

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



AWS Analytics Hands-on Guidebook

1st June 2019

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

© 2019 Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.

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

All trademarks are the property of their owners.

Table of Contents

Table of Contents.....	2
1. Overview.....	4
Lab 개요.....	4
목표.....	4
2. Prerequisite	5
Amazon Web Service account 생성.....	5
S3 버킷 생성	6
파일 업로드	11
3. Create Glue Catalog and Crawling.....	15
Data Catalog databases 생성	15
Crawler 및 Glue Catalog 생성.....	16
4. Exploring Data Analysis with Athena	25
GLUE 카탈로그로 업그레이드	25
S3 data query with Athena.....	26
5. Create RDS and S3 Endpoint.....	35
RDS 생성.....	35
S3 Endpoint 생성	39
6. Glue Dev Endpoint with SageMaker notebook.....	43
Dev Endpoint 생성	43
SageMaker notebook 생성	47
7. Lab: Data Analysis, ETL and ML	51
Git Clone	51
Lab1: Glue Job Basic.....	53
Lab2: Create Job, Run and Debugging.....	54
Lab3: Read and Write Database	67
Lab4: Predict survival on the Titanic with AWS Glue.....	77
Lab5: Python Shell Job	78
8. Closing.....	84
Glue 리소스 삭제	84
S3 버킷 삭제.....	89

RDS 삭제.....	90
IAM Role 삭제 (Optional).....	91
9. Appendix A.....	95

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.

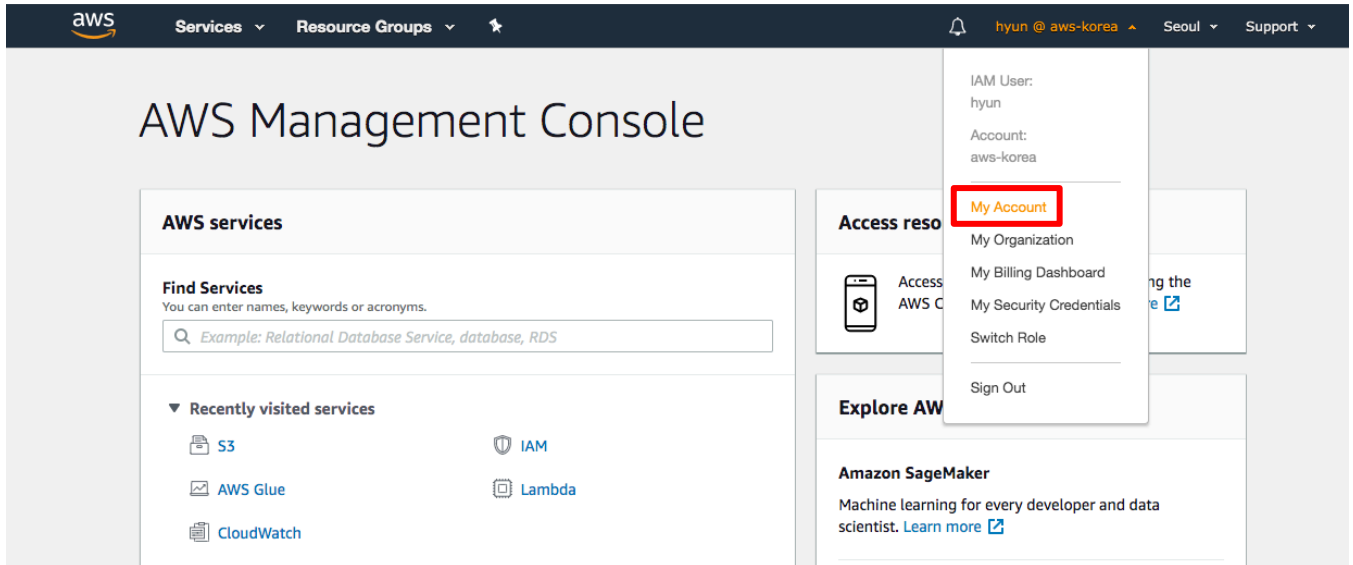
Amazon Web Service account 생성

AWS 계정이 없는 경우 다음 link 를 참고해서 계정을 생성하세요.

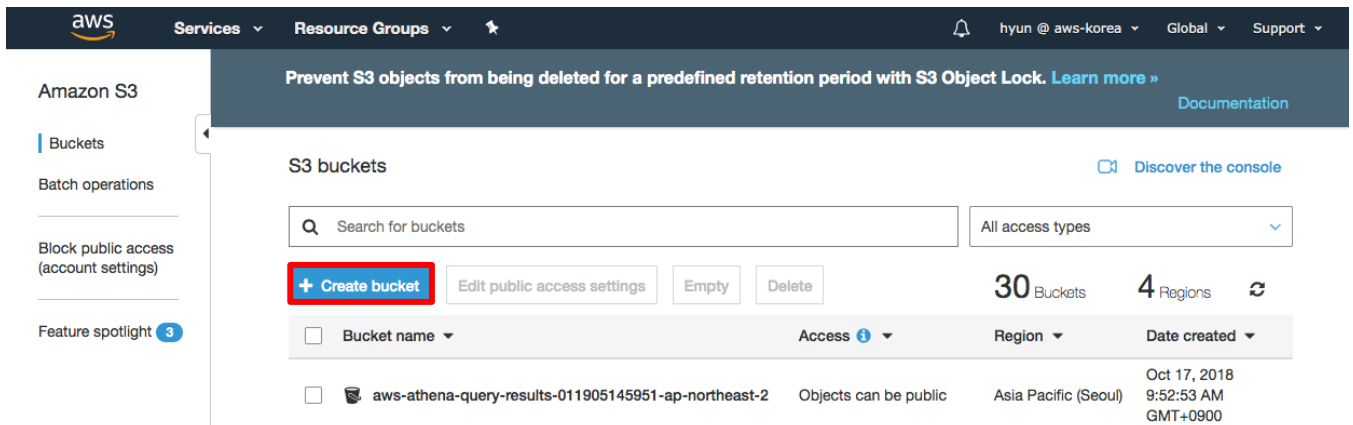
<http://bit.ly/create-aws-account-id>

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.

The screenshot shows the AWS console interface for creating a new S3 bucket. The 'Create bucket' wizard is open, and the first step, 'Name and region', is active. The bucket name is set to 'aws-glue-hol-123456789' and the region is set to 'Asia Pacific (Seoul)'. The 'Create' button is highlighted with a red box, indicating it should be clicked to proceed.

aws
Services ▾ Resource Groups ▾
hyun @ aws-korea ▾ Global ▾ Support ▾

Prevent S3 objects from being deleted for a predefined retention period with S3 Object Lock. [Learn more »](#)

Create bucket

1 Name and region 2 Configure options 3 Set permissions 4 Review

Name and region

Bucket name ⓘ
aws-glue-hol-123456789

Region
Asia Pacific (Seoul) ▾

Copy settings from an existing bucket

Select bucket (optional) 29 Buckets ▾

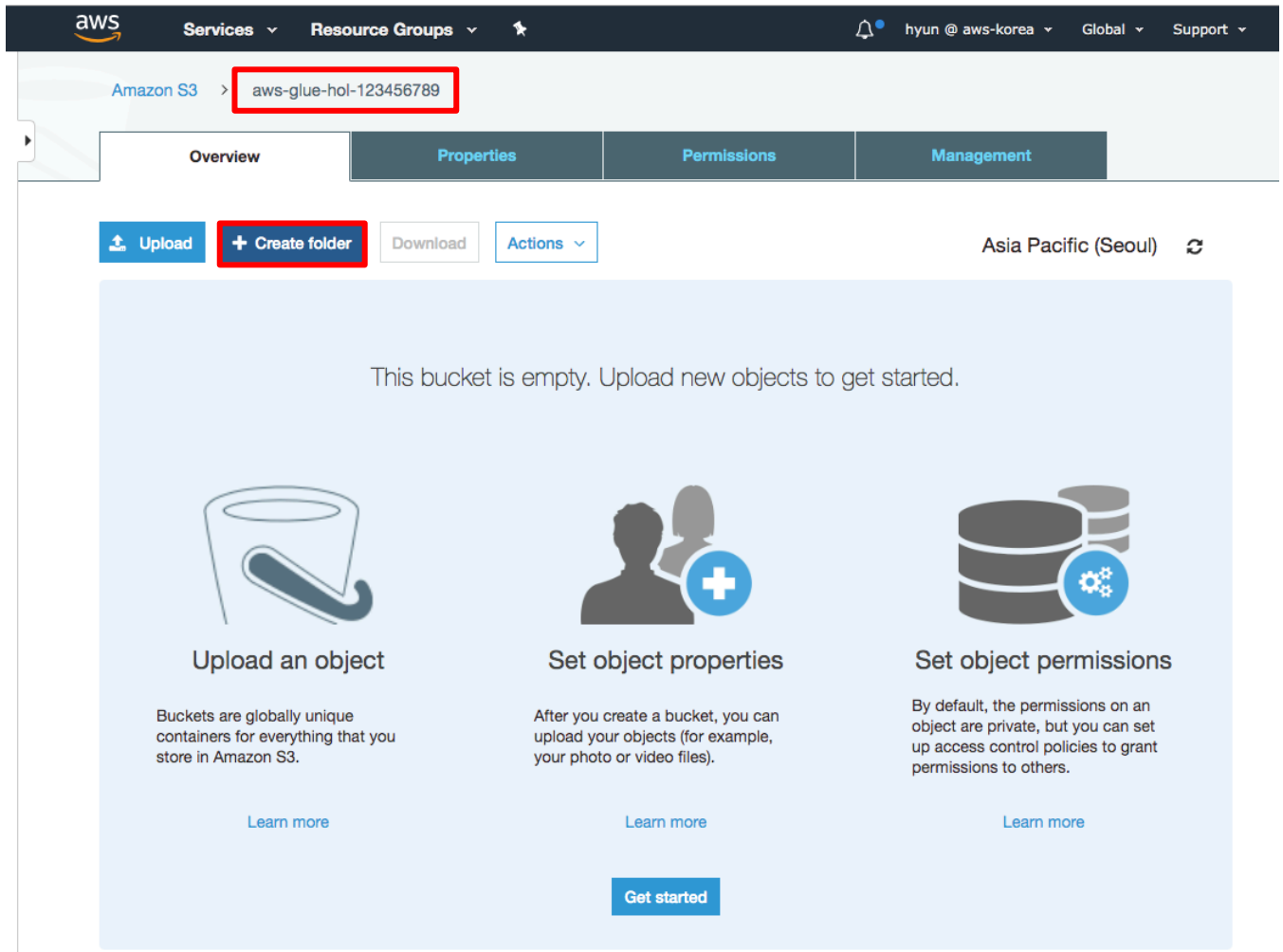
Create Cancel Next

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

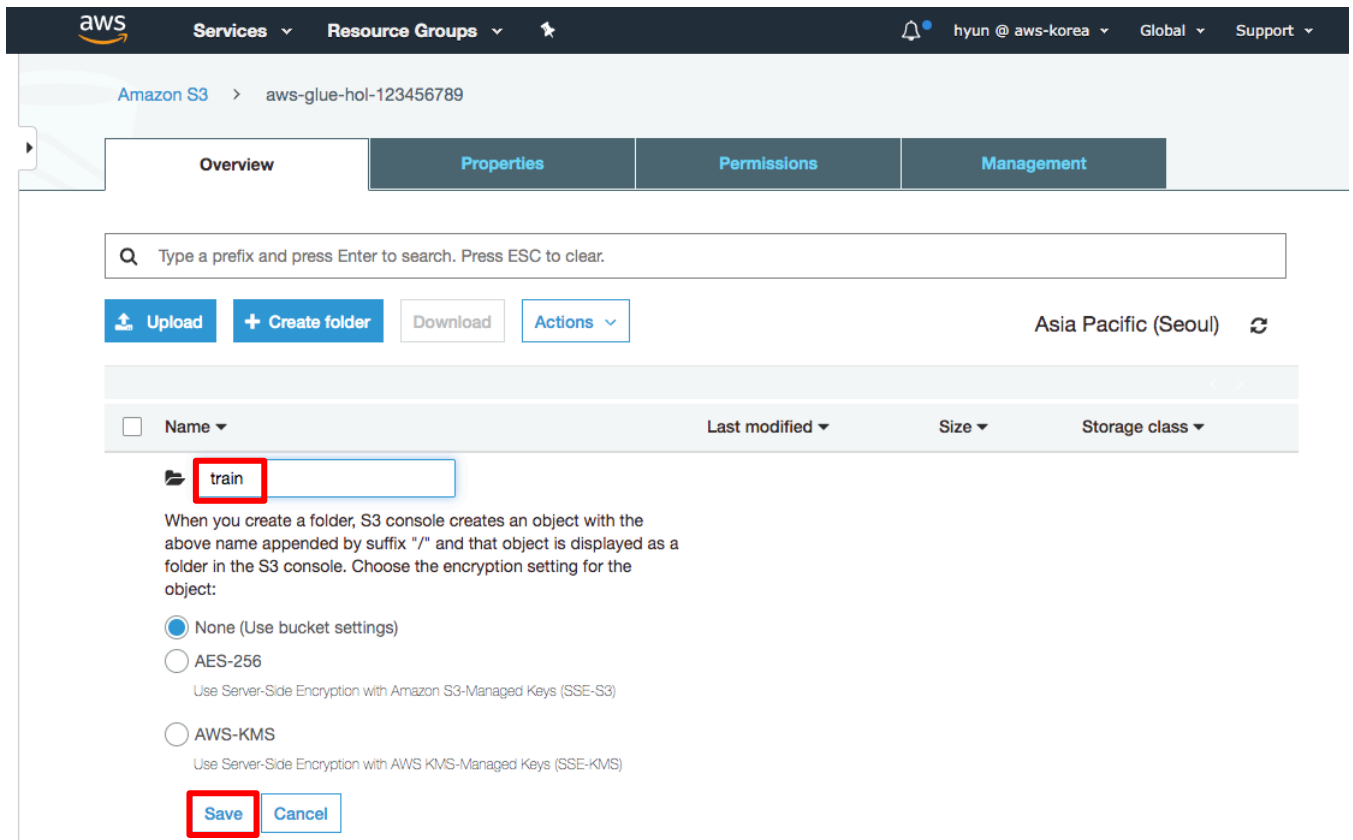
The screenshot shows the AWS Management Console interface for Amazon S3. The left sidebar contains navigation options: Amazon S3, Buckets, Batch operations, Block public access (account settings), and Feature spotlight (2). The main content area is titled 'S3 buckets' and includes a search bar, a dropdown for 'All access types', and buttons for '+ Create bucket', 'Edit public access settings', 'Empty', and 'Delete'. It displays a list of 29 buckets across 4 regions. The bucket 'aws-glue-hol-123456789' is highlighted with a red box.

<input type="checkbox"/>	Bucket name	Access	Region	Date created
<input type="checkbox"/>	aws-athena-query-results-011905145951-ap-n...	Objects can be public	Asia Pacific (Seoul)	Oct 17, 2018 9:52:53 AM GMT+0900
<input type="checkbox"/>	aws-athena-query-results-011905145951-us-w...	Objects can be public	US West (Oregon)	Apr 14, 2019 6:34:44 PM GMT+0900
<input type="checkbox"/>	aws-athena-query-results-ap-southeast-1-0119...	Objects can be public	Asia Pacific (Singapore)	May 17, 2019 10:53:49 AM GMT+0900
<input type="checkbox"/>	aws-athena-query-results-us-west-2-01190514...	Objects can be public	US West (Oregon)	Mar 17, 2019 11:46:49 AM GMT+0900
<input type="checkbox"/>	aws-emr-resources-011905145951-ap-northeas...	Objects can be public	Asia Pacific (Seoul)	Dec 13, 2018 5:53:53 PM GMT+0900
<input type="checkbox"/>	aws-glue-hol-123456789	Bucket and objects not public	Asia Pacific (Seoul)	Jun 3, 2019 10:38:26 AM GMT+0900

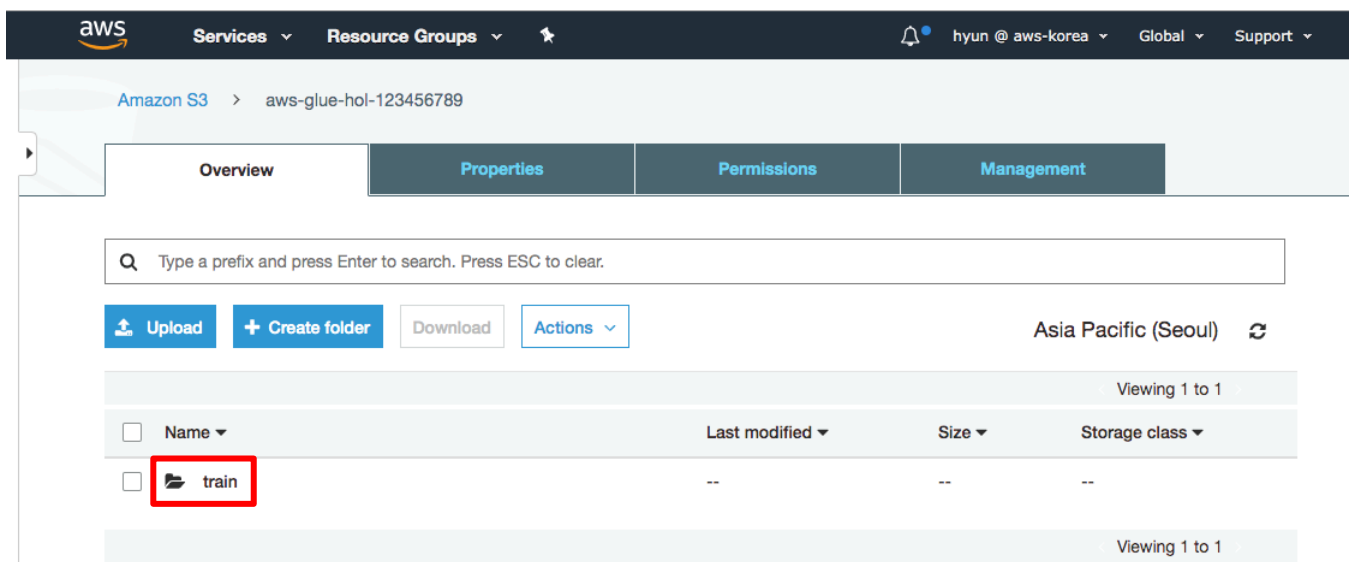
2-5. train 폴더를 생성하기 위해 Create folder 버튼을 클릭합니다.



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



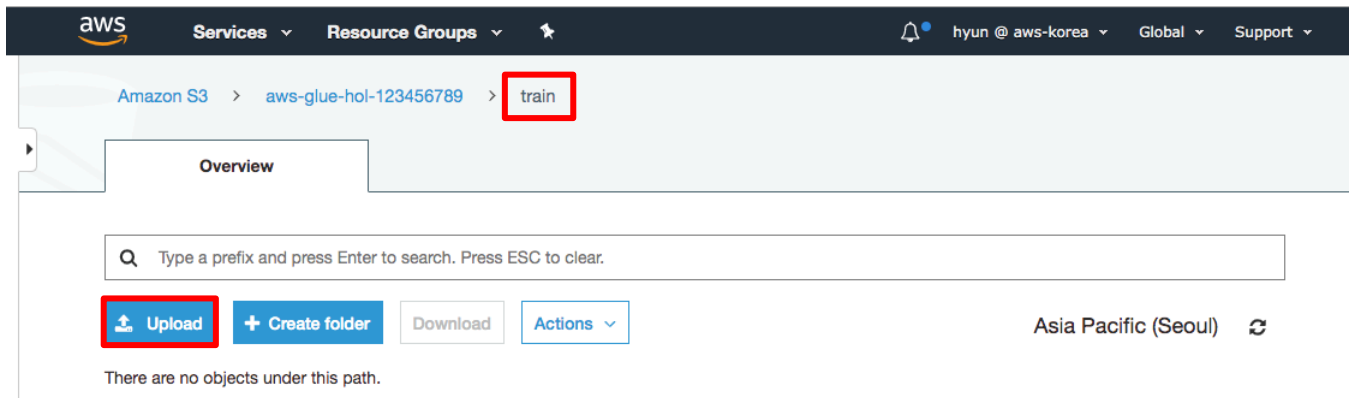
2-7. 아래와 같이 aws-glue-hol-[account-id] 버킷에 총 1 개의 폴더가 생성된 것을 확인할 수 있습니다.



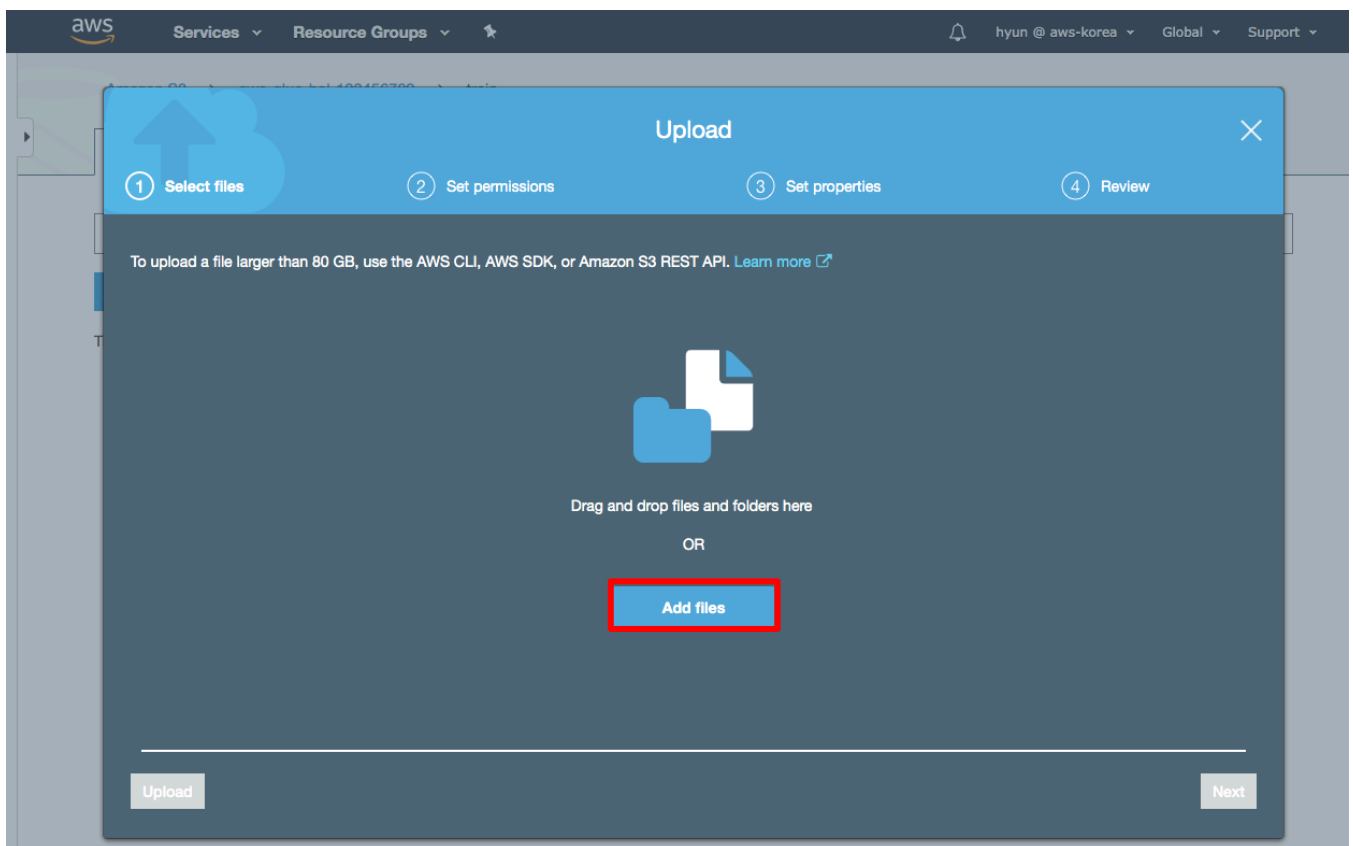
파일 업로드

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>

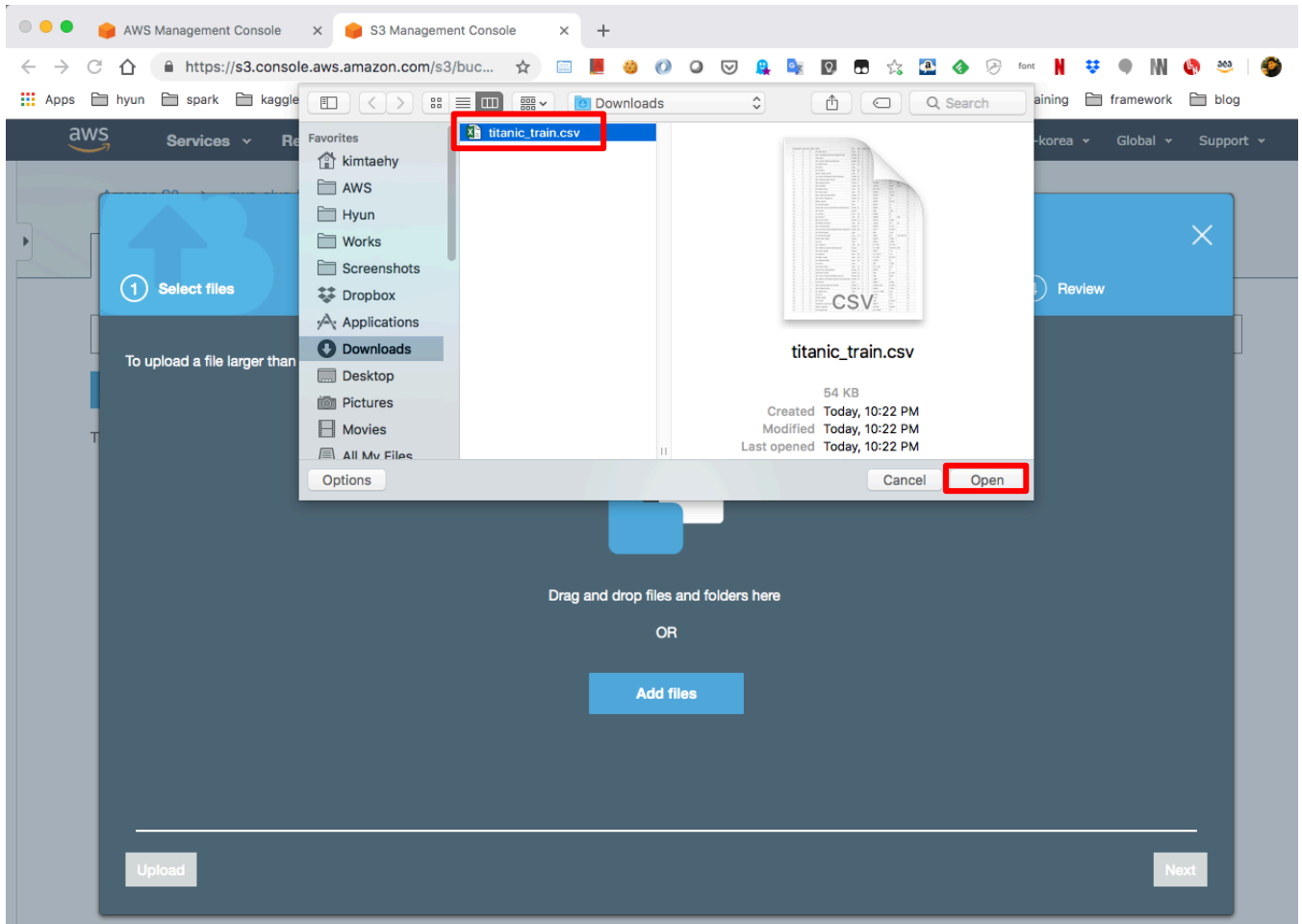


2-9. Add files 를 클릭합니다.

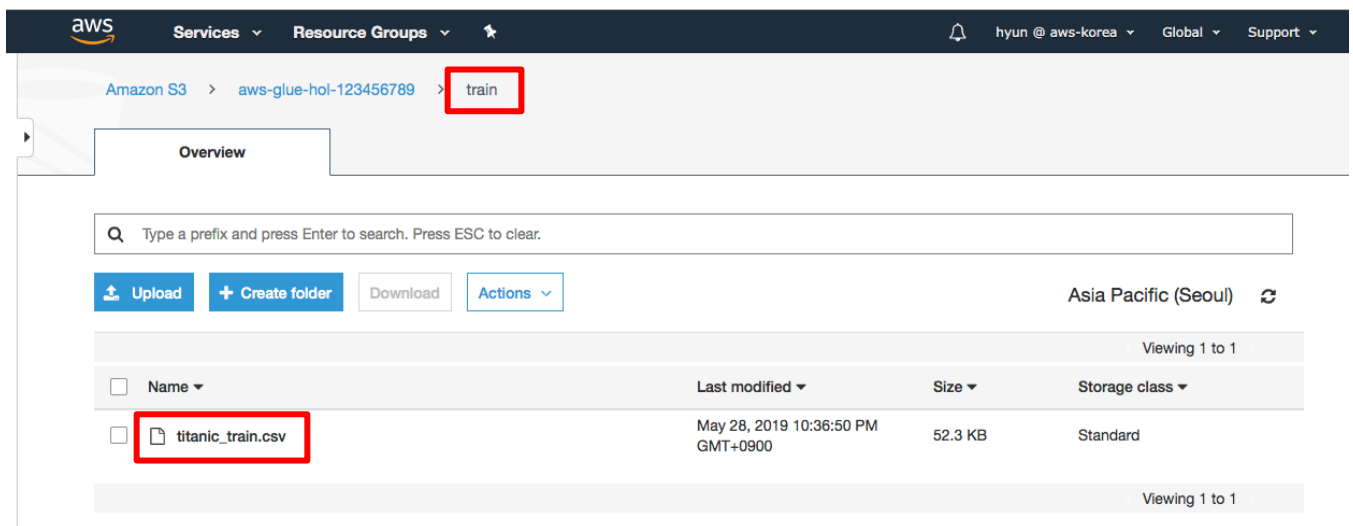
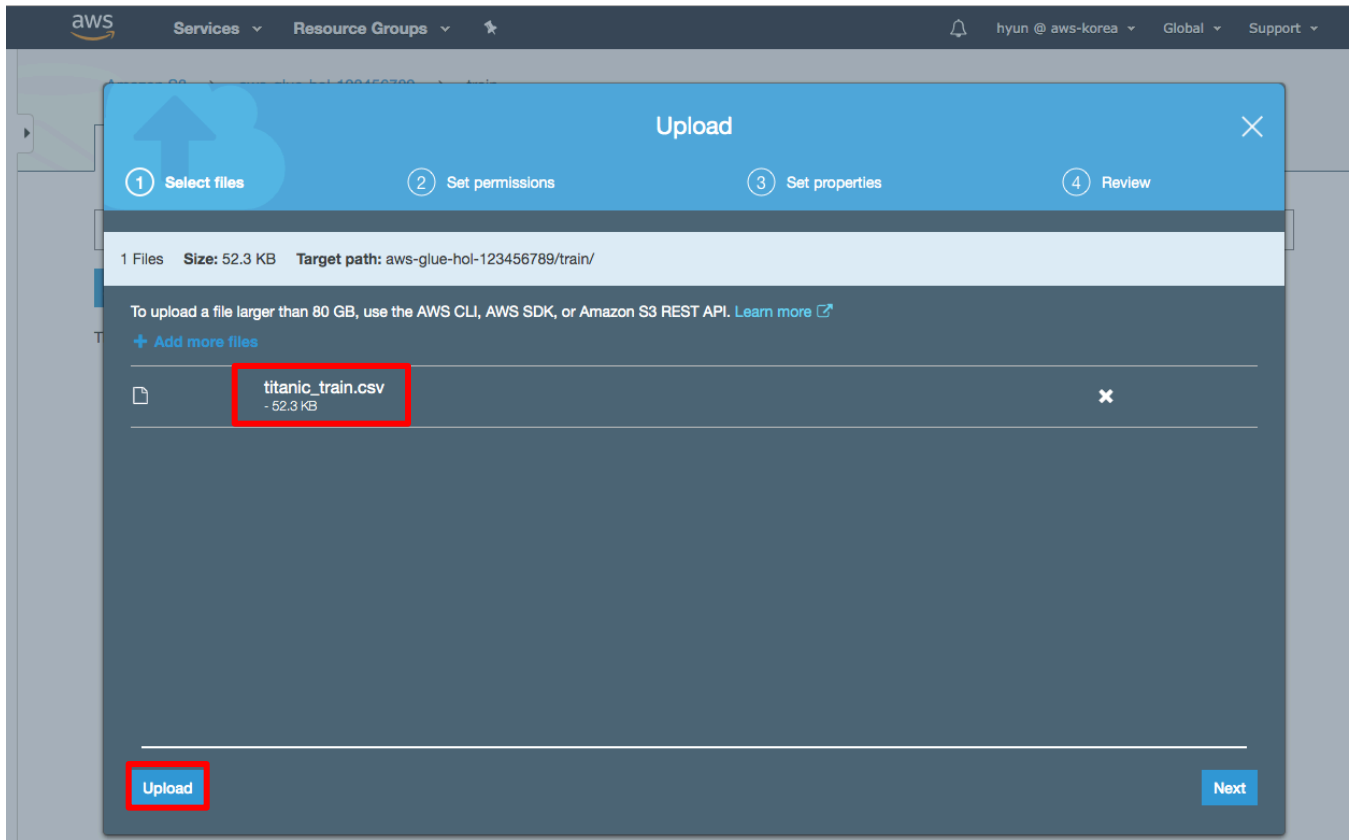


2-10. 다운로드 받은 titanic_train.csv 파일을 선택합니다.

(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.

```
titanic_train.csv
1 PassengerId,Survived,Pclass,Name,Sex,Age,SibSp,Parch,Ticket,Fare,Cabin,Embarked
2 1,0,3,"Mr. Owen Harris",male,22,1,0,A/5 21171,7.25,,S
3 2,1,1,"Mrs. John Bradley (Florence Briggs Thayer)",female,38,1,0,PC 17599,71.2833,C85,C
4 3,1,3,"Miss. Laina",female,26,0,0,STON/O2. 3101282,7.925,,S
5 4,1,1,"Mrs. Jacques Heath (Lily May Peel)",female,35,1,0,113803,53.1,C123,S
6 5,0,3,"Mr. William Henry",male,35,0,0,373450,8.05,,S
7 6,0,3,"Mr. James",male,,0,0,330877,8.4583,,Q
8 7,0,1,"Mr. Timothy J",male,54,0,0,17463,51.8625,E46,S
9 8,0,3,"Master. Gosta Leonard",male,2,3,1,349909,21.075,,S
10 9,1,3,"Mrs. Oscar W (Elisabeth Vilhelmina Berg)",female,27,0,2,347742,11.1333,,S
```

3. Create Glue Catalog and Crawling

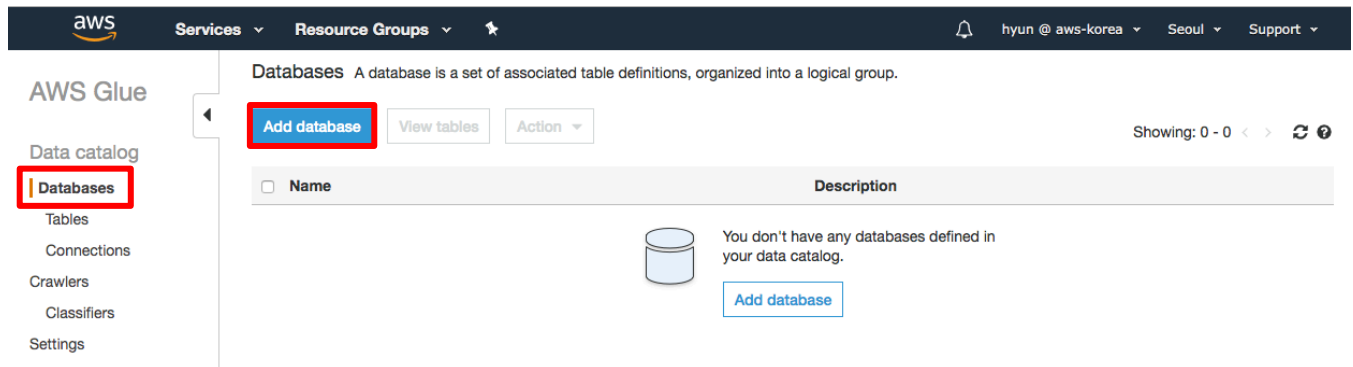
본 단계에서는 AWS Glue 를 통해 데이터 Catalog 를 생성할 수 있도록 Crawling 작업을 구성합니다.

Data Catalog databases 생성

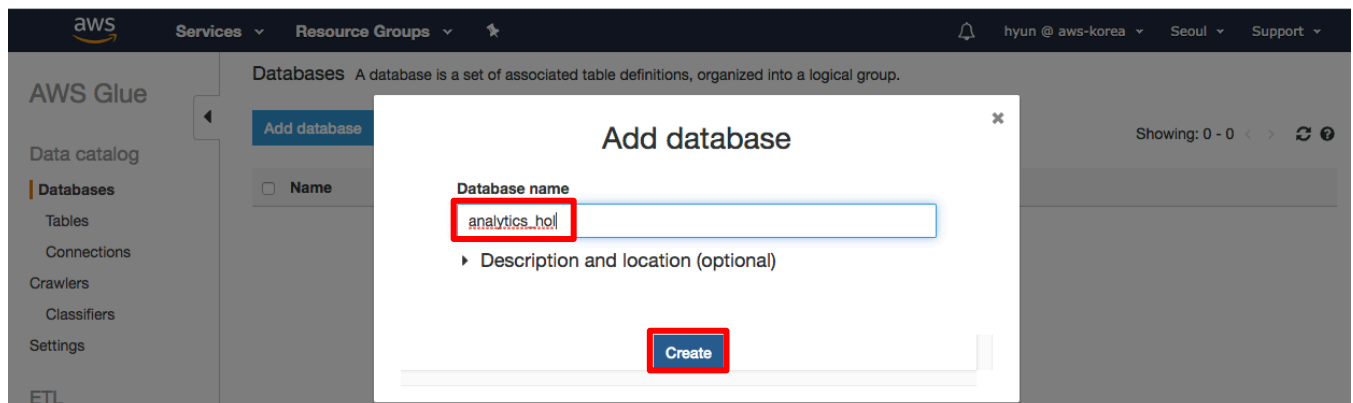
3-1. 서비스에서 AWS Glue 를 선택하여 Glue 콘솔로 이동합니다. 이전에 Glue 를 사용한 적이 없는 경우 초기 안내화면에서 Get started 버튼을 클릭합니다.

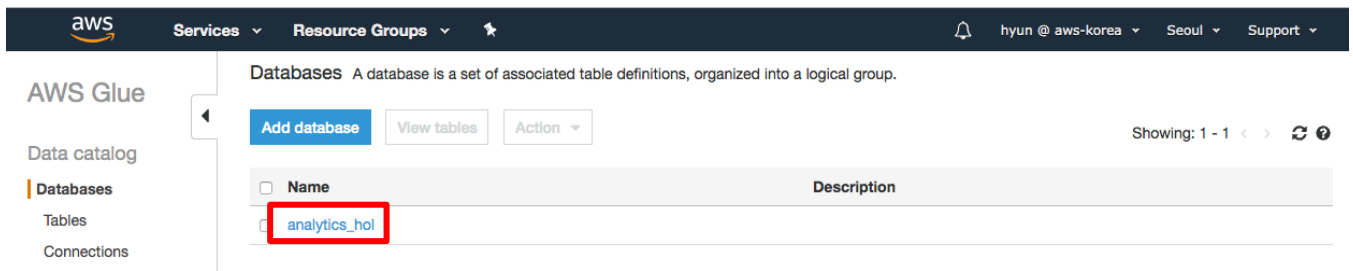


3-2. Glue 콘솔에서 Databases 를 선택한 후 Add database 버튼을 클릭합니다.



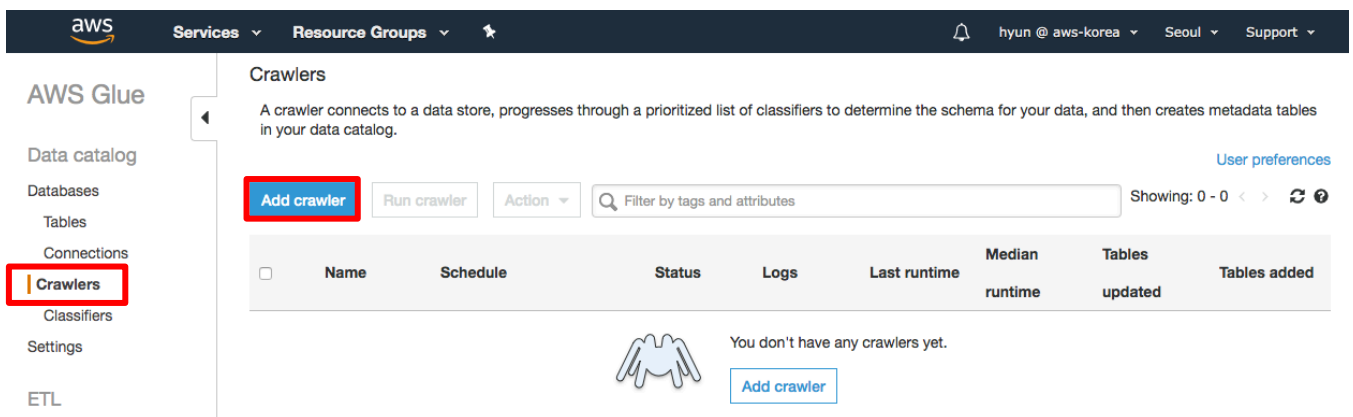
3-3. Database name 에 analytics_hol 를 입력한 후 Create 버튼을 클릭합니다.



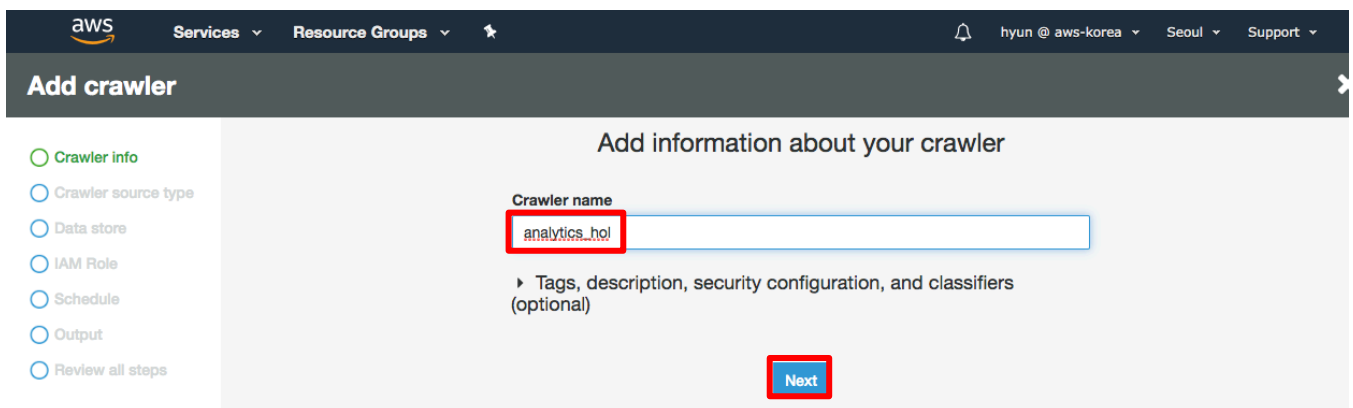


Crawler 및 Glue Catalog 생성

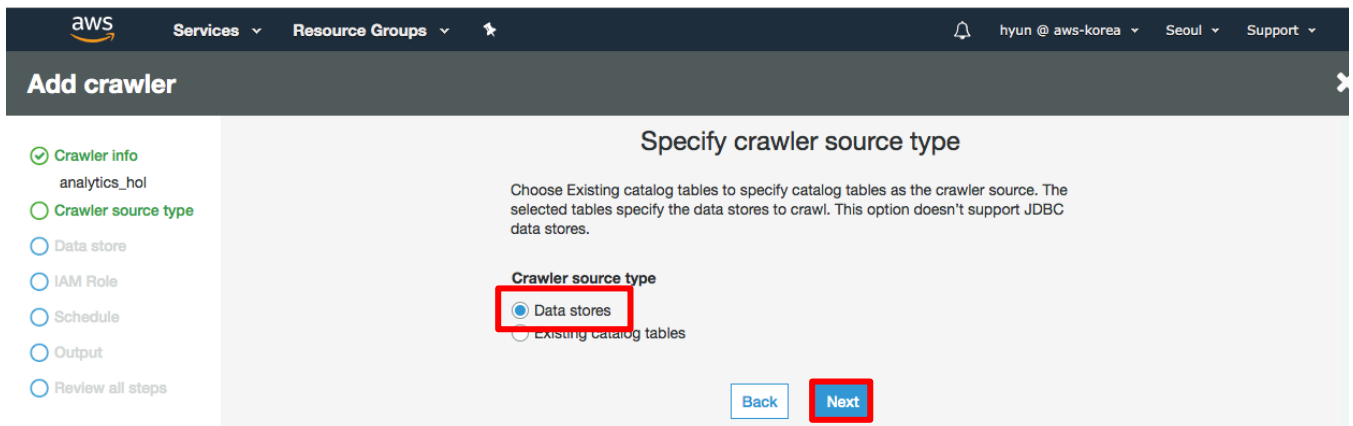
3-4. Glue 콘솔에서 Crawler 를 클릭해서 이동한 후 Add Crawler 버튼을 클릭합니다.



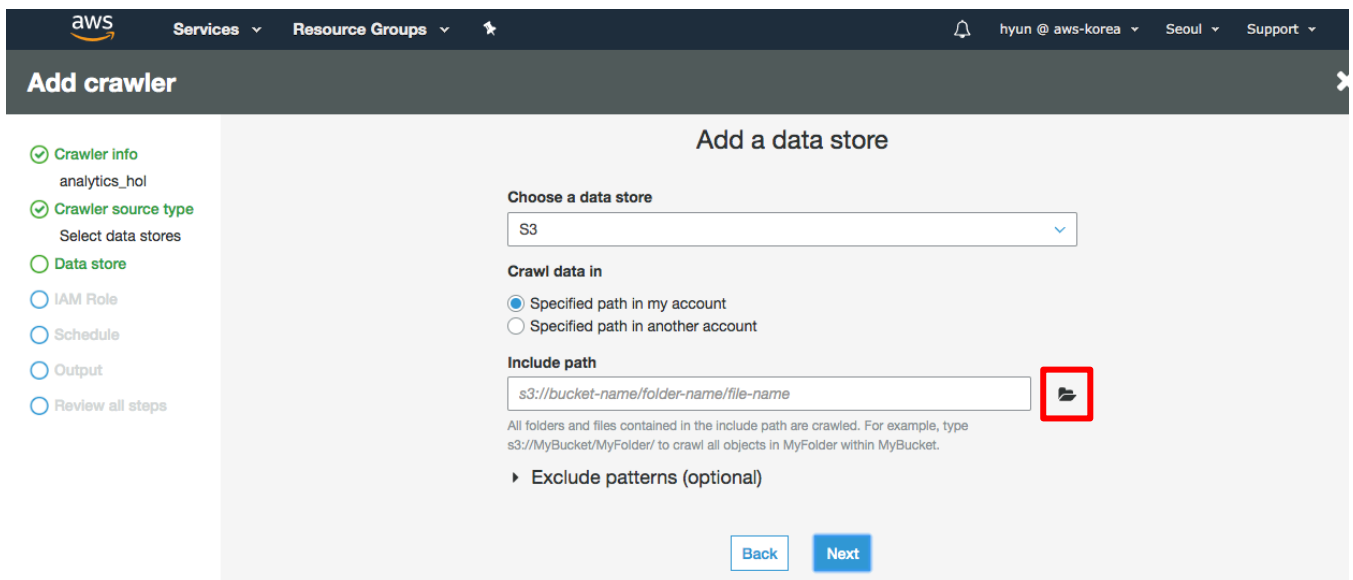
3-5. Crawler info 설정에서 Crawler name 을 analytics_hol 로 입력하고 Next 를 클릭합니다.



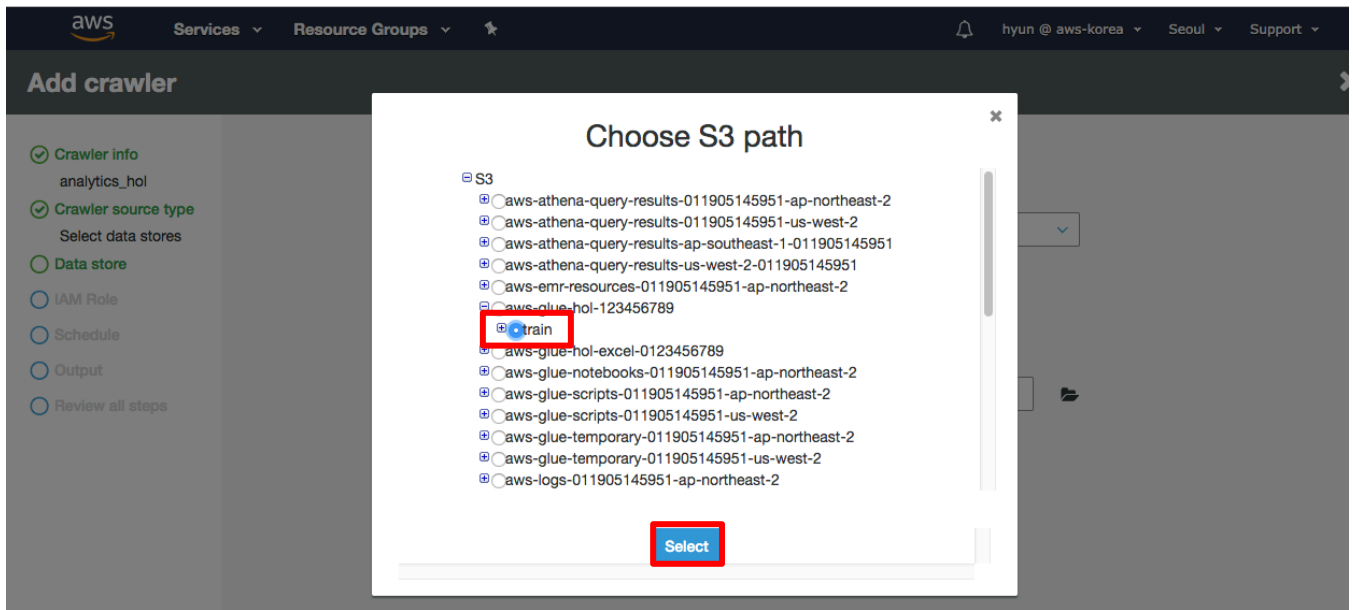
3-6. Crawler source type 설정에서 Data stores 를 선택하고 Next 를 클릭합니다.



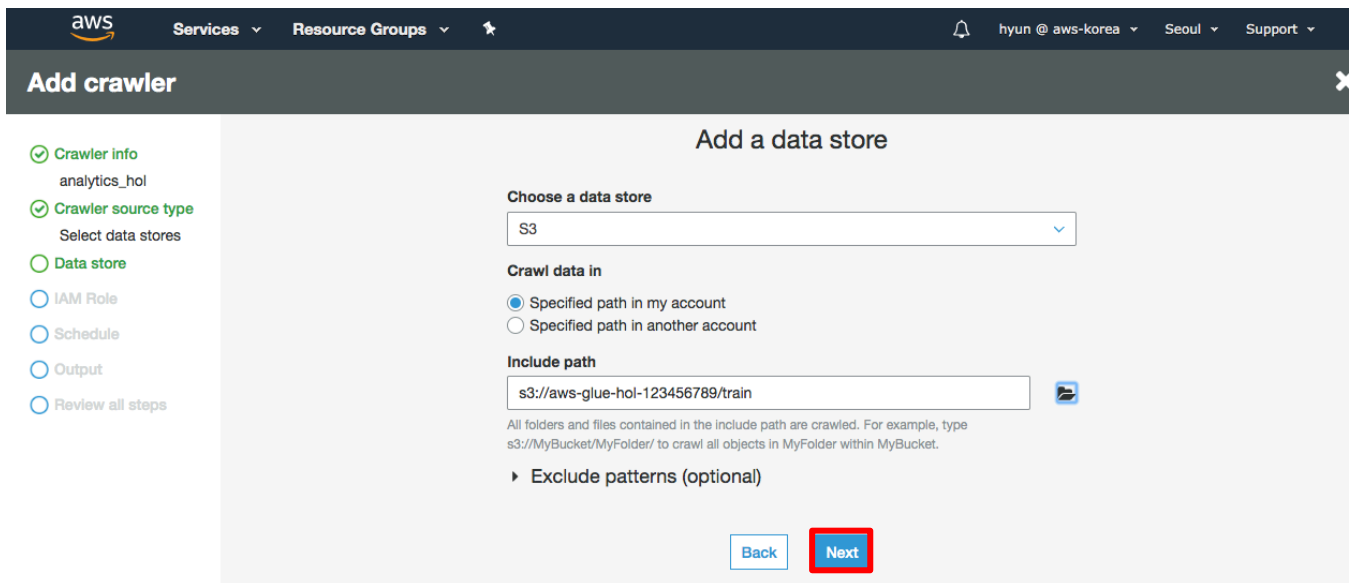
3-7. Data store 설정에서 Include path 오른쪽의 폴더 아이콘을 클릭합니다.



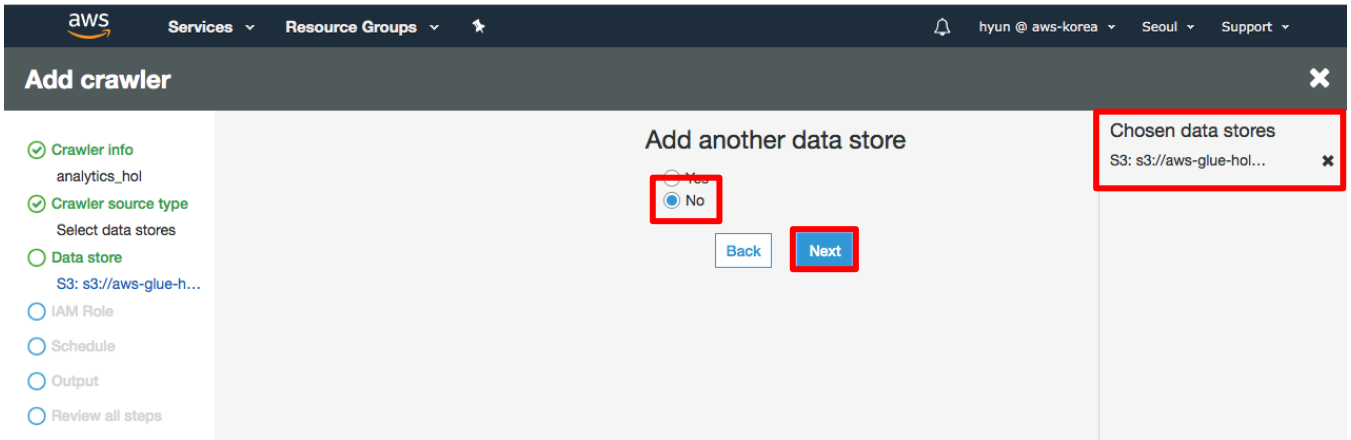
3-8. 팝업 화면에서 aws-glue-hol-[accout-id]/train 폴더를 선택하고 Select 버튼을 클릭합니다.
(파일이 아닌 디렉토리를 선택합니다.)



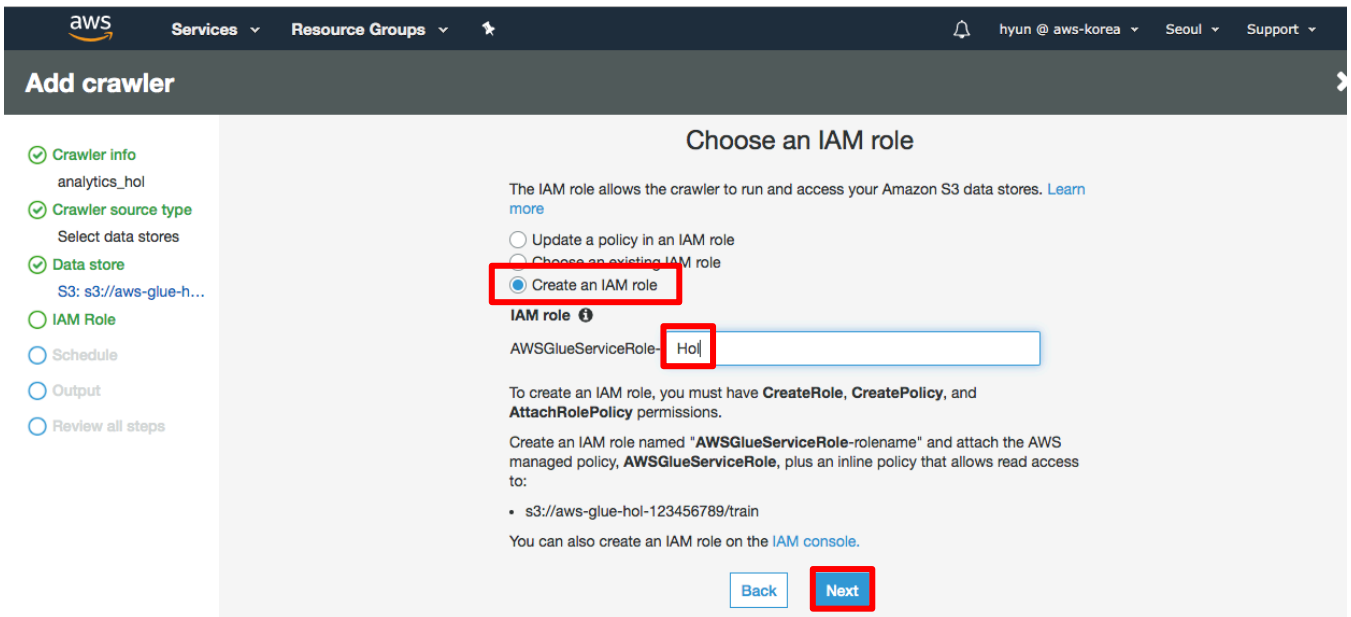
3-9. 나머지 설정을 Default 로 두고 Next 를 클릭합니다.



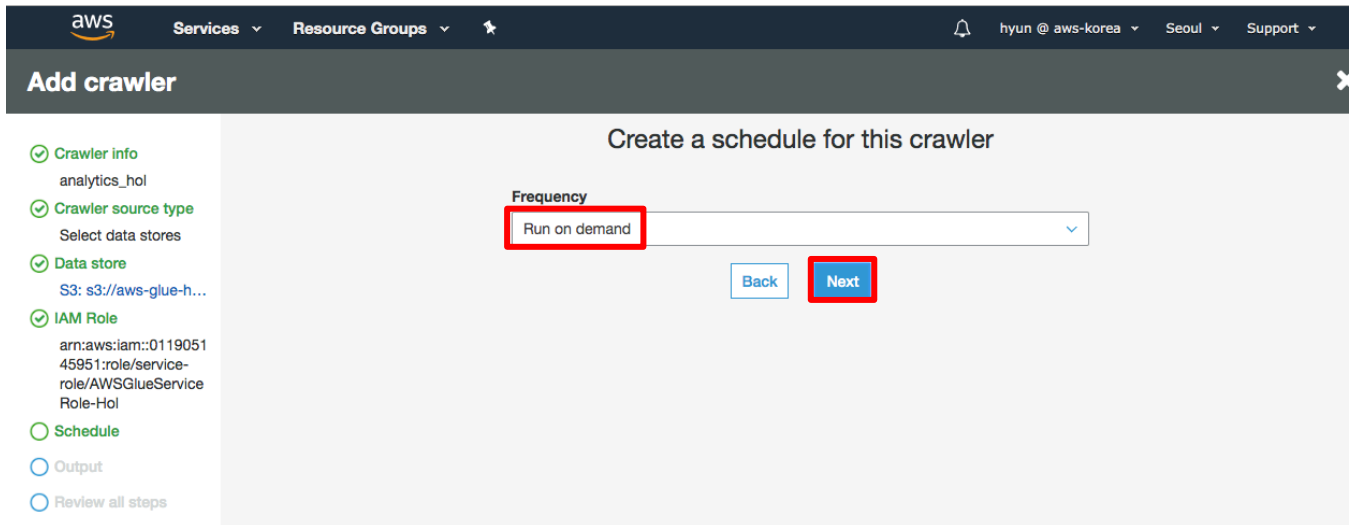
3-10. Add another data store 화면에서 No 를 선택된 상태로 Next 를 클릭합니다. 오른쪽 Chosen data stores 에 앞서 선택한 파일이 표시되는 것을 확인할 수 있습니다.



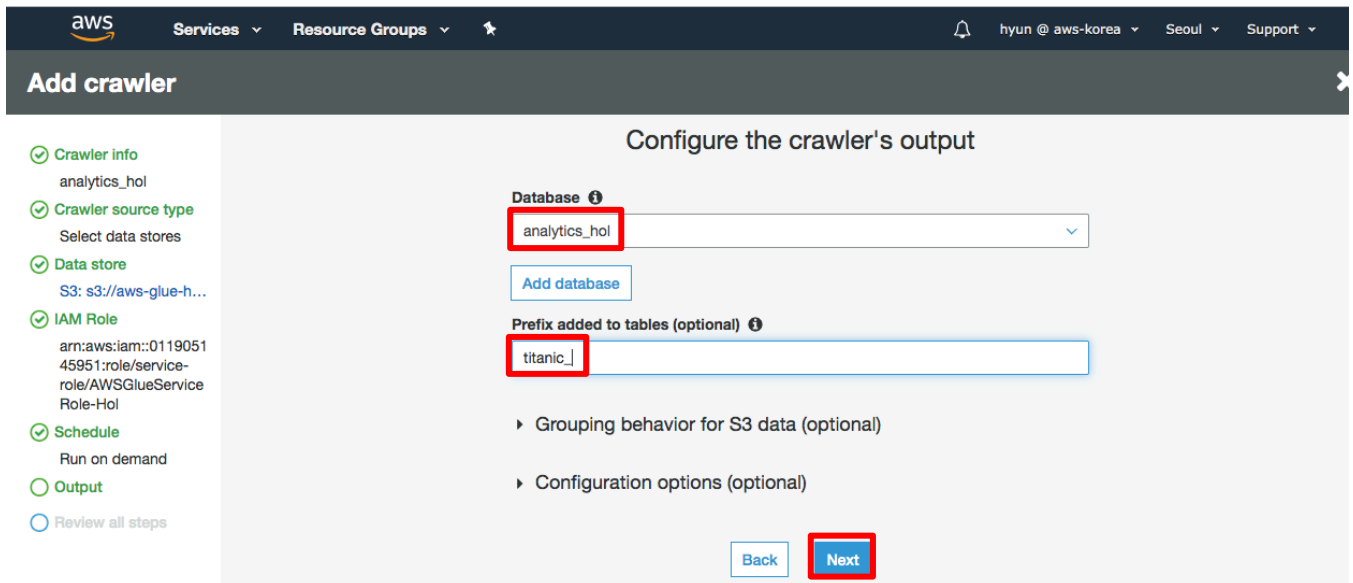
3-11. IAM Role 설정에서 Create an IAM role 을 선택합니다. IAM role 에 Hol 을 입력하고 Next 버튼을 클릭합니다.



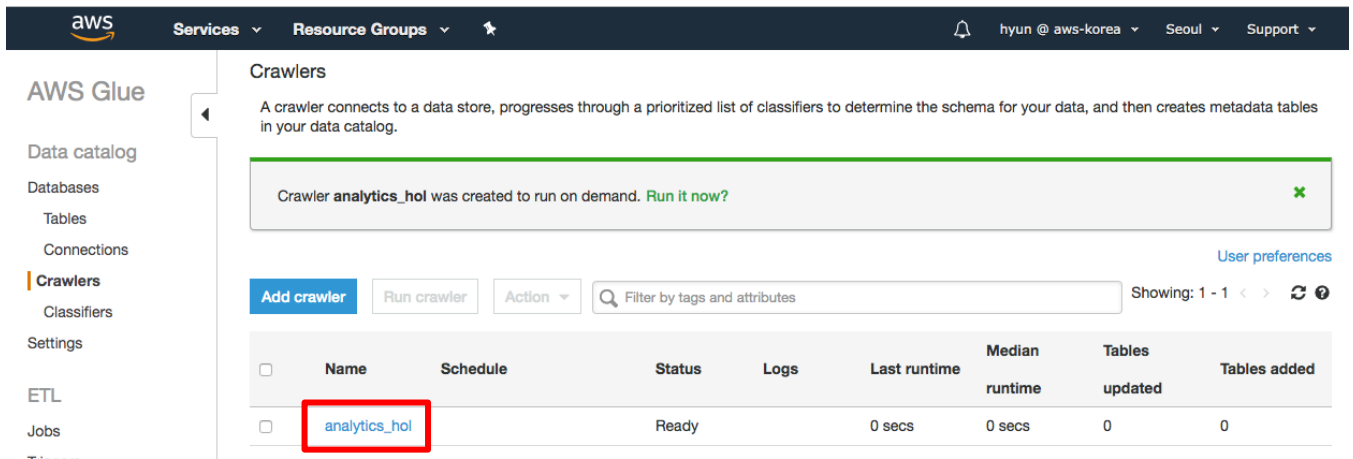
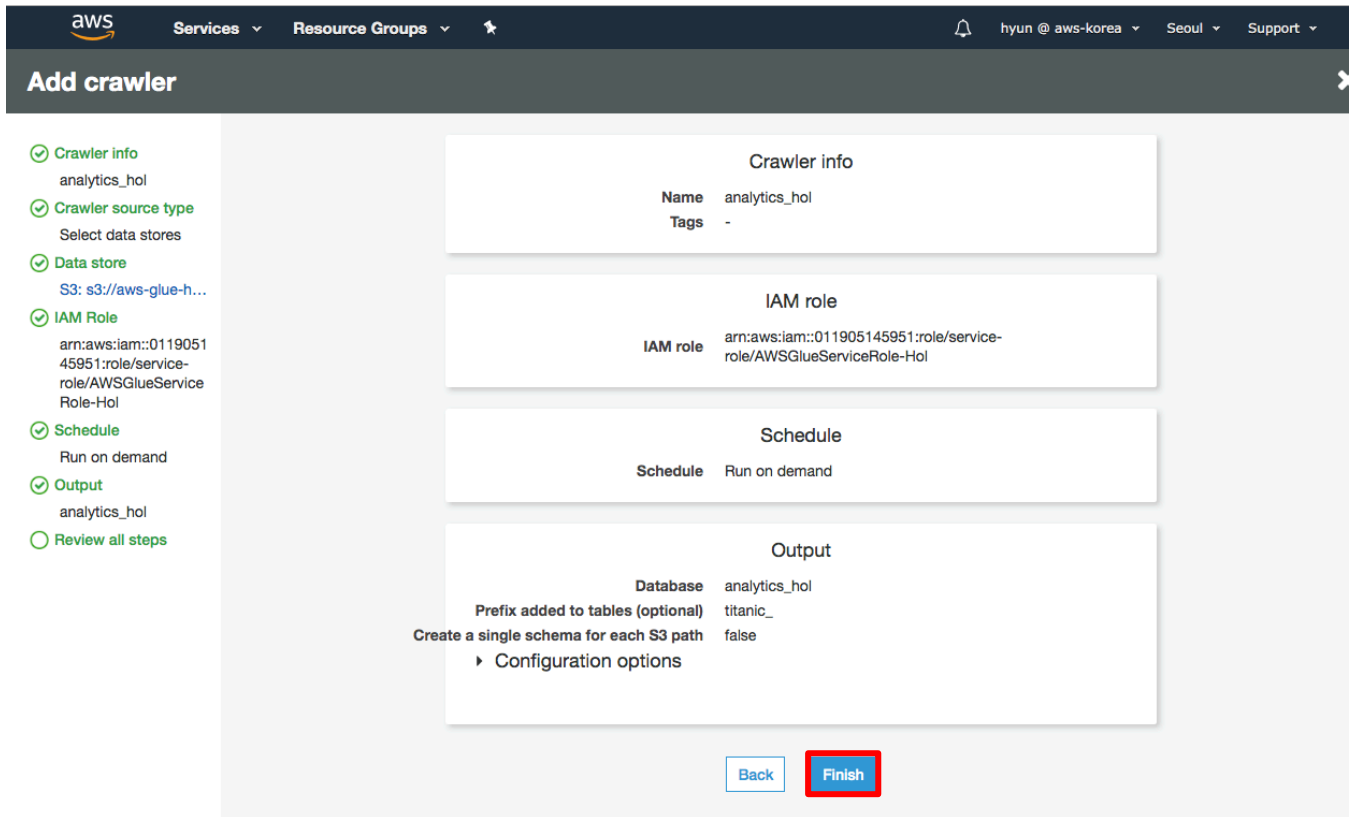
3-12. Schedule 설정은 Run on demand 상태로 두고 Next 를 클릭합니다.



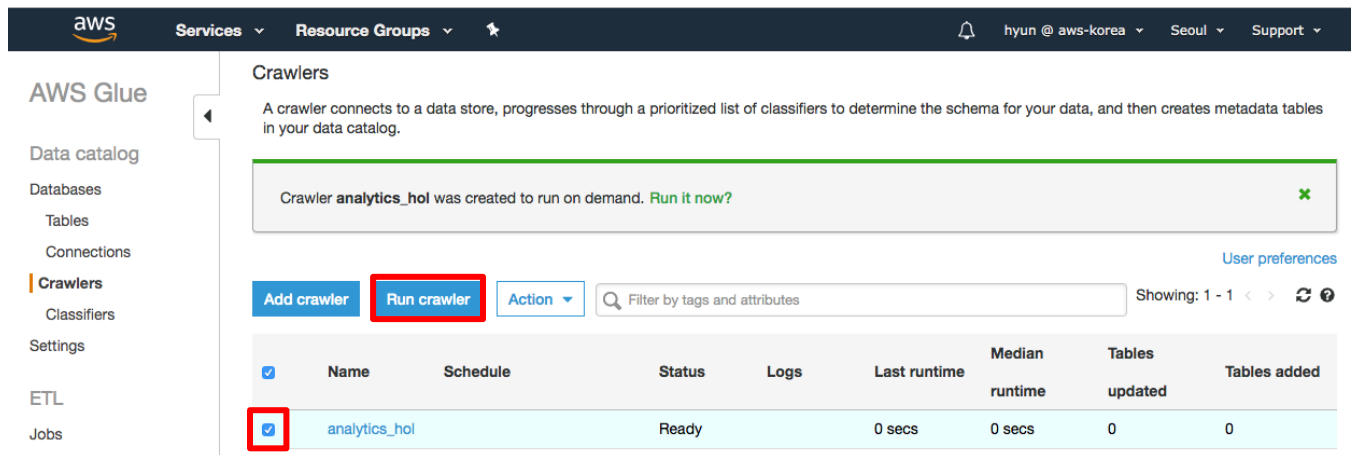
3-13. Output 설정에서는 Output Database 를 analytics_hol 로 선택합니다. 생성할 table 에 prefix 를 붙이기 위해 Prefix added to tables[optional]에 titanic_ 를 입력하고 Next 버튼을 클릭합니다.



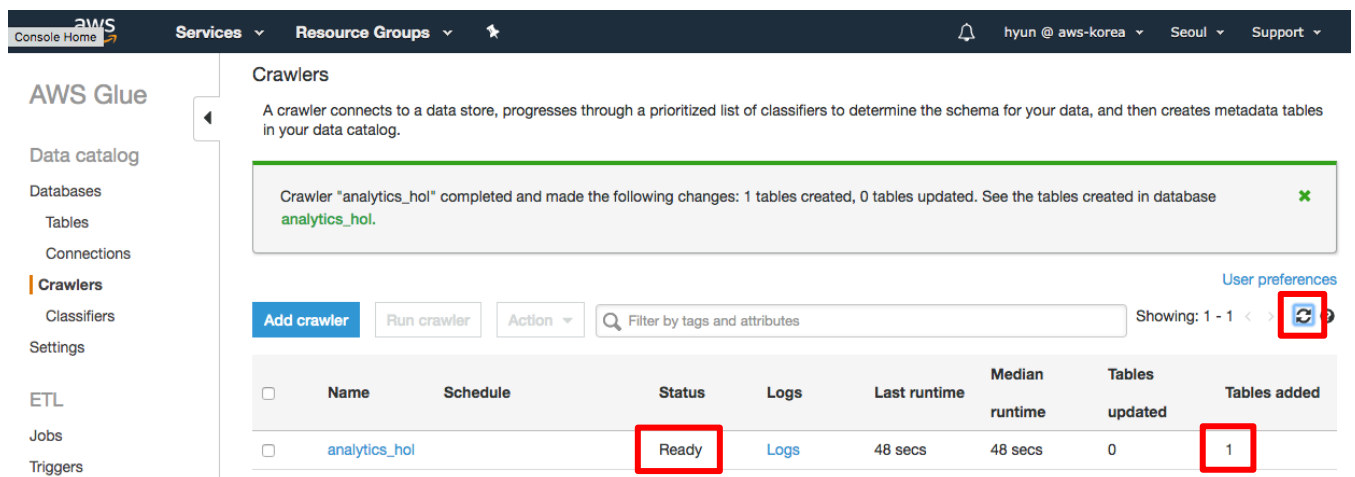
3-14. 내용을 Review 한 후 Finish 버튼을 클릭하면 첫번째 Crawler Job 이 생성됩니다.



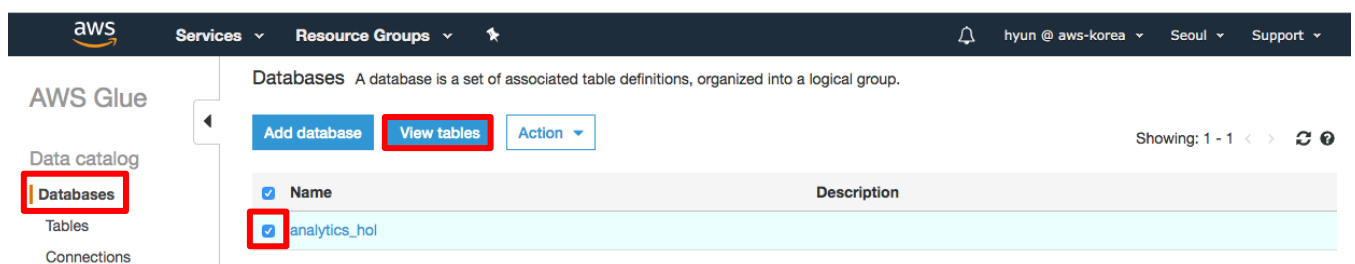
3-15. Job 체크박스를 선택한 후 Run crawler 를 클릭하여 생성한 Crawler 를 실행합니다.



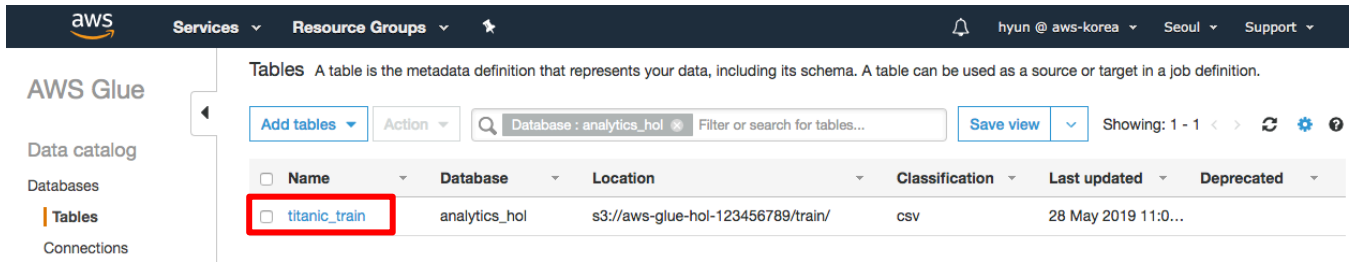
3-16. Crawler Job 이 완료되면 Ready 상태가 되고, 맨 오른쪽 컬럼에서 1 개의 Table 이 생성된 것을 확인할 수 있습니다. (Refresh 버튼으로 상태 변경을 확인할 수 있습니다.)



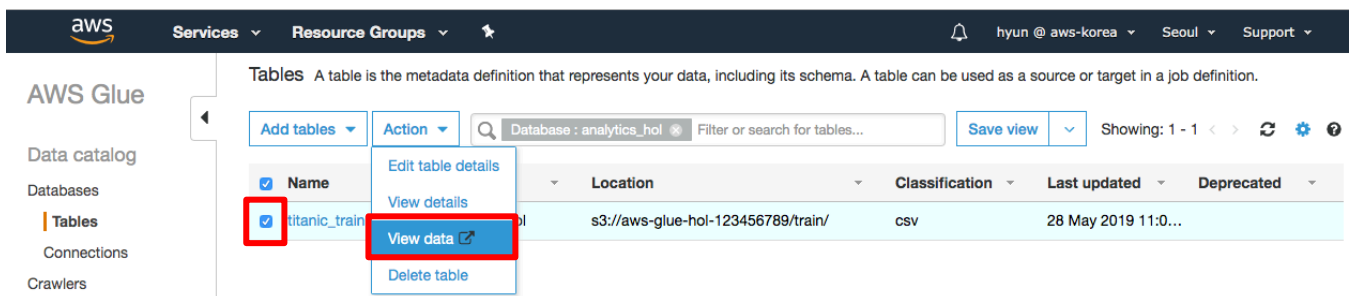
3-17. Databases 메뉴로 이동합니다. analytics_hol database 를 선택하고 View tables 버튼을 클릭합니다.



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.



The screenshot displays the AWS Athena Query Editor interface. At the top, the navigation bar includes the AWS logo, 'Services', 'Resource Groups', and user information 'hyun @ aws-korea' in the 'Seoul' region. The main header shows 'Athena Query Editor' with tabs for 'Saved Queries', 'History', 'AWS Glue Data Catalog', and 'Workgroup : primary'. On the left, a 'Database' sidebar shows 'analytics_hol' selected, with a search filter and a list of tables including 'titanic_train'. The central editor contains a SQL query: `1 SELECT * FROM "analytics_hol"."titanic_train" limit 10;`. Below the query are buttons for 'Run query', 'Save as', 'Create', and 'Format query', along with a status message: '(Run time: 3.15 seconds, Data scanned: 52.32 KB)'. The bottom section, titled 'Results', displays a table with 10 rows of passenger data, including columns for passenger ID, survival status, class, name, sex, age, siblings/parents, ticket, fare, and cabin.

	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	

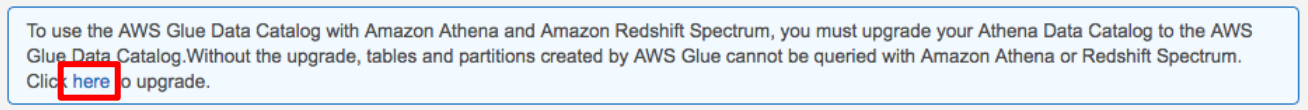
4. Exploring Data Analysis with Athena

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.

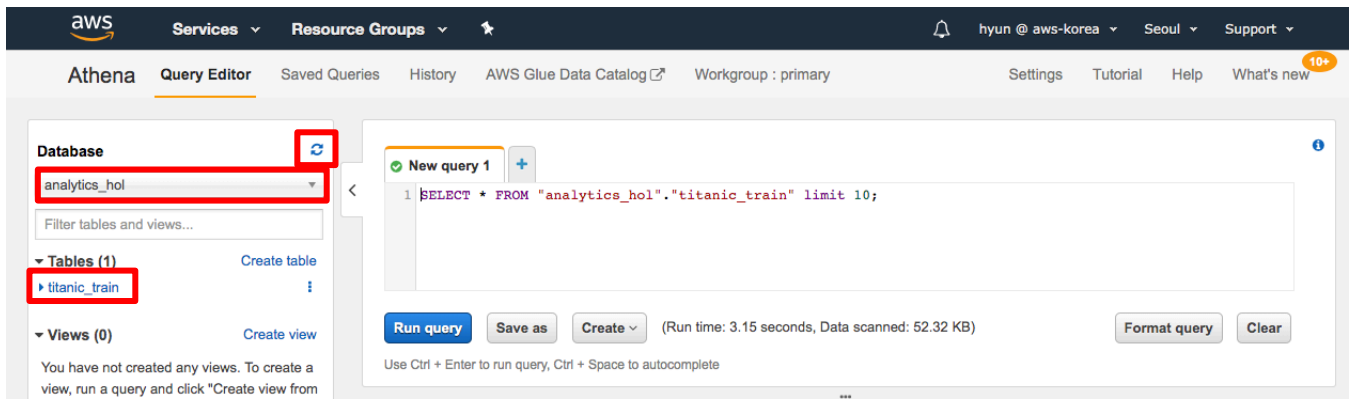
<https://docs.aws.amazon.com/athena/latest/ug/glue-athena.html>




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.

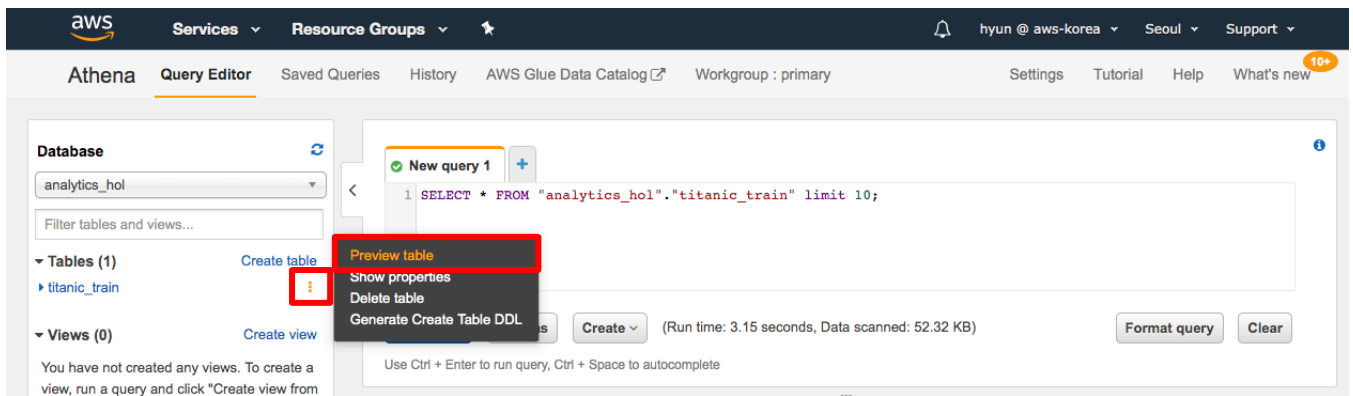


업그레이드가 완료되면 왼쪽 Database 네비게이션에서 analytics_hoi 데이터베이스를 선택할 수 있고 Crawling 작업을 통해 생성한 titanic_train 테이블을 볼 수 있습니다. 데이터베이스나 테이블이 보이지 않을 경우 refresh 버튼을 클릭하세요.



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.



The screenshot shows the AWS Athena Query Editor interface. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and user information 'hyun @ aws-korea'. The main header shows 'Athena Query Editor' and 'Workgroup : primary'. The left sidebar displays the 'Database' section with 'analytics_hol' selected and a list of tables including 'titanic_train'. The main area contains a query editor with the following SQL query:

```
1 SELECT * FROM "analytics_hol"."titanic_train" limit 10;
```

Below the query editor, there are buttons for 'Run query', 'Save as', and 'Create', along with a status message: '(Run time: 3.04 seconds, Data scanned: 52.32 KB)'. The 'Results' section is highlighted with a red border and displays a table with 10 rows of data:

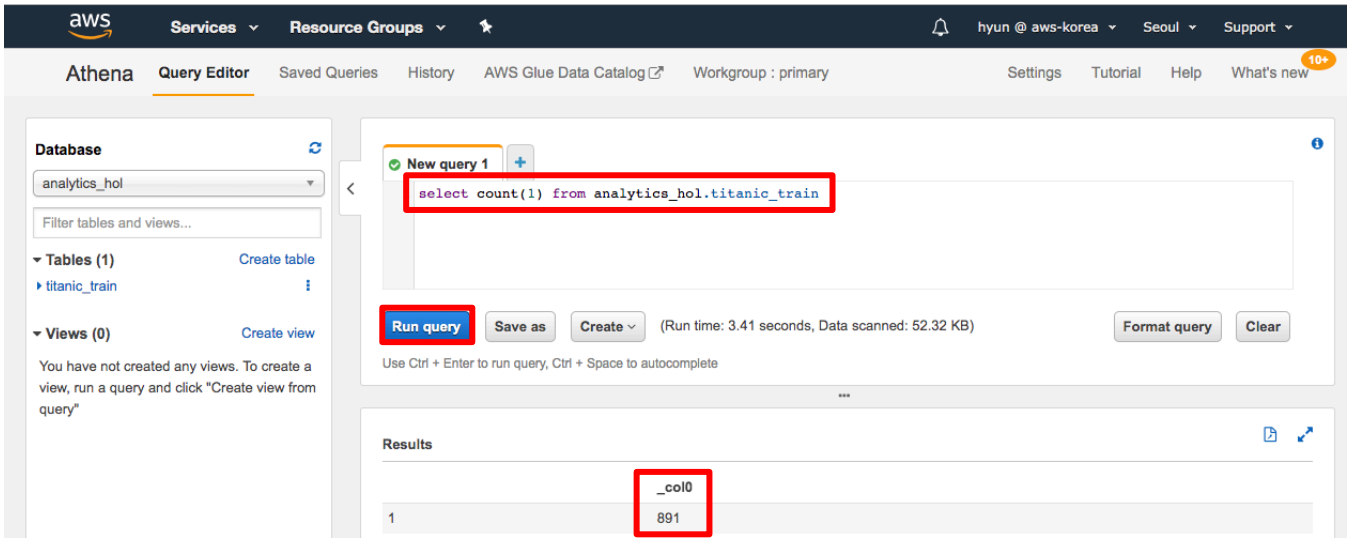
	passengerid	survived	pclass	name	sex	age	sibsp	parch	ticket	fare	c
1	1	0	3	"Mr. Owen Harris"	male	22.0	1	0	A/5 21171	7.25	
2	2	1	1	"Mrs. John Bradley (Florence Briggs Thayer)"	female	38.0	1	0	PC 17599	71.2833	C
3	3	1	3	"Miss. Laina"	female	26.0	0	0	STON/O2. 3101282	7.925	
4	4	1	1	"Mrs. Jacques Heath (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	

4-3. 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
- sibsp: 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)

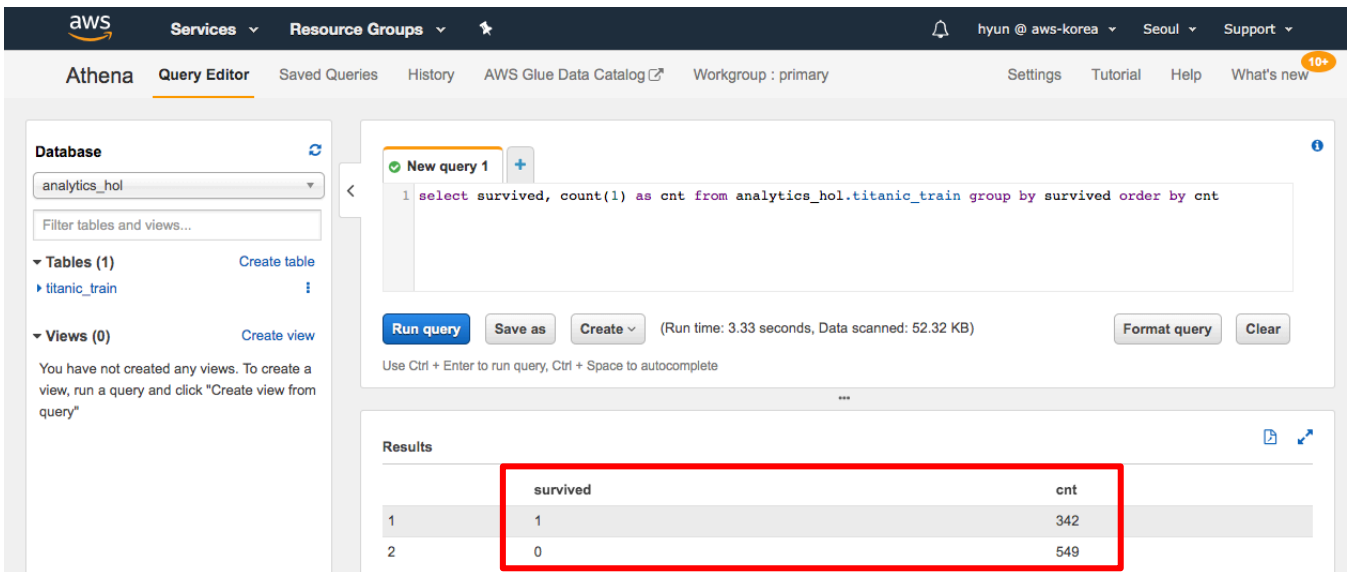
4-4. 전체 데이터 수 확인

`select count(1) from analytics_hol.titanic_train`



4.5. 탑승객 생존율 확인

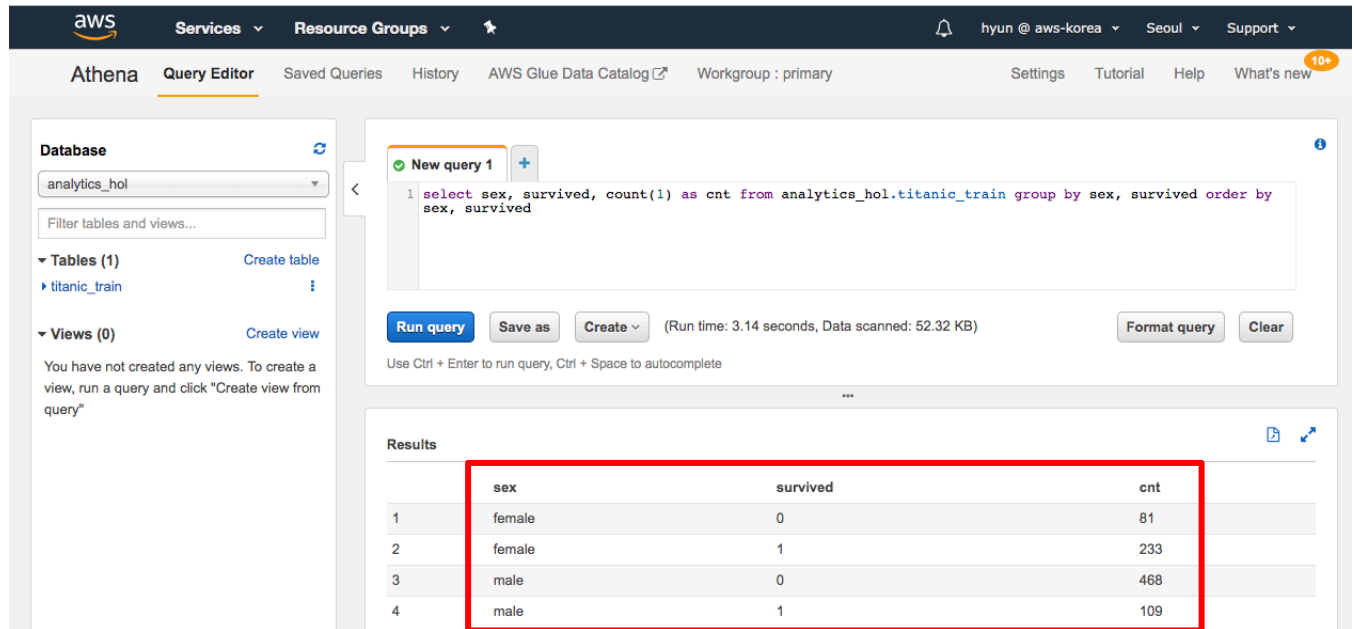
`select survived, count(1) as cnt from analytics_hol.titanic_train group by survived order by cnt`



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
```



The screenshot shows the AWS Athena Query Editor interface. The query editor contains the following SQL query:

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

The query has been executed, and the results are displayed in a table. The table has four columns: **sex**, **survived**, and **cnt**. The results are as follows:

	sex	survived	cnt
1	female	0	81
2	female	1	233
3	male	0	468
4	male	1	109

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
```

The screenshot shows the AWS Athena Query Editor interface. The query editor contains the following SQL query:

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

The query has been executed, and the results are displayed in a table. The results table has the following data:

	pclass	survived	cnt
1	1	0	80
2	1	1	136
3	2	0	97
4	2	1	87
5	3	0	372
6	3	1	119

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

The screenshot shows the AWS Athena Query Editor interface. On the left, the 'Database' section shows 'analytics_hol' and a table named 'titanic_train'. The main area contains a SQL query: `select sibsp, survived, count(1) as cnt from analytics_hol.titanic_train group by sibsp, survived order by sibsp, survived`. Below the query, there are buttons for 'Run query', 'Save as', 'Create', 'Format query', and 'Clear'. The 'Run query' button is highlighted. Below the query, the 'Results' section displays a table with 12 rows and 3 columns: 'sibsp', 'survived', and 'cnt'. A red box highlights the columns 'sibsp', 'survived', and 'cnt' in the results table.

	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

The screenshot shows the AWS Athena Query Editor interface. On the left, the 'Database' section shows 'analytics_hol' and a table named 'titanic_train'. The main area contains a SQL query: `select parch, survived, count(1) as cnt from analytics_hol.titanic_train group by parch, survived order by parch, survived`. Below the query, there are buttons for 'Run query', 'Save as', 'Create', 'Format query', and 'Clear'. The 'Run query' button is highlighted. Below the query, the 'Results' section displays a table with 12 rows and 3 columns: 'parch', 'survived', and 'cnt'. The table is highlighted with a red box.

	parch	survived	cnt
1	0	0	445
2	0	1	233
3	1	0	53
4	1	1	65
5	2	0	40
6	2	1	40
7	3	0	2
8	3	1	3
9	4	0	4
10	5	0	4
11	5	1	1
12	6	0	1

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
```

The screenshot shows the AWS Athena Query Editor interface. The query editor contains the following SQL query:

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

The query has been executed, and the results are displayed in a table. The results table has the following data:

	embarked	survived	cnt
1		1	2
2	C	0	75
3	C	1	93
4	Q	0	47
5	Q	1	30
6	S	0	427
7	S	1	217

2 개의 데이터는 탑승항구가 명시되어 있지 않고, 탑승항구 S의 생존율이 낮은 것을 알 수 있습니다. 따라서, 탑승항구 S의 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

The screenshot shows the AWS Athena Query Editor interface. On the left, the 'Database' section is set to 'analytics_hol' and lists a table named 'titanic_train'. The main area contains a SQL query: 'select embarked, pclass, count(1) as cnt from analytics_hol.titanic_train group by embarked, pclass order by embarked, pclass'. Below the query, the 'Run query' button is visible, along with performance metrics: '(Run time: 3.46 seconds, Data scanned: 52.32 KB)'. The 'Results' section displays a table with the following data:

	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.

The screenshot shows the AWS Management Console interface for Amazon RDS. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and user information for 'hyun @ aws-korea' in the 'Seoul' region. The left sidebar shows the 'Amazon RDS' navigation menu with options like Dashboard, Databases, Performance Insights, Snapshots, Automated backups, Reserved instances, Subnet groups, Parameter groups, Option groups, Events, Event subscriptions, and Recommendations (1). The main content area features a prominent 'Amazon Aurora' card with a blue information icon and a red 'Create database' button. Below this card is a 'Resources' section with a 'Refresh' button, displaying usage statistics for various RDS resources in the Asia Pacific (Seoul) region. To the right is an 'Additional information' section with links to documentation and guides.

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 [X]

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 [X]

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 across all DB clusters owned by your AWS account in the current AWS Region.

The DB cluster identifier is case-insensitive, but is stored as all lowercase (as in "mydbcluster"). Constraints: 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.

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](#)

Constraints: 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)

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

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](#)

Connectivity

Virtual Private Cloud (VPC) [Info](#)
VPC that defines the virtual networking environment for this DB cluster.

Only VPCs with a corresponding DB subnet group are listed.

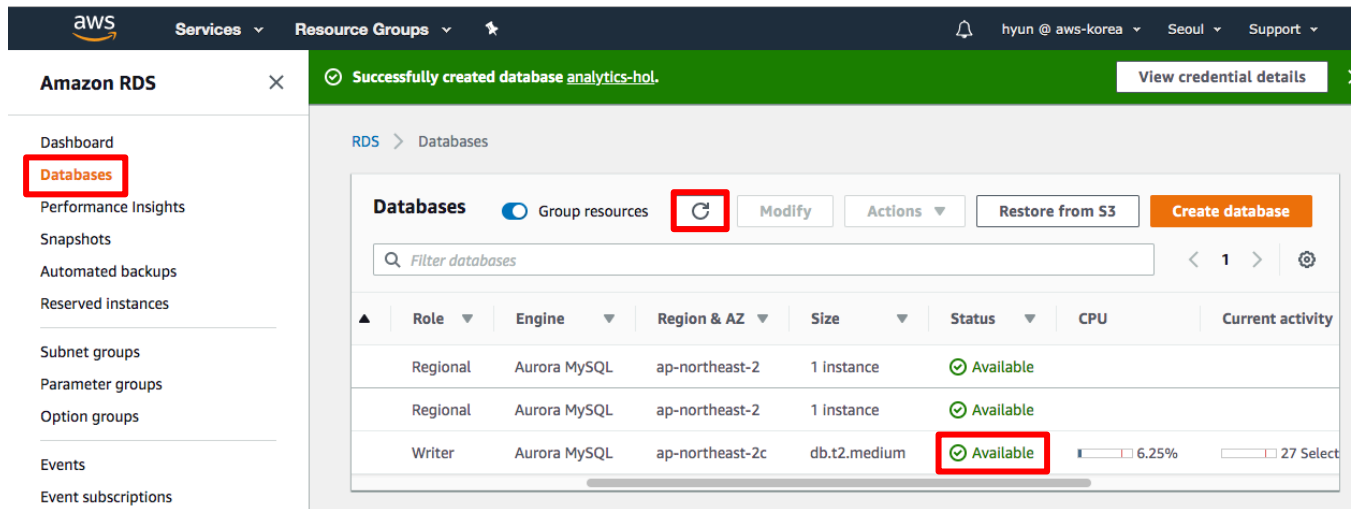
After a database is created, you can't change the VPC selection.

► **Additional connectivity configuration**

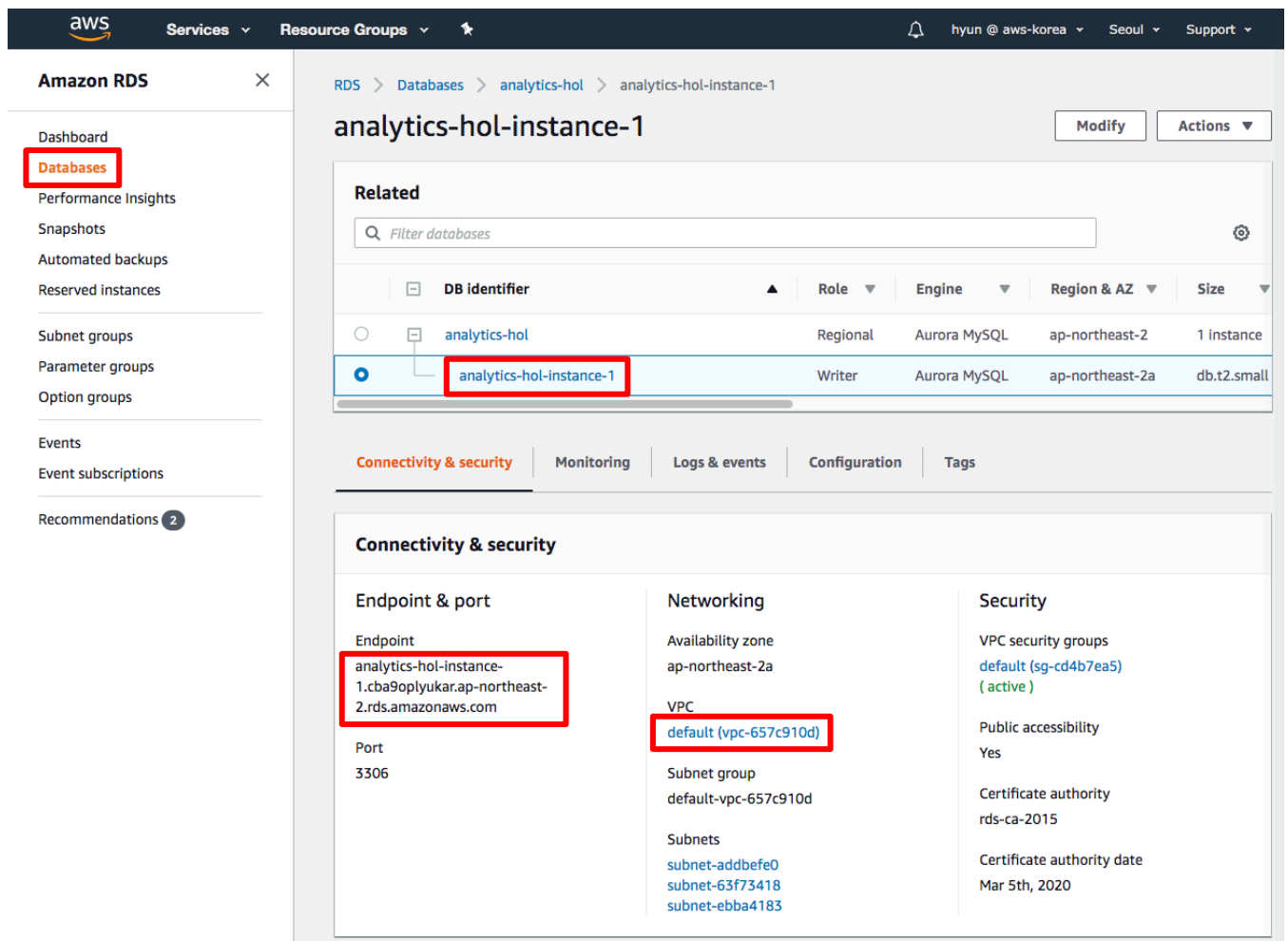
► **Additional configuration**
Database options, encryption enabled, failover, backup enabled, backtrack disabled, Enhanced Monitoring enabled, maintenance, CloudWatch Logs, delete protection disabled

Cancel

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

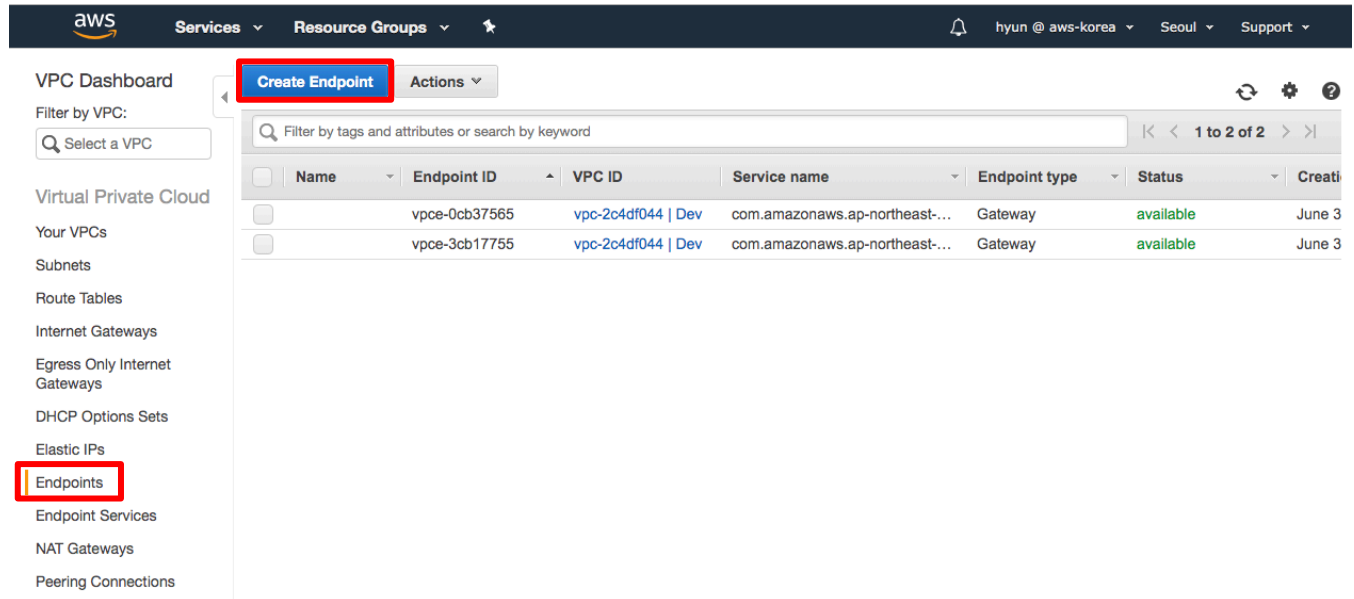


5-4. 생성된 Database 의 Writer 의 DB identifier 를 클릭해서 관련정보를 확인합니다. Endpoint 와 VPC 는 이후 실습에서 사용하니 copy 해서 메모하도록 합니다.



S3 Endpoint 생성

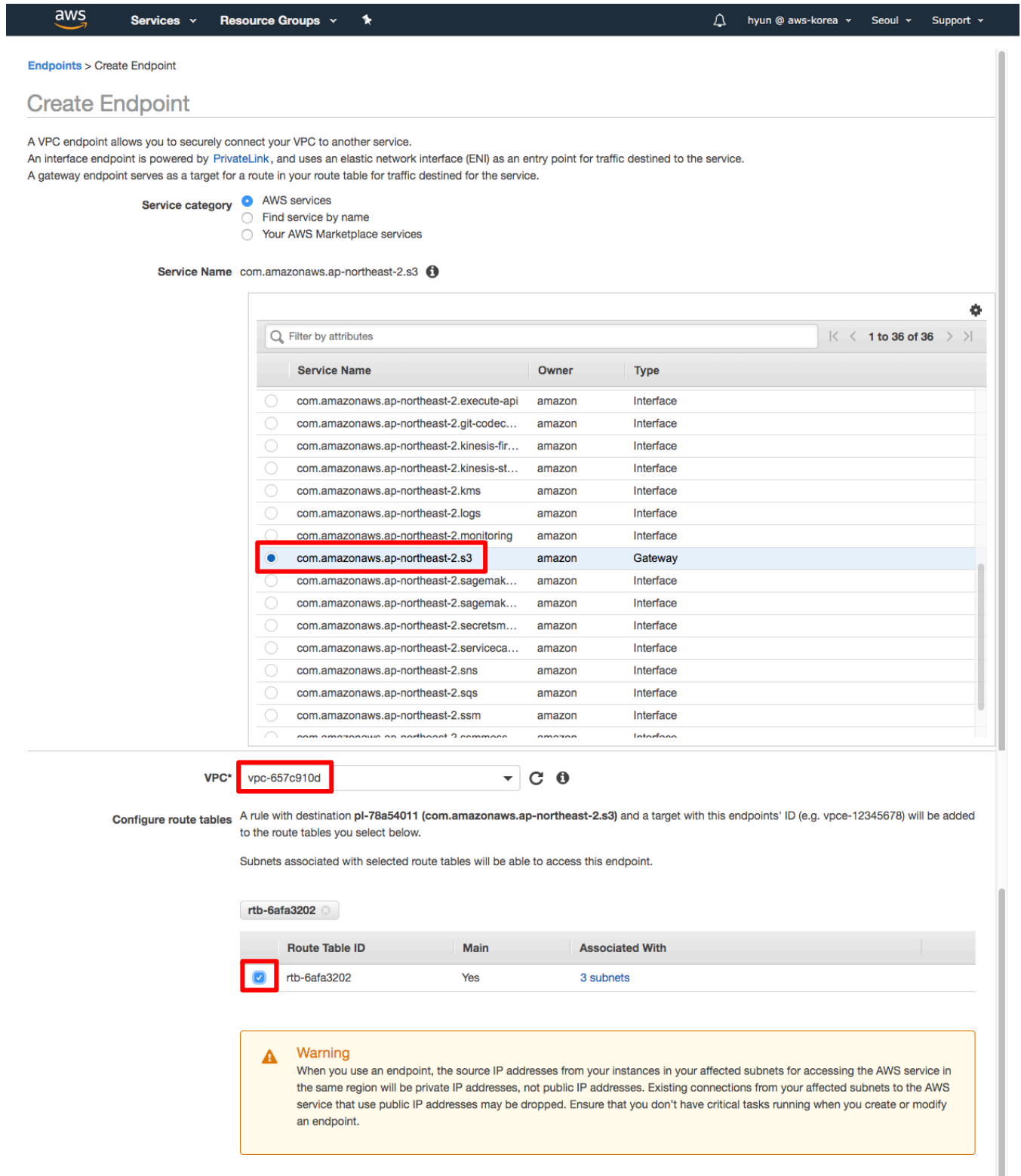
5-5. 생성된 Database 는 VPC 내에 생성되기 때문에 해당 Database 에 접근하기 위해 Glue Job 의 Cluster 는 VPC 내부에서 생성되게 됩니다. Glue Job 이 VPC 내부에서 S3 에 Private 하게 접근하기 위해 지금부터는 S3 Endpoint 를 생성하도록 합니다. VPC 콘솔로 이동합니다. Endpoints 로 이동한 후 Create Endpoint 버튼을 클릭합니다. (RDS 가 생성된 VPC 에 S3 Endpoint 가 이미 생성되어 있는 경우 Skip 하도록 합니다.)



The screenshot shows the AWS VPC console interface. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and user information. The left sidebar shows the 'VPC Dashboard' with various VPC resources listed, and 'Endpoints' is highlighted in red. The main content area shows the 'Create Endpoint' button, also highlighted in red, and a table of existing endpoints.

Name	Endpoint ID	VPC ID	Service name	Endpoint type	Status	Created
	vpce-0cb37565	vpc-2c4df044 Dev	com.amazonaws.ap-northeast-...	Gateway	available	June 3
	vpce-3cb17755	vpc-2c4df044 Dev	com.amazonaws.ap-northeast-...	Gateway	available	June 3

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.)



Endpoints > Create Endpoint

Create Endpoint

A VPC endpoint allows you to securely connect your VPC to another service.
 An interface endpoint is powered by [PrivateLink](#), and uses an elastic network interface (ENI) as an entry point for traffic destined to the service.
 A gateway endpoint serves as a target for a route in your route table for traffic destined to the service.

Service category

- AWS services
- Find service by name
- Your AWS Marketplace services

Service Name com.amazonaws.ap-northeast-2.s3 ⓘ

Service Name	Owner	Type
<input type="radio"/> com.amazonaws.ap-northeast-2.execute-api	amazon	Interface
<input type="radio"/> com.amazonaws.ap-northeast-2.git-codec...	amazon	Interface
<input type="radio"/> com.amazonaws.ap-northeast-2.kinesis-fir...	amazon	Interface
<input type="radio"/> com.amazonaws.ap-northeast-2.kinesis-st...	amazon	Interface
<input type="radio"/> com.amazonaws.ap-northeast-2.kms	amazon	Interface
<input type="radio"/> com.amazonaws.ap-northeast-2.logs	amazon	Interface
<input type="radio"/> com.amazonaws.ap-northeast-2.monitoring	amazon	Interface
<input checked="" type="radio"/> com.amazonaws.ap-northeast-2.s3	amazon	Gateway
<input type="radio"/> com.amazonaws.ap-northeast-2.sagemak...	amazon	Interface
<input type="radio"/> com.amazonaws.ap-northeast-2.sagemak...	amazon	Interface
<input type="radio"/> com.amazonaws.ap-northeast-2.secretsm...	amazon	Interface
<input type="radio"/> com.amazonaws.ap-northeast-2.serviceca...	amazon	Interface
<input type="radio"/> com.amazonaws.ap-northeast-2.sns	amazon	Interface
<input type="radio"/> com.amazonaws.ap-northeast-2.sqs	amazon	Interface
<input type="radio"/> com.amazonaws.ap-northeast-2.ssm	amazon	Interface
<input type="radio"/> com.amazonaws.ap-northeast-2.ssm	amazon	Interface

VPC* vpc-657c910d ⓘ

Configure route tables A rule with destination `pl-78a54011 (com.amazonaws.ap-northeast-2.s3)` and a target with this endpoints' ID (e.g. `vpce-12345678`) will be added to the route tables you select below.

Subnets associated with selected route tables will be able to access this endpoint.

rtb-6afa3202 ⓘ

Route Table ID	Main	Associated With
<input checked="" type="checkbox"/> rtb-6afa3202	Yes	3 subnets

Warning

When you use an endpoint, the source IP addresses from your instances in your affected subnets for accessing the AWS service in the same region will be private IP addresses, not public IP addresses. Existing connections from your affected subnets to the AWS service that use public IP addresses may be dropped. Ensure that you don't have critical tasks running when you create or modify an endpoint.

Policy* Full Access - Allow access by any user or service within the VPC using credentials from any AWS accounts to any resources in this AWS service. All policies — IAM user policies, VPC endpoint policies, and AWS service-specific policies (e.g. Amazon S3 bucket policies, any S3 ACL policies) — must grant the necessary permissions for access to succeed. ?

Custom

Use the [policy creation tool](#) to generate a policy, then paste the generated policy below.

```
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Resource": "*",
      "Principal": "*"
    }
  ]
}
```

* Required Cancel Create endpoint

aws Services ▾ Resource Groups ▾ hyun @ aws-korea ▾ Seoul ▾ Support ▾

[Endpoints](#) > Create Endpoint

Create Endpoint

✔ The following VPC Endpoint was created:

VPC Endpoint ID [vpce-0c81eaea8d90c8360](#)

Close

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

aws Services ▾ Resource Groups ▾ hyun @ aws-korea ▾ Seoul ▾ Support ▾

VPC Dashboard

Filter by 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

<input type="checkbox"/>	Name	Endpoint ID	VPC ID	Service name	Endpoint type	Status
<input checked="" type="checkbox"/>		vpce-0c81eaea8d...	vpce-657c910d	com.amazonaws.ap-northeast-...	Gateway	available
<input type="checkbox"/>		vpce-0cb37565	vpce-2c4df044 Dev	com.amazonaws.ap-northeast-...	Gateway	available
<input type="checkbox"/>		vpce-3cb17755	vpce-2c4df044 Dev	com.amazonaws.ap-northeast-...	Gateway	available

Endpoint: vpce-0c81eaea8d90c8360 ⌵ ⌵ ⌵

Details Route Tables Policy Tags

Endpoint ID	vpce-0c81eaea8d90c8360	VPC ID	vpce-657c910d
Status	available	Creation Time	May 29, 2019 at 10:52:17 AM UTC+9
Service name	com.amazonaws.ap-northeast-2.s3	Endpoint type	Gateway
DNS Names			

aws Services Resource Groups hyun @ aws-korea Seoul Support

VPC Dashboard Create Endpoint Actions

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

NAT Gateways

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
<input checked="" type="checkbox"/>		vpce-0c81eaea8d...	vpce-657c910d	com.amazonaws.ap-northeast-...	Gateway	available
<input type="checkbox"/>		vpce-0cb37565	vpce-2c4df044 Dev	com.amazonaws.ap-northeast-...	Gateway	available
<input type="checkbox"/>		vpce-3cb17755	vpce-2c4df044 Dev	com.amazonaws.ap-northeast-...	Gateway	available

Endpoint: vpce-0c81eaea8d90c8360

Details **Route Tables** Policy Tags

Manage Route Tables

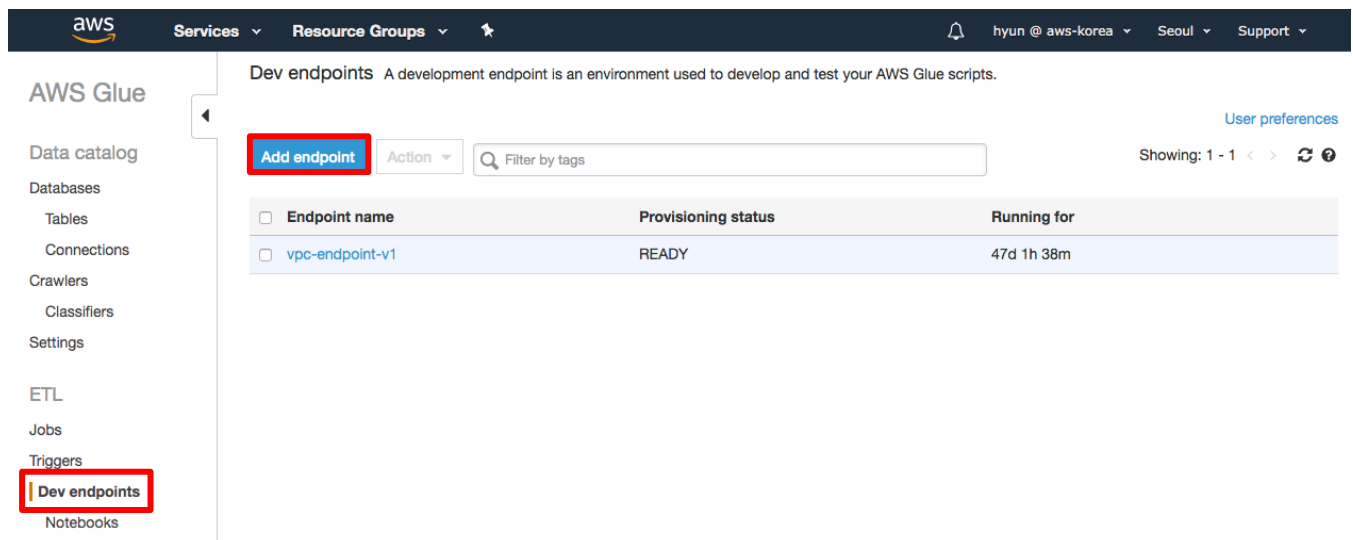
Route Table ID	Main	Associated With
rtb-6afa3202	Yes	3 subnets

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.



The screenshot shows the AWS Glue console interface. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and user information. The left sidebar lists various AWS Glue services, with 'Dev endpoints' highlighted in a red box. The main content area displays the 'Dev endpoints' page, featuring an 'Add endpoint' button (highlighted in a red box), an 'Action' dropdown, and a search bar. Below this is a table with the following data:

<input type="checkbox"/>	Endpoint name	Provisioning status	Running for
<input type="checkbox"/>	vpc-endpoint-v1	READY	47d 1h 38m

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.

Add development endpoint

Set up your development endpoint

Use a development endpoint to set up an AWS Glue environment that you can use to develop your scripts with other software, such as notebooks, integrated development environments (IDEs), and a REPL shell.

Development endpoint name
analytics-hol

IAM role
AWSGlueServiceRole-Hol

Ensure that this role has permission to your Amazon S3 sources, targets, and any libraries used by the development endpoint. [Create IAM role.](#)

▼ Security configuration, script libraries, and job parameters (optional)

Security configuration
None

Data processing units (DPUs)
2

Python library path
s3://bucket-name/folder-name/file-name

Dependent jars path
s3://bucket-name/folder-name/file-name

▶ Tags (optional)

▶ Catalog options (optional)

Next

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.

Add development endpoint

Networking

How do you want to provide the network information used to create the development endpoint?

When required, this information determines which VPC, subnet, and security groups are shared with your data stores and the development endpoint. The development endpoint uses this networking information to connect to your data stores in an AWS Glue environment.

Skip networking information
Networking information is optional if you only connect to S3 data stores.

Choose a connection
AWS Glue obtains the VPC, subnet, and security groups from the connection.

Choose a VPC, subnet, and security groups
AWS Glue presents you with VPCs, subnets, and security groups that can be used to create the development endpoint.

VPC
Choose a VPC with access to an S3 endpoint.
vpc-657c910d

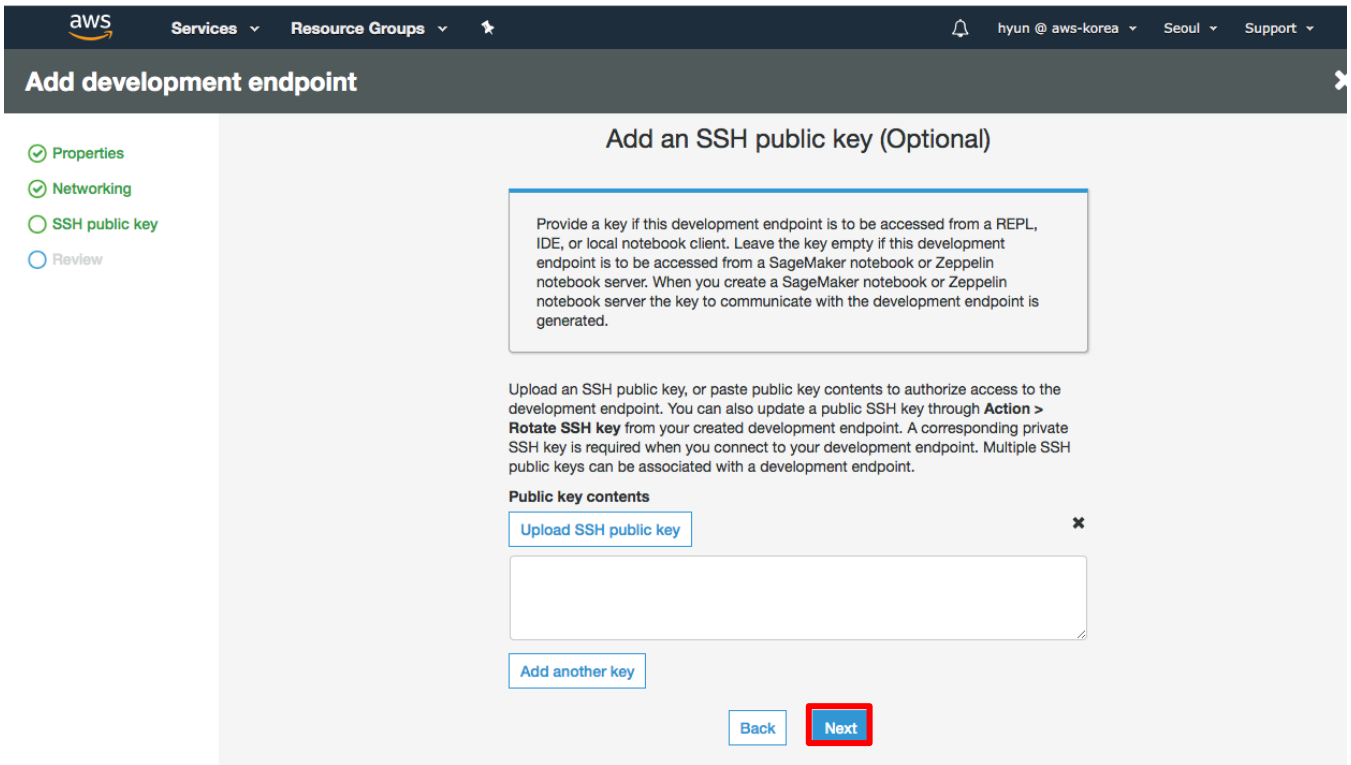
Subnet (Optional)
Choose the subnet within your VPC.
subnet-addbefe0

Security Groups (Optional)
Choose up to 4 security groups with a self-referencing rule.

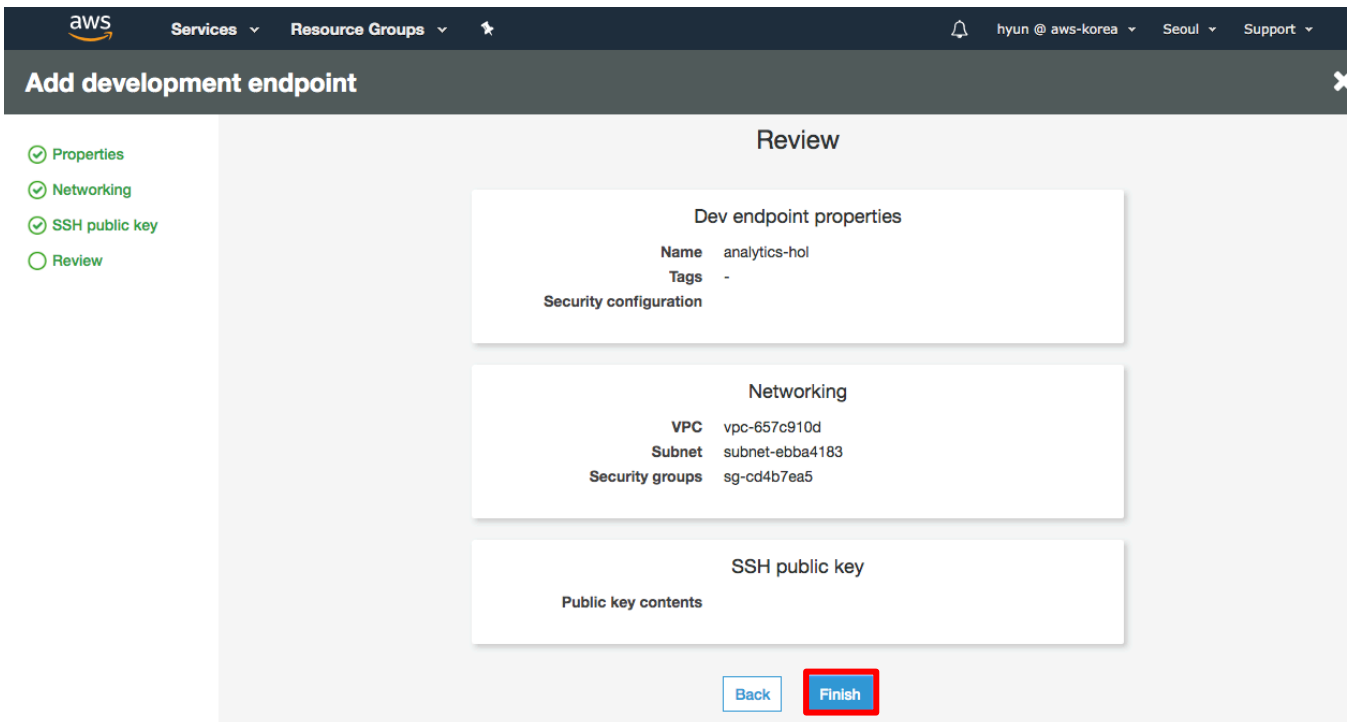
<input checked="" type="checkbox"/> Group ID	Group name
<input checked="" type="checkbox"/> sg-cd4b7ea5	default

Back Next

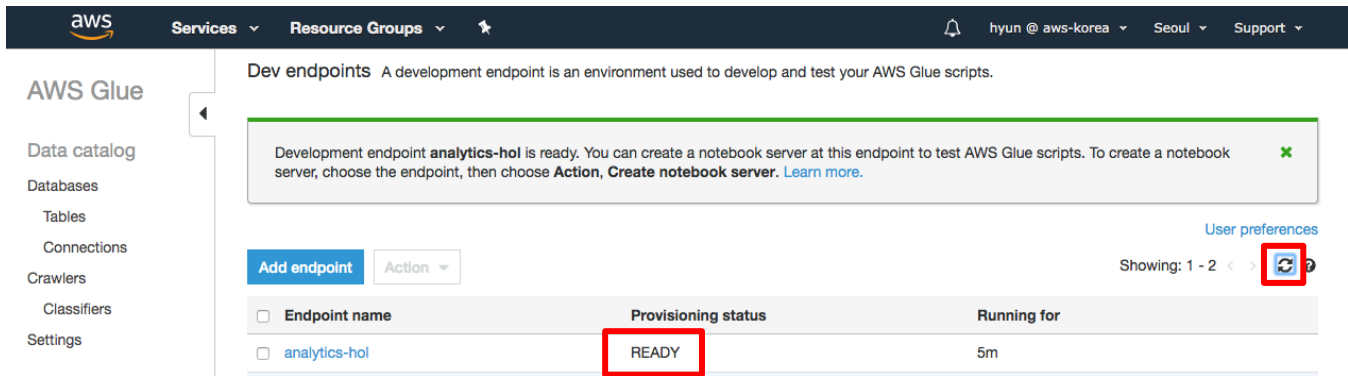
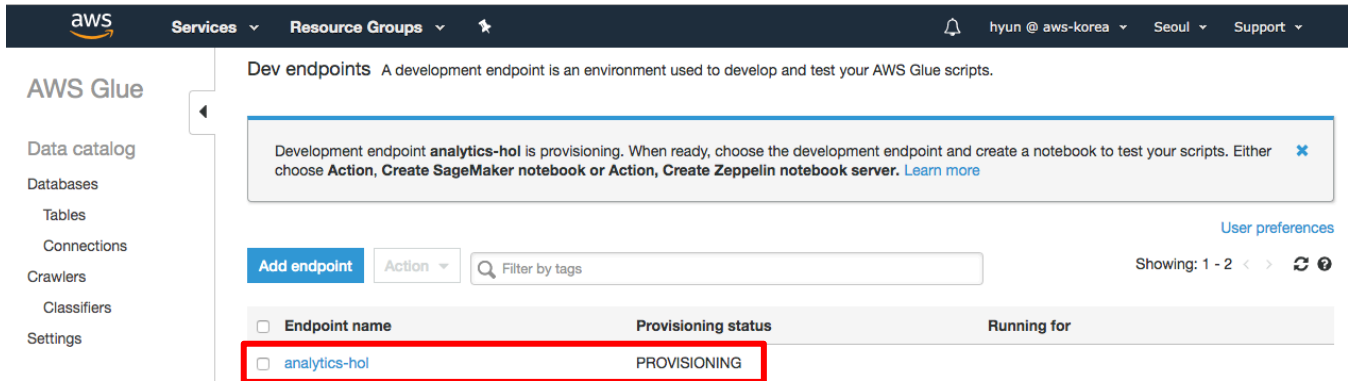
6-4. SSH Public key(Optional) 단계에서는 기본 설정으로 Next 버튼을 클릭합니다.



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

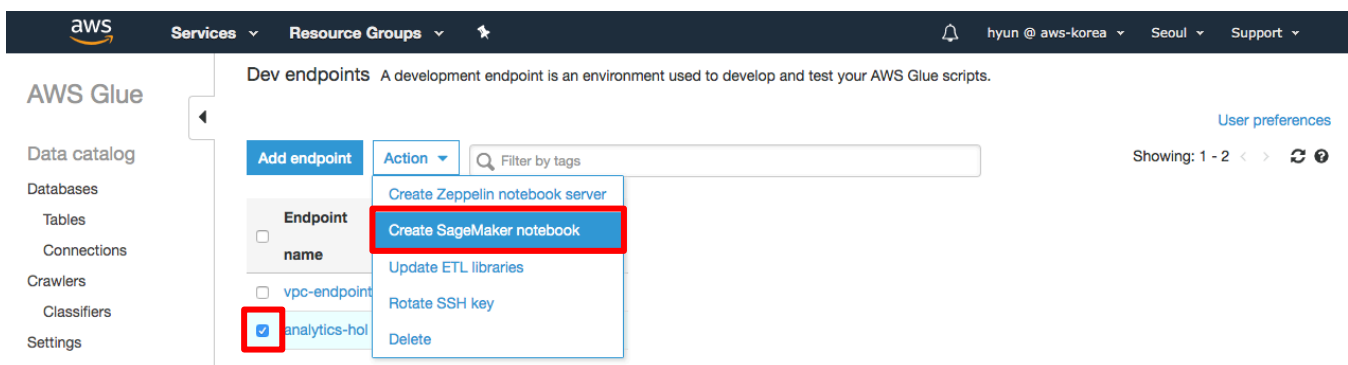


6-6. Endpoint 생성이 완료되면 PROVISIONING 상태에서 READY 상태로 바뀝니다. 오른쪽 상단의 리프레시 버튼을 클릭하여 Provisioning status 가 READY 상태로 변경된 것을 확인합니다.



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)

NOTEBOOK NAME
aws-glue- analytics-hol
Name may contain letters (A-Z), numbers(0-9), hyphens(-), or underscores(_).

ATTACH TO DEVELOPMENT ENDPOINT
analytics-hol
Notebook instances require permissions to call other services including SageMaker and S3.
 Update a policy in an IAM role
 Choose an existing IAM role
 Create an IAM role

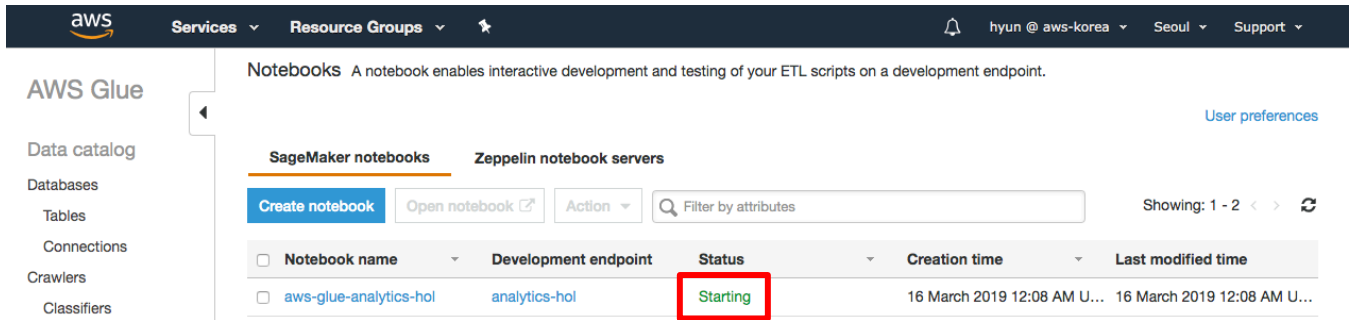
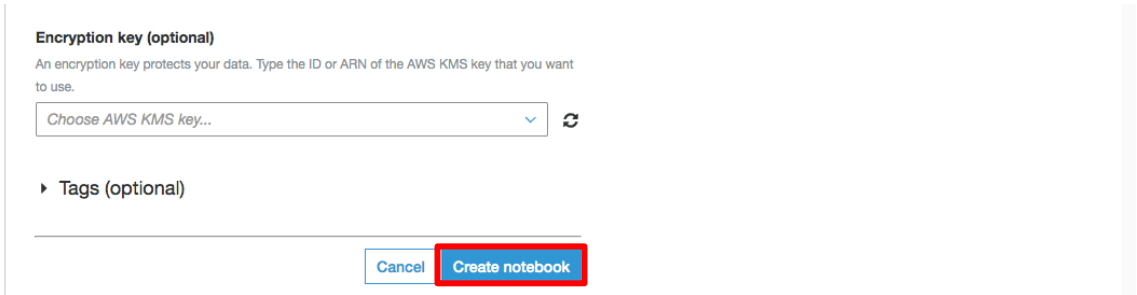
IAM ROLE
AWSGlueServiceSageMakerNotebookRole- Hol
To create an IAM role, you must have **CreateRole**, **CreatePolicy**, and **AttachRolePolicy** permissions.
You can also create an IAM role on the [IAM console](#).

VPC (optional)
Choose the VPC name that contains your data store.
vpc-657c910d

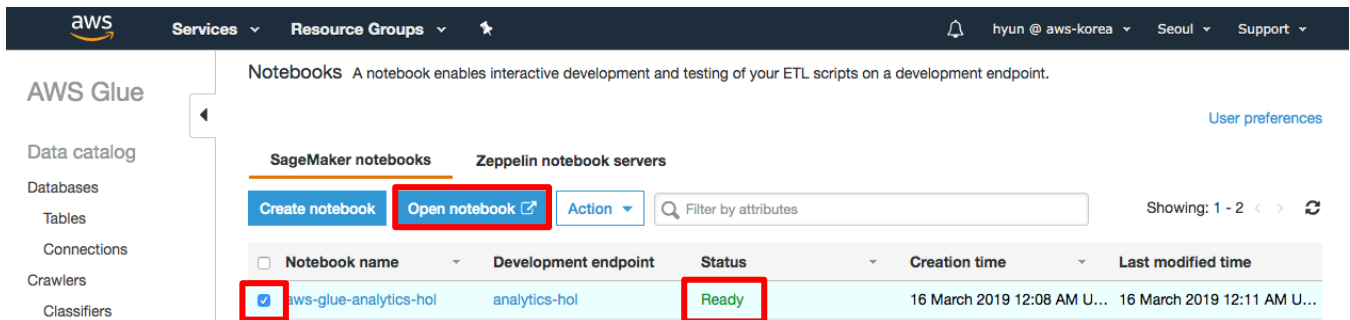
SUBNET
Choose the subnet within your VPC.
subnet-ebba4183

SECURITY GROUPS
Choose one or more security groups that allow access to the data store in your VPC. AWS Glue associates these security groups to the ENI attached to your subnet. To allow AWS Glue components to communicate and also prevent access from other networks, at least one chosen security group must specify a self-referencing inbound rule for all TCP ports.

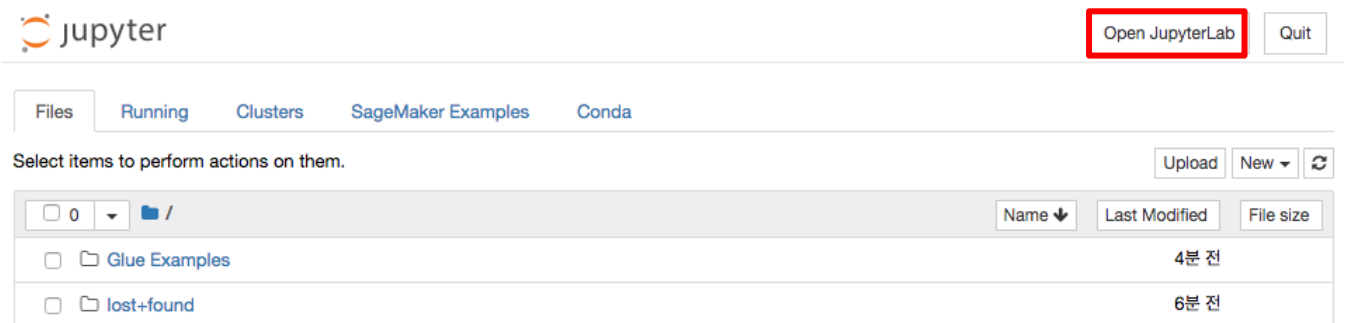
Group ID	Group name
<input checked="" type="checkbox"/> sg-cd4b7ea5	default
<input type="checkbox"/> sg-04ab7229945a99cdf	Dev
<input type="checkbox"/> sg-079da6c9b53206741	ElasticMapReduce-master
<input type="checkbox"/> sg-0a986baefea1601b9	ElasticMapReduceEditors-Editor
<input type="checkbox"/> sg-0a9c44a47fc70992	deep learning

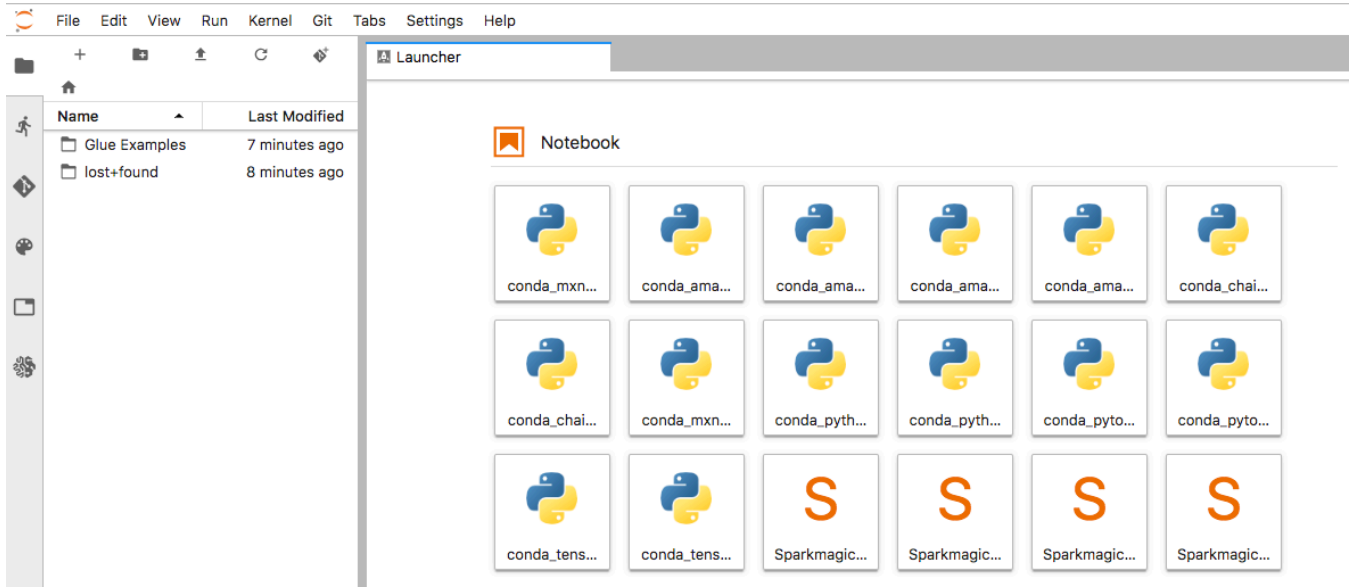


6-9. 생성한 aws-glue-analytics-hol SageMaker notebook 의 Status 가 Ready 로 변경되면 해당 notebook 을 선택하고 Open notebook 버튼을 클릭하여 SageMaker notebook 을 실행합니다.



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



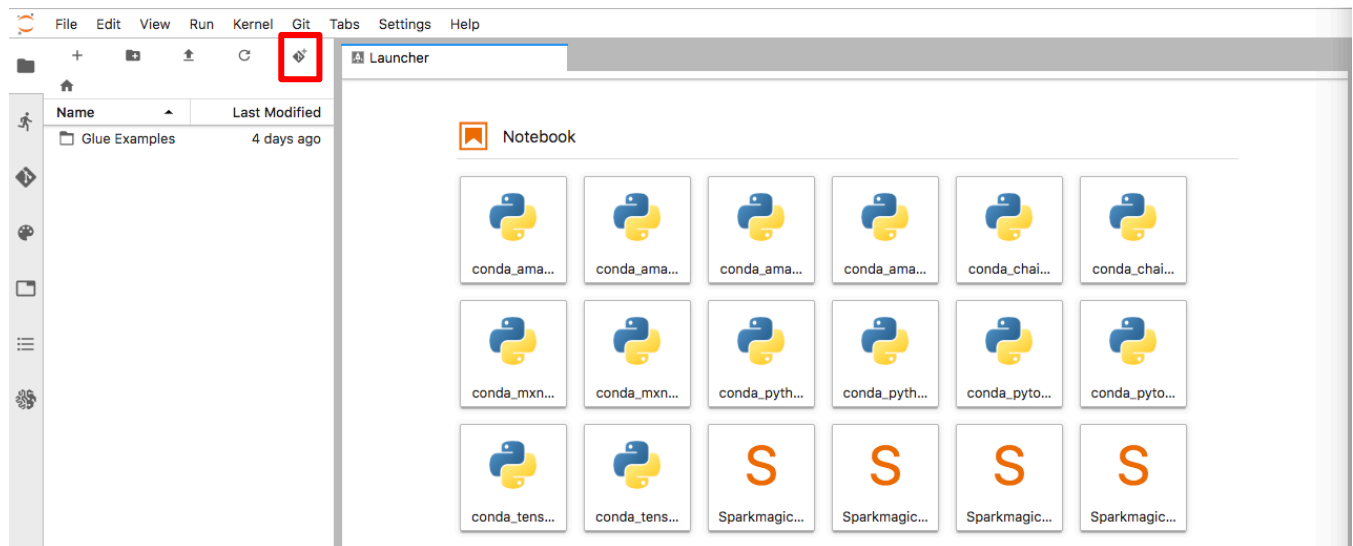


7. Lab: Data Analysis, ETL and ML

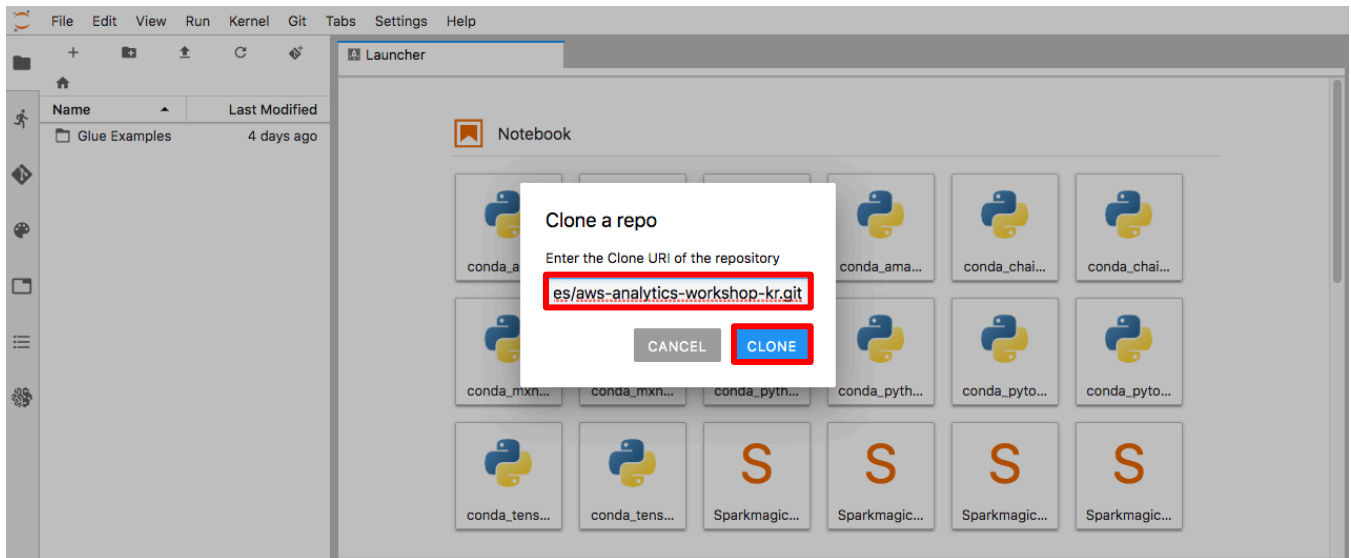
이번 장에서는 JupyterLab 에서 데이터를 분석하고 ETL 을 진행하며 다양한 Machine Learning Model 을 생성하고 성능을 테스트하는 작업을 진행하겠습니다.

Git Clone

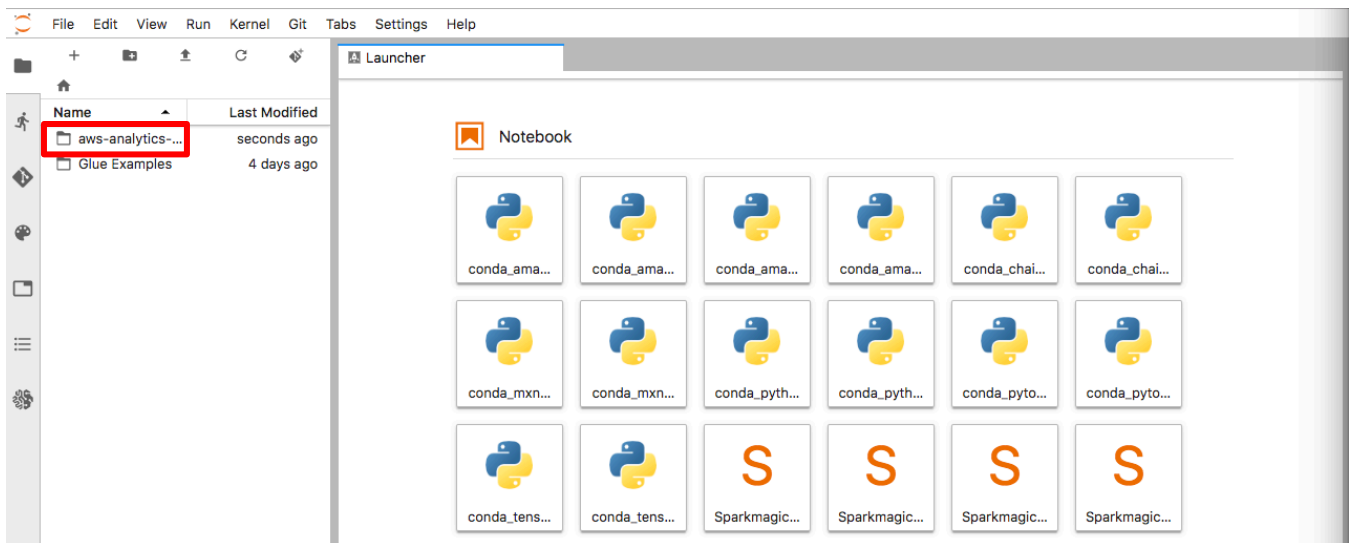
7-1. JupyterLab 에서 Git Clone 버튼을 클릭합니다.



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



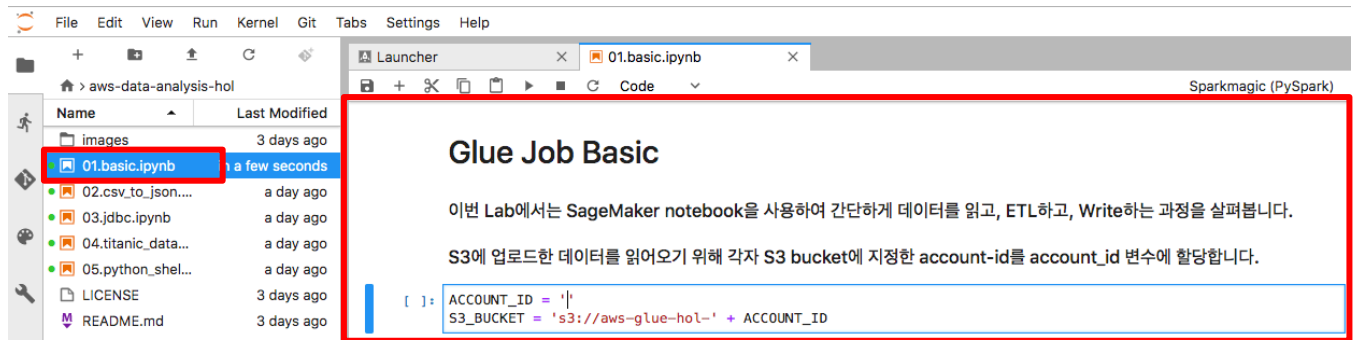
7-3. aws-analytics-workshop-kr 폴더가 생성된 것을 확인할 수 있습니다.



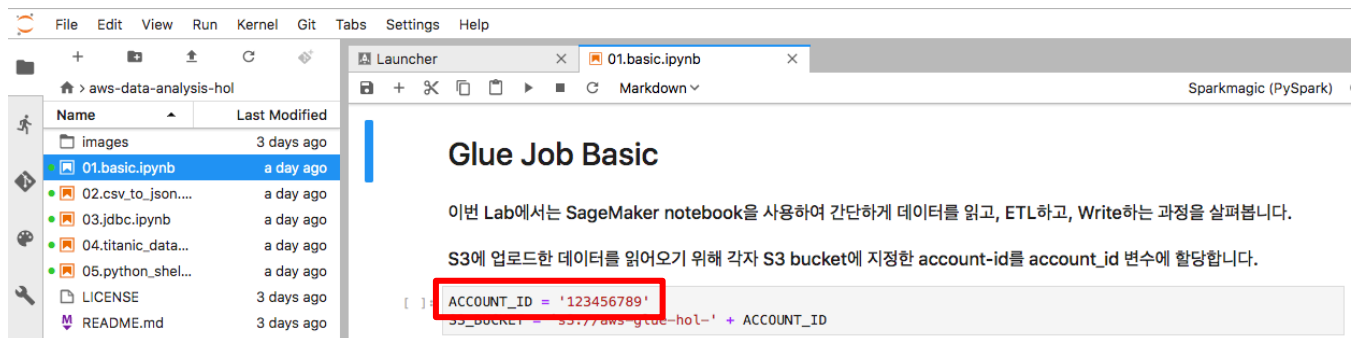
Lab1: Glue Job Basic

Lab1 에서는 Glue Job 을 위한 기본 Template 을 살펴봅니다.

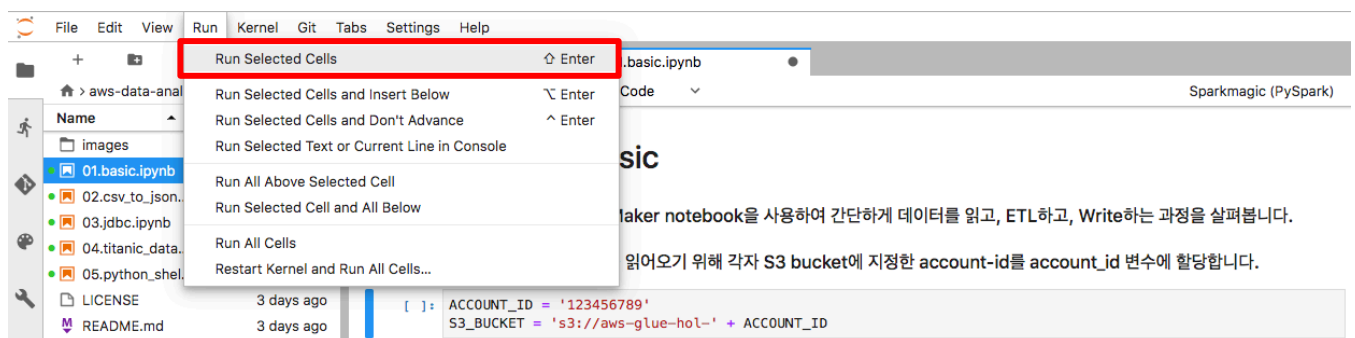
7-4. 생성된 aws-analytics-workshop-kr/glue-deep-dive-hol 폴더를 클릭하여 이동 후 01.basic_job_template.ipynb 노트북 파일을 클릭하여 엽니다.



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



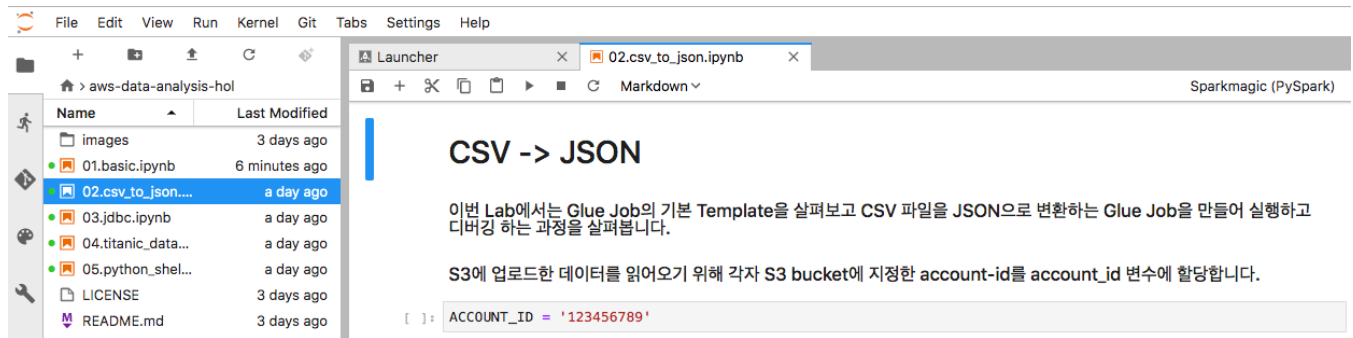
7-6. 노트북의 설명과 코드의 주석을 살펴가며 코드를 하나씩 실행하고 결과를 확인합니다. 코드 실행은 실행할 코드 셀을 선택한 후 키보드의 Shift + Enter 로 할 수도 있고, 메뉴에서 Run->Run Selected Cells 를 선택해서 할 수도 있습니다.



Lab2: Create Job, Run and Debugging

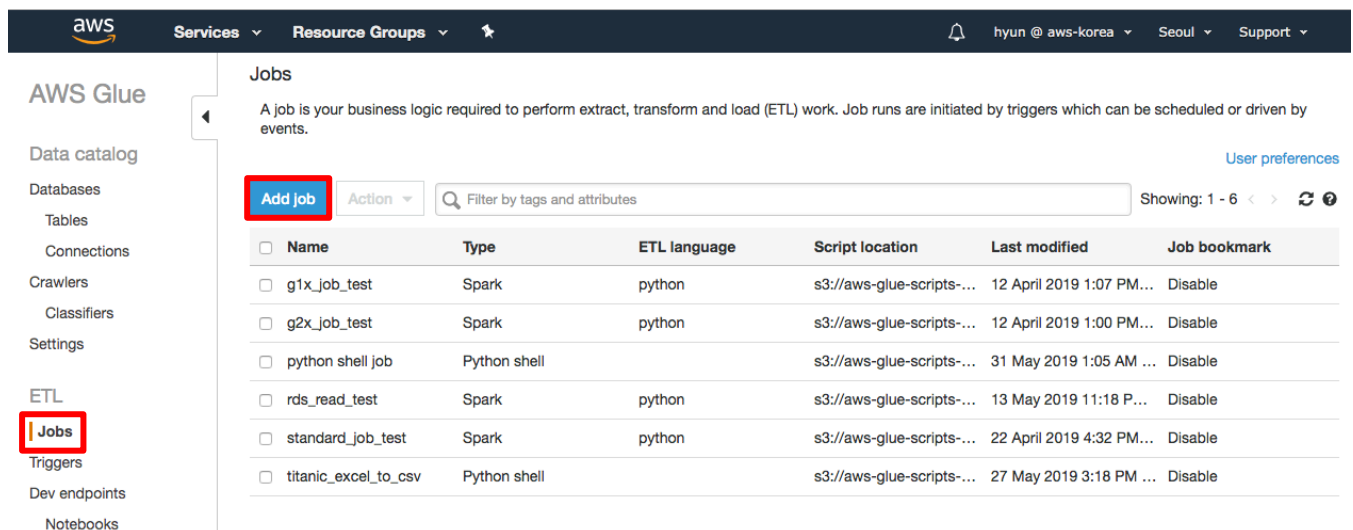
Lab2 에서는 csv 포맷 파일을 json 포맷으로 변경하는 Glue Job 을 생성한 후 실행 및 디버깅하는 과정을 진행해보도록 하겠습니다.

7-7. 02.csv_to_json.ipynb 노트북 파일을 클릭하여 열고 첫 번째 코드 Cell 의 ACCOUNT_ID 값을 본인의 account_id 로 대체합니다.

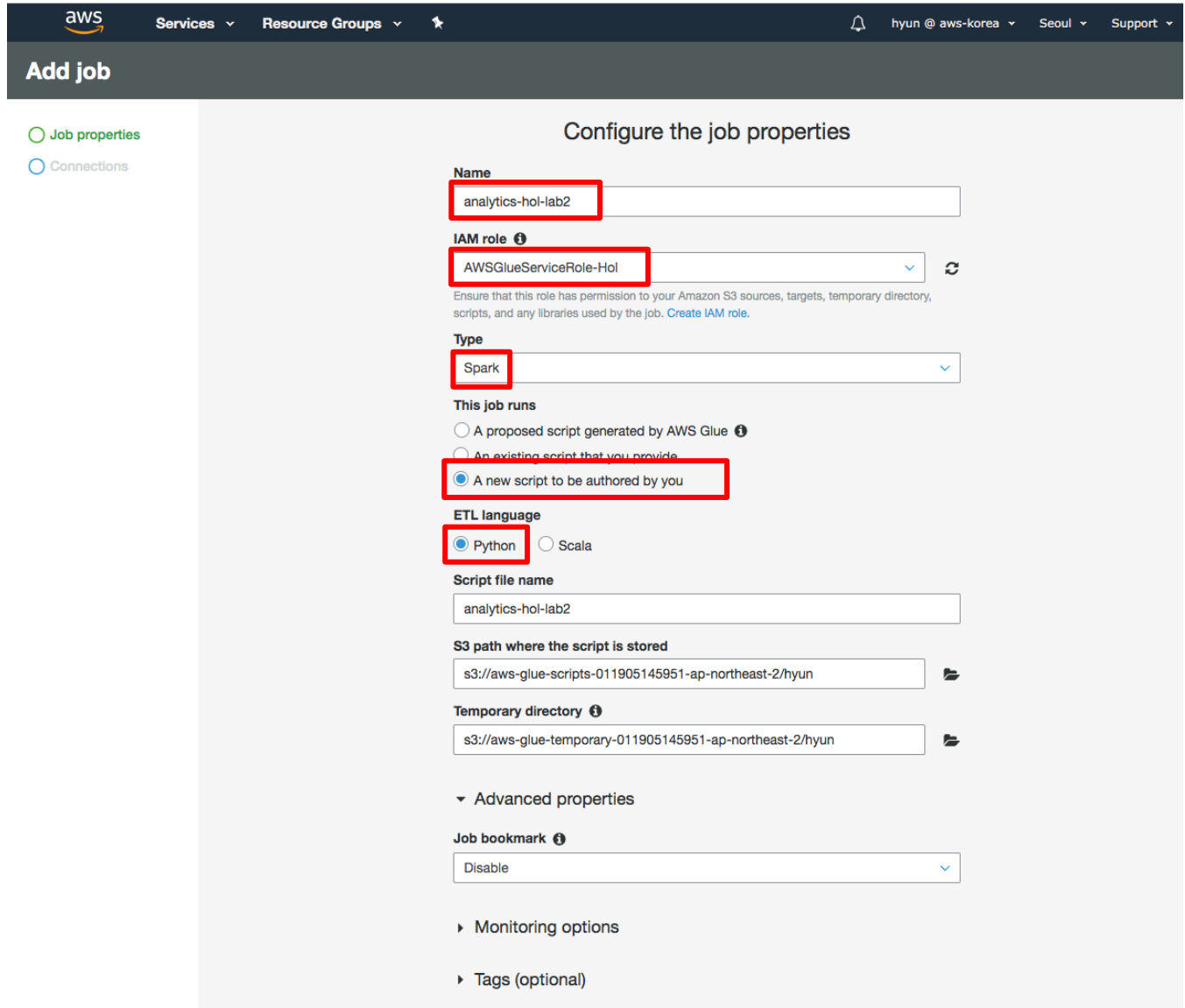


7-8. 노트북 가이드에 따라 "File Read & Write with Spark API"와 "File Read & Write with Glue API" Lab 을 진행합니다. Lab 의 마지막 코드 Cell 은 Glue Job 에서만 실행되고 Notebook 에서는 실행되지 않으니 Notebook 에서 실행하지 않습니다.

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.)



aws Services ▾ Resource Groups ▾ hyun @ aws-korea ▾ Seoul ▾ Support ▾

Add job

- Job properties
- Connections

Configure the job properties

Name
analytics-hol-lab2

IAM role ⓘ
AWSGlueServiceRole-Hol

Ensure that this role has permission to your Amazon S3 sources, targets, temporary directory, scripts, and any libraries used by the job. [Create IAM role.](#)

Type
Spark

This job runs

- A proposed script generated by AWS Glue ⓘ
- An existing script that you provide
- A new script to be authored by you

ETL language

- Python
- Scala

Script file name
analytics-hol-lab2

S3 path where the script is stored
s3://aws-glue-scripts-011905145951-ap-northeast-2/hyun

Temporary directory ⓘ
s3://aws-glue-temporary-011905145951-ap-northeast-2/hyun

▾ Advanced properties

Job bookmark ⓘ
Disable

- Monitoring options
- Tags (optional)

▼ 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	Value
Type key...	Type value...

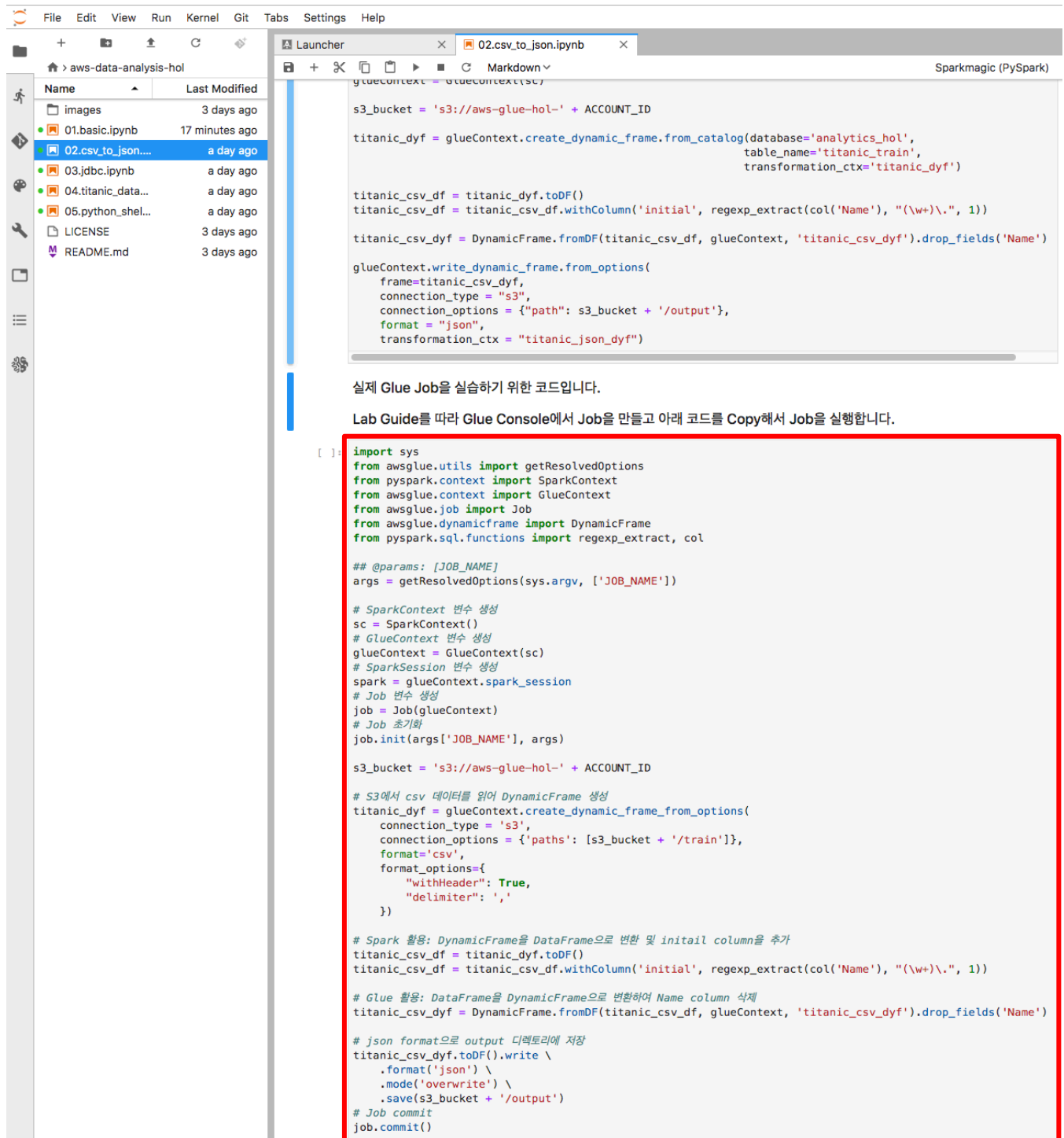
▶ Catalog options (optional)

Next

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

The screenshot shows the AWS console interface for adding a job. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and user information. The main header is 'Add job'. On the left, there are two tabs: 'Job properties' (checked) and 'Connections'. Under 'Job properties', 'analytics-hol-lab2' is selected. The 'Connections' section is active, displaying a list of available connections and a list of required connections. The 'All connections' list includes 'analytics_hol' and 'dev', each with a 'Select' button. The 'Required connections' list is empty, showing 'No items selected'. At the bottom, there are two buttons: 'Back' and 'Save job and edit script', with the latter highlighted by a red rectangular box.

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.



The screenshot shows a Jupyter Notebook interface with a file explorer on the left and a code editor on the right. The code in the notebook is as follows:

```

import sys
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.dynamicframe import DynamicFrame
from pyspark.sql.functions import regexp_extract, col

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

# SparkContext 변수 생성
sc = SparkContext()
# GlueContext 변수 생성
glueContext = GlueContext(sc)
# SparkSession 변수 생성
spark = glueContext.spark_session
# Job 변수 생성
job = Job(glueContext)
# Job 초기화
job.init(args['JOB_NAME'], args)

s3_bucket = 's3://aws-glue-hol-' + ACCOUNT_ID

# S3에서 csv 데이터를 읽어 DynamicFrame 생성
titanic_dyf = glueContext.create_dynamic_frame_from_options(
    connection_type = 's3',
    connection_options = {'paths': [s3_bucket + '/train']},
    format='csv',
    format_options={
        "withHeader": True,
        "delimiter": ','
    })

# Spark 활용: DynamicFrame을 DataFrame으로 변환 및 initial column을 추가
titanic_csv_df = titanic_dyf.toDF()
titanic_csv_df = titanic_csv_df.withColumn('initial', regexp_extract(col('Name'), "(\\w+)\\.\\.", 1))

# Glue 활용: DataFrame을 DynamicFrame으로 변환하여 Name column 삭제
titanic_csv_dyf = DynamicFrame.fromDF(titanic_csv_df, glueContext, 'titanic_csv_dyf').drop_fields('Name')

# json format으로 output 디렉토리에 저장
titanic_csv_dyf.toDF().write \
    .format('json') \
    .mode('overwrite') \
    .save(s3_bucket + '/output')
# Job commit
job.commit()

```

실제 Glue Job을 실행하기 위한 코드입니다.

Lab Guide를 따라 Glue Console에서 Job을 만들고 아래 코드를 Copy해서 Job을 실행합니다.

The diagram cannot be generated. Check the annotations in your script.

```

1 import sys
2 from awsglue.utils import getResolvedOptions
3 from pyspark.context import SparkContext
4 from awsglue.context import GlueContext
5 from awsglue.job import Job
6 from awsglue.dynamicframe import DynamicFrame
7 from pyspark.sql.functions import regexp_extract, col
8
9 ## @params: [JOB_NAME]
10 args = getResolvedOptions(sys.argv, ['JOB_NAME'])
11
12 # SparkContext 변수 생성
13 sc = SparkContext()
14 # GlueContext 변수 생성
15 glueContext = GlueContext(sc)
16 # SparkSession 변수 생성
17 spark = glueContext.spark_session
18 # Job 변수 생성
19 job = Job(glueContext)
20 # Job 초기화
21 job.init(args['JOB_NAME'], args)
22
23 s3_bucket = 's3://aws-glue-hol-' + ACCOUNT_ID
24
25 # S3에서 csv 데이터를 읽어 DynamicFrame 생성
26 titanic_dyf = glueContext.create_dynamic_frame_from_options(
27     connection_type = 's3',
28     connection_options = {'paths': [s3_bucket + '/train']},
29     format='csv',
30     format_options={
31         "withHeader": True,
32         "delimiter": ','
33     })
34
35 # Spark 활용: DynamicFrame을 DataFrame으로 변환 및 initial column을 추가
36 titanic_csv_df = titanic_dyf.toDF()
37 titanic_csv_df = titanic_csv_df.withColumn('initial', regexp_extract(col('Name'), "(\\w+)\\.\\.", 1))
38
39 # Glue 활용: DataFrame을 DynamicFrame으로 변환하여 Name column 삭제
40 titanic_csv_dyf = DynamicFrame.fromDF(titanic_csv_df, glueContext, 'titanic_csv_dyf').drop_fields('Name')
41
42 # json format으로 output 디렉토리에 저장
43 titanic_csv_dyf.toDF().write \
44     .format('json') \
45     .mode('overwrite') \
46     .save(s3_bucket + '/output')
47 # Job commit
48 job.commit()
    
```

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

Parameters (optional)

Review and override parameter values, as needed, before running this job. Changes affect this run only. Edit a job to change default parameter values.

- ▶ Advanced properties
- ▶ Monitoring options
- ▶ Tags
- ▶ Security configuration, script libraries, and job parameters

Only job **analytics-hol-lab2** is run. Jobs dependent on the completion of job **analytics-hol-lab2** will not be run. To run a job and trigger dependent jobs, define an on-demand trigger.

Run job

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

The job run failed. For help, post the error message on the [AWS Glue forum](#) or contact [AWS support](#) with the job run ID: jr_9f07025d46b7e0b6cd7eb36dd0970a7eff4d05a683e41048d53db00edca89777

Job: analytics-hol-lab2

Insert template at cursor ⓘ

The diagram cannot be generated. Check the annotations in your script.

```

1 import sys
2 from awsglue.utils import getResolvedOptions
3 from pyspark.context import SparkContext
4 from awsglue.context import GlueContext
5 from awsglue.job import Job
6 from awsglue.dynamicframe import DynamicFrame
7 from pyspark.sql.functions import regexp_extract, col
8
    
```

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

The job run failed. For help, post the error message on the [AWS Glue forum](#) or contact [AWS support](#) with the job run ID: jr_9f07025d46b7e0b6cd7eb36dd0970a7eff4d05a683e41048d53db00edca89777

Job: analytics-hol-lab2

Insert template at cursor ⓘ

The diagram cannot be generated. Check the annotations in your script.

```

1 import sys
2 from awsglue.utils import getResolvedOptions
3 from pyspark.context import SparkContext
4 from awsglue.context import GlueContext
5 from awsglue.job import Job
6 from awsglue.dynamicframe import DynamicFrame
7 from pyspark.sql.functions import regexp_extract, col
8
    
```


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

The screenshot shows the AWS Glue console interface. On the left is a navigation menu with categories like Data catalog, Databases, Connections, Crawlers, Settings, ETL, Jobs, Triggers, Dev endpoints, Notebooks, Security, and Tutorials. The main content area is titled 'Jobs' and contains a list of jobs. The job 'analytics-hol-lab2' is selected, and its 'History' tab is active. Below the job list, a table shows the run history for this job. The first run, with ID 'jr_9f07025d...', is marked as 'Failed'. A tooltip is displayed over the error icon, showing the message: 'NameError: name 'ACCOUNT_ID' is not defined'.

Name	Type	ETL language	Script location	Last modified	Job bookmark
<input checked="" type="checkbox"/> analytics-hol-lab2	Spark	python	s3://aws-glue-scripts-...	1 June 2019 10:58 AM...	Disable
<input type="checkbox"/> g1x_job_test	Spark	python	s3://aws-glue-scripts-...	12 April 2019 1:07 PM...	Disable
<input type="checkbox"/> g2x_job_test	Spark	python	s3://aws-glue-scripts-...	12 April 2019 1:00 PM...	Disable
<input type="checkbox"/> python shell job	Python shell		s3://aws-glue-scripts-...	31 May 2019 1:05 AM ...	Disable
<input type="checkbox"/> rds_read_test	Spark	python	s3://aws-glue-scripts-...	13 May 2019 11:18 P...	Disable
<input type="checkbox"/> standard_job_test	Spark	python	s3://aws-glue-scripts-...	22 April 2019 4:32 PM...	Disable
<input type="checkbox"/> titanic_excel_to_csv	Python shell		s3://aws-glue-scripts-...	27 May 2019 3:18 PM ...	Disable

Run ID	Retry attempt	Run status	Error	Logs	Error logs	Maximum capacity	Executor time	Timeout	Delay	Triggered by	Start time	End time
<input type="radio"/> jr_9f07025d...	-	Failed	NameError: name 'ACCOUNT_ID' is not defined	Logs	Error logs	2	1 min	2880 mins			1 June...	1 June...

7-17. Error logs 를 클릭합니다.

The screenshot shows the AWS Glue Jobs console. On the left is a navigation menu with categories like Data catalog, ETL, Jobs, Triggers, Dev endpoints, Security, and Tutorials. The main area displays a list of jobs with columns for Name, Type, ETL language, Script location, Last modified, and Job bookmark. The job 'analytics-hoi-lab2' is selected. Below the list, there are tabs for History, Details, Script, and Metrics. The 'History' tab is active, showing a table of job runs. The first run, 'jr_9f07025d...', is in a 'Failed' state. The 'Error logs' link for this run is highlighted with a red box.

Name	Type	ETL language	Script location	Last modified	Job bookmark
<input checked="" type="checkbox"/> analytics-hoi-lab2	Spark	python	s3://aws-glue-scripts-...	1 June 2019 10:58 AM...	Disable
<input type="checkbox"/> g1x_job_test	Spark	python	s3://aws-glue-scripts-...	12 April 2019 1:07 PM...	Disable
<input type="checkbox"/> g2x_job_test	Spark	python	s3://aws-glue-scripts-...	12 April 2019 1:00 PM...	Disable
<input type="checkbox"/> python shell job	Python shell		s3://aws-glue-scripts-...	31 May 2019 1:05 AM ...	Disable
<input type="checkbox"/> rds_read_test	Spark	python	s3://aws-glue-scripts-...	13 May 2019 11:18 P...	Disable
<input type="checkbox"/> standard_job_test	Spark	python	s3://aws-glue-scripts-...	22 April 2019 4:32 PM...	Disable
<input type="checkbox"/> titanic_excel_to_csv	Python shell		s3://aws-glue-scripts-...	27 May 2019 3:18 PM ...	Disable

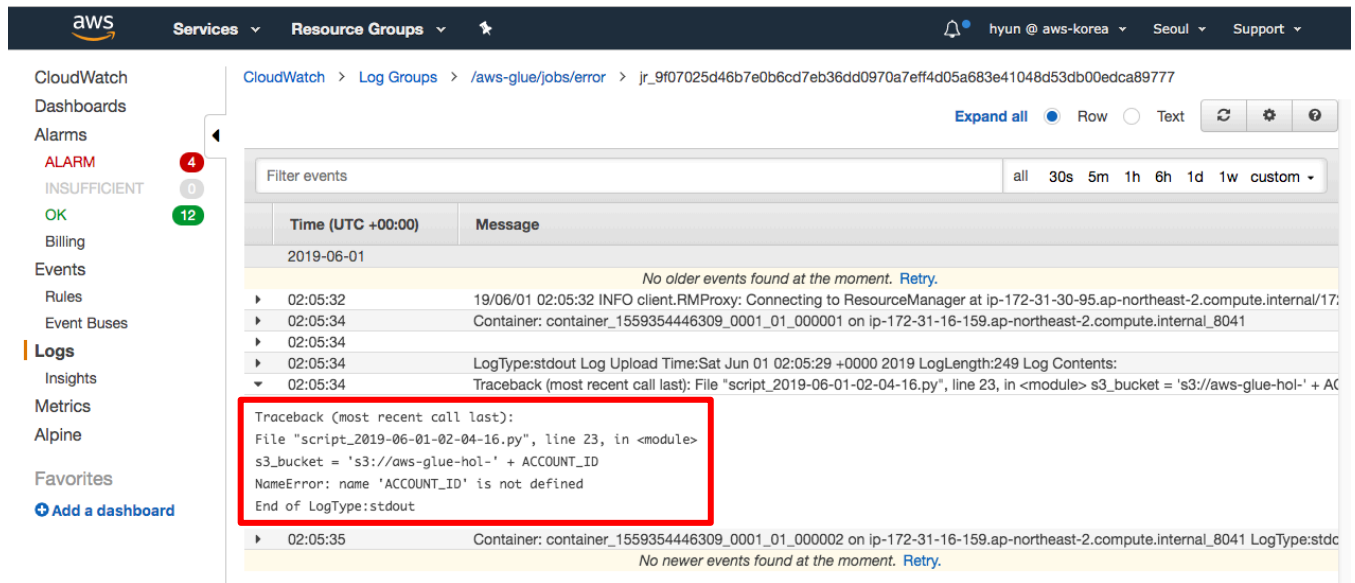
Run ID	Retry attempt	Run status	Error	Logs	Error logs	Maximum capacity	Execution time	Timeout	Delay	Triggered by	Start time	End time
<input type="radio"/> jr_9f07025d...	-	Failed	!	...	Logs	2	1 min	2880 mins			1 June...	1 June...

7-18. 로그를 클릭합니다.

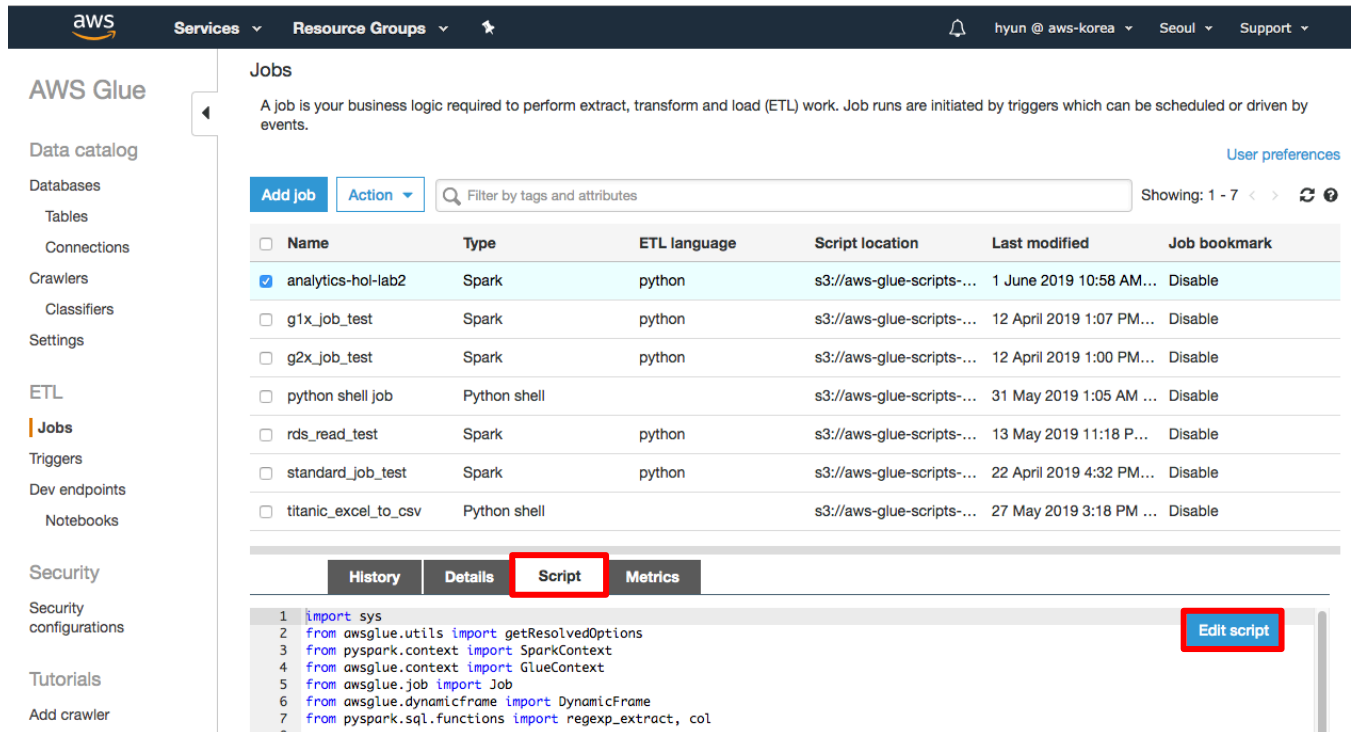
The screenshot shows the AWS CloudWatch console. The left navigation menu includes CloudWatch, Dashboards, Alarms, Events, Rules, Event Buses, Logs, Insights, Metrics, and Alpine. The main area displays the 'Log Groups' view for the path '/aws-glue/jobs/error'. A specific log group is selected, and the log events are shown in a table. The log event at 02:05:34 is highlighted with a red box, showing a traceback error message.

Time (UTC +00:00)	Message
2019-06-01	No older events found at the moment. Retry.
02:05:32	19/06/01 02:05:32 INFO client.RMPProxy: Connecting to ResourceManager at ip-172-31-30-95.ap-northeast-2.compute.internal/172.31.30.95
02:05:34	Container: container_1559354446309_0001_01_000001 on ip-172-31-16-159.ap-northeast-2.compute.internal_8041
02:05:34	LogType:stdout Log Upload Time:Sat, Jun 01 02:05:29 +0000 2019 Log length:249 Log Contents:
02:05:34	Traceback (most recent call last): File "script_2019-06-01-02-04-16.py", line 23, in <module> s3_bucket = 's3://aws-glue-hoi-' + A...
02:05:35	Container: container_1559354446309_0001_01_000002 on ip-172-31-16-159.ap-northeast-2.compute.internal_8041 LogType:stdout
	No newer events found at the moment. Retry.

7-19. 로그 메시지에서 에러 메시지와 에러 위치를 확인합니다. Line 23 에서 에러가 발생한 것을 확인할 수 있고 'ACCOUNT_ID' is not defined 에러 메시지를 확인할 수 있습니다.



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



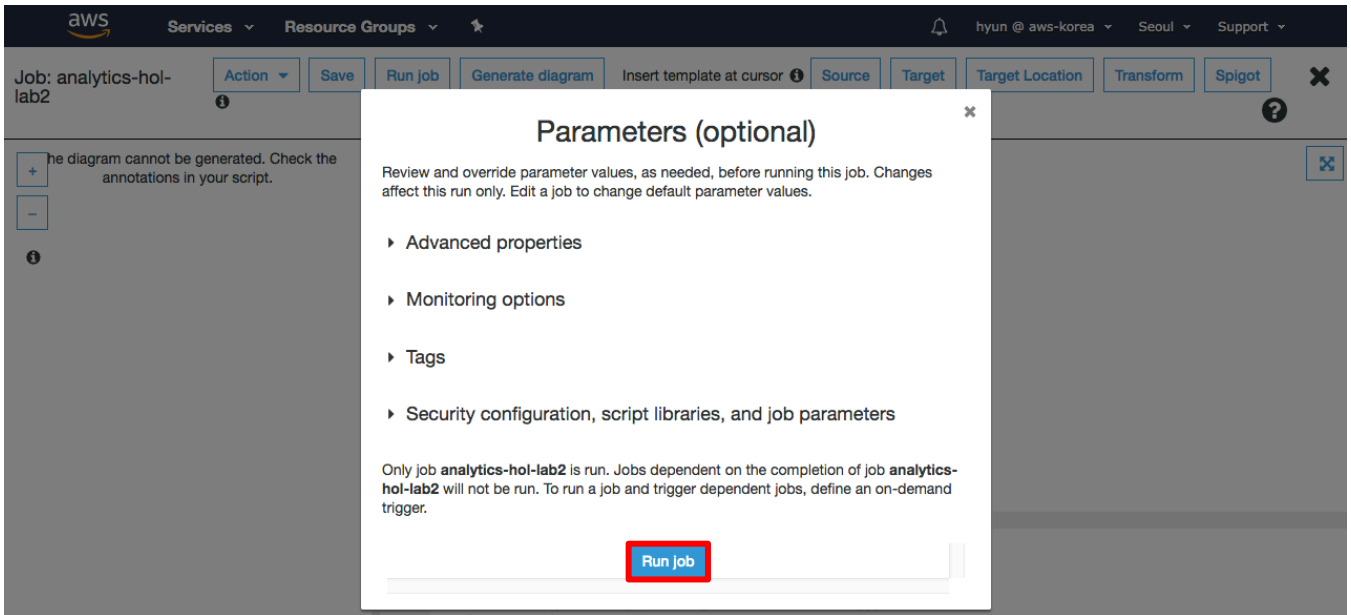
7-21. 아래 화면과 같이 line 23 의 ACCOUNT_ID 부분을 본인의 account_id 로 대체하고 Save 버튼을 클릭한 후 Run job 버튼을 클릭합니다.

The diagram cannot be generated. Check the annotations in your script.

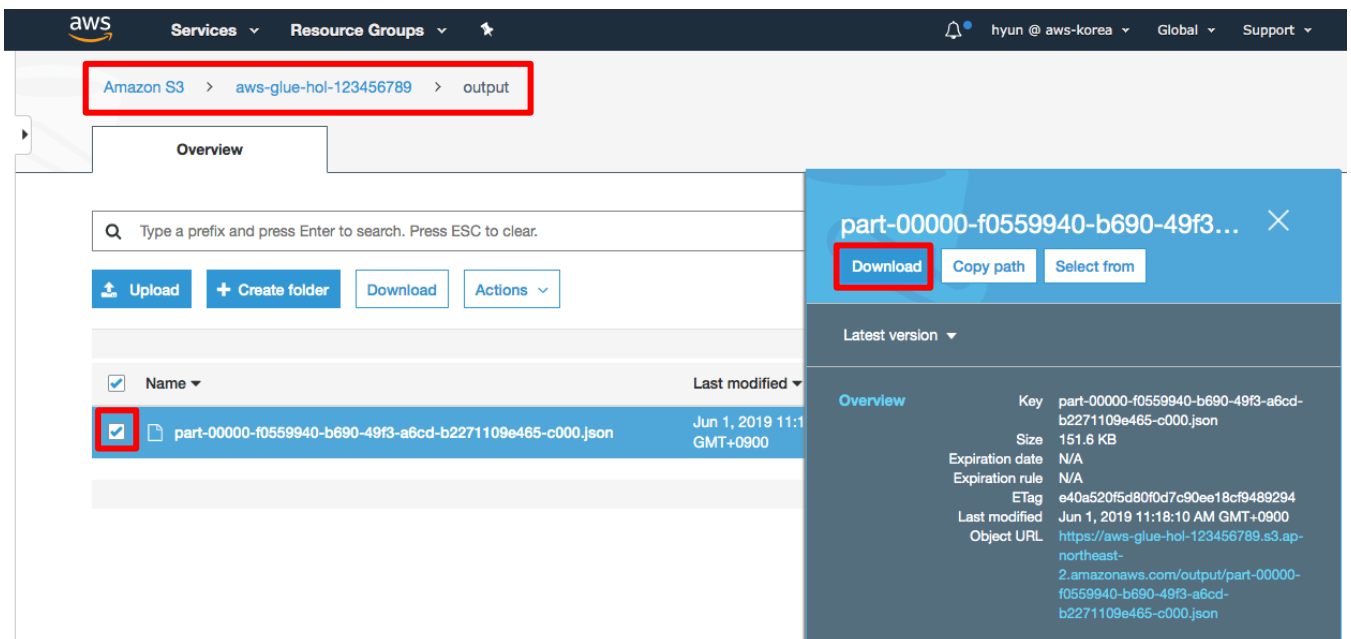
```

1 import sys
2 from awsglue.utils import getResolvedOptions
3 from pyspark.context import SparkContext
4 from awsglue.context import GlueContext
5 from awsglue.job import Job
6 from awsglue.dynamicframe import DynamicFrame
7 from pyspark.sql.functions import regexp_extract, col
8
9 ## @params: [JOB_NAME]
10 args = getResolvedOptions(sys.argv, ['JOB_NAME'])
11
12 # SparkContext 변수 생성
13 sc = SparkContext()
14 # GlueContext 변수 생성
15 glueContext = GlueContext(sc)
16 # SparkSession 변수 생성
17 spark = glueContext.spark_session
18 # Job 변수 생성
19 job = Job(glueContext)
20 # Job 초기화
21 job.init(args['JOB_NAME'], args)
22
23 s3_bucket = 's3://aws-glue-hol-' + '123456789'
24
25 # S3에서 csv 데이터를 읽어 DynamicFrame 생성
26 titanic_dyf = glueContext.create_dynamic_frame_from_options(
27     connection_type = 's3',
28     connection_options = {'paths': [s3_bucket + '/train']},
29     format='csv',
30     format_options={
31         "withHeader": True,
32         "delimiter": ','
33     })
34
35 # Spark 활용: DynamicFrame을 DataFrame으로 변환 및 initial column을 추가
36 titanic_csv_df = titanic_dyf.toDF()
37 titanic_csv_df = titanic_csv_df.withColumn('initial', regexp_extract(col('Name'), "(\\w+\\.)", 1))
38
39 # Glue 활용: DataFrame을 DynamicFrame으로 변환하여 Name column 삭제
40 titanic_csv_dyf = DynamicFrame.fromDF(titanic_csv_df, glueContext, 'titanic_csv_dyf').drop_fields('Name')
41
42 # json format으로 output 디렉토리에 저장
43 titanic_csv_dyf.toDF().write \
44     .format('json') \
45     .mode('overwrite') \
46     .save(s3_bucket + '/output')
47 # Job commit
48 job.commit()
    
```

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

```
titanic_train.csv
1 PassengerId,Survived,Pclass,Name,Sex,Age,SibSp,Parch,Ticket,Fare,Cabin,Embarked
2 1,0,3,"Mr. Owen Harris",male,22,1,0,A/5 21171,7.25,,S
3 2,1,1,"Mrs. John Bradley (Florence Briggs Thayer)",female,38,1,0,PC 17599,71.2833,C85,C
4 3,1,3,"Miss. Laina",female,26,0,0,STON/O2. 3101282,7.925,,S
5 4,1,1,"Mrs. Jacques Heath (Lily May Peel)",female,35,1,0,113803,53.1,C123,S
```

After

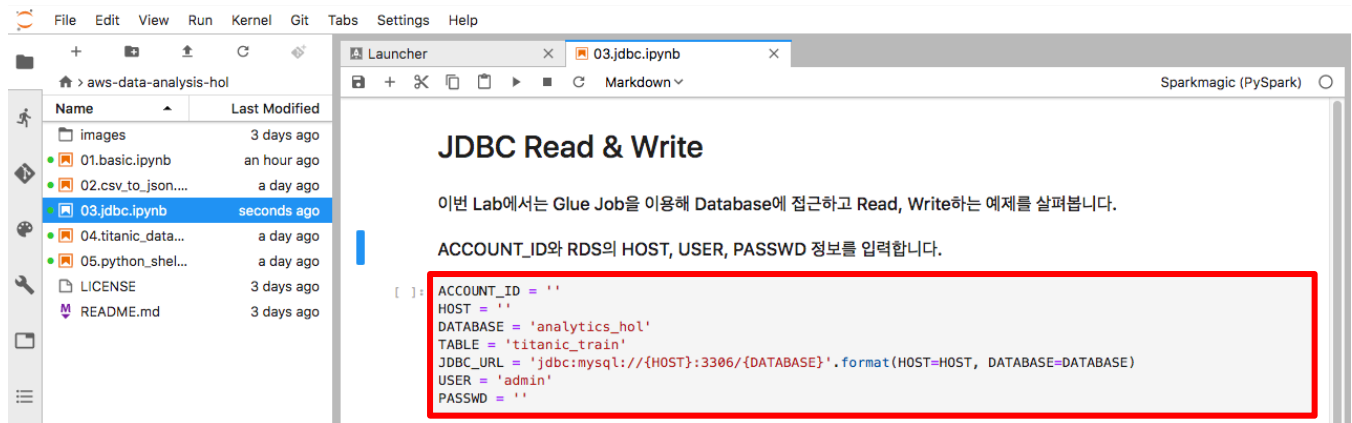
```
part-00000-f91d3ec3-28d0-4695-96ea-b1d88ce6059b-c000.json
1 {"PassengerId":1,"Survived":0,"Pclass":3,"Name":"Mr. Owen Harris","Sex":"male","Age":22,"SibSp":1,"Parch":0,"Ticket":"A/5 21171","Fare":7.25,"Cabin":"","Embarked":"S","initial":"Mr"}
2 {"PassengerId":2,"Survived":1,"Pclass":1,"Name":"Mrs. John Bradley (Florence Briggs Thayer)", "Ticket":"PC 17599", "Fare":71.2833, "Cabin":"C85", "Embarked":"C", "initial":"Mrs"}
3 {"PassengerId":3,"Survived":1,"Pclass":3,"Name":"Miss. Laina", "Sex":"female", "Age":26, "Cabin":"","Embarked":"S", "initial":"Miss"}
4 {"PassengerId":4,"Survived":1,"Pclass":1,"Name":"Mrs. Jacques Heath (Lily May Peel)", "113803", "Fare":53.1, "Cabin":"C123", "Embarked":"S", "initial":"Mrs"}
5 {"PassengerId":5,"Survived":0,"Pclass":3,"Name":"Mr. William Henry", "Sex":"male", "A":26, "Cabin":"","Embarked":"S", "initial":"Mr"}
```

7-24. 생성된 Job 에 대한 스케줄 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.

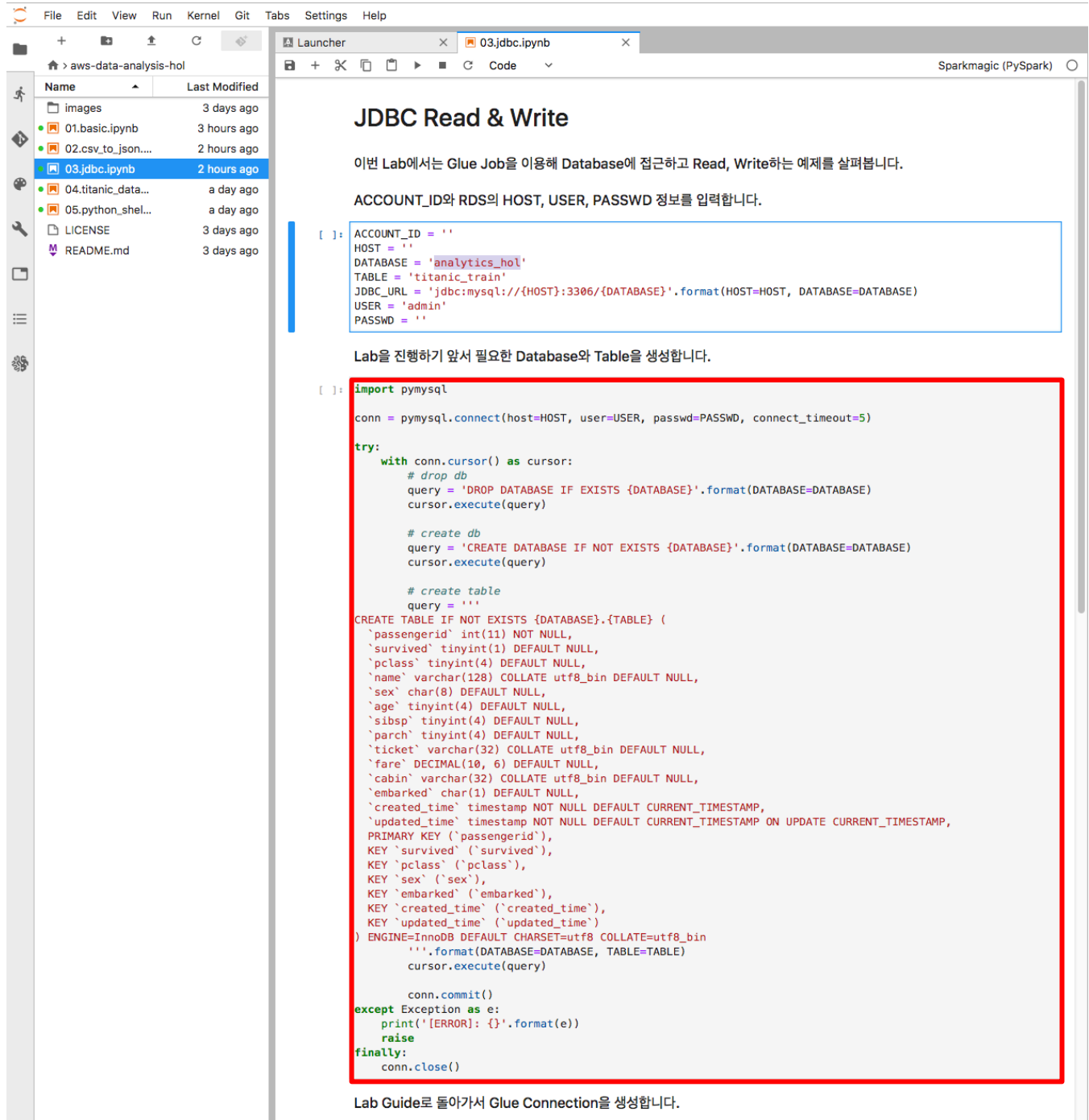
7-25. Lab3 진행을 위해 03.jdbc.ipynb 노트북 파일을 클릭하여 엽니다. ACCOUNT_ID 와 RDS 의 HOST, USER, PASSWD 정보를 입력하고 실행합니다. (DATABASE 와 TABLE 은 Jupyter Notebook 에 있는 이름을 그대로 사용합니다.)



The screenshot shows a Jupyter Notebook interface with a file explorer on the left and a code editor on the right. The file explorer shows a directory named 'aws-data-analysis-hol' with several files, including '03.jdbc.ipynb' which is selected. The code editor displays the following code:

```
[ ]: ACCOUNT_ID = ''
HOST = ''
DATABASE = 'analytics_hol'
TABLE = 'titanic_train'
JDBC_URL = 'jdbc:mysql://{HOST}:3306/{DATABASE}'.format(HOST=HOST, DATABASE=DATABASE)
USER = 'admin'
PASSWD = ''
```

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



JDBC Read & Write

이번 Lab에서는 Glue Job을 이용해 Database에 접근하고 Read, Write하는 예제를 살펴봅니다.

ACCOUNT_ID와 RDS의 HOST, USER, PASSWD 정보를 입력합니다.

```
[ ]: ACCOUNT_ID = ''
HOST = ''
DATABASE = 'analytics_ho'
TABLE = 'titanic_train'
JDBC_URL = 'jdbc:mysql://{HOST}:3306/{DATABASE}'.format(HOST=HOST, DATABASE=DATABASE)
USER = 'admin'
PASSWD = ''
```

Lab을 진행하기 앞서 필요한 Database와 Table을 생성합니다.

```
[ ]: import pymysql

conn = pymysql.connect(host=HOST, user=USER, passwd=PASSWD, connect_timeout=5)

try:
    with conn.cursor() as cursor:
        # drop db
        query = 'DROP DATABASE IF EXISTS {DATABASE}'.format(DATABASE=DATABASE)
        cursor.execute(query)

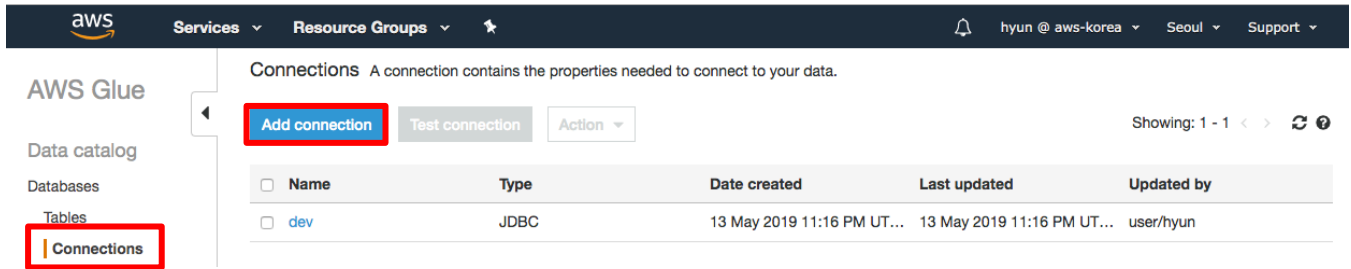
        # create db
        query = 'CREATE DATABASE IF NOT EXISTS {DATABASE}'.format(DATABASE=DATABASE)
        cursor.execute(query)

        # create table
        query = '''
CREATE TABLE IF NOT EXISTS {DATABASE}.{TABLE} (
    `passengerid` int(11) NOT NULL,
    `survived` tinyint(1) DEFAULT NULL,
    `pclass` tinyint(4) DEFAULT NULL,
    `name` varchar(128) COLLATE utf8_bin DEFAULT NULL,
    `sex` char(8) DEFAULT NULL,
    `age` tinyint(4) DEFAULT NULL,
    `sibsp` tinyint(4) DEFAULT NULL,
    `parch` tinyint(4) DEFAULT NULL,
    `ticket` varchar(32) COLLATE utf8_bin DEFAULT NULL,
    `fare` DECIMAL(10, 6) DEFAULT NULL,
    `cabin` varchar(32) COLLATE utf8_bin DEFAULT NULL,
    `embarked` char(1) DEFAULT NULL,
    `created_time` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
    `updated_time` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    PRIMARY KEY (`passengerid`),
    KEY `survived` (`survived`),
    KEY `pclass` (`pclass`),
    KEY `sex` (`sex`),
    KEY `embarked` (`embarked`),
    KEY `created_time` (`created_time`),
    KEY `updated_time` (`updated_time`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin
''',format(DATABASE=DATABASE, TABLE=TABLE)
        cursor.execute(query)

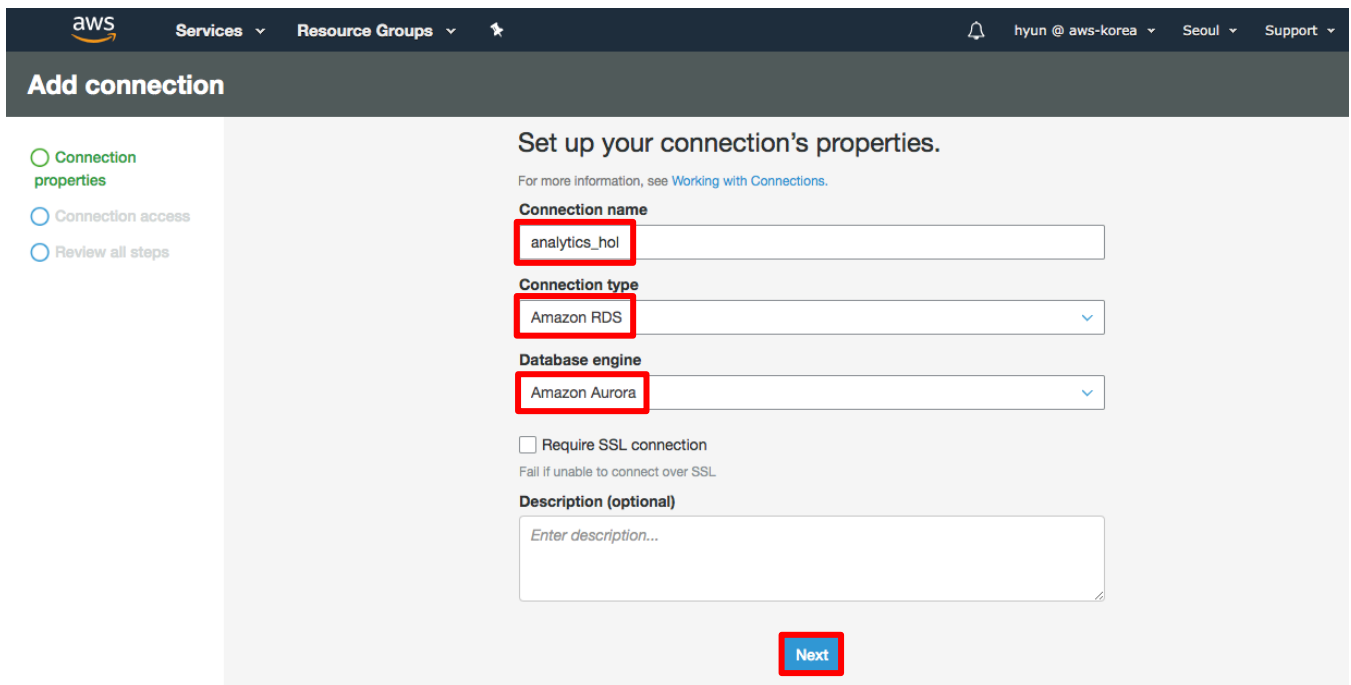
        conn.commit()
except Exception as e:
    print('[ERROR]: {}'.format(e))
    raise
finally:
    conn.close()
```

Lab Guide로 돌아가서 Glue Connection을 생성합니다.

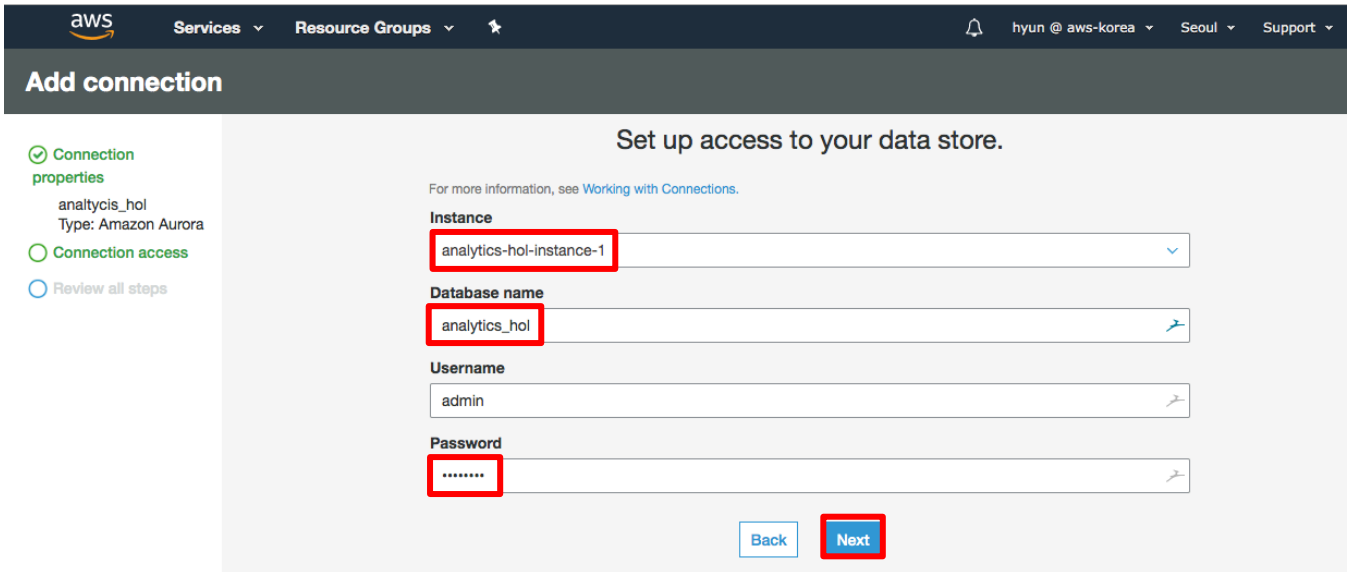
7-27. RDS 에 Database 와 Table 이 생성되었으면 Glue Connection 생성을 위해 Glue 콘솔로 이동 후 Add connection 버튼을 클릭합니다.



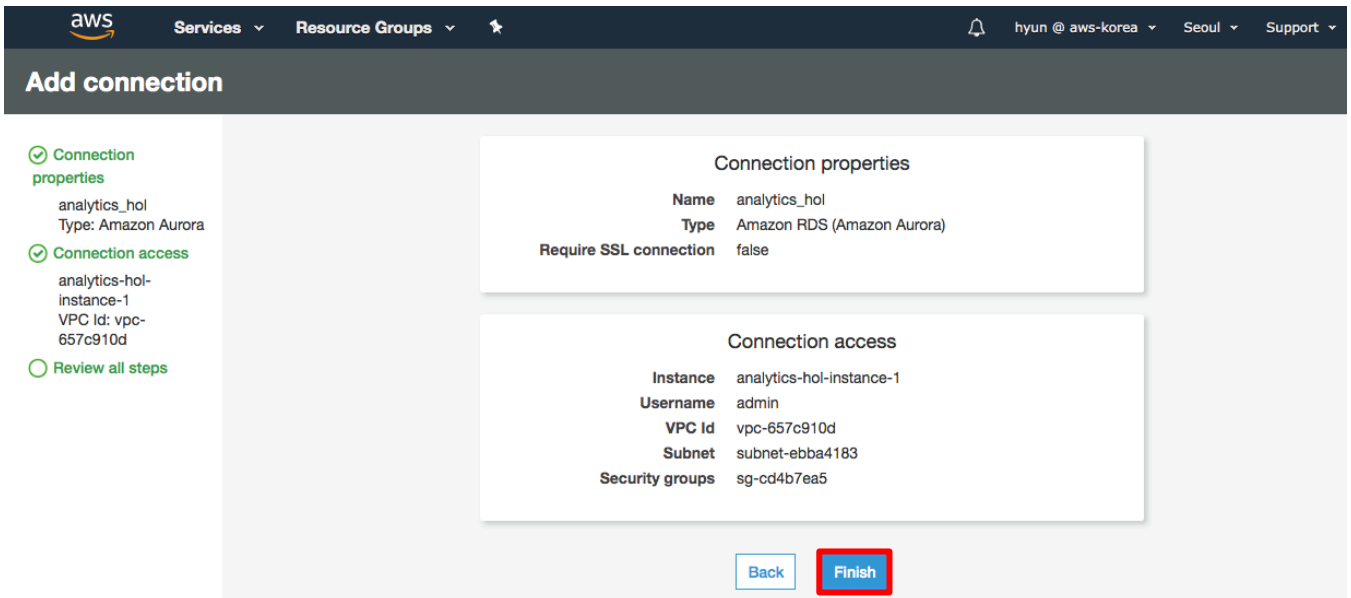
7-28. Connection name 에 analytics_hol, Connection type 에 Amazon RDS, Database engine 에 Amazon Aurora 를 입력하고 Next 버튼을 클릭합니다.

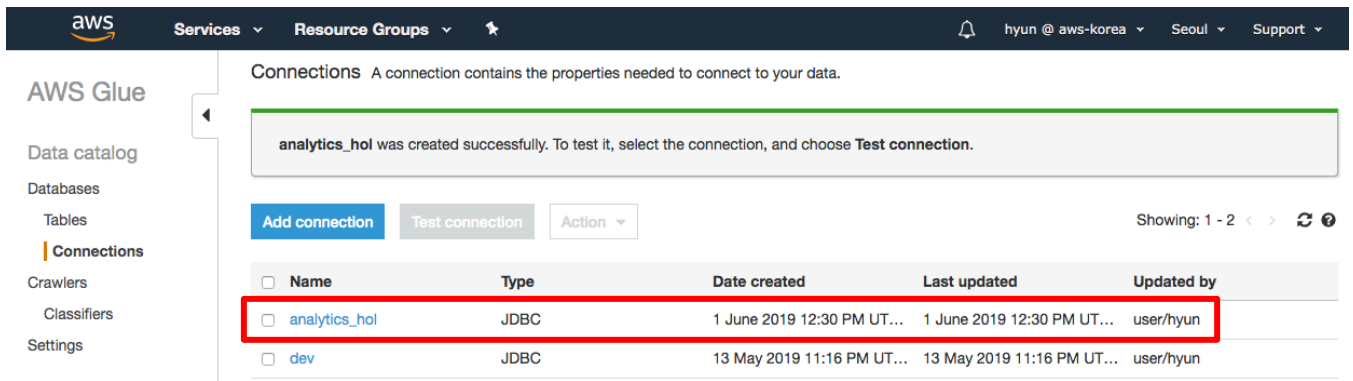


7-29. Instance 에 생성한 RDS analytics-hol-instance-1 을 선택하고, Database name 에 analytics_hol, Username admin 과 Password 를 입력한 후 Next 버튼을 클릭합니다.

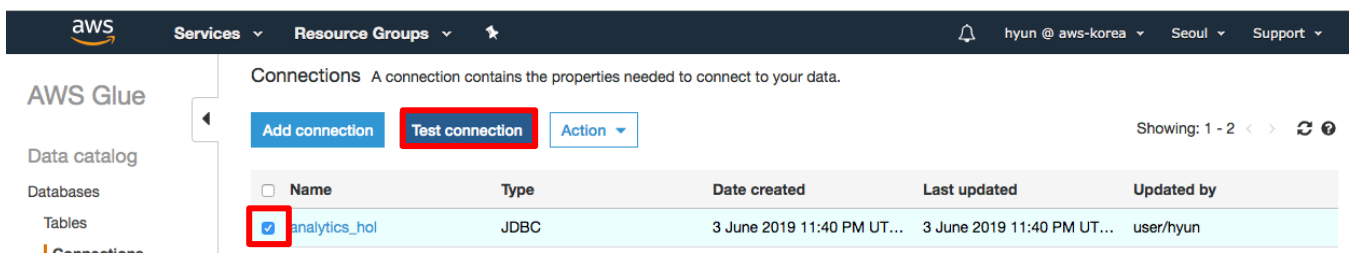


7-30. 지금까지 작성한 내용을 Review 한 후 Finish 버튼을 클릭하면 analytics_hol Connection 이 생성된 것을 확인할 수 있습니다.

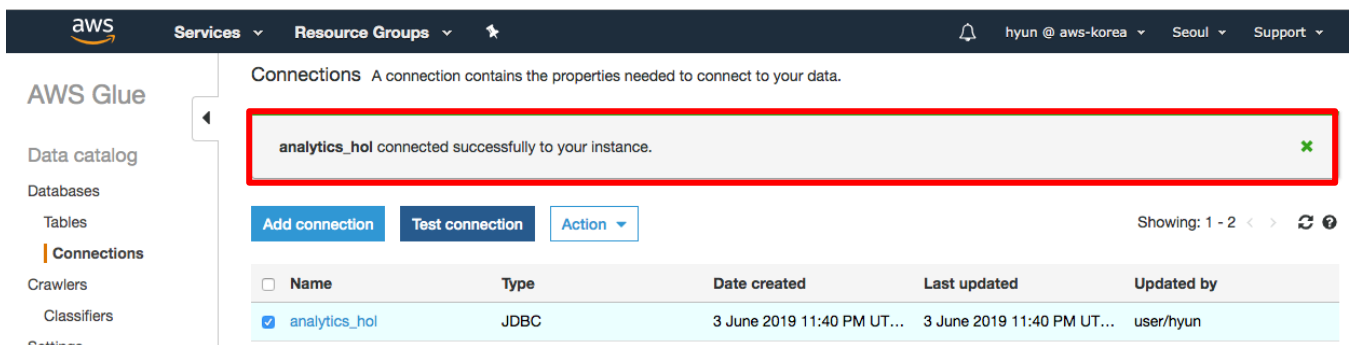
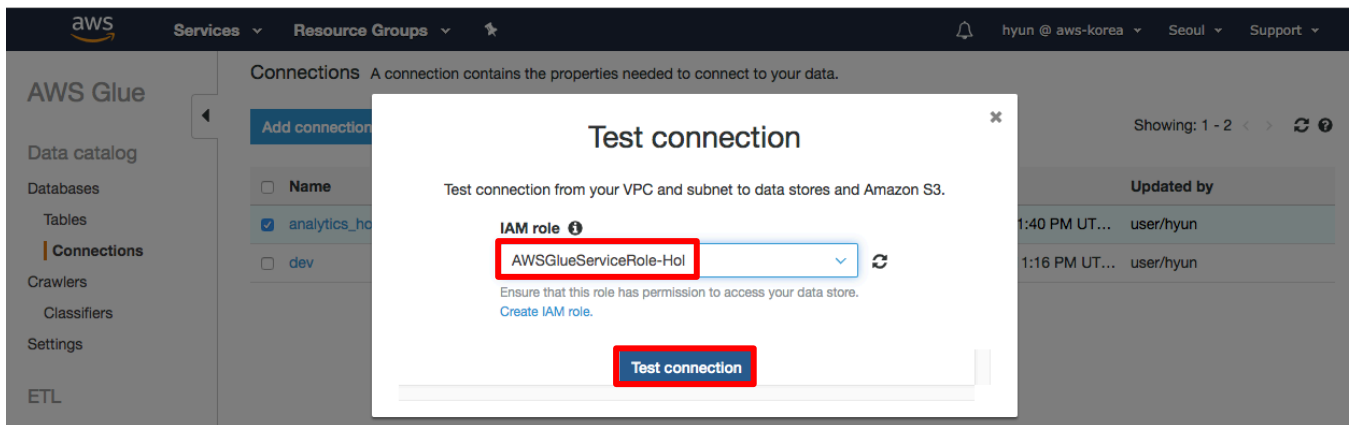




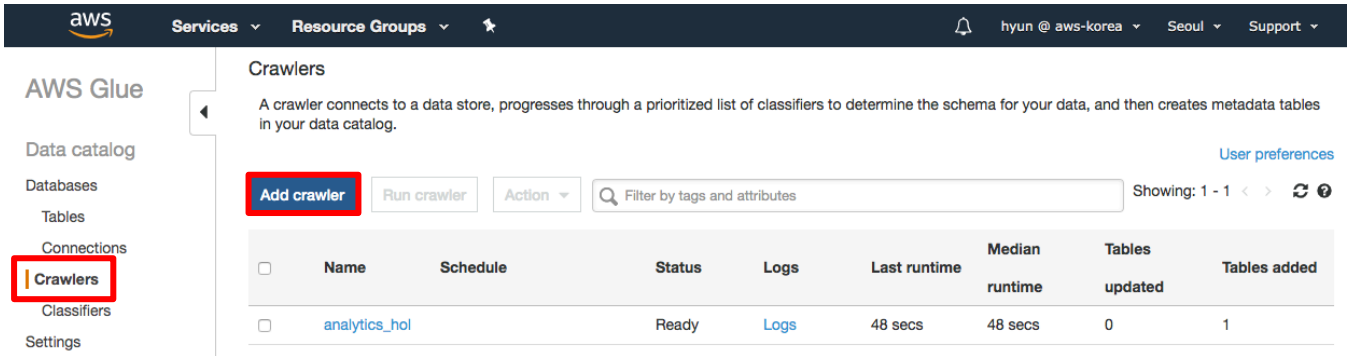
7-31. 다음은 지금 생성한 Connection 이 정상적으로 연결되는지 테스트하기 위해 analytics_hol connection 을 선택하고 Test connection 버튼을 클릭합니다.



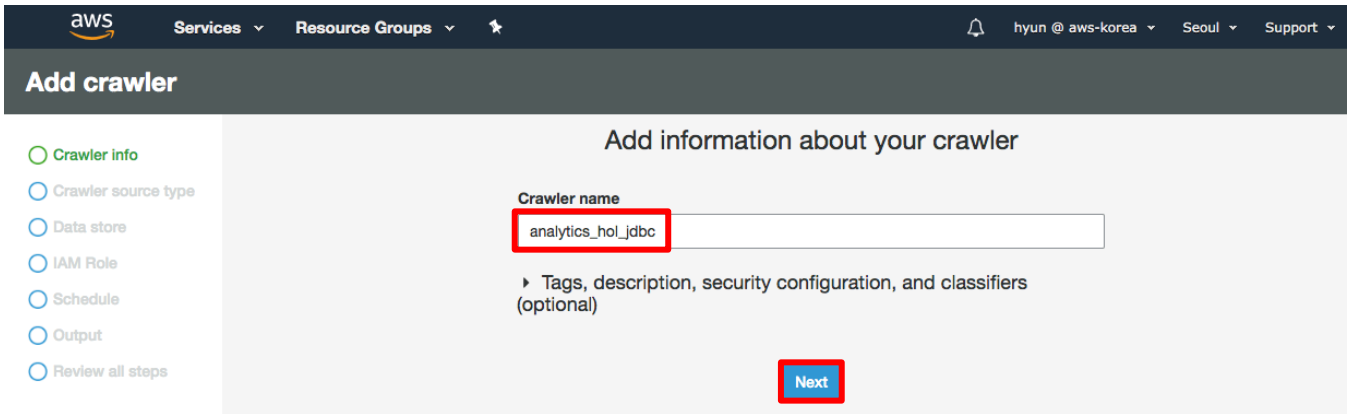
7-32. IAM role 에 AWSGlueServiceRole-Hol 을 선택하고 Test connection 버튼을 클릭합니다.



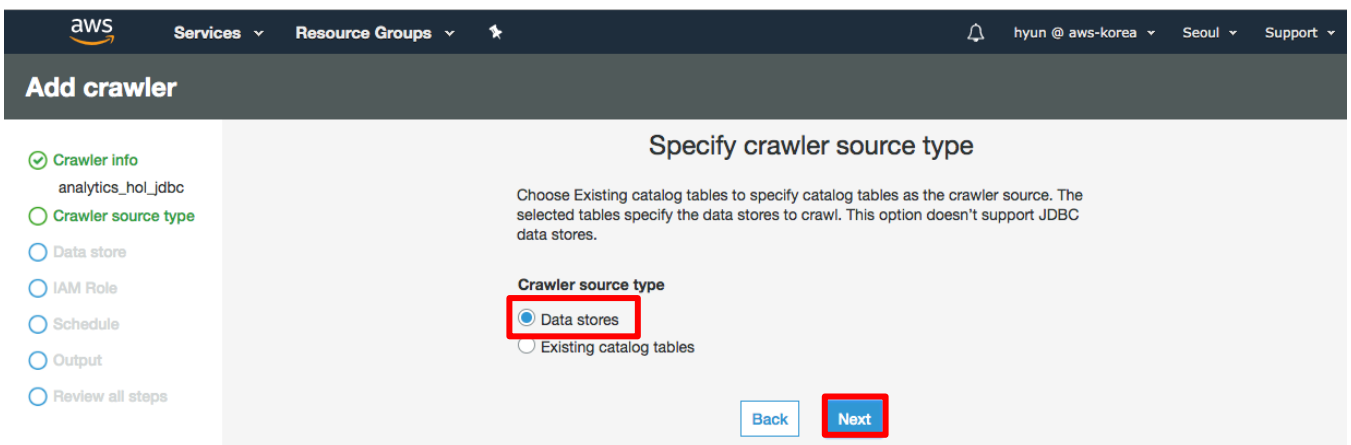
7-33. 다음은 지금 생성한 Connection 을 이용해서 RDS 에 생성된 Table 을 Glue Catalog 에 등록합니다. Glue 콘솔에 Crawlers 로 이동하여 Add crawler 버튼을 클릭합니다.



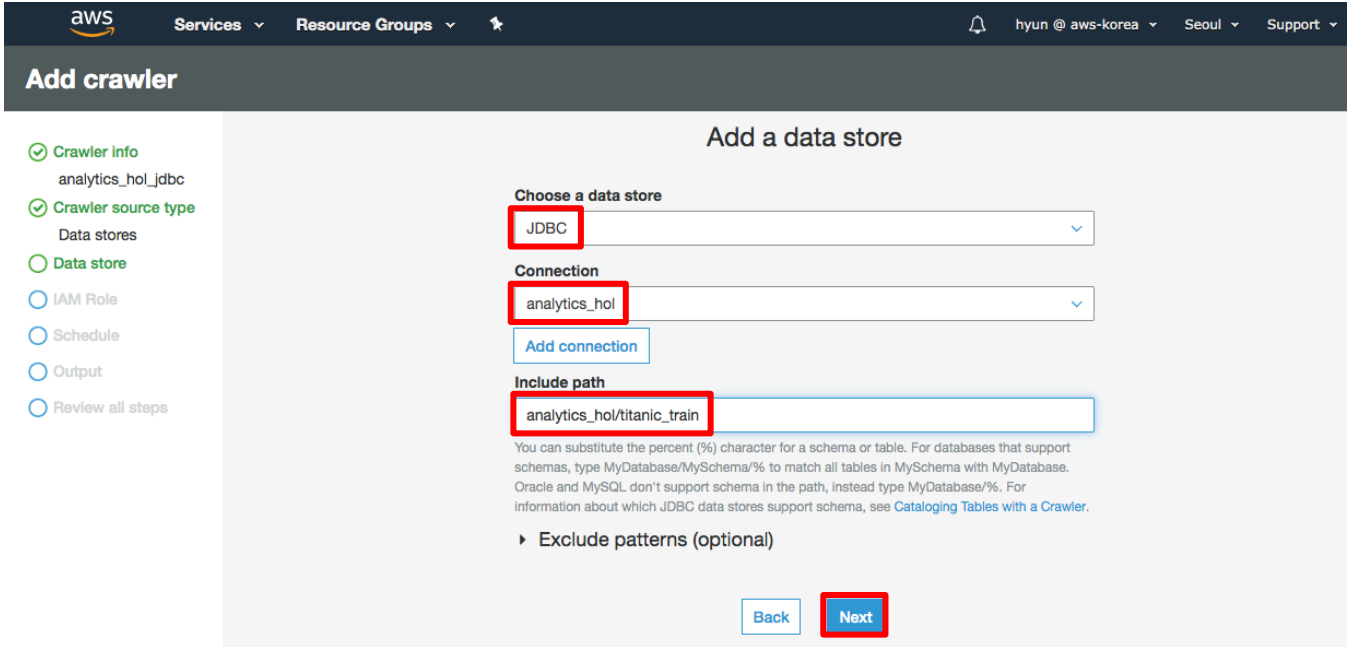
7-34. Crawler name 에 analytics_hol_jdbc 를 입력하고 Next 버튼을 클릭합니다.



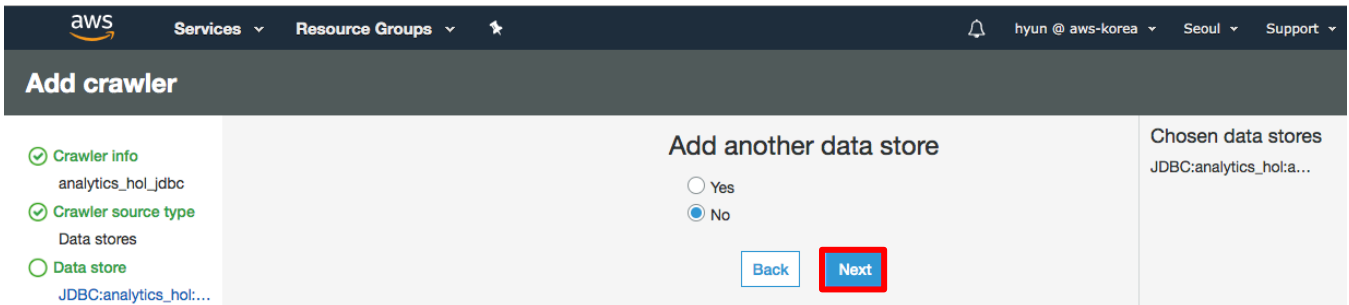
7-35. Crawler source type 을 기본 Data stores 로 두고 Next 버튼을 클릭합니다.



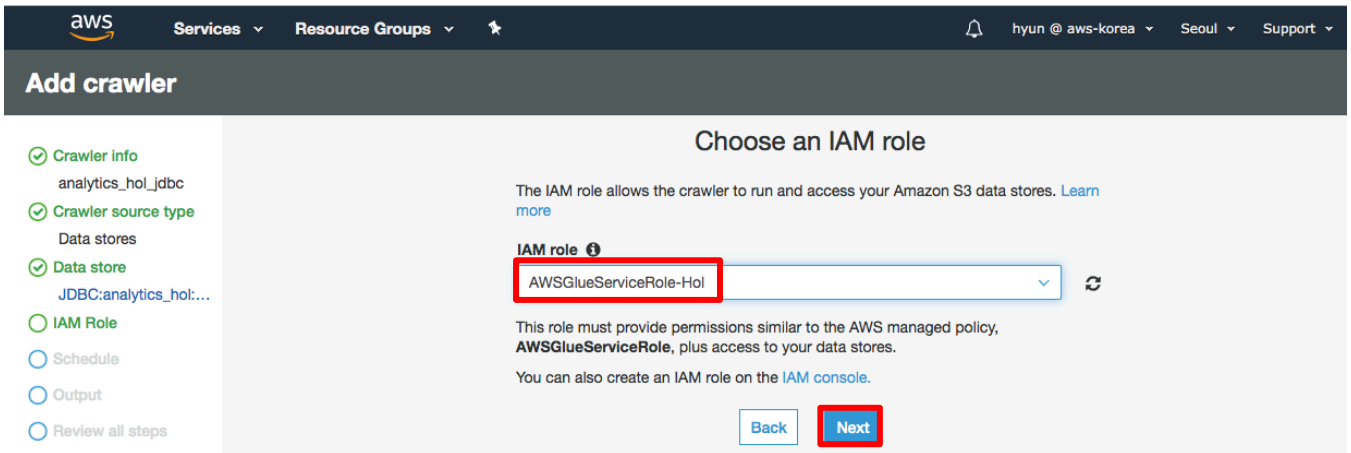
7-36. Choose a data store 에 JDBC, Connection 에 이전에 생성한 analytics_hol, include path 에 analytics_hol/titanic_train 을 입력하고 Next 버튼을 클릭합니다. Include path 는 database 와 table 이름을 의미합니다.



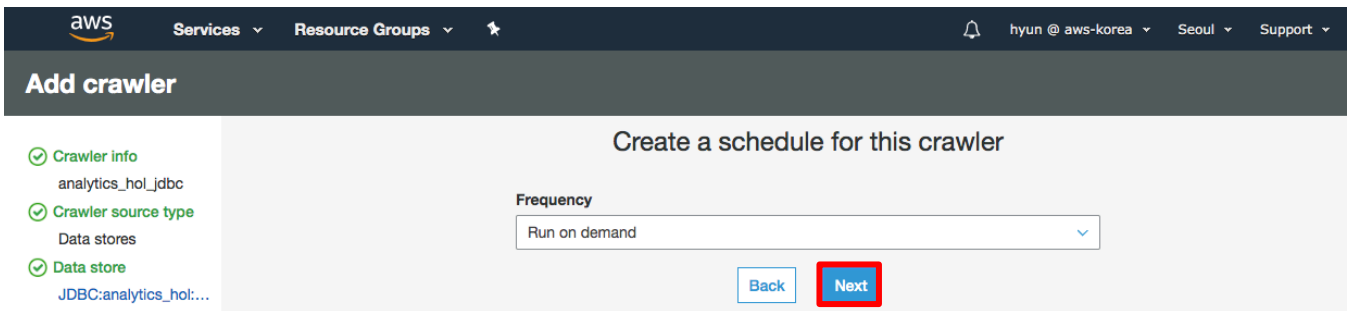
7-37. Add another data store 에서는 기본 No 인 상태에서 Next 버튼을 클릭합니다.



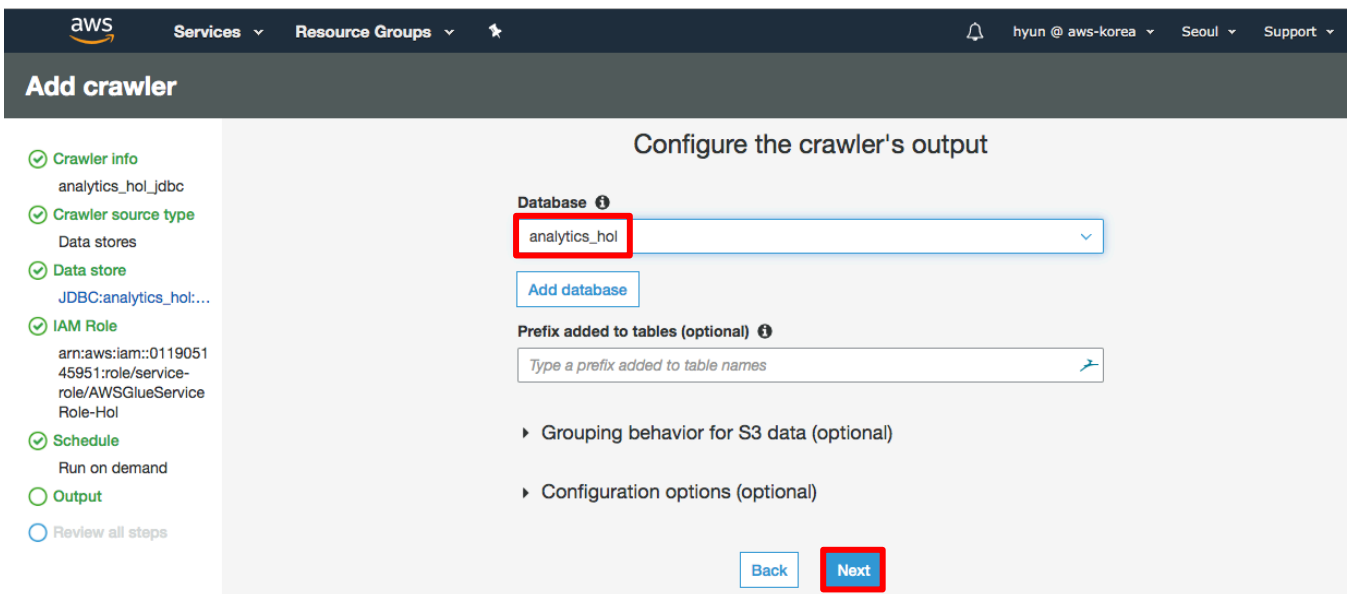
7-38. IAM role 에서 이전에 생성한 AWSGlueServiceRole-Hol 을 선택하고 Next 버튼을 클릭합니다.



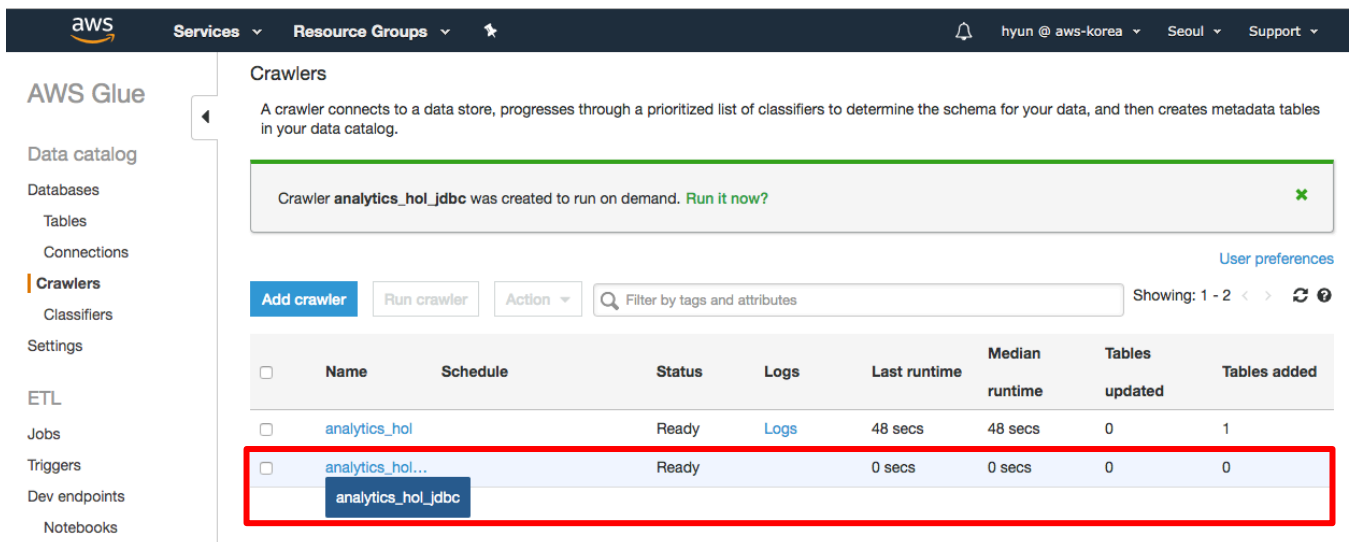
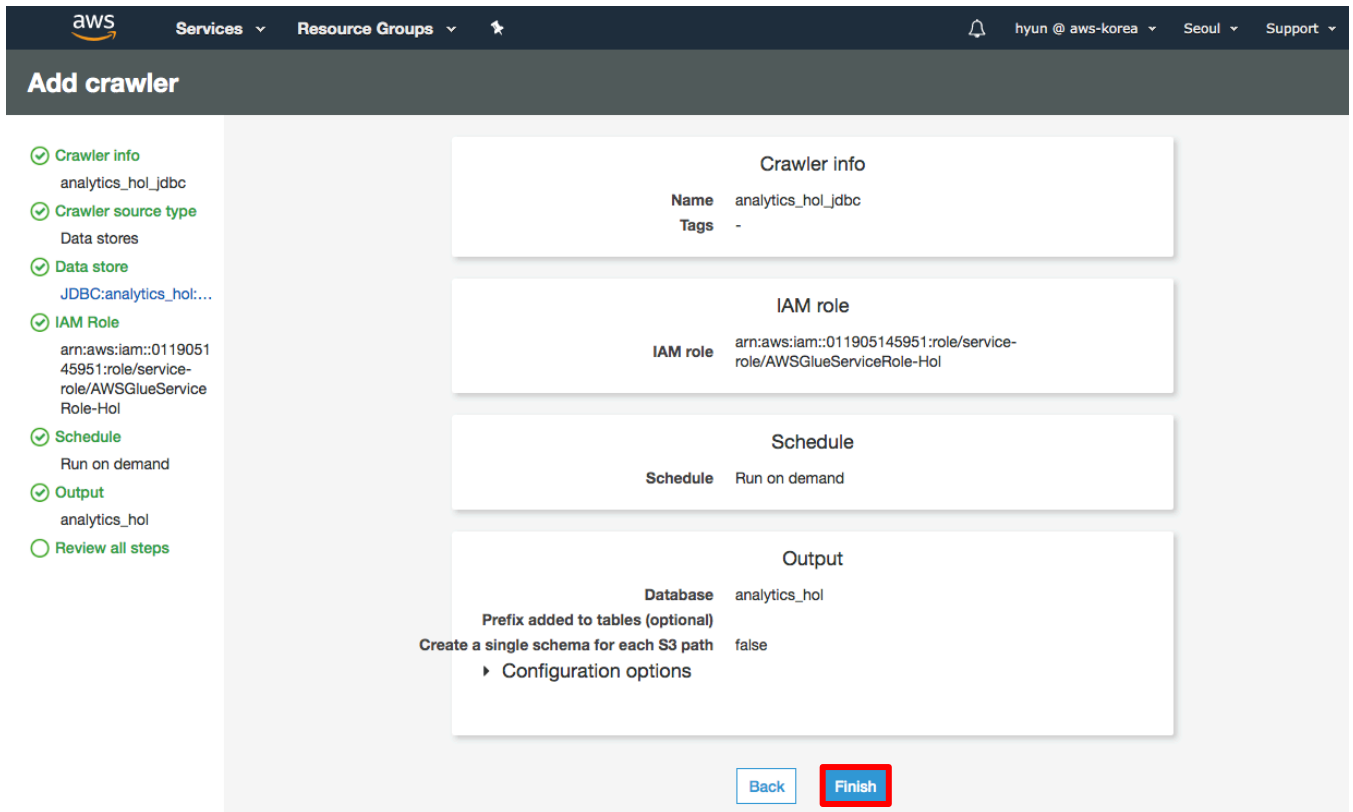
7-39. Schedule 설정은 Run on demand 상태로 두고 Next 버튼을 클릭합니다.



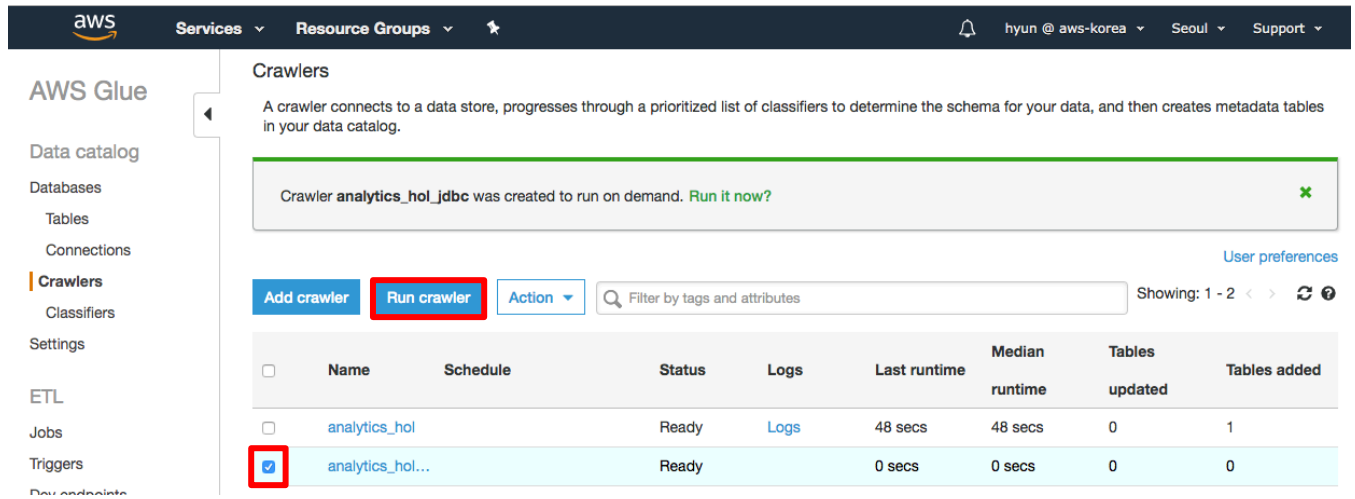
7-40. Glue Database 는 analytics_hol 을 선택하고 Next 버튼을 클릭합니다.



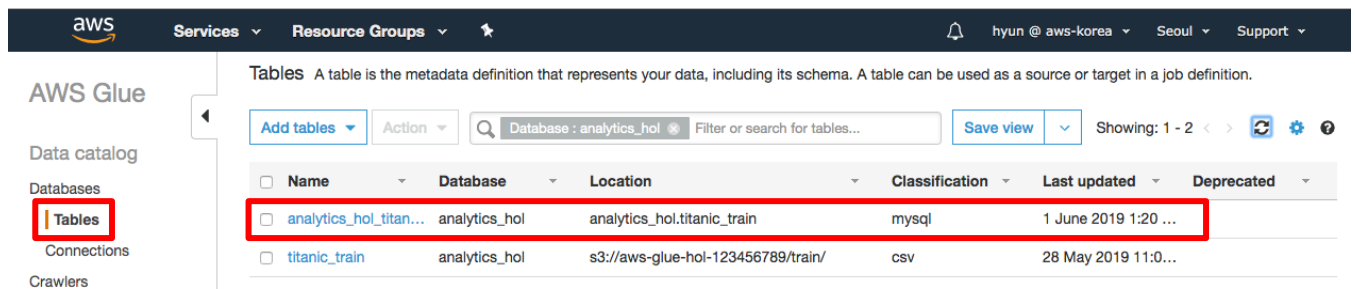
7-41. Review 후 Finish 버튼을 클릭하면 analytics_hol_jdbc crawler 가 정상적으로 생성된 것을 확인할 수 있습니다.



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 Lab에서는 실제 빅데이터를 분석하는 과정에서 Glue를 어떻게 활용할 수 있는지 kaggle titanic competition dataset을 가지고 분석, ETL, Feature Engineering 및 Spark ML을 활용한 Modeling까지 실습을 해보도록 하겠습니다.

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

The screenshot shows a JupyterLab environment. On the left, a file explorer displays the directory structure of 'aws-data-analysis-hol', with '02.titanic_data...' selected. The main area shows a notebook with the following content:

Titanic: Predict survival on the Titanic with AWS Glue.

타이타닉 사고에서 어떤 승객이 살아남을 수 있을 지 예측해보세요

Kaggle은 2010년 설립된 예측모델 및 분석 대회 플랫폼입니다.

기업 및 단체에서 데이터와 해결과제를 등록하면, 데이터 과학자들이 이를 해결하는 모델을 개발하고 경쟁하는 곳입니다.

이번 Lab에서는 Kaggle에서 입문자용 tutorial로 사용되는 titanic competition을 Glue와 Spark ML을 사용해서 데이터 분석, ETL, ML을 사용한 Prediction까지 실습해보도록 하겠습니다.

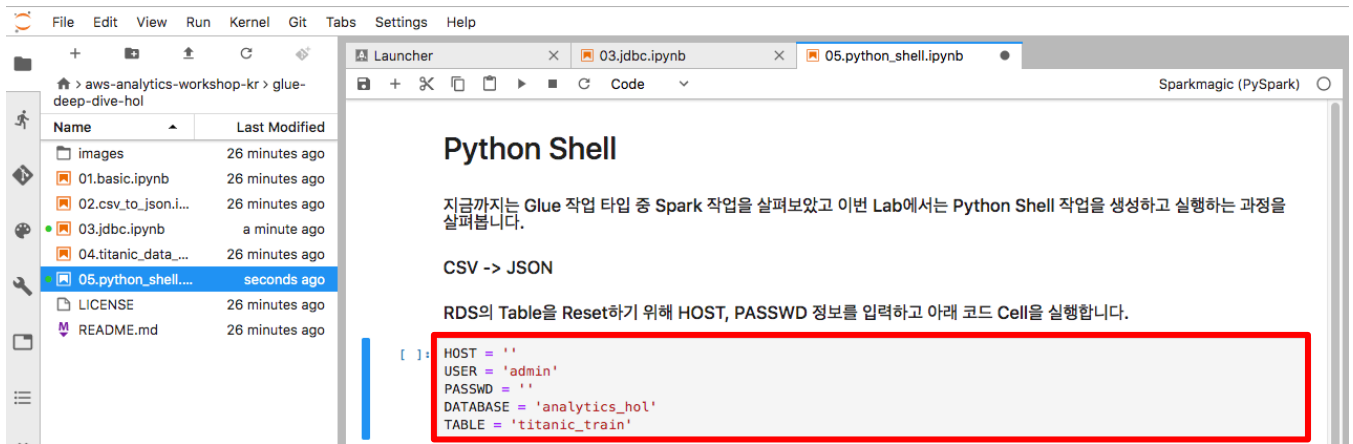
The notebook also features a large image of the Titanic ship at night, illuminated by its lights and smokestacks.

7-46. 노트북 가이드에 따라 Lab을 진행합니다.

Lab5: Python Shell Job

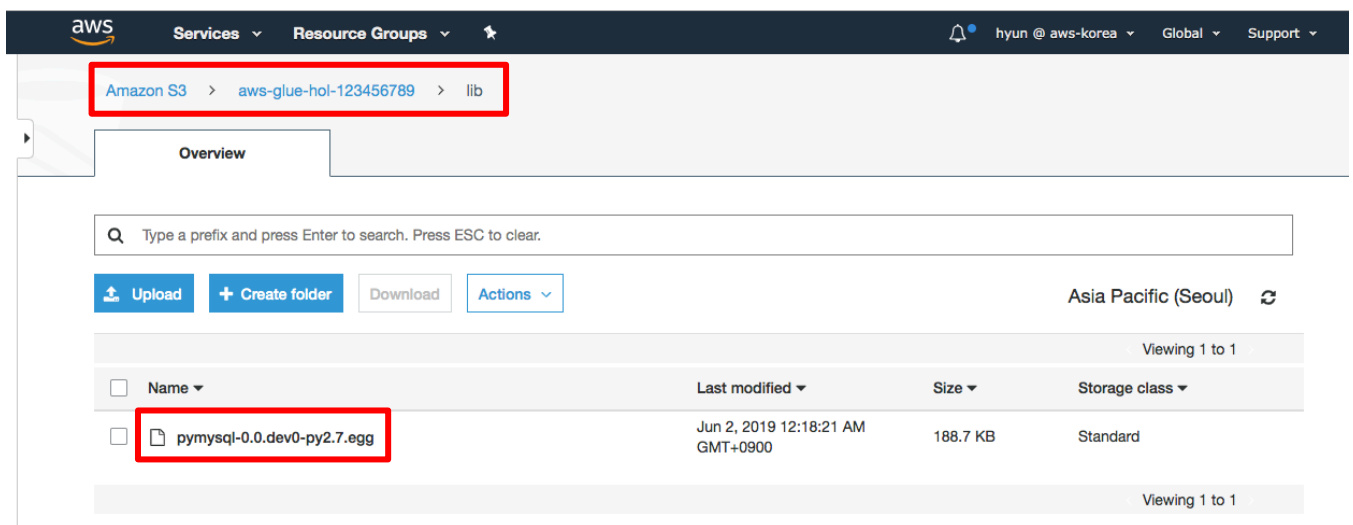
지금까지는 작업 타입 중 Spark 과 관련된 실습을 해보았습니다. 이번 Lab5 에서는 장비 한대에서 처리 가능한 작은 데이터를 다룰 때 사용할 수 있는 Python Shell 작업 타입을 실습 해보도록 하겠습니다.

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

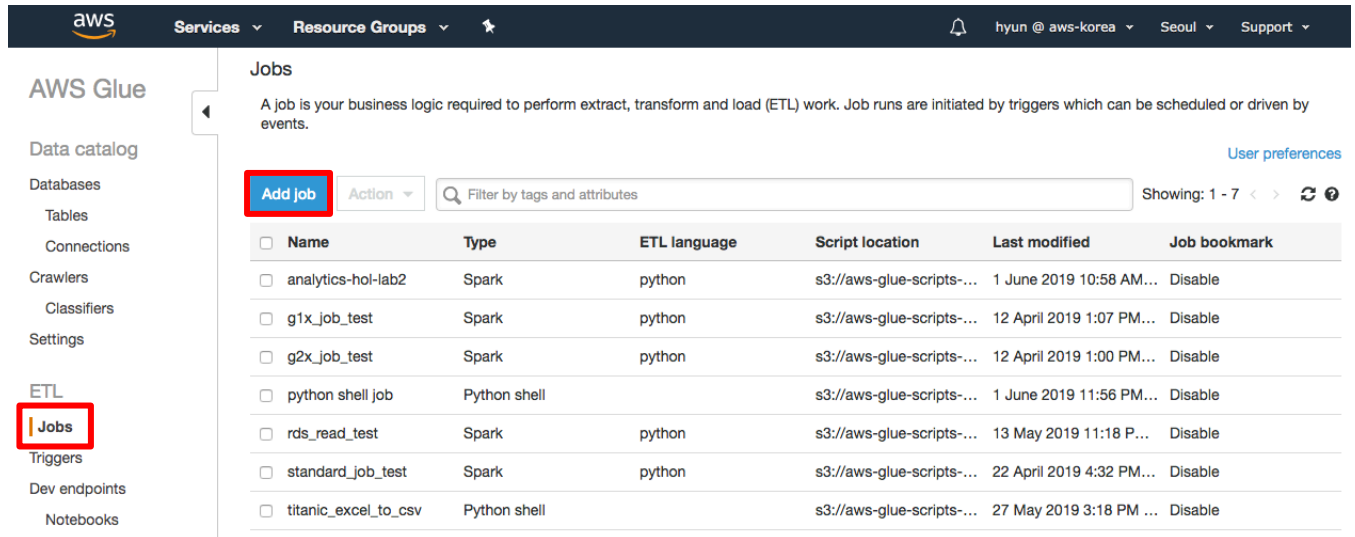


7-48. Python Shell Job 에서 RDS 에 접속하기 위해 <http://bit.ly/hol-pymysql> egg 파일을 다운로드 받습니다. (Python Shell 에서 사용하기 위한 egg 파일을 생성하는 방법은 Appendix A 를 참고하세요)

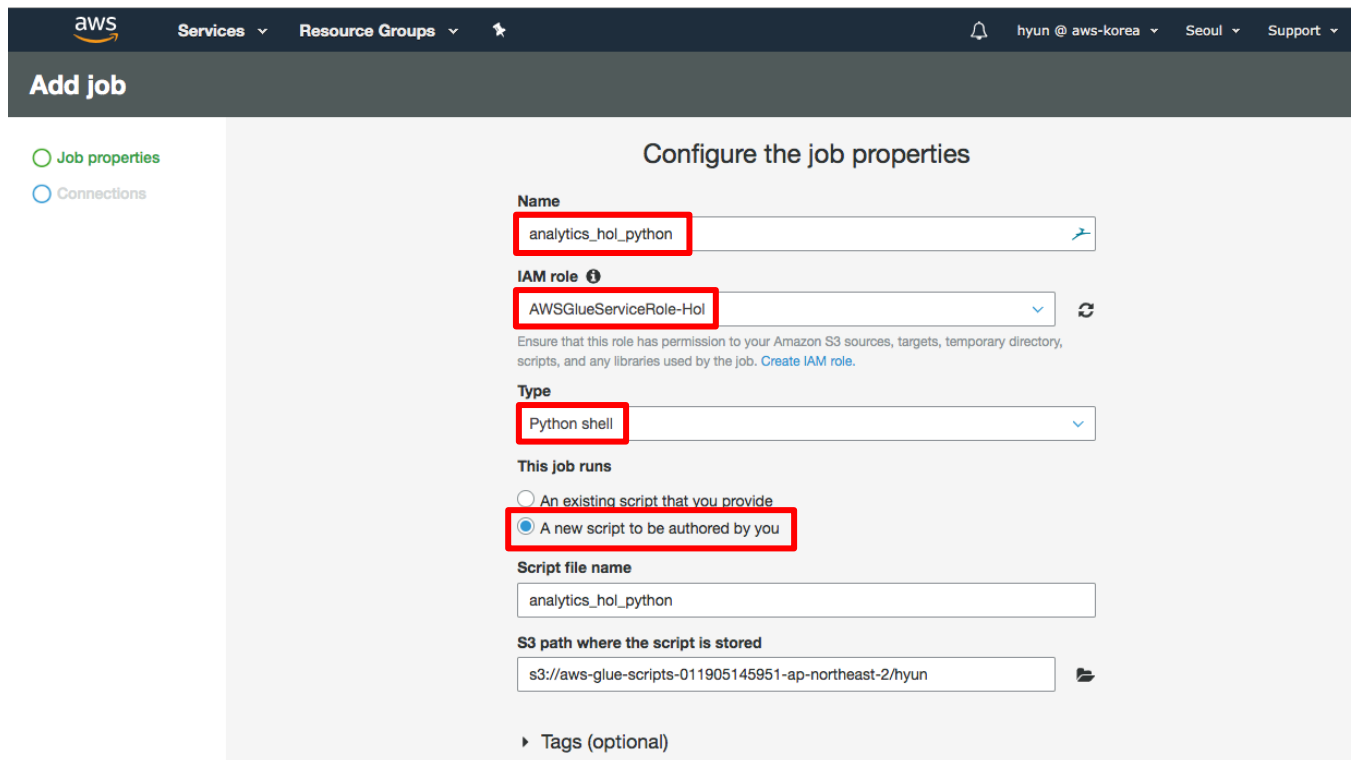
7-49. S3 콘솔로 이동 후 aws-glue-hol-[account-id] 버킷에 lib 디렉토리를 생성하고 위에서 다운로드 받은 파일을 업로드 합니다.



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

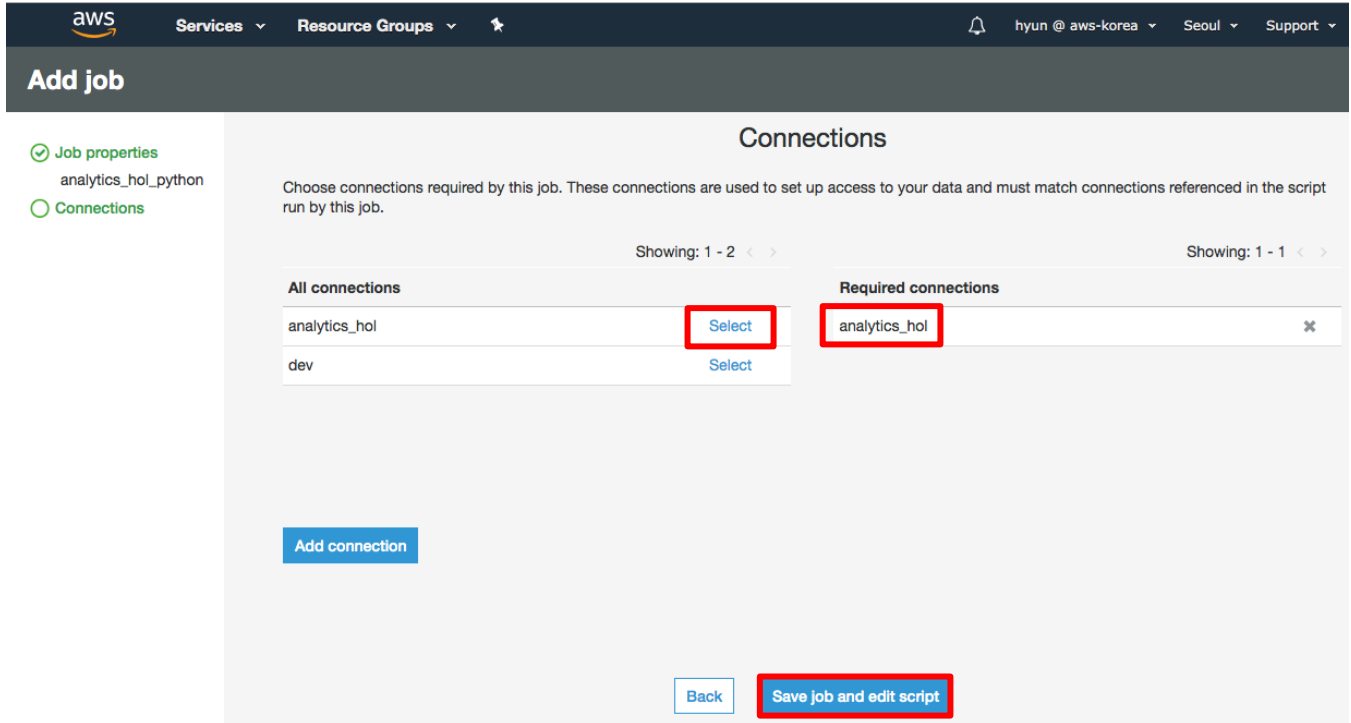
Job parameters

Key	Value
Type key...	Type value...

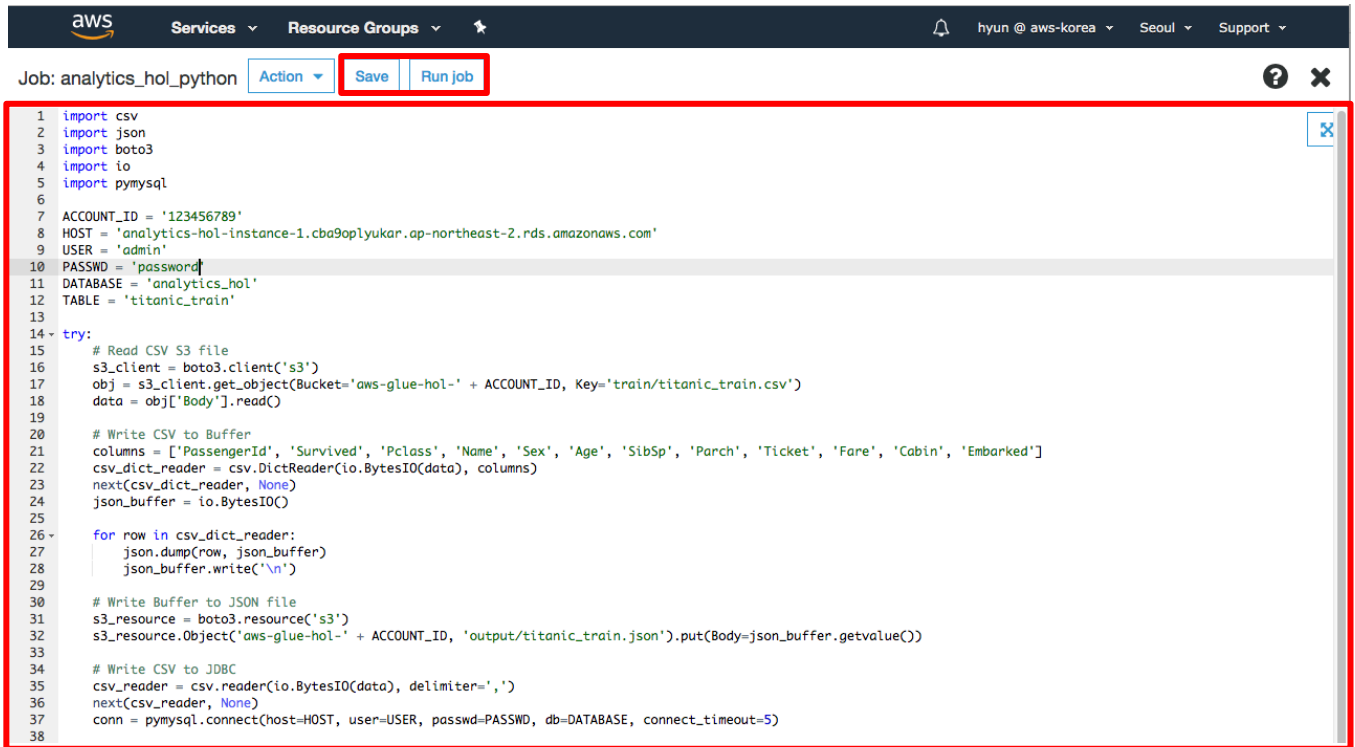
► Catalog options (optional)

Next

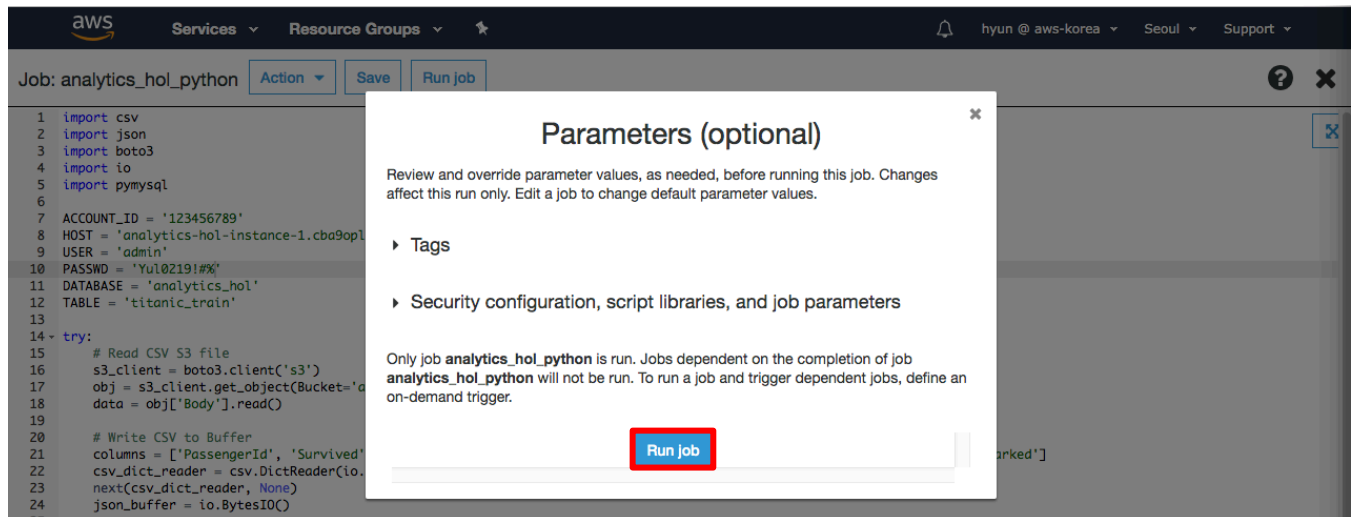
7-52. Glue Python Shell Job 에서 RDS 에 접속하기 위해 analytics_hol connection select 버튼을 클릭하고 Save job and edit script 버튼을 클릭합니다. analytics_hol 을 선택하면 Required connections 에 아래 그림과 같이 표시되는 것을 확인할 수 있습니다.



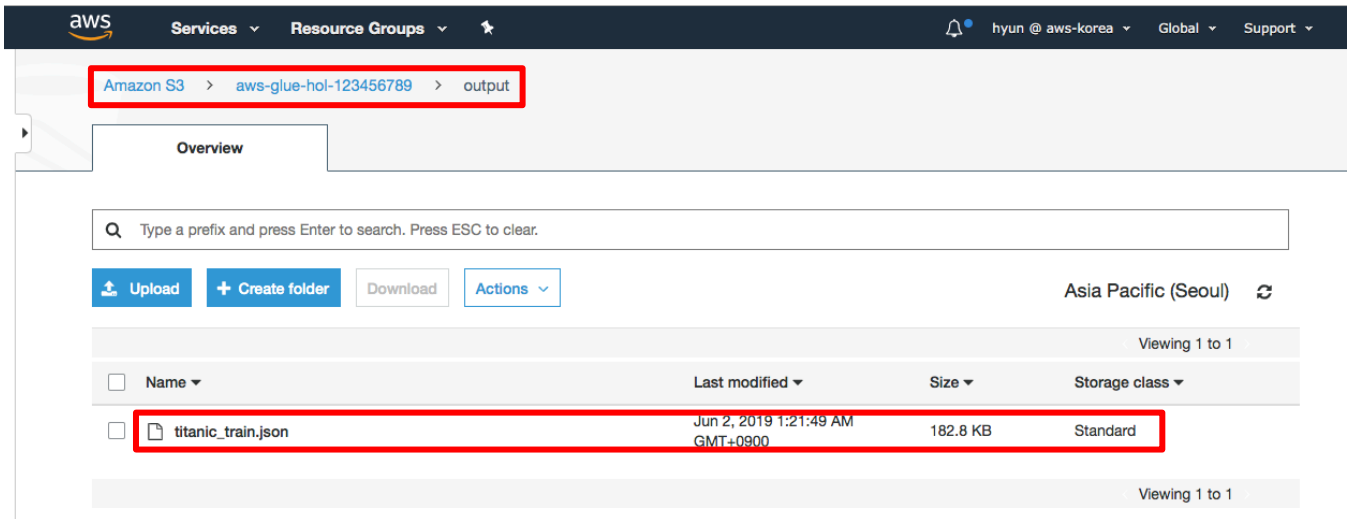
7-53. 05.python_shell.ipynb 노트북의 세 번째 코드 Cell 을 복사해서 Script Edit 창에 붙여넣고 ACCOUNT_ID, HOST, USER, PASSWD 를 입력한 후 Save 버튼과 Run job 버튼을 순서대로 클릭합니다.



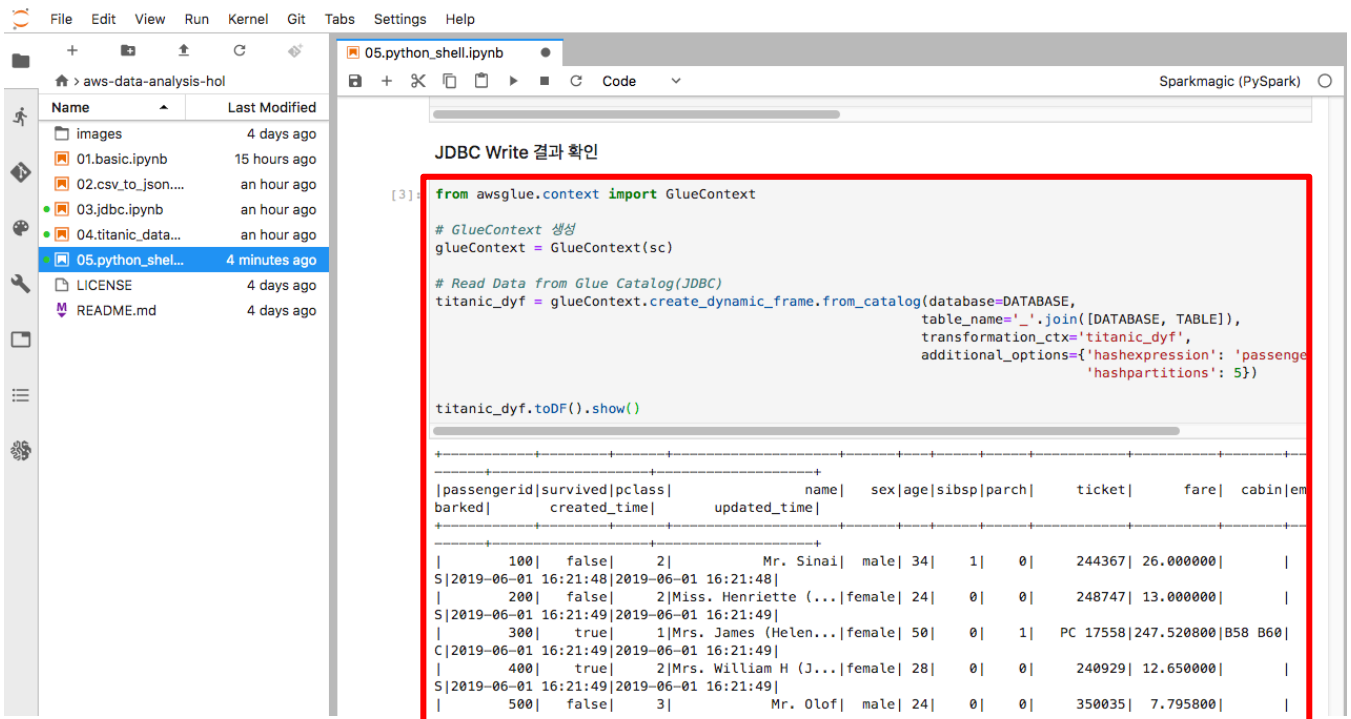
7-54. 다시 Run job 버튼을 클릭합니다.



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.



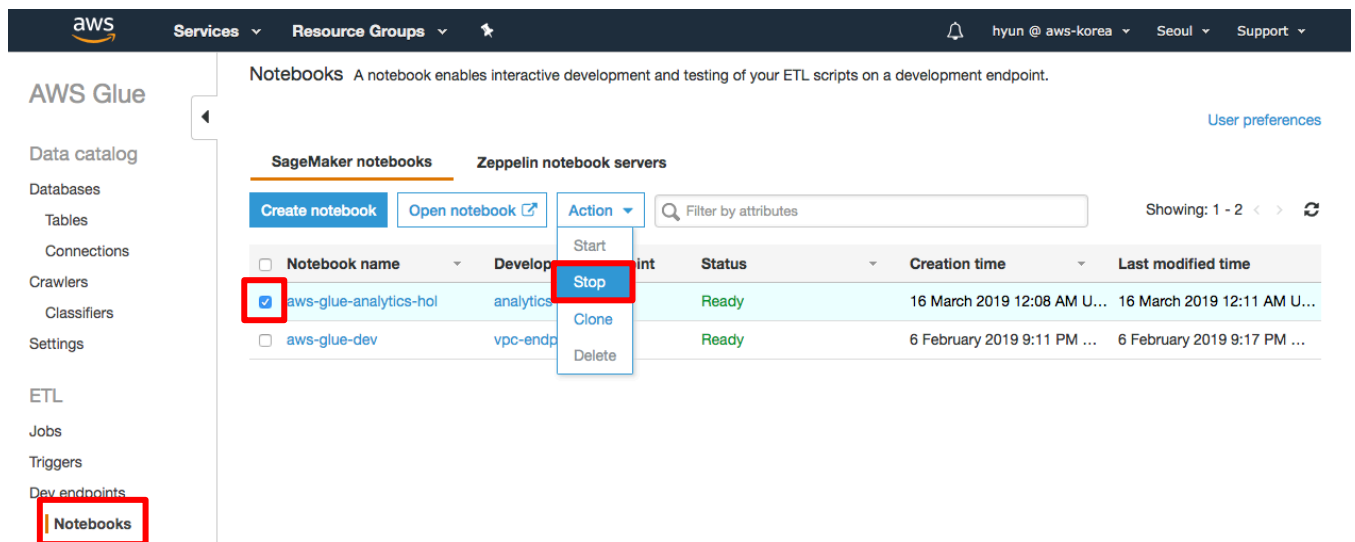
8. Closing

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

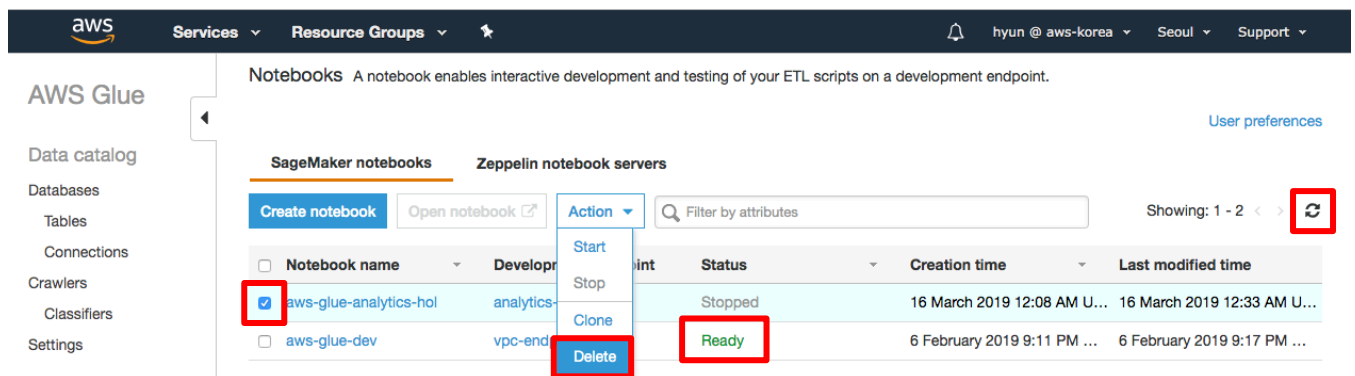
Glue 리소스 삭제

SageMaker notebook 을 먼저 삭제한 후 Dev endpoint 를 삭제합니다.

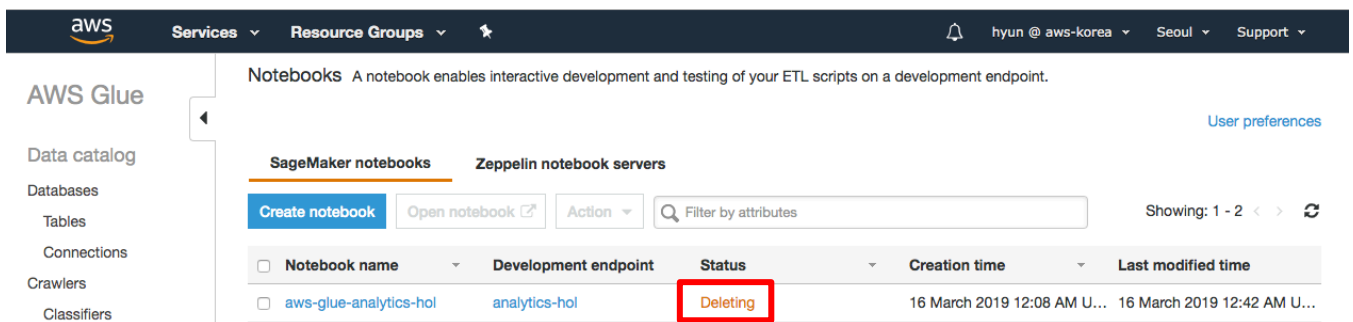
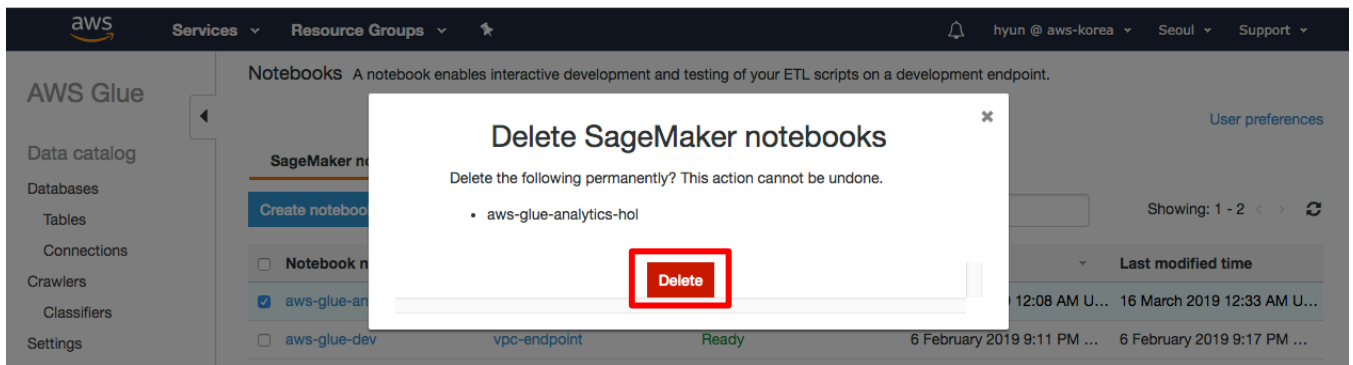
8-1. Glue 콘솔의 Dev endpoints->Notebooks 를 클릭하여 이동합니다. aws-glue-analytics-hol 노트북을 체크하고 Action 에서 Stop 을 선택합니다.



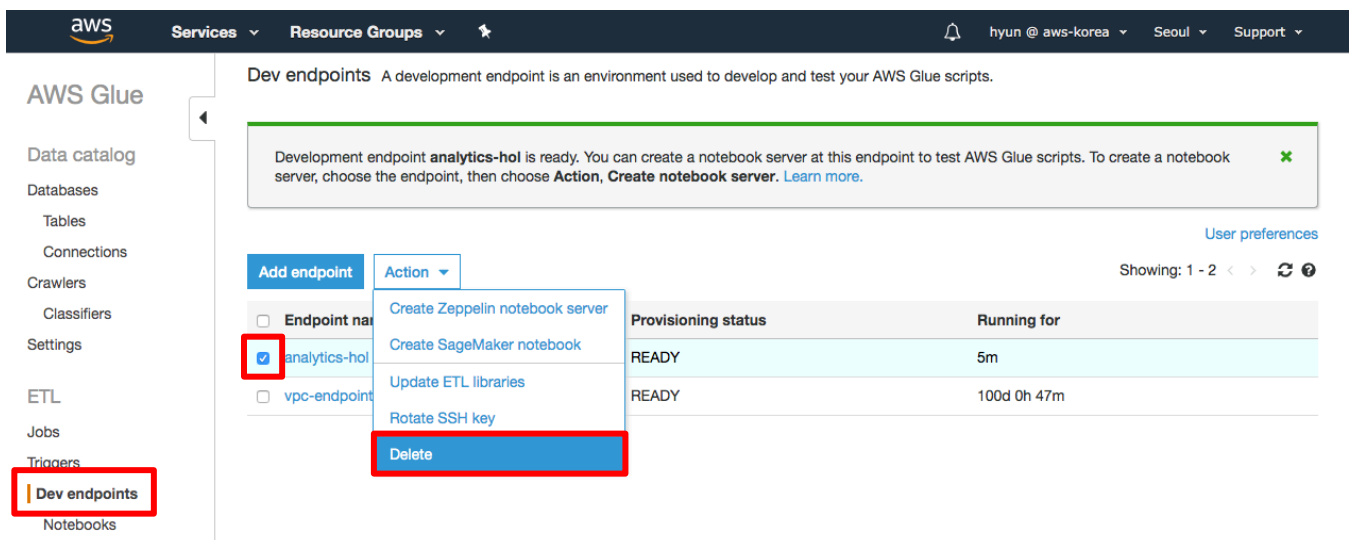
8-2. 리프레쉬 버튼을 눌러 Status 가 Stopping 에서 Stopped 으로 변경된 것을 확인한 후, Action 에서 Delete 를 선택합니다.



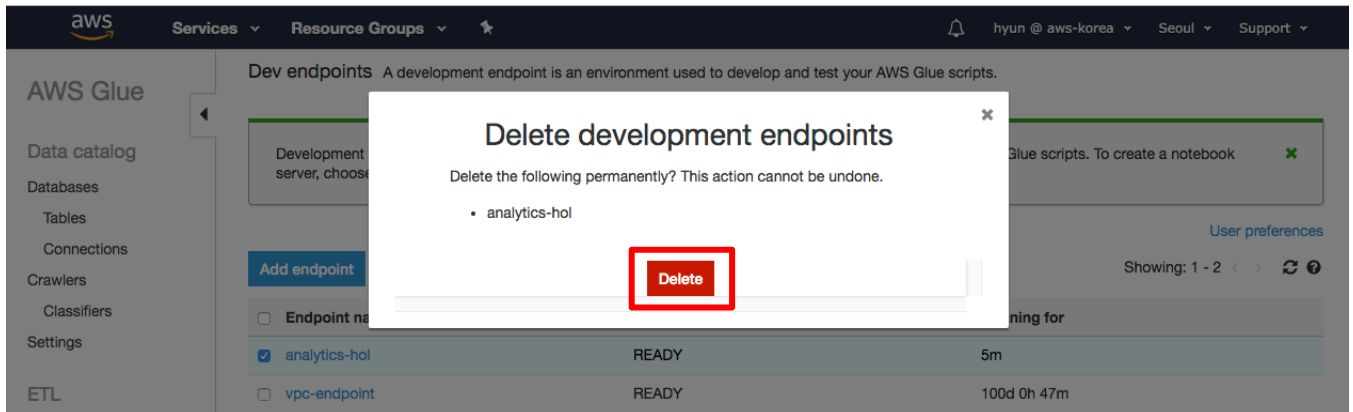
8-3. Delete 버튼을 클릭하면 aws-glue-analytics-hol 노트북 삭제가 완료됩니다.



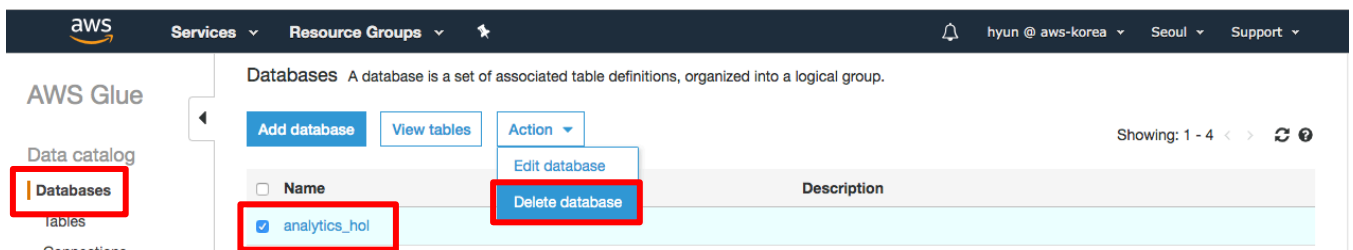
8-4. Glue 콘솔 Dev endpoints 를 클릭해서 이동한 후 analytics-hol 엔드포인트를 선택하고 Action 에서 Delete 를 선택합니다.



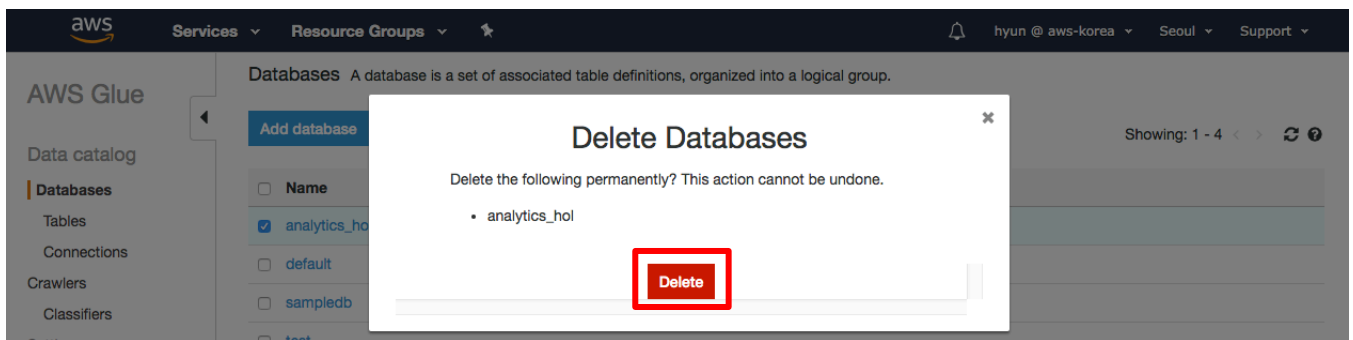
8-5. Delete 버튼을 클릭하면 analytics-hol 엔드포인트 삭제가 완료됩니다.



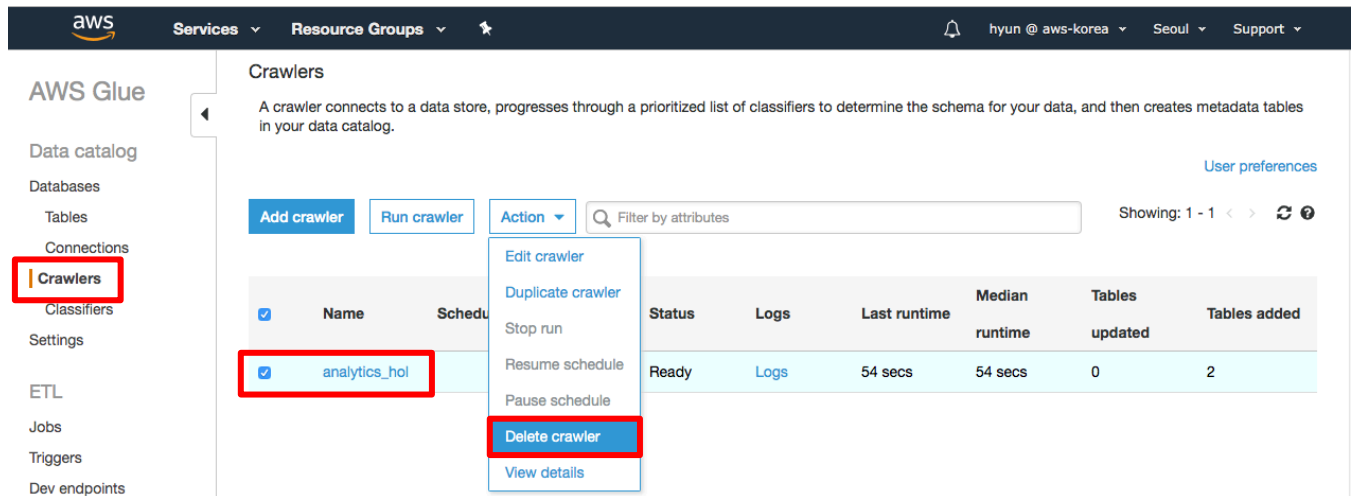
8-6. Glue 콘솔에서 Databases 를 클릭합니다. analytics_hol 데이터베이스를 선택하고 Action 에서 Delete database 를 선택합니다.



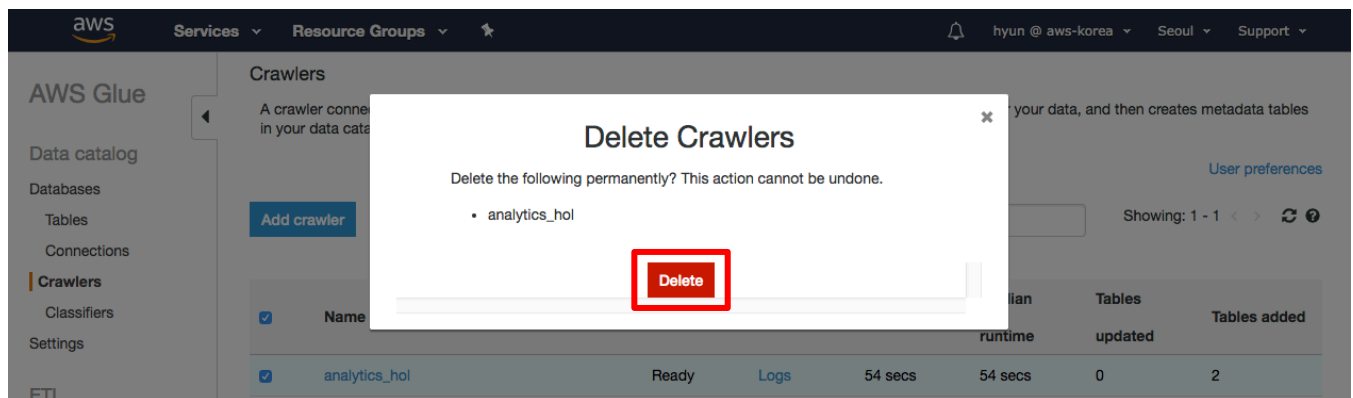
8-7. Delete 버튼을 눌러 삭제합니다.



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

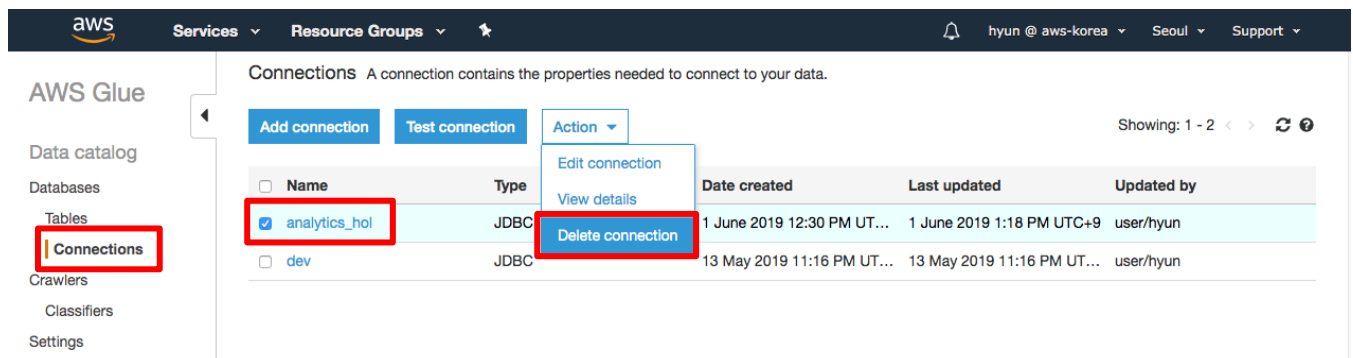


8-9. Delete 버튼을 클릭하여 삭제합니다.

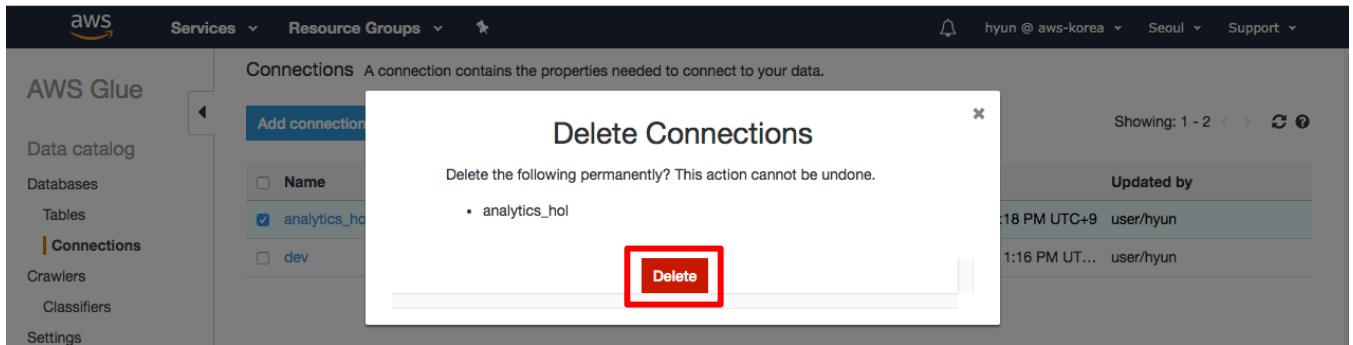


8-10. analytics_hol_jdbc 크롤러도 동일한 과정으로 삭제합니다.

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

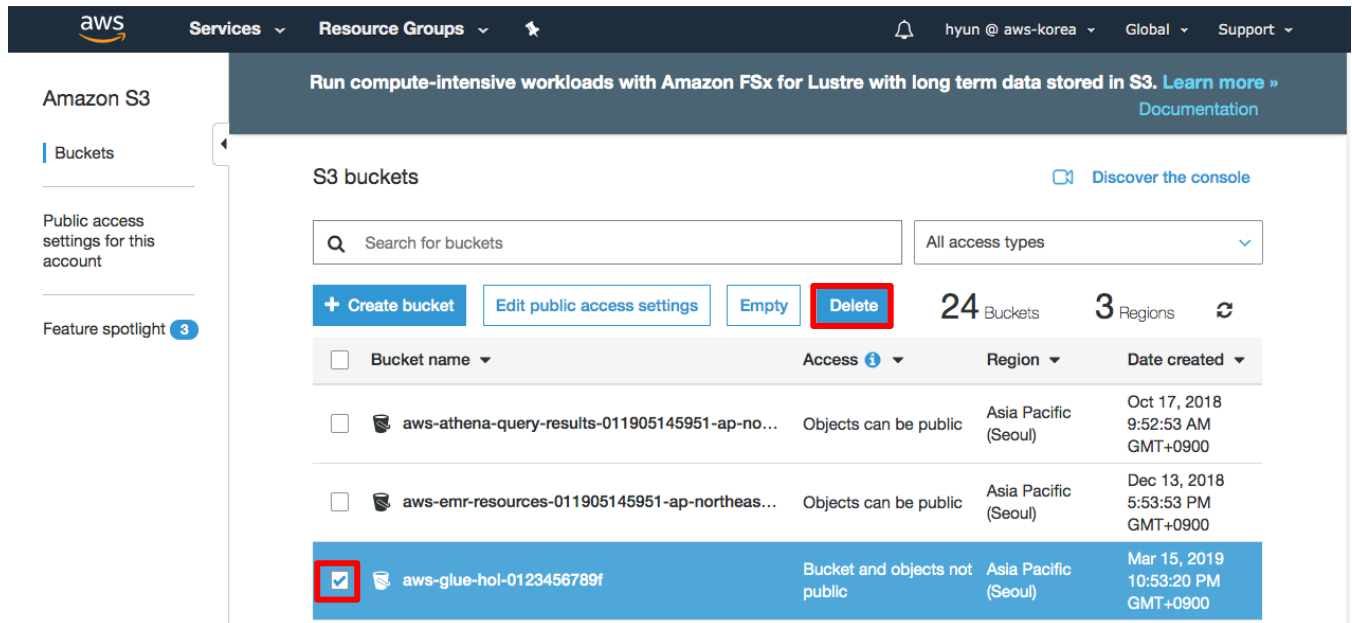


8-12. Delete 버튼을 클릭하여 삭제합니다.

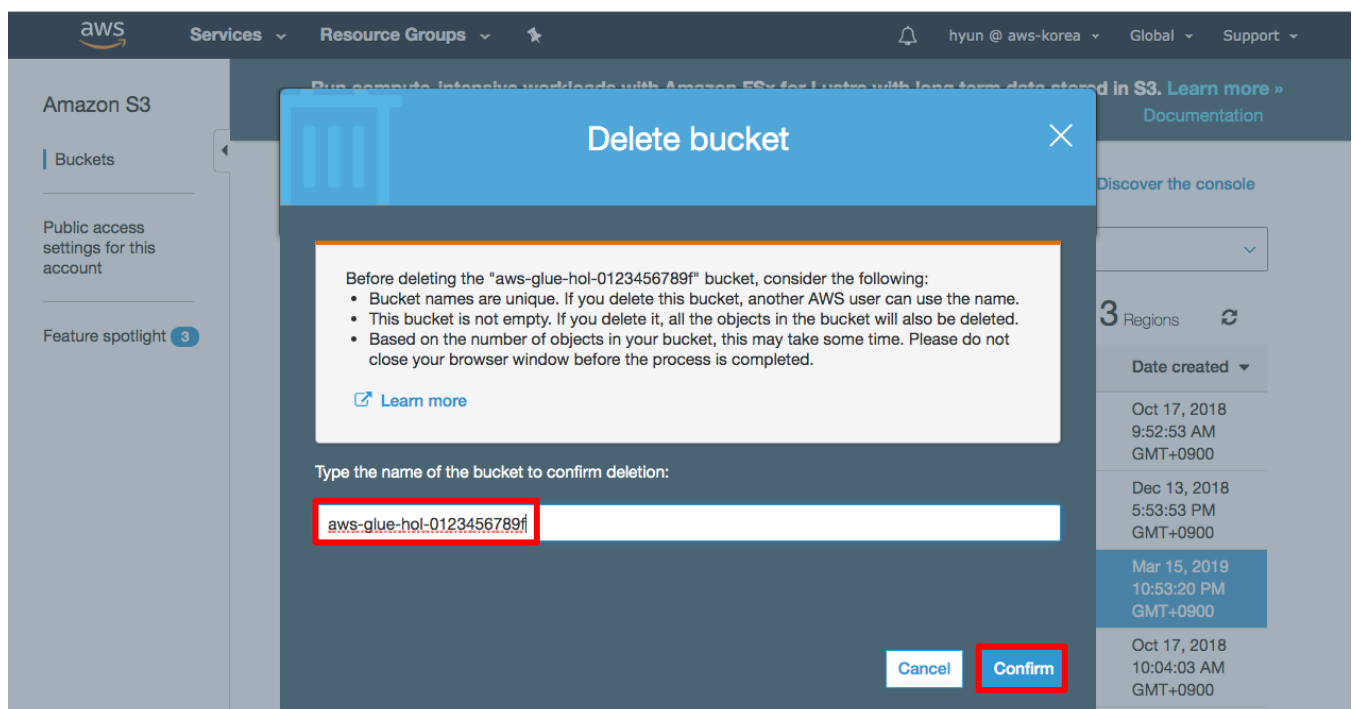


S3 버킷 삭제

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

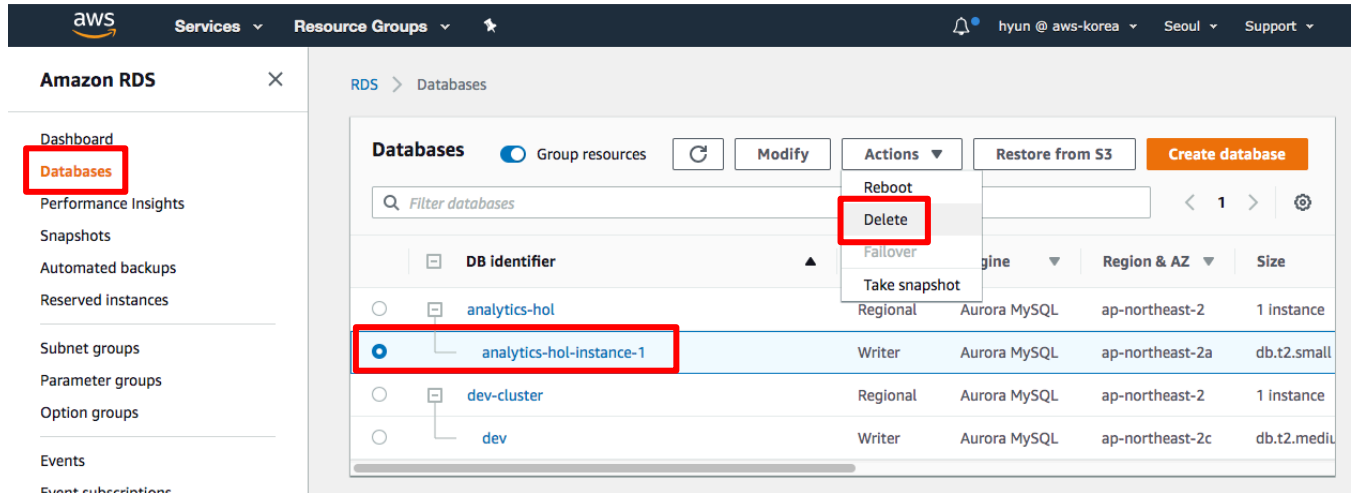


8-14. aws-glue-hol-[account-id]을 한번 더 입력한 후 Confirm 버튼을 클릭하면 버킷 삭제가 완료됩니다.

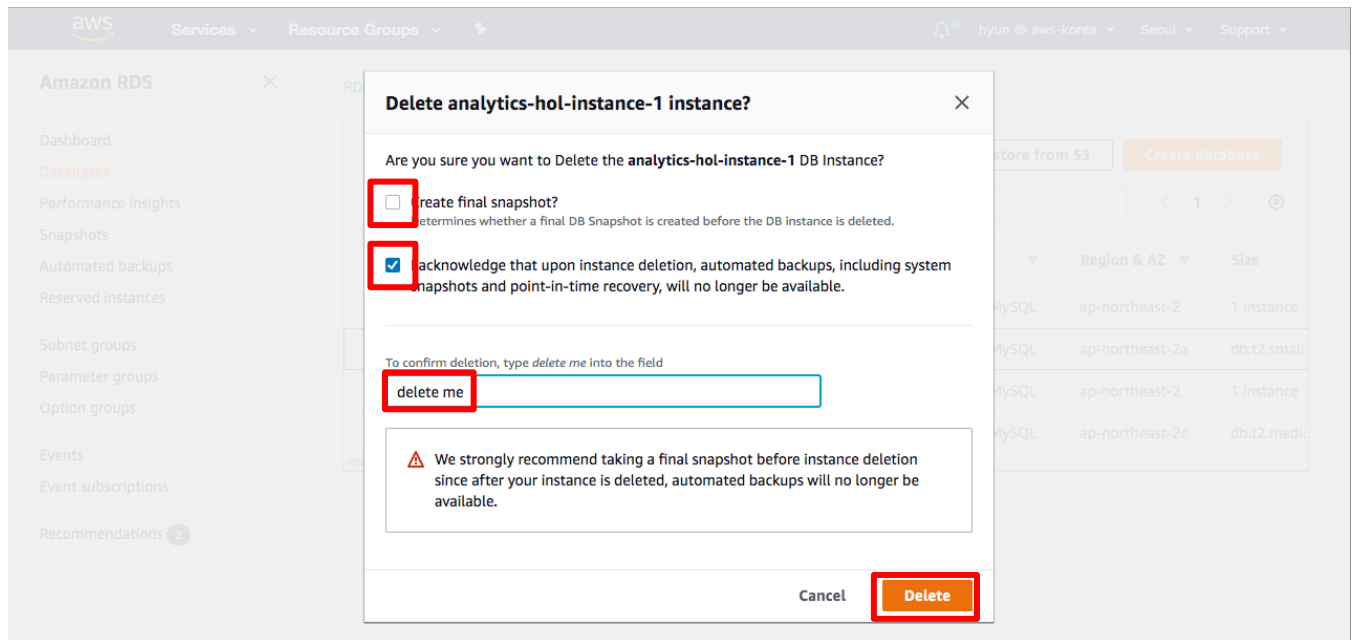


RDS 삭제

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



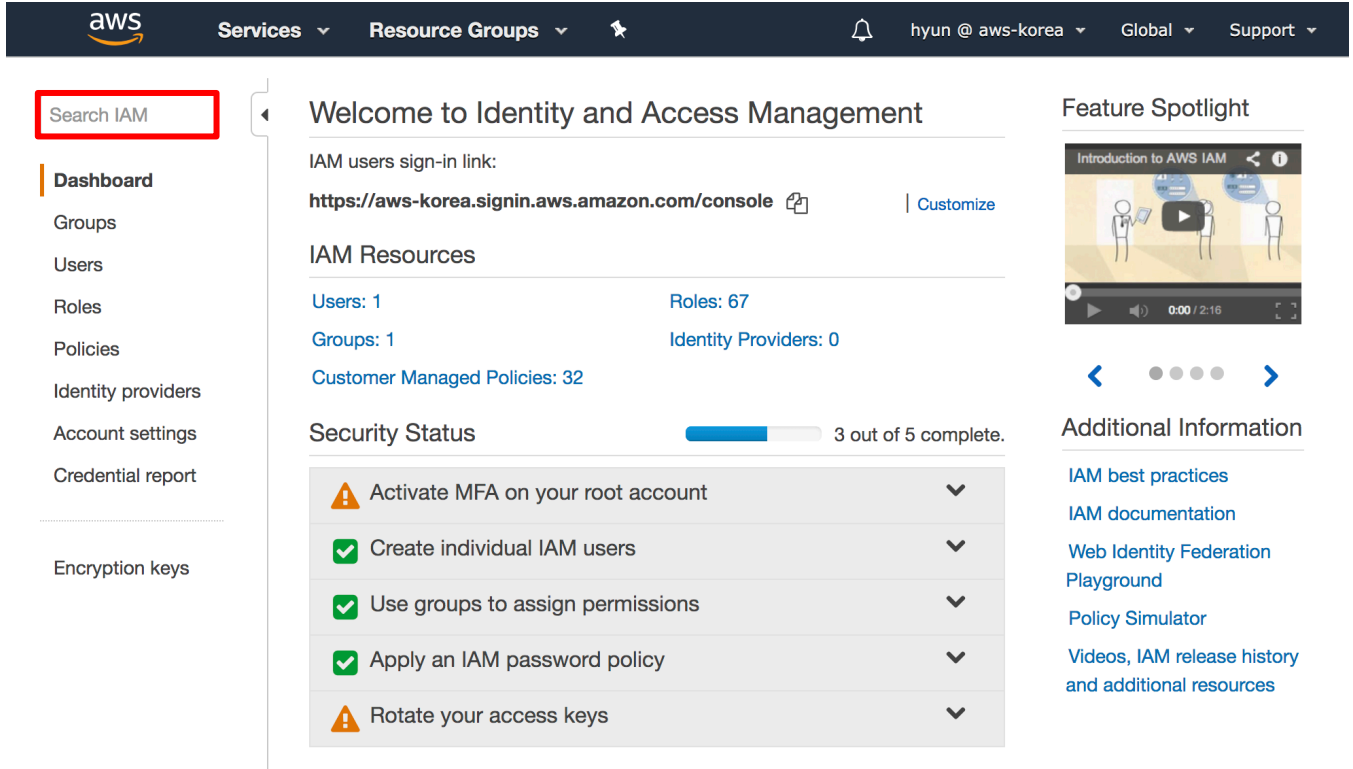
8-16. Create final snapshot 체크박스를 해제하고, I acknowledge 체크박스를 체크합니다. 삭제를 하기 위해 delete me 라고 confirm 메시지를 입력한 후 Delete 버튼을 클릭합니다.



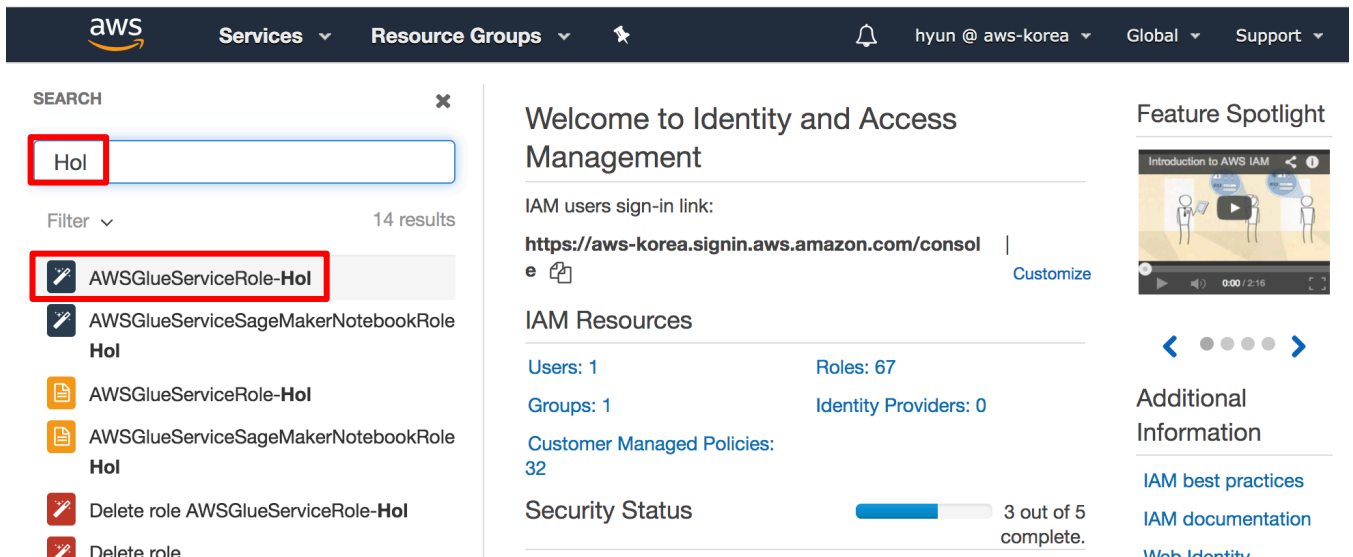
8-17. 삭제되는데까지 약 5 분정도 소요됩니다. Refresh 버튼을 눌러 삭제가 완료된 것을 확인합니다.

IAM Role 삭제 (Optional)

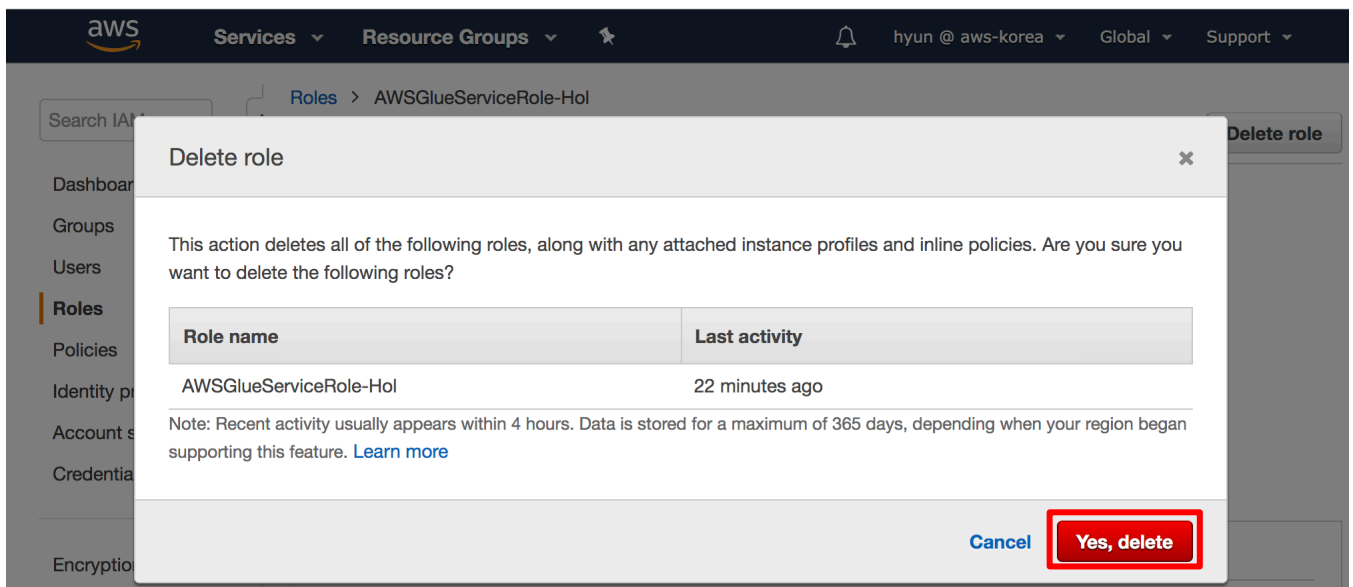
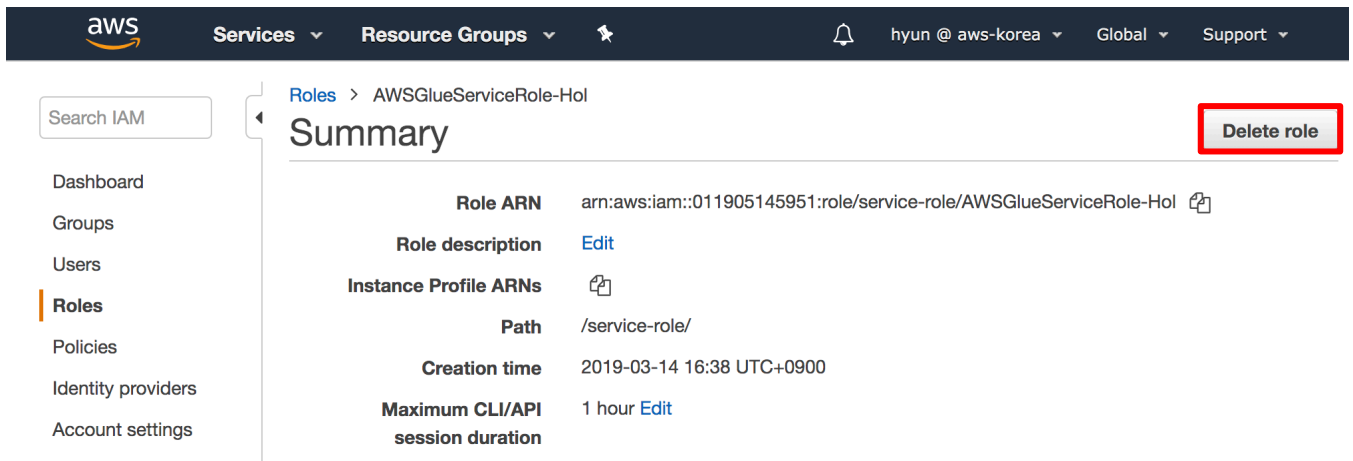
8-18. IAM 콘솔로 이동합니다. Search IAM 텍스트 박스에 Hol 을 입력합니다.



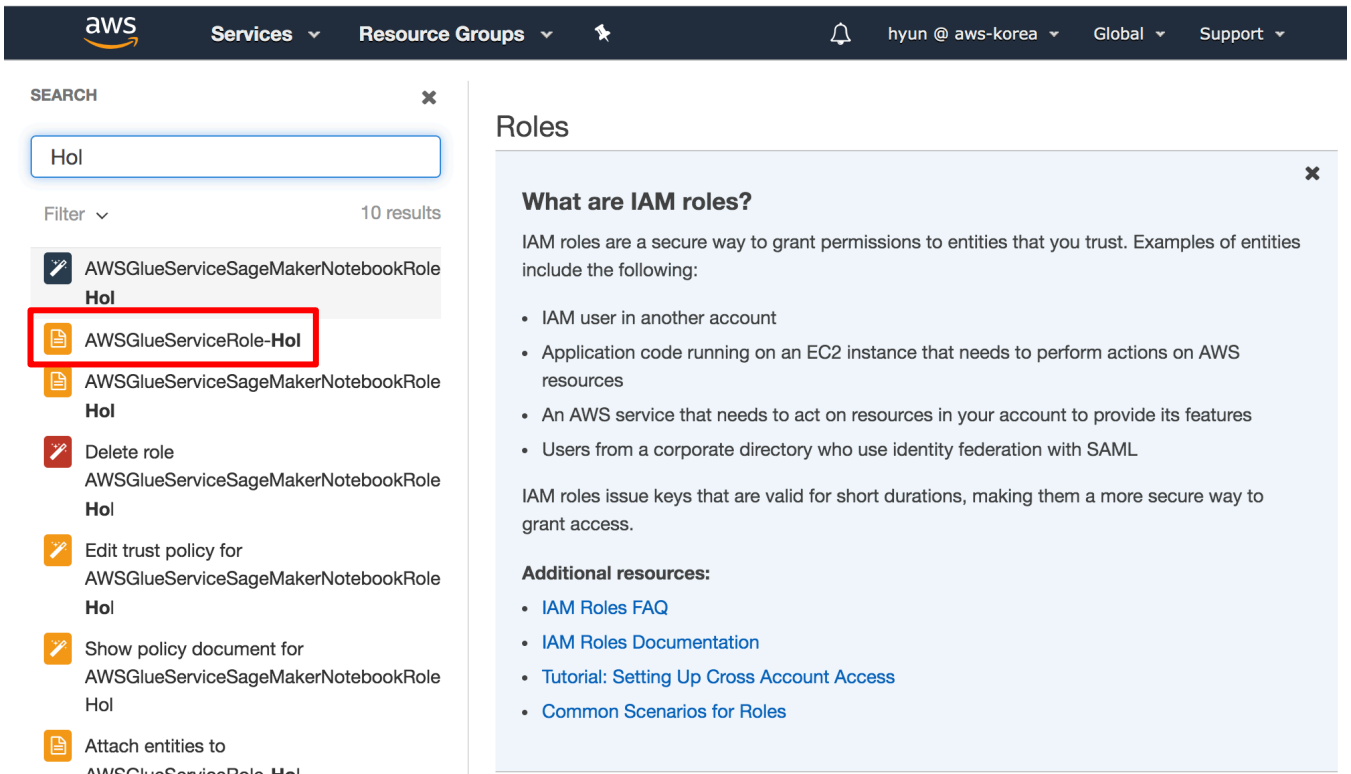
8-19. IAM 콘솔로 이동합니다. Search IAM 텍스트 박스에 Hol 을 입력하고 검색 결과 중 AWSGlueServiceRoleHol 을 선택합니다.



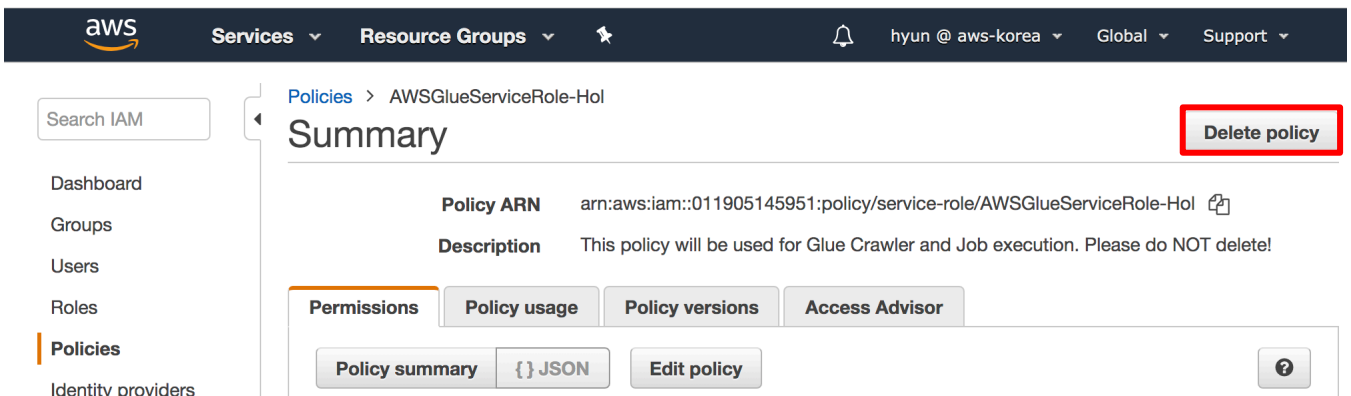
8-20. 오른쪽 상단에 있는 Delete role 버튼을 클릭하여 선택한 Role 을 삭제합니다.

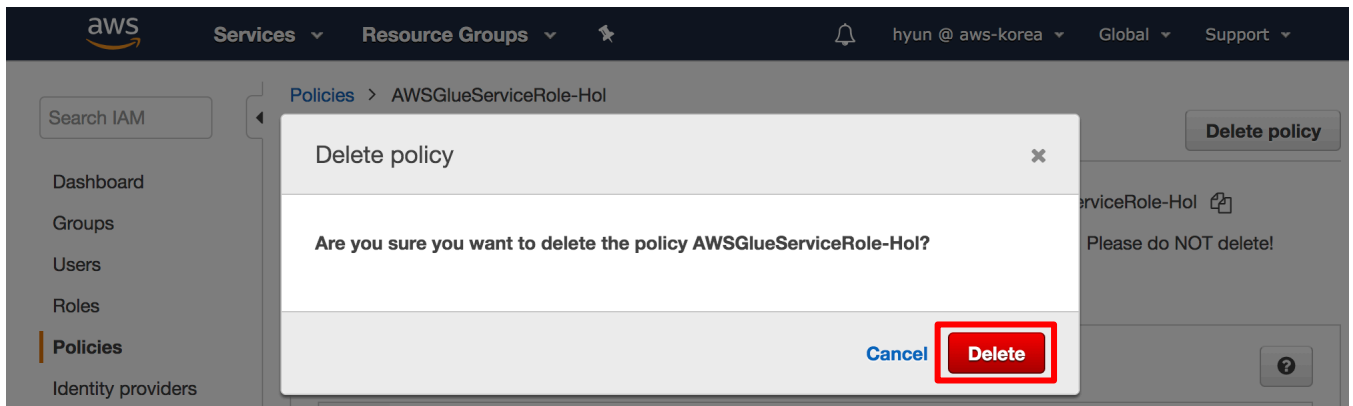


8-21. 다시 한 번 Hol 검색을 하고 AWSGlueServiceRole-Hol 을 선택합니다.



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



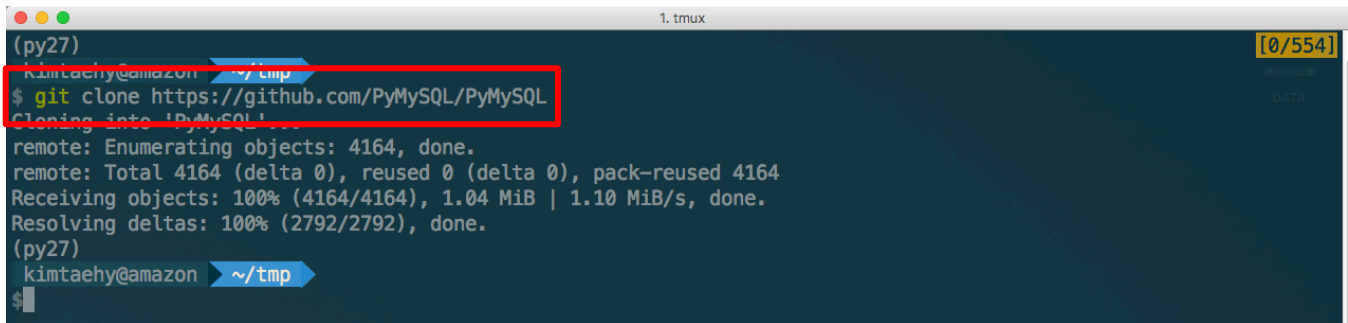


8-23. AWSGlueServiceSageMakerNotebookRole-Hol 에 대해서도 Role 과 Policy 를 같은 방법으로 삭제합니다.

9. Appendix A

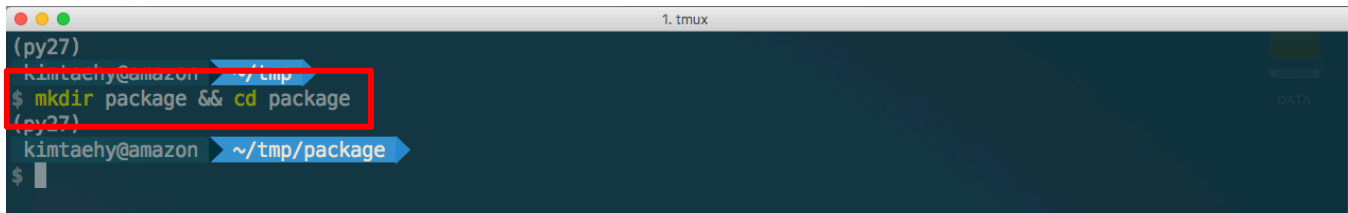
Glue Python Shell 은 Glue 에 설치되어있지 않은 외부 라이브러리를 사용할 수 있도록 지원하고 있습니다. Lab 실습 중 RDS 에 접근하기 위해 pymysql 을 egg 파일로 S3 에 업로드하고 Glue Script 에서 사용했습니다. 이번 장에서는 외부 라이브러리를 사용하기 위한 egg 파일을 생성하는 방법에 대해서 설명 드리도록 하겠습니다.

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



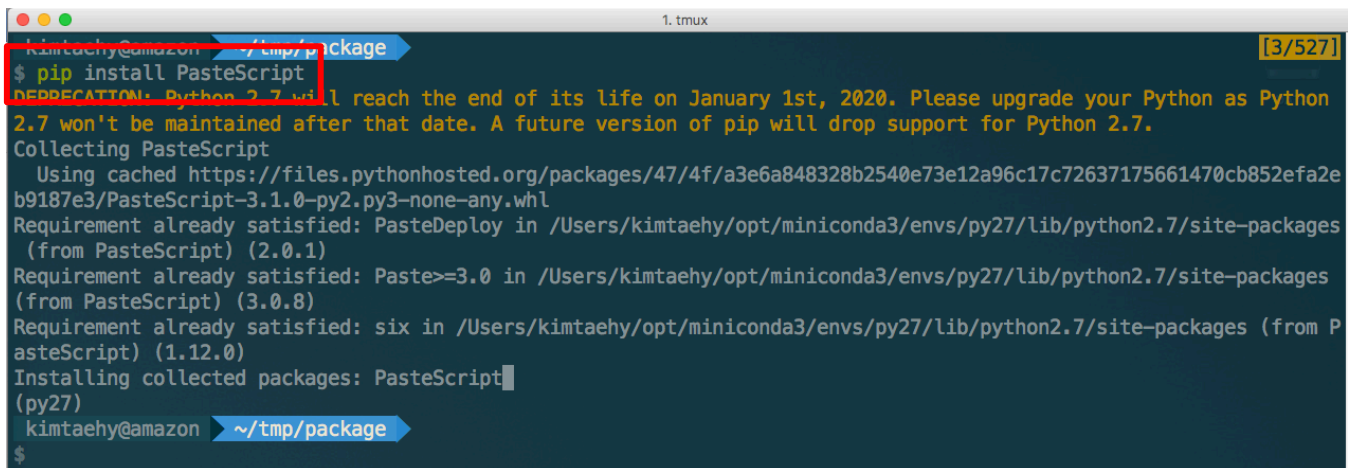
```
(py27) kimtaehy@amazon ~/tmp
$ git clone https://github.com/PyMySQL/PyMySQL
Cloning into 'PyMySQL'...
remote: Enumerating objects: 4164, done.
remote: Total 4164 (delta 0), reused 0 (delta 0), pack-reused 4164
Receiving objects: 100% (4164/4164), 1.04 MiB | 1.10 MiB/s, done.
Resolving deltas: 100% (2792/2792), done.
(py27) kimtaehy@amazon ~/tmp
$
```

9-2. package 디렉토리를 생성하고 이동합니다.



```
(py27) kimtaehy@amazon ~/tmp
$ mkdir package && cd package
(py27) kimtaehy@amazon ~/tmp/package
$
```

9-3. PasteScript 를 설치합니다.



```
(py27) kimtaehy@amazon ~/tmp/package
$ pip install PasteScript
DEPRECATION: Python 2.7 will reach the end of its life on January 1st, 2020. Please upgrade your Python as Python 2.7 won't be maintained after that date. A future version of pip will drop support for Python 2.7.
Collecting PasteScript
  Using cached https://files.pythonhosted.org/packages/47/4f/a3e6a848328b2540e73e12a96c17c72637175661470cb852efa2eb9187e3/PasteScript-3.1.0-py2.py3-none-any.whl
Requirement already satisfied: PasteDeploy in /Users/kimtaehy/opt/miniconda3/envs/py27/lib/python2.7/site-packages (from PasteScript) (2.0.1)
Requirement already satisfied: Paste>=3.0 in /Users/kimtaehy/opt/miniconda3/envs/py27/lib/python2.7/site-packages (from PasteScript) (3.0.8)
Requirement already satisfied: six in /Users/kimtaehy/opt/miniconda3/envs/py27/lib/python2.7/site-packages (from PasteScript) (1.12.0)
Installing collected packages: PasteScript
(py27) kimtaehy@amazon ~/tmp/package
$
```



9-4. 디렉토리 레이아웃을 생성합니다.

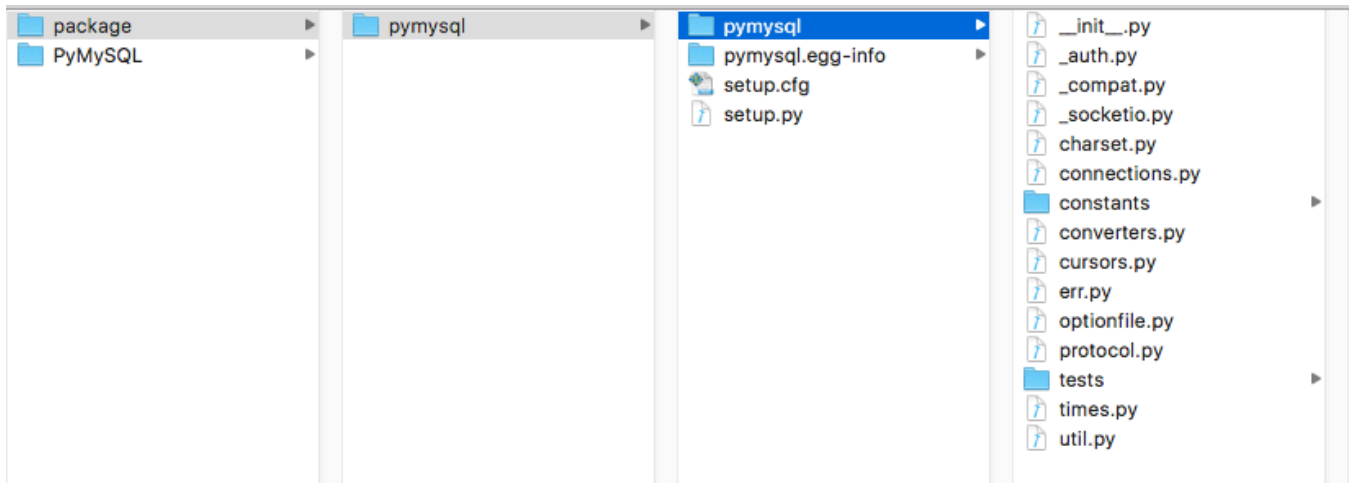
```
(py27)
kimtaehy@amazon ~/tmp/package [0/599]
$ paster create -t basic_package pymysql
Selected and implied templates:
PasteScript#basic_package A basic setuptools-enabled package

Variables:
egg:      pymysql
package:  pymysql
project:  pymysql
Enter version (Version (like 0.1)) ['']:
Enter description (One-line description of the package) ['']:
Enter long_description (Multi-line description (in reST)) ['']:
Enter keywords (Space-separated keywords/tags) ['']:
Enter author (Author name) ['']:
Enter author_email (Author email) ['']:
Enter url (URL of homepage) ['']:
Enter license_name (License name) ['']:
Enter zip_safe (True/False: if the package can be distributed as a .zip file) [False]:
Creating template basic_package
Creating directory ./pymysql
  Recursing into +package+
    Creating ./pymysql/pymysql/
      Copying __init__.py to ./pymysql/pymysql/__init__.py
      Copying setup.cfg to ./pymysql/setup.cfg
      Copying setup.py_tmpl to ./pymysql/setup.py
Running /Users/kimtaehy/opt/miniconda3/envs/py27/bin/python setup.py egg_info
(py27)
kimtaehy@amazon ~/tmp/package 6s
```

9-5. git clone 받은 라이브러리를 복사합니다.

```
(py27)
kimtaehy@amazon ~/tmp/package
$ cp -pr ../PyMySQL/pymysql ./pymysql/
(py27)
kimtaehy@amazon ~/tmp/package
$
```

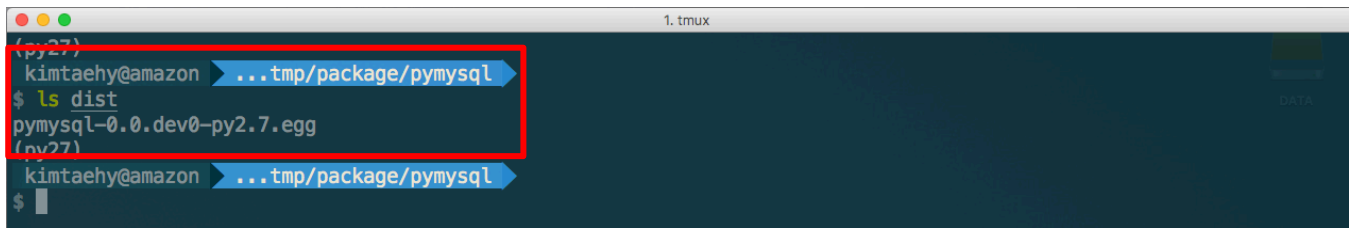
9-6. 최종 생성된 디렉토리 구조는 다음과 같습니다.



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

```
kimtaehy@amazon > ...tmp/package/pymysql
$ ls
pymysql      pymysql.egg-info  setup.cfg      setup.py
(py27)
kimtaehy@amazon > ...tmp/package/pymysql
$ python setup.py bdist_egg
running bdist_egg
running egg_info
writing pymysql.egg-info/PKG-INFO
writing top-level names to pymysql.egg-info/top_level.txt
writing dependency_links to pymysql.egg-info/dependency_links.txt
writing entry points to pymysql.egg-info/entry_points.txt
reading manifest file 'pymysql.egg-info/SOURCES.txt'
writing manifest file 'pymysql.egg-info/SOURCES.txt'
installing library code to build/bdist.macosx-10.6-x86_64/egg
running install_lib
running build_py
creating build
creating build/lib
creating build/lib/pymysql
copying pymysql/_init_.py -> build/lib/pymysql
copying pymysql/_auth.py -> build/lib/pymysql
copying pymysql/_compat.py -> build/lib/pymysql
copying pymysql/_socketio.py -> build/lib/pymysql
copying pymysql/charset.py -> build/lib/pymysql
copying pymysql/connections.py -> build/lib/pymysql
copying pymysql/converters.py -> build/lib/pymysql
copying pymysql/cursors.py -> build/lib/pymysql
copying pymysql/err.py -> build/lib/pymysql
```

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



```
(py27)
kimtaehy@amazon > ..tmp/package/pymysql
$ ls dist
pymysql-0.0.dev0-py2.7.egg
(py27)
kimtaehy@amazon > ..tmp/package/pymysql
$
```