SKT Practical Class Schedule

<https://docs.google.com/document/d/1R0RVyrHxQ9cle6Oo8pCLuNpCQ9DcvGCsPcibvlNzbYc/edit?usp=sharing>

공유 링크: <https://goo.gl/UibNcT>

====================================================================

**Day3:**

* **Every 50 min + 10 min (휴식): 예) 9:50am, 10:50am, 1:50pm, …**
* **Linkedin info for your name**

**9am - 9:30m:** Day 2 리뷰 Q&A; 팀별 테이블, 투토리알, 발표슬라이드 준비 완성되어있는지 다시체크;

* 각 팀별 최소 3개의 Yelp Data insights 분석 및 시각화
* 각 팀별 github 만듬, URL link for Linkedin
* 스파크, 제플린 소개
* Spark codes: <https://goo.gl/VL7iNS>
* Spark SQL Tutorial: <https://spark.apache.org/docs/latest/sql-programming-guide.html>

**9:30am - 10:30am:** 하이브 테이블을 Spark SQL 로 데이타 분석 및 시각화하는것 개요 소개 및 실행

* day2의 Q1, 2, 3, 4

**9:50am - 11:45am:** 하이브 테이블을 Spark SQL 로 데이타 분석 및 시각화

* day2의 Q5, 6, 7, 8, 9

**1:00pm - 2:50pm:** 하이브 테이블을 Spark SQL 로 데이타 분석 및 시각화 (Cont’d)

* 팀별 3개 데이터 인사이트 Spark SQL (HiveQL) 및 그래프를 가지고 발표 준비
* 발표:

1. 서론

2. 어떤 데이타 insight을 왜, 비지니스적으로 왜 중요한가 (10%)

3. 코드 및 그래프: 최소 3 (40%, 40%)

4. 특이점: 스피드, 코딩등에서 (10%)

5. 결론

**3:00pm - 3:50pm:** 팀 1, 2, 3 발표 (각팀 약 15분 발표및 Q&A)

**4:00pm - 4:50pm:** 팀 4, 5, 6 발표

**5:00pm - 5:50pm:** 팀별 투토리알, 프리젠테이션 슬라이드를 팀 github에 업로드

* 클래스 종강전 퇴근전에 github url 다음의 표에 링크

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **팀 번호** | **시스템 어드민** | **개발분석** | **투토리알&발표슬라이드 작성** | **github url** |
| **1** | 김정미 | [양중근](https://www.linkedin.com/in/joongkeun-yang-ba790086) | [정원석](https://www.linkedin.com/in/won-sug-jung-14b88250/) | <https://github.com/gardenstone/1> |
| **2** | [송선영](http://linkedin.com/in/sunyoung-song-79b690142) | [김장현](https://www.linkedin.com/in/janghyun-kim-816bbb27/) | 김형석 | <https://github.com/aizoakjh/sk-bigdataeng-002> |
| **3** | 신행철 | 이소연 | 백규진 |  |
| **4 (4명)** | [남병출](https://www.linkedin.com/in/byungchul-nam-b843b1151/) | 안세환 | [배형민](https://www.linkedin.com/in/hyungmin-bae-2a989a10/), [홍남식](https://www.linkedin.com/in/%EB%82%A8%EC%8B%9D-%ED%99%8D-320013167/) | <https://github.com/baehyungmin/DAL>  https://docs.google.com/document/d/1yRJbnKg3l64PZB07rYYbWOKCEN6jHRNy0a-83yRwi0I/edit |
| **5 (4명)** | 유건우 | 김지혁 | 이유진, [이대식](http://linkedin.com/in/daesik-lee-bb92a721), | https://docs.google.com/presentation/d/1-SXK4Eutqr6CFNY7ZlYXZYEpVzR4tuNrp7AXHLMYIPA/edit?usp=sharing |
| **6 (4명)** | 김만수 | 민은영 | [황종윤](https://www.linkedin.com/in/jongyun-hwang-3783a141/), [김상진](https://www.linkedin.com/in/sangjin-kim-51489b166) | https://github.com/mansoov/group6 |

====================================================================

**Day2:**

* **Every 50 min + 10 min (휴식): 예) 9:50am, 10:50am, 1:50pm, …**
* **Linkedin info for your name**

**9am - 9:30m:** Day 1 리뷰 Q&A; 팀별 테이블, 투토리알, 발표슬라이드 준비 완성되어있는지 다시체크;

**9:30am - 10:30am:** 하이브 테이블로 데이타 분석

1. **Business TABLE**

SELECT count(business\_id) FROM business4;

SELECT count(distinct business\_id) FROM business4;

**174567**

* Create **EXPLODED** table with flattened categories  
  Since our file contains an array of categories we need to flatten those categories in order to be able to query them easily. We use LATERAL VIEW explode function for the column **categories** and alias it as **cat\_exploded**

CREATE TABLE exploded

ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe' STORED AS TEXTFILE

LOCATION '/user/dalgual/yelp/exploded' AS

SELECT \* FROM business4 LATERAL VIEW explode(categories) c AS cat\_exploded;

* **Q1.1** List the values of the columns business\_id, categories, cat\_exploded from the table EXPLODED with 10 records selected:

SELECT business\_id, categories, cat\_exploded from EXPLODED LIMIT 10;

SELECT count(business\_id) FROM exploded;

667527

SELECT count(distinct business\_id) FROM exploded;

174067

// Consider NULL value with outer explode

CREATE TABLE exploded2

AS

SELECT \* FROM business4 LATERAL VIEW OUTER explode(categories) c AS cat\_exploded;

SELECT count(distinct business\_id) FROM exploded2;

174567

* **Q1.2** Find total number of businesses in the data set:

SELECT count(business\_id) FROM business4;

**2. Restaurant TABLE**

* Create restaurant table from the table EXPLODED with the column cat\_exploded="Restaurants":

// Table in Json

CREATE TABLE IF NOT EXISTS restaurants

ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'

STORED AS TEXTFILE AS

SELECT \* FROM exploded WHERE cat\_exploded="Restaurants";

// Table in text

CREATE TABLE IF NOT EXISTS restaurants2

AS

SELECT \* FROM exploded WHERE cat\_exploded="Restaurants";

* **Q2.1.** List 5 records with the columns **name, review\_count, stars, cat\_exploded category** from your restaurants table created -  **cat\_exploded** column is renamed as **category** :

SELECT name, review\_count, stars, cat\_exploded category FROM restaurants LIMIT 5;

* **Q2.2.** query columns [name and nested object (attributes > ambience > romantic) of restaurants table:   
  a) To select nested columns i as following: parent.child.grandchild

SELECT name, attributes.ambience.romantic FROM restaurants LIMIT 5;

* **Q2.3** find the number of restaurants in the dataset

SELECT count(distinct business\_id) FROM restaurants;

**54618**

SELECT count(business\_id) FROM restaurants;

**54618**

<https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-web-interfaces.html>

**10:30am - 11:45am:** 하이브 데이타 분석 (힌트제공)

**3. Review TABLE**

* Create Review\_Filtered Table
  + using JOIN for **review** and **restaurants** with the same business\_id
  + NOTE: In order to improve the performance, do you prefer **review** JOIN **restaurant** or the other way?
* **Q3.1.** find the number of restaurant reviews from Review\_Filtered Table

**4. Users TABLE**

* User table 에서 find the number of all distinct users in dataset:

**5. Users/Elite TABLE**

* **Q5.1.** select 4 records with the COLUMN **elite** from **USERS** table, which is generated from user.json to see elite users: NOTE: table should be USERS not USER
* **Q5.2.** Create **Elite\_Users** Table by exploding **elite** column as **elite\_year** column using LATERAL VIEW in order to separate elite users: For example, [2016, 2017, 2018] => 3 exploded records as follows:

2016

2017

2018



**11:45am - 1pm: Lunch  
  
1pm - 2:00pm: 하이브 데이타 분석 (계속)  
  
2:00pm - 3:00pm: 하이브 데이타 분석 (힌트제공)**

**3pm - 5:00pm:** 데이타 분석 (힌트) 및 시각화 (Hue, 엑셀 파워뷰 설치)

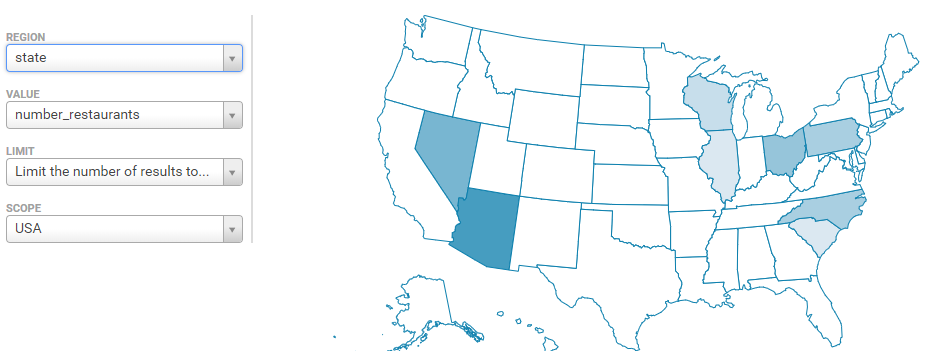
**6. Create Tip Table**

**7. 데이터 분석 심층 및 시각화**

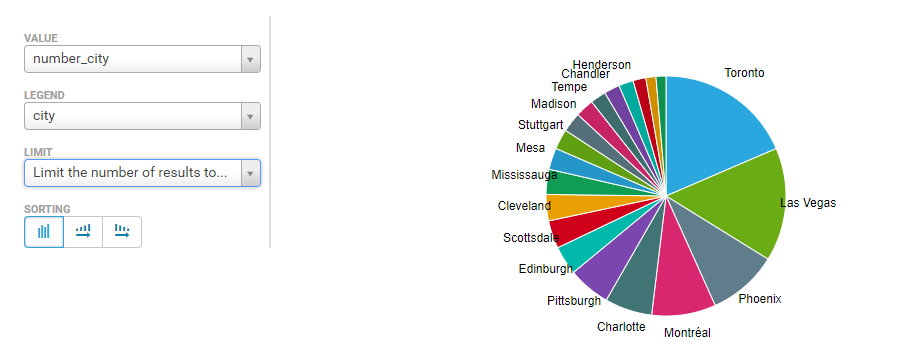
1) In order to map restaurants across United, Select the columns (**state**, count of **business\_id** as **number\_restaurants** ) by grouping data of **restaurants** table by STATE and then order it by **number\_restaurants**

**Q7.1:** 시각화

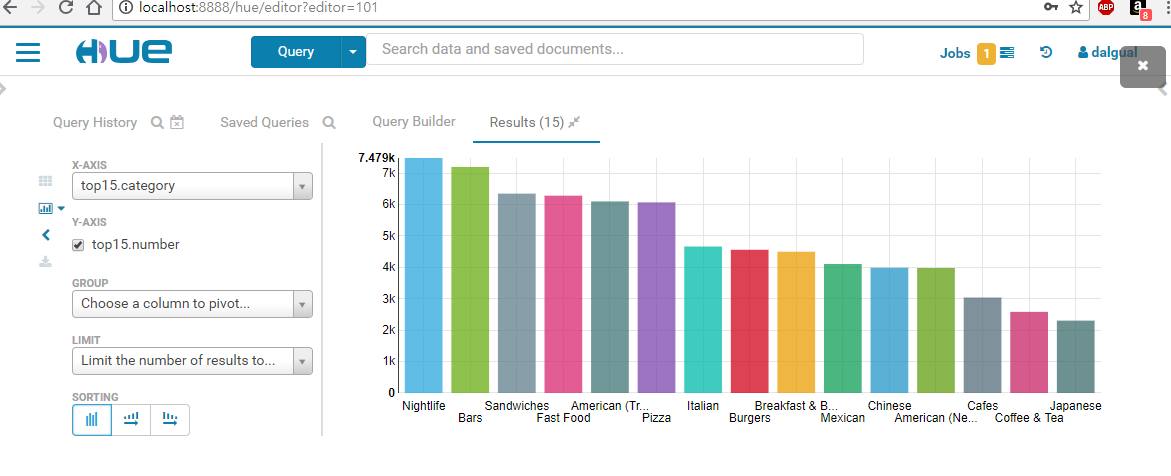
* 시각화: marker map



2) Which Cities Have The Highest Number Of Restaurants? In order to map restaurants across United, Select the columns (**city**, count of **business\_id** as **number\_city** ) by grouping data of restaurants table by **city** and then order it by **number\_city**



3) Find out Top 15 Sub-Categories Of Restaurants from tables **exploded**  and **restaurants** with **business\_id** and grouping by **cat\_exploded** column of the table **exploded,** which are not in("Restaurants","Food")

c

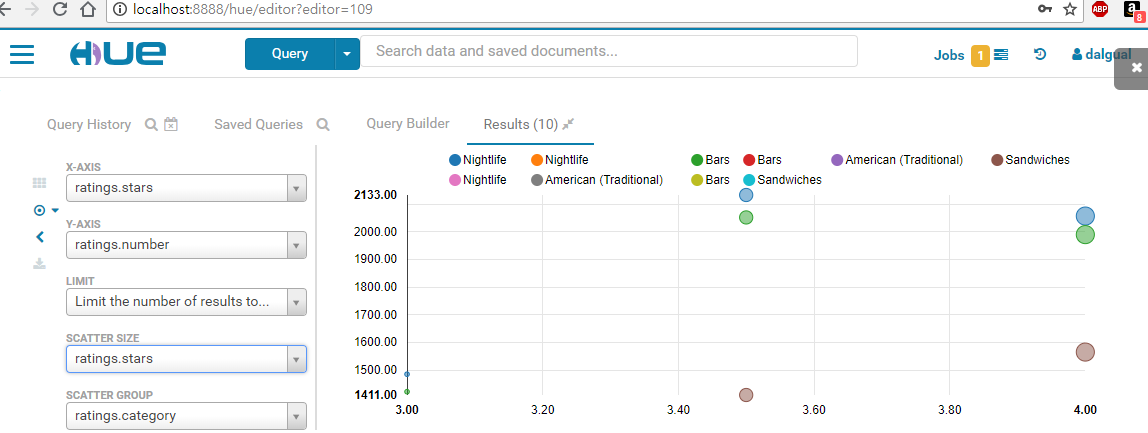
4) Distribution of ratings vs categories:

테이블 ratings를 앞서 만든 테이블 exploded 와 restaurant을 가지고 같은 business\_id로 조인하여 다음의 조건을 만족하여 만드세요:

* exploded 의 cat\_exploded 값은 다음의 값중 하나: "Nightlife","Bars", "Sandwiches", "Fast Food","American (Traditional)"
* exploded 의 cat\_exploded 와 stars 컬럼을 가지고 Group By 하고 stars 컬럼으로 ASC 하여 정렬하세요;
* 다음의 HiveQL로 Hue에서 그래프를 다음과 같이 만듬: select \* from ratings order by number DESC;

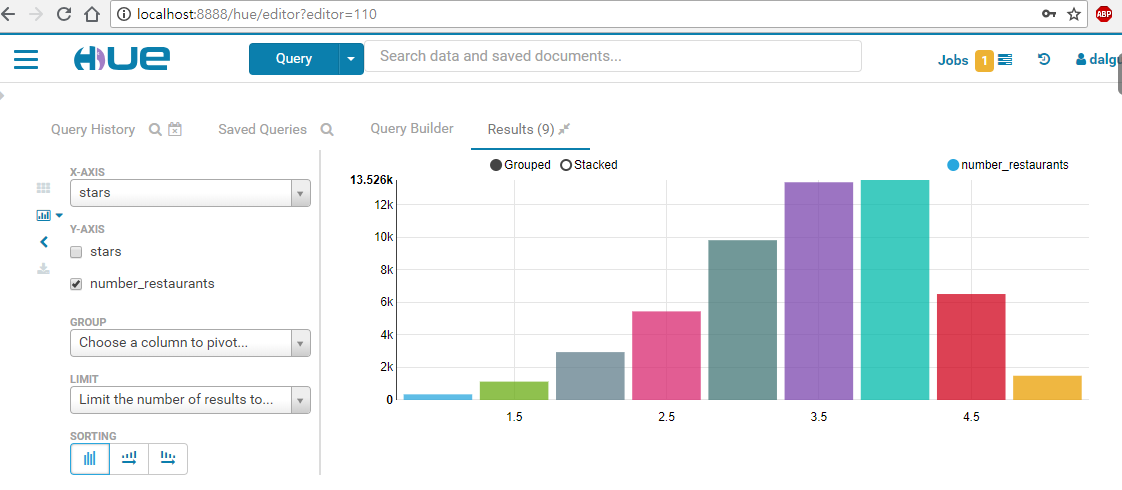
CREATE TABLE ratings  
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe' AS  
SELECT e.cat\_exploded category, e.stars stars, count(e.cat\_exploded) number FROM exploded e JOIN restaurants re  
ON e.business\_id = re.business\_id  
WHERE e.cat\_exploded in ("Nightlife","Bars", "Sandwiches", "Fast Food","American (Traditional)")  
GROUP BY e.cat\_exploded, e.stars ORDER BY stars ASC;

select \* from ratings order by number DESC;



**3:30 - 5pm**

(5) What ratings do the majority of restaurants have?



6) Rating distribution in restaurant reviews

**힌트:**

CREATE TABLE stars2

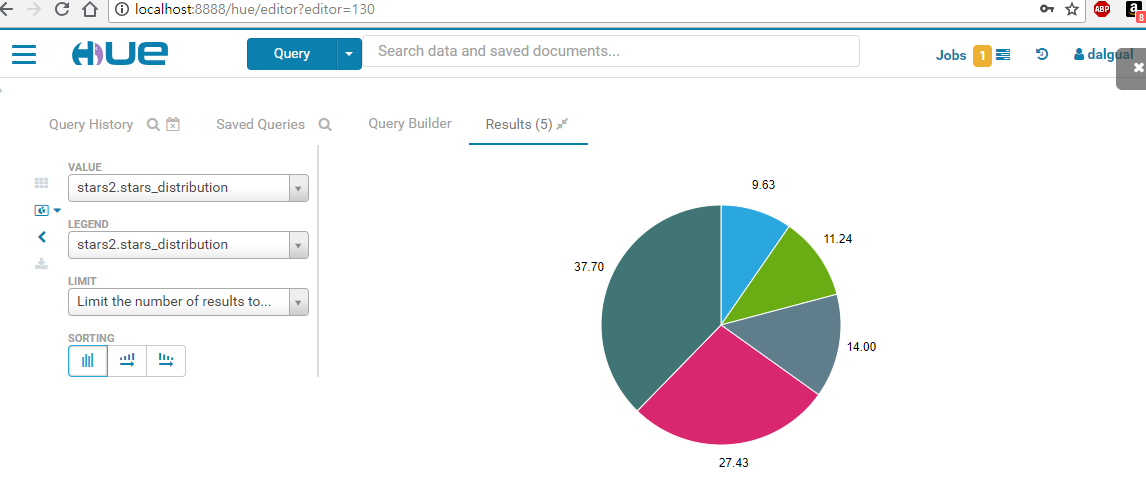
AS  
SELECT stars, [FIll-In] stars\_distribution   
FROM review\_filtered  
GROUP BY stars;

FIl-In:

round((a \* 100.0 / b), 2)

a. count(stars)

b. a의 sum over 함수



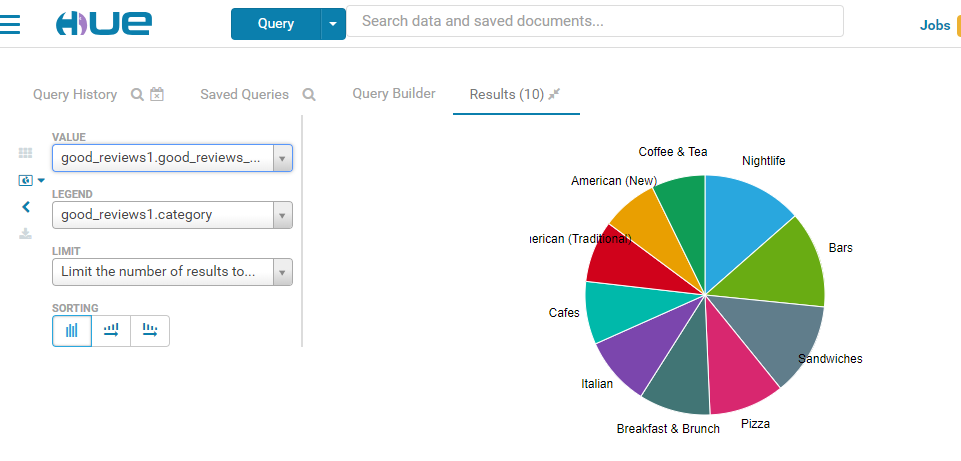
7) Which restaurants get bad vs good reviews?

1. Good Review

**# 힌트:**

good\_reviews1 테이블을 exploded e 와 restaurants re 테이블을 business\_id로 조인하여 컬럼은 (e.cat\_exploded category, count(e.cat\_exploded) good\_reviews\_number) 을 가지고 다음의 조건을 만족하게 만드세요:  
- e.cat\_exploded NOT IN ("Restaurants","Food")

- re.stars>=4

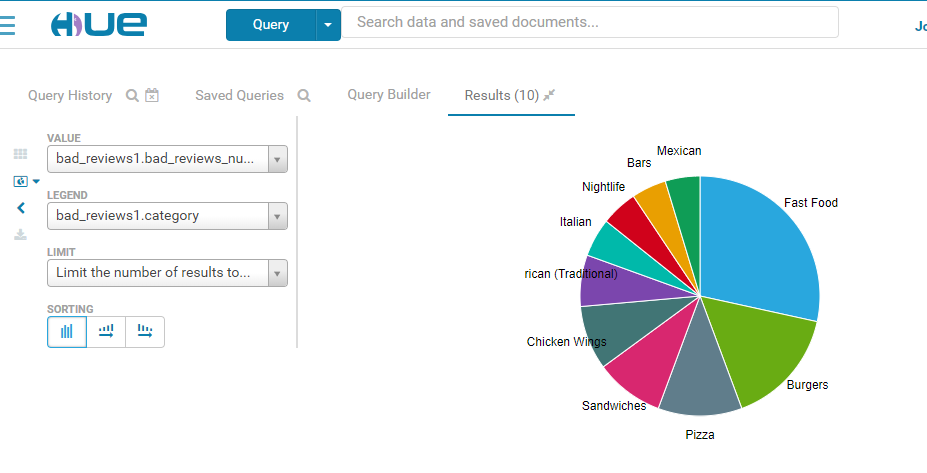


b. Bad Review

**# 힌트:**

good\_reviews1 테이블을 exploded e 와 restaurants re 테이블을 business\_id로 조인하여 컬럼은 (e.cat\_exploded category, count(e.cat\_exploded) good\_reviews\_number) 을 가지고 다음의 조건을 만족하게 만드세요:  
- e.cat\_exploded NOT IN ("Restaurants","Food")

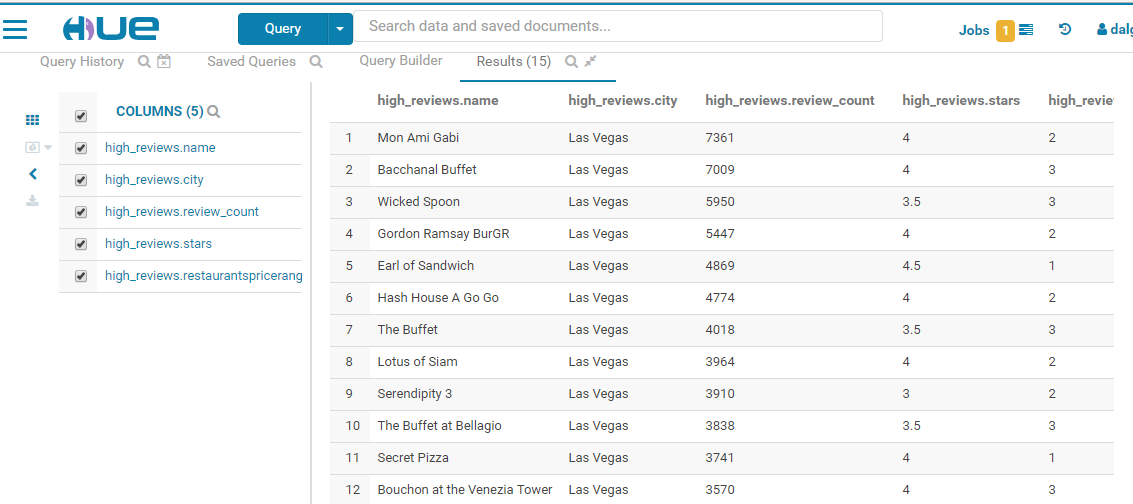
- re.stars<=2



8) Which restaurants have the most reviews?

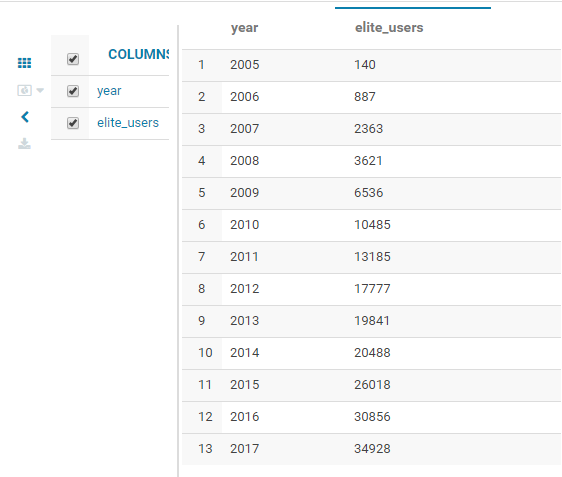
힌트: 다음의 차트에 있는 컬럼들 [ name, city, review\_count, stars, attributes=>restaurantspricerange2]을 가진 테이블을 restaurant 테이블을 기본으로 하여 만드세요

* WHERE review\_count >= 1000  
  ORDER BY review\_count DESC LIMIT 15;

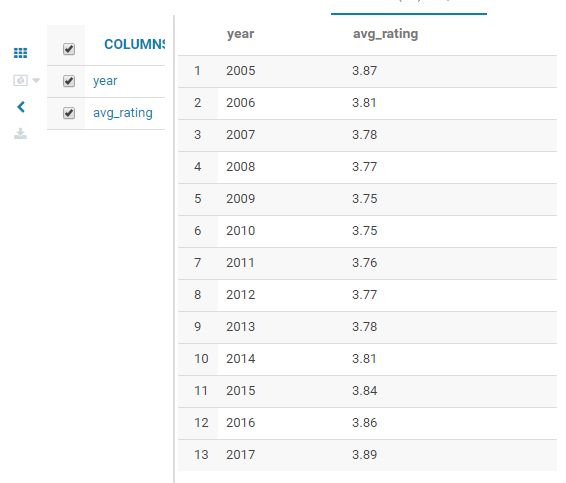


9) What number of yelp users are elite?  
Do they rate differently than non -elite users?

1. Average rating by all users: 힌트: users테이블에서 average\_stars 컬럼을 기본으로.
2. Average rating by elite users: 힌트: elite\_users테이블에서 average\_stars 컬럼을 기본으로.
3. Count number of elite users by year: 힌트: elite\_users테이블에서 유니크한 user\_id 컬럼을 기본으로 GROUP BY elite\_year.



d. Count average reviews by elite users by year: 힌트: elite\_users테이블에서 average\_stars 컬럼을 기본으로 GROUP BY elite\_year.



**5:00pm - 5:50pm:** 투토리알/발표 슬라이드 체크

=====================================================================

**Day1:**

9am - 9:30m: 강사소개 및 자기 소개

<https://www.slideshare.net/dalgual/ai-on-big-data>   
<https://www.linkedin.com/in/jongwook-woo-7081a85>   
<https://www.linkedin.com/in/henrypark/>   
  
9:30am - 10:00m: 실습 계획 소개, 팀 구성, 데이터 소개 및 아마존 클라우드 하둡 EMR 소개

* 팀: 5팀 (3명) & 2팀 (4명) (총 23명, 7팀) => 시스템 어드민, 개발분석, 투토리알&발표슬라이드
* Yelp 소개
* 아마존 EMR: S3, EMR 시작, HDFS, Glue, 키, 포트 오픈, ssh, tunneling 힌트

**NOTE**: 매 4시간 팀 회의로 롤 로테이션

|  |  |  |  |
| --- | --- | --- | --- |
| **팀 번호** | **시스템 어드민** | **개발분석** | **투토리알&발표슬라이드 작성** |
| **1** | 김정미 | [양중근](https://www.linkedin.com/in/joongkeun-yang-ba790086) | [정원석](https://www.linkedin.com/in/won-sug-jung-14b88250/) |
| **2** | [송선영](http://linkedin.com/in/sunyoung-song-79b690142) | [김장현](https://www.linkedin.com/in/janghyun-kim-816bbb27/) | 김형석 |
| **3** | 신행철 | 이소연 | 백규진 |
| **4 (4명)** | 남병출 | 안세환 | [배형민](https://www.linkedin.com/in/hyungmin-bae-2a989a10/), 홍남식 |
| **5 (4명)** | 유건우 | 김지혁 | 이유진, [이대식](http://linkedin.com/in/daesik-lee-bb92a721), |
| **6 (4명)** | 김만수 | 민은영 | 황종윤, 김상진([링크드인](https://www.linkedin.com/in/sangjin-kim-51489b166)) |
| **7 (4명)** | 홍남식 | 김승현 | 유현주 |

10 am - 11:45am: 팀별 역할 분담 & 아마존 하둡 EMR 환경 구축

- Instructors check: Hue, Zeppelin Web UI

- 1) Download data and extract it.

Data set URL and information about the data set:<https://www.yelp.com/dataset> ; Download **yelp\_dataset.tar.gz** file from

2) Putty, Git Bash, AWS CLI

1) Use shell terminal or gitbash to untar yelp\_dataset.tar.gz file

• gitbash: <http://www.techoism.com/how-to-install-git-bash-on-windows/>

• tar -zxvf yelp\_dataset.tar

• will generate 6 jsons and some pdf fiels: ./dataset/business.json, checkin.json, photos.json, review.json, tip.json, user.json

2) AWS CLI install

• Google AWS CLI download:<https://aws.amazon.com/cli/>

• Windows:<https://s3.amazonaws.com/aws-cli/AWSCLI64.msi>

• Git bash or shell terminal: which aws

• Need your key .pem to connect your S3 to upload json files

* <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html#having-ec2-create-your-key-pair>

• Donwload pem from your AWS account: ~/.ssh/yours.pem

• Aws Access Key ID, AWSSecretKey

• AWS Configure to connect to AWS:

$ aws configure

AWS Access Key ID [None]: \*\*\*\*\*\*\*\*\*\*\*\*

AWS Secret Access Key [None]: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Default region name [None]:

Default output format [None]:

# pem file should be in .ssh folder: <https://www.ssh.com/ssh/keygen/>

$ssh-keygen

# AWS CLI: <https://aws.amazon.com/cli/>

# AWS Region: <https://docs.aws.amazon.com/ko_kr/general/latest/gr/rande.html>

#

|  |  |
| --- | --- |
| 아시아 태평양(서울) | ap-northeast-2 |

$ aws s3 ls

$ aws s3 cp photos.json s3://yelp/photos

upload: .\photos.json to s3://yelp/photos/photos.json

$ aws s3 cp review.json s3://yelp/review

11:45am - 1pm: Lunch

1pm - 1:30: AWS CLI 소개

1:30 - 2:30pm: 데이터 수집 및 S3에 업로드

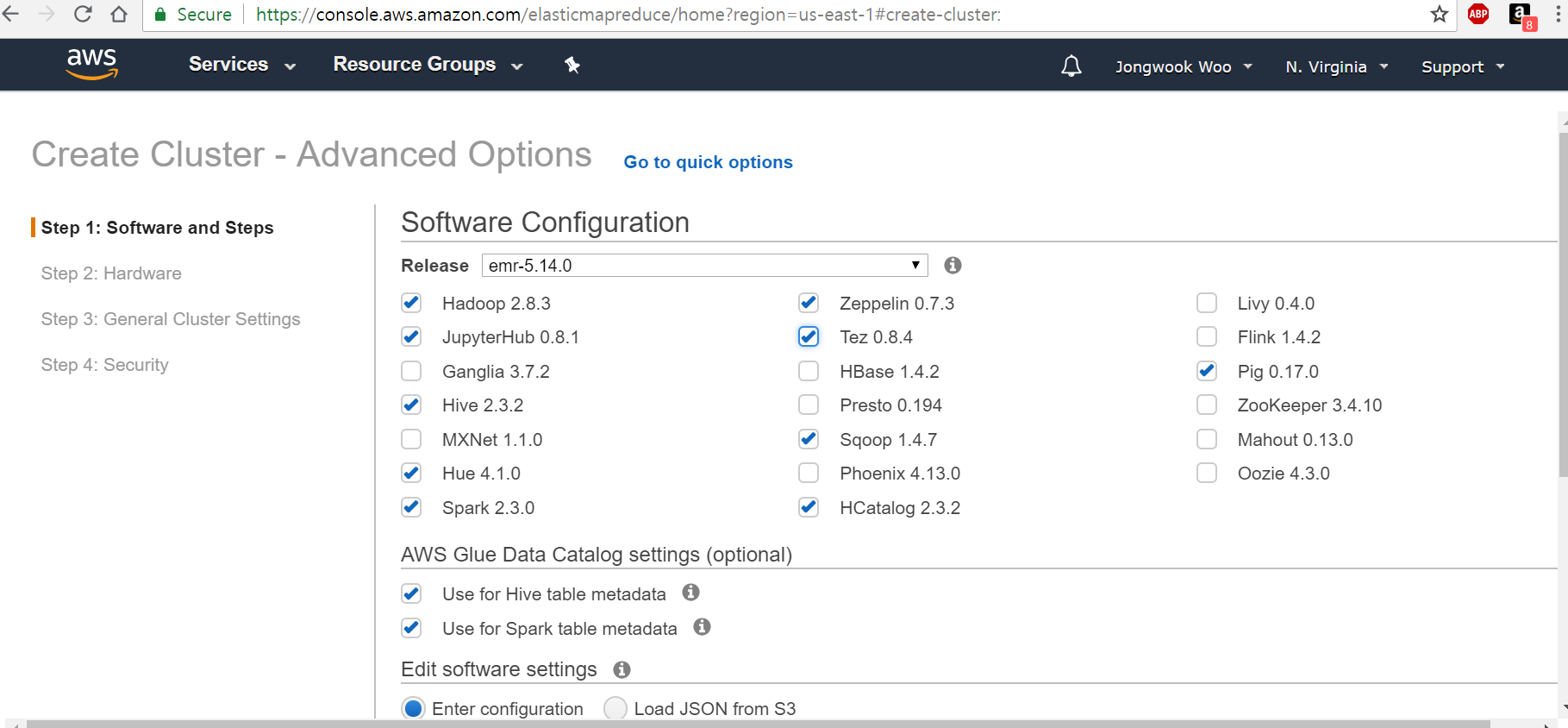
- 데이타 다운로드

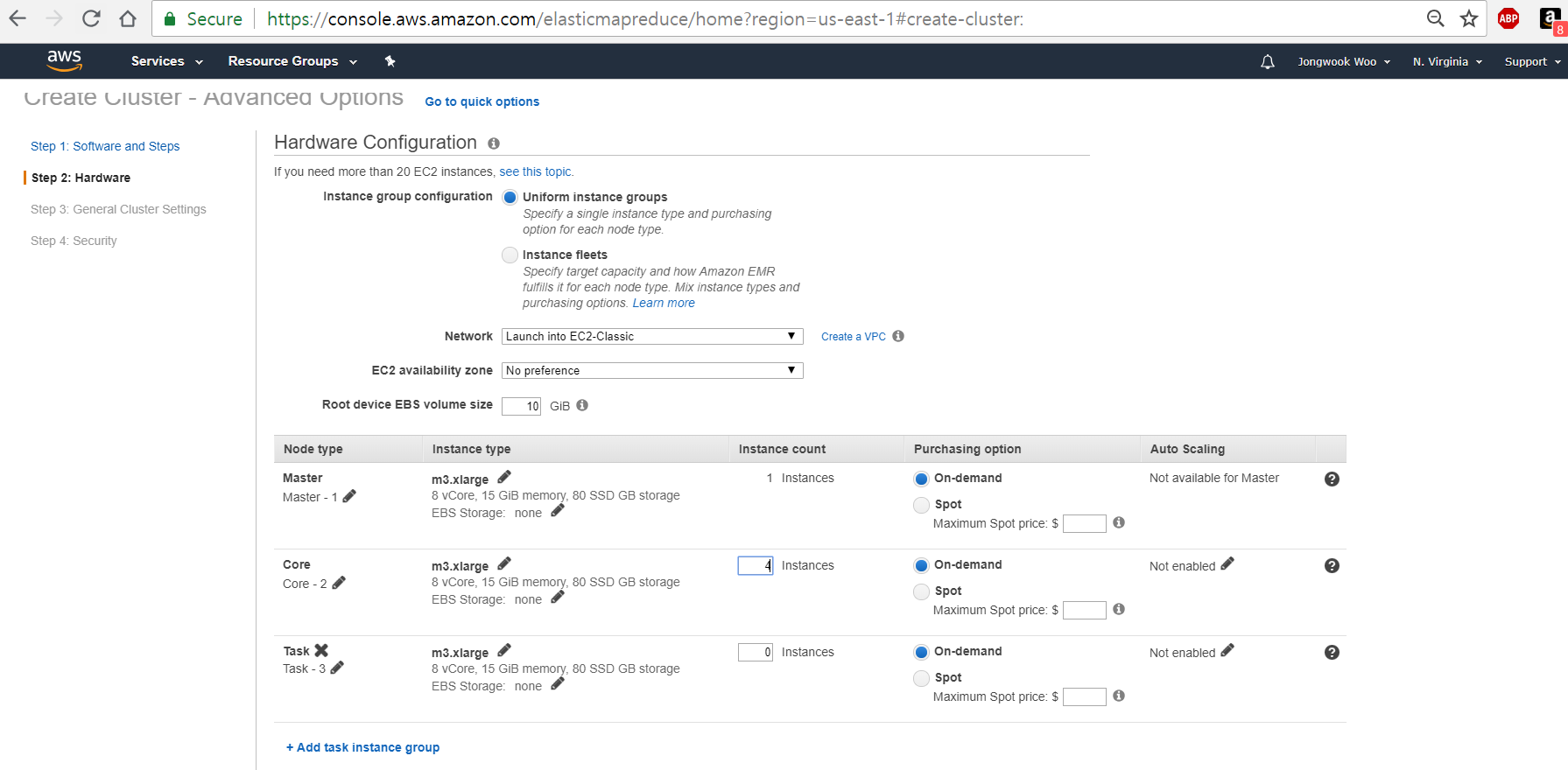
- S3 bucket 만듬: ../yelp/

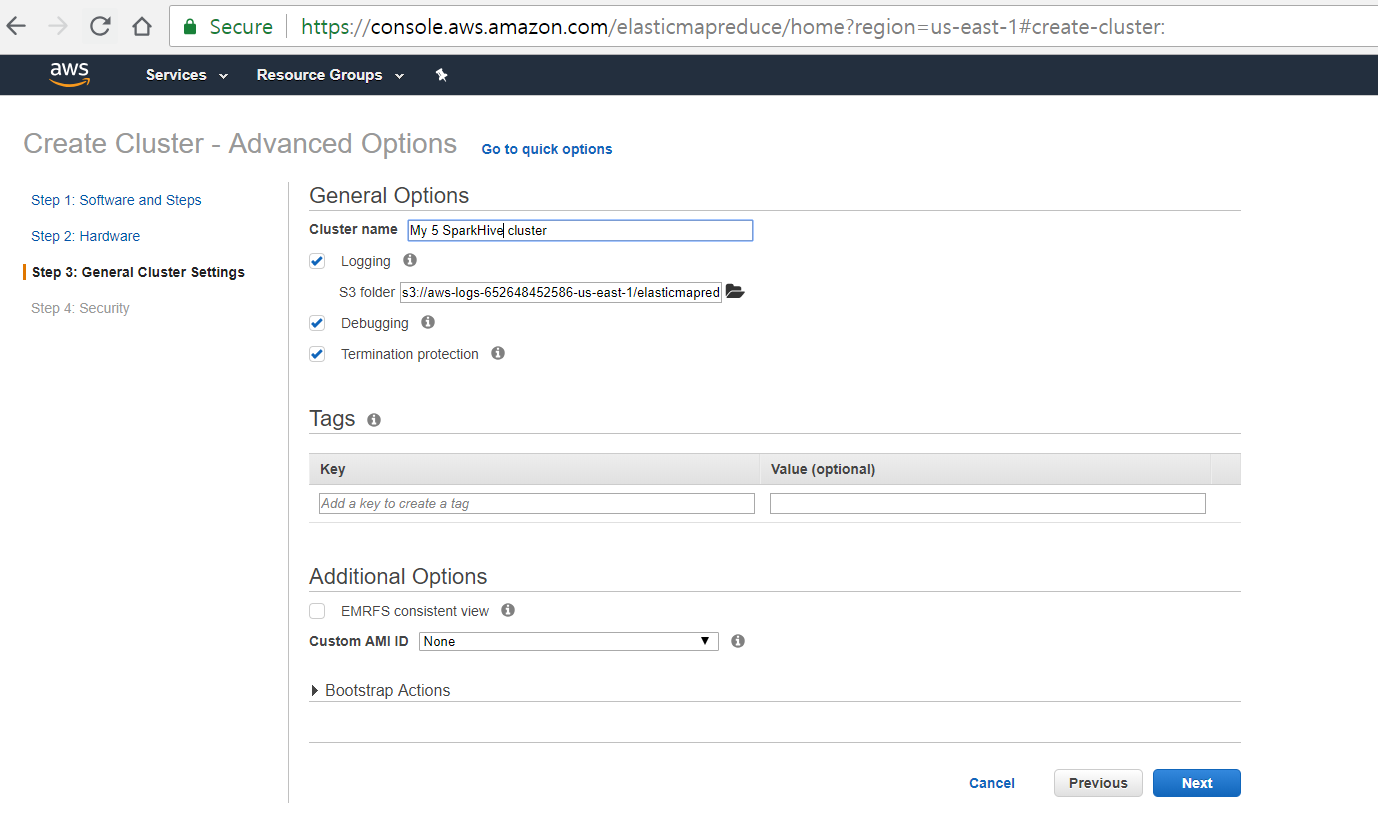
- AWS CLI를 이용한 데이터 업로드

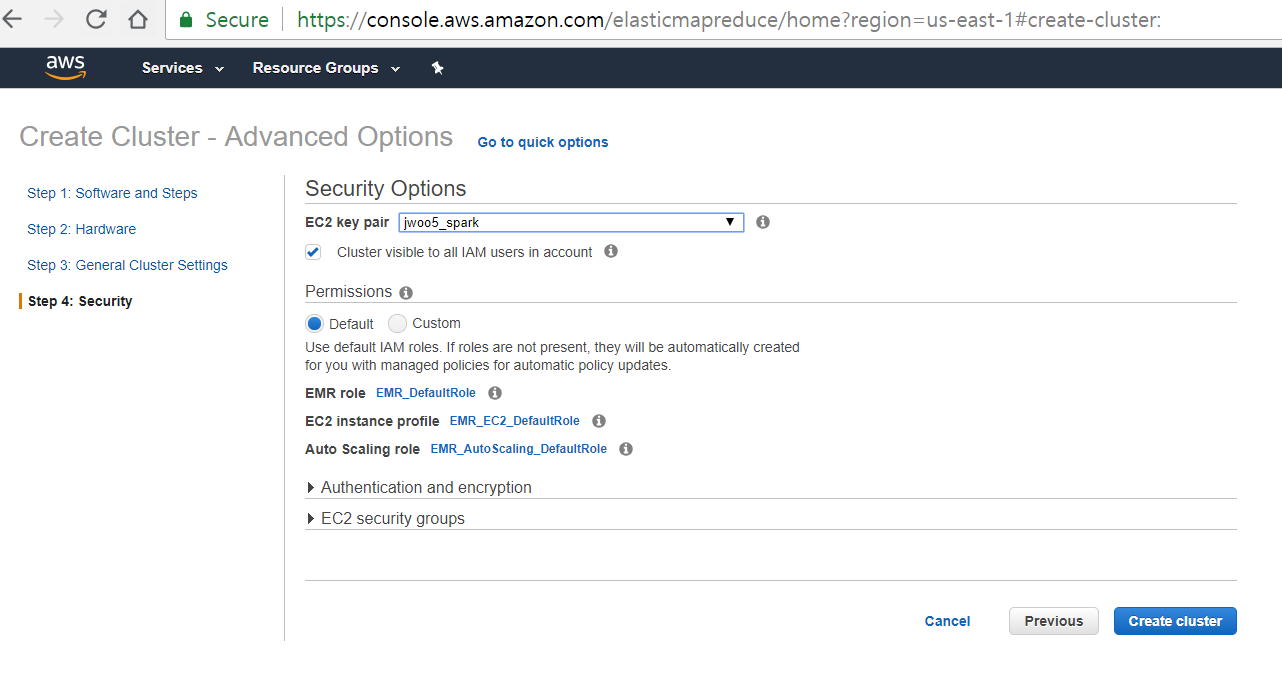
- Instructors check: Hue, Zeppelin Web UI

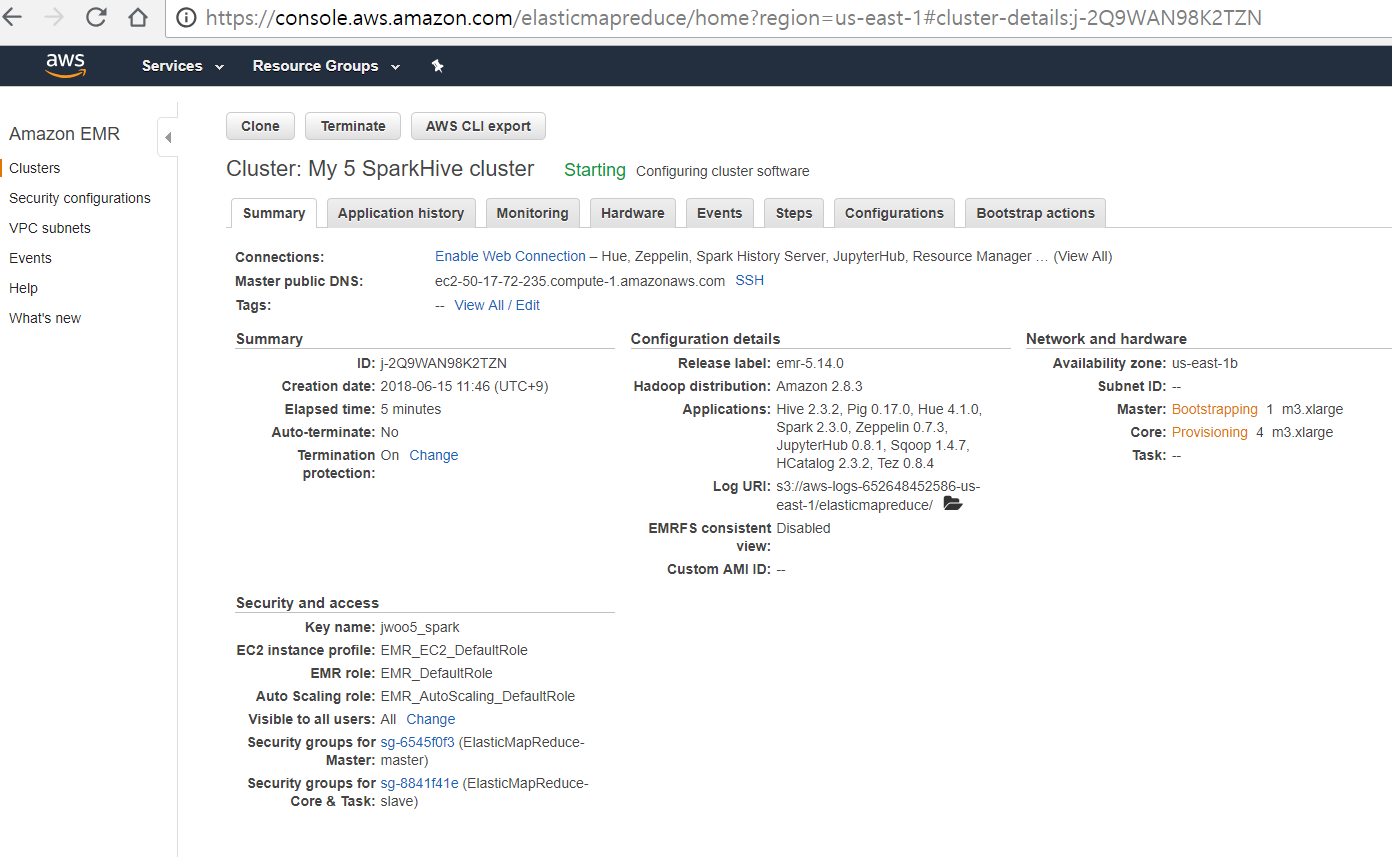
* EMR launch











2:30pm - 3pm: 하이브 external table, Json SerDe, S3 데이터 로드

* tunneling: Ports: Hue: 8888, Zeppelin: 8890
* <https://blog.trackets.com/2014/05/17/ssh-tunnel-local-and-remote-port-forwarding-explained-with-examples.html?utm_source=cronweekly.com>

3pm - 5:00pm: 하이브 테이블 구축 및 테스트

* 4 tables: review, users, tip, business
* <https://github.com/rcongiu/Hive-JSON-Serde>

ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'  
STORED AS TEXTFILE  
LOCATION 's3://hipicdatasets/yelp/review';

Business & Checkin:

hours struct<friday:string, monday:string, saturday:string, sunday:string, thursday:string, tuesday:string, wednesday:string>,

Noise\_Level:string,  
Open24Hours:boolean,  
Outdoor\_Seating:boolean,  
Restaurants\_Attire:string,  
Restaurants\_Counter\_Service:boolean,  
Restaurants\_Delivery:boolean,  
Restaurants\_Good\_For\_Groups:boolean,  
Restaurants\_Reservations:boolean,  
Restaurants\_Table\_Service:boolean,  
Restaurants\_Take\_Out:boolean,  
Smoking:string,  
WheelchairAccessible:boolean,  
WiFi:string,CREATE EXTERNAL TABLE IF NOT EXISTS business (  
address string,  
business\_id string,  
categories array<string>,  
city string,  
hours struct<friday:string, monday:string, saturday:string, sunday:string, thursday:string, tuesday:string, wednesday:string>,  
is\_open int,  
latitude double,  
longitude double,  
name string,  
neighborhood string,  
postal\_code string,  
review\_count int,  
stars double,  
state string,  
Attributes struct<   
Accepts\_Insurance:boolean,  
Ages\_Allowed:string,  
Alcohol:string,  
Bike\_Parking:boolean,  
Business\_Accepts\_Bitcoin:boolean,  
Business\_Accepts\_Credit\_Cards:boolean,  
By\_Appointment\_Only:boolean,  
Byob:boolean,  
BYOB\_Corkage:string,  
Caters :boolean,  
Coat\_Check:boolean,  
Corkage:boolean,  
Dogs\_Allowed:boolean,  
Drive\_Thru:boolean,  
Good\_For\_Dancing:boolean,  
Good\_For\_Kids:boolean,  
Happy\_Hour:boolean,  
Has\_TV:boolean,  
  
Ambience:struct<   
Casual:boolean,  
Classy:boolean,  
Divey:boolean,  
Hipster:boolean,  
Intimate:boolean,  
Romantic:boolean,  
Touristy:boolean,  
Trendy:boolean,  
Upscale:boolean>,  
BestNights:struct<  
Friday1:boolean,  
Monday1:boolean,  
Saturday1:boolean,  
Sunday1:boolean,  
Thursday1:boolean,  
Tuesday1:boolean,  
Wednesday1:boolean>,  
BusinessParking:struct<  
Garage:boolean,  
Lot:boolean,  
Street:boolean,  
Valet:boolean,  
Validated:boolean>,  
DietaryRestrictions:struct<  
Dairy\_Free:boolean,  
Gluten\_Free:boolean,  
Halal:boolean,  
Kosher:boolean,  
Soy\_Free:boolean,  
Vegan:boolean,  
Vegetarian:boolean>,  
GoodForMeal:struct<   
Breakfast:boolean,  
Brunch:boolean,  
Dessert:boolean,  
Dinner:boolean,  
Latenight:boolean,  
Lunch:boolean>,  
HairSpecializesIn:struct<  
Africanamerican:boolean,  
Asian:boolean,  
Coloring:boolean,  
Curly:boolean,  
Extensions:boolean,  
Kids:boolean,  
Perms:boolean,  
Straightperms:boolean>,  
Music:struct<  
BackgroundMusic:boolean,  
Dj:boolean,  
Jukebox:boolean,  
Karaoke:boolean,  
Live:boolean,  
NoMusic:boolean,  
Video:boolean>,  
restaurantspricerange2:int>)  
ROW FORMAT serde 'org.apache.hive.hcatalog.data.JsonSerDe'  
LOCATION 's3://hipicdatasets/yelp/business';

* Instructors check: 4 tables with SELECT of HiveQL

**NOTE:** As of 3:50pm, Team 5 done

**More work:** Create Parquet table Review2 for Review table, which is stored in HDFS

And all other tables as well: 4 tables: Review2 , users2, tip2, business2

5:00pm - 5:50pm: 투토리알 체크

=====================================================================

------------------------- Q&A

Instructor’s HiveQL:

CREATE TABLE IF NOT EXISTS review\_filtered

ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe' AS

SELECT re.business\_id, r.stars, r.user\_id FROM restaurants re JOIN review r

ON r.business\_id = re.business\_id;

// Team 2’s Hive QL

create table review\_filtered\_test as

select

b.business\_id

, a.name

, a.review\_count

, a.stars

, a.category

, b.review\_id

, b.user\_id

, b.stars as review\_stars

, b.review\_date

, b.text

, b.useful

, b.funny

, b.cool

from

restaurant a

, review2 b

where

a.business\_id = b.business\_id