21.01.06

2021년 1월 6일 수요일 오후 4:09

문제199. 리눅스 설치를 완료하고 아래의 2가지를 설정해 놓으시오.

- 게스트 확장 설치
- scott 계정에서 공유폴더 접근

1. centos 리눅스를 새로 생성한다.

가상환경 이미지 이름 : centos_2

2. 하둡 설치

21.01.07

2021년 1월 7일 목요일 오전 9:43

1. 하둡

- 하둡(hadoop) : 대용량 데이터를 분 처리할 수 있는 자바 기반의 오픈소스 프레임워크로서, 하둡은 분산 파일 시스템인 HDFS(Hadoop Distributed File System)에 데이터를 저장하고, 분산처리 시스템인 멥리듀스를 이용해 데이터를 처리한다.

현업에서 데이터 분석가들이 하둡을 활용하는 방법

1. 하둡에 고객 데이터(정형화 된 데이터)를 저장하고 나이대 별로 분리해서 csv 파일을 생성한다.
2. 나이대별로 분리한 csv 파일 데이터를 가지고 군집분석
3. 나이대별로 분리한 csv 파일 데이터를 가지고 연관분석

데이터의 종류

1. 정형화된 데이터 : emp와 같이 컬럼과 row로 이루어진 rdbms에 저장되는 테이블 형태의 데이터
2. 반정형화된 데이터 : 웹로그와 sns 데이터, html, json
3. 비정형화된 데이터 : 동영상이나 이미지 데이터, 텍스트 데이터

rdbms	비 rdbms
Oracle mssql mysql maria(무료) PostgreSQL	하둡 (무료) 스칼라

하둡이 나온 배경지식

- 구글에서 쌓이는 수많은 데이터들을 RDBMS(오라클)에 입력하고 데이터를 저장하려고 시도 했으나 너무 데이터가 많아서 실패를 하고 자체적으로 빅데이터를 저장할 기술을 개발 했다. 그리고 대외적으로 이 기술에 대한 논문을 하나 발표했다.

  웹페이지의 글들을 오라클에 입력하려면 테이블에 입력할 수 있는 insert문장으로 만들어서 입력을 해야하는데 그렇게 만들 수가 없어서 다른 기술을 개발했다.

  그 논문을 더그커팅이 읽고 자바로 구현했다. 자바를 몰라도 하둡의 데이터를 쉽게 다룰 수 있도록 NoSQL을 제공해서 SQL과 같은 언어(Hive)로 쉽게 빅데이터를 다룰 수 있게 되었다.

NoSQL (Not Only SQL)

- NoSQL은 전통적인 RDBMS와 다른 DBMS를 지칭하기 위한 용어

  그 예로는 Hive, Pig, Mongo db 등이 있다.

하둡의 장점

1. 무료
2. 분산처리 가능

- 여러대의 노드(컴퓨터)를 묶어서 하나의 서버처럼 보이게 하고 여러 노드의 자원을 이용해서 데이터를 처리하기 때문에 처리하는 속도가 아주 빠르다.

하둡의 단점

1. 유지보수가 어렵다.
2. 네임노드가 다운되면 고가용성이 지원 되지 않는다.
3. 한번 저장된 파일은 수정할 수 없다. (기존 데이터에 append는 되는데 update는 되지 않는다.)

하이브(Hive) 사용 예

```
hive> select count(*) from emp;
```

위의 hive를 java로 동 변환해서 결과를 출력해준다.

하둡의 주요 배포판

1. Cloudera 에서 나온 CDH
2. Hortonworks 에서 나온 HDP
3. 아마존에서 나온 EMR(Elastic Mapreduce)
4. Hstreaming 에서 나온 Hsteraming

http://hadoop.apache.org

하둡 설치

1. java 설치
2. keygen 생성
3. 하둡 설치

하둡 설치에 앞서 먼저 해야할 일

1. 현업에서 처럼 리눅스 버에 putty와 같은 접속 프로그램을 이용해서 접속하도록 설정한다.

http://cafe.daum.net/oracleoracle/Sfno/231

문제200. putty로 centos 리눅스 서버에 접속한 화면을 캡처하시오.

문제201. /home/scott 디렉토리로 이동해서 test100이라는 디렉토리를 만드시오.

```
$ cd /home/scott

$ mkdir test100
```

문제202. 리눅스 명령어로 1부터 5까지 자를 출력하는 쉘스크립트를 작성하고 아래와 같이 실행하시오.

```
$ sh a.sh

$ vi a.sh

for i in {1..5}
do
      echo $i
done

$ sh a.sh
```

문제203. 공유폴더로 접근하시오.

```
$ cd /media/sf_data

$ ls -l emp.txt
-rwxrwx---. 1 root vboxsf 1036 Dec 29 03:40 emp.txt
```

문제204. 공유폴더에 있는 emp.txt /home/scott 밑으로 복사하시오.

```
$ cp emp.txt /home/scott/

$ cd

$ ls -l emp.txt
-rwxrwx---. 1 scott scott 1036 Jan  7 00:12 emp.txt
```

문제205. 집으로 와서 emp.txt를 cat으로 여시오.

```
$ cd

$ cat emp.txt
```

문제206. 공유폴더에 있는 dept.txt도 /home/scott 밑으로 복사하고 cat으로 여시오.

$ cp /media/sf_data/dept.txt .

$ cat dept.txt

문제207. 아래의 3개 파일을 공유폴더에서 복사하여 /home/scott 밑에 복사하시오.

1. jdk-7u60-linux-i586.gz
2. hadoop-1.2.1.tar.gz
3. protobuf-2.5.0.tar.gz

$ cd /media/sf_data

$ cp jdk-7u60-linux-i586.gz /home/scott/
$ cp hadoop-1.2.1.tar.gz /home/scott/
$ cp protobuf-2.5.0.tar.gz /home/scott/

$ cd
$ ls -rlt

문제208. centos2를 다시 복제해서 centos2_3으로 만들고 지금까지의 내용을 다시 수행하시오.

12기
_centos...

21.01.08

2021년 1월 8일 금요일 오전 9:42

하둡 : 자바로 만든 분산 파일 시스템

하둡 설치

- 리눅스 설치 후 putty 로 접속할 수 있도록 환경구성
- 하둡설치를 위해 필요한 3가지 파일 준비
    - jdk 파일
    - hadoop 설치 파일
    - prototype 파일
- 자바 설치
- ssh로 리눅스 서버에 접속할 때 패스워드를 물어보지 않도록 설정

문제209. hive에서 emp 테이블을 생성하시오.

```
drop table emp;

create table emp
(empno int,
ename string,
job string,
mgr int,
hiredate string,
sal int,
comm int,
deptno int)
 ROW FORMAT DELIMITED
 FIELDS TERMINATED BY ','
 LINES TERMINATED BY '\n'
 STORED AS TEXTFILE ;
```

문제210. 하둡 파일 시스템에 emp2.csv 를 올리시오.

```
$ cp /media/sf_data/emp2.csv /home/scott/

$ hadoop  fs  -put  emp2.csv  emp2.csv

$ hadoop  fs  -ls  emp2.csv
```

문제211. emp2.csv 를 emp 에 로드 하시오.

```
$ cd hive-0.12.0/bin
$ ./hive
```

emp2.csv 를 /home/scott 에 올린다.

```
hive>  load  data  inpath  '/user/scott/emp2.csv'
     overwrite  into  table   emp;

hive> select  * from  emp;
```

문제212. 월급이 3000인 사원의 이름과 월급을 출력하시오.

```
hive> select ename, sal
   > from emp
   > where sal = 3000;
```

문제213. 직업이 SALESMAN인 사원들의 이름과 직업을 출력하시오.

```
hive> select ename, job
   > from emp
   > where job = 'SALESMAN';
```

문제214. 월급이 1200 이상이고 직업이 SALESMAN인 사원들의 이름과 월급과 직업을 출력하시오.

```
hive> select ename, sal, job
   > from emp
   > where sal >= 1200 and job = 'SALESMAN';
```

emp 와 같은 작은 데이터를 검색하기 위해서 하둡을 사용하는 것이 아니라 대용량 데이터를 빠르게 검색하기 위해서 하둡을 사용한다.

문제215. dept 테이블을 생성하시오.

1. dept.csv 파일을 하둡 파일 시스템에 올린다.
2. dept 테이블 생성 스크립트를 가지고 생성한다.
3. 하둡 파일 시스템에 올라온 dept.csv 파일을 dept 테이블에 입력한다.

```
$ cp /media/sf_data/dept2.csv /home/scott/

$ hadoop fs -put dept2.csv dept2.csv

$ hadoop fs -ls dept2.csv
Found 1 items
-rw-r--r--   3 scott supergroup         84 2021-01-08 00:12 /user/scott/dept2.csv

create table dept
(deptno int,
dname string,
```

```
loc string)
 ROW FORMAT DELIMITED
 FIELDS TERMINATED BY ','
 LINES TERMINATED BY '\n'
 STORED AS TEXTFILE ;

hive> load data inpath '/user/scott/dept2.csv'
   > overwrite into table dept;

hive> select * from dept;
```

문제216. 이름과 부서위치를 출력하시오.

```
hive> select e.ename, d.loc
   > from emp e join dept d
   > on (e.deptno = d.deptno);
```

hive에서 조인 하려면 오라클 조인 분법 말고 1999 ansi 문법으로 조인해야 한다.

문제217. DALLAS 에서 근무하는 사원들의 이름과 월급과 부서위치를 출력하시오.

```
hive> select e.ename, e.sal, d.loc
   > from emp e join dept d
   > on (e.deptno = d.deptno)
   > where d.loc = 'DALLAS';
```

문제218. 직업, 직업별 토탈월급을 출력하시오.

```
hive> select job, sum(sal)
   > from emp
   > group by job;
```

문제219. 부서번호, 부서번호별 평균 월급을 출력하는데 부서번호별 평균 월급이 높은 것부터 출력하시오.

```
hive> select deptno, avg(sal)
   > from emp
   > group by deptno
   > order by 2 desc;
```

그룹함수를 사용해서 정렬할 때 order by 절에 컬럼별칭을 사용하거나 숫자를 입력해야 한다.

문제220. 위의 결과를 다시 출력하는데 소수점 이하는 나오지 않도록 반올림 하시오.

```
hive> select deptno, round(avg(sal))
    > from emp
    > group by deptno
    > order by 2 desc;
```

oracle과 hive 함수 차이점

oracle	hive
to_char(hiredate, 'RRRR')	year(to_date(hiredate))
to_char(hiredate, 'MM')	month(to_date(hiredate))
to_char(hiredate, 'DD')	day(to_date(hiredate))

문제221. 이름과 입사한 년도(4자리)를 출력하시오.

```
hive> select ename, year(to_date(hiredate))
    > from emp;
```

문제222. 1981년도에 입사한 사원들의 이름과 입사일을 출력하시오.

```
hive> select ename, hiredate
    > from emp
    > where year(to_date(hiredate)) = 1981;
```

오라클을 이용하지 않고 하둡을 이용하는 이유

1. 무료
2. 분산파일 시스템의 장점을 이용할 수 있다.
   - 여러개의 컴퓨터를 하나로 묶어서 슈퍼컴퓨터를 만들 수 있다.
3. 큰 텍스트 파일을 테이블에 insert하지 않고 os에 두고 링크만 걸어서 select 할 수 있다.
   - 오라클의 external table 과 유사하다.

영화 평점에 대한 큰 데이터를 내려받아 hive에서 분석

1. 대용량 텍스트 파일의 압축파일을 다운로드 받는다.

```
$ cd
$ wget http://www.grouplens.org/system/files/ml-1m.zip
```

2. 위의 압축파일의 압축을 해제한다.

```
$ unzip ml-1m.zip
$ cd ml-1m
```

문제223. movies.dat 파일의 위의 10줄만 확인하시오.

$ head 10 movies.dat

문제224. movies.dat 파일의 총 라인수가 어떻게 되는지 확인하시오.

$ wc -l movies.dat

문제225. ratings.dat 파일의 총 라인수가 어떻게 되는지 확인하시오.

$ wc -l ratings.dat

문제226. ratings.dat 파일의 위의 10줄만 출력하시오.

$ head 10 ratings.dat

ratings.dat의 컬럼 : UserID::MovieID::Rating::Timestamp
movies.dat의 컬럼 : MovieID::Title::Genres

위의 컬럼에 대한 자세한 소개는 README.txt를 참고

리눅스 os 어디에서 든지 hive라고 치면 hive에 접속하는 방법

$ vi .bashrc

export HIVE_HOME=/home/scott/hive-0.12.0
export PATH=$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$PATH:$HIVE_HOME/bin:$PATH

위의 두줄 추가

$ . .bashrc

ratings.dat 와 movies.dat 빅데이터를 하둡에 저장하고 hive를 이용해서 SQL로 분석

1. ratings.dat 와 movies.dat 의 컬럼과 컬럼 구분을 콤마(,)로 데이터 전처리를 해야한다.
   - $ head -10 movies.dat
     - 컬럼과 컬럼이 :: 으로 구분 되어 있는 것을 확인
   - $ sed s/::/,/g movies.dat >> movies_coma.dat
   - $ head -5 movies_coma.dat
   - $ sed s/::/,/g ratings.dat >> ratings_coma.dat
   - $ head -5 ratings_coma.dat
   - $ sed s/::/,/g users.dat >> users_coma.dat
   - $ head -5 users_coma.dat
     - sed 명령어를 이용해서 :: 을 , 로 변경 하고 저장

2. hive에서 테이블 생성 스크립트를 만든다. (테이블명 : ratings, movies, users)
   - $ hive

   - create table users

(userid int,
gender string,
age int,
occupation int,
zipcode string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE ;

- create table movies
(movieid int,
title string,
genres string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE ;

- create table ratings
(userid int,
movieid int,
rating int,
tstamp string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE ;

3. ratings.dat, movies.dat, users.dat 파일을 하둡 파일 시스템에 올린다.

- $ cd ml-1m
- $ hadoop fs -put movies_coma.dat movies_coma.dat
- $ hadoop fs -put ratings_coma.dat ratings_coma.dat
- $ hadoop fs -put users_coma.dat users_coma.dat
- $ hadoop fs -ls

4. 하둡 파일 시스템에 올린 세개의 파일을 각각 ratings, movies, users 테이블에 로드한다.

- load data local inpath '/home/scott/ml-1m/movies_coma.dat'
overwrite into table movies;

- load data local inpath '/home/scott/ml-1m/users_coma.dat'
overwrite into table users;

- load data local inpath '/home/scott/ml-1m/ratings_coma.dat'
overwrite into table ratings;

- hive> select *
> from movies limit 5;

- hive> select *
> from users limit 5;

- hive> select *
  > from ratings limit 5;

문제227. ratings 테이블에서 영화번호, 영화번호별 영화평점의 평균값을 출력하는데 영화번호별 영화 평점의 평균값이 높은 것부터 출력하시오.

```
hive> select movieid, avg(rating) avg_rating
  > from ratings
  > group by movieid
  > order by avg_rating desc;
```

21.01.11

2021년 1월 11일 월요일 오전 9:48

현업에서 하둡, 하이브, 스칼라 이용방법

1. 하둡과 하이브와 스칼라에서는 데이터 (csv, text)를 로드하는 테이블 생성을 위주로 한다.
2. 하둡과 하이브와 스칼라에서는 데이터 검색을 해서 분석 (GUI 툴 다람쥐 squirrel를 이용 한다.)
3. 하이브와 파이썬 연동, 스칼라와 파이썬 연동을 해서 데이터 시각화와 분석

리눅스와 하둡 시험(NCS 시험) : 질문을 스스로 정하고 데이터를 구해서 하둡에 넣고 분석을 해서 시각화한 결과물 (워드, 한글, PPT)로 제출

```
$ start-all.sh
$ jps
```

6개의 프로세서가 올라오지 않을 때 해결 방법

1. $ stop-all.sh
2. 하둡 data 노드와 name 노드의 디렉토리의 데이터를 모두 지운다.
   - $ cd $HADOOP_HOME/dfs
   - $ cd name
   - $ rm -rf *
   - $cd ..
   - $cd data
   - $rm -rf *
3. 네임노드를 포멧다.
4. $ stop-all.sh
5. $ jps

문제228. 다시 emp와 dept 테이블을 생성하고 데이터를 로드하시오.

```
$ hadoop  fs  -put  emp2.csv  emp2.csv

$ hadoop fs -put dept2.csv dept2.csv
$ hive

create table emp
(empno int,
ename string,
job string,
mgr int,
hiredate string,
sal int,
comm int,
deptno int)
 ROW FORMAT DELIMITED
```

```
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE ;

create table dept
       ( deptno  int,
        dname  string,
        loc     string )
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE ;

hive>  load  data  inpath  '/user/scott/emp2.csv'
     overwrite  into  table   emp

hive> load data inpath '/user/scott/dept2.csv'
   > overwrite into table dept;
```

문제229. 이름, 월급, 월급에 대한 순위를 출력하시오.

```
hive> select ename, sal, dense_rank() over (order by sal desc) rnk
   > from emp;
```

문제230. 이름, 월급, ntile 함수를 이용해서 월급의 등급을 출력하시오.

(월급의 등급은 4등급으로 분류하시오.)

```
hive> select ename, sal, ntile(4) over (order by sal desc) grade
   > from emp;
```

문제231. 아래의 SQL을 Hive로 구현하시오.

```
SQL>
select deptno, listagg(ename, ',') within group (order by ename)
        from emp
        group by deptno;

hive> select deptno, CONCAT_WS(',', COLLECT_SET(ename))
   > from emp
   > group by deptno;
```

문제232. 사원번호, 이름, 월급 월급의 누적치를 출력하시오.

```
hive> select empno, ename, sal, sum(sal) over(order by empno) sumsal
   > from emp;
```

문제233. 오라클의 데이터 분석함수의 lag와 lead 함수가 hive에서 되는

지 확인하시오.

```
SQL>
select deptno, ename, sal,
            lag(sal,1) over (order by empno) lag,
            lead(sal,1) over (order by empno) lead
       from emp;
```

```
hive> select deptno, ename, sal,
   > lag(sal,1,0) over (order by empno) lag,
   > lead(sal,1,0) over (order by empno) lead
   > from emp;
```

문제234. 오라클의 데이터 분석함수의 lag와 lead 함수가 hive에서 되는지 확인하시오.

```
SQL>
select deptno, ename, sal,
            lag(sal,1) over (partition by deptno
                              order by empno) lag,
            lead(sal,1) over (partition by deptno
                              order by empno) lead
       from emp;
```

```
hive> select deptno, ename, sal,
   > lag(sal,1,0) over (partition by deptno order by empno) lag,
   > lead(sal,1,0) over (partition by deptno order by empno) lead
   > from emp;
```

문제235. 부서번호, 이름, 월급, 자기가 속한 부서번호의 평균월급을 출력하시오.

```
hive> select deptno, ename, sal,
   > avg(sal) over (partition by deptno) avgsal
   > from emp;
```

hive에서 결과 출력에 컬럼명 나오게 하는 방법

```
hive> set hive.cli.print.header=true;
```

문제236. 위의 결과를 다시 출력하는데 자기의 월급이 자기가 속한 부서번호의 평균월급보다 더 큰 사원들만 출력하시오.

```
hive> select *
   > from (select deptno, ename, sal,
   > avg(sal) over (partition by deptno) avgsal
```

```
  > from emp) aaa
  > where sal > avgsal;
```

hive에서 from 절의 서브쿼리를 사용하려면 반드시 alias를 사용해야 한다.

문제237. DW에서 많이 사용하는 SQL중 하나인 WITH절인 Hive에서도 되는지 확인하시오.

with절은 hive 1.2.1버전은 지원되지 않고 그 이후 버전부터 가능하다.

```
SQL>
with emp77 as (select job, sum(sal) sumsal
                        from emp
                        group by job)
        select job, sumsal
                from emp77;
```

```
hive> with emp77 (select job, sum(sal) sumsal
  > from emp
  > group by job)
  > select job, sumsal
  > from emp77;
```

문제238. DW에서 많이 사용하는 집계값 구하는 쿼리를 수행하시오. 부서번호, 부서번호별 토탈월급을 출력하는데 맨 아래에 전체 토탈월급이 출력되게 하시오.

```
SQL>
select deptno, sum(sal)
        from emp
        group by rollup(deptno);
```

```
hive> select deptno, sum(sal)
  > from emp
  > group by deptno with rollup;
```

문제239. 위의 결과에서 전체 토탈월급이 아래에 나오도록 order by 절을 사용하시요.

```
hive> select deptno, sum(sal) sumsal
  > from emp
  > group by deptno with rollup
  > order by sumsal;
```

문제240. 케글에서 코로나 데이터 분석에 필요한 코로나 데이터를 내려

받아 리눅스 시스템에 넣으시오.

https://www.kaggle.com/kimjihoo/coronavirusdataset

$ unzip archive.zip

문제241. PatientInfo.csv를 hive의 테이블로 구성하시오.

$ vi PatientInfo.csv
맨 위의 컬럼행을 지우고 저장

$ hadoop fs -put PatientInfo.csv PatientInfo.csv

```
create table patient

(patient_id String,
sex String,
age String,
country String,
province String,
city String,
infection_case String,
infected_by String,
contact_number int,
symptom_onset_date String,
confirmed_date String,
released_date String,
deceased_date String,
state String)
 ROW FORMAT DELIMITED
 FIELDS TERMINATED BY ','
 LINES TERMINATED BY '\n'
 STORED AS TEXTFILE ;
```

문제242. 코로나 환자 데이터를 하둡에 올리고 hive 로 아래와 같이 위에 5건만 조회하고 결과를 캡처하시오.

```
hive> load data inpath '/user/scott/PatientInfo.csv'
    > overwrite into table patient;

select * from patient limit 5;
```

문제243. 나이대(age), 나이대별 인원수를 출력하시오.

```
hive> select age, count(*)
    > from patient
    > group by age;
```

patient 테이블 다시 생성 스크립트

```
drop table patient;

create table patient
(patient_id string,
sex string,
age  string,
country string,
province string,
city string,
infection_case string,
infected_by string,
contact_number int,
symptom_onset_date string,
confirmed_date string,
released_date string,
decreased_date string,
state string)
ROW FORMAT DELIMITED
 FIELDS TERMINATED BY ','
 LINES TERMINATED BY '\n'
 STORED AS TEXTFILE;

hive> exit;

$ hadoop  fs  -put  PatientInfo.csv   PatientInfo.csv

$ hive

hive>  load  data  inpath  '/user/scott/PatientInfo.csv'
       overwrite  into   table  patient;
```

문제244. 위의 결과를 다시 출력하는데 인원수가 높은 것 부터 출력하시오.

```
hive> select age, count(*) cnt
   > from patient
   > group by age
   > order by cnt desc;
```

문제245. 오라클의 grouping sets 가 hive에서도 지원되는지 확인하시오.

```
SQL>
select deptno, sum(sal)
       from emp
       group by grouping sets(deptno,());

hive> select deptno, sum(sal)
```

```
  > from emp
  > group by deptno grouping sets(deptno,());
```

문제246. 나이대, 나이대별 인원수를 출력하는데 맨 아래에 전체 인원수도 출력하시오.

```
hive> select age, count(*) cnt
  > from patient
  > group by age grouping sets(age,())
  > order by cnt;
```

문제247. DW쪽에서 가장 빈번히 수행하는 쿼리문이 아래의 SQL인데 아래의 SQL을 hive로 구현하시오.

```
SQL>
select sum(decode(deptno, 10, sal)) "10",
            sum(decode(deptno, 20, sal)) "20",
            sum(decode(deptno, 30, sal)) "30"
       from emp;

hive> set hive.cli.print.header=true;

select sum(case when deptno=10 then sal end) dept10,
       sum(case when deptno=20 then sal end) dept20,
       sum(case when deptno=30 then sal end) dept30
from emp;
```

문제248. DW 쪽에서 많이 사용하는 아래의 SQL에 group by 까지 사용한 결과를 hive로 구현하시오.

```
SQL>
select job, sum(decode(deptno, 10, sal)) "10",
            sum(decode(deptno, 20, sal)) "20",
            sum(decode(deptno, 30, sal)) "30"
       from emp
       group by job;

select job, sum(case when deptno=10 then sal end) dept10,
            sum(case when deptno=20 then sal end) dept20,
            sum(case when deptno=30 then sal end) dept30
       from emp
       group by job;
```

현업에서 hive 사용할 때 csv 파일을 받았으면 csv 파일을 가지고 table 생성을 잘 할 수 있어야 한다.

문제249. 코로나 데이터 중에 감염경로를 알려주는 Case.csv를 가지고 hive에 테이블을 생성하시오.

```
$ hadoop fs -put Case.csv Case.csv

create table doong

(case_id String,
province String,
city String,
group String,
infection_case String,
confirmed int,
latitude float,
longitude float)
 ROW FORMAT DELIMITED
 FIELDS TERMINATED BY ','
 LINES TERMINATED BY '\n'
 STORED AS TEXTFILE ;

hive> exit;

$ hadoop  fs  -put  Case.csv   Case.csv

$ hive

hive>  load  data  inpath  '/user/scott/Case.csv'
       overwrite  into   table  doong;
```

문제250. 감염경로(infection_case)를 중복제거해서 출력하시오.

```
SQL>
select distinct infection_case
       from doong;

hive> select distinct infection_case
   > from doong;
```

tajo를 이용해서 데이터 검색하기

- hive보다 훨씬 빠른 noSQL이다. 자체 설계한 처리 엔진을 통해 Hive보다 훨씬 빠른 처리시간을 보여준다.

tajo 설치

```
       ##############################################################################
```

스파크 설치

http://cafe.daum.net/oracleoracle/Sfno/327

문제251. id가 1202인 사원의 이름과 나이를 출력하시오.

```
sql("select name, age from employee where id = 1202").show()
```

문제252. 이름이 satish 인사원의 이름과 나이를 출력하시오.

```
sql("select name, age from employee where name = 'satish'").show()
```

문제253. emp테이블을 스칼라에서 생성하시오.

```
scala>  val sqlContext = new org.apache.spark.sql.hive.HiveContext(sc)

scala>  sqlContext.sql("create table  IF NOT EXISTS emp (empno int, ename string, job string, mgr int, hiredate string, sal int, comm int, deptno int) ROW FORMAT DELIMITED  FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n'")
```

문제254. 스칼라에서 생선한 emp 테이블에 emp.csv 를 로드 하시오.

문제255. 스칼라에서 dept 테이블을 생성하고 dept 테이블에 데이터를 입력하시오.

21.01.12

2021년 1월 12일 화요일 오전 11:01

스칼라에서 SQL이 수행되지 않을 때

1. 스파크의 sbin 디렉토리로 이동
   $ cd /home/scott/spark/sbin
   $ ls

- start-all.sh 와 stop-all.sh 가 있는지 확인한다.

  $ ./stop-all.sh

  $ ./start-all.sh

  $ spark-shell

스칼라에서 emp 테이블 생성하는 방법

1. hive SQL 을 스칼라에서 사용 명령어

```
scalar> val sqlContext = new org.apache.spark.sql.hive.HiveContext(sc)
```

2. emp 테이블을 생성 명령어

```
scalar>sqlContext.sql("create table  IF NOT EXISTS emp (empno int, ename string, job string, mgr int, hiredate string, sal int, comm int, deptno int) ROW FORMAT DELIMITED  FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n'")
```

3. emp 테이블에 emp2.txt 를 로드 명령어

```
scalar>sqlContext.sql("LOAD DATA LOCAL INPATH '/home/scott/emp2.txt' INTO TABLE emp")
```

4. emp 테이블의 내용을 확인 명령어

```
scala>  sql("select * from emp").show()
```

5. emp 테이블이 전체 몇건인 확인하는 명령어

```
scalar>  sql("select * from emp").count()
res7: Long = 14
```

emp 테이블을 drop고 다시 생성하는 방법

```
scalar>  sql("drop  table  emp")
```

문제256. dept2.csv를 복사해서 dept2.txt로 생성하고 스칼라에서

dept 테이블을 생성하시오.

```
$ cp  dept2.csv  dept2.txt

$ cat  dept2.txt

scalar> val sqlContext = new org.apache.spark.sql.hive.HiveContext(sc)

scalar> sql("drop  table  dept")

scalar>  sqlContext.sql( "CREATE TABLE IF NOT EXISTS dept(deptno int, dname string, loc string) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n'")

scalar> sqlContext.sql("LOAD DATA LOCAL INPATH '/home/scott/dept2.txt' INTO TABLE dept")

scalar>  sql("select * from dept").show()
```

문제257. 스칼라에서 부서번호가 30번인 사원들의 모든 데이터를 출력하시오.

```
scala> sql('select  *  from  emp  where  deptno=30').show()
```

문제258. 위에서 출력된 결과의 건수를 확인하시오.

```
scala> sql('select  *  from  emp  where  deptno=30').count()
```

문제259. 직업이 SALESMAN인 사원들의 이름과 직업과 월급을 출력하시오.

```
scala> sql("select  ename, job, sal  from  emp where  job='SALESMAN'").show()
```

문제260. 부서 테이블 전체를 출력하시오.

```
scala> sql("select * from  dept").show()
```

문제261. 이름과 부서위치를 출력하시오.
(1999 ANSI 조인 문법으로 수행)

```
scala> sql("select  e.ename, d.loc  from  emp  e  join dept  d
          on (e.deptno=d.deptno)").show()
```

문제262. DALLAS에서 근무하는 사원들의 이름과 부서위치를 출력하

시오.

```
scala> sql("""select  e.ename, d.loc  from  emp  e  join dept  d
            on (e.deptno=d.deptno)
           where  d.loc='DALLAS'""").show()
```

더블 쿼테이션 마크 3개를 둘러주면 된다.

문제263. 아우터 조인이 가능한지 확인하시오.

```
SQL>
select e.ename, d.loc
       from emp e right outer join dept d
       on (e.deptno = d.deptno)
```

```
scala> sql("""select  e.ename,  d.loc
             from  emp  e   right  outer  join  dept   d
             on  ( e.deptno = d.deptno)""").show()
```

문제264. JONES 보다 더 많은 월급을 받는 사원들의 이름과 월급을 출력하시오.

```
sql(""" select ename, sal
      from emp
      where sal > (select sal from emp where ename='JONES') """).show()
```

문제265. ALLEN 보다 늦게 입사한 사원들의 이름과 입사일을 출력하시오.

```
scala> sql("""select ename, hiredate
              from emp
              where hiredate > (select hiredate
                                from emp
                               where ename = 'ALLEN')""").show()
```

문제266. 이름, 월급, 월급에 대한 순위를 출력하시오.

```
scala> sql("""select ename,sal, rank()over (order by sal desc)
             from emp""").show()
```

문제267. 직업, 직업별 토탈월급을 출력하시오.

```
scala> sql("""select job, sum(sal)
    |   from emp
```

```
   |  group by job""").show()
```

스파크 명령어 매뉴얼 : http://cafe.daum.net/oracleoracle/Sfno/385

문제268. 위의 결과중에 숫자로 나오는 값에 대한 통계값을 출력하시오.

```
scala> sql("""select job, sum(sal)
    | from emp
    | group by job""").describe().show()
```

문제269. 사원 테이블을 출력하는데 맨 위에 한 행만 출력하시오.

```
scala> sql("""select *
    | from emp""").head()
```

데이터 분석을 위한큰그림에서의 하둡과 스칼라의 역할

1. 빅데이터를 저장하기 위한 테이블 생성
2. 테이블에서 데이터를 전처리하고 필요한 데이터 검색
3. 검색한 데이터를 csv 파일로 저장해서 파이썬, R로 보낸다.
4. 파이썬,R 에서 가져온 csv파일로 데이터 시각화

문제270. 위에서 출력한 직업과 직업별 토탈월급의 결과를 csv 파일로 저장하시오.

```
scala> sql("select job, sum(sal) from emp group by
job").coalesce(1).write.option("header","true").option("sep",",").mode("overwrite").csv("/home/scott/
dd")
```

write.option("header","true") : 컬럼명 출력

option("sep",",") : csv 파일 형태 만든다.

mode("overwrite").csv("/home/scott/dd") : /home/scott/dd 라는 폴더 안에 생성

아래의 결과를 복사해서 윈도우 notepad에 붙여넣고 job.csv로 저장한다.

```
job,sum(sal)
ANALYST,6000
SALESMAN,5600
CLERK,4150
MANAGER,8275
PRESIDENT,5000
```

윈도우에서 spyder 을 실행하고 아래의 내용을 코딩한다.

```
import pandas as pd

emp = pd.read_csv("c:\\data\\job.csv") # csv 을 불러와서 emp 데이터 프레임생성
print (emp.columns) # 컬럼명 확인
```

문제271. job.csv의 내용을 막대그래프로 시각화 하시오.

```
import pandas as pd

emp = pd.read_csv("c:\\data\\job.csv")
result = emp['sum(sal)']
result.index = emp['job']
result.plot (kind='bar',color='red')
```

문제272. 부서번호, 부서번호별 토탈월급을 막대 그래프로 그리시오.

(스칼라를 이용하여 데이터 전처리 및 검색을 하고 파이썬에서 시각화하는 큰 그림을 구현)

```
scala> sql("""select deptno,sum(sal)
    | from emp
    | group by
detpno""").coalesce(1).write.option("header","true").option("sep",",").mode("overwrite").csv("/home/
scott/ff")
```

```
deptno,sum(sal)
20,10875
10,8750
30,9400
```

spyder

```
import pandas as pd

emp = pd.read_csv("c:\\data\\deptsal.csv")
result = emp['sum(sal)']
result.index = emp['deptno']
result.plot (kind='bar',color='blue')
```

하둡, 하이브, 스칼라는 빅데이터를 저장하는 테이블 생성 및 저장 하고 파이썬이나 R로 시각화 또는 분석을 한다.

대용량 데이터는 csv 파일을 따로 다운로드 받게 해야하는데 방법은 아래와 같다.

1. mobaxterm 프로그램을 다운로드 받아서 설치한다.
2. mobaxterm 으로 리눅스 centos에 접속한다.
3. mobaxterm 에서 파일을 다운로드 받는다.

문제273. mobaxterm을 이용해서 sample.csv를 /home/scott 밑에 올

리시오.

현업에서는 오라클 vm 으로 가상환경을 만드는게 아니라 실제 리눅스 서버를 사용하므로 노트북에 있는 파일을 별도의 컴퓨터로 구성되어 있는 리눅스 서버에 올리려면 mobaxterm과 같은 툴을 이용하면 된다.

리눅스에 설치되어 있는 아나콘다의 spyder를 내 노트북의 화면에 띄우게 하는 방법

http://cafe.daum.net/oracleoracle/Sfno/411

http://cafe.daum.net/oracleoracle/Sfno/412

mobaxterm -> settings -> configuration -> x11 -> x11 remote access -> full 로 변경

복제 환경에 리눅스 환경에 아나콘다 설치하기

http://cafe.daum.net/oracleoracle/Sfno/127

문제274. emp2.csv를 리눅스의 spyder 에서 판다스로 불러오고 프린트 하시오.

```
import pandas as pd

emp = pd.read_csv('/home/scott/emp2.csv',header = None)

print (emp)
```

문제275. emp2.csv에서 emp로 로드하고 사원 테이블의 월급을 막대 그래프로 시각화 하시오.

```
import pandas as pd

emp = pd.read_csv('/home/scott/emp2.csv',header = None)

result = emp[5]
result.index = emp[1]
result.plot (kind = 'bar',color = 'red')
```

리눅스 하둡 시험문제 제출 포멧

문제276. Case.csv를 스칼라로 로드해서 테이블 생성한 후에 지역, 지역별 코로나 감염자 수를 count하고 결과를 가지고 파이썬에서 막대

그래프로 시각화 하시오.

21.01.13

2021년 1월 13일 수요일 오전 9:45

하둡 분산 파일 시스템 명령어

1. ls 명령어

- 지정된 디렉토리에 있는 파일의 정보를 출력

```
$ cd
$ ls -l emp2.csv
$ hadoop fs -put /home/scott/emp2.csv /user/scott/emp3.csv
$ hadoop fs -ls /user/scott/emp3.csv
```

하둡 파일 시스템에 린 파일들을 웹페이지로 확인하는 방법

```
$ export DISPLAY = 192.168.19.31:0.0
$ xclock
$ firefox centos:50070
```

2. lsr 명령어

- 현재 디렉토리 뿐만 아니라 하위 디렉토리까지 조회하는 명령어

```
$ hadoop fs -lsr /user/
```

문제277. 하둡 파일 시스템의 /(루트) 밑에 있는 모든 하위 디렉토리를 조회하시오.

```
$ hadoop fs -lsr /
```

3. du 명령어

- 파일의 용량을 확인하는 명령어

```
$ hadoop fs -du
```

4. dus 명령어

- 파일의 전체 합계 용량을 확인하는 명령어

```
$ hadoop fs -dus
```

5. cat 명령어

- 지정된 파일의 내용을 확인하는 명령어

```
$ hadoop fs -cat /user/scott/emp3.csv
```

6. text 명령어

- 지정된 파일의 내용을 화면에 출력하는 명령어

```
$ hadoop fs -text /user/scott/emp3.csv
```

문제278. 하둡 파일 시스템의 리눅스 /home/scott 밑에 있는 Case.csv를

/user/scott 밑에 case3.csv로 올리시오.

$ hadoop fs -put /home/scott/Case.csv /user/scott/case3.csv

문제279. 하둡 파일 시스템에 올린 case3.csv의 내용을 조회하시오.

$ hadoop fs -cat /user/scott/case3.csv

7. mkdir 명령어
- 디렉토리를 생성하는 명령어
  $ hadoop fs -mkdir /user/scott/test

문제280. 위에서 만든 test 디렉토리가 잘 생성 되었는지 확인하시오.

$ hadoop fs -ls

문제281. 리눅스에 /home/scott 밑에 있는 emp2.csv를 /user/scott/test 에 emp2.csv 로 올리시오.

$ hadoop fs -put /home/scott/emp2.csv /user/scott/test/emp2.csv

8. put 명령어
- 하둡 파일 시스템에 파일을 올리는 명령어
  $ hadoo fs -put /home/scott/emp2.csv /user/scott/test/emp5.csv

9. get 명령어
- 하둡 파일 시스템에서 파일을 리눅스 디렉토리로 내리는 명령어
  $ cd
  $ rm emp2.csv
  $ ls -l emp2.csv
  $ hadoop fs -get /user/scott/test/emp2.csv /home/scott/emp2.scv

10. rm명령어
- 하둡 파일 시스템에 있는 파일을 지우는 명령어
  $ hadoop fs -lsr /user/
  $ hadoop fs -rm /user/scott/emp3.csv

문제282. 리눅스에 /home/scott 밑에있는 emp2.csv를 하둡의 /user/scott 밑에 emp3.csv로 올리시오.

$ hadoop fs -put /home/scott/emp2.csv /user/scott/emp3.csv

11. rmr 명령어
    - 하둡 파일 시스템의 디렉토리를 삭제하는 명령어
      $ hadoop fs -rmr /user/scott/test

문제283. 코로나 데이터중 감염자 정보중에 성별 데이터가 있는 csv 파일을 하둡 파일 시스템의 /user/scott 밑에 올리시오.

```
$ hadoop fs -put /home/scott/TimeGender.csv /user/scott/TimeGender.csv
```

12. grep 명령어
    - 파일에서 특정문자의 행의 데이터를 검색하는 명령어
      $ hadoop fs -cat /user/scott/emp7.csv | grep -i 'salesman'

문제284. 하둡 파일 시스템에 올린 코로나 데이터인 TimeGender.csv 파일을 cat으로 여시오.

```
$ hadoop fs -cat /user/scott/TimeGender.csv
```

문제285. 하둡 파일 시스템에 올린 코로나 데이터인 TimeGender.csv 파일에서 남자가 모두 몇명인지 확인하시오.

```
$ hadoop fs -cat /user/scott/TimeGender.csv | grep -i 'male' | wc -l
```

문제286. 판다스를 이용해서 원형 그래프를 그리시오.

1. 리눅스 /home/scott/ 밑에 있는 emp2.csv를 판다스로 로드해서 emp2.csv를 emp 데이터 프레임으로 만들어서 프린트 하시오.

```
import pandas as pd

emp = pd.read_csv('/home/scott/emp2.csv', header = None)

print (emp)
```

2. 직업과 직업별 인원수를 출력하시오.

```
import pandas as pd

emp = pd.read_csv('/home/scott/emp2.csv')
result = emp[2].value_counts()

print (result)
```

3. 직업과 직업별 인원수로 막대그래프를 그리시오.

```
import pandas as pd

emp = pd.read_csv('/home/scott/emp2.csv')
result = emp[2].value_counts()

result.plot(kind = 'bar')
```

kind 옵션

'pie' 원형 그래프 'line' 선그래프
'bar' 수직 막대그래프 'kde' 커널 밀도 그래프
'barh' 수평 막대그래프 'area' 면적 그래프
'his' 히스토그램 그래프 'scatter' 산점도 그래프
'box' 박스(사분위수) 그래프 'hexbin' 고밀도 산점도 그래프

문제287. 판다스로 직업, 직업별 토탈 월급을 출력하시오.

```
import pandas as pd

emp = pd.read_csv('/home/scott/emp2.csv')
result = emp.groupby('job')['sal'].sum()

print (result)
```

13. awk 명령어
    - 특정 컬럼을 검색하는 명령어
      $ hadoop fs -cat /user/scott/emp2.csv | grep -i 'scott' | awk -F "," '{print $2,$6}'

데이터가 콤마(,)로 구분되어 있으면 awk에 -F 옵션을 써서 ","로 구분 되어있다는 것을 알려줘야한다.

14. count 명령어
    - 지정된 디렉토리의 파의 갯수를 확인하는 명령어
      $ hadoop fs -lsr /user/scott

15. 하둡 파일 시스템 명령어 매뉴얼 보는 방법
      $ hadoop fs -help

타조 설치

타조에서 emp 테이블 생성

```
CREATE EXTERNAL TABLE emp (
 empno INT ,
 ename text ,
 job TEXT ,
 mgrno INT,
 hiredate text,
 sal INT,
 comm INT,
 deptno int
) USING CSV WITH ('text.delimiter'=',') LOCATION 'file:///home/scott/emp2.csv';
```

문제288. 월급이 1000에서 3000 사이인 사원들의 이름과 월급을 출력하시오.

```
orcl> select ename, sal
> from emp
> where sal between 1000 and 3000;
```

마리아 디비 설치

1. scott 에서 sudo 명령어를 수행할 수 있도록 설정

https://myjamong.tistory.com/3

2. scott 에서 maria 디비를 설치

http://cafe.daum.net/oracleoracle/Sfno/454

마리아 디비에서 emp 테이블 생성

```
MariaDB [(none)]> create database orcl;
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]> use orcl;
Database changed
```

mysql 테이블 스크...

문제289. DALLAS에서 근무하는 사원들의 이름과 부서위치를 출력하시오.

문제290. JONES 보다 더 많은 월급을 받는 사원들의 이름과 월급을 출력하시오.

```
MariaDB [orcl]> select ename, sal
   -> from emp
   -> where sal > (select sal
```

```
  -> from emp
  -> where ename='JONES');
```

문제291. 이름이 SCOTT인 사원의 월급을 9000으로 변경하시오.

hive는 데이터 수정이 안되지만 maria db는 가능하다.

```
MariaDB [orcl]> update emp
  -> set sal = 9000
  -> where ename = 'SCOTT';
```

마리아 db에서 autocommit 설정 확인하는 방법

```
MariaDB [orcl]> select @@autocommit;
```

AutoCommit 설정

```
MariaDB [orcl]> set autocommit = 1;
```

AutoCommit 해제

```
MariaDB [orcl]> set autocommit = 0;
```

마리아 db 접속 방법

```
[root@centos yum.repos.d]# mysql -u scott -p
Enter password:

MariaDB [(none)]> use orcl;
```

Maria db에서 파티션 테이블 생성하는 방법

1. 파티션 테이블이 필요한 이유 : 데이터가 대용량 데이터이면 테이블에서 데이터를 검색할 때 시간이 많이 걸린다. 인덱스를 생성해도 인덱스도 사이즈가 크기 때문에 인덱스를 통한 데이터 검색 효과를 볼 수가 없다. 이럴 때 파티션 테이블을 생성하면 된다.

ex)

emp 테이블을 부서번호마다 별도의 파티션에 데이터가 저장되게 파티션 테이블을 구성하시오.

부서번호 10번 : 1번 파티션

부서번호 20번 : 2번 파티션

부서번호 30번 : 3번 파티션

```
Create table emp_partition
(empno int(4) not null,
Ename varchar(10),
Job varchar(9),
Mgr int(4),
Hiredate date,
Sal int(7),
Comm int(7),
Deptno int(4) ) partition by list(deptno)
              (partition p1 values in (10),
               partition p2 values in (20),
```

```
            partition p3 values in (30) );

INSERT INTO emp_partition VALUES (7839,'KING','PRESIDENT',NULL,'81-11-17',5000,NULL,10);
INSERT INTO emp_partition VALUES (7698,'BLAKE','MANAGER',7839,'81-05-01',2850,NULL,30);
INSERT INTO emp_partition VALUES (7782,'CLARK','MANAGER',7839,'81-05-09',2450,NULL,10);
INSERT INTO emp_partition VALUES (7566,'JONES','MANAGER',7839,'81-04-01',2975,NULL,20);
INSERT INTO emp_partition VALUES (7654,'MARTIN','SALESMAN',7698,'81-09-10',1250,1400,30);
INSERT INTO emp_partition VALUES (7844,'TURNER','SALESMAN',7698,'81-08-21',1500,0,30);
INSERT INTO emp_partition VALUES (7900,'JAMES','CLERK',7698,'81-12-11',950,NULL,30);
INSERT INTO emp_partition VALUES (7521,'WARD','SALESMAN',7698,'81-02-23',1250,500,30);
INSERT INTO emp_partition VALUES (7902,'FORD','ANALYST',7566,'81-12-11',3000,NULL,20);
INSERT INTO emp_partition VALUES (7369,'SMITH','CLERK',7902,'80-12-09',800,NULL,20);
INSERT INTO emp_partition VALUES (7788,'SCOTT','ANALYST',7566,'82-12-22',3000,NULL,20);
INSERT INTO emp_partition VALUES (7876,'ADAMS','CLERK',7788,'83-01-15',1100,NULL,20);
INSERT INTO emp_partition VALUES (7934,'MILLER','CLERK',7782,'82-01-11',1300,NULL,10);
commit;
```

문제292. mariadb에서 emp_partition3 라는 이름으로 파티션 테이블을 생성하는데 피티션 키 컬럼을 job으로 해서 생성하시오.

```
create table emp_partition3
(empno int(4) not null,
ename varchar(10),
job varchar(9),
mgr int(4),
hiredate date,
sal int(7),
comm int(7),
deptno int(4) ) partition by list columns(job)
            (partition j1 values in ('ANALYST'),
             partition j2 values in ('CLERK'),
             partition j3 values in ('MANAGER'),
             partition j4 values in ('PRESIDENT'),
             partition j5 values in ('SALESMAN') );

INSERT INTO emp_partition3 VALUES (7839,'KING','PRESIDENT',NULL,'81-11-17',5000,NULL,10);
INSERT INTO emp_partition3 VALUES (7698,'BLAKE','MANAGER',7839,'81-05-01',2850,NULL,30);
INSERT INTO emp_partition3 VALUES (7782,'CLARK','MANAGER',7839,'81-05-09',2450,NULL,10);
INSERT INTO emp_partition3 VALUES (7566,'JONES','MANAGER',7839,'81-04-01',2975,NULL,20);
INSERT INTO emp_partition3 VALUES (7654,'MARTIN','SALESMAN',7698,'81-09-10',1250,1400,30);
INSERT INTO emp_partition3 VALUES (7844,'TURNER','SALESMAN',7698,'81-08-21',1500,0,30);
INSERT INTO emp_partition3 VALUES (7900,'JAMES','CLERK',7698,'81-12-11',950,NULL,30);
INSERT INTO emp_partition3 VALUES (7521,'WARD','SALESMAN',7698,'81-02-23',1250,500,30);
INSERT INTO emp_partition3 VALUES (7902,'FORD','ANALYST',7566,'81-12-11',3000,NULL,20);
INSERT INTO emp_partition3 VALUES (7369,'SMITH','CLERK',7902,'80-12-09',800,NULL,20);
INSERT INTO emp_partition3 VALUES (7788,'SCOTT','ANALYST',7566,'82-12-22',3000,NULL,20);
INSERT INTO emp_partition3 VALUES (7876,'ADAMS','CLERK',7788,'83-01-15',1100,NULL,20);
INSERT INTO emp_partition3 VALUES (7934,'MILLER','CLERK',7782,'82-01-11',1300,NULL,10);
commit;
```

21.01.14

2021년 1월 14일 목요일 오전 9:43

시스템 구축

1. 리눅스 centos 설치
2. 하둡 설치
3. hive 설치
4. 스칼라 설치
5. 파이썬 아나콘다 설치
6. maria db 설치
7. 부가적인 작업
    a. 원격에서 putty로 접속 할 수 있도록 설정
    b. mobaxterm을 이용하여 파일 전송과 윈도우 에서 리눅스의 spyder를 실행

구현한 리눅스 서버를 활용하여 데이터 시각화 :

1. 리눅스와 하둡 수업의 시험 평가물 제출 포멧
2. 주식 데이터를 웹 스크롤링 해서 주식 데이터를 마리아 디비에 저장하고 파이썬과 연동해서 데이터 시각화

마리아 디비와 파이썬 연동

http://cafe.daum.net/oracleoracle/Sfno/469

문제293. 마리아 디비와 파이썬 연동된 상태에서 spyder에서 이름과 월급을 출력하시오.

```
import mysql.connector

config = {
   "user": "root",
   "password": "1234",
   "host": "192.168.56.101", #local
   "database": "orcl", #Database name
   "port": "3456" #port는 최초 설치 시 입력한 값(기본값은 3306)
}

conn = mysql.connector.connect(**config)

# db select, insert, update, delete 작업 객체
cursor = conn.cursor()

# 실행할 select 문 구성
```

```
sql = """SELECT *
                from emp
                order by 1 desc"""

# cursor 객체를 이용해서 수행한다.
cursor.execute(sql)

# select 된 결과 셋 얻어오기
resultList = cursor.fetchall()  # tuple 이 들어있는 list

import  pandas   as  pd

emp = pd.DataFrame(resultList)
emp.columns=['empno', 'ename', 'job', 'mgr', 'hiredate', 'sal', 'comm', 'deptno']
print (emp[['ename', 'sal']] )
```

문제294. 월급이 3000 이상인 사원들의 이름과 월급을 출력하시오.

```
import mysql.connector

config = {
   "user": "root",
   "password": "1234",
   "host": "192.168.56.101", #local
   "database": "orcl", #Database name
   "port": "3456" #port는 최초 설치 시 입력한 값(기본값은 3306)
}

conn = mysql.connector.connect(**config)

# db select, insert, update, delete 작업 객체
cursor = conn.cursor()

# 실행할 select 문 구성
sql = """SELECT *
                from emp
                order by 1 desc"""

# cursor 객체를 이용해서 수행한다.
cursor.execute(sql)

# select 된 결과 셋 얻어오기
resultList = cursor.fetchall()  # tuple 이 들어있는 list

import  pandas   as  pd

emp = pd.DataFrame(resultList)
emp.columns=['empno', 'ename', 'job', 'mgr', 'hiredate', 'sal', 'comm', 'deptno']
print ( emp[['ename', 'sal']] [ emp['sal'] >= 3000 ] )
```

문제295. 직업과 직업별 토탈 월급을 출력하시오.

```
import mysql.connector

config = {
   "user": "root",
   "password": "1234",
   "host": "192.168.56.101", #local
   "database": "orcl", #Database name
   "port": "3456" #port는 최초 설치 시 입력한 값(기본값은 3306)
}

conn = mysql.connector.connect(**config)

# db select, insert, update, delete 작업 객체
cursor = conn.cursor()

# 실행할 select 문 구성
sql = """SELECT job, sum(sal)
                from emp
                group by job"""

# cursor 객체를 이용해서 수행한다.
cursor.execute(sql)

# select 된 결과 셋 얻어오기
resultList = cursor.fetchall()  # tuple 이 들어있는 list

import  pandas  as  pd

emp = pd.DataFrame(resultList)
emp.columns=['job','sumsal']
print(emp)
```

문제296. 직업, 직업별 토탈월급을 출력하는데 직업이 SALESMAN은 제외하고 직업별 토탈월급이 4000 이상인 것만 출력하는데 직업별 토탈월급이 높은 것부터 출력하시오.

```
import mysql.connector

config = {
   "user": "root",
   "password": "1234",
   "host": "192.168.56.101", #local
   "database": "orcl", #Database name
   "port": "3456" #port는 최초 설치 시 입력한 값(기본값은 3306)
}

conn = mysql.connector.connect(**config)
```

```
# db select, insert, update, delete 작업 객체
cursor = conn.cursor()

# 실행할 select 문 구성
sql = """SELECT job, sum(sal)
                from emp
                where  job !='SALESMAN'
                group by job
                having  sum(sal) >= 4000
                order  by  2  desc"""

# cursor 객체를 이용해서 수행한다.
cursor.execute(sql)

# select 된 결과 셋 얻어오기
resultList = cursor.fetchall()  # tuple 이 들어있는 list

import  pandas   as  pd

emp = pd.DataFrame(resultList)
emp.columns=['job','sumsal']
print(emp)
```

문제297. 부서위치, 부서위치별 토탈월급을 출력하시오.

```
import mysql.connector

config = {
   "user": "root",
   "password": "1234",
   "host": "192.168.56.101", #local
   "database": "orcl", #Database name
   "port": "3456" #port는 최초 설치 시 입력한 값(기본값은 3306)
}

conn = mysql.connector.connect(**config)

# db select, insert, update, delete 작업 객체
cursor = conn.cursor()

# 실행할 select 문 구성
sql = """SELECT d.loc, sum(e.sal)
         from emp e join dept d
         on (e.deptno = d.deptno)
         group by d.loc"""

# cursor 객체를 이용해서 수행한다.
cursor.execute(sql)

# select 된 결과 셋 얻어오기
```

```
resultList = cursor.fetchall()  # tuple 이 들어있는 list

import  pandas   as  pd

emp = pd.DataFrame(resultList)
emp.columns=['loc','sumsal']
print(emp)
```

문제298. 위의 결과를 막대 그래프로 시각화 하시오.

```
import mysql.connector

config = {
   "user": "root",
   "password": "1234",
   "host": "192.168.56.101", #local
   "database": "orcl", #Database name
   "port": "3456" #port는 최초 설치 시 입력한 값(기본값은 3306)
}

conn = mysql.connector.connect(**config)

# db select, insert, update, delete 작업 객체
cursor = conn.cursor()

# 실행할 select 문 구성
sql = """SELECT d.loc, sum(e.sal)
         from emp e join dept d
         on (e.deptno = d.deptno)
         group by d.loc"""

# cursor 객체를 이용해서 수행한다.
cursor.execute(sql)

# select 된 결과 셋 얻어오기
resultList = cursor.fetchall()  # tuple 이 들어있는 list

import  pandas   as  pd

emp = pd.DataFrame(resultList)
emp.columns=['loc','sumsal']

result = emp['sumsal'].astype(int)
result.index = emp['loc']
result.plot (kind='bar', color='red')
```

문제299. 현업에서 많이 사용하는 마리아 디비와 mySQL을 편하게 사용할 수 있도록 하는 오라클의 sqldeveloper와 같은 툴을 설치하고 리눅스 서버의 마리아 디비의 emp 테이블을 조회하시오.

http://cafe.daum.net/oracleoracle/Sfno/477

문제300. workbench 에서 아래의 문제 결과를 출력하시오.
직업, 직업별 평균 월급을 출력하는데 집업별 평균월급이 2000 이상인 것만 출력하고 직업별 평균월급이 높은 것 부터 출력하시오.

```
select job, avg(sal)
        from emp
   group by job
   having avg(sal) > 2000
   order by avg(sal) desc;
```

마리아디비의 장점

1. 무료
2. mySQL과 기능이 같다.
3. 리눅스의 csv, text파일도 로드 할 수 있다.

리눅스의 csv 파일을 마리아 db로 로드하는 방법

http://cafe.daum.net/oracleoracle/Sfno/478

문제301. 도시명, 도시명별 토탈 감염자수를 출력하시오.

```
select city, sum(confirmed)
        from cov_case
        group by city;
```

문제302. 위의 결과를 다시 출력하는데 도시명이 null이 아닌것만 출력하시오.

```
select city, sum(confirmed)
        from cov_case
        where trim(city) is not null and city != '-'
         group by city;
```

trim을 사용하면 양쪽 공백을 잘라 버리겠다.

문제303. 위의 결과를 다시 출력하는데 토탈 확진자수가 높은 것부터 출력하시오.

```
select city, sum(confirmed)
```

```
        from cov_case
        where trim(city) is not null and city != '-'
        group by city
        order by 2 desc;
```

문제304. 위의 결과를 다시 출력하는데 토탈 확진자수가 100명 이상인 것만 출력하시오.

```
select city, sum(confirmed)
        from cov_case
        where trim(city) is not null and city != '-'
        group by city
        having sum(confirmed) >= 100
        order by 2 desc;
```

문제305. 위의 결과를 막대 그래프로 시각화 하시오.

```
import mysql.connector
import  pandas as  pd

config = {
   "user": "root",
   "password": "1234",
   "host": "192.168.56.101", #local
   "database": "orcl", #Database name
   "port": "3456" #port는 최초 설치 시 입력한 값(기본값은 3306)
}

conn = mysql.connector.connect(**config)

# db select, insert, update, delete 작업 객체
cursor = conn.cursor()

# 실행할 select 문 구성
sql = """ SELECT ifnull(city,'esc'),  round(sum(confirmed)) sum2
      FROM cov_case
      where trim(city) is not null and city !='-'
      group by city
      having sum(confirmed) > 50
      order by sum2 desc
        """

# sql을 실행해서 cursor(메모리) 에 담는다.
cursor.execute(sql)

# cursor에 담긴 과 셋 얻어오기
resultList = cursor.fetchall()  # tuple 이 들어있는 list
df = pd.DataFrame(resultList)  # 판다스 데이터 프레임으로 변환
df.columns = ['city', 'cnt']  # 컬럼명을 만든다.
```

```
print(df[['city','cnt']])

# 시각화
result = df['cnt']
result.index = df['city']
result.plot(kind='bar', color='red')
```

os의 csv 파일을 maria 디비에 넣고 SQL로 데이터 전처리를 한 후 파이썬으로 시각화 한다.

문제306. province를 중복 제거해서 출력하시오.

```
select distinct province
        from cov_case;
```

문제307. 지역(province), 지역별 토탈 확진자수를 막대 그래프로 시각화 하시오.

```
import mysql.connector
import  pandas as  pd

config = {
   "user": "root",
   "password": "1234",
   "host": "192.168.56.101", #local
   "database": "orcl", #Database name
   "port": "3456" #port는 최초 설치 시 입력한 값(기본값은 3306)
}

conn = mysql.connector.connect(**config)

# db select, insert, update, delete 작업 객체
cursor = conn.cursor()

# 실행할 select 문 구성
sql = """ select distinct province, sum(confirmed)
        from cov_case
   group by province
   order by 2 desc;
        """

# cursor 객체를 이용해서 수행한다.
cursor.execute(sql)

# select 된 결과 셋 얻어오기
resultList = cursor.fetchall()  # tuple 이 들어있는 list
df = pd.DataFrame(resultList)
df.columns = ['city', 'cnt']
print(df[['city','cnt']])
```

시각화

```
result = df['cnt']
result.index = df['city']
result.plot(kind='bar', color='red')
```

하둡과 리눅스 ncs 시험 제출물로 대체

제출물 만드는 순서

1. 큰 질문 : 코로나는 남자와 여자중에 누가 더 많이 감염되었는가 ?
2. csv 데이터 구한다.
3. 마리아 디비에 테이블로 생성한다.
4. 파이썬에서 시각화를 한다.
5. 시각화 그래프와 함께 짧게 시각화한 결과를 설명

제출 형식 : 카페에 제출물 게시판에 다음주 금요일 까지 올리기

문제308. 코로나는 남자와 여자중에 누가 더 많이 감염되었는지 확인하시오.

1. 큰 질문 : 코로나는 남자와 여자중에 누가 더 많이 감염되었는가 ?
2. csv 데이터 구한다.

- $ head PatientInfo.csv

3. 마리아 디비에 테이블로 생성한다.

```
drop  table  pati;

create   table  pati
( patient_id     float,
  sex          varchar(30),
  age            varchar(30),
  country       varchar(30),
  province       varchar(30),
  city            varchar(30),
  infection_case     varchar(60),
  infected_by        varchar(60),
  contact_number     varchar(30),
  symptom_onset_date   varchar(30),
  confirmed_date    varchar(30),
  released_date    varchar(30),
  deceased_date    varchar(30),
  state           varchar(30) ) ;


LOAD DATA LOCAL INFILE '/home/scott/PatientInfo.csv'
REPLACE
```

```
INTO TABLE  orcl.pati
fields TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES
(patient_id,sex,age,country,province,city,infection_case,infected_by,contact_number,symptom_on
set_date,confirmed_date,released_date,deceased_date,state );
```

성별, 성별별 인원수를 출력

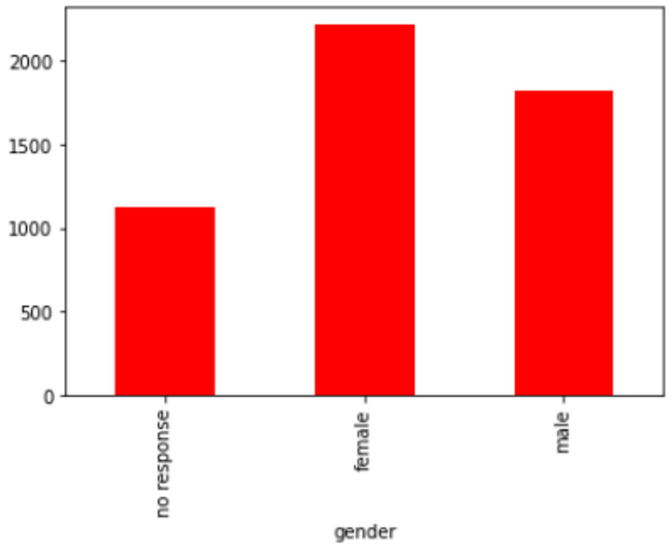
```
select case when sex = ' ' then 'no response'
        else sex end gender, count(*)
                from pati
                group by sex;
```

4. 파이썬에서 시각화를 한다.

```
import mysql.connector
import  pandas as  pd

config = {
   "user": "root",
   "password": "1234",
   "host": "192.168.56.101", #local
   "database": "orcl", #Database name
   "port": "3456" #port는 최초 설치 시 입력한 값(기본값은 3306)
}

conn = mysql.connector.connect(**config)

# db select, insert, update, delete 작업 객체
cursor = conn.cursor()

# 실행할 select 문 구성
sql = """ select case when sex = ' ' then 'no response'
        else sex end gender, count(*)
                from pati
                group by sex;
        """

# cursor 객체를 이용해서 수행한다.
cursor.execute(sql)

# select 된 결과 셋 얻어오기
resultList = cursor.fetchall()  # tuple 이 들어있는 list
df = pd.DataFrame(resultList)
df.columns = ['gender', 'cnt']
print(df[['gender','cnt']])

# 시각화
result = df['cnt']
```

```
result.index = df['gender']
result.plot(kind='bar', color='red')
```

5. 시각화 그래프와 함께 짧게 시각화한 결과를 설명



설명 : 확진자 수는 남자는 몇명이고 여자는 몇명이고 무응답은 몇명이었습니다.
그래프를 보면 남자보다는 여자가 더 많이 확진 되었음을 확인할 수 있습니다.

공공 데이터 포털:
https://www.data.go.kr/
캐글:
https://www.kaggle.com/datasets