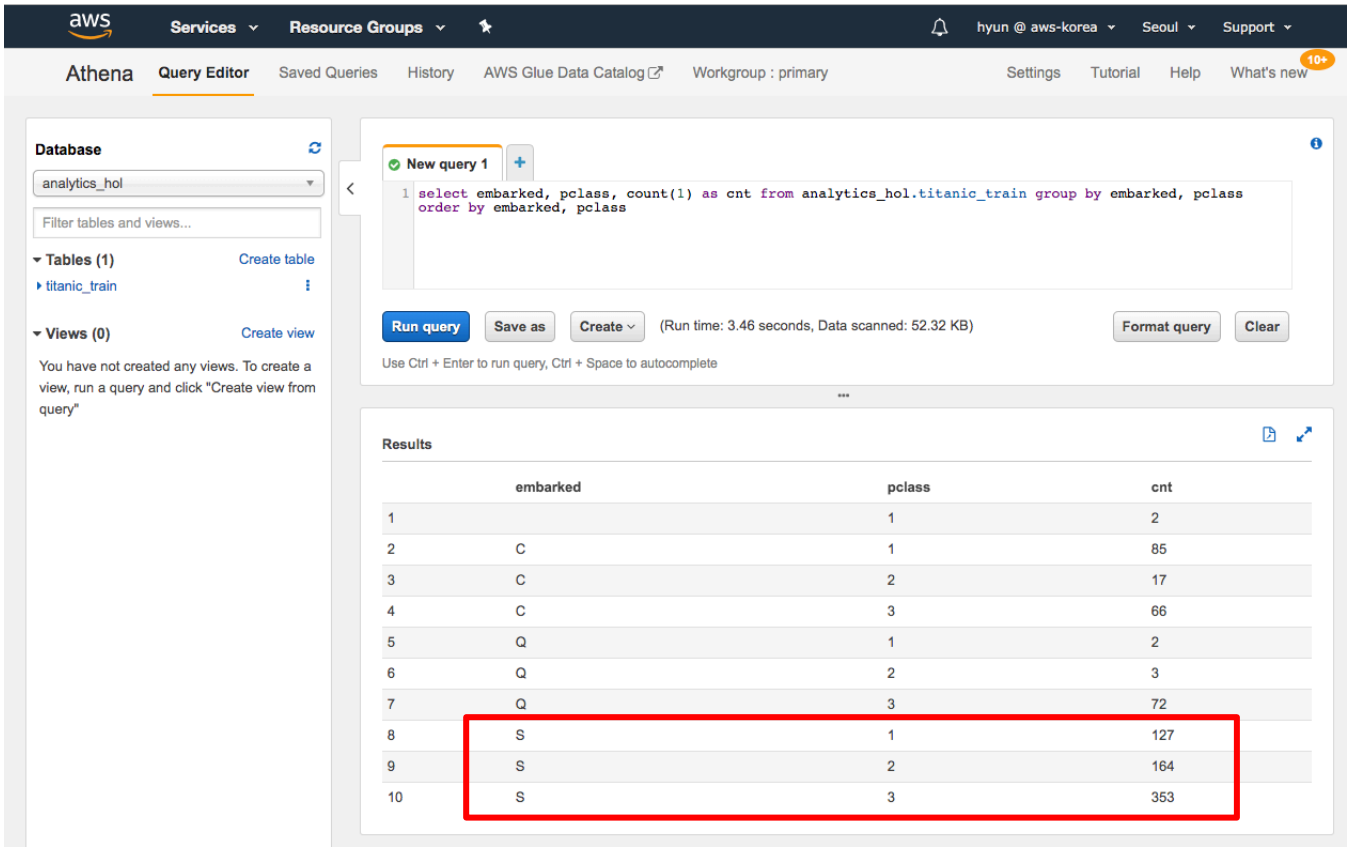
4.10. 탑승항구에 따른 생존율 확인

select embarked, survived, count(1) as cnt from analytics_hol.titanic_train group by embarked, survived order by embarked, survived



2 개의 데이터는 탑승항구가 명시되어 있지 않고, 탑승항구 S 의 생존율이 낮은 것을 알 수 있습니다. 따라서, 탑승항구 S 의 생존율이 낮은 이유를 알기 위해 탑승항구와 탑승석의 관계를 살펴볼 필요가 있습니다.

select embarked, pclass, count(1) as cnt from analytics_hol.titanic_train group by embarked, pclass order by embarked, pclass



	embarked	pclass	cnt
1		1	2
2	C	1	85
3	C	2	17
4	C	3	66
5	Q	1	2
6	Q	2	3
7	Q	3	72
8	S	1	127
9	S	2	164
10	S	3	353

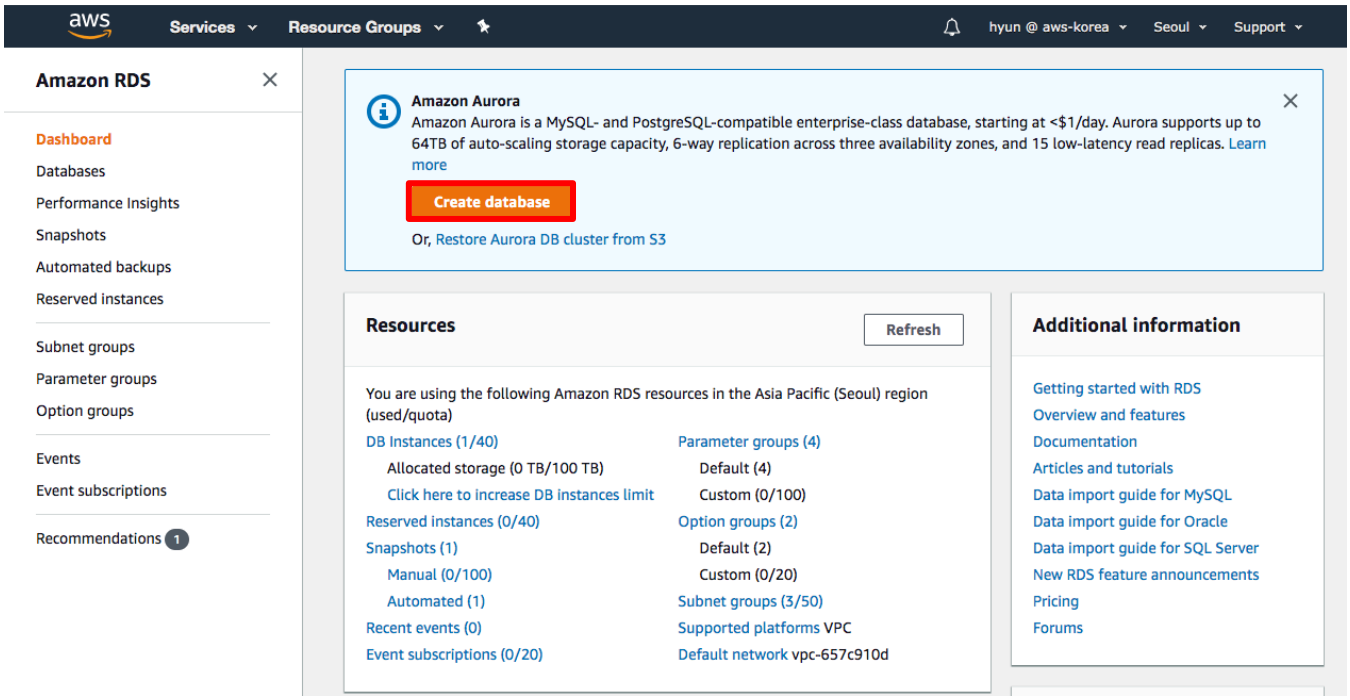
탑승항구 S 의 생존율이 낮은 이유는 대다수 탑승객이 3 등석이기 때문임을 알 수 있었습니다.

5. Create RDS and S3 Endpoint

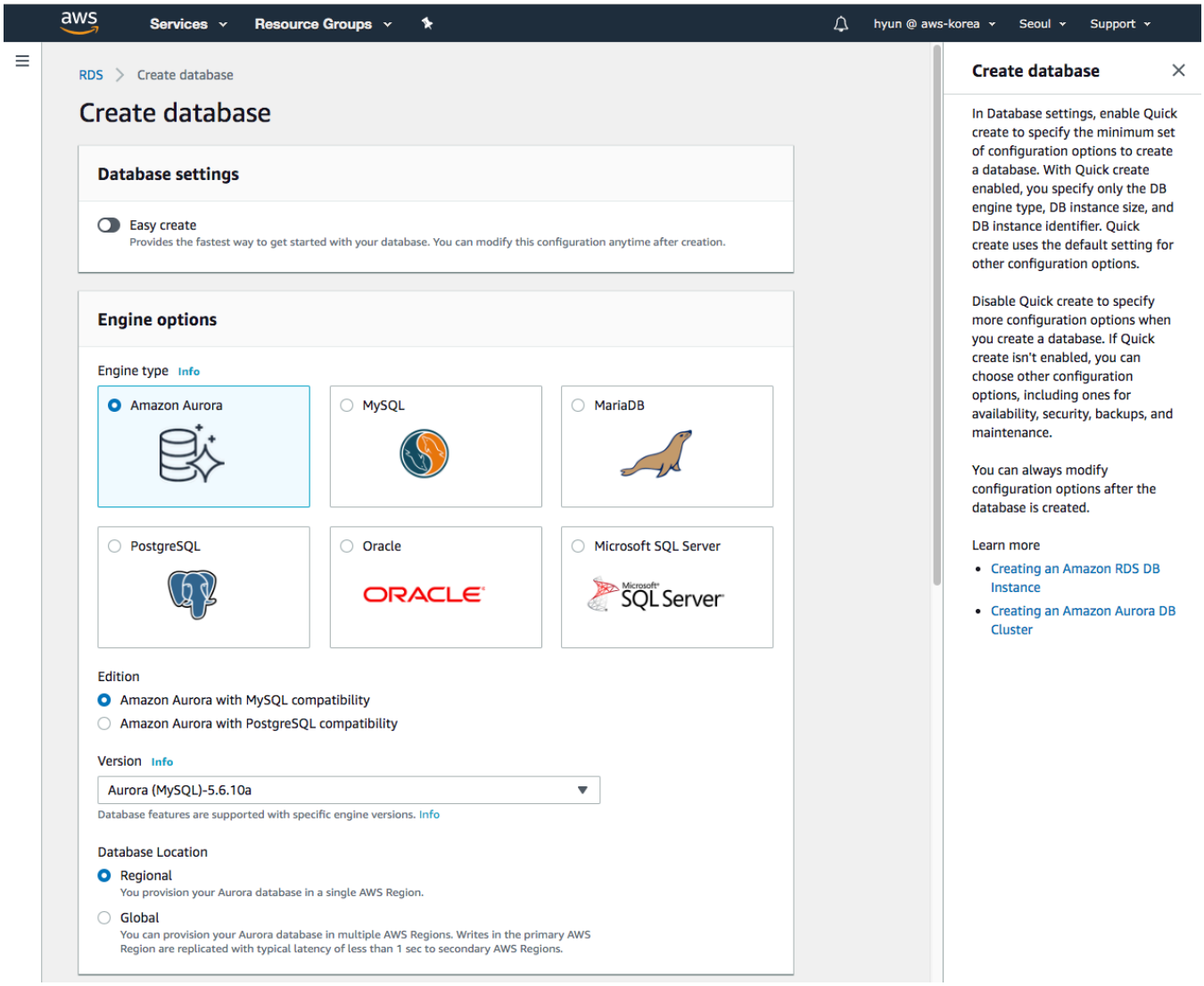
이번 장에서는 이후에 진행될 RDS Read & Write 실습을 위해 RDS 와 S3 Endpoint 를 생성하도록 하겠습니다.

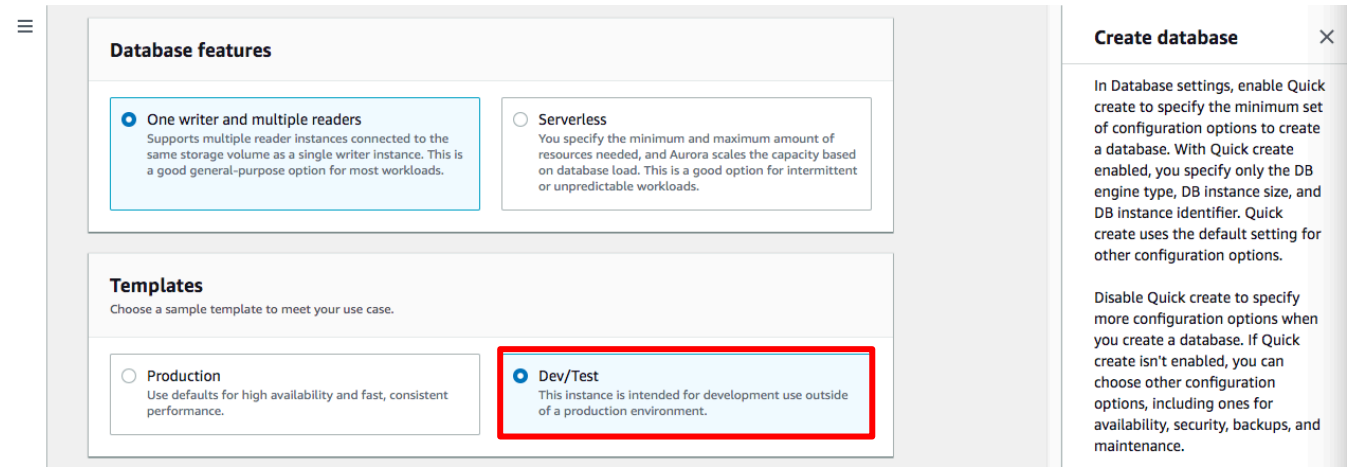
RDS 생성

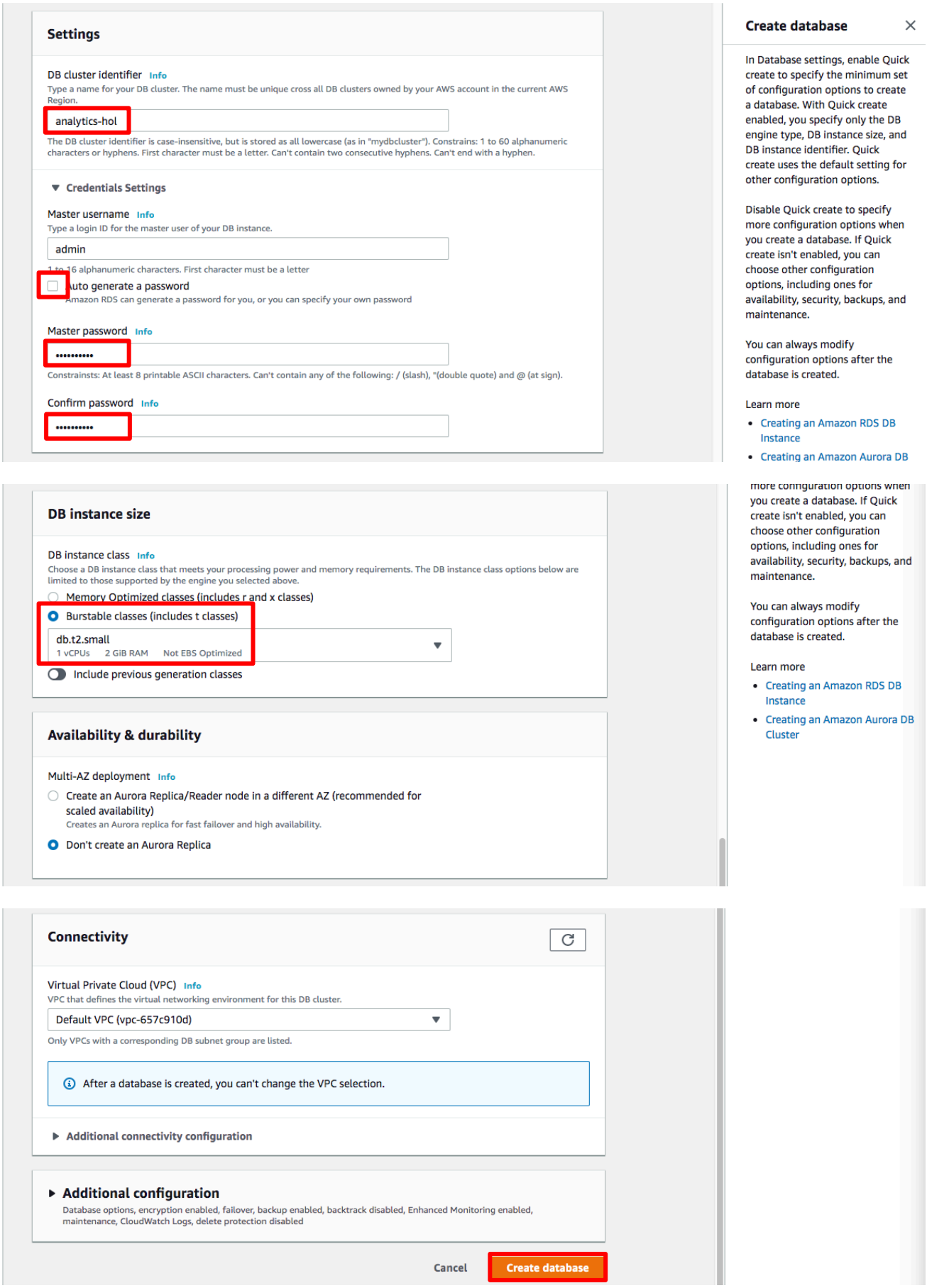
5-1. RDS 콘솔로 이동합니다. Create database 버튼을 클릭합니다.



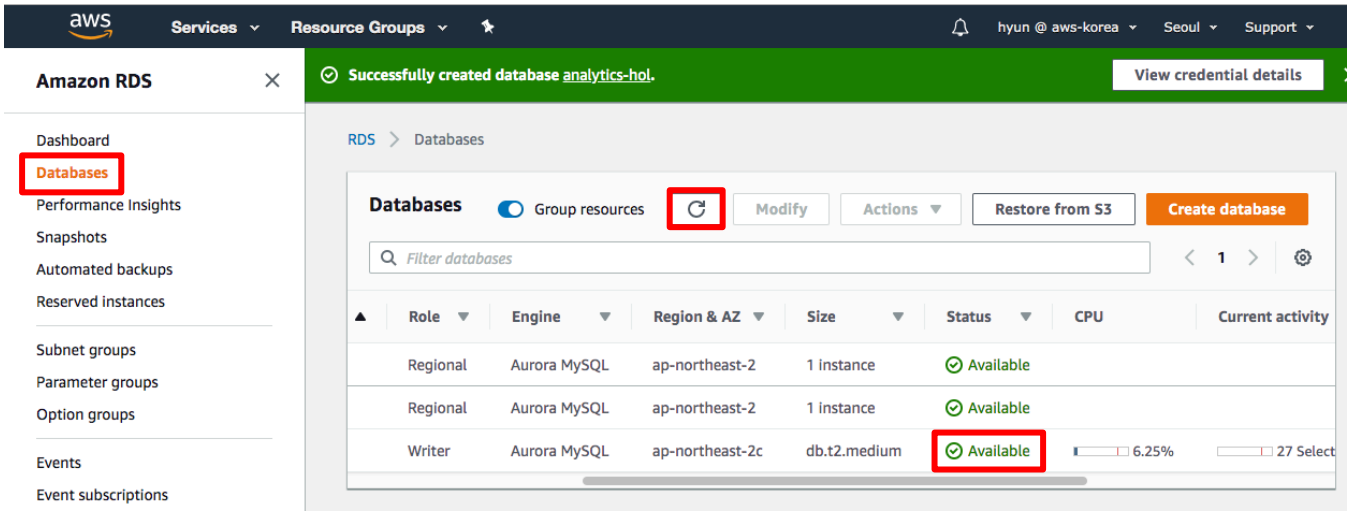
5-2. Create database 에서 Templates 은 Dev/Test 로 선택하고, DB cluster identifier 는 analytics-hol 로 입력합니다. Credentials Settings 에서 Auto generate a password 체크박스를 해제하고, Master password 와 Confirm password 를 입력합니다. (이후 실습에서 RDS 접속할 때 사용하게 됩니다.) DB instance size 는 Burstable classes 의 db.t2.small 를 선택합니다. 그 외 나머지 설정은 기본으로 두고 Create database 버튼을 클릭합니다. Database 생성까지 약 3 분이 소요됩니다.


aws
Services
Resource Groups
hyun @ aws-korea
Seoul
Support
RDS > Create database
Create database
Database settings
Easy create
Provides the fastest way to get started with your database. You can modify this configuration anytime after creation.
Engine options
Engine type Info
Amazon Aurora
MySQL
MariaDB
PostgreSQL
Oracle
Microsoft SQL Server
Edition
Amazon Aurora with MySQL compatibility
Amazon Aurora with PostgreSQL compatibility
Version Info
Aurora (MySQL)-5.6.10a
Database features are supported with specific engine versions. Info
Database Location
Regional
You provision your Aurora database in a single AWS Region.
Global
You can provision your Aurora database in multiple AWS Regions. Writes in the primary AWS Region are replicated with typical latency of less than 1 sec to secondary AWS Regions.
Create database
In Database settings, enable Quick create to specify the minimum set of configuration options to create a database. With Quick create enabled, you specify only the DB engine type, DB instance size, and DB instance identifier. Quick create uses the default setting for other configuration options.
Disable Quick create to specify more configuration options when you create a database. If Quick create isn't enabled, you can choose other configuration options, including ones for availability, security, backups, and maintenance.
You can always modify configuration options after the database is created.
Learn more
Creating an Amazon RDS DB Instance
Creating an Amazon Aurora DB Cluster

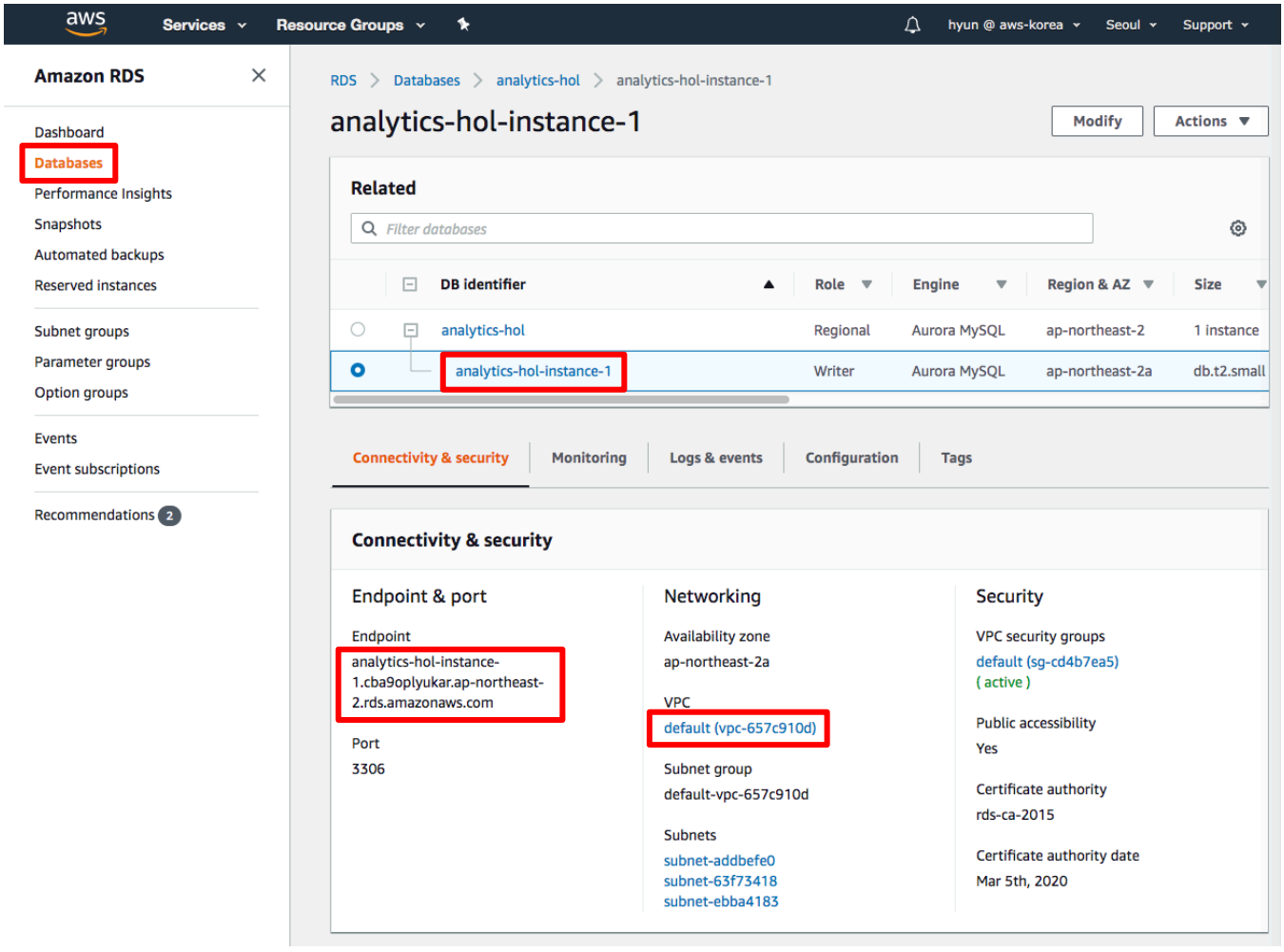

Database features
One writer and multiple readers
Supports multiple reader instances connected to the same storage volume as a single writer instance. This is a good general-purpose option for most workloads.
Serverless
You specify the minimum and maximum amount of resources needed, and Aurora scales the capacity based on database load. This is a good option for intermittent or unpredictable workloads.
Templates
Choose a sample template to meet your use case.
Production
Use defaults for high availability and fast, consistent performance.
Dev/Test
This instance is intended for development use outside of a production environment.
Create database
In Database settings, enable Quick create to specify the minimum set of configuration options to create a database. With Quick create enabled, you specify only the DB engine type, DB instance size, and DB instance identifier. Quick create uses the default setting for other configuration options.
Disable Quick create to specify more configuration options when you create a database. If Quick create isn't enabled, you can choose other configuration options, including ones for availability, security, backups, and maintenance.


Settings
DB cluster identifier Info
Type a name for your DB cluster. The name must be unique cross all DB clusters owned by your AWS account in the current AWS Region.
analytics-hol
The DB cluster identifier is case-insensitive, but is stored as all lowercase (as in "mydbcluster"). Constrains: 1 to 60 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.
Credentials Settings
Master username Info
Type a login ID for the master user of your DB instance.
admin
1 to 16 alphanumeric characters. First character must be a letter
Auto generate a password
Amazon RDS can generate a password for you, or you can specify your own password
Master password Info
Constrainsts: At least 8 printable ASCII characters. Can't contain any of the following: / (slash), "(double quote) and @ (at sign).
Confirm password Info
Create database
In Database settings, enable Quick create to specify the minimum set of configuration options to create a database. With Quick create enabled, you specify only the DB engine type, DB instance size, and DB instance identifier. Quick create uses the default setting for other configuration options.
Disable Quick create to specify more configuration options when you create a database. If Quick create isn't enabled, you can choose other configuration options, including ones for availability, security, backups, and maintenance.
You can always modify configuration options after the database is created.
Learn more
Creating an Amazon RDS DB Instance
Creating an Amazon Aurora DB
DB instance size
DB instance class Info
Choose a DB instance class that meets your processing power and memory requirements. The DB instance class options below are limited to those supported by the engine you selected above.
Memory Optimized classes (includes r and x classes)
Burstable classes (includes t classes)
db.t2.small
1 vCPUs 2 GiB RAM Not EBS Optimized
Include previous generation classes
Availability & durability
Multi-AZ deployment Info
Create an Aurora Replica/Reader node in a different AZ (recommended for scaled availability)
Creates an Aurora replica for fast failover and high availability.
Don't create an Aurora Replica
Connectivity
Virtual Private Cloud (VPC) Info
VPC that defines the virtual networking environment for this DB cluster.
Default VPC (vpc-657c910d)
Only VPCs with a corresponding DB subnet group are listed.
After a database is created, you can't change the VPC selection.
Additional connectivity configuration
Additional configuration
Database options, encryption enabled, failover, backup enabled, backtrack disabled, Enhanced Monitoring enabled, maintenance, CloudWatch Logs, delete protection disabled
Cancel
Create database

5-3. 생성이 완료되면 아래와 같이 생성된 Database 의 Status 가 Available 로 변경되는 것을 확인할 수 있습니다.

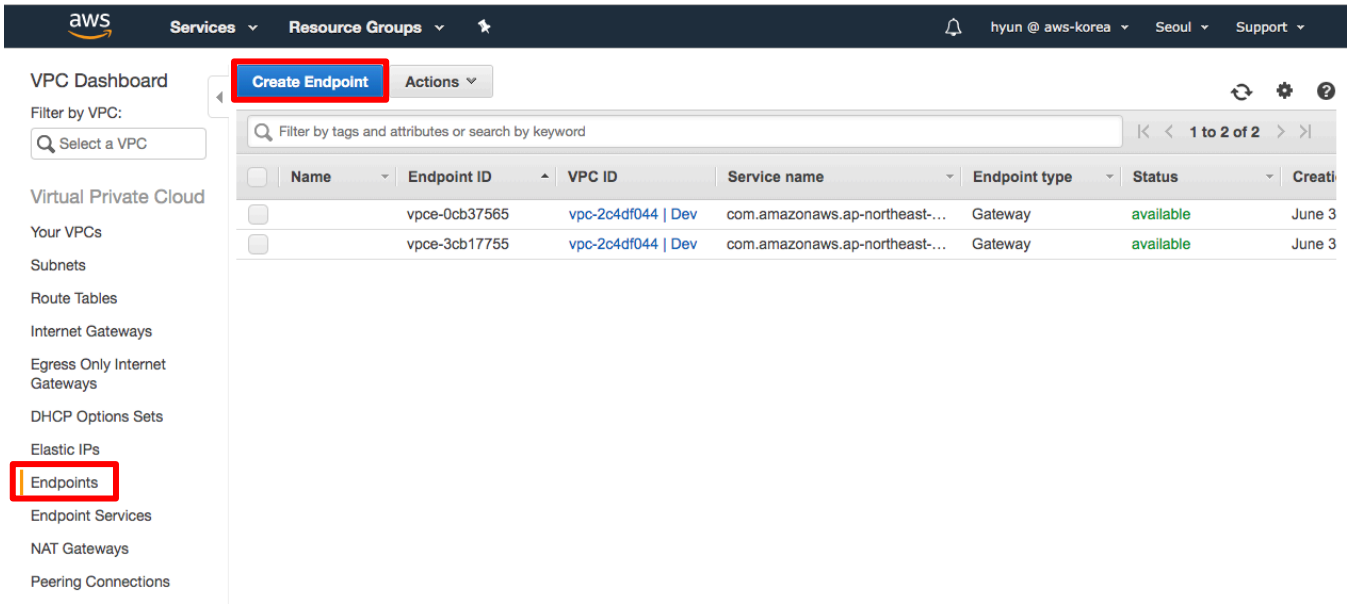


5-4. 생성된 Database 의 Writer 의 DB identifier 를 클릭해서 관련정보를 확인합니다. Endpoint 와 VPC 는 이후 실습에서 사용하니 copy 해서 메모하도록 합니다.

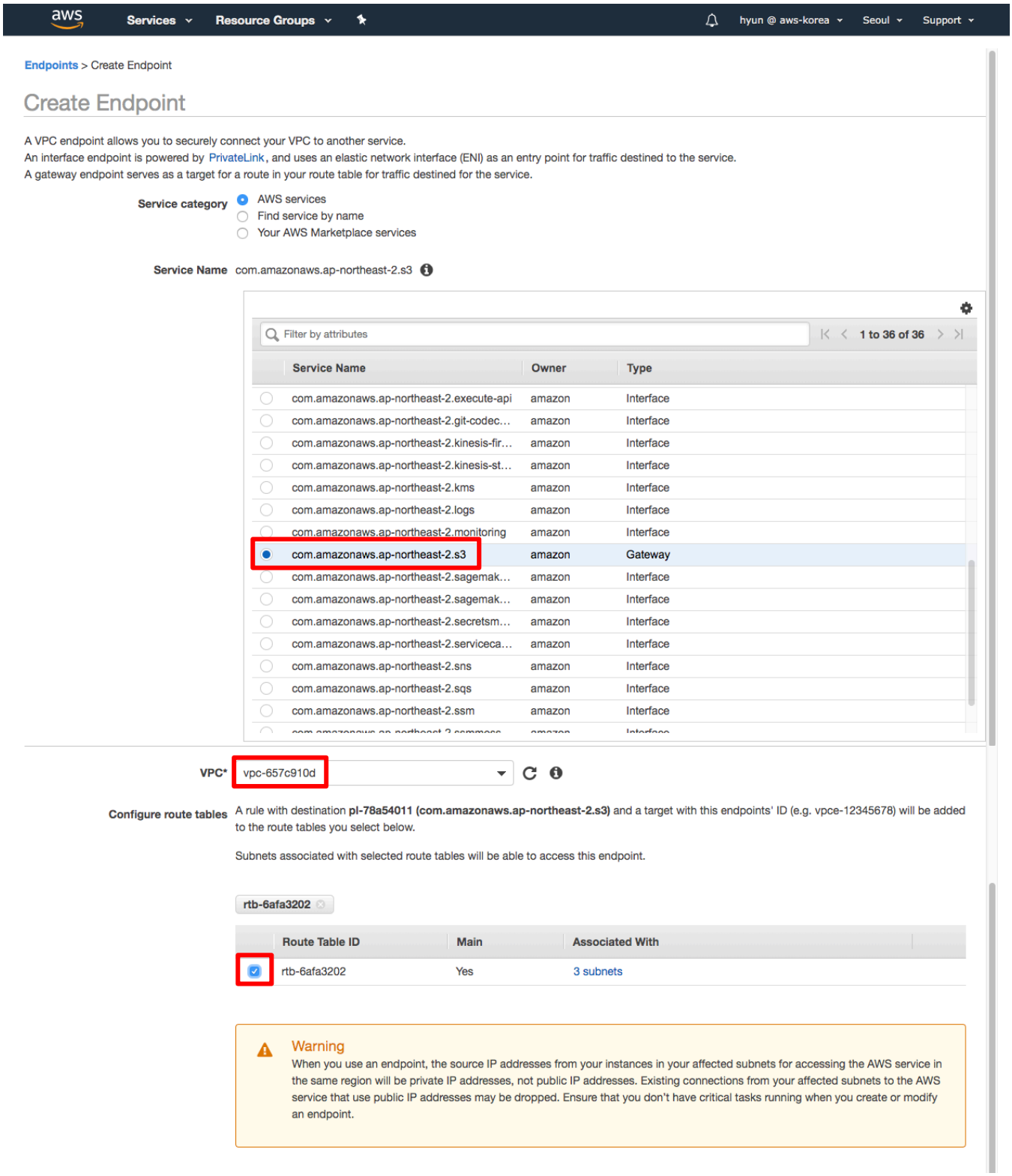


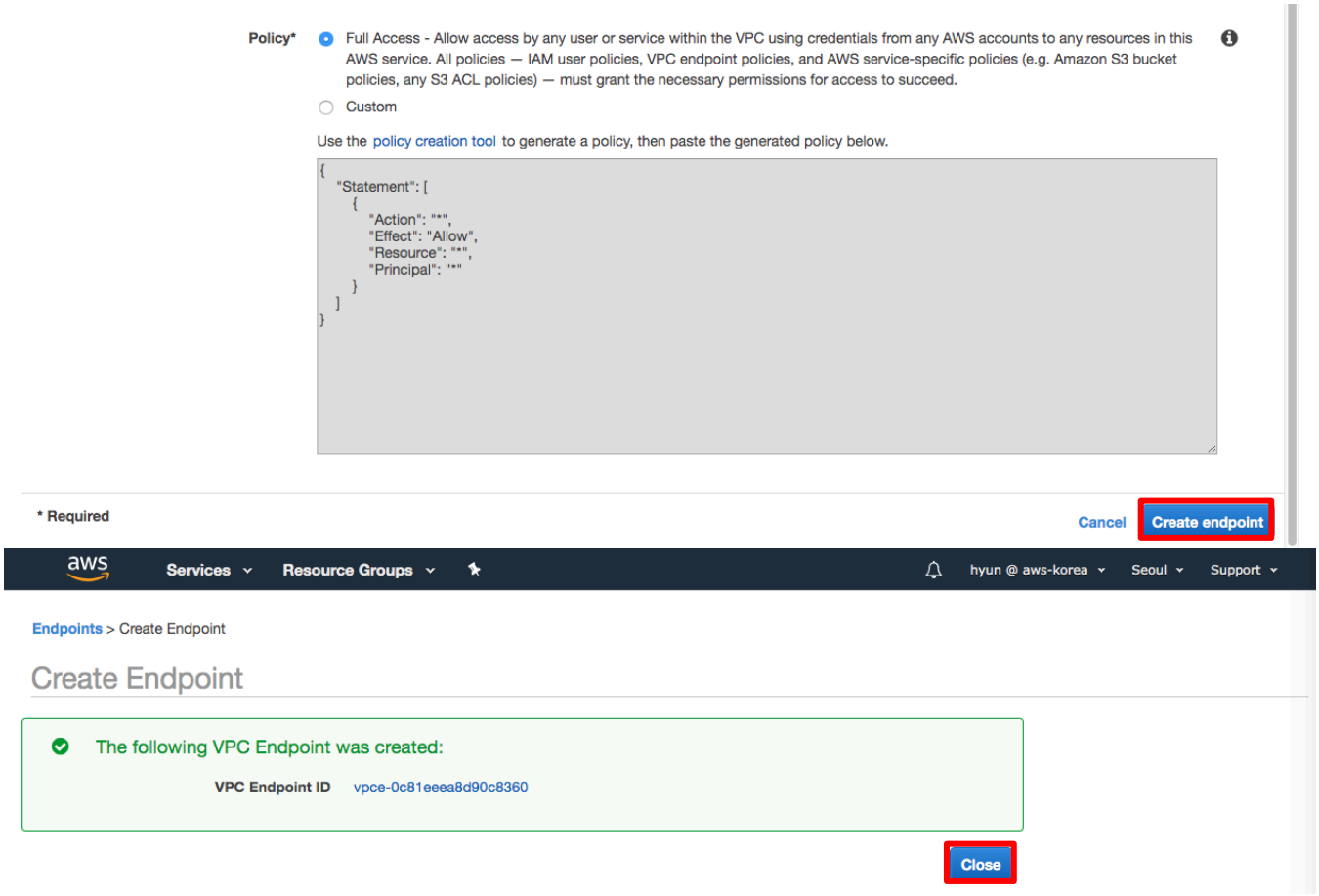
S3 Endpoint 생성

5-5. 생성된 Database 는 VPC 내에 생성되기 때문에 해당 Database 에 접근하기 위해 Glue Job 의 Cluster 는 VPC 내부에서 생성되게 됩니다. Glue Job 이 VPC 내부에서 S3 에 Private 하게 접근하기 위해 지금부터는 S3 Endpoint 를 생성하도록 합니다. VPC 콘솔로 이동합니다. Endpoints 로 이동한 후 Create Endpoint 버튼을 클릭합니다. (RDS 가 생성된 VPC 에 S3 Endpoint 가 이미 생성되어 있는 경우 Skip 하도록 합니다.)

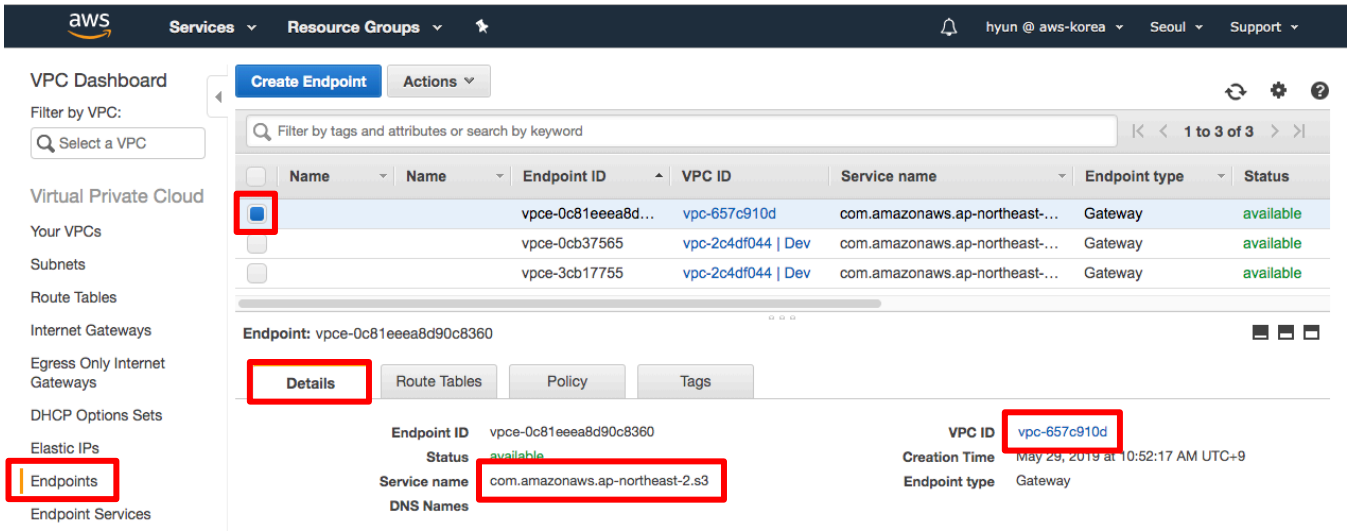


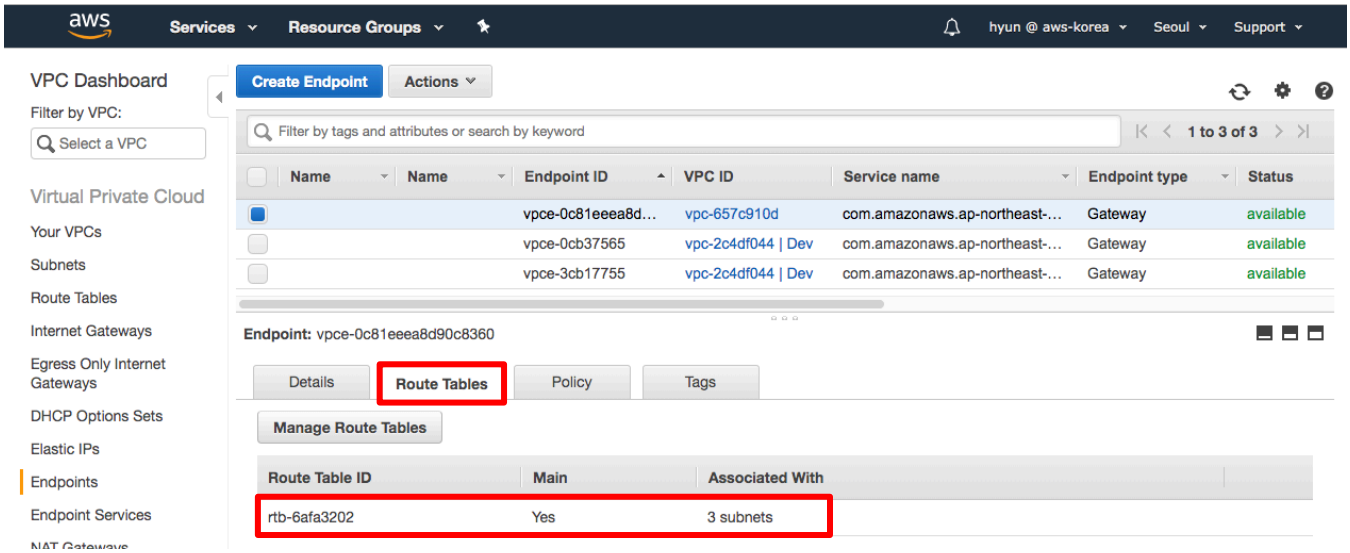
5-6. s3 로 끝나는 Service Name 을 선택하고 Database 생성 때 메모한 VPC 와 동일한 VPC 가 선택되어 있는 지 확인합니다. Configure route tables 에서 해당 VPC 를 위해 설정된 Route Table ID 를 선택하고 Create endpoint 버튼을 클릭합니다. (Route Table ID 가 여러 개 있어 어떤 ID 를 선택해야하는 지 모를 경우 SA 에게 문의하여 확인 후 진행하도록 합니다.)





5-7. 아래와 같이 정상적으로 S3 Endpoint 가 생성되었는지 확인합니다.



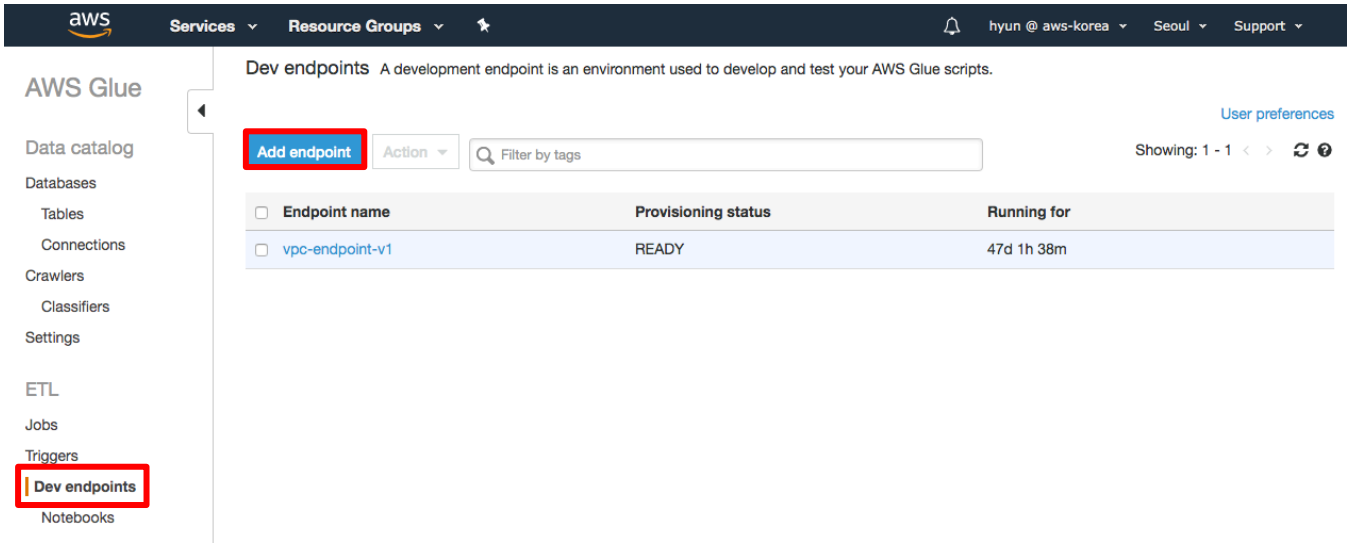

aws
Services
Resource Groups
hyun @ aws-korea
Seoul
Support
VPC Dashboard
Filter by VPC:
Select a VPC
Virtual Private Cloud
Your VPCs
Subnets
Route Tables
Internet Gateways
Egress Only Internet Gateways
DHCP Options Sets
Elastic IPs
Endpoints
Endpoint Services
Create Endpoint
Actions
Filter by tags and attributes or search by keyword
1 to 3 of 3
Name
Name
Endpoint ID
VPC ID
Service name
Endpoint type
Status
vpce-0c81eeea8d...
vpc-657c910d
com.amazonaws.ap-northeast-...
Gateway
available
vpce-0cb37565
vpc-2c4df044 | Dev
com.amazonaws.ap-northeast-...
Gateway
available
vpce-3cb17755
vpc-2c4df044 | Dev
com.amazonaws.ap-northeast-...
Gateway
available
Endpoint: vpce-0c81eeea8d90c8360
Details
Route Tables
Policy
Tags
Manage Route Tables
Route Table ID
Main
Associated With
rtb-6afa3202
Yes
3 subnets

6. Glue Dev Endpoint with SageMaker notebook

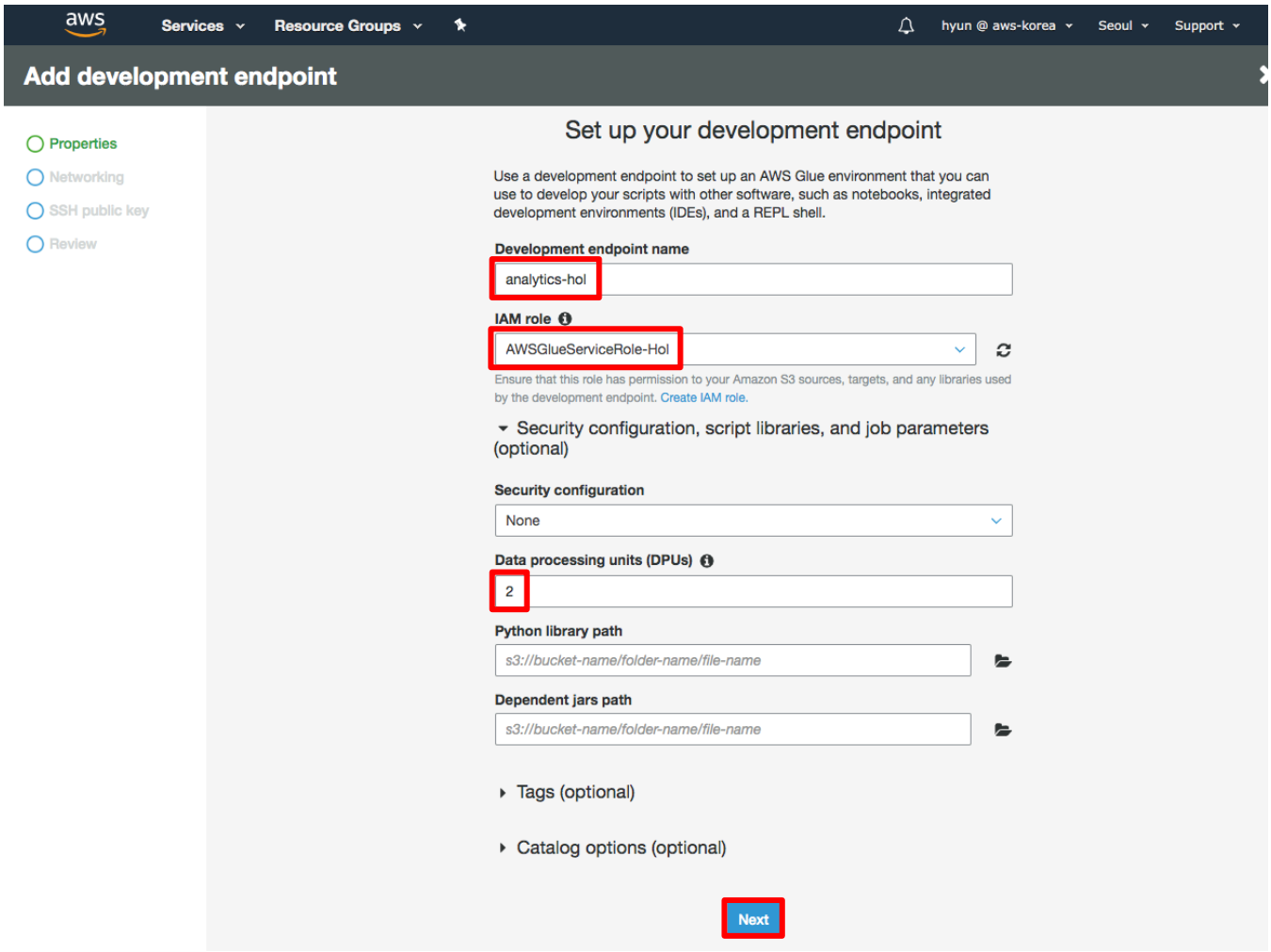
다음은 ETL 작업 및 머신러닝 모델 개발을 위해 Dev endpoint 와 SageMaker notebook 을 생성하겠습니다.

Dev Endpoint 생성

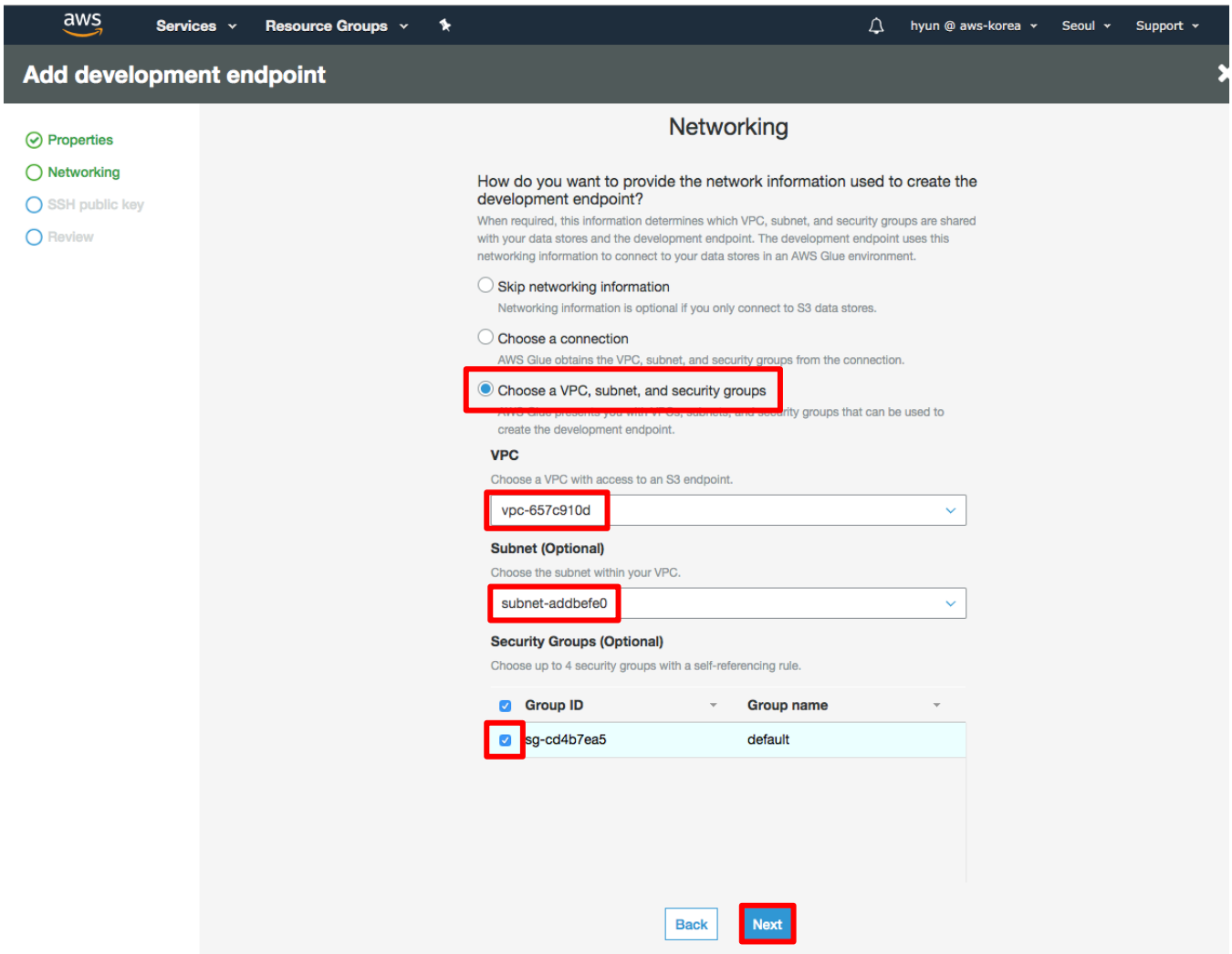
6-1. Glue 콘솔로 이동합니다. Dev endpoints 로 이동한 후, Add endpoint 버튼을 클릭합니다.



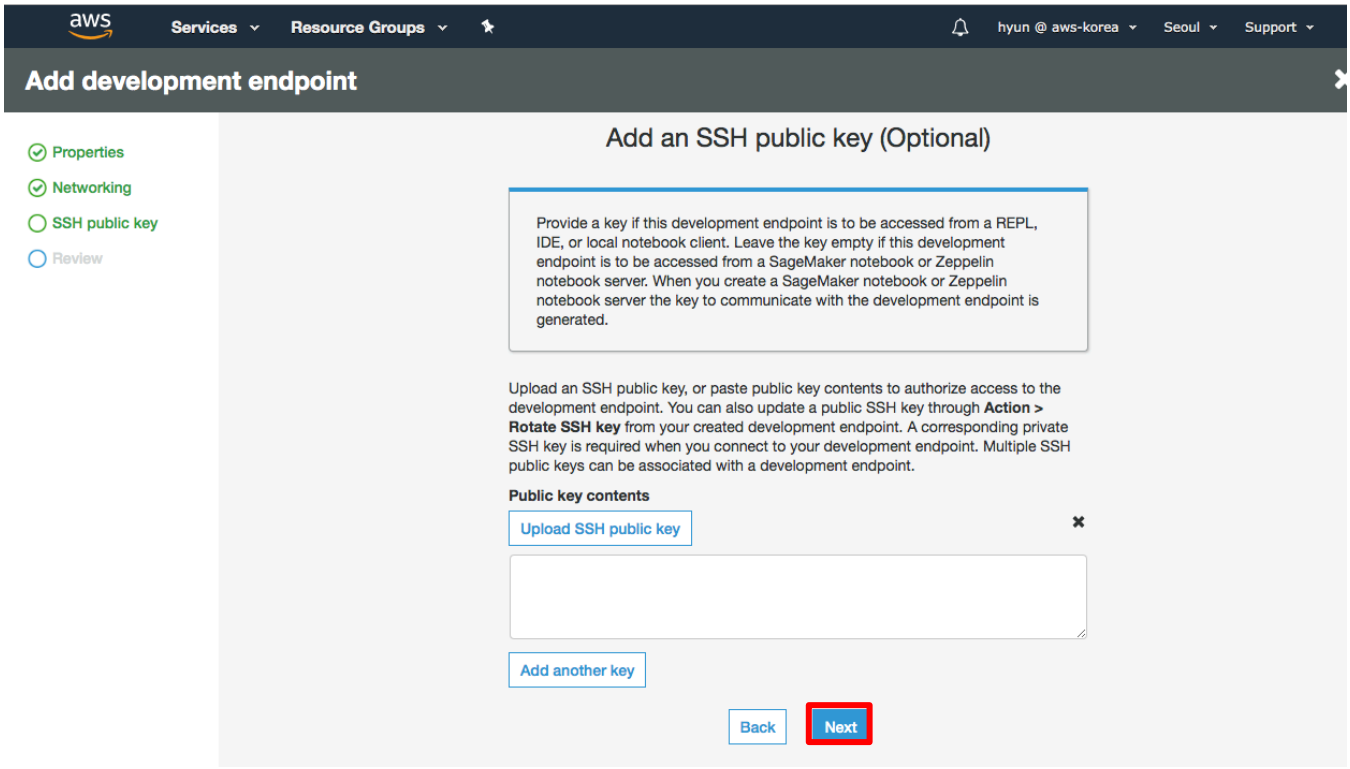
6-2. Development endpoint name 을 analytics-hol 로, IAM role 은 3 장에서 생성한 AWSGlueServiceRole-Hol 을 선택, Data processing units(DPUs)는 2 로 선택하고 Next 를 클릭합니다.



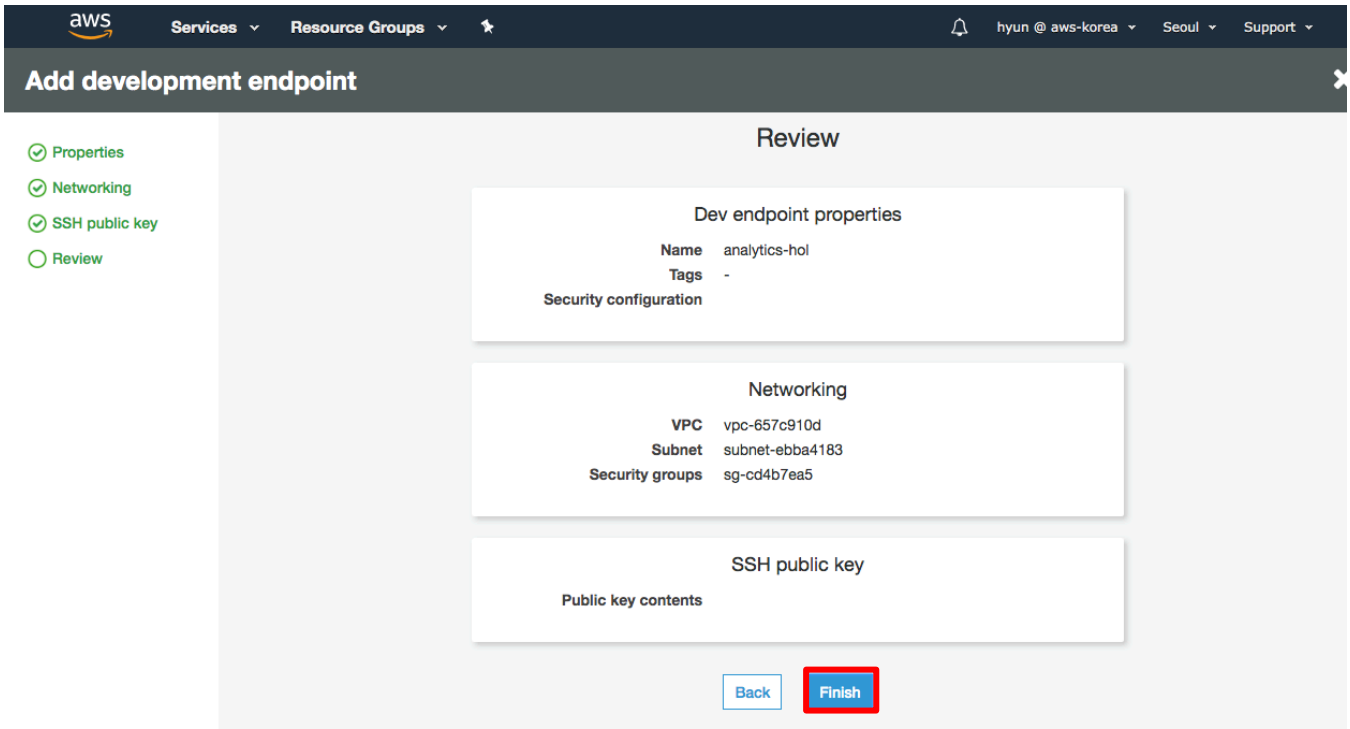
6-3. Choose a VPC, subnet, and security groups 를 선택하고 VPC 는 RDS 가 생성된 VPC 와 동일한 VPC 를 선택합니다. 3 개의 subnet 중 하나를 선택하고 security group 체크박스를 선택한 후 Next 버튼을 클릭합니다.



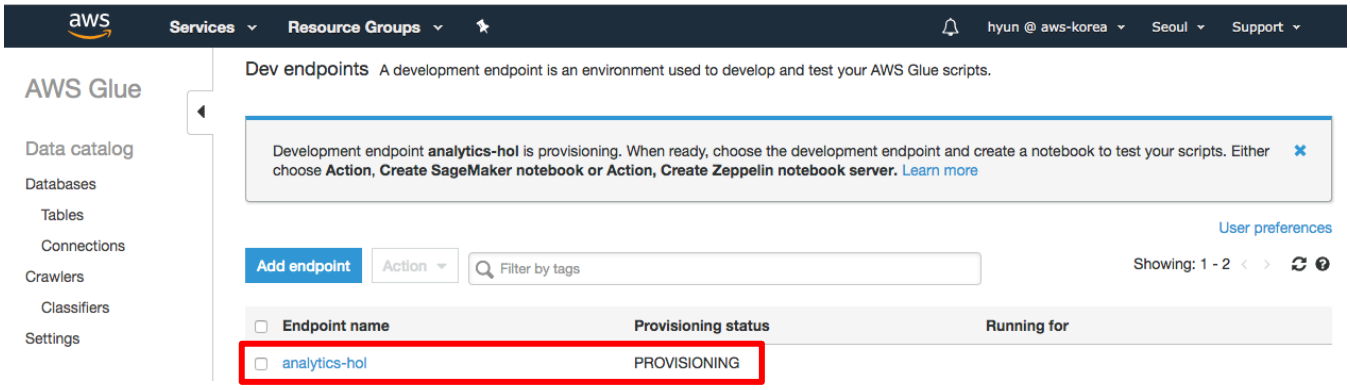
6-4. SSH Public key(Optional) 단계에서는 기본 설정으로 Next 버튼을 클릭합니다.

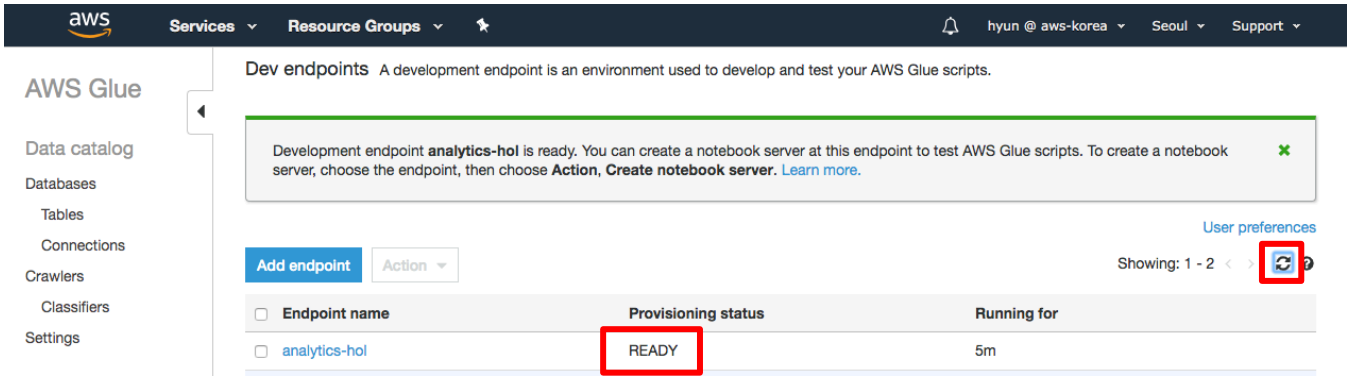


6-5. 내용을 리뷰한 후 Finish 버튼을 클릭하면 생성이 시작됩니다. (약 6 분 소요)



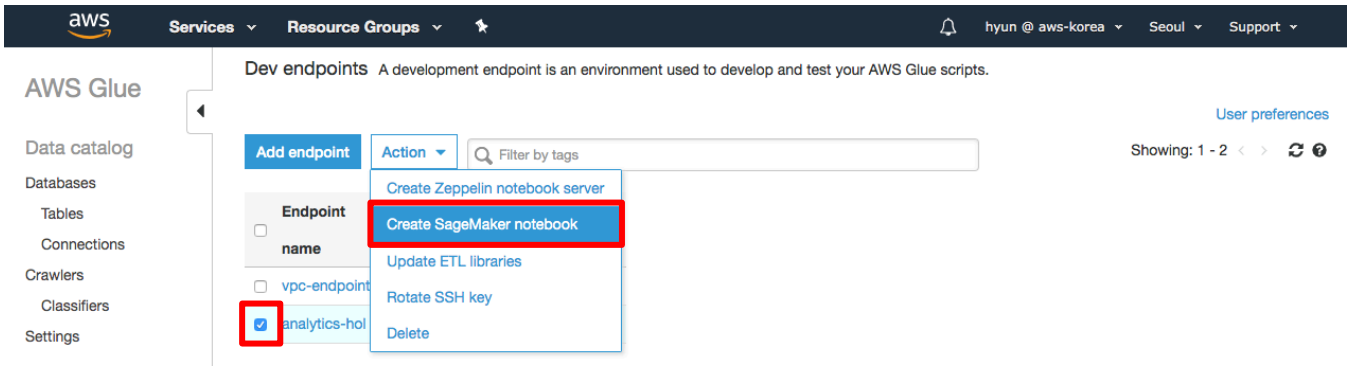
6-6. Endpoint 생성이 완료되면 PROVISIONING 상태에서 READY 상태로 바뀝니다. 오른쪽 상단의 리프래시 버튼을 클릭하여 Provisioning status 가 READY 상태로 변경된 것을 확인합니다.



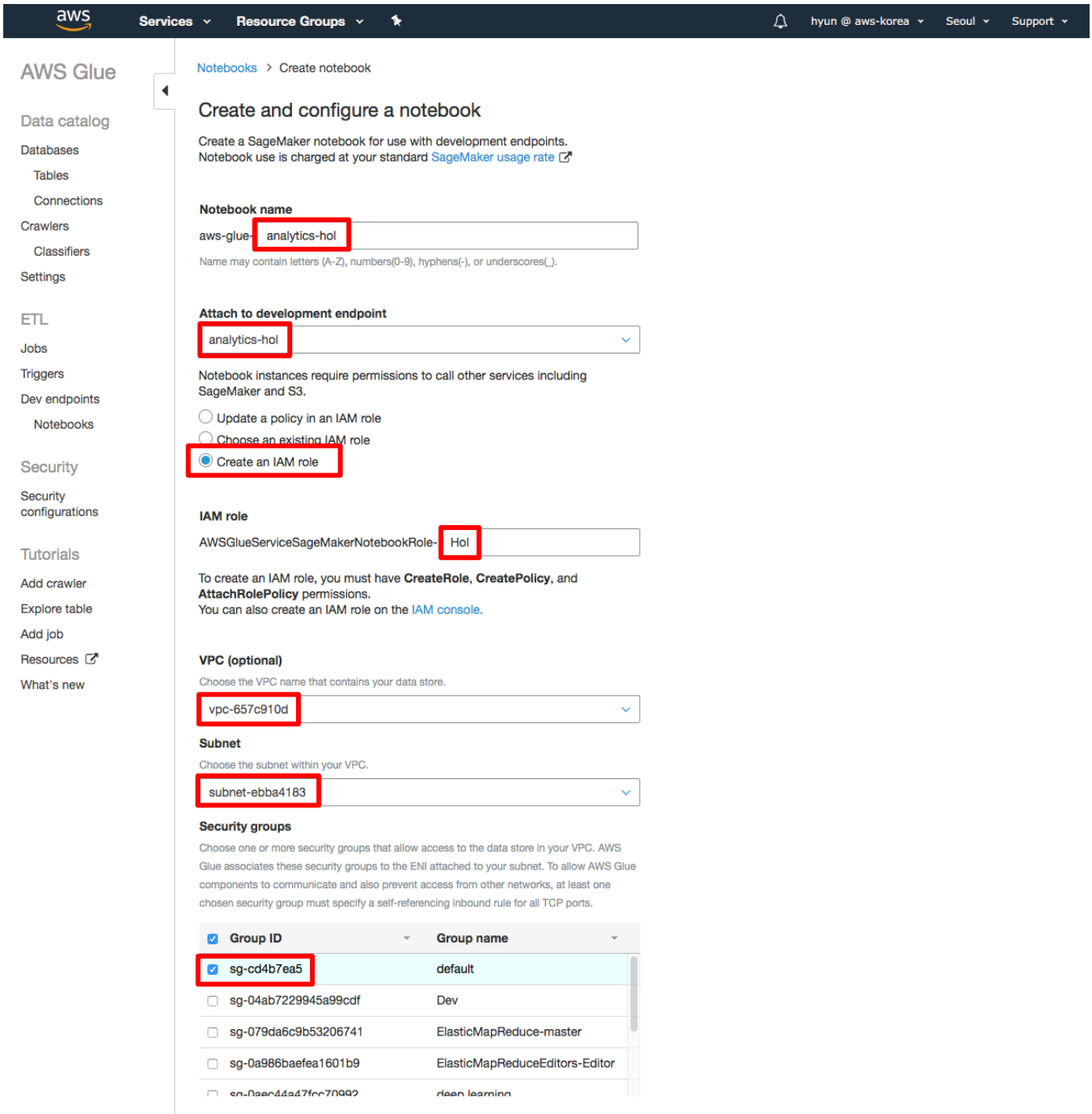


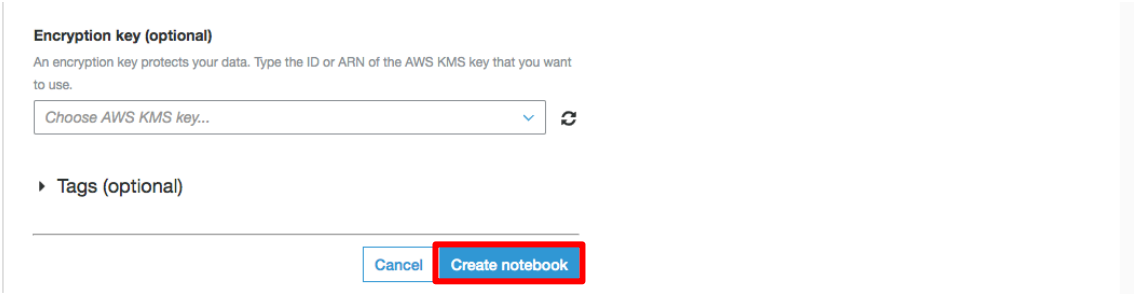
SageMaker notebook 생성

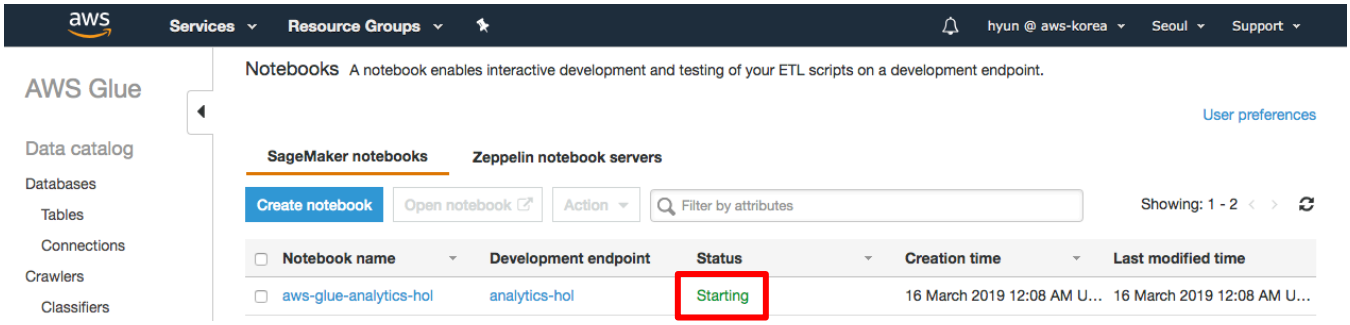
6-7. READY 상태의 Dev endpoint 를 선택하고 Action 에서 Create SageMaker notebook 을 선택합니다.



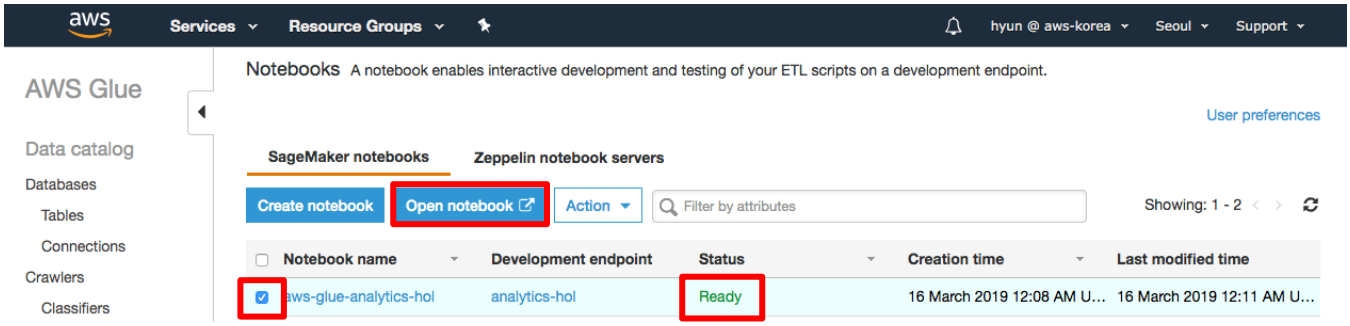
6-8. Notebook name 에 analytics-hol 이라고 입력합니다. Attach to development endpoint 에 analytics-hol 을 선택하고 Create an IAM role 을 선택합니다. IAM role 에 Hol 이라고 입력하고 VPC 에 Database 가 생성되었던 VPC 와 동일한 지, Subnet 은 해당 VPC 에 속한 Subnet 인 지 확인 후 Create notebook 버튼을 눌러 SageMaker notebook 을 생성합니다. (약 5 분 소요)



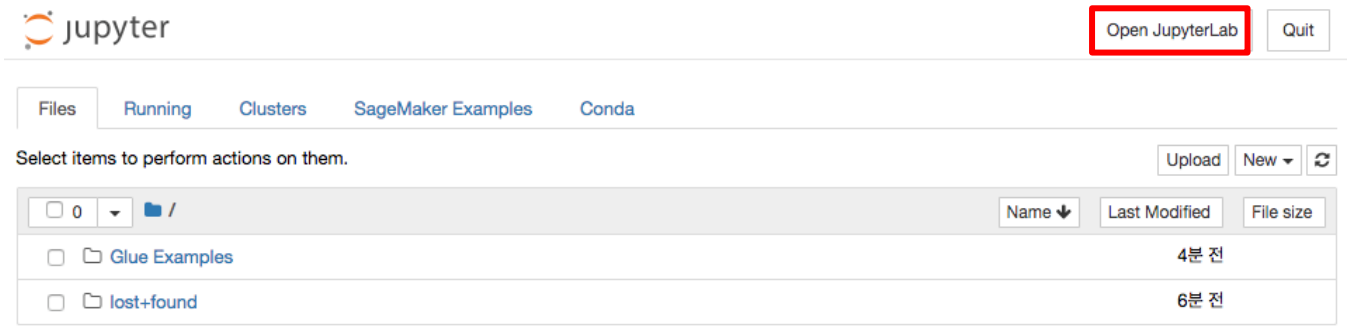


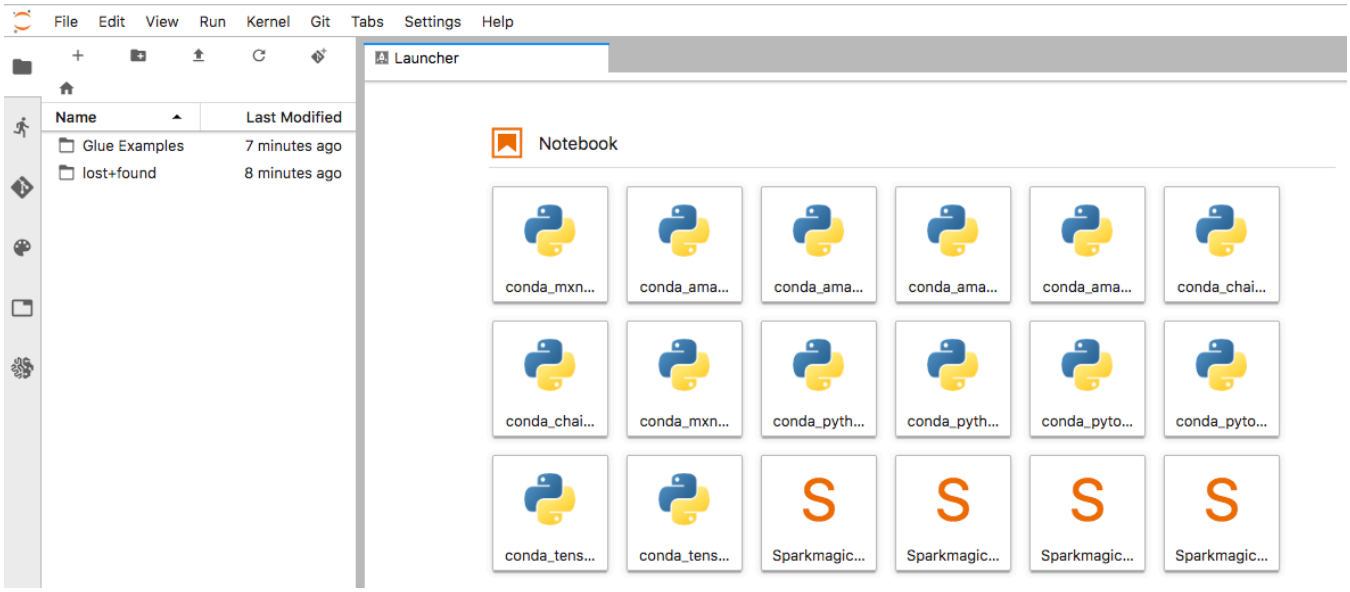


6-9. 생성한 aws-glue-analytics-hol SageMaker notebook 의 Status 가 Ready 로 변경되면 해당 notebook 을 선택하고 Open notebook 버튼을 클릭하여 SageMaker notebook 을 실행합니다.



6-10. SageMaker notebook 이 실행되면 Open JupyterLab 버튼을 클릭하여 JupyterLab 을 실행합니다.



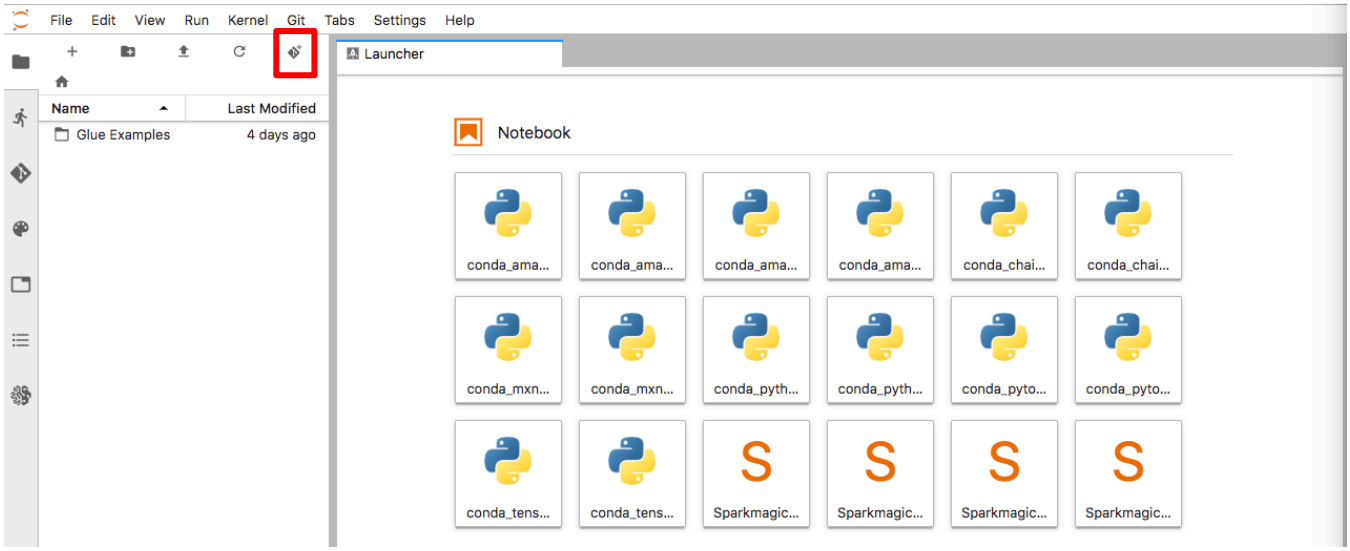

File
Edit
View
Run
Kernel
Git
Tabs
Settings
Help
Launcher
Name
Last Modified
Glue Examples
7 minutes ago
lost+found
8 minutes ago
Notebook
conda_mxn...
conda_ama...
conda_ama...
conda_ama...
conda_ama...
conda_chai...
conda_chai...
conda_mxn...
conda_pyth...
conda_pyth...
conda_pyto...
conda_pyto...
conda_tens...
conda_tens...
Sparkmagic...
Sparkmagic...
Sparkmagic...
Sparkmagic...

7. Lab: Data Analysis, ETL and ML

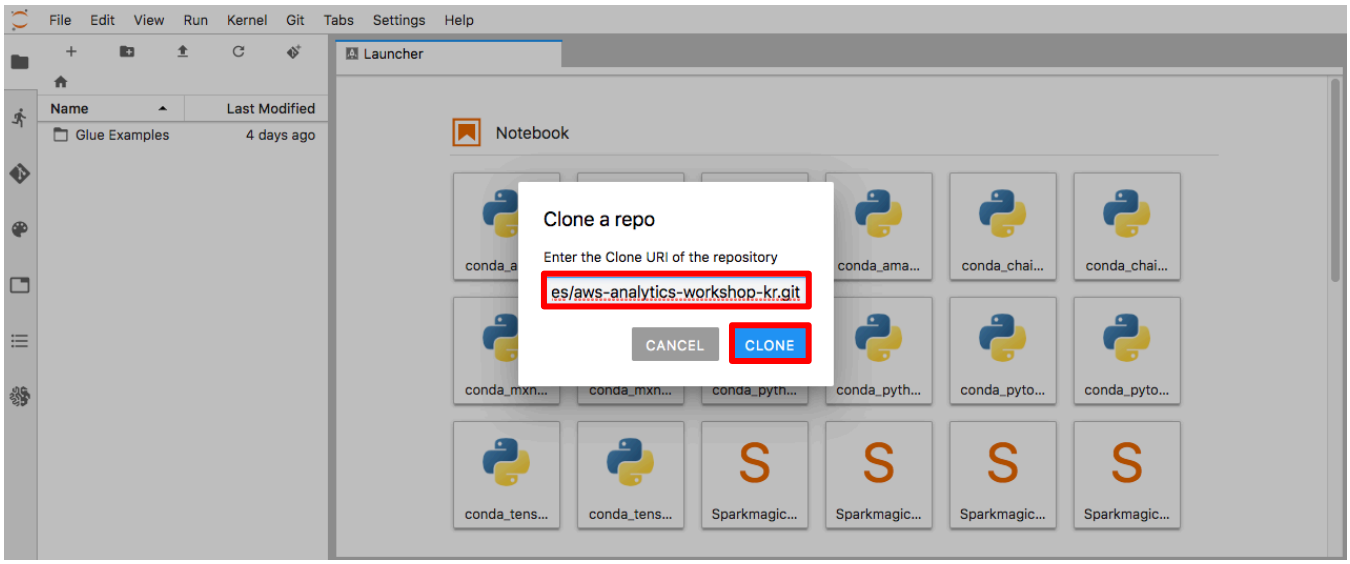
이번 장에서는 JupyterLab 에서 데이터를 분석하고 ETL 을 진행하며 다양한 Machine Learning Model 을 생성하고 성능을 테스트하는 작업을 진행하겠습니다.

Git Clone

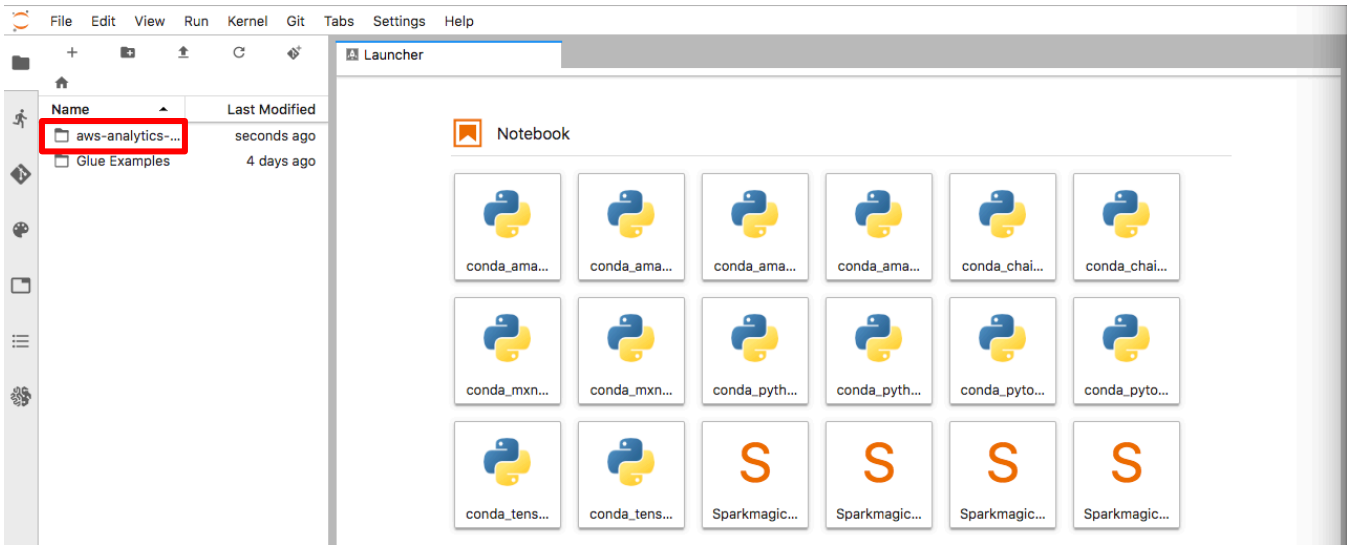
7-1. JupyterLab 에서 Git Clone 버튼을 클릭합니다.



7-2. https://github.com/aws-samples/aws-analytics-workshop-kr.git 을 입력하고 CLONE 버튼을 클릭합니다.



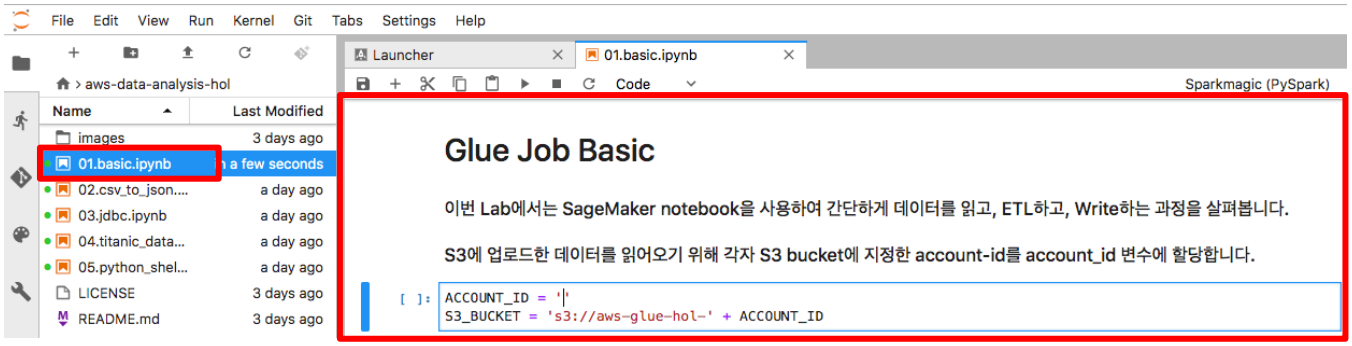
7-3. aws-analytics-workshop-kr 폴더가 생성된 것을 확인할 수 있습니다.



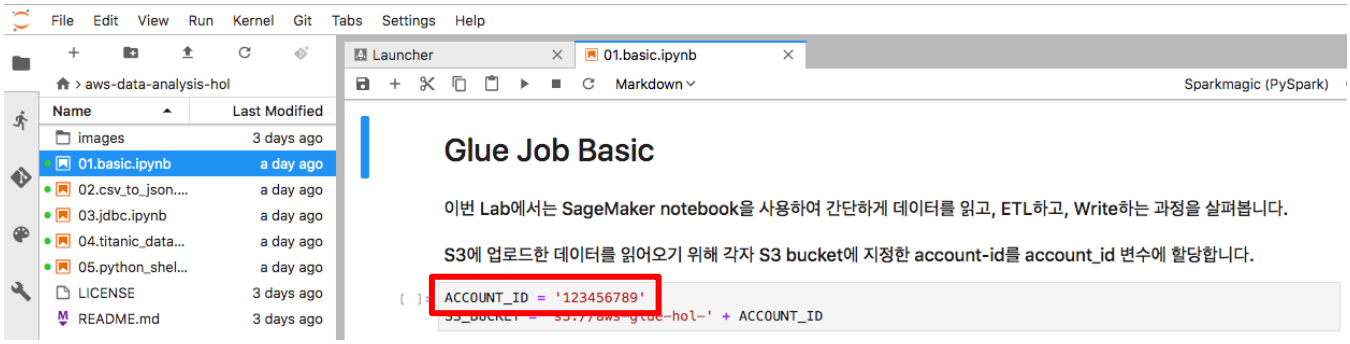
Lab1: Glue Job Basic

Lab1 에서는 Glue Job 을 위한 기본 Template 을 살펴봅니다.

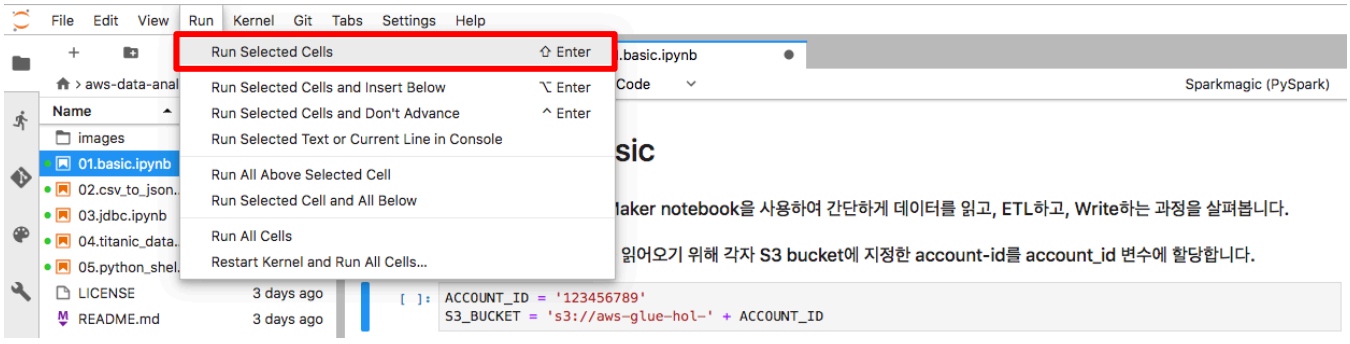
7-4. 생성된 aws-analytics-workshop-kr/glue-deep-dive-hol 폴더를 클릭하여 이동 후 01.basic_job_template.ipynb 노트북 파일을 클릭하여 오픈합니다.



7-5. 01.basic_job_template.ipynb 노트북에서 첫번 째 코드 Cell 의 account_id 값을 본인의 account_id 로 대체합니다.



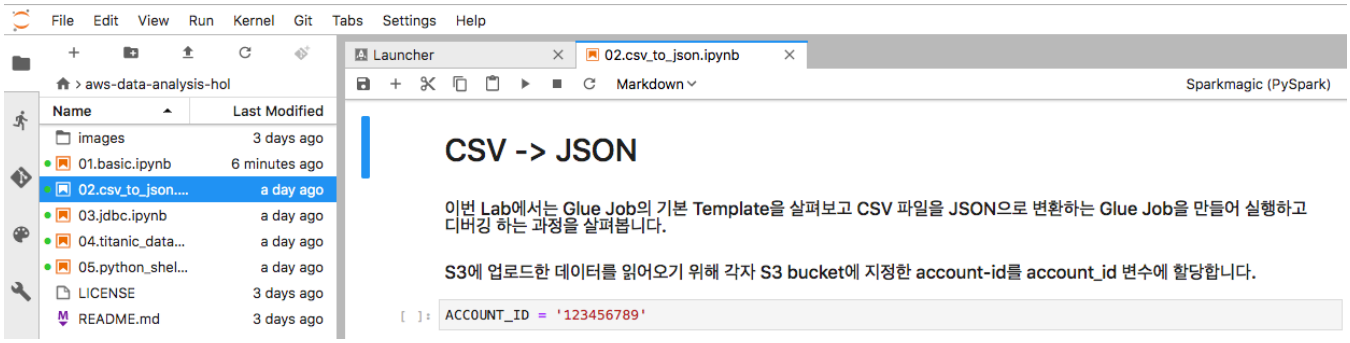
7-6. 노트북의 설명과 코드의 주석을 살펴가며 코드를 하나씩 실행하고 결과를 확인합니다. 코드 실행은 실행할 코드 셀을 선택한 후 키보드의 Shift + Enter 로 할 수도 있고, 메뉴에서 Run->Run Selected Cells 를 선택해서 할 수도 있습니다.



Lab2: Create Job, Run and Debugging

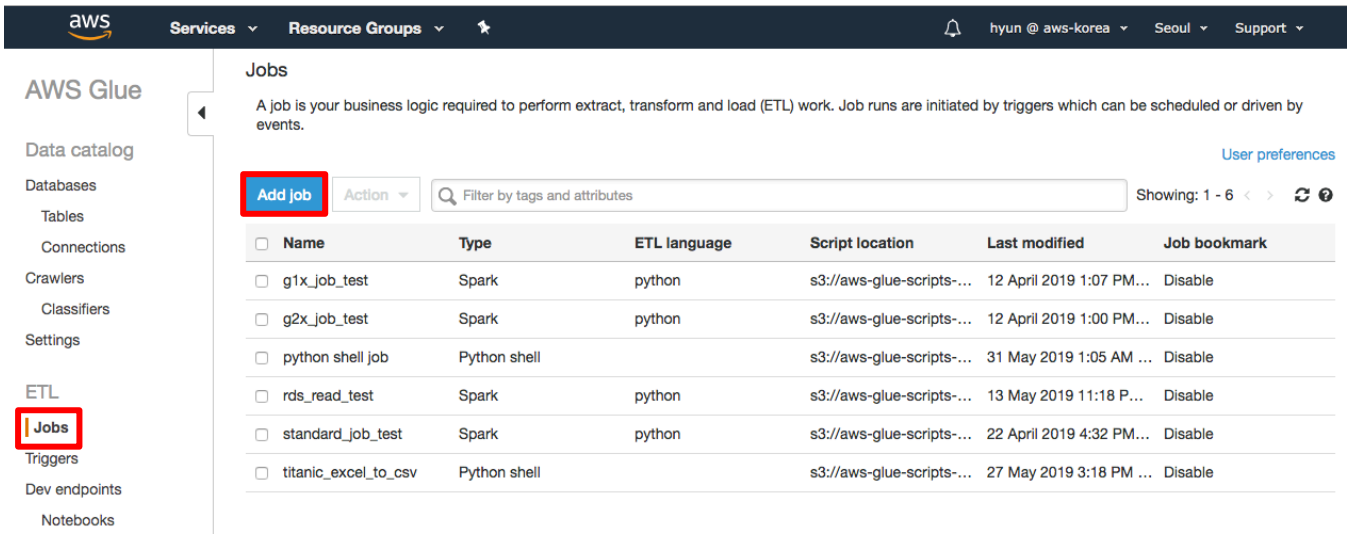
Lab2 에서는 csv 포맷 파일을 json 포맷으로 변경하는 Glue Job 을 생성한 후 실행 및 디버깅하는 과정을 진행해보도록 하겠습니다.

7-7. 02.csv_to_json.ipynb 노트북 파일을 클릭하여 오픈하고 첫번 째 코드 Cell 의 ACCOUNT_ID 값을 본인의 account_id 로 대체합니다.

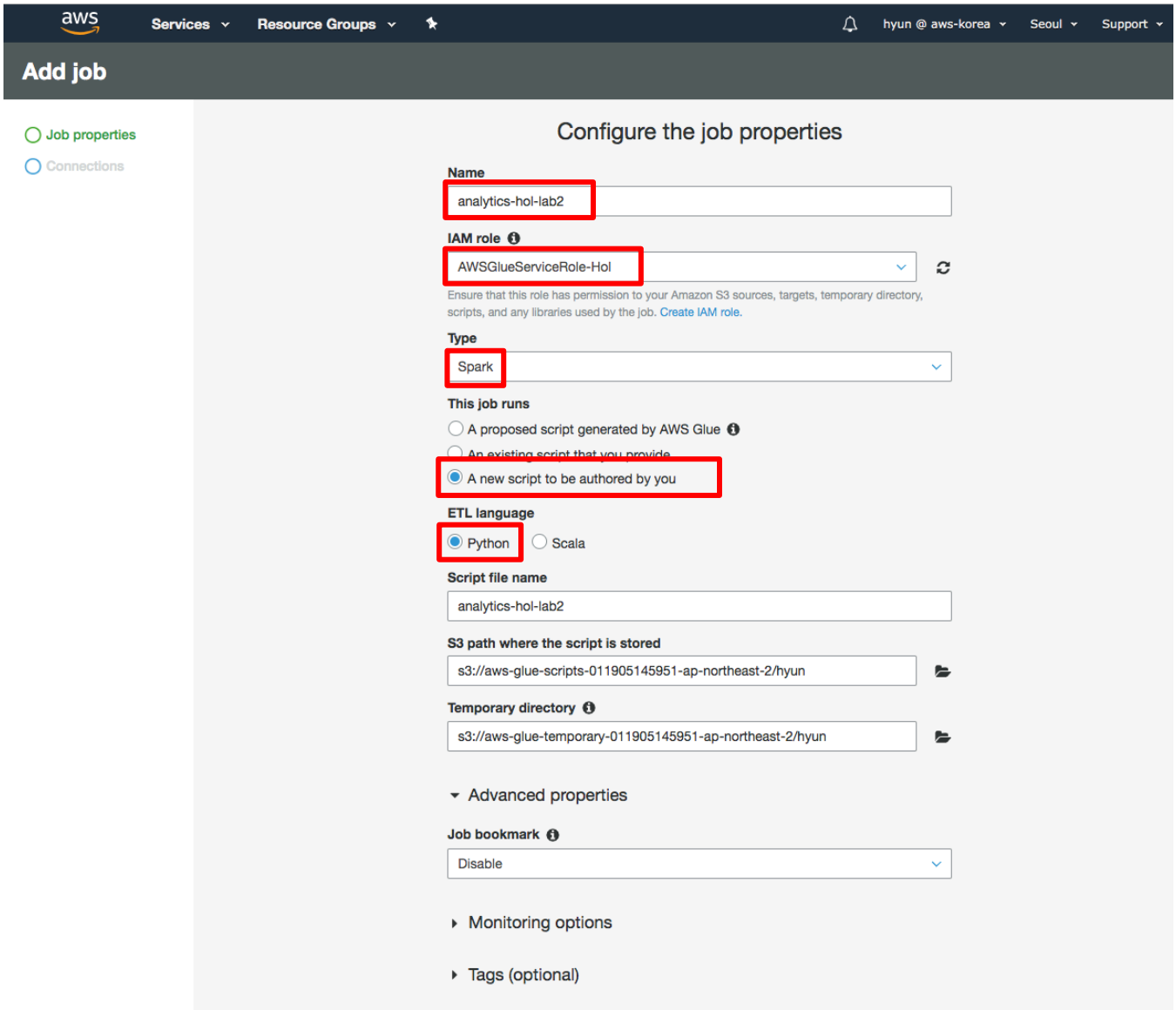


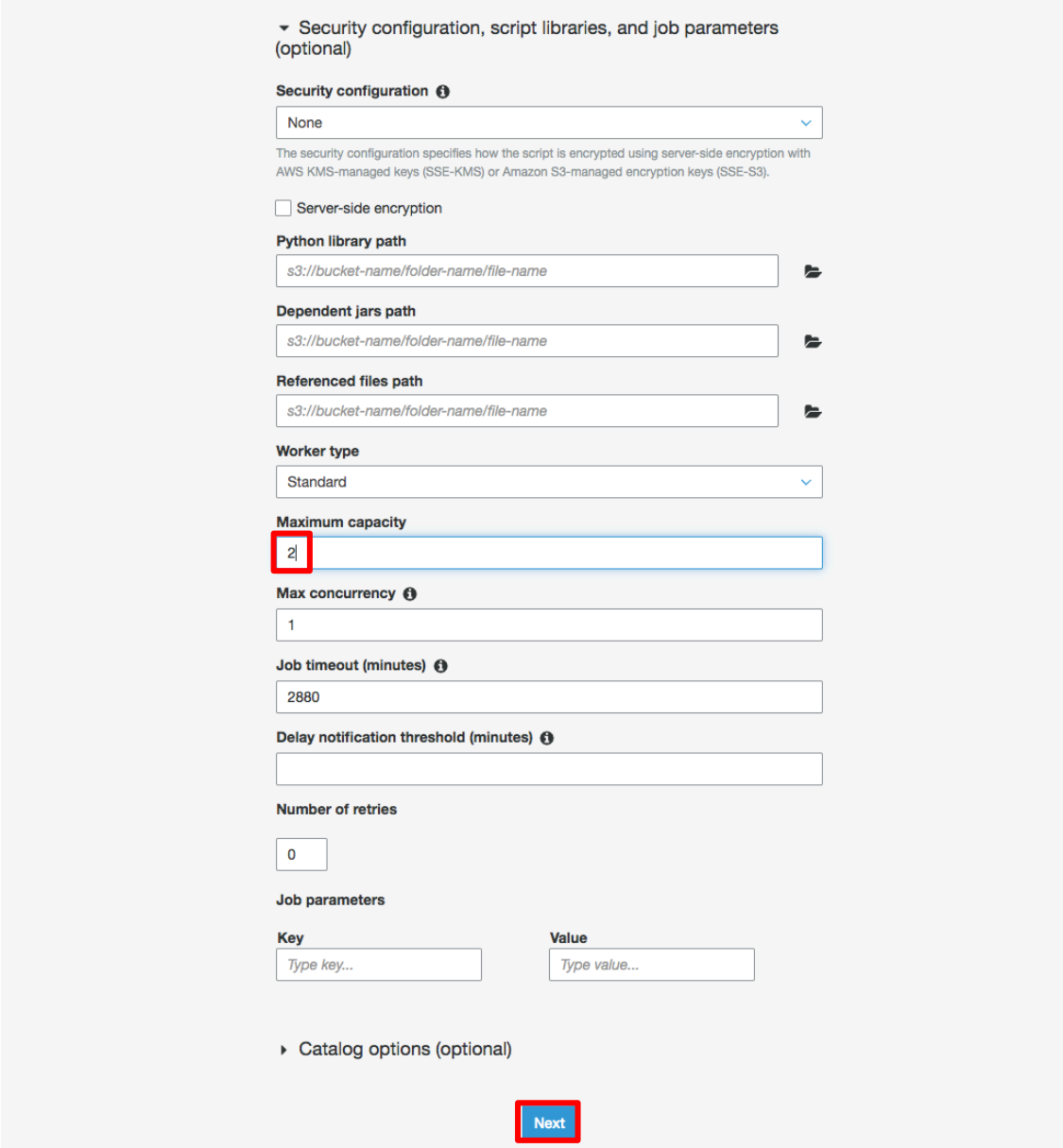
7-8. 노트북 가이드에 따라 "File Read & Write with Spark API"와 "File Read & Write with Glue API" Lab 을 진행합니다. Lab 의 마지막 코드 Cell 은 Glue Job 에서만 실행되고 Notebook 에서는 실행되지 않으니 Notebook 에서 실행하지 않습니다.

7-9. Glue Job 생성을 위해 Glue 콘솔로 이동합니다. Jobs 로 이동한 후 Add job 버튼을 클릭합니다.

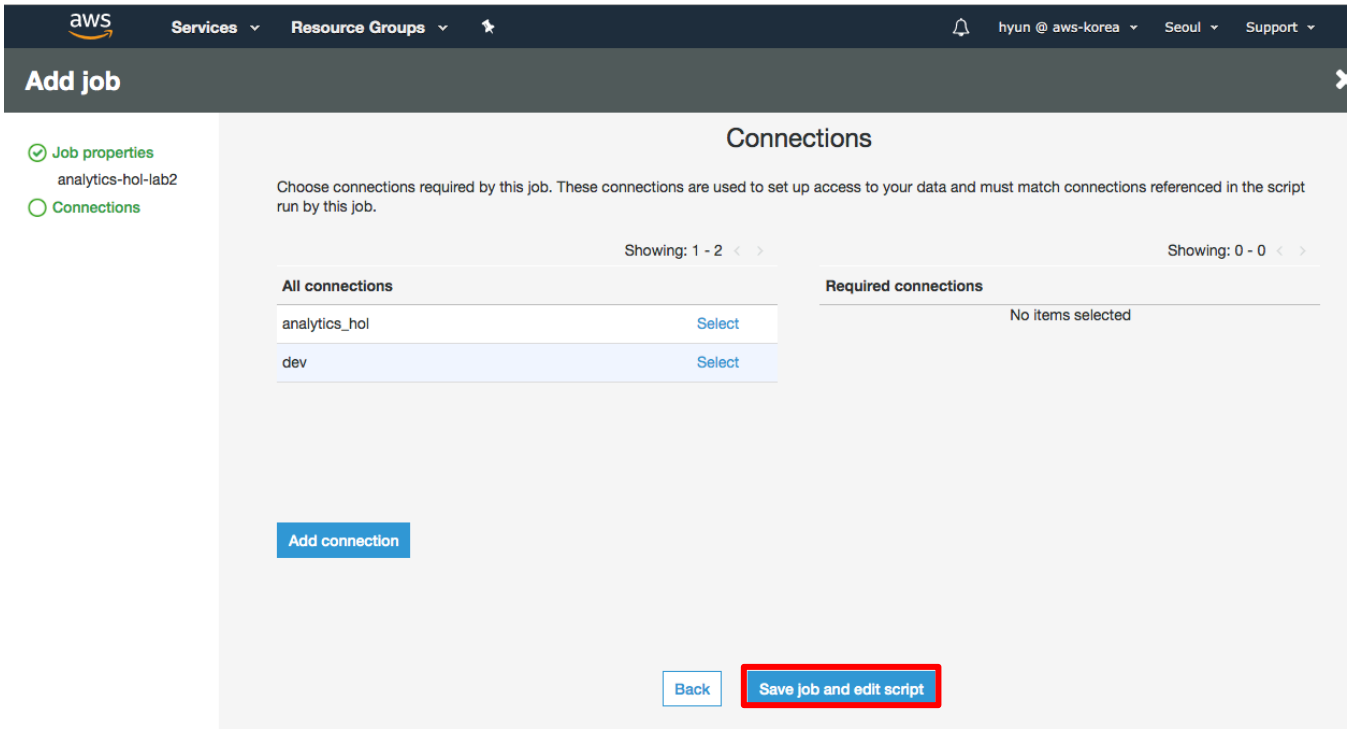


7-10. Name 에 analytics-hol-lab2, IAM role 에 AWSGlueServiceRole-Hol, Type 에 Spark, This job runs 에 A new script to be authored by you, ETL language 에 Python, Maximum capacity 에 2 를 입력하고 Next 버튼을 클릭합니다. (Maximum cacapcity 는 생성하는 Job Cluster 의 DPU 수를 지정합니다.)

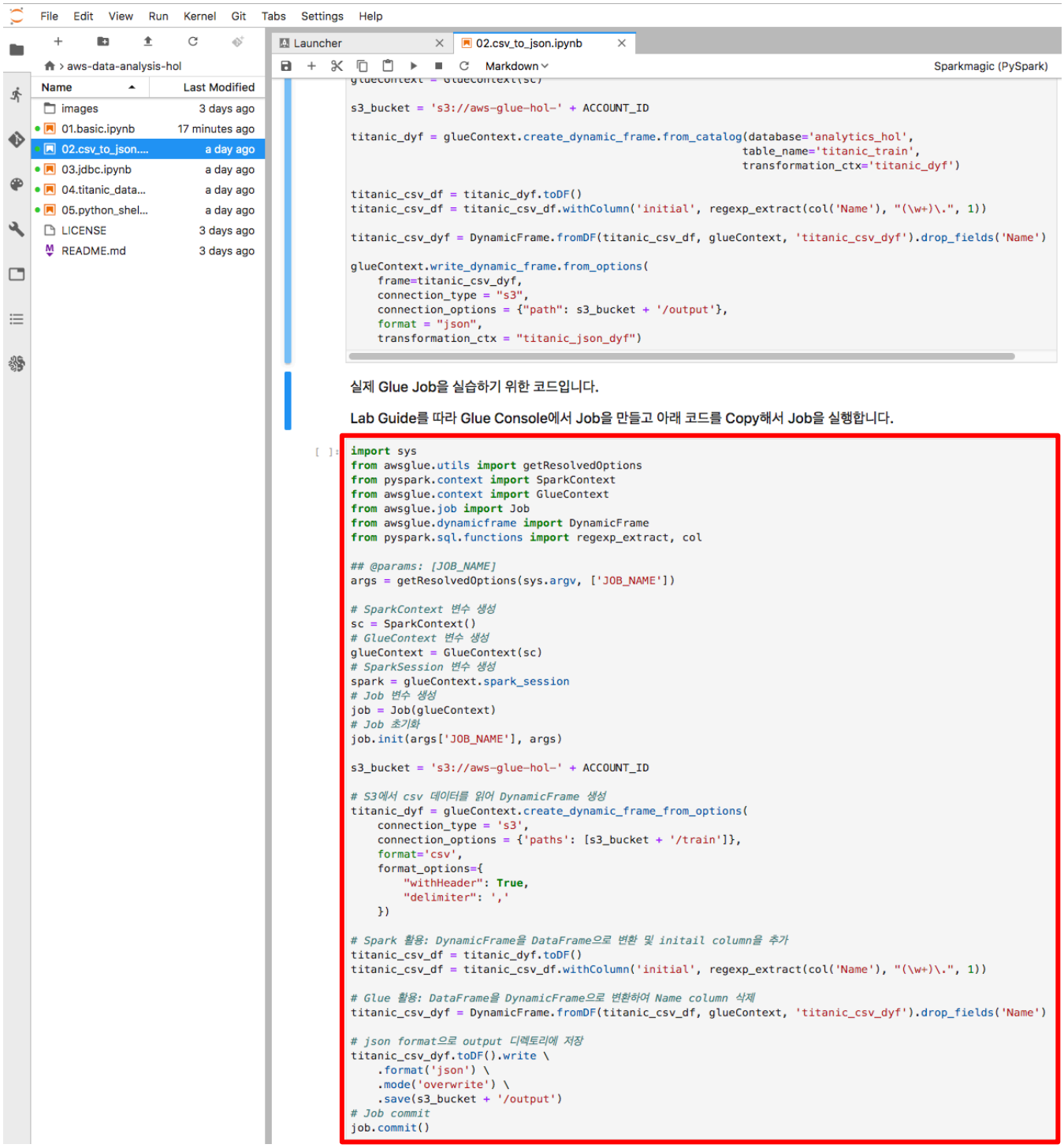


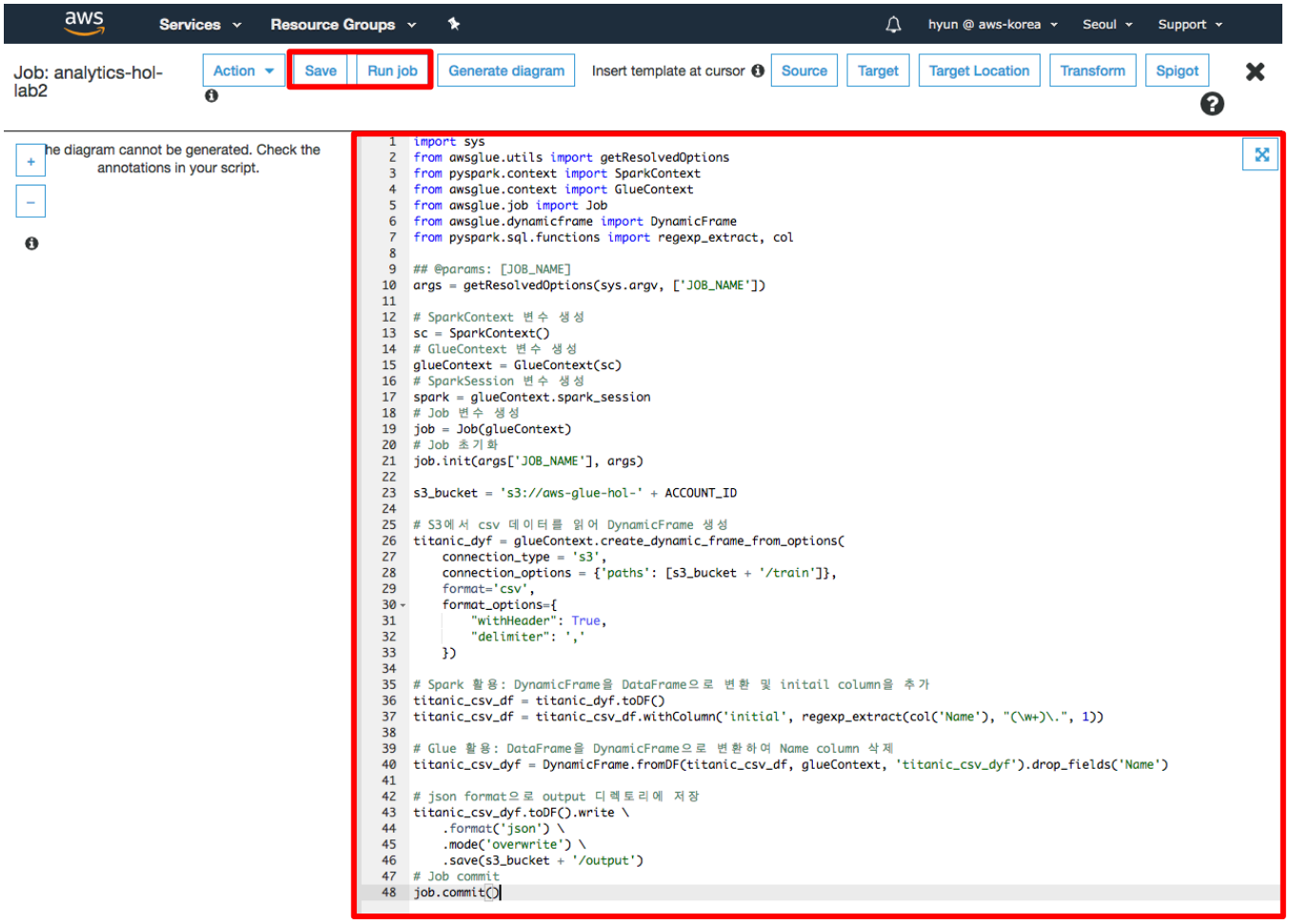

Security configuration, script libraries, and job parameters (optional)
Security configuration
None
The security configuration specifies how the script is encrypted using server-side encryption with AWS KMS-managed keys (SSE-KMS) or Amazon S3-managed encryption keys (SSE-S3).
Server-side encryption
Python library path
s3://bucket-name/folder-name/file-name
Dependent jars path
s3://bucket-name/folder-name/file-name
Referenced files path
s3://bucket-name/folder-name/file-name
Worker type
Standard
Maximum capacity
2
Max concurrency
1
Job timeout (minutes)
2880
Delay notification threshold (minutes)
Number of retries
0
Job parameters
Key
Type key...
Value
Type value...
Catalog options (optional)
Next

7-11. 기본 설정으로 두고 Save job and edit script 버튼을 클릭합니다.

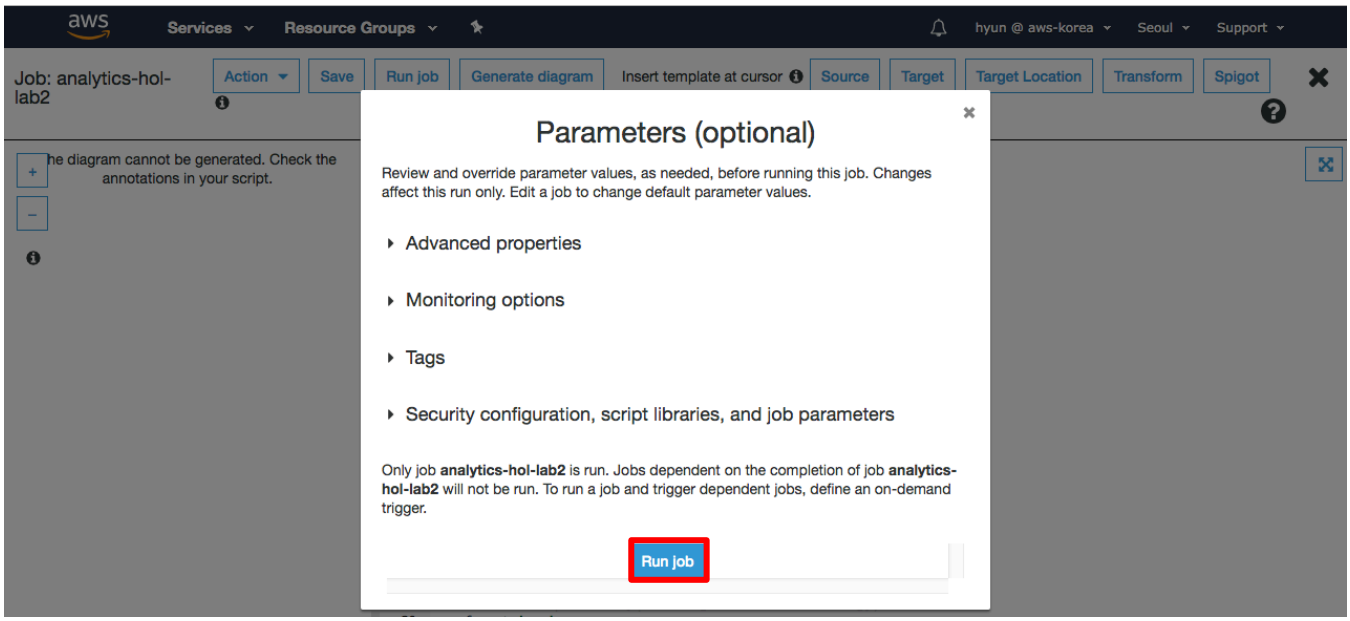


7-12. 02.csv_to_json.ipynb 노트북의 마지막 코드 Cell Script 를 복사합니다. Glue Job Editor 에 붙여 넣고 Save 버튼을 눌러 스크립트를 저장한 후, Run job 버튼을 클릭합니다.

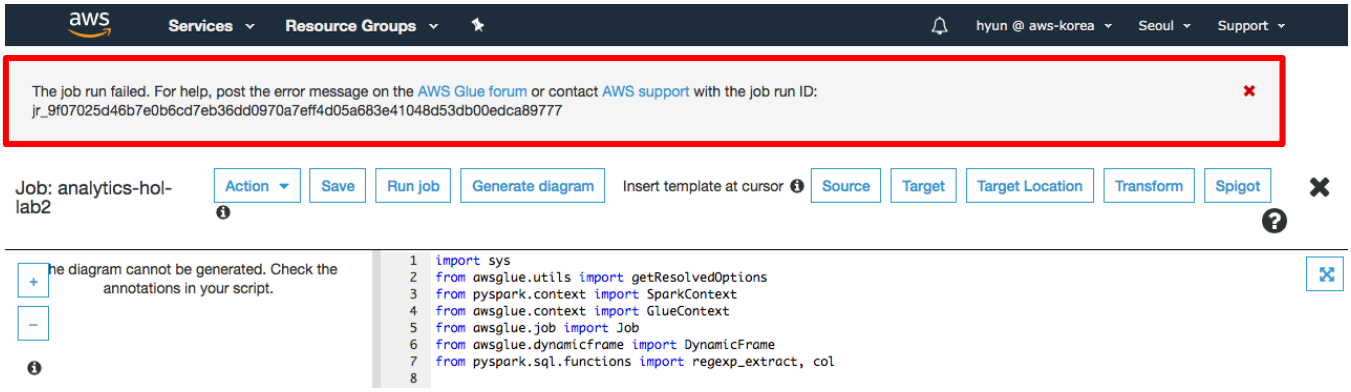




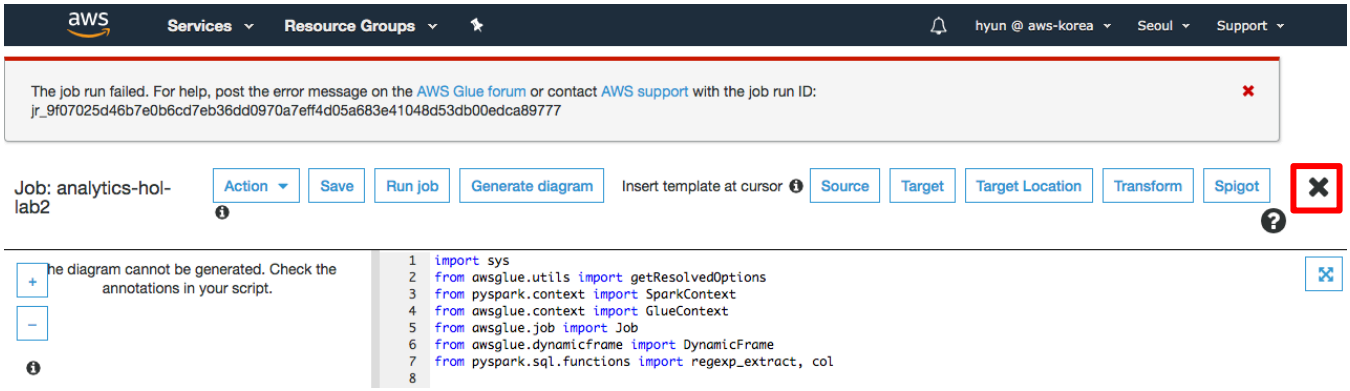
7-13. 다시 Run job 버튼을 클릭하여 Job 을 실행하고 실행이 완료될 때 까지 기다립니다.



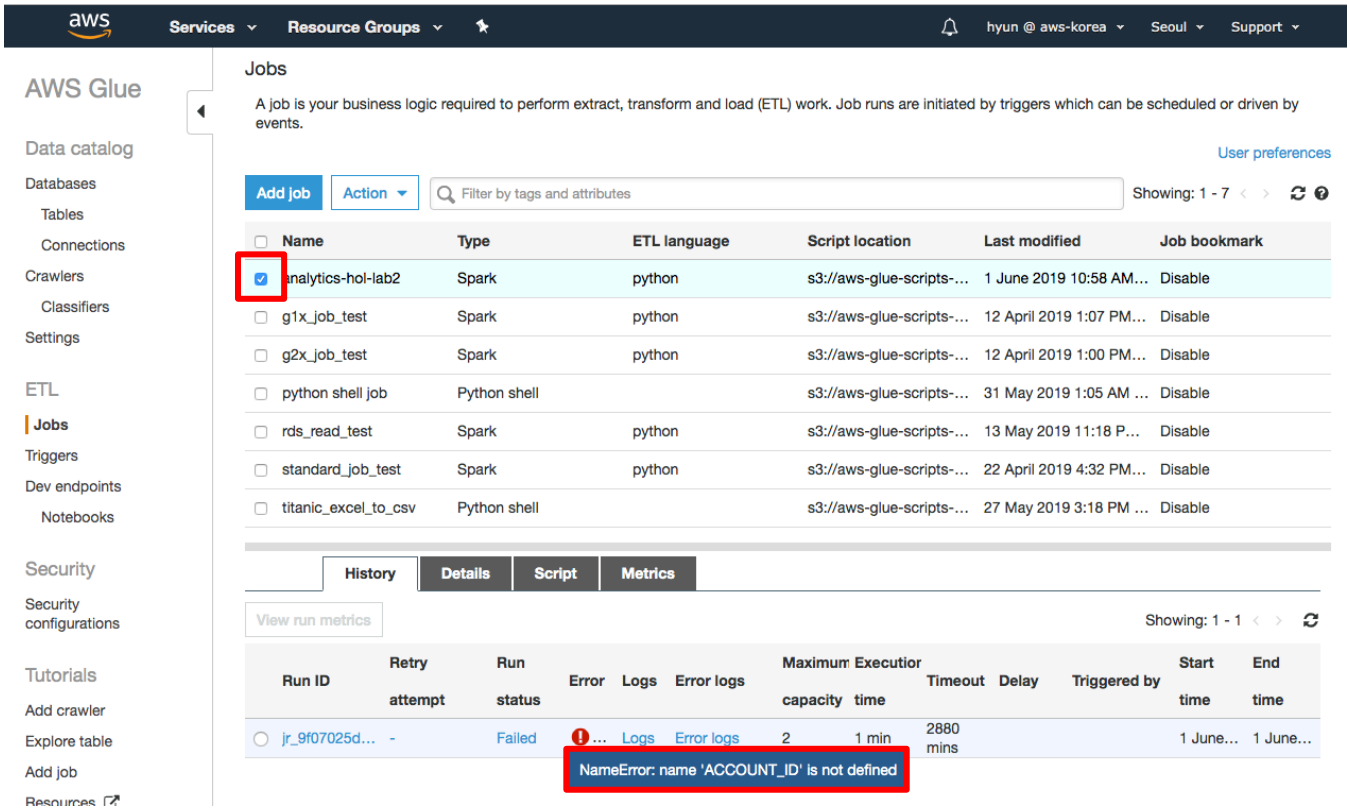
7-14. 다음과 같은 Error 를 내면서 Job 이 실패하는 것을 확인할 수 있습니다.



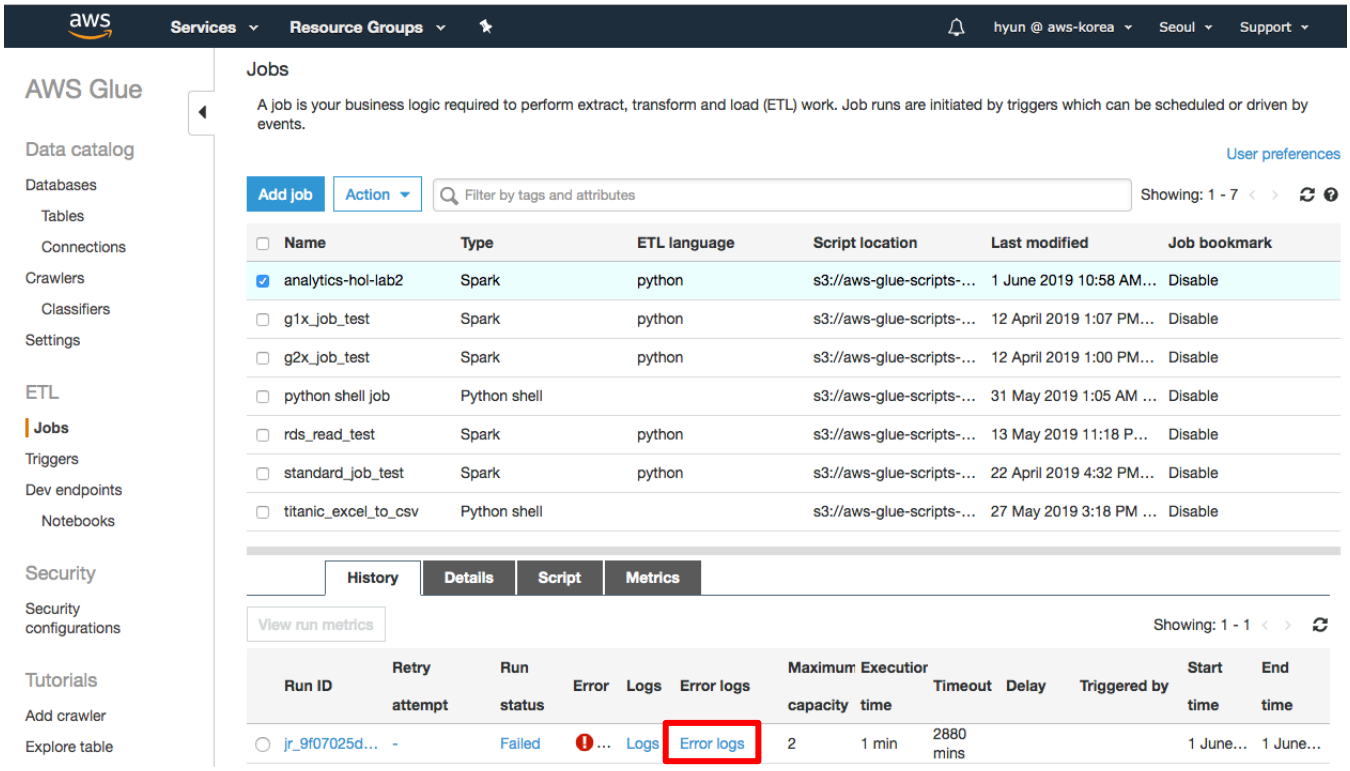
7-15. 에러 메세지 확인을 위해 화면 오른쪽 상단의 X 버튼을 클릭합니다.



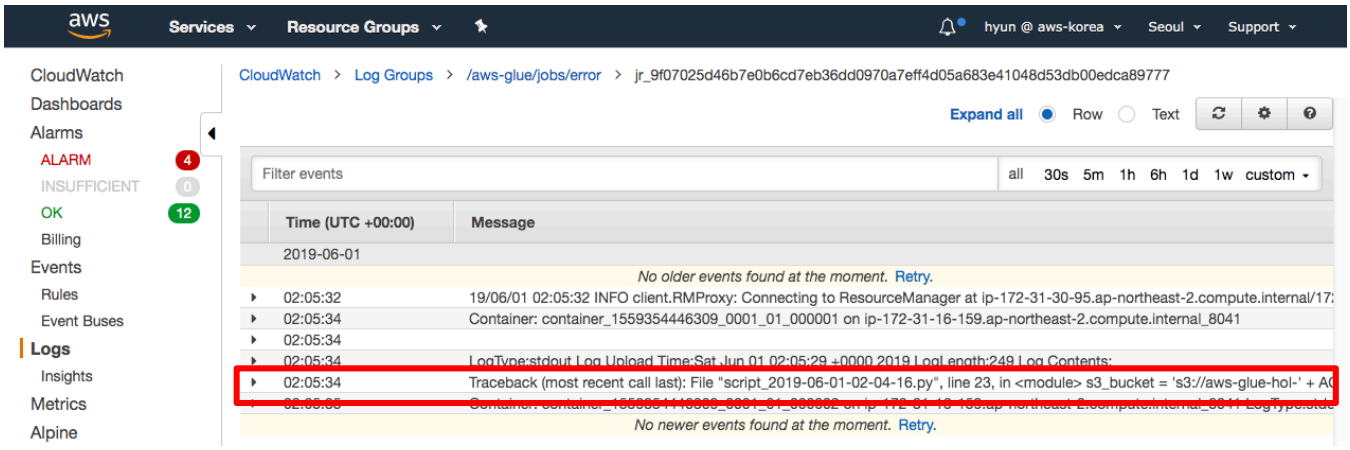
7-16. analytics-hol-lab2 job 을 선택하고 Error 부분에 빨간색 느낌표에 마우스를 올리면 다음 화면과 같이 'ACCOUNT_ID' is not defined 라는 에러 메세지를 확인할 수 있습니다.



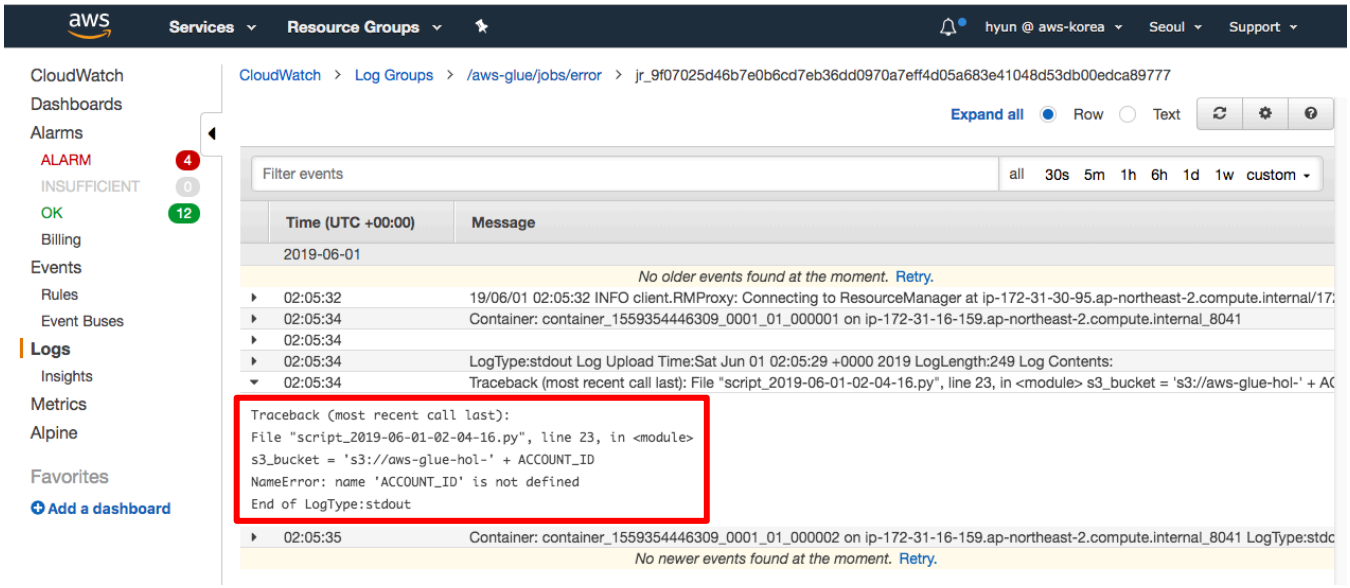
7-17. Error logs 를 클릭합니다.



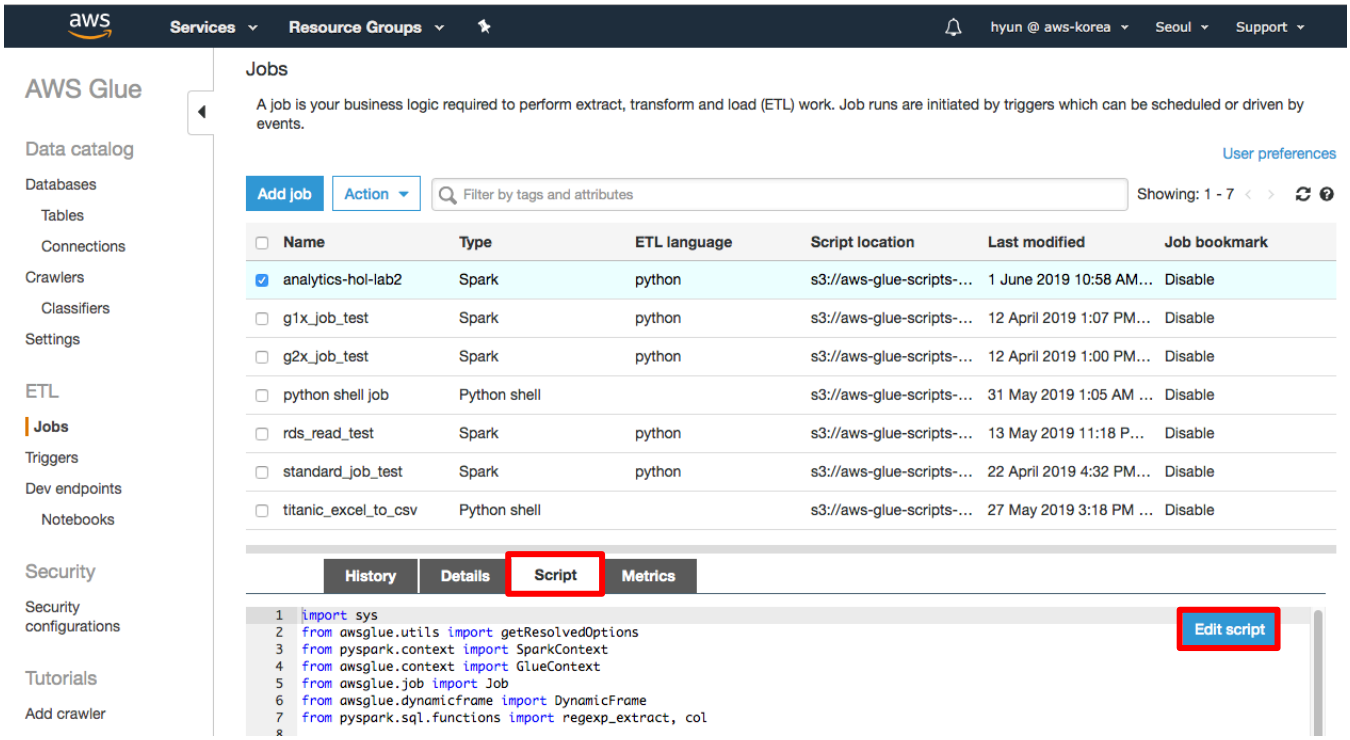
7-18. 로그를 클릭합니다.



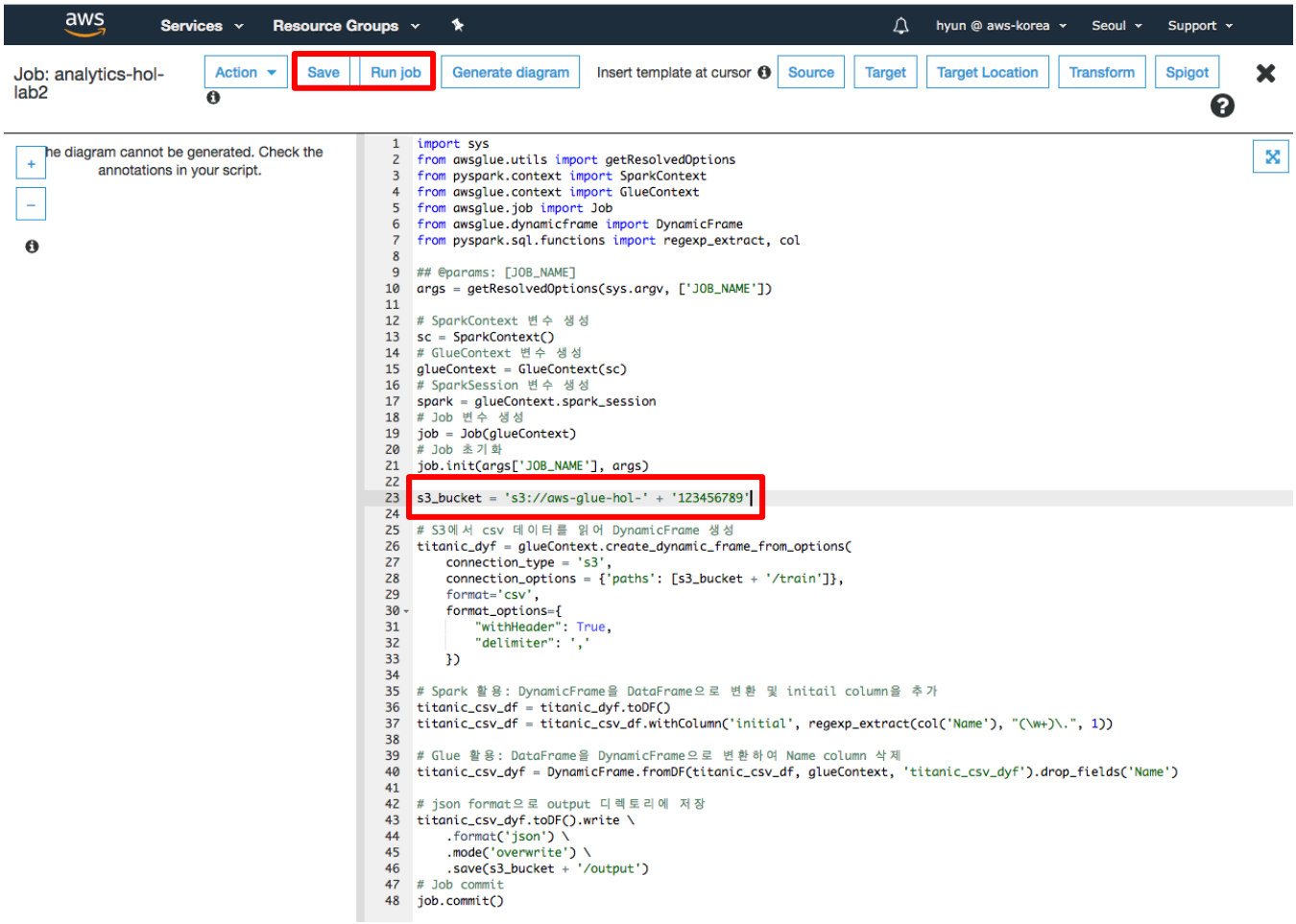
7-19. 로그 메세지에서 에러 메세지와 에러 위치를 확인합니다. Line 23 에서 에러가 발생한 것을 확인할 수 있고 'ACCOUNT_ID' is not defined 에러 메세지를 확인할 수 있습니다.



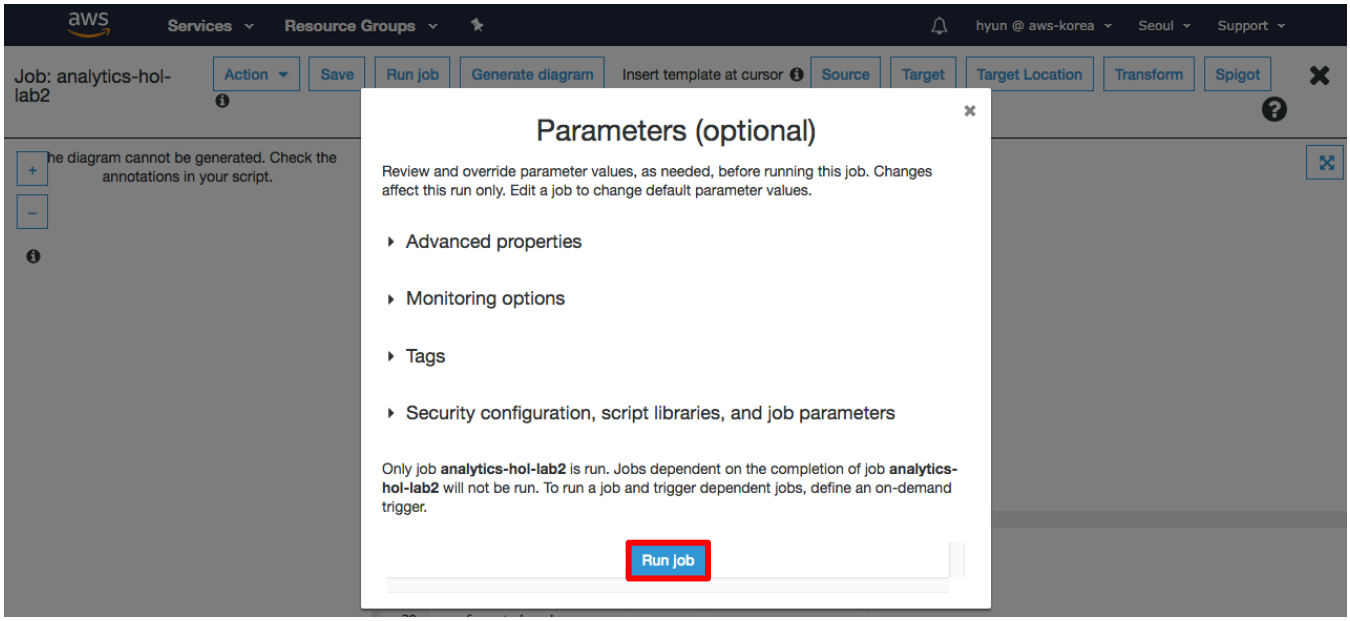
7-20. 다시 Glue 콘솔로 돌아가서 Script 탭 버튼을 클릭하고 오른쪽에 있는 Edit script 버튼을 클릭합니다.



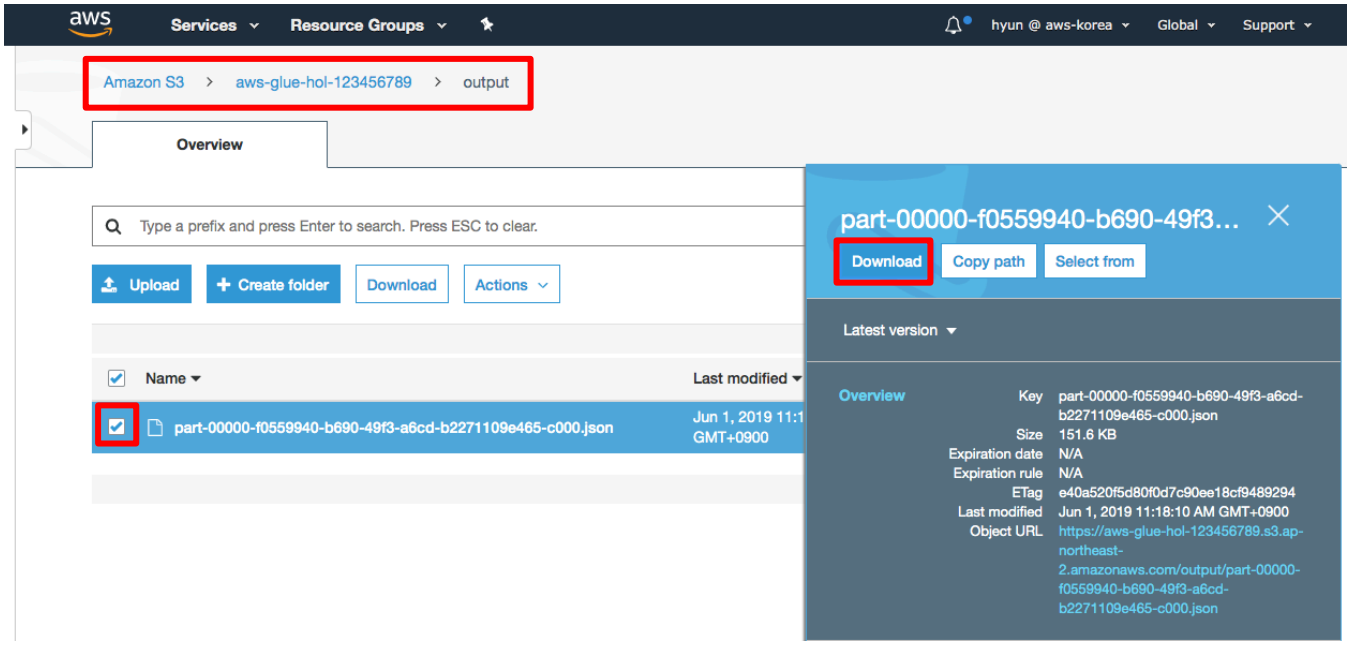
7-21. 아래 화면과 같이 line 23 의 ACCOUNT_ID 부분을 본인의 account_id 로 대체하고 Save 버튼을 클릭한 후 Run job 버튼을 클릭합니다.



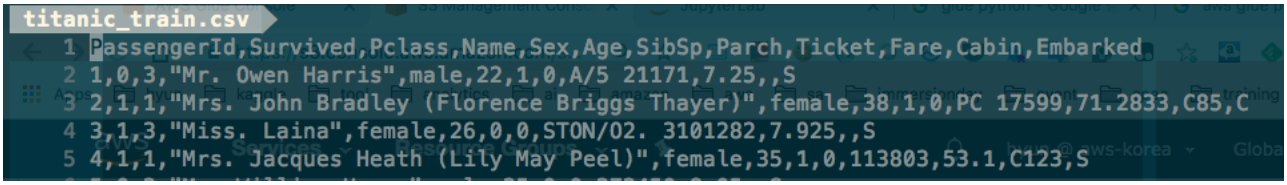
7-22. 다시 Run job 버튼을 클릭하여 Job 을 실행합니다.



7-23. 실행이 완료되면 S3 콘솔의 aws-glue-hol-[account_id] 버킷 output 디렉토리에 생성된 part-00000-* 로 시작되는 파일을 다운로드 받아 json 포맷 파일이 생성된 것을 확인합니다.



Before



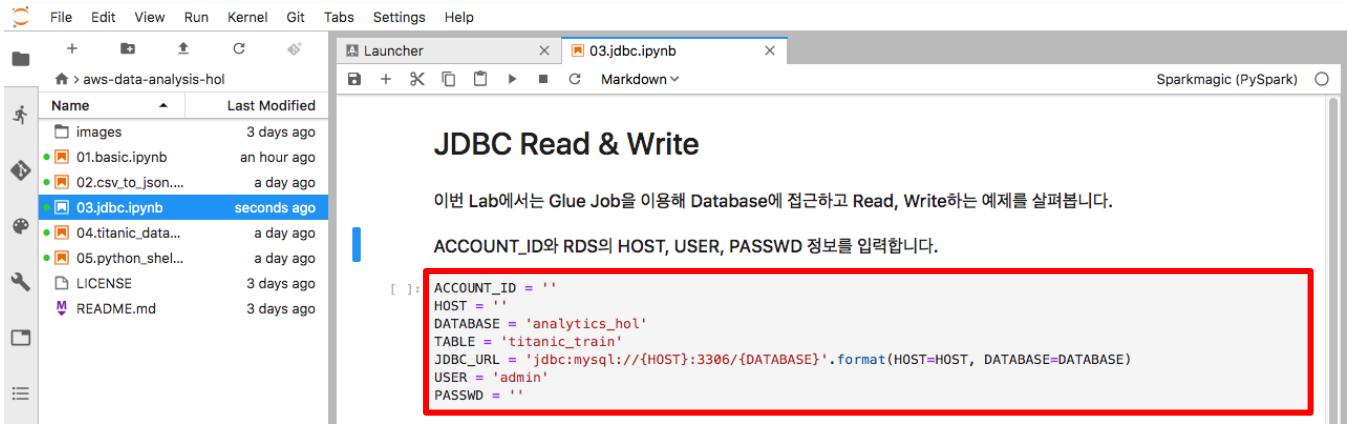
After

7-24. 생성된 Job 에 대한 스케쥴 관련 내용은 Lab2 실습 완료 후 설명 드리도록 하겠습니다.

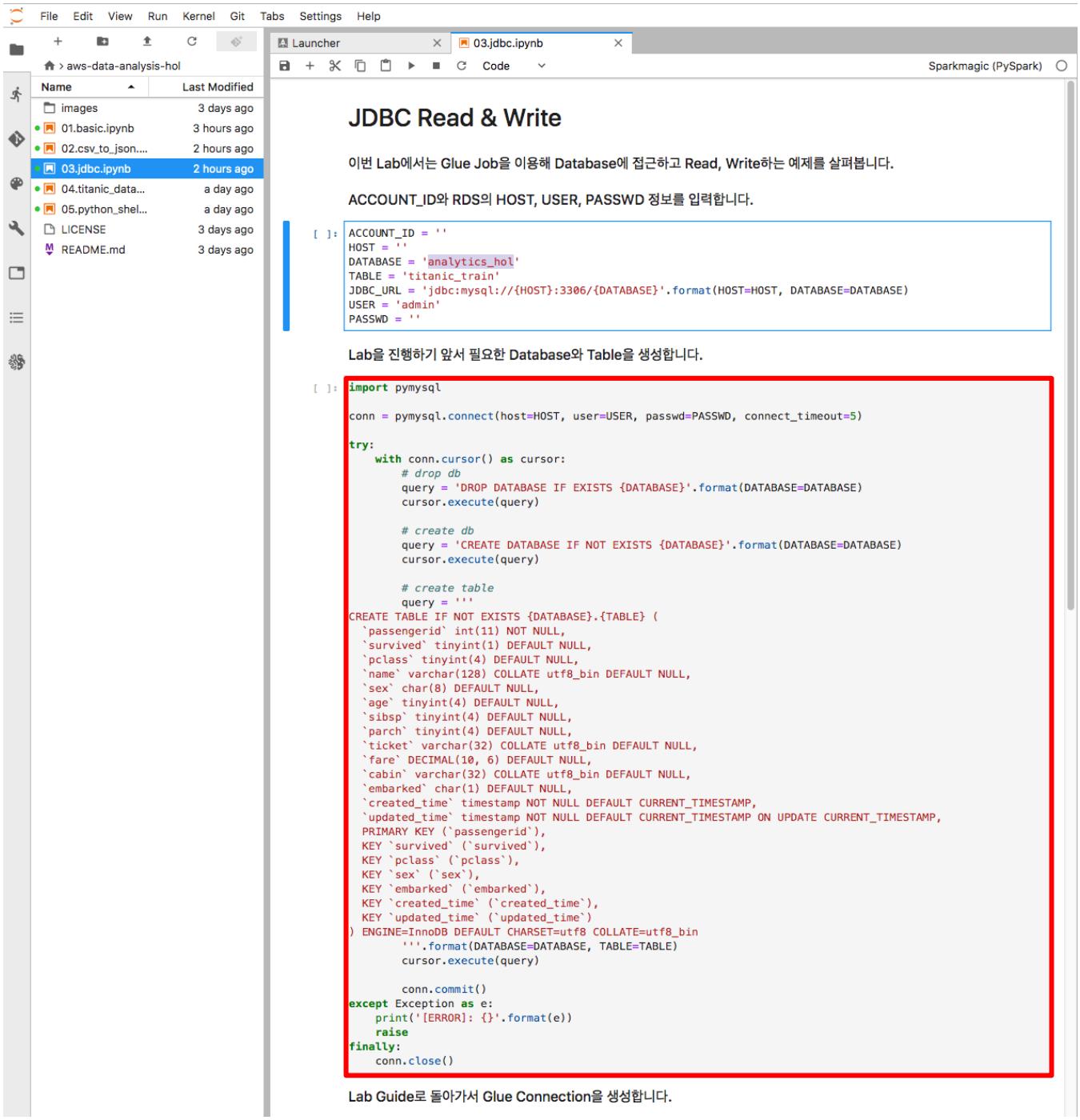
Lab3: Read and Write Database

Lab2 에서는 파일을 읽고 파일로 쓰는 실습을 해보았습니다. Lab3 에서는 Database 에 있는 데이터를 읽고 쓰는 실습을 해보도록 하겠습니다.

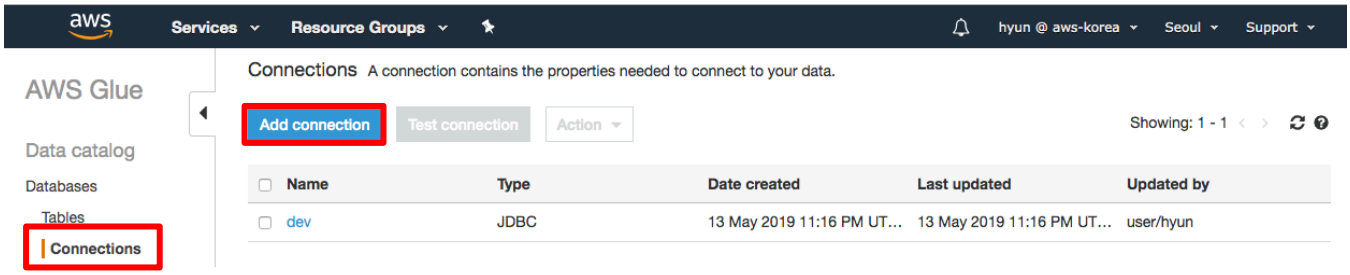
7-25. Lab3 진행을 위해 03.jdbc.ipynb 노트북 파일을 클릭하여 오픈합니다. ACCOUNT_ID 와 RDS 의 HOST, USER, PASSWD 정보를 입력하고 실행합니다. (DATABASE 와 TABLE 은 Jupyter Notebook 에 있는 이름을 그대로 사용합니다.)



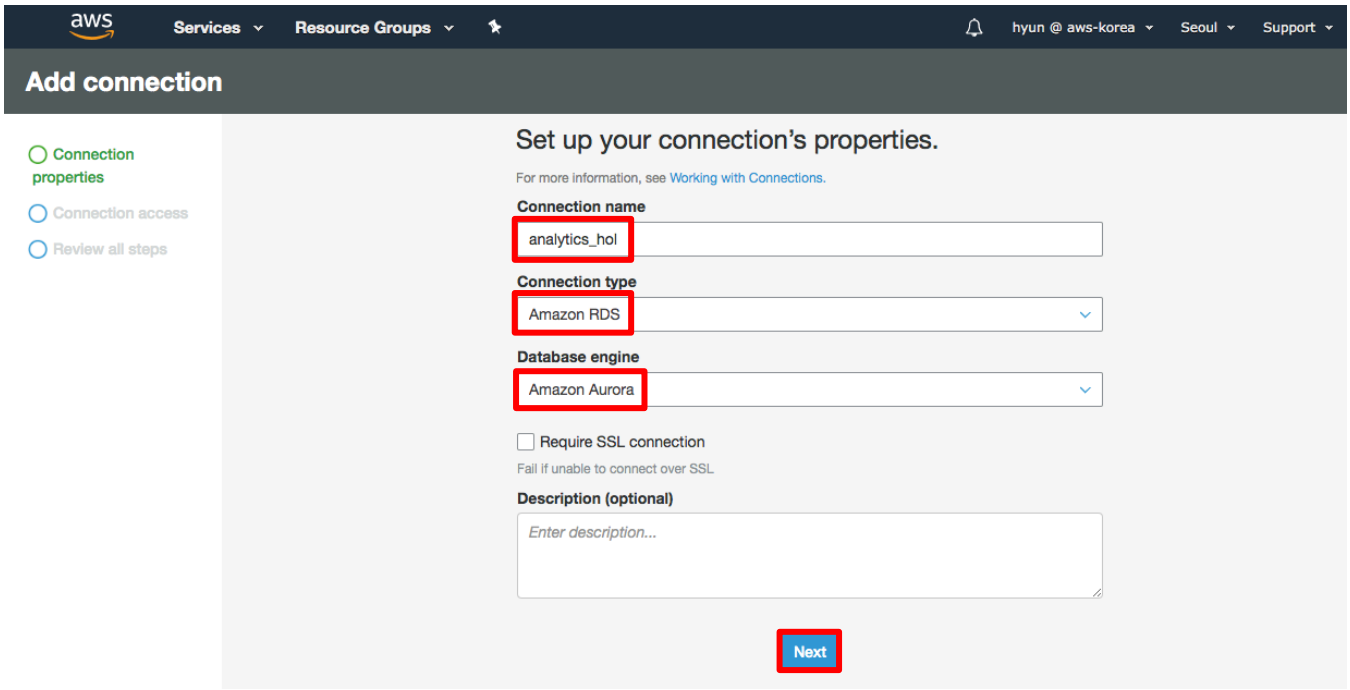
7-26. 2 번째 Cell 을 실행하여 RDS 에 Database 와 Table 을 생성합니다.



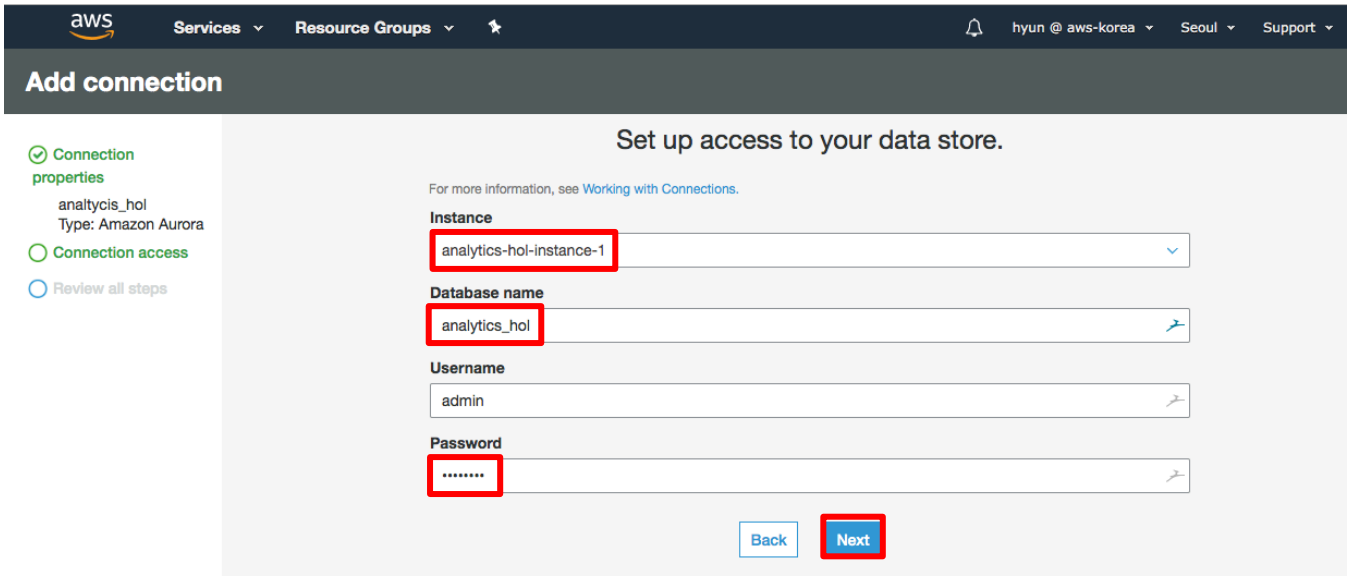
7-27. RDS 에 Database 와 Table 이 생성되었으면 Glue Connection 생성을 위해 Glue 콘솔로 이동 후 Add connection 버튼을 클릭합니다.



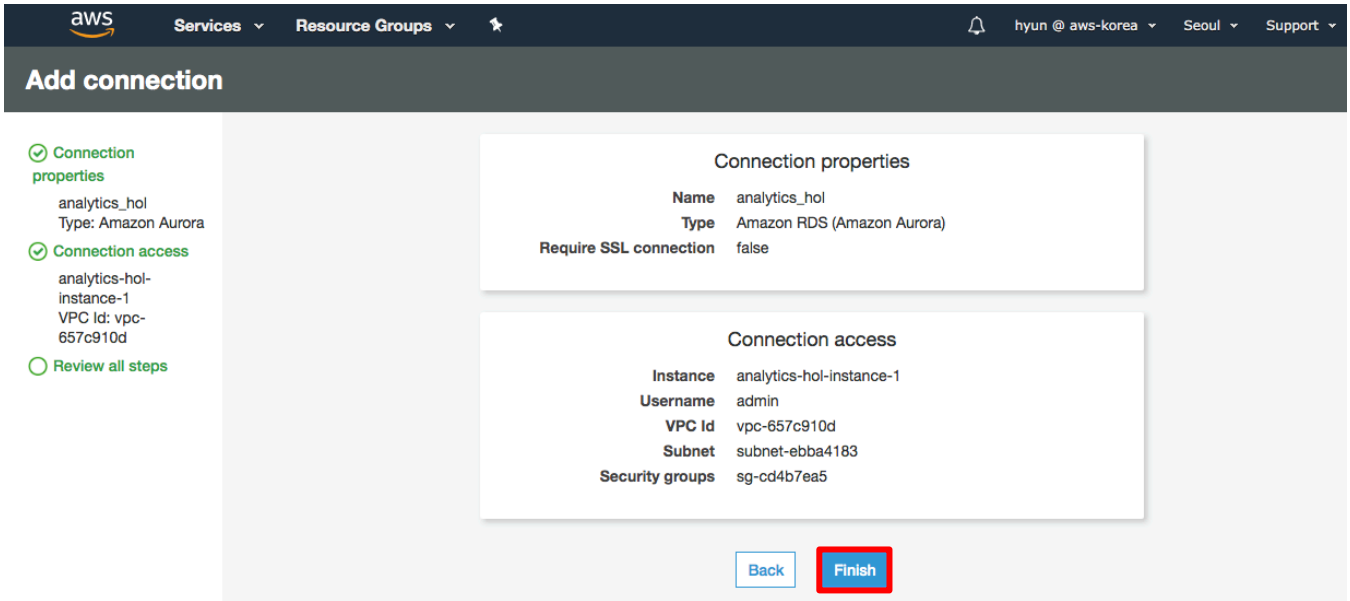
7-28. Connection name 에 analytics_hol, Connection type 에 Amazon RDS, Database engine 에 Amazon Aurora 를 입력하고 Next 버튼을 클릭합니다.

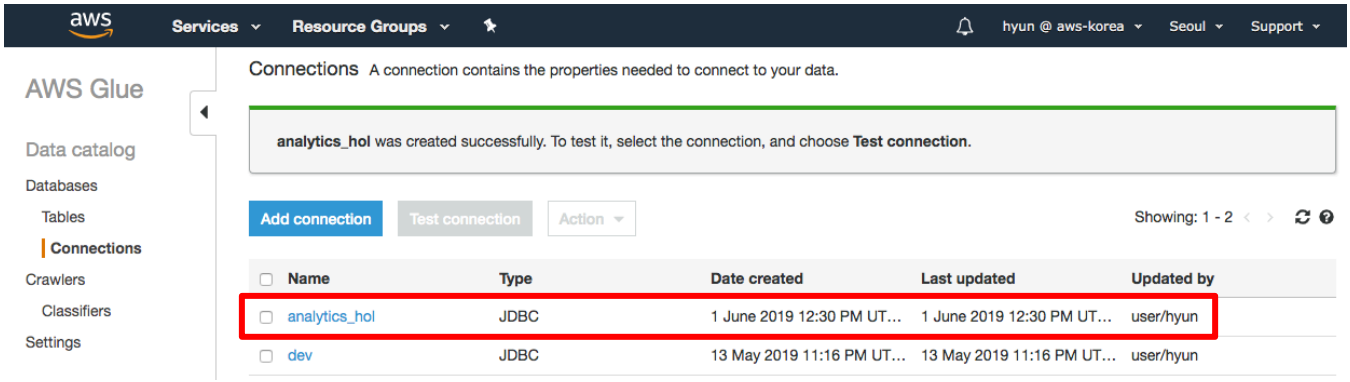


7-29. Instance 에 생성한 RDS analytics-hol-instance-1 을 선택하고, Database name 에 analytics_hol, Username admin 과 Password 를 입력한 후 Next 버튼을 클릭합니다.

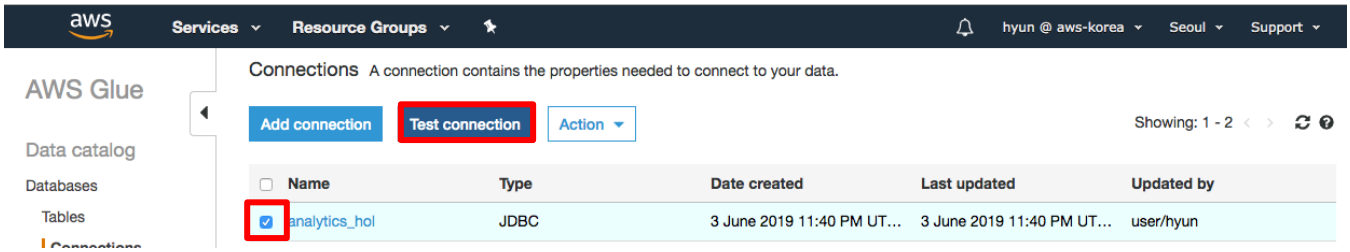


7-30. 지금까지 작성한 내용을 Review 한 후 Finish 버튼을 클릭하면 analytics_hol Connection 이 생성된 것을 확인할 수 있습니다.

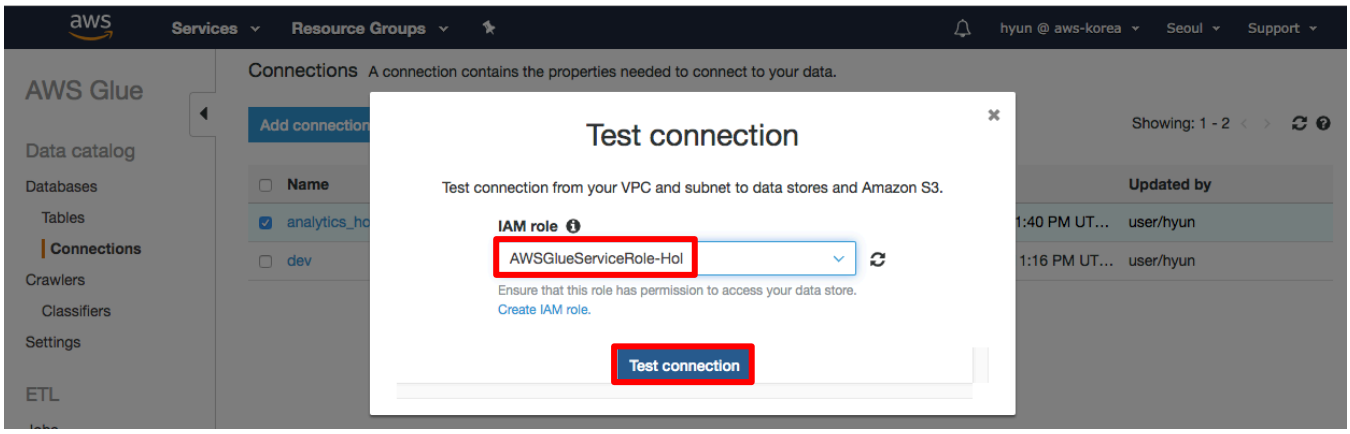


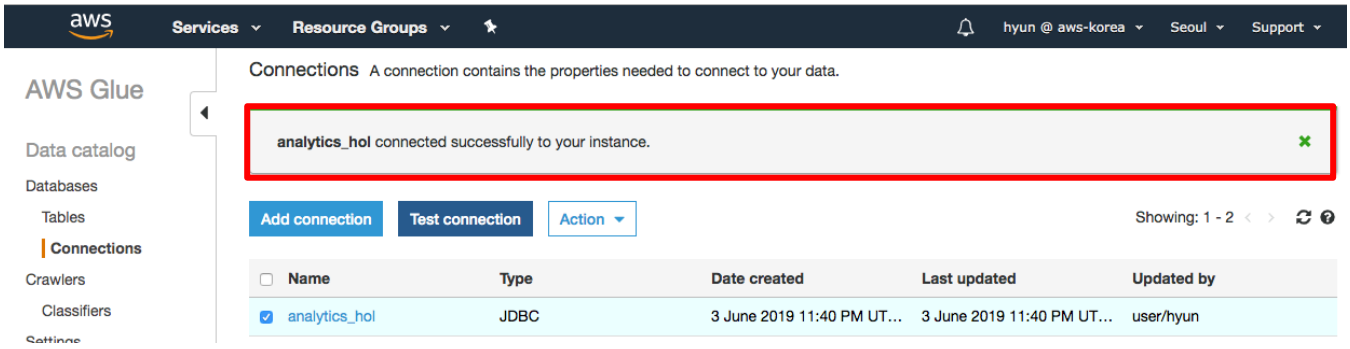


7-31. 다음은 지금 생성한 Connection 이 정상적으로 연결되는지 테스트하기 위해 analytics_hol connection 을 선택하고 Test connection 버튼을 클릭합니다.

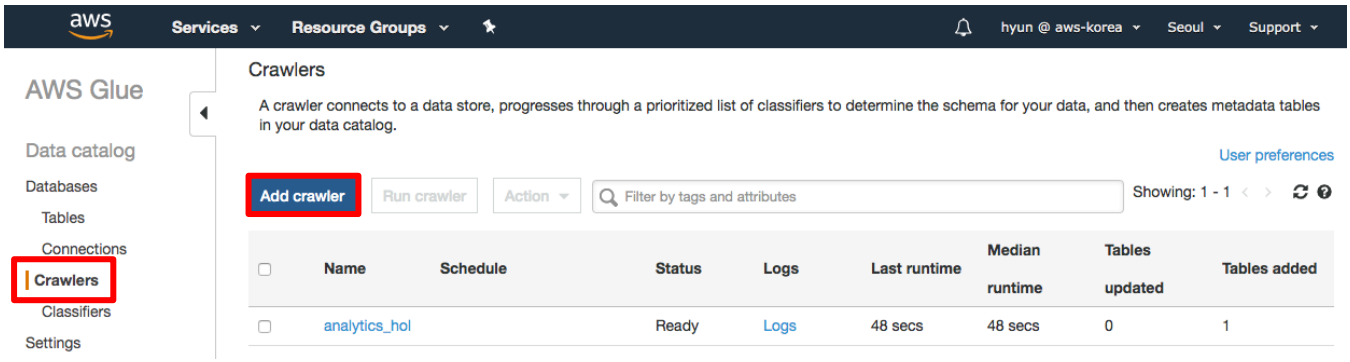


7-32. IAM role 에 AWSGlueServiceRole-Hol 을 선택하고 Test connection 버튼을 클릭합니다.

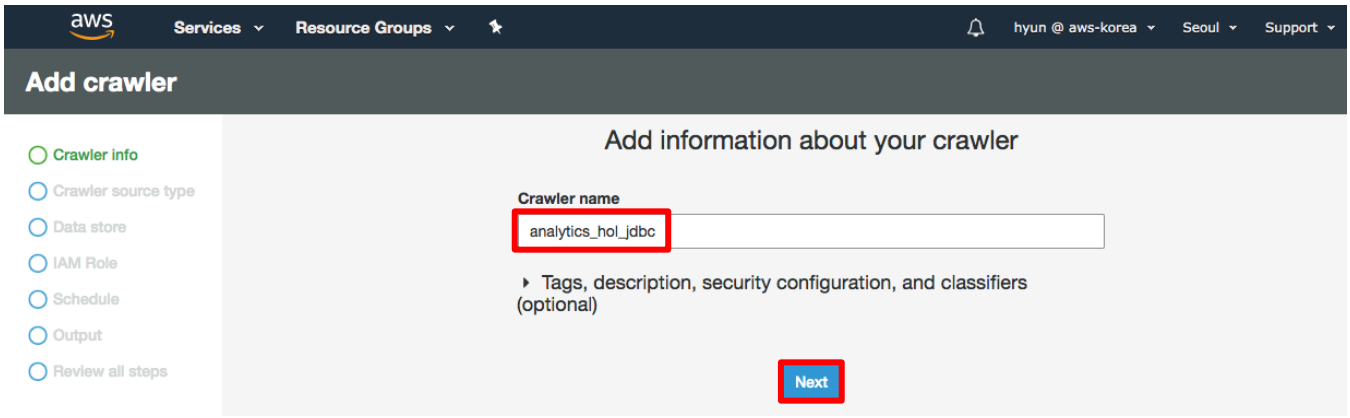




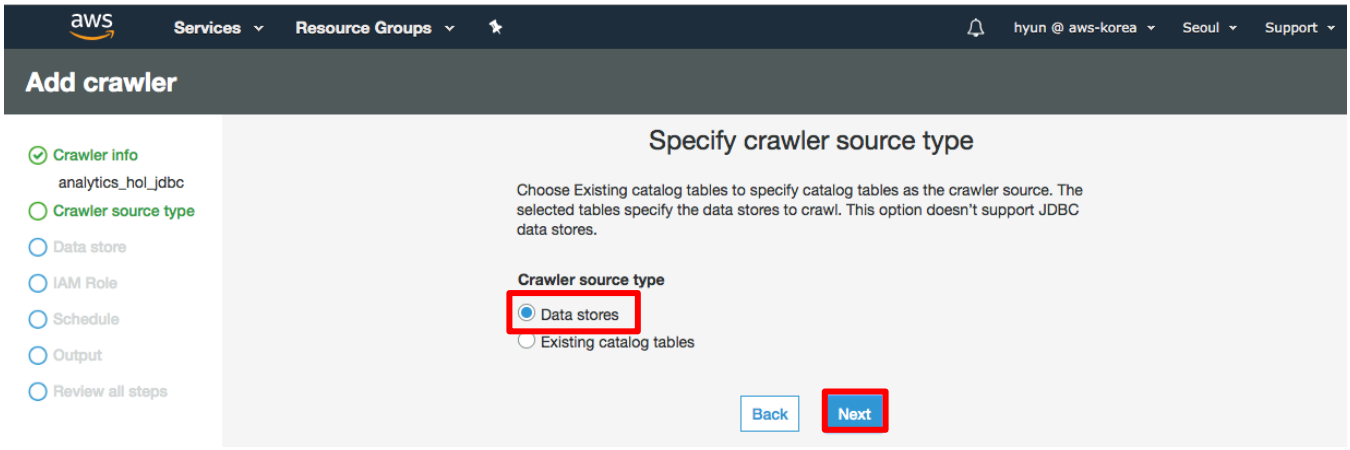
7-33. 다음은 지금 생성한 Connection 을 이용해서 RDS 에 생성된 Table 을 Glue Catalog 에 등록합니다. Glue 콘솔에 Crawlers 로 이동하여 Add crawler 버튼을 클릭합니다.



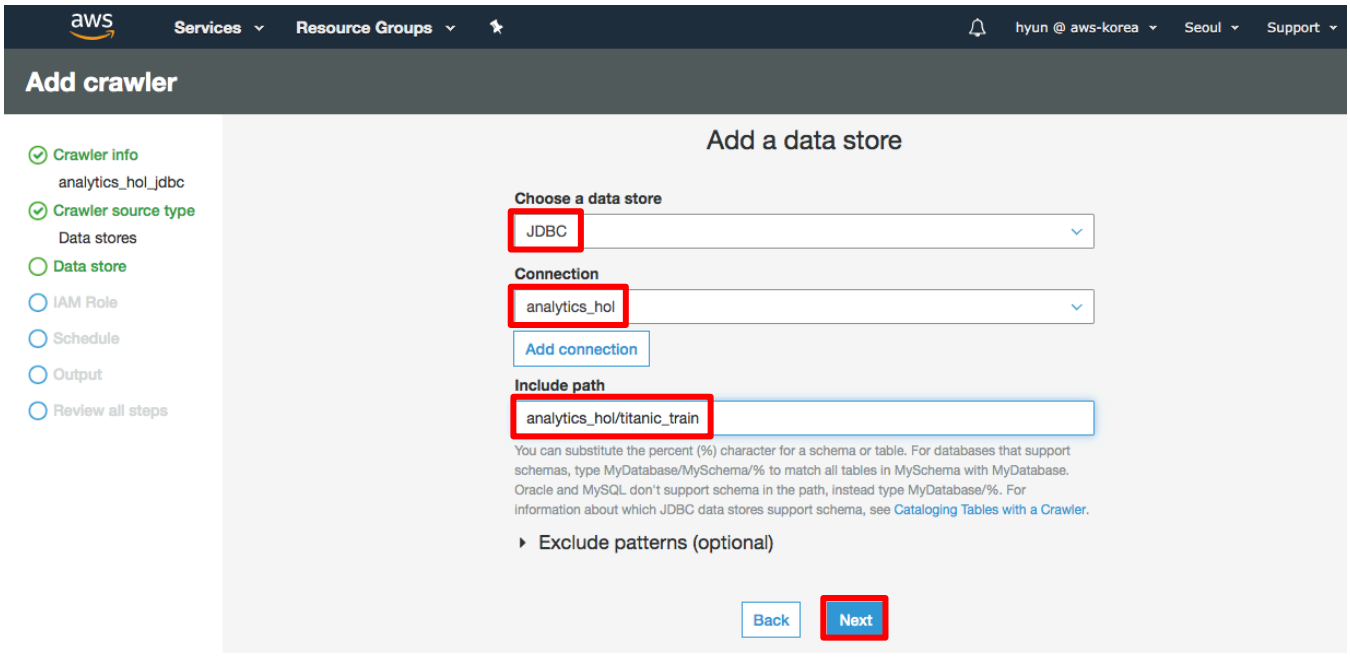
7-34. Crawler name 에 analytics_hol_jdbc 를 입력하고 Next 버튼을 클릭합니다.



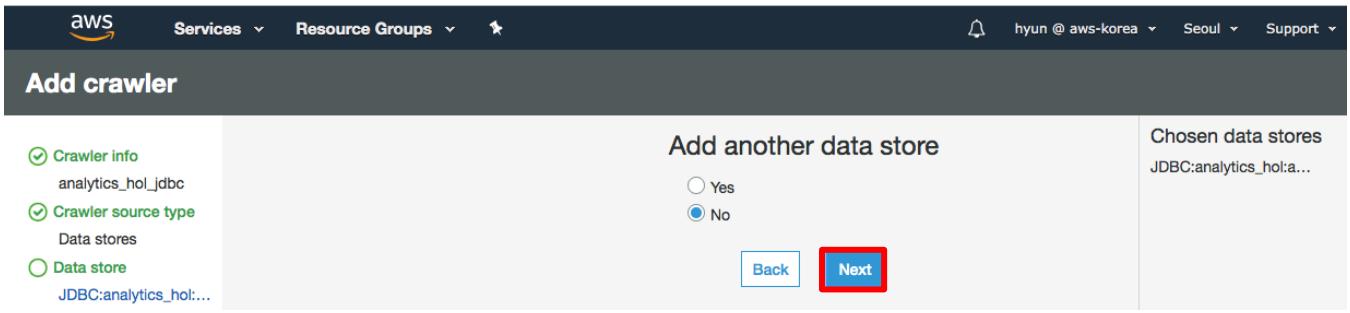
7-35. Crawler source type 을 기본 Data stores 로 두고 Next 버튼을 클릭합니다.



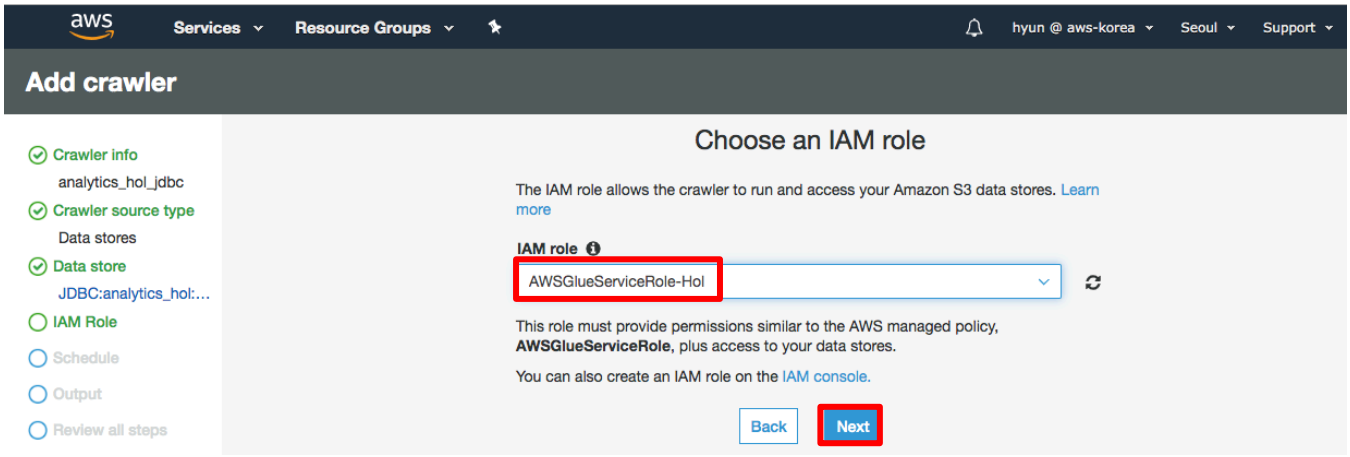
7-36. Choose a data store 에 JDBC, Connection 에 이전에 생성한 analytics_hol, include path 에 analytics_hol/titanic_train 을 입력하고 Next 버튼을 클릭합니다. Include path 는 database 와 table 이름을 의미합니다.



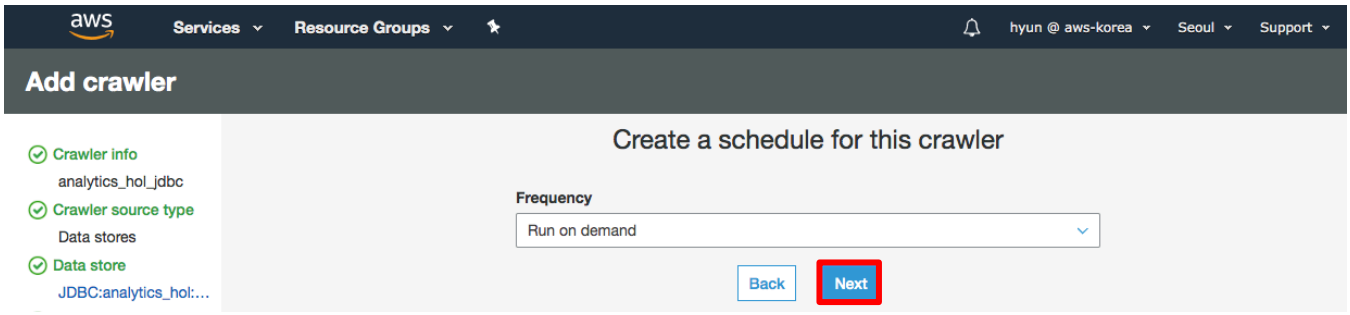
7-37. Add another data store 에서는 기본 No 인 상태에서 Next 버튼을 클릭합니다.



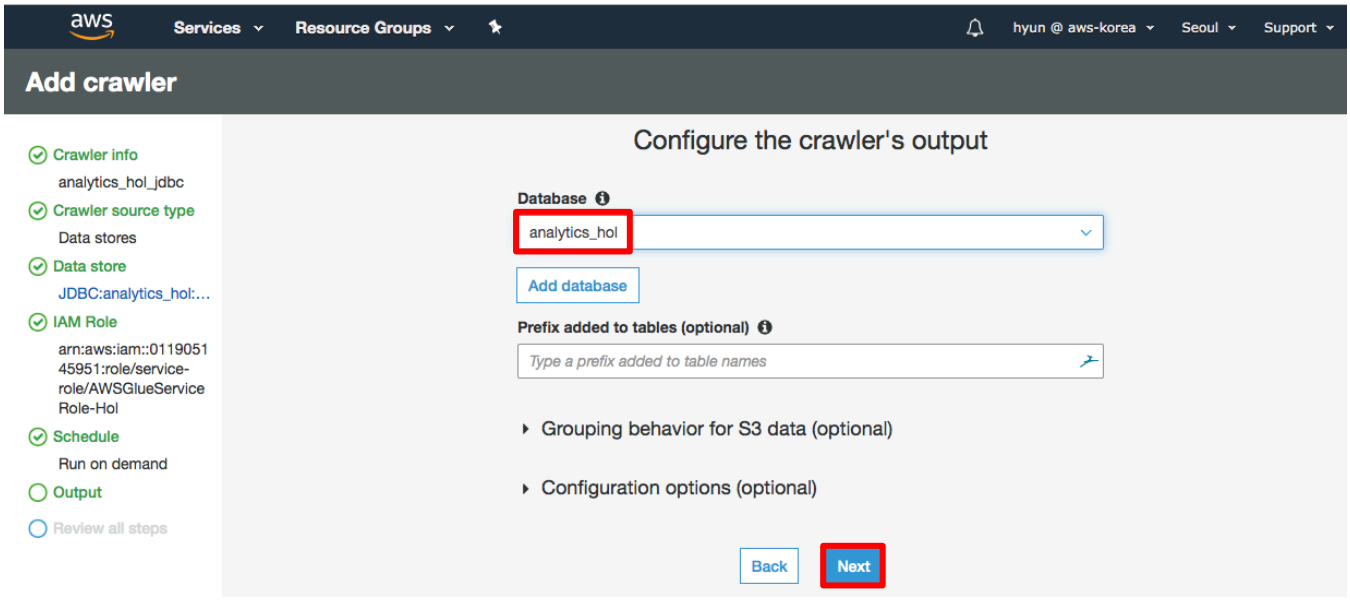
7-38. IAM role 에서 이전에 생성한 AWSGlueServiceRole-Hol 을 선택하고 Next 버튼을 클릭합니다.



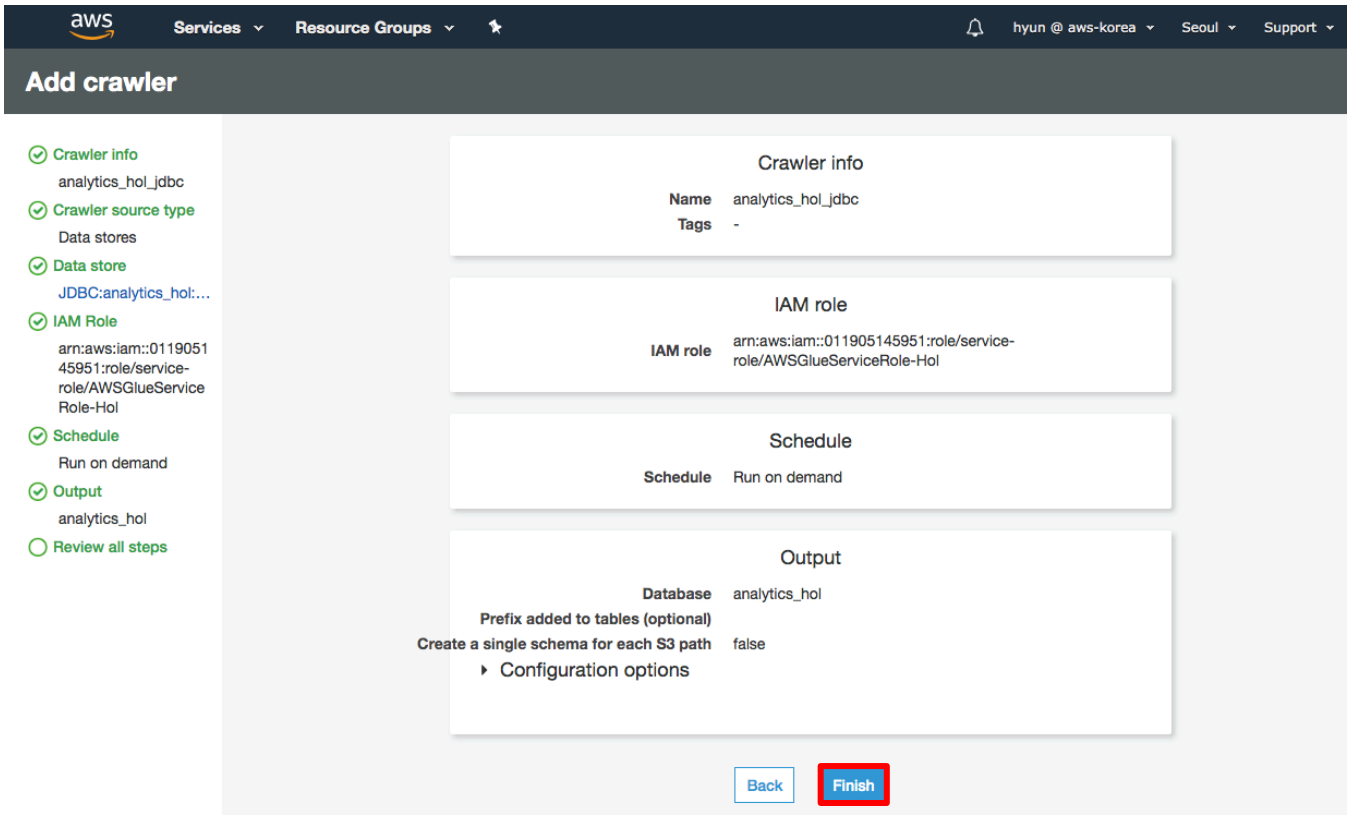
7-39. Schedule 설정은 Run on demand 상태로 두고 Next 버튼을 클릭합니다.

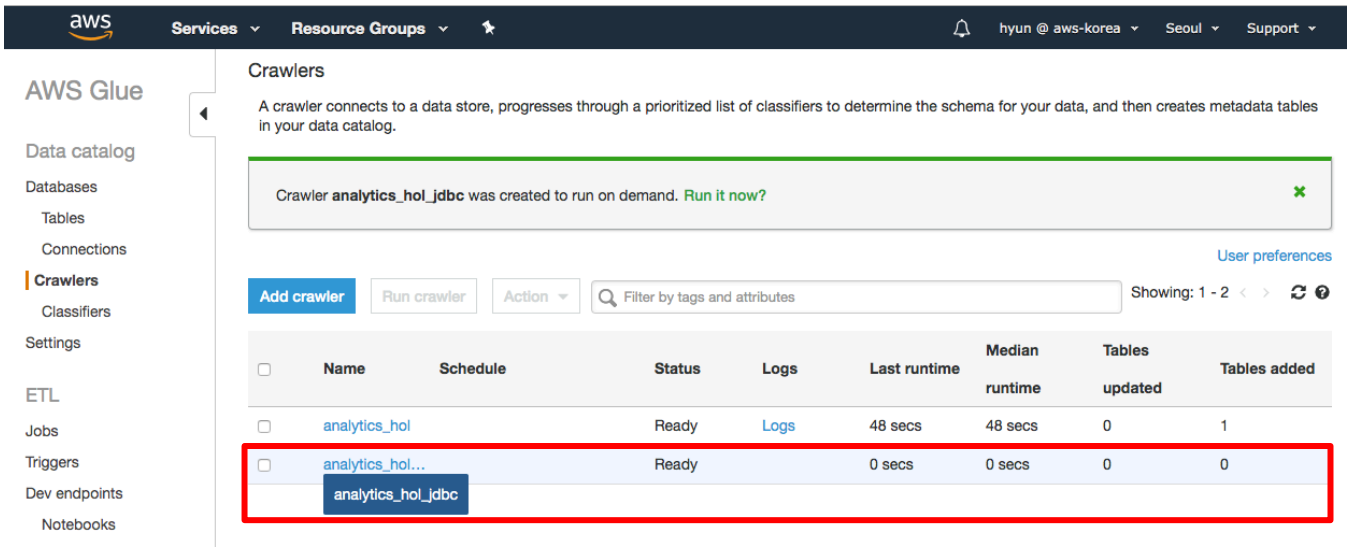


7-40. Glue Database 는 analytics_hol 을 선택하고 Next 버튼을 클릭합니다.

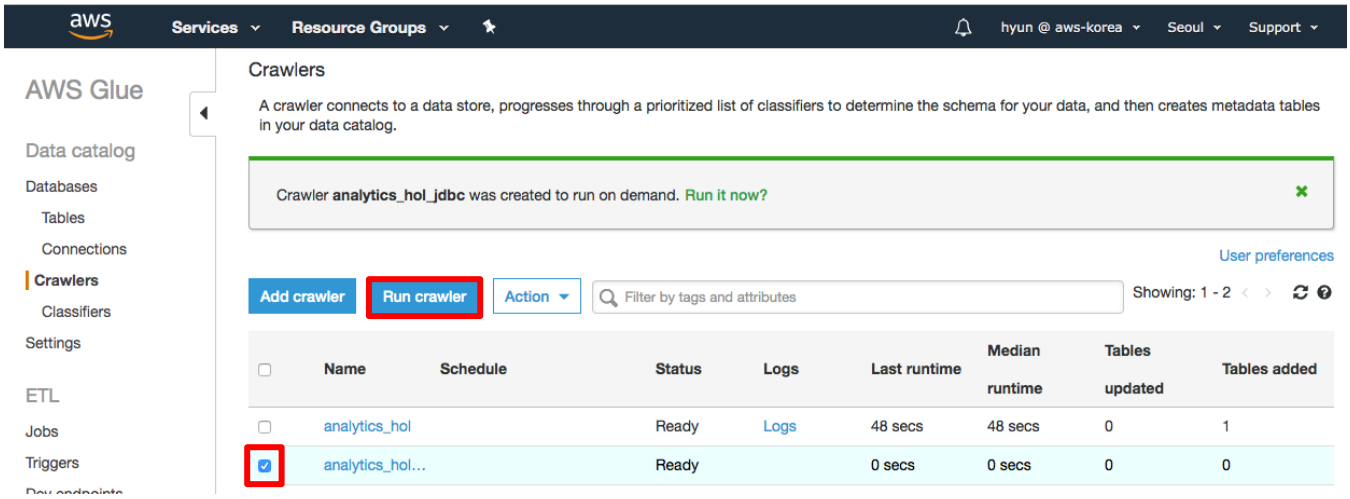


7-41. Review 후 Finish 버튼을 클릭하면 analytics_hol_jdbc crawler 가 정상적으로 생성된 것을 확인할 수 있습니다.

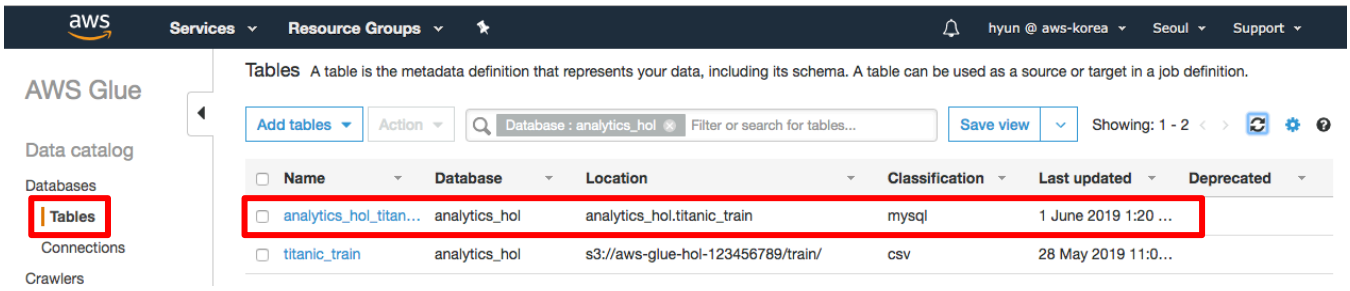




7-42. analytics_hol_jdbc crawler 를 선택하고 Run crawler 를 수행합니다.



7-43. crawler 수행이 완료된 후 Glue 콘솔의 catalog 의 Tables 로 이동해서 analytics_hol_titanic_train table 이 정상적으로 생성된 것을 확인합니다. 이제 Glue Job 에서 Glue Catalog 를 통해 JDBC Database 접근이 가능해졌습니다.

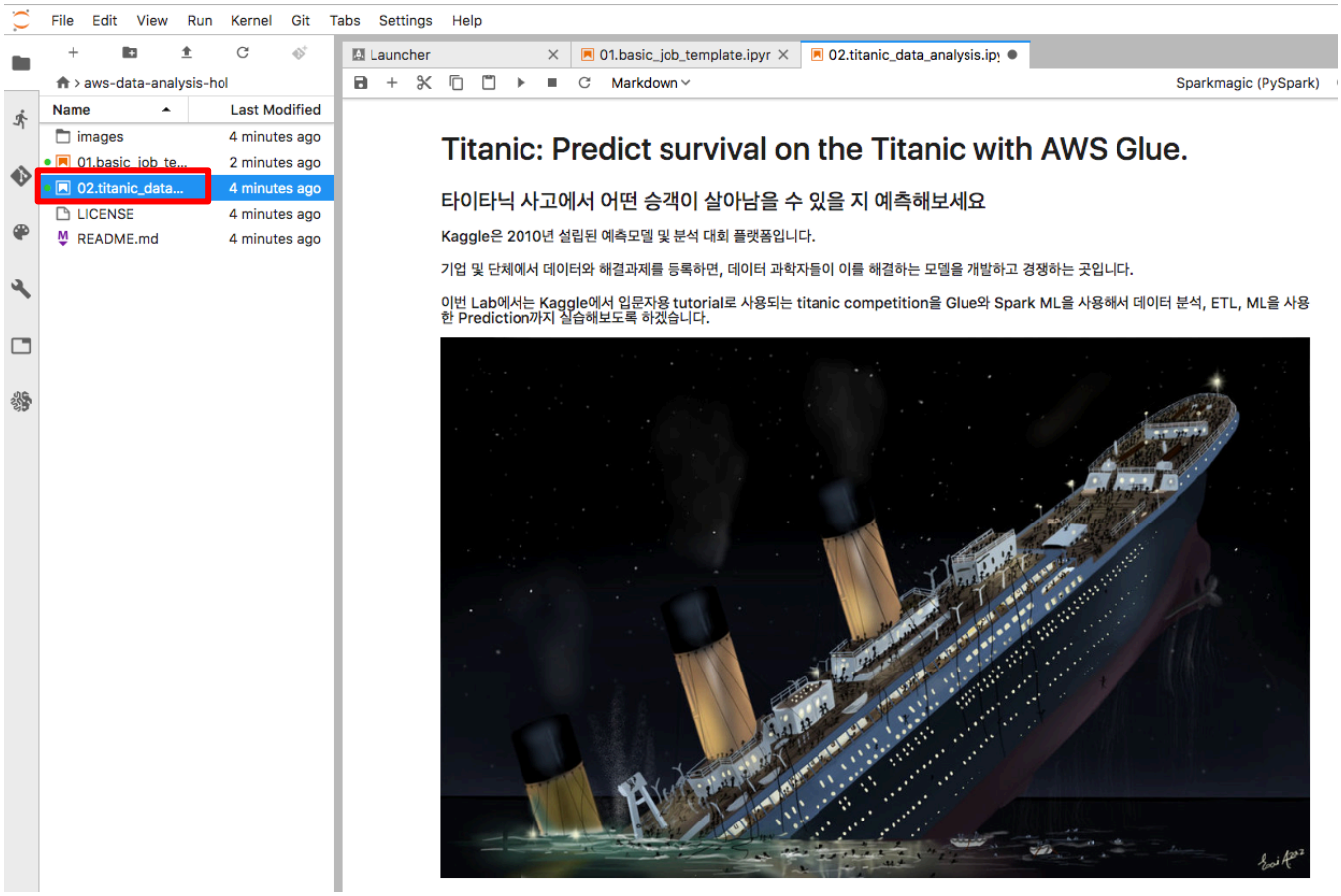


7-44. Jupyter Notebook 으로 돌아가 이후 'JDBC Write with Glue & Spark API', 'JDBC Read with Glue API', 'JDBC Read with Spark API' Lab 을 진행합니다.

Lab4: Predict survival on the Titanic with AWS Glue

이번 Lab 에서는 실제 빅데이터를 분석하는 과정에서 Glue 를 어떻게 활용할 수 있는지 kaggle titanic competition dataset 을 가지고 분석, ETL, Feature Enginnering 및 Spark ML 을 활용한 Modeling 까지 실습을 해보도록 하겠습니다.

7-45. Lab4 진행을 위해 04.titanic_data_analysis.ipynb 노트북 파일을 클릭하여 오픈합니다.

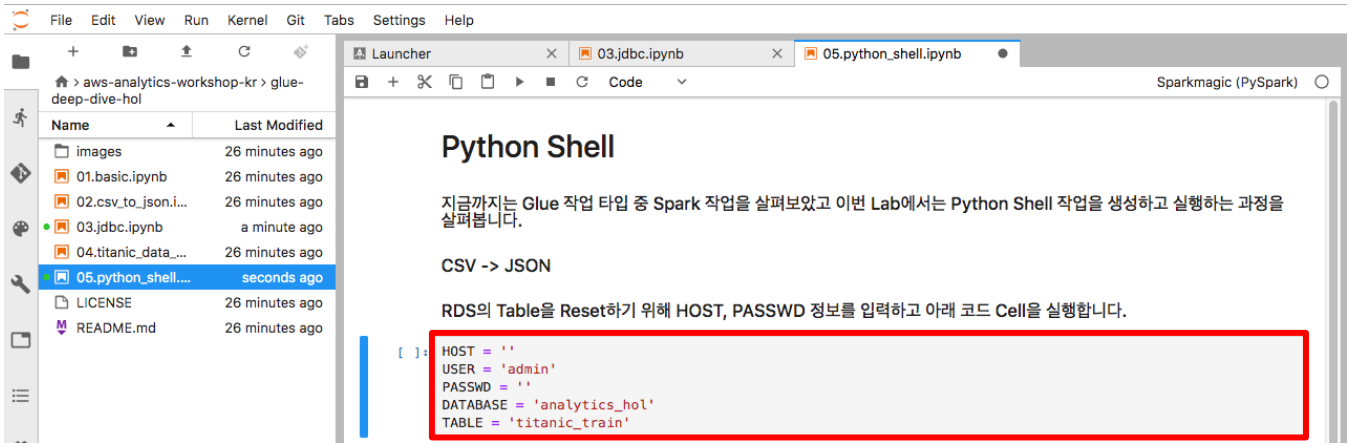


7-46. 노트북 가이드에 따라 Lab 을 진행합니다.

Lab5: Python Shell Job

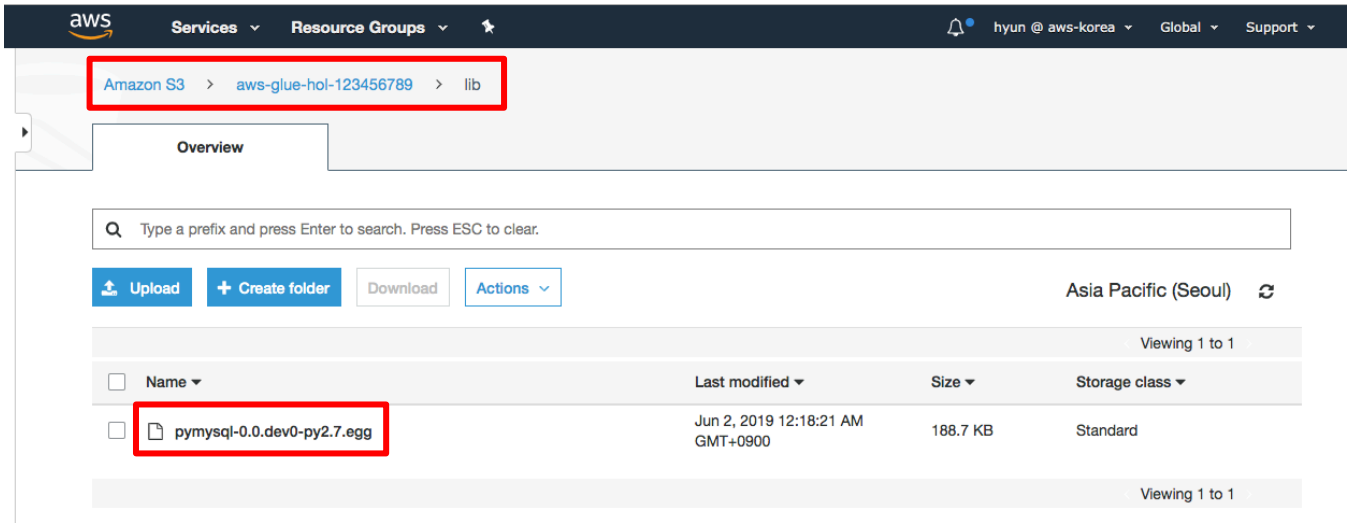
지금까지는 작업 타입 중 Spark 과 관련된 실습을 해보았습니다. 이번 Lab5 에서는 장비 한대에서 처리 가능한 작은 데이터를 다룰 때 사용할 수 있는 Python Shell 작업 타입을 실습 해보도록 하겠습니다.

7-47. Lab5 진행을 위해 05.python_shell.ipynb 노트북 파일을 클릭하여 오픈하고 RDS 의 Table 을 Reset 하기 위해 HOST, USER, PASSWD 정보를 입력하고 첫 번째, 두 번째 코드 Cell 을 실행합니다.

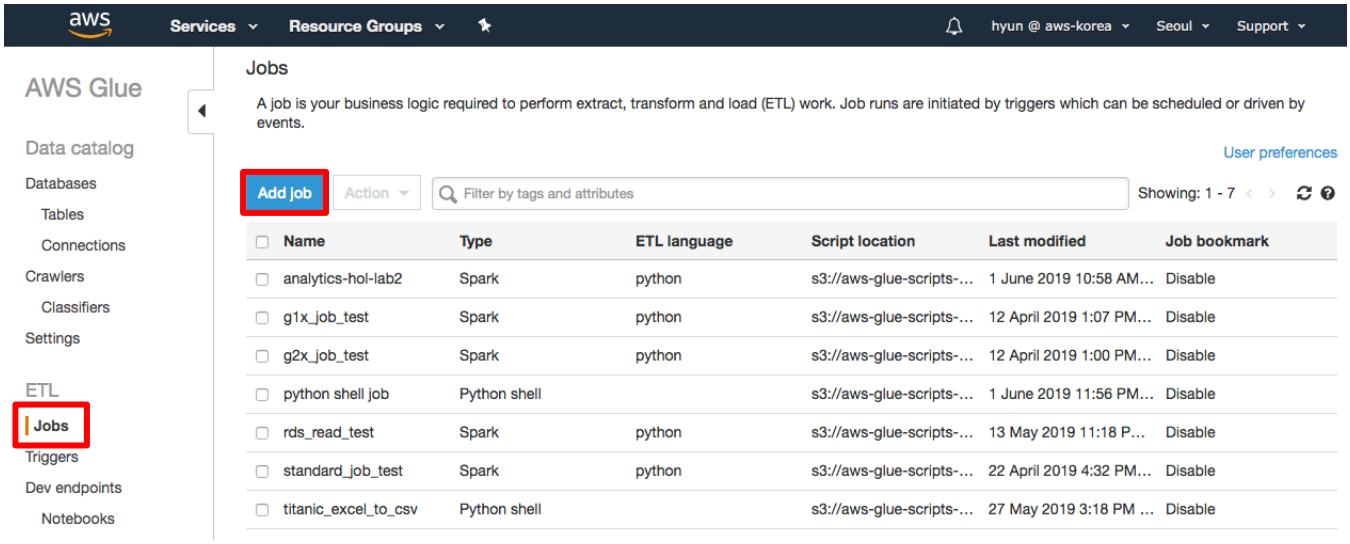


7-48. Python Shell Job 에서 RDS 에 접속하기 위해 http://bit.ly/hol-pymysql egg 파일을 다운로드 받습니다. (Python Shell 에서 사용하기 위한 egg 파일을 생성하는 방법은 Appendix A 를 참고하세요)

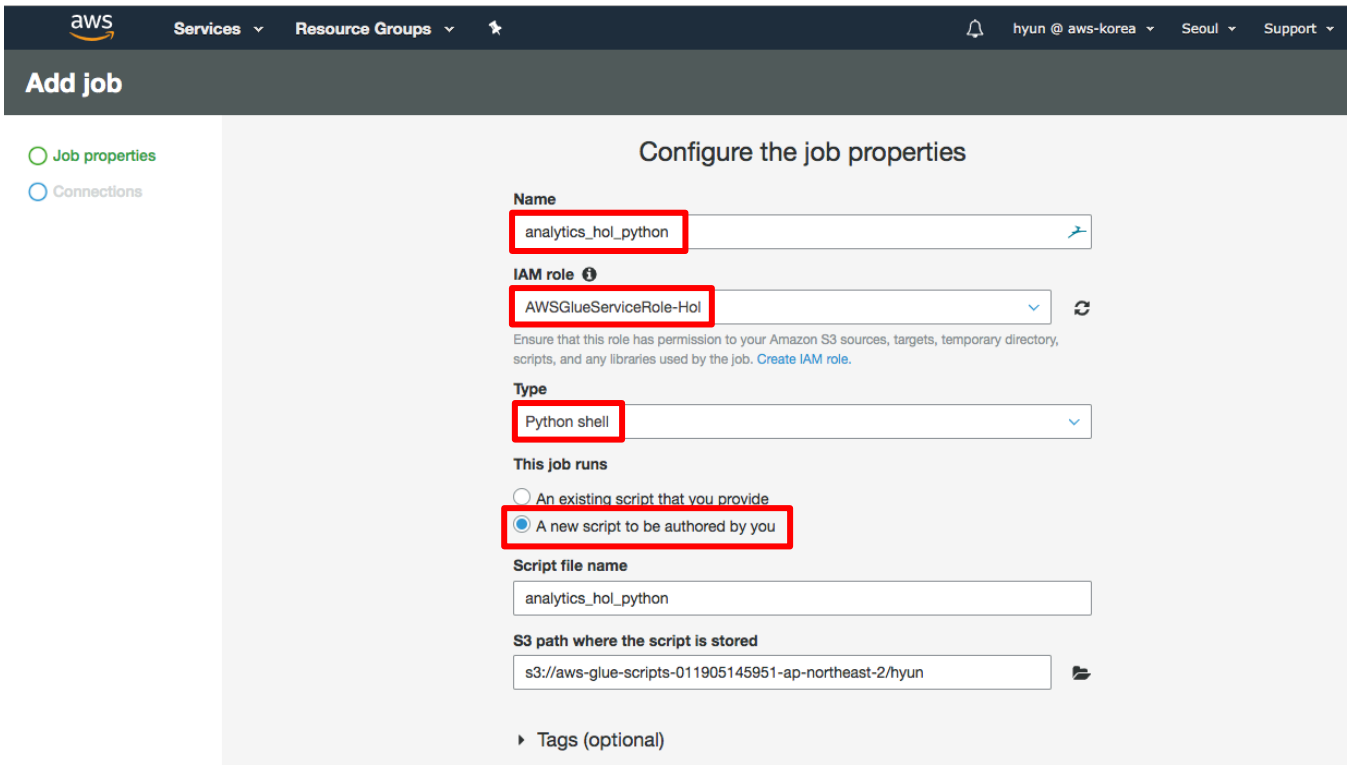
7-49. S3 콘솔로 이동 후 aws-glue-hol-[account-id] 버킷에 lib 디렉토리를 생성하고 위에서 다운로드 받은 파일을 업로드 합니다.

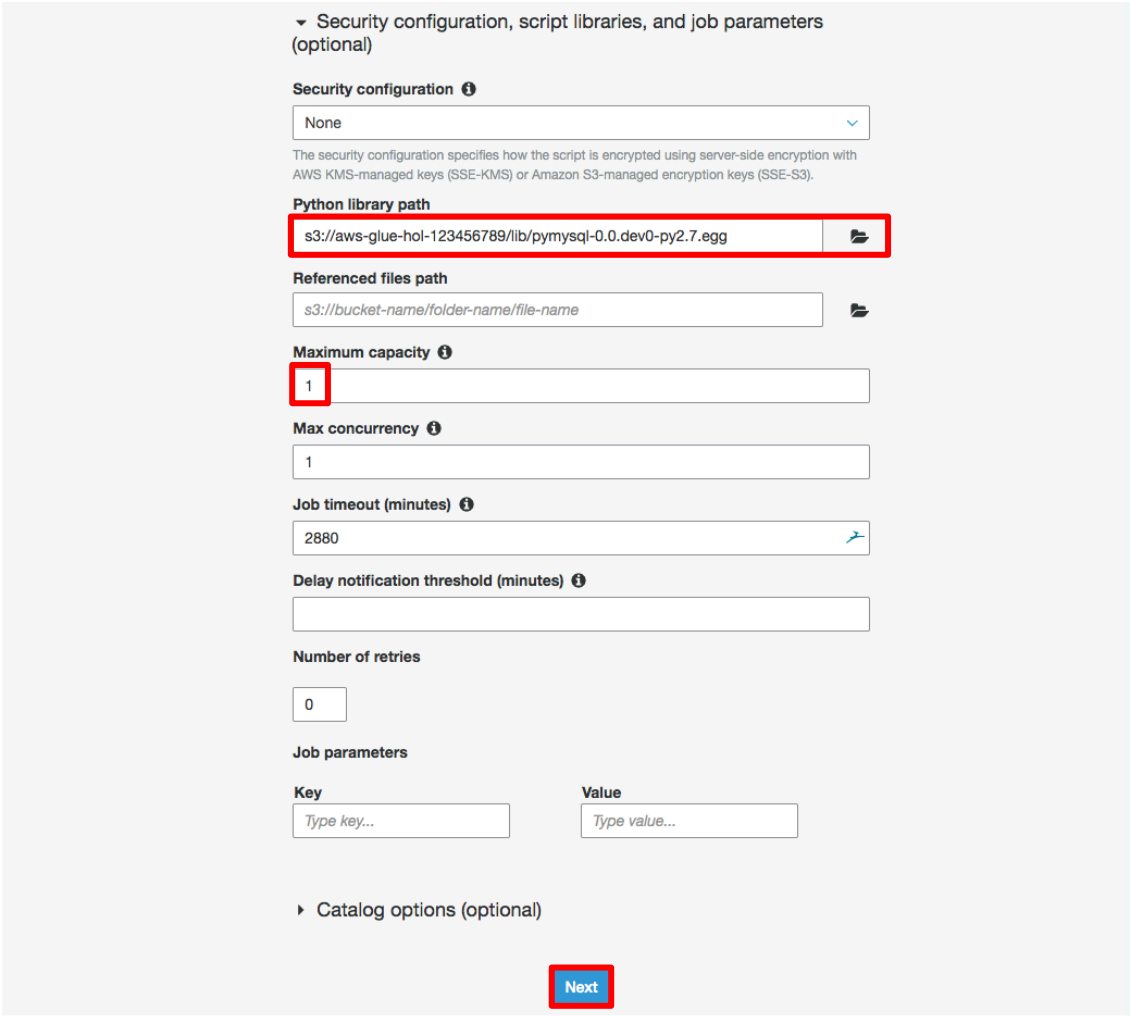


7-50. Python Shell Job 을 생성하기 위해 Glue 콘솔의 Jobs 로 이동한 후 Add job 버튼을 클릭합니다.

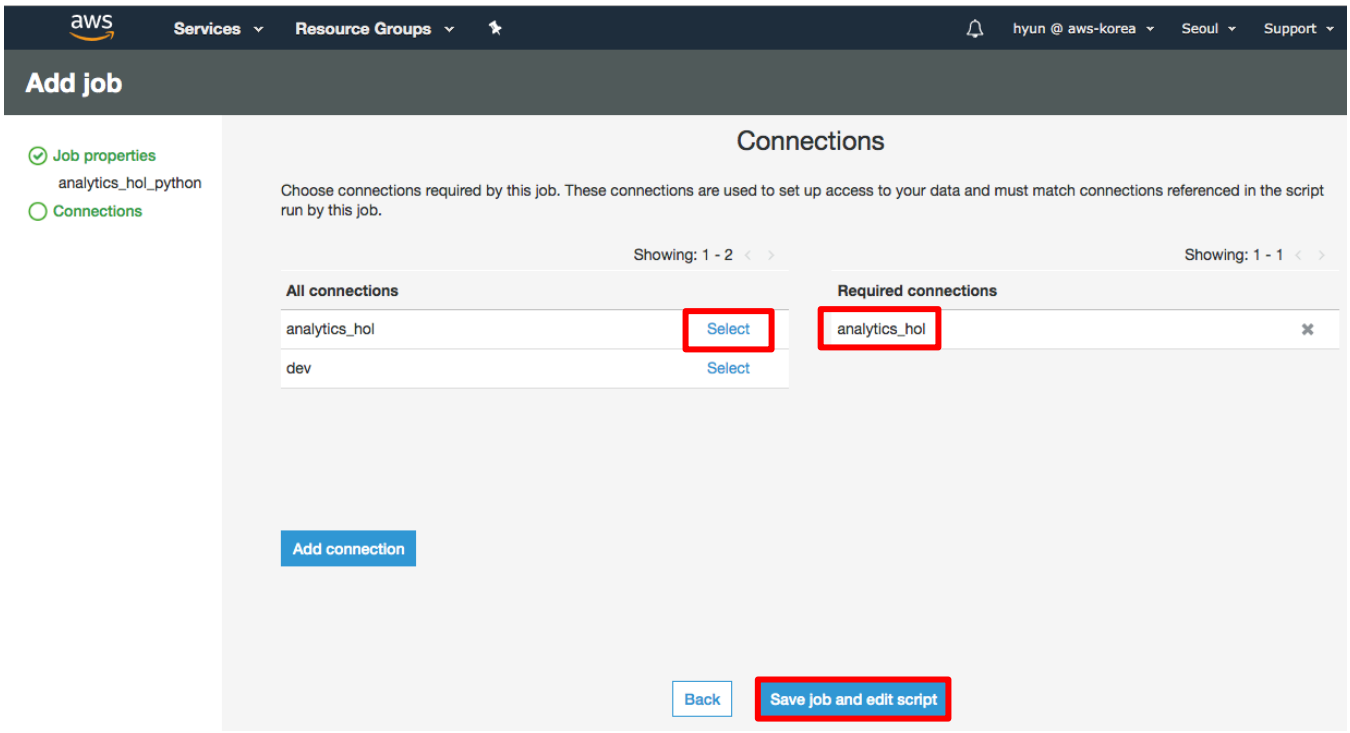


7-51. Configure the job properties 에서 Name 은 analytics_hol_python, IAM role 은 AWSGlueServiceRole-Hol, Type 은 Python shell, This job runs 는 A new script to be authored by you, Python library path 는 위에서 업로드한 egg 파일, Maximum capacity 를 1 을 입력하고 Next 버튼을 클릭합니다.

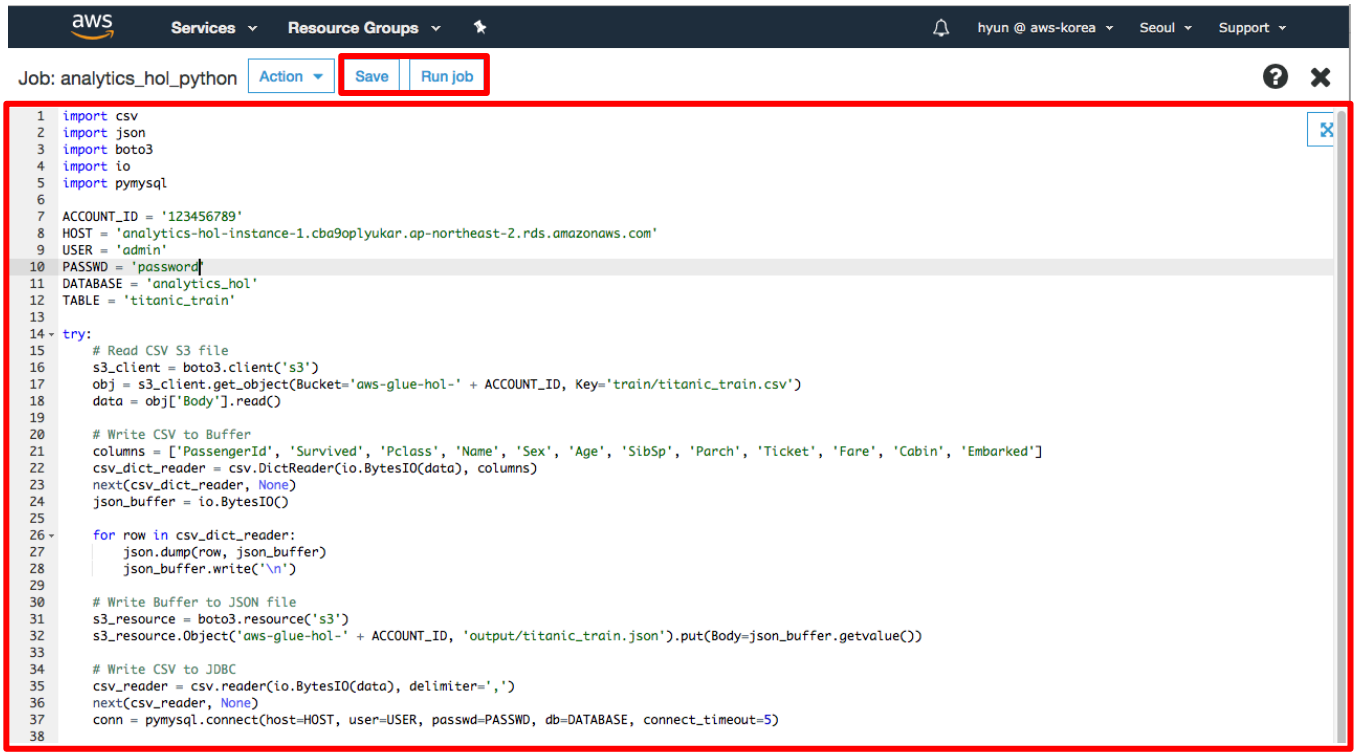



Security configuration, script libraries, and job parameters (optional)
Security configuration
None
The security configuration specifies how the script is encrypted using server-side encryption with AWS KMS-managed keys (SSE-KMS) or Amazon S3-managed encryption keys (SSE-S3).
Python library path
s3://aws-glue-hol-123456789/lib/pymysql-0.0.dev0-py2.7.egg
Referenced files path
s3://bucket-name/folder-name/file-name
Maximum capacity
1
Max concurrency
1
Job timeout (minutes)
2880
Delay notification threshold (minutes)
Number of retries
0
Job parameters
Key
Type key...
Value
Type value...
Catalog options (optional)
Next

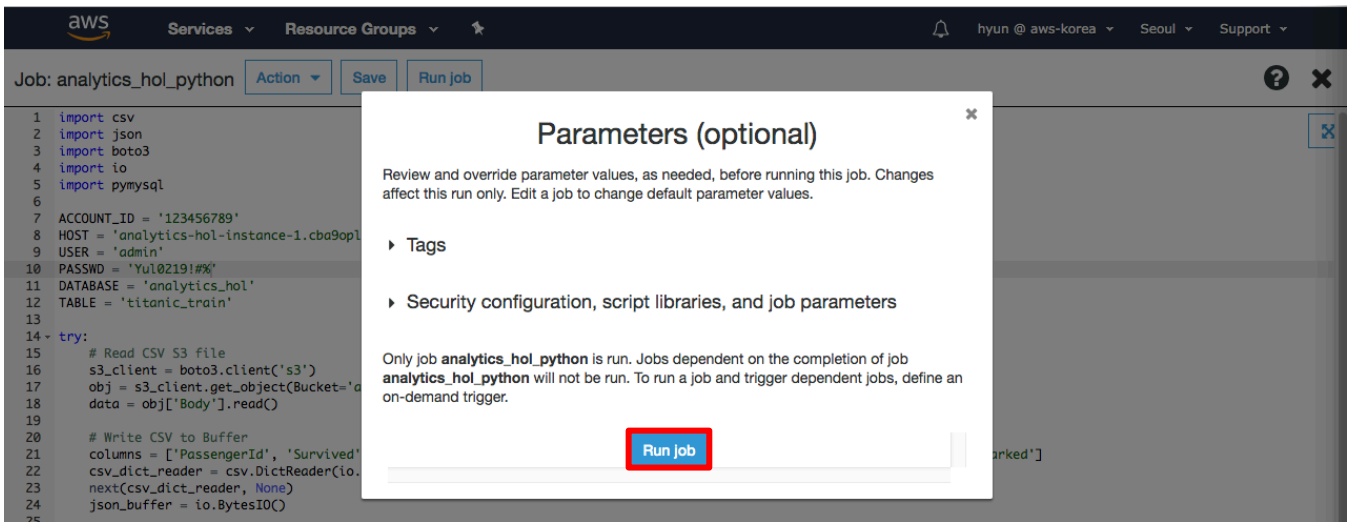
7-52. Glue Python Shell Job 에서 RDS 에 접속하기 위해 analytics_hol connection select 버튼을 클릭하고 Save job and edit script 버튼을 클릭합니다. analytics_hol 을 선택하면 Required connections 에 아래 그림과 같이 표시되는 것을 확인할 수 있습니다.



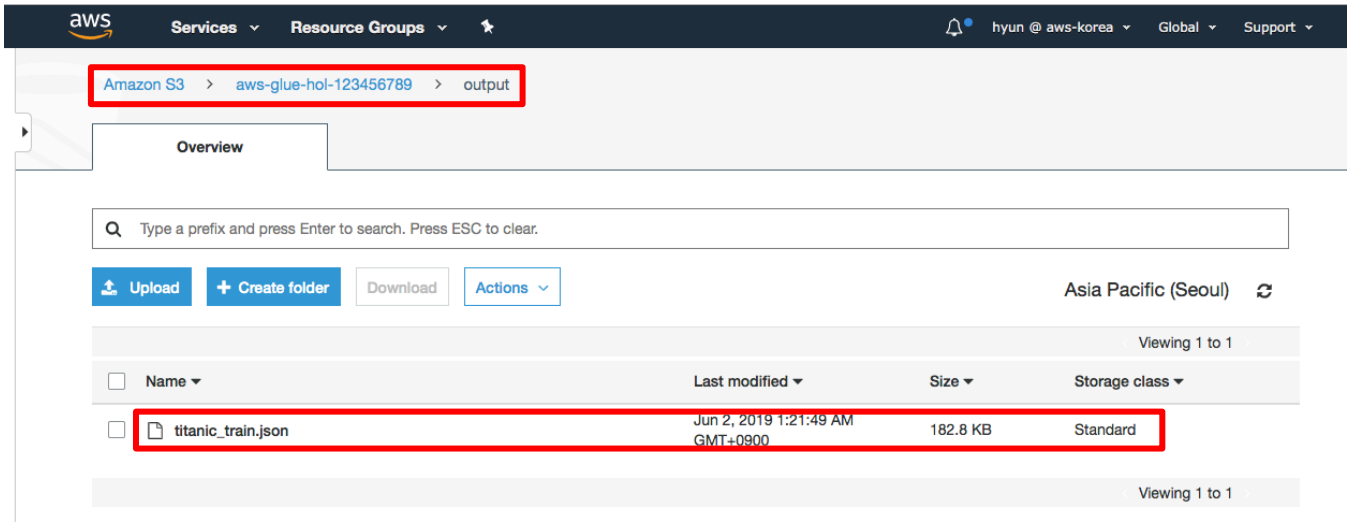
7-53. 05.python_shell.ipynb 노트북의 세 번째 코드 Cell 을 복사해서 Script Edit 창에 붙여넣고 ACCOUNT_ID, HOST, USER, PASSWD 를 입력한 후 Save 버튼과 Run job 버튼을 순서대로 클릭합니다.



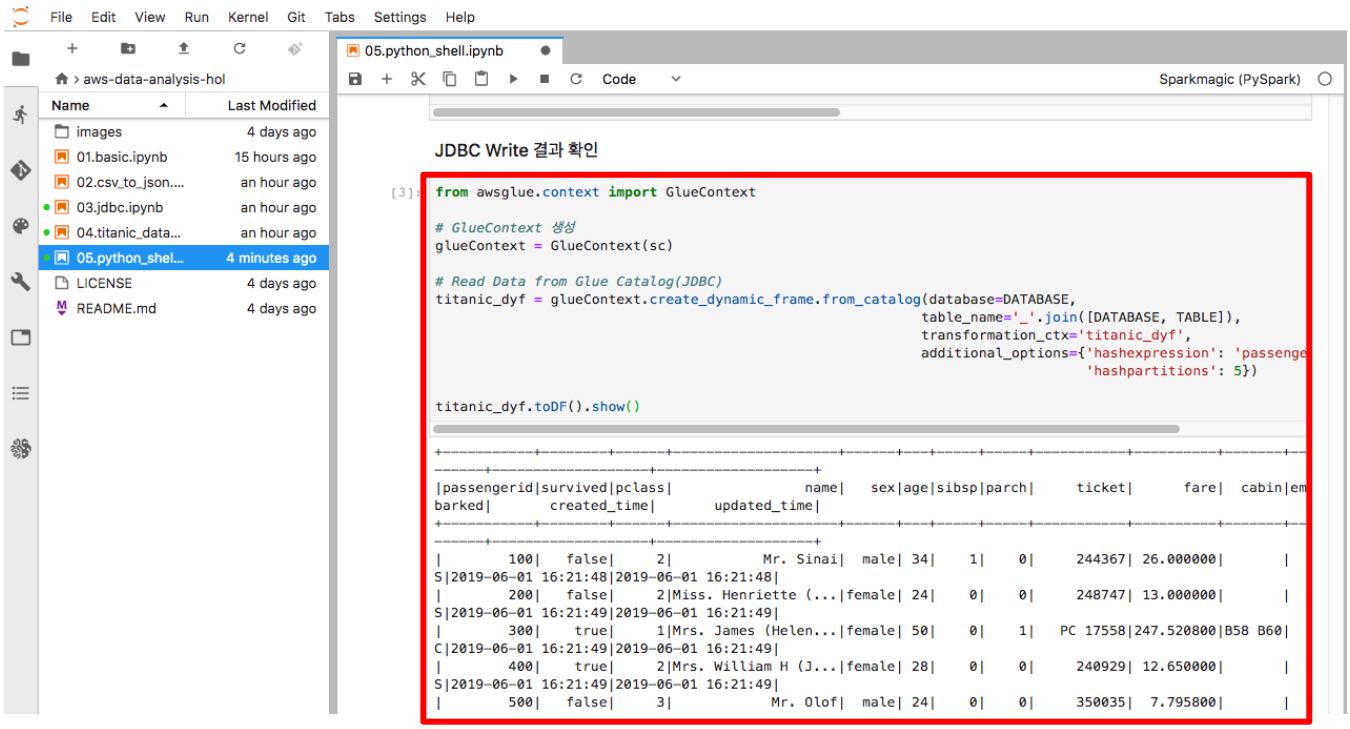
7-54. 다시 Run job 버튼을 클릭합니다.



7-55. S3 에 정상적으로 JSON 파일이 생성되었는지 확인합니다.



7-56. JDBC 에 정상적으로 데이터가 입력되었는지 확인하기 위해 05.python_shell.ipynb 노트북의 마지막 코드 Cell 을 실행하고 결과를 확인합니다.



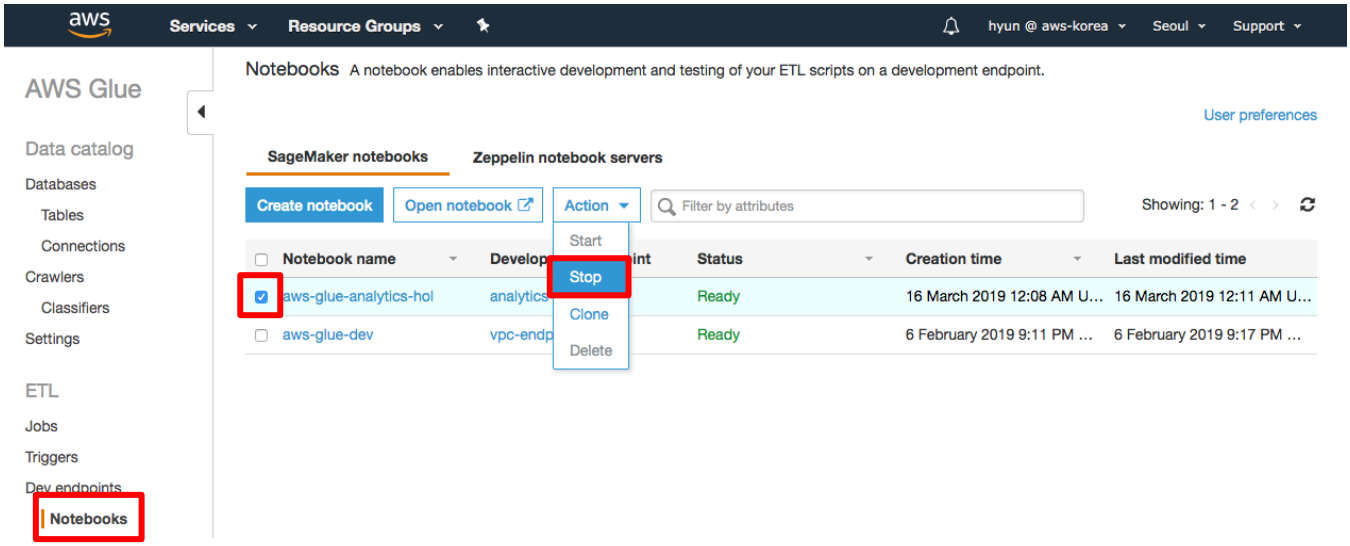
8. Closing

모든 실습이 완료되었습니다. 워크샵 이후 비용이 발생되는 것을 방지하기 위해 아래 단계에 따라 모든 리소스를 종료/삭제 해주세요.

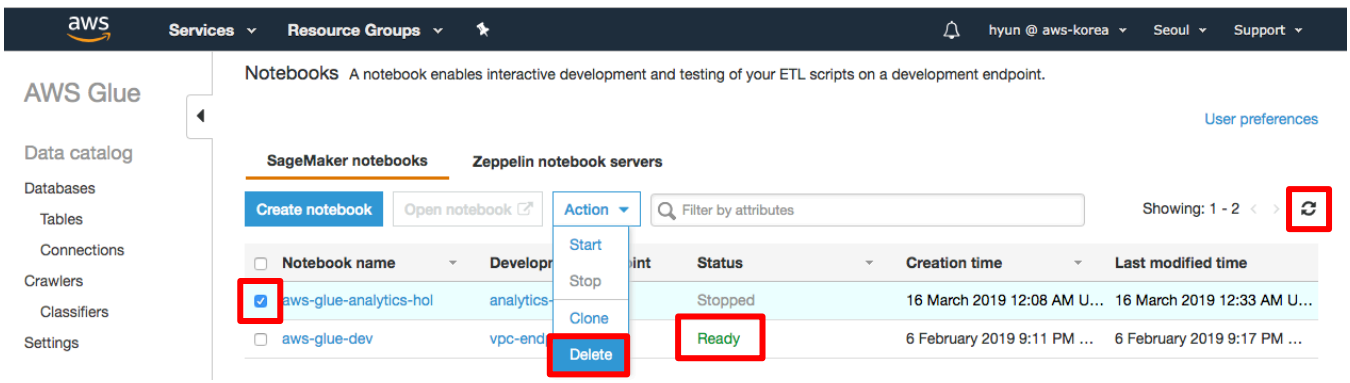
Glue 리소스 삭제

SageMaker notebook 을 먼저 삭제한 후 Dev endpoint 를 삭제합니다.

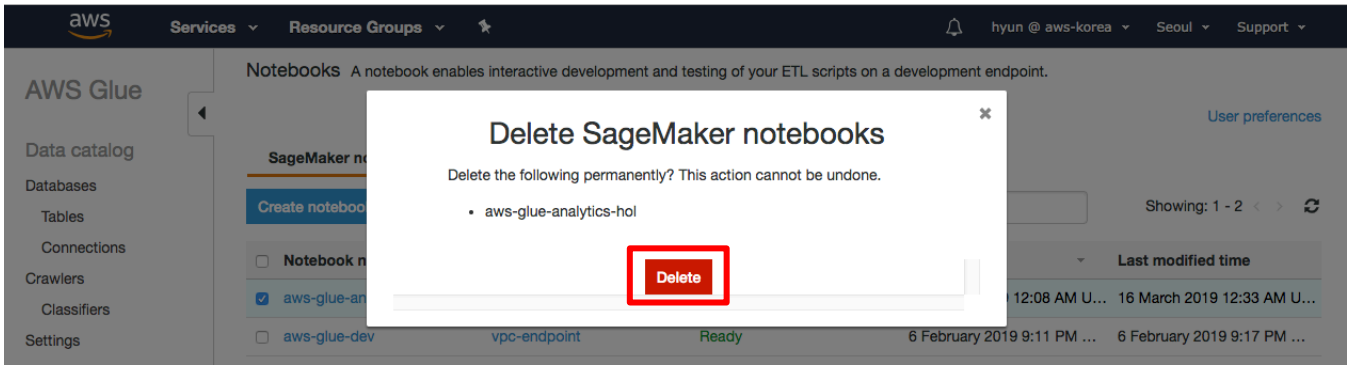
8-1. Glue 콘솔의 Dev endpoints->Notebooks 를 클릭하여 이동합니다. aws-glue-analytics-hol 노트북을 체크하고 Action 에서 Stop 을 선택합니다.

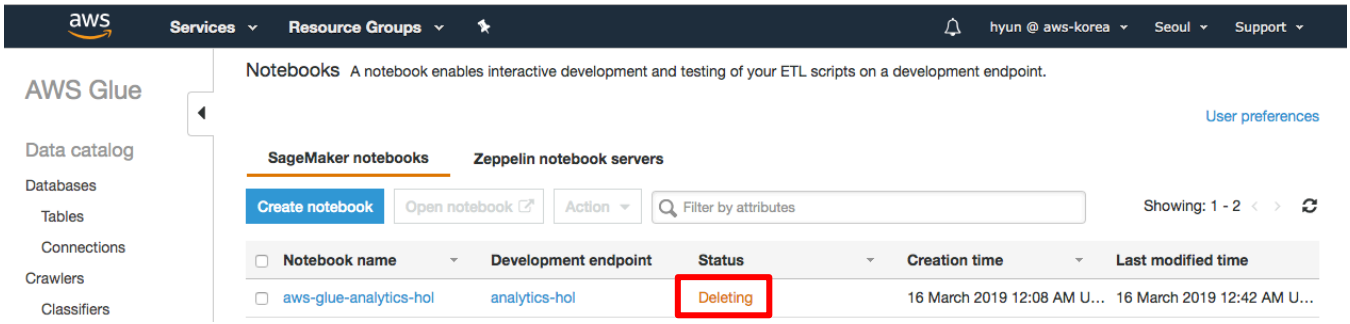


8-2. 리프레쉬 버튼을 눌러 Status 가 Stopping 에서 Stopped 으로 변경된 것을 확인한 후, Action 에서 Delete 를 선택합니다.

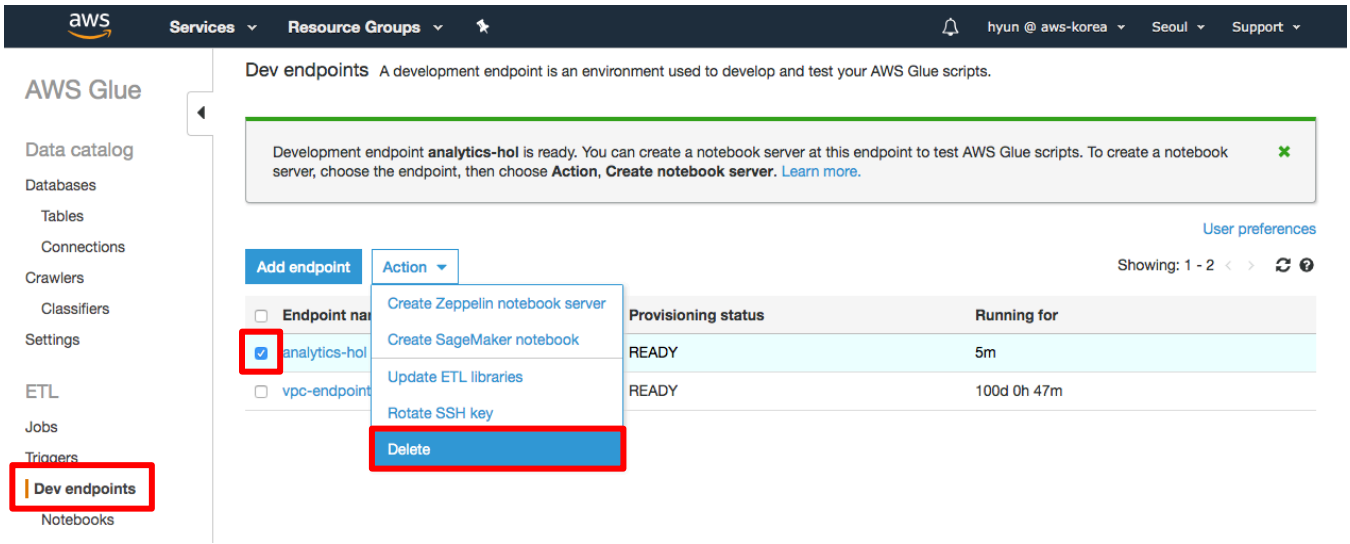


8-3. Delete 버튼을 클릭하면 aws-glue-analytics-hol 노트북 삭제가 완료됩니다.

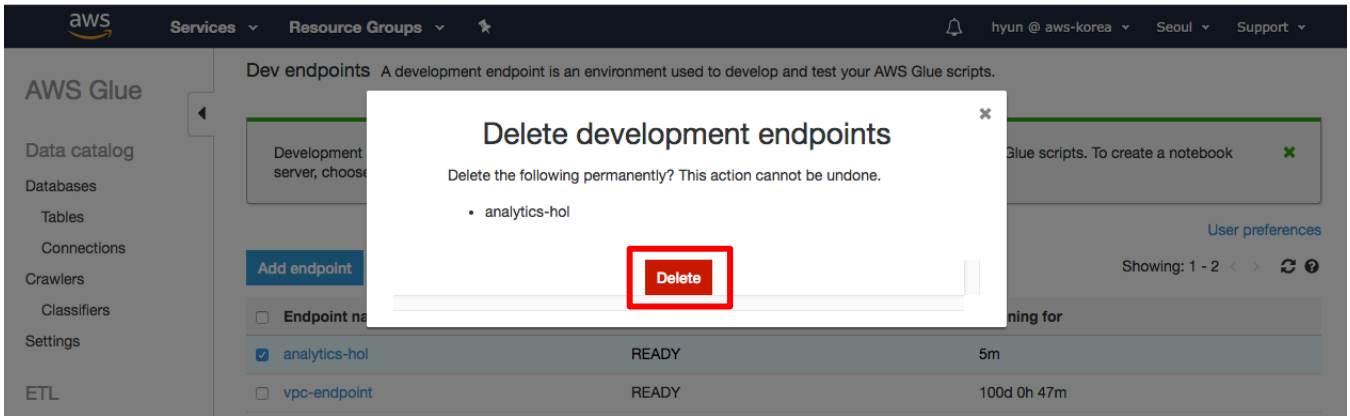




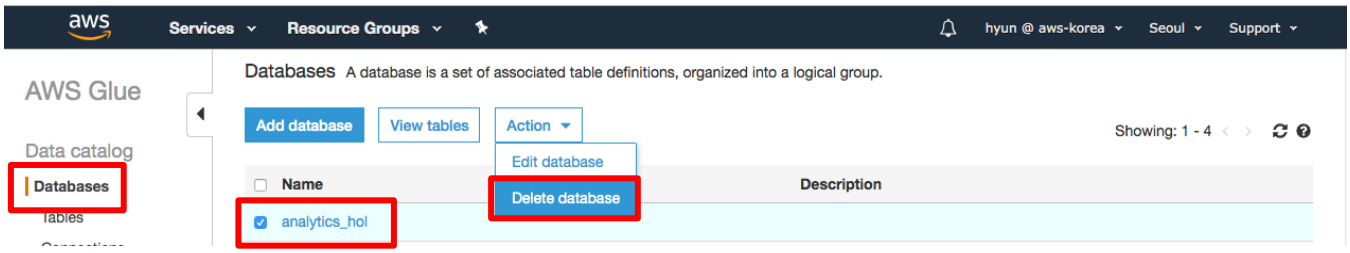
8-4. Glue 콘솔 Dev endpoints 를 클릭해서 이동한 후 analytics-hol 엔드포인트를 선택하고 Action 에서 Delete 를 선택합니다.



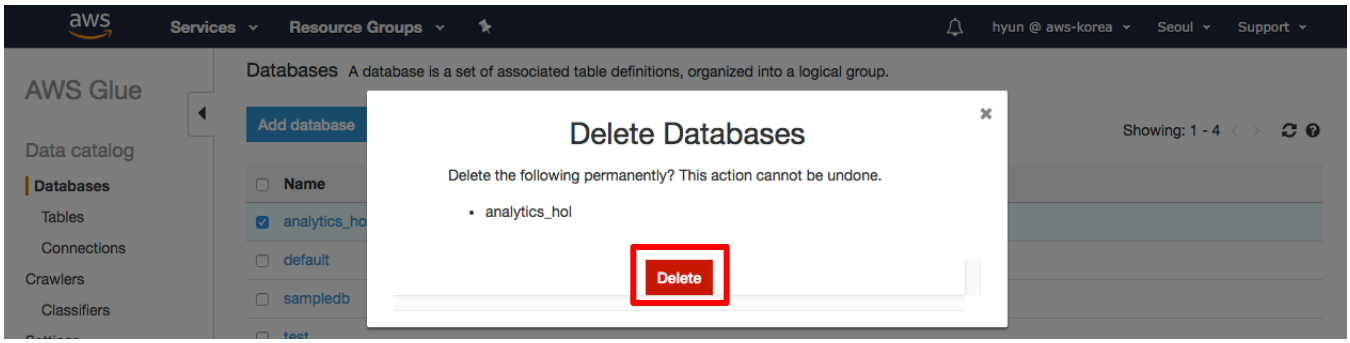
8-5. Delete 버튼을 클릭하면 analytics-hol 엔드포인트 삭제가 완료됩니다.



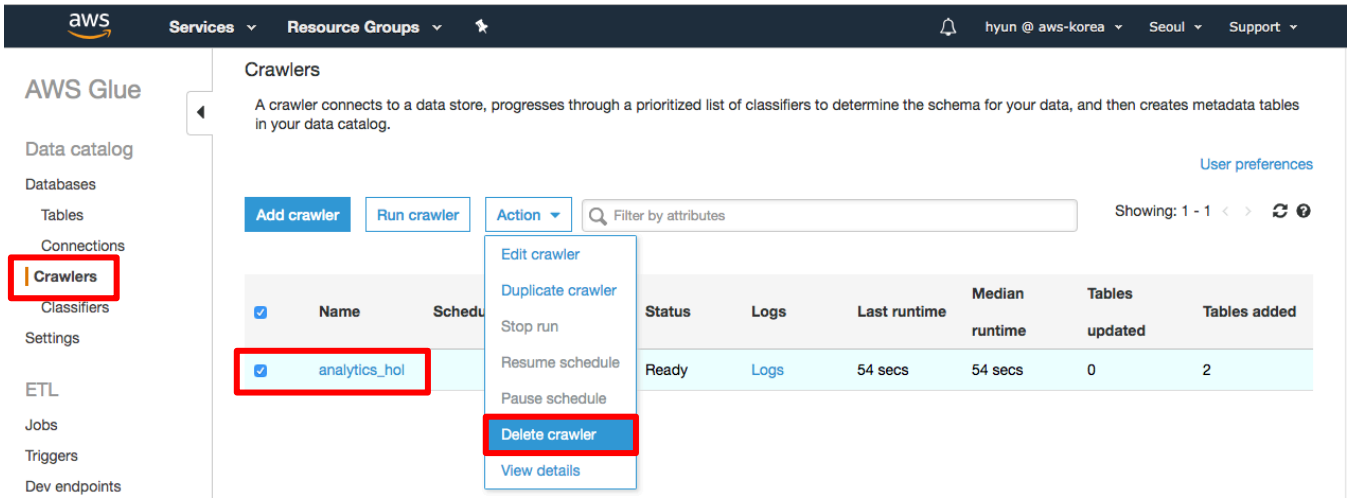
8-6. Glue 콘솔에서 Databases 를 클릭합니다. analytics_hol 데이터베이스를 선택하고 Action 에서 Delete database 를 선택합니다.



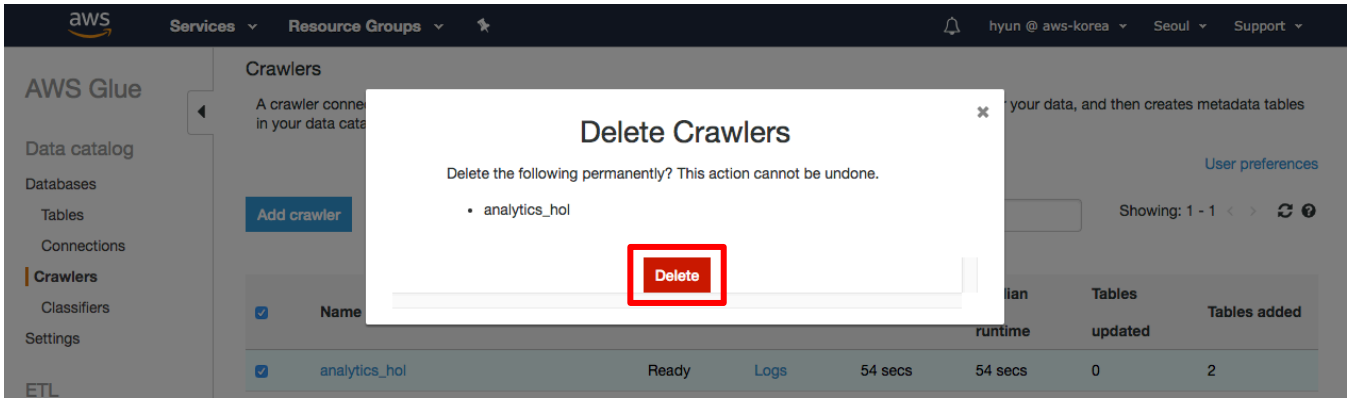
8-7. Delete 버튼을 눌러 삭제합니다.



8-8. Glue 콘솔에서 Crawler 를 클릭해서 이동한 후, analytics_hol 크롤러를 선택하고 Action 에서 Delete crawler 를 선택합니다.

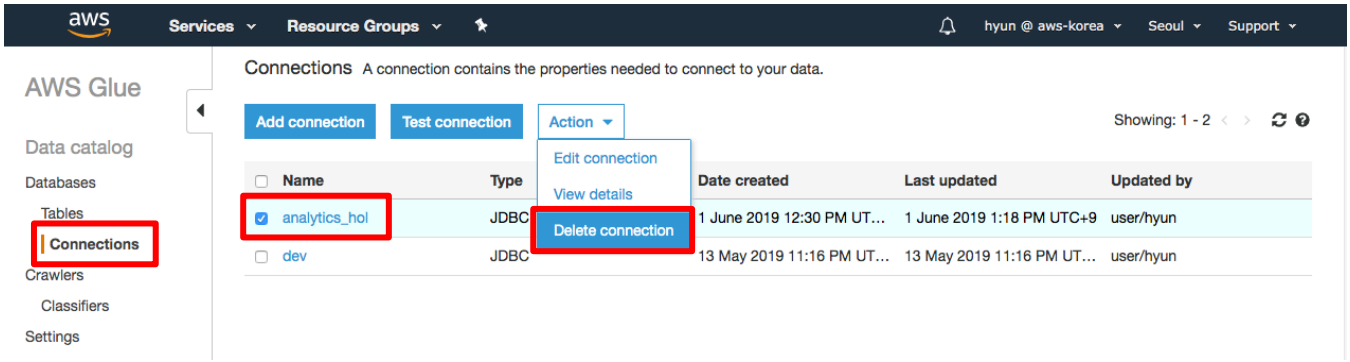


8-9. Delete 버튼을 클릭하여 삭제합니다.

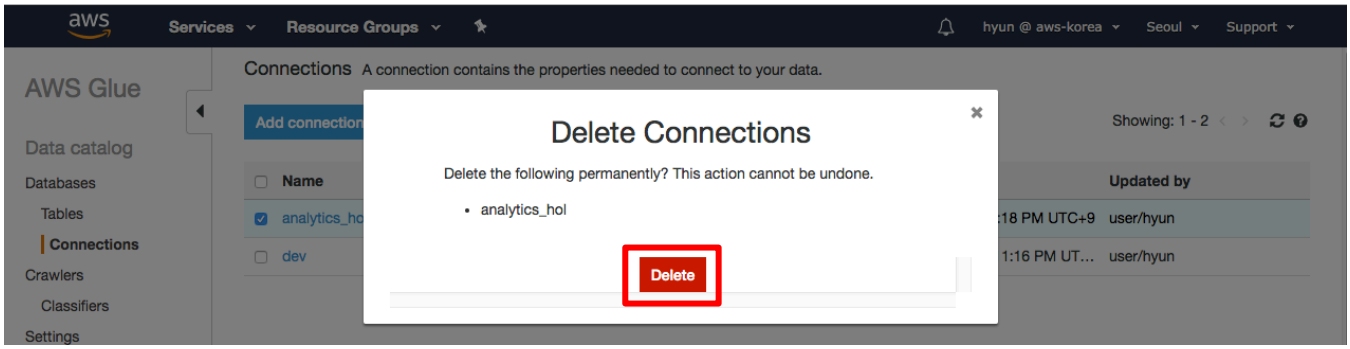


8-10. analytics_hol_jdbc 크롤러도 동일한 과정으로 삭제합니다.

8-11. Glue 콘솔에서 Connections 를 클릭해서 이동한 후, analytics_hol connection 을 선택하고 Action 에서 Delete connection 을 선택합니다.

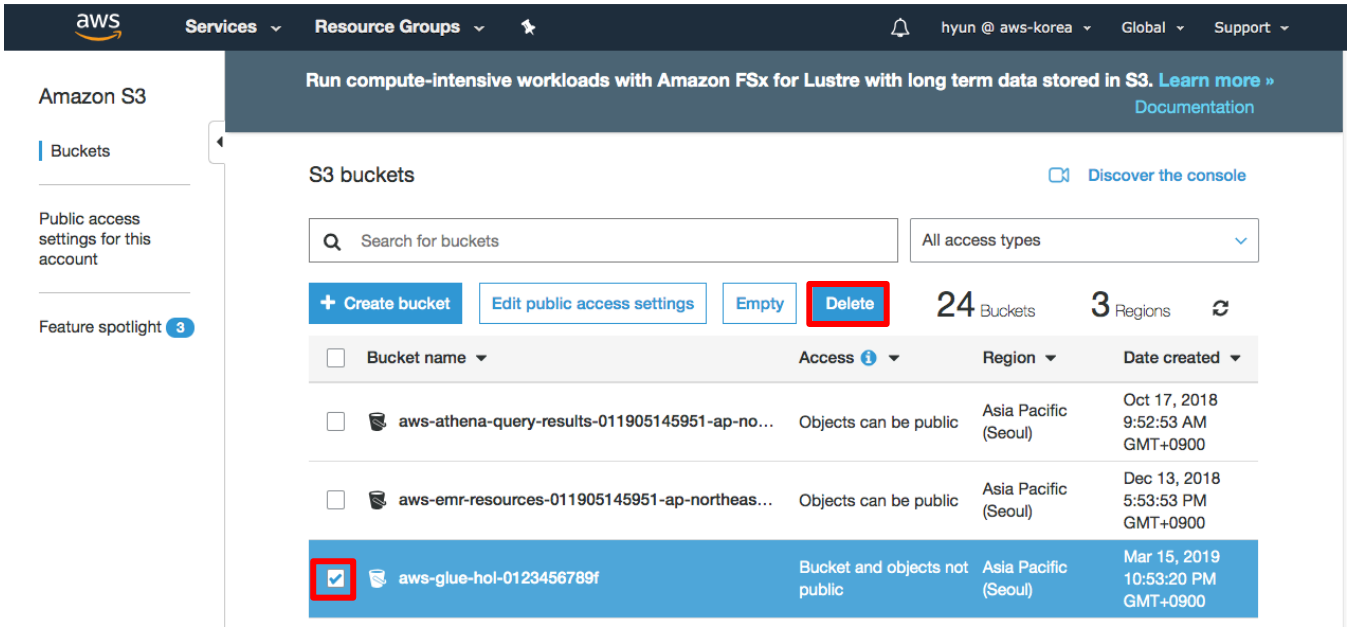


8-12. Delete 버튼을 클릭하여 삭제합니다.

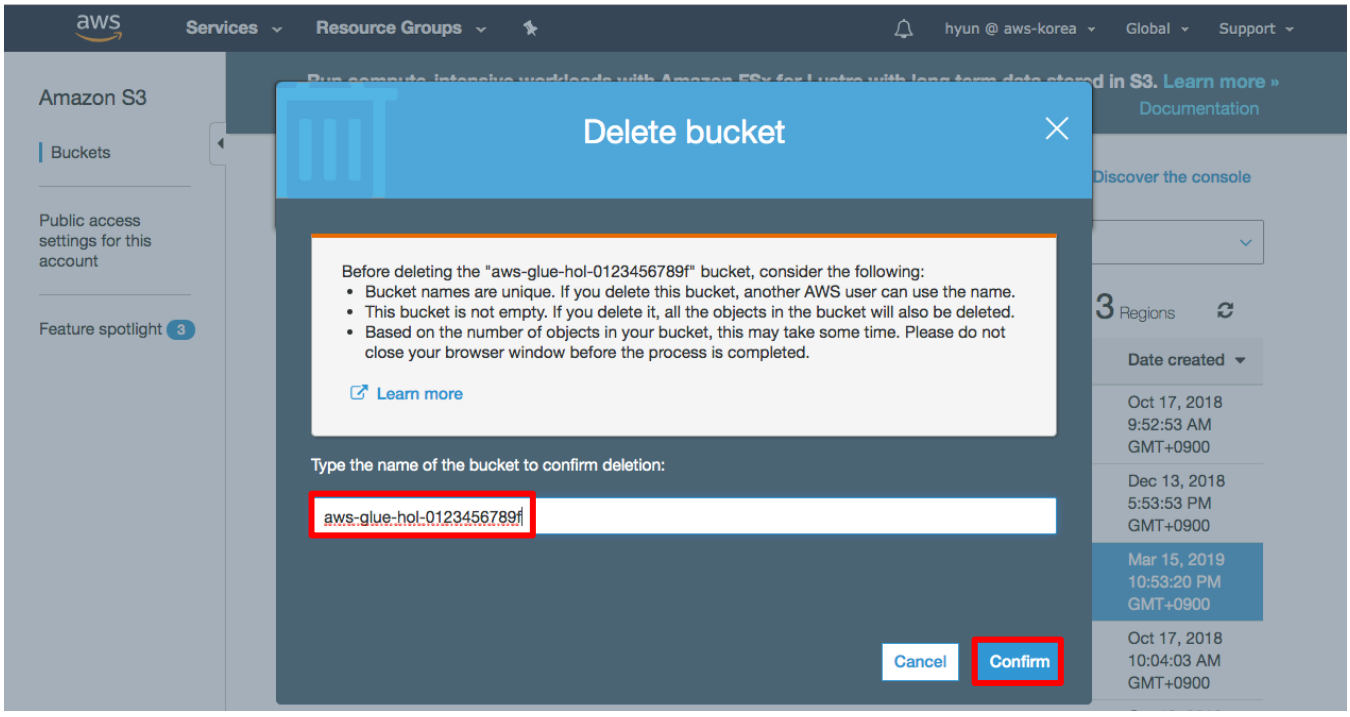


S3 버킷 삭제

8-13. S3 콘솔로 이동합니다. 워크샵을 위해 생성한 aws-glue-hol-[account-id] 버킷을 체크한 후 Delete 버튼을 클릭합니다.

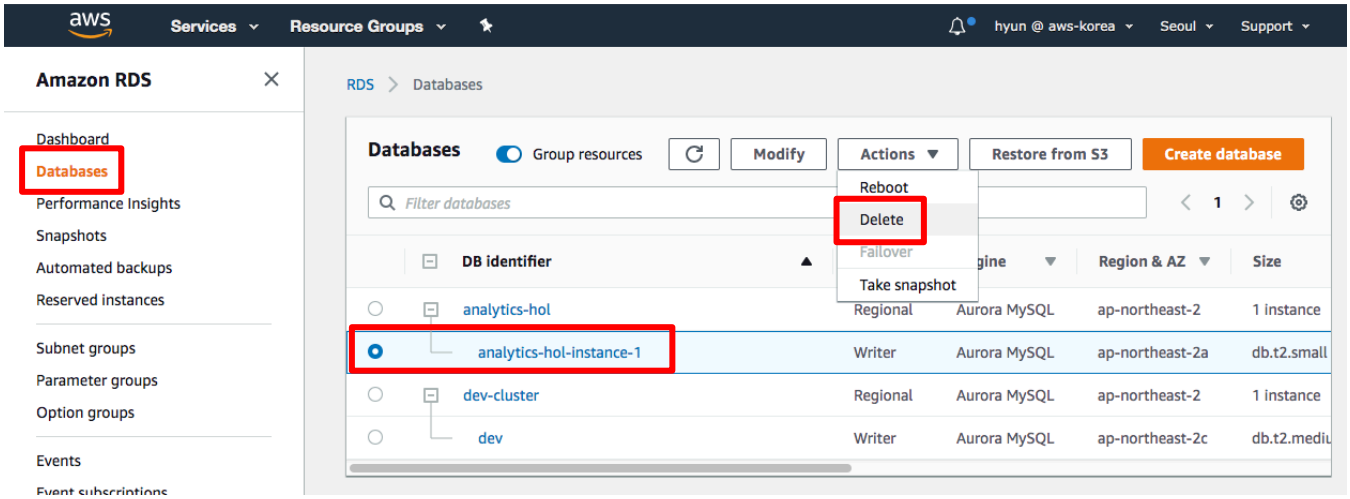


8-14. aws-glue-hol-[account-id]을 한번 더 입력한 후 Confirm 버튼을 클릭하면 버킷 삭제가 완료됩니다.

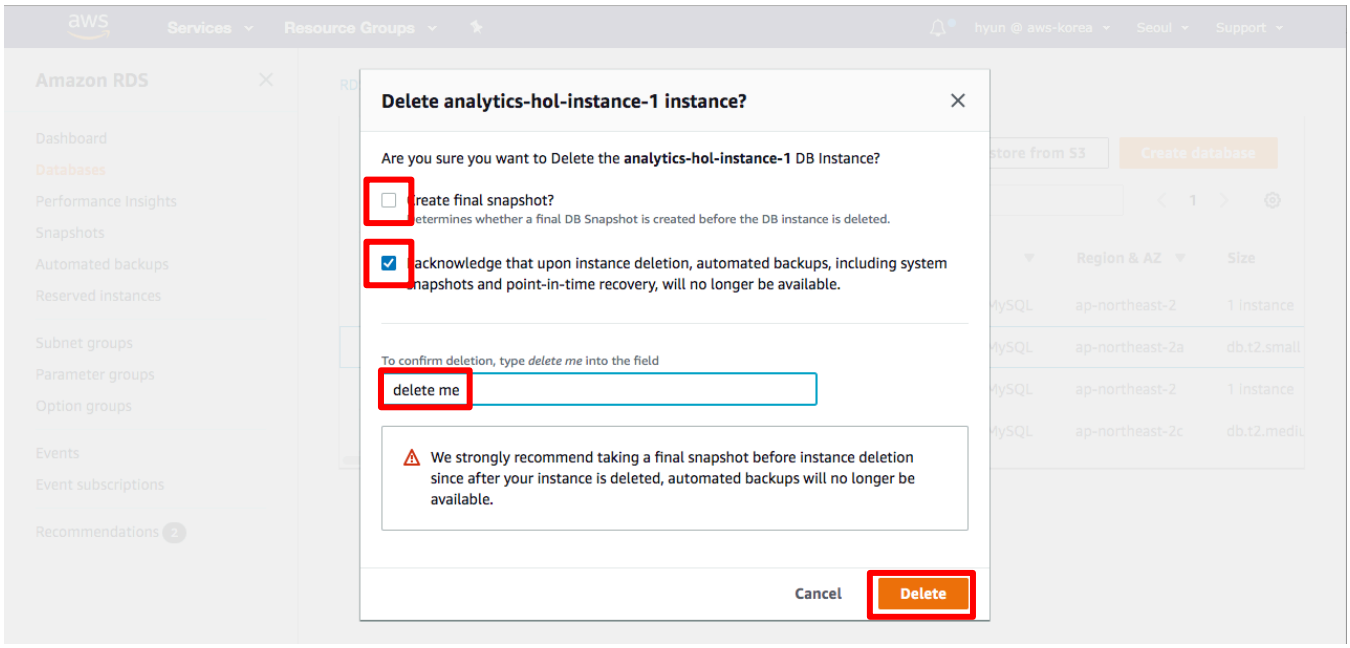


RDS 삭제

8-15. RDS 콘솔로 이동 후 Databases 를 선택합니다. analytics-hol-instance-1 indentifier 를 선택하고 Actions 의 Delete 를 선택합니다.



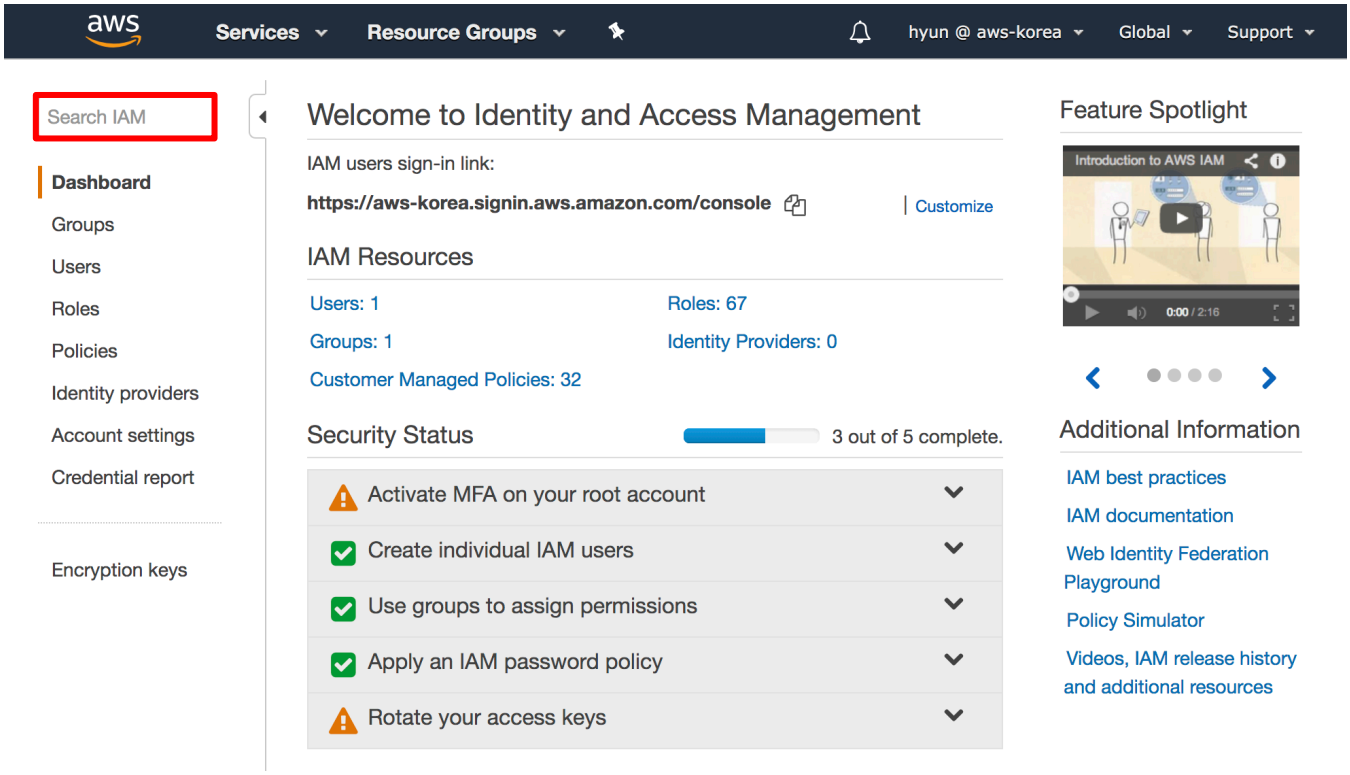
8-16. Create final snapshot 체크박스를 해제하고, I acknowledge 체크박스를 체크합니다. 삭제를 하기 위해 delete me 라고 confirm 메세지를 입력한 후 Delete 버튼을 클릭합니다.



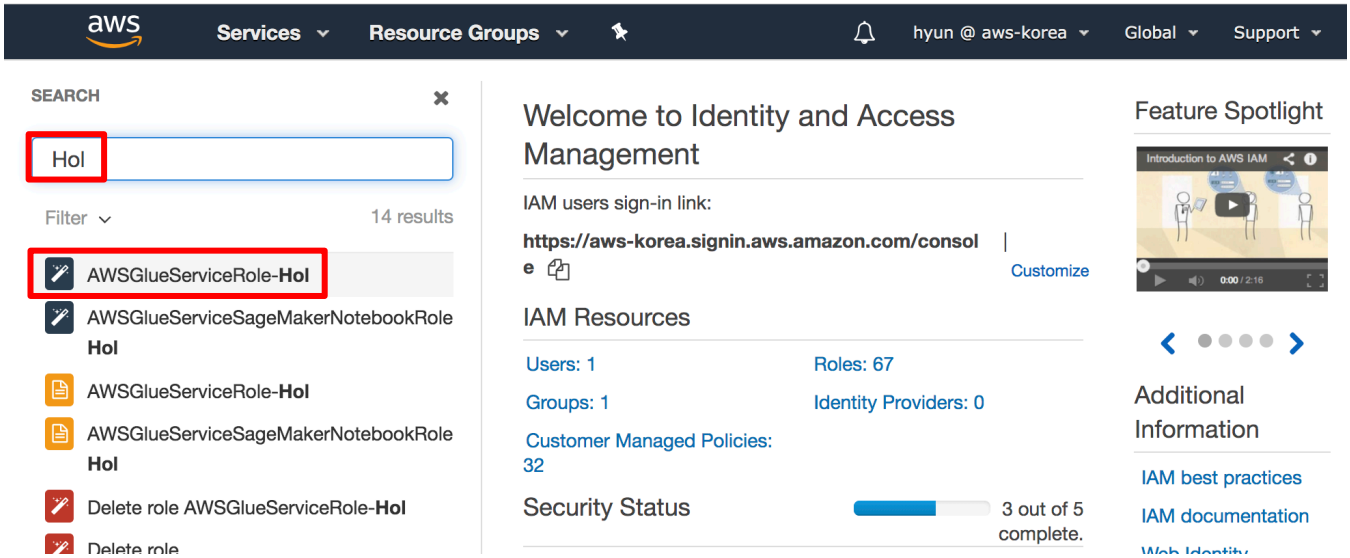
8-17. 삭제되는데까지 약 5 분정도 소요됩니다. Refresh 버튼을 눌러 삭제가 완료된 것을 확인합니다.

IAM Role 삭제 (Optional)

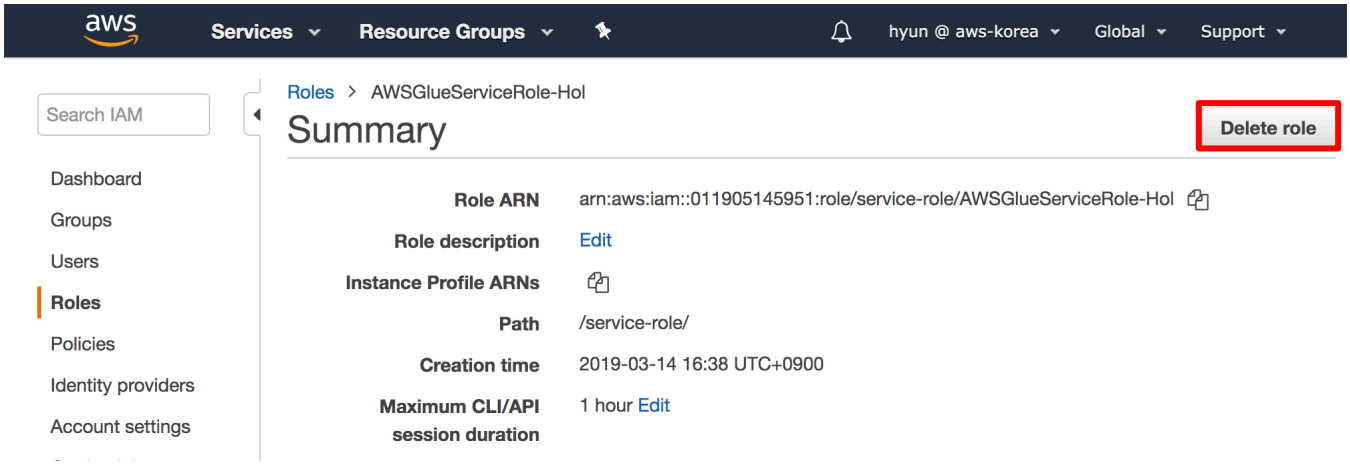
8-18. IAM 콘솔로 이동합니다. Search IAM 텍스트 박스에 Hol 을 입력합니다.

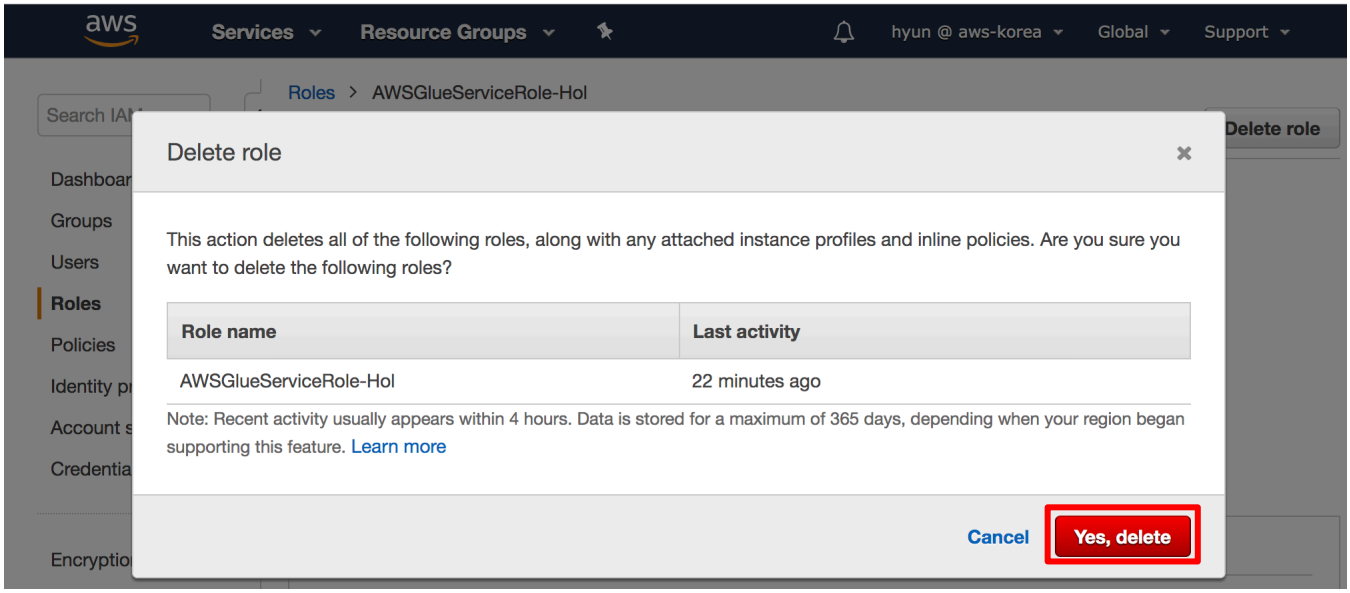


8-19. IAM 콘솔로 이동합니다. Search IAM 텍스트 박스에 Hol 을 입력하고 검색 결과 중 AWSGlueServiceRoleHol 을 선택합니다.

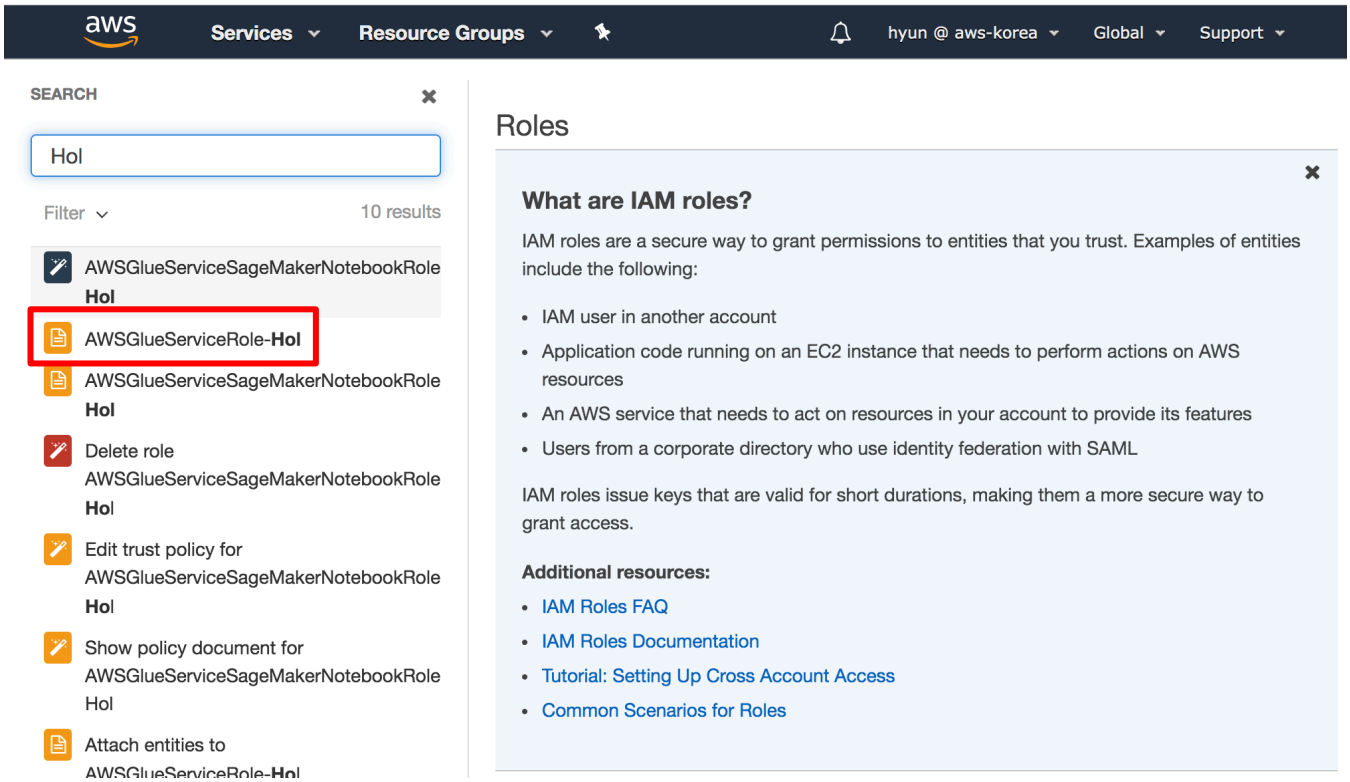


8-20. 오른쪽 상단에 있는 Delete role 버튼을 클릭하여 선택한 Role 을 삭제합니다.

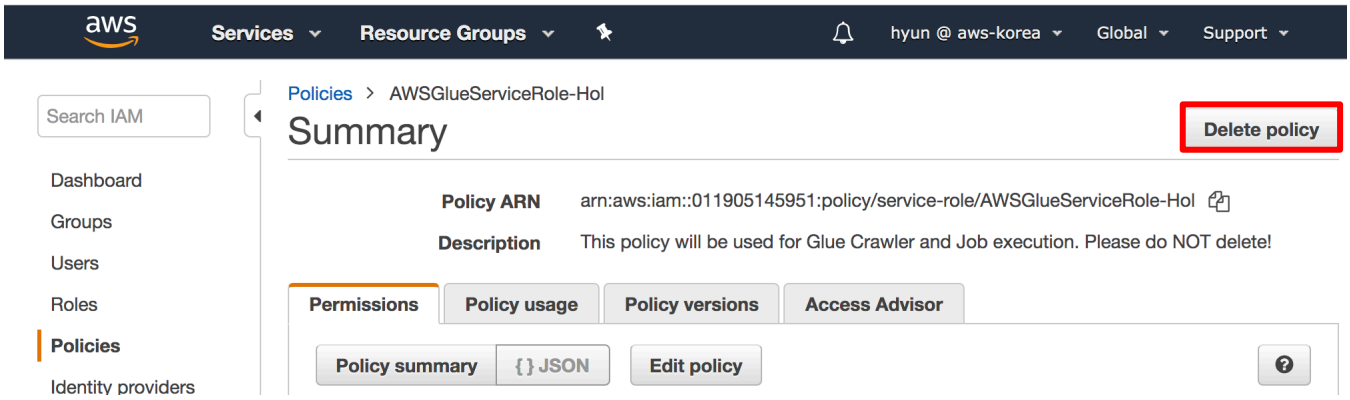


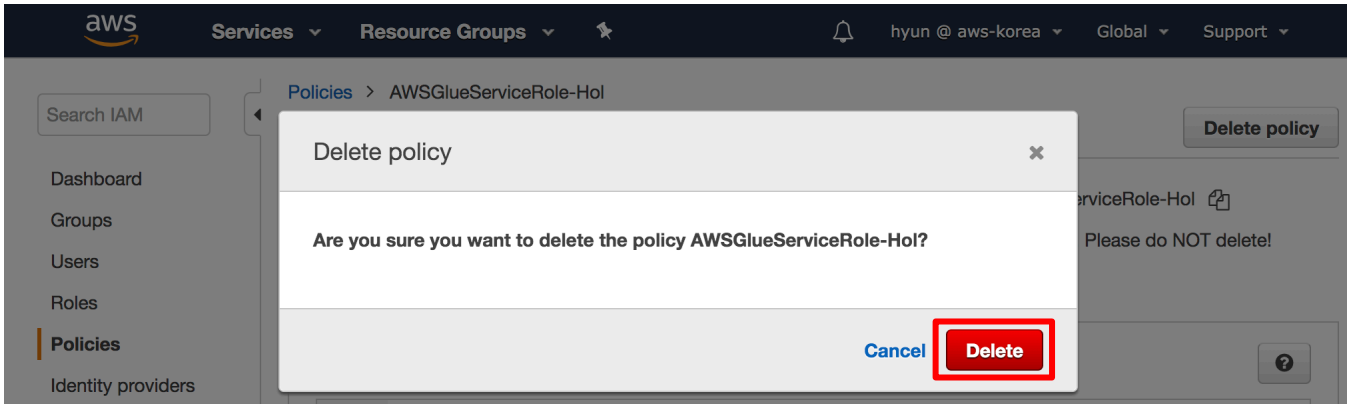


8-21. 다시 한 번 Hol 검색을 하고 AWSGlueServiceRole-Hol 을 선택합니다.



8-22. Delete policy 버튼을 클릭하여 선택한 Policy 를 삭제합니다.



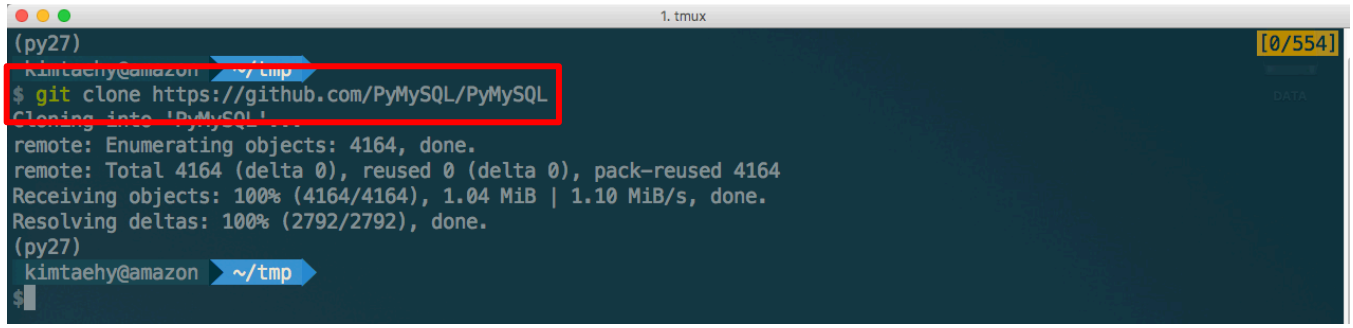


8-23. AWSGlueServiceSageMakerNotebookRole-Hol 에 대해서도 Role 과 Policy 를 같은 방법으로 삭제합니다.

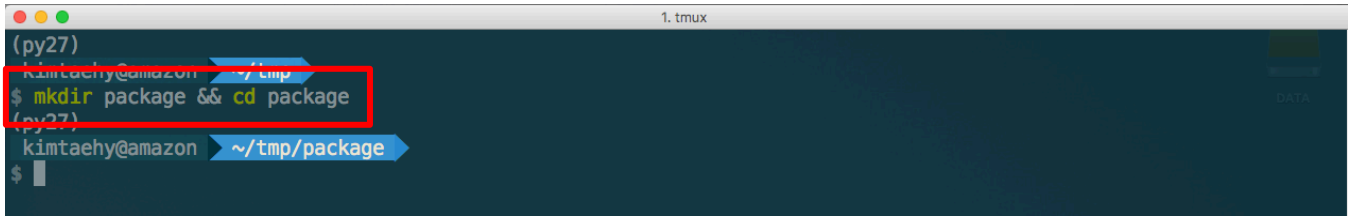
9. Appendix A

Glue Python Shell 은 Glue 에 설치되어있지 않은 외부 라이브러리를 사용할 수 있도록 지원하고 있습니다. Lab 실습 중 RDS 에 접근하기 위해 pymysql 을 egg 파일로 S3 에 업로드하고 Glue Script 에서 사용했습니다. 이번 장에서는 외부 라이브러리를 사용하기 위한 egg 파일을 생성하는 방법에 대해서 설명 드리도록 하겠습니다.

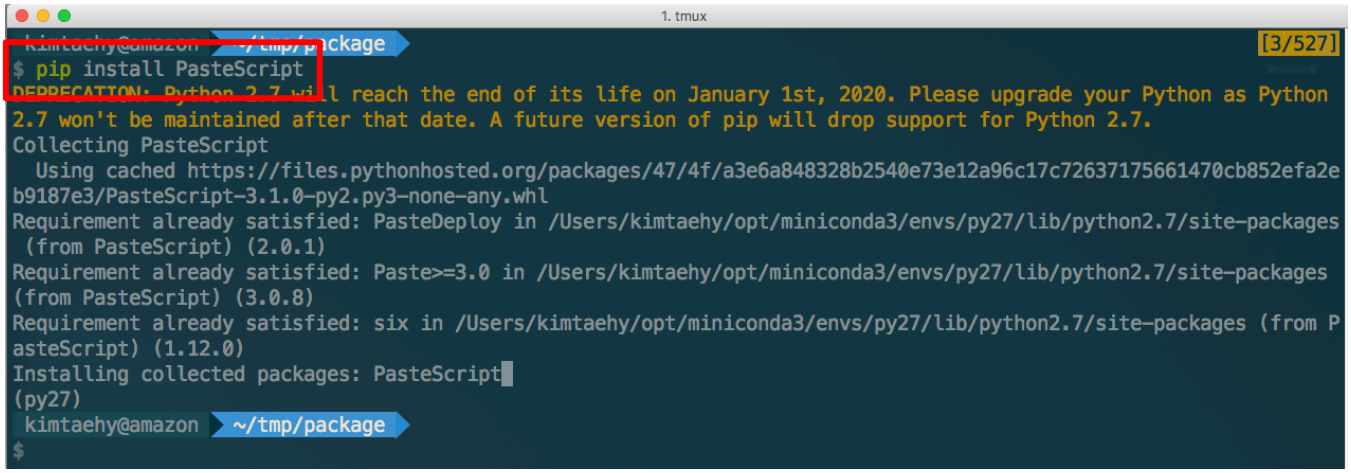
9-1. 사용할 외부 라이브러리를 git clone 받습니다. (https://github.com/PyMySQL/PyMySQL.git)



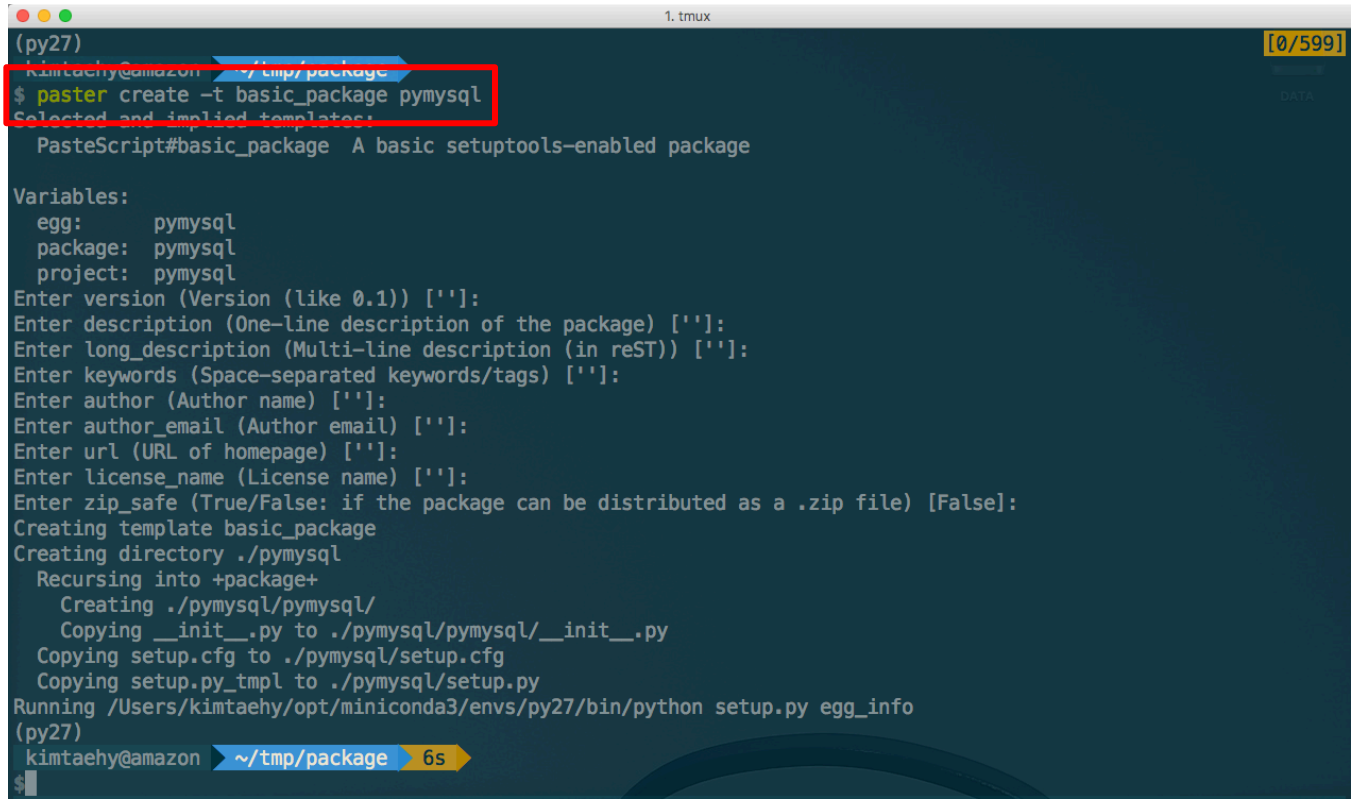
9-2. package 디렉토리를 생성하고 이동합니다.



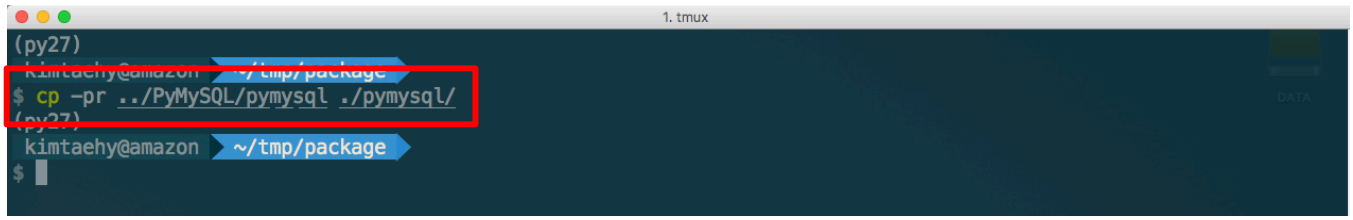
9-3. PasteScript 를 설치합니다.



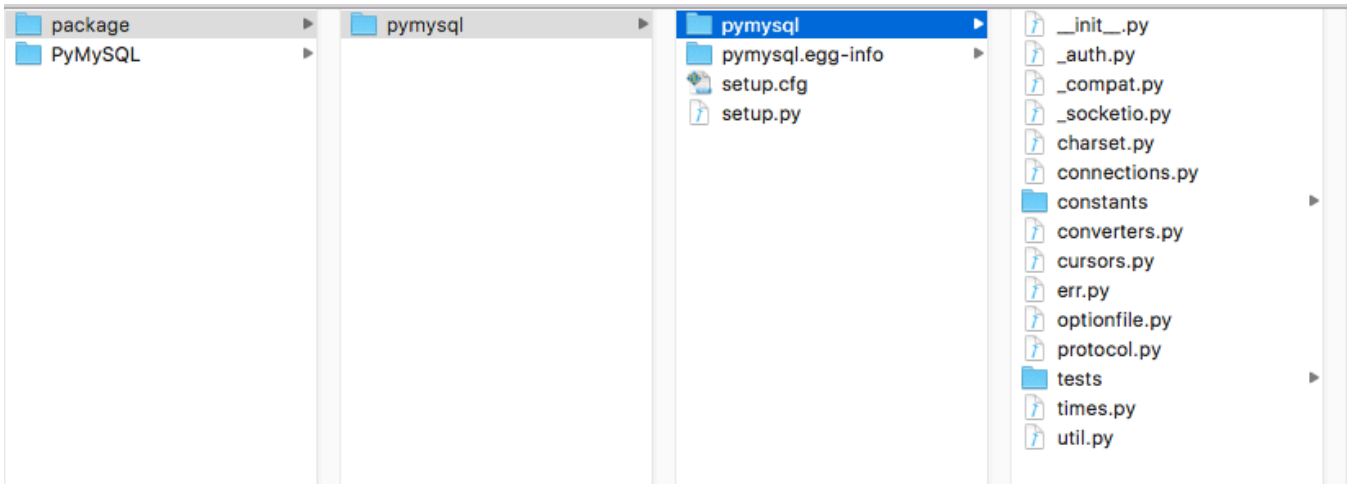
9-4. 디렉토리 레이아웃을 생성합니다.



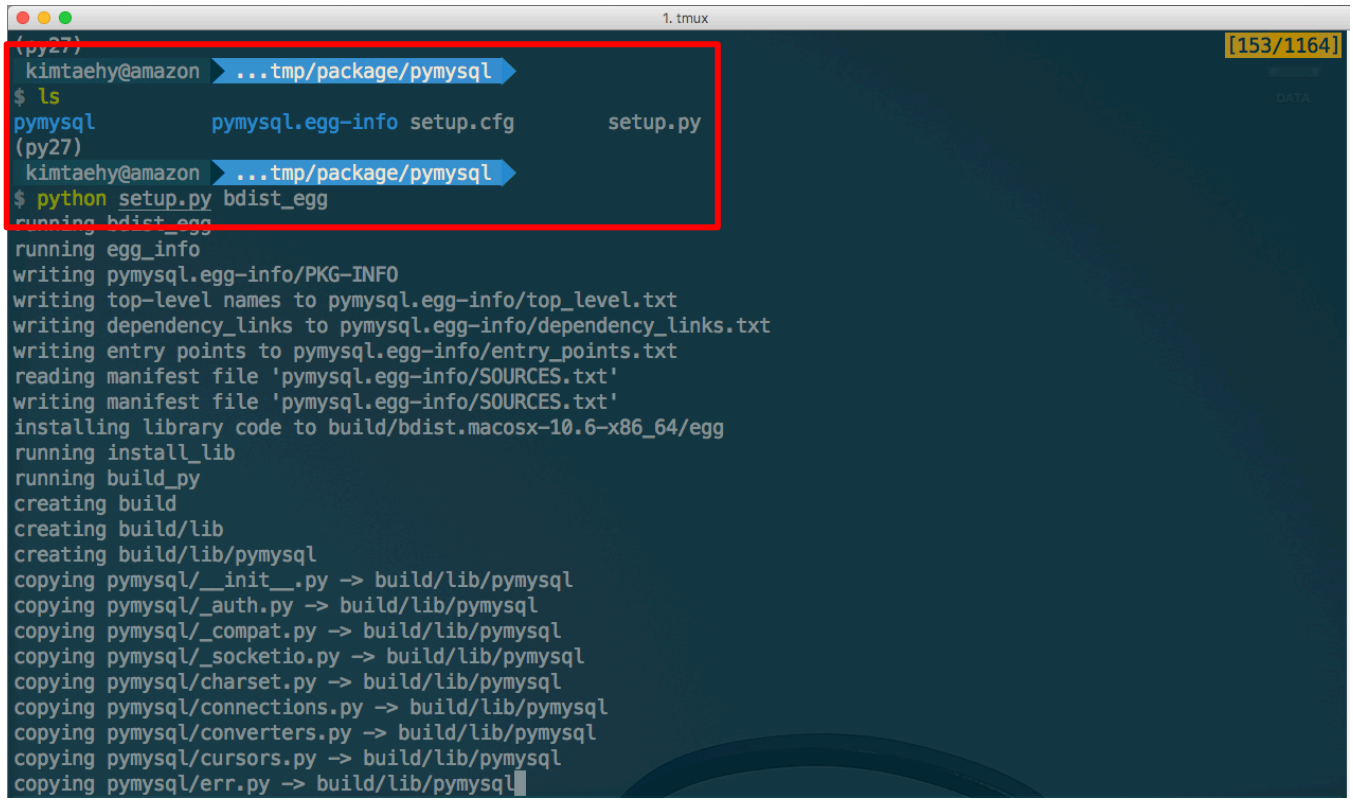
9-5. git clone 받은 라이브러리를 복사합니다.



9-6. 최종 생성된 디렉토리 구조는 다음과 같습니다.



9-7. setup.py 파일이 있는 pymysql 디렉토리로 이동하고 egg 파일을 생성합니다.



9-8. egg 파일이 생성되었는 지 확인되면 이제 Glue Python Shell 에서 사용할 수 있습니다.