==========하둡 수업==========

■ 하둡 수업

* 하둡 목차

1. 하둡을 배워야 하는 이유
2. 하둡 설치
3. 하둡 분산 파일 시스템 명령어
4. Hive
5. Pig
6. Tajo
7. sqoop으로 오라클과 연동
8. 맵리듀스를 java로 수행하기

■ 1. 하둡을 배워야하는 이유

하둡 ? 대용량 데이터를 분산 처리할 수 있는 자바기반의 오픈소스 프레임워크

구글에서 구글에 쌓여지는 수많은 빅데이터(웹페이지, 데이터..)들을 구글에서도 처음에는 RDBMS(오라클)에 입력하고 데이터를 저장하고 처리하려는 시도를 했으나 너무 데이터가 많아서 실패를 하고 자체적으로 빅데이터를 저장할 기술을 개발을 했다.

대외적으로는 논문을 하나 발표했다.
그 논문을 더그커팅(하둡을 만든이)이 읽고 자바로 구현을 했다.
그 이름을 뭘로 할까 고민을 하다가 더그커팅의 애기가 노란 코끼리 장난감을 가지고 놀면서 Hadoop이라고 한걸 듣고 hadoop이라고 이름을 지었다.

그래서 그 뒤로 hadoop을 편하게 이용할 수 있도록 개발한 모든 하둡 생태계에 개발 프로그램 이름들이 다 동물 이름으로 지어지게 되었다.

Hadoop ----------->> Hive (벌떼)
Pig (돼지)
Tajo (타조)

하둡의 장점? 분산처리가 가능하다.
↓
여러대의 노드를 묶어서 마치 하나의 서버처럼 보이게 하고 여러 노드의 자원을 이용해서 데이터를 처리하기 때문에 처리하는 속도가 빠른 장점이 있다.

예 : 한대의 서버로 1테라 바이트의 데이터를 처리하는데 걸리는 시간이 2시간 반이 걸린다고 한다면
하둡으로 여러대의 서버를 병렬로 작업한다면 2분내에 데이터를 읽을 수 있다.

예 : 2008년 뉴욕 타임즈의 130년 분량의 신문기사 1100만 페이지를 하둡을 이용해서 하루만에 pdf로 변환을 했고 비용이 200만원 밖에 안들었다.
하둡이 아닌 일반 서버로 처리했다면 14년이 걸린다.

★ 하둡 에코 시스템

```
빅데이터 분석                R, Python 등을 이용해서 분석
     ↑
   NoSQL                     Hbase Membase Cassandra
(빅데이터 저장)                      Redis MongoDB CouchDB
     ↑                           ↑
분산 처리지원                 Hive Pig Sqoop Zookeeper
     ↑                           ↑
분산 배치처리                 하둡(Hadoop) - MapReduce
분산 파일관리                 하둡(Hadoop) - HDFS
```

★ Hive 사용 예

1.

```
SQL> select * from emp;

hive> select * from emp;
```

2.

```
SQL> select * from emp where rownum <= 5;
hive> select * from emp limit 5;
        ↓
     NoSQL ------------> Not only SQL
```

빅데이터를 하둡에서 검색하기 위해서는 자바를 알아야 하는데 자바를 모르더라도 SQL과 비슷한 언어로 빅데이터를 검색할 수 있게 지원해주는 언어

★ 하둡 구성

데이터의 위치 정보
↑
네임노드(Meta 데이터 관리)

데이터 노드 ----- 데이터 노드 ------ 데이터 노드
(실 data) (실 data) (실 data)

총 3군데 노드에 각각 데이터를 복제를 한다.

야후의 경우는 약 5만대를 연결해서 하둡을 사용.
페이스북은 약 1만대 이상의 하둡 클러스터를 이용

아마존, 이베이, 마이스페이스, 야후, NHN, Daum, kt, skt..... 등이 하둡을 사용하고 있다.

- 하둡의 장점 : 저렴한 구축 비용과 비용대비 빠른 데이터 처리

- 하둡의 단점 : 1. 무료이다 보니 유지보수가 어렵다.
2. 네임노드가 다운되면 고가용성이 지원이 안된다.
3. 한번 저장한 파일을 수정할 수 없다.
기존 데이터 append는 가능한데 수정은 안된다.

정형화된 데이터 (표 형태) 비정형화된 데이터
↓(저장) ↓(저장)
오라클 하둡

* 하둡을 설치하기 위해서 리눅스 시스템에 올려야 하는 파일(oracle 홈디렉토리에)

1. hadoop-1.2.1.tar.gz
2. jdk-7u60-linux-i586.gz
3. hive-0.12.0.tar.gz

* 하둡 주요 배포판

리눅스도 centos, redhat, ubuntu 처럼 배포판이 있듯이 하둡도 배포판이 있다.

1. Cloudera 업체에서 나온 CDH <---- 가장 높은 신뢰
2. Hortonworks에서 나온 HDP
3. 아마존에서 나온 EMR(Elastic Mapreduce)
4. Hstreaming에서 나온 Hstreaming

* 하둡 홈페이지
https://hadoop.apache.org/

★ 하둡하면 머리속에 기억하고 있어야할 큰 내용

1. NoSQL : (1) hive ----------> big data (의료용 데이터, 갤럭시 7 핸드폰의 로그데이터
(2) Tajo sns에 올리는 글들)
(3) pig
(4) mongodb

2. RDBMS : (1) Oracle ----------> business data (small data)
(2) MySQL

■ 하둡 설치 (실습_00.Hadoop install.txt 참고)

1. java 설치 :
   하둡이 자바로 만들어져 있기 때문에 java를 설치해야 한다.

2. java 환경설정
   java 홈디렉토리가 어디다 라고 지정

3. keygen 생성
   여러노드(서버)들을 묶어서 하둡을 운영할 것이기 때문에 내 컴퓨터에서 상대방 컴퓨터로 접속 할때 패스워드를 매번 물어보지 않고 그냥 바로 접속하게 하려면 keygen을 생성해야 한다.

4. 하둡 설치파일을 올린 후 압축을 푼다.
   hadoop-1.2.1.tar.gz

5. 하둡을 홈 디렉토리로 설정한다

6. 하둡을 운영하기 위한 xml파일 3개를 수정한다.
   1. core-site.xml
   2. mapred-site.xml
   3. hdfs-site.xml

7. 하둡 네임노드를 포멧한다.
8. 하둡을 시작시킨다.
9. 하둡이 잘 시작되었는지 확인한다.

리눅스 시스템을 켜고 하둡시스템을 이용하려면
[orcl:~]$ start-all.sh <--- 이 명령어를 날려야 한다.

★ Hive 설치

하이브란 ? 페이스북에서 만든 오픈소스

java를 몰라도 rdbms 에 익숙한 데이터 분석가들을 위해서 SQL을 이용해서 멥리듀싱 프로그래밍을 지원하는 프로그램

SQL ----------> java
하이브

emp.csv 를 가지고 하둡에서 이름과 월급을 조회하려면 java를 알아야 한다.

* HiveQL 이란?
- 하이브에서 사용되는 데이터는 HDFS(하둡파일 시스템)
에 저장되므로 SELECT 는 되는데 update, delete명령어는 지원안함

- from 절의 서브쿼리 사용가능 (in line view)
- select문 사용시 having절 사용 불가능
- Stored procedure(PL/SQL)는 지원안함 (Hive 2.0 가능)

오라클의 PL/SQL인 프로시져를 사용하려면 Java로 해야한다.

★ 설치

1. hive 설치파일의 압축을 푼다.

```
[orcl:~]$ tar xvzf hive-0.12.0.tar.gz
```

2. hive로 접속한다.

```
[orcl:~]$ cd /home/oracle/hive-0.12.0/bin
[orcl:bin]$ ./hive

hive> show tables;
OK
Time taken: 5.011 seconds
```

3. emp 테이블을 생성한다.

- /home/oracle/밑에 emp2.csv를 올린다.

```
# cd /media/sf_Share
# ls
# chown -R oracle:oinstall emp2.csv
# cp emp2.csv /home/oracle/
```

- hive에서 emp 테이블을 생성한다.

```
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 ;
```

```
        OK
        Time taken: 4.568 seconds
```

- emp.csv 를 emp테이블에 로드한다.
 하둡 파일 시스템에 emp2.csv 를 올리시오 !

```
$ . .bash_profile
$ hadoop  fs  -put  emp2.csv  emp2.csv
$ hadoop  fs  -ls  emp2.csv
 emp2.csv 를 /home/oracle 에 올린다.

hive> load  data  inpath  '/user/oracle/emp2.csv'
    > overwrite  into  table   emp;
Loading data to table default.emp
Time taken: 4.098 seconds
hive> select  * from  emp;
OK
7839    KING    PRESIDENT       0       1981-11-17      5000    0       10
7698    BLAKE   MANAGER 7839    1981-05-01      2850    0       30
7782    CLARK   MANAGER 7839    1981-05-09      2450    0       10
7566    JONES   MANAGER 7839    1981-04-01      2975    0       20
7654    MARTIN  SALESMAN        7698    1981-09-10      1250    1400    30
7499    ALLEN   SALESMAN        7698    1981-02-11      1600    300     30
7844    TURNER  SALESMAN        7698    1981-08-21      1500    0       30
7900    JAMES   CLERK   7698    1981-12-11      950     0       30
7521    WARD    SALESMAN        7698    1981-02-23      1250    500     30
7902    FORD    ANALYST 7566    1981-12-11      3000    0       20
7369    SMITH   CLERK   7902    1980-12-09      800     0       20
7788    SCOTT   ANALYST 7566    1982-12-22      3000    0       20
7876    ADAMS   CLERK   7788    1983-01-15      1100    0       20
7934    MILLER  CLERK   7782    1982-01-11      1300    0       10
Time taken: 0.294 seconds, Fetched: 14 row(s)
```

어느 디렉토리에서든 하이브에 들어갈 수 있게 시스템에 경로를 설정한다.

```
[orcl:~]$ vi .bash_profile

========================================
추가
export HIVE_HOME=/home/oracle/hive-0.12.0
export PATH=$HIVE_HOME/bin:$PATH
========================================

[orcl:~]$ . .bash_profile
```

■ 3. 하둡 분산 파일 시스템 명령어

문제 2 ~ 18

1. ls -> 지정된 디렉토리에 있는 파일의 정보를 출력
2. lsr -> 현재 디렉토리 뿐만 아니라 하위 디렉토리까지 조회
3. du -> 파일의 용량 확인
4. dus -> 파일의 전체 합계 용량 확인
5. cat -> 지정된 파일의 내용을 화면에 출력
6. text -> zip 파일 형태도 text 형대로 화면에 출력
7. mkdir -> 디렉토리 생성
8. put -> 파일을 하둡 파일 시스템에 올리는 명령어
9. copyFromLocal -> 파일 복사
10. get -> 하둡 파일 시스템의 파일을 리눅스 디렉토리로 내리는 명령어
11. getmerge -> 지정된 경로에 있는 모든 파일의 내용을 합친후 하나의 파일로 복사하는 명령어
12. mv -> 파일을 이동하는 명령어
13. moveFromLocal -> 복사후 원본 파일 삭제
14. rm -> 파일 삭제
15. rmr -> 디렉토리 삭제
16. count -> 지정된 디렉토리의 파일의 갯수 확인
17. tail -> 파일의 마지막 내용 확인
18. chmod -> 권한 변경
19. chown -> 소유자 변경
20. tochz -> 0 바이트 파일 생성
21. stat -> 통계정보 조회
22. expuge -> 휴지통 비우기
23. grep -> 파일에서 특정 문자의 라인을 검색
24. awk -> 파일의 특정 컬럼을 검색

★ hive에서 data 분석함수 사용하기

문제 19 ~ 22

★ 오라클과 hive의 함수 비교

문제 23 ~ 39

1. 문자함수

오라클	vs	hive
upper		upper
lower		lower
initcap		x
substr		substr
concat		concat
trim		trim
ltrim		ltrim
rtrim		rtrim
lpad		x
rpad		x

2. 숫자함수

round	round
trunc	x
power	power

3. 날짜함수

months_between	months_between
add_months	add_months

next_day	next_day
last_day	last_day

4. 변환함수

to_date	to_date
to_char	cast
to_number	cast
	year
	month
	day

■ Tajo 설치 방법 정리

문제 40 ~ 56

0. /media/sf_Share 밑에 있는 타조 설치 파일을 /home/oracle 밑으로 복사한다.

```
# cd  /media/sf_Share

# cp tajo-0.11.1.tar.gz  /home/oracle/
```

1. 타조 설치파일의 소유자를 oracle 로 변경한다.

```
# chown -R  oracle:oinstall  tajo-0.11.1.tar.gz
```

2. oracle 유저에서 타조 설치 파일의 압축을 푼다.

```
$ tar -xvzf tajo-0.11.1.tar.gz
```

3. tajo 디렉토리가 생겼는지 확인한다.

```
$ ls -ld tajo*
```

4. tajo 홈디렉토리의 이름을 tajo 로 변경한다.

```
$ mv tajo-0.11.1-desktop-r1 tajo
```

5. tajo 디렉토리로 들어가서 conf 디렉토리로 간다.

```
$ cd  tajo
```

6. tajo 환경설정을 한다.

```
$ cd tajo

$ ls -ld  bin

$ sh bin/configure.sh

Enter JAVA_HOME [default: /u01/app/java/jdk1.7.0_60]

그냥 엔터

Would you like advanced configure? [y/N]  N
                                          ↑
                                    N이라고 하고 엔터
```

7. tajo 를 시작 시킨다.

```
$ sh bin/startup.sh
```

8. tajo 로 접속을 한다.

```
[orcl:tajo]$ sh bin/tsql
```

9. orcl 라는 데이터 베이스를 만들고 orcl 데이터베이스에 접속한다.

```
default> create database orcl;

default> Wc orcl;

You are now connected to database "orcl" as user "oracle".
```

10. emp 테이블을 생성한다.

```
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/oracle/emp2.csv';
```

11. emp 테이블을 조회한다.

```
orcl> select * from emp;
```

★ tajo 에서 한글 깨지지 않게 하는 방법
문제 57 ~ 61

```
$ cd  /home/oracle/tajo/tajo/conf

$ vi  tajo-env.sh

맨 아래에 아래의 내용을 추가

LANG="ko_KR.UTF-8"
export LANG="ko_KR.UTF-8"

$ 저장하고 나와서 다시 타조 접속
```

■ 6. pig

문제 62 ~ 79

1. 야후에서 만든 NoSQL
   (야후에서는 40%이상의 job을 pig로 처리한다.)

2. "돼지들은 어떠한 것도 잘 먹는다" 라는 슬로건을 갖는다.
   어떠한 데이터든 잘 소화할 수 있다.

3. 사용자 정의 함수(오라클의 프로시저와 같은 언어)를 지원한다. (SQL + 프로그래밍 언어)
   ↓
   엑셀의 매크로와 같은 기능

★ Pig 설치

1. pig 설치 파일을 /home/oracle/ 밑에 올린다.

pig-0.12.0.tar.gz

2. pig 설치 파일의 압축을 푼다.

```
$ tar xvf pig-0.12.0.tar.gz
```

3. pig 환결설정을 한다.

```
$ mv pig-0.12.0 pig
$ cd pig
$ cd conf
$ vi pig.properties
```

```
fs.default.name=hfdf://localhost:9000
mapred.job.tracker=localhost:9000
```

4. .bash_profile 에 PIG_HOME 디렉토리를 지정한다.

```
$ cd
$ vi .bash_profile
```

```
export PIG_HOME=/home/oracle/pig
export PATH=$PIG_HOME/bin:$PATH
```

5. . .bash_profile 을 실행한다.

```
$ . .bash_profile
```

6. pig 에 접속한다.

```
$ pig -x local
```

■ Mongo DB

★ 우분투 설치

ubuntu-16.04.2-desktop-amd64

★ Mongo DB 설치 (우분투 os)
문제 81 ~ 84

https://velopert.com/436 블로그를 참고한다.

1. MongoDB Pulbic GPG Key 등록

```
$ sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv EA312927
```

2. MongoDB 를 위한 list file 생성 (자신의 Ubuntu 버전에 맞게 입력하세요)

```
# Ubuntu 14.04
$ echo "deb http://repo.mongodb.org/apt/ubuntu xenial/mongodb-org/3.2 multiverse" | sudo tee/etc/apt/sources.list.d/mongodb-org-3.2.list
```

3. apt-get 을 이용하여 설치

```
$ sudo apt-get update
# latest stable version 설치
$ sudo apt-get install -y mongodb-org
```

4. Next, start MongoDB with systemctl

```
$ sudo systemctl start mongod
```

5. 서버가 제대로 실행됐는지 확인

```
$ sudo systemctl status mongod
# [initandlisten] waiting for connections on port <port>
$ sudo systemctl enable mongod
```

6. MongoDB 서버 접속

```
$ mongo

MongoDB shell version: 3.2.22
connecting to: test
```

★ mongodb에서 사용하는 연산자
문제 85 ~ 111

1. 비교 연산자

$eq	=	같다
$gt	>	크다
$gte	>=	크거나 같다
$lt	<	작다
$lte	<=	작거나 같다

$ne != 같지 않다.

2. 논리 연산자
$and
$or
$not

3. 산술 연산자
$add 더하기
$subtract 빼기
$multiply 곱하기
$devide 나누기
$mod 나머지

★ Mongodb를 툴에서 사용하는 방법

https://www.robomongo.org/
↓
Robo 3T 다운로드 (robo3t-1.2.1-windows-x86_64-3e50a65.exe)

★ mongodb에서의 조인문장
문제 113 ~ 115

★ Mongodb에서의 DML문
문제 116 ~ 119
1. insert
2. update
3. delete

■ MySQL

문제 120 ~ 124
설치법 : https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-centos-7 참고

```
drop table emp;
drop table dept;

CREATE TABLE dept
(deptNO int,
DNAME VARCHAR(14),
LOC VARCHAR(13) );

INSERT INTO dept VALUES (10, 'ACCOUNTING', 'NEW YORK');
INSERT INTO dept VALUES (20, 'RESEARCH', 'DALLAS');
INSERT INTO dept VALUES (30, 'SALES', 'CHICAGO');
INSERT INTO dept VALUES (40, 'OPERATIONS', 'BOSTON');

CREATE TABLE emp (
empNO int NOT NULL,
ENAME VARCHAR(10),
JOB VARCHAR(9),
MGR int,
HIREDATE DATE,
SAL int,
COMM int,
deptNO int );

INSERT INTO emp VALUES (7839,'KING','PRESIDENT',NULL,'81-11-17',5000,NULL,10);
INSERT INTO emp VALUES (7698,'BLAKE','MANAGER',7839,'81-05-01',2850,NULL,30);
INSERT INTO emp VALUES (7782,'CLARK','MANAGER',7839,'81-05-09',2450,NULL,10);
INSERT INTO emp VALUES (7566,'JONES','MANAGER',7839,'81-04-01',2975,NULL,20);
INSERT INTO emp VALUES (7654,'MARTIN','SALESMAN',7698,'81-09-10',1250,1400,30);
INSERT INTO emp VALUES (7499,'ALLEN','SALESMAN',7698,'81-02-11',1600,300,30);
INSERT INTO emp VALUES (7844,'TURNER','SALESMAN',7698,'81-08-21',1500,0,30);
INSERT INTO emp VALUES (7900,'JAMES','CLERK',7698,'81-12-11',950,NULL,30);
INSERT INTO emp VALUES (7521,'WARD','SALESMAN',7698,'81-02-23',1250,500,30);
INSERT INTO emp VALUES (7902,'FORD','ANALYST',7566,'81-12-11',3000,NULL,20);
INSERT INTO emp VALUES (7369,'SMITH','CLERK',7902,'80-12-09',800,NULL,20);
INSERT INTO emp VALUES (7788,'SCOTT','ANALYST',7566,'82-12-22',3000,NULL,20);
INSERT INTO emp VALUES (7876,'ADAMS','CLERK',7788,'83-01-15',1100,NULL,20);
INSERT INTO emp VALUES (7934,'MILLER','CLERK',7782,'82-01-11',1300,NULL,10);

commit;
```

RDBMS : 1. Oracle
2. MySQL

★ Oracle vs MySQL의 차이를 총정리
문제 125 ~ 149

	oracle	vs	MySQL
1.	nvl		ifnull
2.	sysdate		sysdate()
3.	months_between		period_diff
4.	add_months		period_add
5.	last_day		last_day

6. to_char format
7. decode if
8. case case
9. rollup with rollup
10. rank, dense rank 지원안함

★ MySQL에서 DDL문
- 오라클과의 차이점
external table의 관리가 오라클과 다르다.

```
mysql> CREATE TABLE emp2 (
    -> EMPNO int NOT NULL,
    -> ENAME VARCHAR(10),
    -> JOB VARCHAR(9),
    -> MGR int,
    -> HIREDATE DATE,
    -> SAL int,
    -> COMM int,
    -> DEPTNO int );
Query OK, 0 rows affected (0.04 sec)

mysql> load  data  local infile '/home/oracle/emp2.csv'
    ->  into  table  emp2  fields  terminated  by ',';
Query OK, 15 rows affected, 8 warnings (0.00 sec)
Records: 15  Deleted: 0  Skipped: 0  Warnings: 8

mysql> select * from emp2;
+-------+--------+-----------+------+------------+------+------+--------+
| EMPNO | ENAME  | JOB       | MGR  | HIREDATE   | SAL  | COMM | DEPTNO |
+-------+--------+-----------+------+------------+------+------+--------+
|  7839 | KING   | PRESIDENT |    0 | 1981-11-17 | 5000 |    0 |     10 |
|  7698 | BLAKE  | MANAGER   | 7839 | 1981-05-01 | 2850 |    0 |     30 |
|  7782 | CLARK  | MANAGER   | 7839 | 1981-05-09 | 2450 |    0 |     10 |
|  7566 | JONES  | MANAGER   | 7839 | 1981-04-01 | 2975 |    0 |     20 |
|  7654 | MARTIN | SALESMAN  | 7698 | 1981-09-10 | 1250 | 1400 |     30 |
|  7499 | ALLEN  | SALESMAN  | 7698 | 1981-02-11 | 1600 |  300 |     30 |
|  7844 | TURNER | SALESMAN  | 7698 | 1981-08-21 | 1500 |    0 |     30 |
|  7900 | JAMES  | CLERK     | 7698 | 1981-12-11 |  950 |    0 |     30 |
|  7521 | WARD   | SALESMAN  | 7698 | 1981-02-23 | 1250 |  500 |     30 |
|  7902 | FORD   | ANALYST   | 7566 | 1981-12-11 | 3000 |    0 |     20 |
|  7369 | SMITH  | CLERK     | 7902 | 1980-12-09 |  800 |    0 |     20 |
|  7788 | SCOTT  | ANALYST   | 7566 | 1982-12-22 | 3000 |    0 |     20 |
|  7876 | ADAMS  | CLERK     | 7788 | 1983-01-15 | 1100 |    0 |     20 |
|  7934 | MILLER | CLERK     | 7782 | 1982-01-11 | 1300 |    0 |     10 |
|     0 | NULL   | NULL      | NULL | NULL       | NULL | NULL |   NULL |
+-------+--------+-----------+------+------------+------+------+--------+
15 rows in set (0.00 sec)
```

■ sqoop

문제 150
★ 스쿱
- 오라클과 hive와의 데이터 연동
- 오라클의 emp테이블을 hive로 바로 로드할 수 있다.

* 스쿱 설치 과정

1. 스쿱설치 파일을 올린다.

```
sqoop-1.4.6.bin__hadoop-1.0.0.tar.gz
```

2. 스쿱설치 파일의 압축을 푼다.

```
# cd  /media/sf_Share
# chown  -R  oracle:oinstall sqoop-1.4.6.bin__hadoop- 1.0.0.tar.gz

# cp  sqoop-1.4.6.bin__hadoop-1.0.0.tar.gz  /home/oracle/

# su - oracle

$ tar xvzf  sqoop-1.4.6.bin__hadoop-1.0.0.tar.gz
```

3. ojdbc6.jar 파일을 sqoop 라이브러리로 이동한다.

```
# cd /media/sf_Share
# chown -R oracle:oinstall ojdbc6.zip
# cp ojdbc6.zip  /home/oracle/
# su - oracle
$ unzip  ojdbc6.zip
$ ls -l ojdbc6.jar
$ mv sqoop-1.4.6.bin__hadoop-1.0.0 sqoop
$ cp /home/oracle/ojdbc6.jar  /home/oracle/sqoop/lib/
```

4. 스쿱 디렉토리의 bin 디렉토리로 가서 스쿱을 실행한다.

```
$ cd /home/oracle
$ cd sqoop
$ cd bin
$ ./sqoop
```

4개의 경고가 뜨면 정상

5. .bash_profile 에 sqoop 홈 디렉토리를 지정한다.

```
$ cd /home/oracle
$ vi .bash_profile

export  SQOOP_HOME=/home/oracle/sqoop
export  PATH=$SQOOP_HOME/bin:$PATH

$ . .bash_profile
```

★ hive가 정상인지 확인한다.

```
[orcl:~]$ start-all.sh

[orcl:~]$ jps
6873 TaskTracker
6967 Jps
6344 NameNode
6617 SecondaryNameNode
6731 JobTracker
6475 DataNode

[orcl:~]$ hive

hive>
```

★ 오라클이 정상인지 확인한다.

```
[orcl:~]$ sqlplus scott/tiger

hive> drop table dept;
```

6. 오라클의 dept 테이블을 hive 로 로드한다.

1. 오라클에서 hive로 데이터 이행을 위한 스크립트를 준비한다.

```
$ vi table_import.sh

#!/bin/bash

oracle_table=`echo $3 | tr '[a-z]' '[A-Z]'`
hadoop_table=`echo $3 | tr '[A-Z]' '[a-z]'`

sqoop import --username $1 ₩
--password $2 ₩
--connect jdbc:oracle:thin:@localhost:1521:orcl ₩
--table $oracle_table ₩
--hive-import ₩
--hive-table $hadoop_table ₩
--hive-overwrite ₩
-m 1

[orcl:~]$ sh table_import.sh scott tiger DEPT
              :
              :
19/01/04 16:00:12 INFO hive.HiveImport: OK
19/01/04 16:00:12 INFO hive.HiveImport: Time taken: 0.498 seconds
19/01/04 16:00:12 INFO hive.HiveImport: Hive import complete.

hive> select * from dept;
OK
10.0    ACCOUNTING      NEW YORK
20.0    RESEARCH        DALLAS
30.0    SALES   CHICAGO
40.0    OPERATIONS      BOSTON
Time taken: 3.869 seconds, Fetched: 4 row(s)
```

SQL --> 파이썬 --> 리눅스 --> 하둡 --> 딥러닝(파이썬)--> R

프로그래밍 -- 시스템 엔지니어 -- 프로그래밍 -- 이론

■ 10. 맵리듀스를 java로 수행하기

(실습_02.Java 를 이용한 HDFS 입출력.txt 참고)

```
[orcl:~]$ start-all.sh

[orcl:~]$ jps
6039 TaskTracker
5647 DataNode
5782 SecondaryNameNode
6148 Jps
5881 JobTracker
5516 NameNode
```

1. hadoop 홈디렉토리에 자바 실행 파일인 jar 파일의 위치가 어디인지 설정하는 환경설정을 한다.

2. 하둡 홈디렉토리 밑에 labs 라는 디렉토리를 만들고 거기에 SingleFileWriteRead.java 파일을 생성한다.

↓

텍스트 파일 한개를 하둡 파일 시스템에 올리는 자바 코드

3. 두개의 텍스트 파일을 하나로 합쳐서 하둡 파일 시스템에 올리기 위해 PutMerge.java라는 파일을 생성한다.

4. 자바로 WordCount를 수행한다.

↓

겨울왕국 대본의 단어와 단어의 갯수를 출력한다.

SQL --> regexpress_count 함수로 수행
파이썬 --> len 함수로 수행
리눅스 --> wc 함수로 수행
하둡 --> 자바코드로 수행
R --> ?

■ 문제 모음

문제 2. 하둡 파일 시스템 명령어의 help를 조회하시오 !

```
[orcl:~]$ hadoop fs -help
hadoop fs is the command to execute fs commands. The full syntax is:
                    :
                    :
                    :
-help [cmd]:    Displays help for given command or all commands if none
                is specified.

[orcl:~]$ hadoop fs -help du
-du <path>:     Show the amount of space, in bytes, used by the files that
                match the specified file pattern.  Equivalent to the unix
                command "du -sb <path>/*" in case of a directory,
                and to "du -b <path>" in case of a file.
                The output is in the form
                        name(full path) size (in bytes)
```

문제 3. 겨울왕국 대본 winter.txt를 하둡 파일시스템에 올리시오 !

```
[orcl:~]$ hadoop fs -put /home/oracle/winter.txt winter.txt
[orcl:~]$ hadoop fs -ls winter.txt
Found 1 items
-rw-r--r--   3 oracle supergroup     222718 2018-12-28 11:43 /user/oracle/winter.txt
```

문제 4. 하둡 파일시스템에 emp2.csv파일을 올리시오 !

```
[orcl:~]$ hadoop fs -put /home/oracle/emp2.csv emp2.csv
[orcl:~]$ hadoop fs -ls emp2.csv
Found 1 items
-rw-r--r--   3 oracle supergroup        630 2018-12-28 11:50 /user/oracle/emp2.csv
```

문제 5. 하둡 파일 시스템에 올라간 emp2.csv를 cat으로 조회하시오 ! (점심시간 문제)

```
[orcl:~]$ hadoop fs -cat emp2.csv
7839,KING,PRESIDENT,0,1981-11-17,5000,0,10
7698,BLAKE,MANAGER,7839,1981-05-01,2850,0,30
7782,CLARK,MANAGER,7839,1981-05-09,2450,0,10
7566,JONES,MANAGER,7839,1981-04-01,2975,0,20
7654,MARTIN,SALESMAN,7698,1981-09-10,1250,1400,30
7499,ALLEN,SALESMAN,7698,1981-02-11,1600,300,30
7844,TURNER,SALESMAN,7698,1981-08-21,1500,0,30
7900,JAMES,CLERK,7698,1981-12-11,950,0,30
7521,WARD,SALESMAN,7698,1981-02-23,1250,500,30
7902,FORD,ANALYST,7566,1981-12-11,3000,0,20
7369,SMITH,CLERK,7902,1980-12-09,800,0,20
7788,SCOTT,ANALYST,7566,1982-12-22,3000,0,20
7876,ADAMS,CLERK,7788,1983-01-15,1100,0,20
7934,MILLER,CLERK,7782,1982-01-11,1300,0,10
```

문제 6. 직업이 salesman인 사원들의 모든 데이터를 출력하시오 !

```
[orcl:~]$ hadoop fs -cat emp2.csv | grep -i 'salesman'

[orcl:~]$ hadoop fs -cat emp2.csv | awk -F ',' '$3=="SALESMAN"'

7654,MARTIN,SALESMAN,7698,1981-09-10,1250,1400,30
7499,ALLEN,SALESMAN,7698,1981-02-11,1600,300,30
7844,TURNER,SALESMAN,7698,1981-08-21,1500,0,30
7521,WARD,SALESMAN,7698,1981-02-23,1250,500,30
```

문제 7. 월급이 3000인 사원의 이름과 월급을 출력하시오 !

```
[orcl:~]$ hadoop fs -cat emp2.csv | awk -F ',' '$6==3000 {print $2, $6}'
FORD 3000
SCOTT 3000
```

문제 8. dept2.csv를 하둡 파일 시스템에 올리시오 !

```
[orcl:~]$ hadoop fs -put /home/oracle/dept2.csv dept2.csv
[orcl:~]$ hadoop fs -ls dept2.csv
Found 1 items
-rw-r--r--   3 oracle supergroup         84 2018-12-28 14:31 /user/oracle/dept2.csv
[orcl:~]$ hadoop fs -cat dept2.csv
10,ACCOUNTING,NEW YORK
20,RESEARCH,DALLAS
30,SALES,CHICAGO
40,OPERATIONS,BOSTON
```

문제 9. hive로 접속해서 dept테이블을 생성하시오 !

```
============================================================
[orcl:~]$ hive

Logging initialized using configuration in jar:file:/home/oracle/hive-0.12.0/lib/hive-common-
0.12.0.jar!/hive-log4j.properties
hive> show tables;
OK
Time taken: 6.46 seconds
hive>  create table dept
    >  (deptno   int,
    >   dname    string,
    >   loc      string )
    >  ROW FORMAT DELIMITED
    >     FIELDS TERMINATED BY ','
    >     LINES TERMINATED BY 'Wn'
    >     STORED AS TEXTFILE;
OK
Time taken: 0.371 seconds
hive> show tables;
OK
```

```
dept
Time taken: 0.083 seconds, Fetched: 1 row(s)
============================================================
```

문제 10. emp2.csv는 emp테이블로 로드하고 dept2.csv 는 dept테이블로 로드하시오 !

```
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 ;

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

 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/oracle/dept2.csv'
    > overwrite  into  table   dept;
```

문제 11. hive에서 월급이 3000인 사원들의 이름과 월급을 출력하시오 !

```
hive> select ename, sal
    > from emp
    > where sal = 3000;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_201812281026_0001, Tracking URL = http://localhost:50030/jobdetails.jsp?
jobid=job_201812281026_0001
Kill Command = /u01/app/hadoop/hadoop-1.2.1/libexec/../bin/hadoop job  -kill job_201812281026_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2018-12-28 15:15:33,170 Stage-1 map = 0%,  reduce = 0%
2018-12-28 15:15:36,341 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.06 sec
2018-12-28 15:15:37,351 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.06 sec
2018-12-28 15:15:38,364 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.06 sec
MapReduce Total cumulative CPU time: 1 seconds 60 msec
Ended Job = job_201812281026_0001
MapReduce Jobs Launched:
Job 0: Map: 1   Cumulative CPU: 1.06 sec   HDFS Read: 838 HDFS Write: 21 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 60 msec
OK
FORD    3000
```

```
SCOTT   3000
Time taken: 10.457 seconds, Fetched: 2 row(s)
```

문제 12. 직업, 직업별 토탈월급을 출력하시오 !

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

Total MapReduce CPU Time Spent: 2 seconds 310 msec
OK
ANALYST 6000
CLERK   4150
MANAGER 8275
PRESIDENT       5000
SALESMAN        5600
Time taken: 21.692 seconds, Fetched: 5 row(s)
```

문제 13. 위의 결과를 다시 출력하는데 토탈월급이 높은 것부터 출력하시오 !

```
hive> select job, sum(sal) as sumsal
    > from emp
    > group by job
    > order by sumsal desc;

Total MapReduce CPU Time Spent: 4 seconds 90 msec
OK
MANAGER 8275
ANALYST 6000
SALESMAN        5600
PRESIDENT       5000
CLERK   4150
Time taken: 36.645 seconds, Fetched: 5 row(s)
```

※ hive에서 그룹함수를 order by 할때는 컬럼별칭을 사용해야 한다.

문제 14. 부서번호, 부서번호별 평균월급을 출력하는데 부서번호별 평균월급이 낮은것 부터 출력하시오 !.

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

Total MapReduce CPU Time Spent: 4 seconds 230 msec
OK
30      1566.6666666666667
20      2175.0
10      2916.6666666666665
Time taken: 37.706 seconds, Fetched: 3 row(s)
```

문제 15. emp와 dept를 조인해서 이름과 부서위치를 출력하시오 !

※ hive 에서는 1999 ansi 조인문법만 지원된다.

```
hive> select e.ename, d.loc
    >   from emp e join dept d
```

```
    >   on( e.deptno = d.deptno);

Total MapReduce CPU Time Spent: 790 msec
OK
KING    NEW YORK
BLAKE   CHICAGO
CLARK   NEW YORK
JONES   DALLAS
MARTIN  CHICAGO
ALLEN   CHICAGO
TURNER  CHICAGO
JAMES   CHICAGO
WARD    CHICAGO
FORD    DALLAS
SMITH   DALLAS
SCOTT   DALLAS
ADAMS   DALLAS
MILLER  NEW YORK
Time taken: 11.044 seconds, Fetched: 14 row(s)
```

문제 16. 직업, 직업별 토탈월급을 출력하는데 직업이 SALESMAN은 제외하고 출력하고 직업별 토탈월급이 4000이상인
 것만 출력하시오 !

※ hive에서는 having절이 지원 안된다.

※ hive에서 from절의 서브쿼리를 사용할 때는 별칭을 사용해야 한다.

```
hive> select *
    >   from (select job, sum(sal) as sumsal
    >           from emp
    >           where job != 'SALESMAN'
    >           group by job) a
    >   where a.sumsal >= 5000;

Total MapReduce CPU Time Spent: 2 seconds 940 msec
OK
ANALYST 6000
MANAGER 8275
PRESIDENT       5000
```

문제 17. 부서위치, 부서위치별 토탈월급을 출력하시오 !

```
hive> select d.loc,  sum(e.sal) as sumsal
    >   from emp e join dept d
    >   on ( e.deptno = d.deptno)
    >   group by d.loc;

Total MapReduce CPU Time Spent: 2 seconds 540 msec
OK
CHICAGO 9400
DALLAS  10875
NEW YORK        8750
Time taken: 24.485 seconds, Fetched: 3 row(s)
```

문제 18. full outer join이 가능한지 확인해보시오 !

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

Total MapReduce CPU Time Spent: 3 seconds 70 msec
OK
MILLER  NEW YORK
CLARK   NEW YORK
KING    NEW YORK
JONES   DALLAS
FORD    DALLAS
SMITH   DALLAS
SCOTT   DALLAS
ADAMS   DALLAS
MARTIN  CHICAGO
ALLEN   CHICAGO
TURNER  CHICAGO
JAMES   CHICAGO
WARD    CHICAGO
BLAKE   CHICAGO
NULL    BOSTON
Time taken: 22.249 seconds, Fetched: 15 row(s)
```

문제 19. 이름, 월급, 순위를 출력하는데 순위가 월급이 높은 사원부터 출력되게 하시오 !

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

Total MapReduce CPU Time Spent: 2 seconds 620 msec
OK
KING    5000    1
FORD    3000    2
SCOTT   3000    2
JONES   2975    4
BLAKE   2850    5
CLARK   2450    6
ALLEN   1600    7
TURNER  1500    8
MILLER  1300    9
WARD    1250    10
MARTIN  1250    10
ADAMS   1100    12
JAMES   950     13
SMITH   800     14
Time taken: 21.356 seconds, Fetched: 14 row(s)
```

문제 20. 부서번호, 이름, 월급, 순위를 출력하는데 순위가 부서번호별로 각각 월급이 높은 사원순으로 순위를 부여하시오 !

```
hive> select deptno, ename, sal, rank() over (partition by deptno order by sal desc)
    > from emp;
```

```
Total MapReduce CPU Time Spent: 2 seconds 430 msec
OK
10      KING    5000    1
10      CLARK   2450    2
10      MILLER  1300    3
20      FORD    3000    1
20      SCOTT   3000    1
20      JONES   2975    3
20      ADAMS   1100    4
20      SMITH   800     5
30      BLAKE   2850    1
30      ALLEN   1600    2
30      TURNER  1500    3
30      MARTIN  1250    4
30      WARD    1250    4
30      JAMES   950     6
Time taken: 20.024 seconds, Fetched: 14 row(s)
```

문제 21. 사원번호, 이름, 월급, 월급의 누적치가 출력되게 하시오 !

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

Total MapReduce CPU Time Spent: 2 seconds 230 msec
OK
7369    SMITH   800     800
7499    ALLEN   1600    2400
7521    WARD    1250    3650
7566    JONES   2975    6625
7654    MARTIN  1250    7875
7698    BLAKE   2850    10725
7782    CLARK   2450    13175
7788    SCOTT   3000    16175
7839    KING    5000    21175
7844    TURNER  1500    22675
7876    ADAMS   1100    23775
7900    JAMES   950     24725
7902    FORD    3000    27725
7934    MILLER  1300    29025
Time taken: 20.16 seconds, Fetched: 14 row(s)
```

문제 22. 부서번호, 사원번호, 이름, 월급, 월급의 누적치가 출력되게 하는데 부서번호별로 각각 월급의 누적치가 출력되게 하시오 !

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

Total MapReduce CPU Time Spent: 2 seconds 460 msec
OK
10      7782    CLARK   2450    2450
10      7839    KING    5000    7450
10      7934    MILLER  1300    8750
20      7369    SMITH   800     800
20      7566    JONES   2975    3775
20      7788    SCOTT   3000    6775
20      7876    ADAMS   1100    7875
20      7902    FORD    3000    10875
30      7499    ALLEN   1600    1600
```

```
30      7521    WARD    1250    2850
30      7654    MARTIN  1250    4100
30      7698    BLAKE   2850    6950
30      7844    TURNER  1500    8450
30      7900    JAMES   950     9400
Time taken: 20.175 seconds, Fetched: 14 row(s)
```

문제 23. 이름을 출력하는데 이름을 첫글자만 출력하고 그 첫글자를 소문자로 출력하시오 !

```
hive> select lower(substr(ename,1,1))
    > from emp;

Total MapReduce CPU Time Spent: 1 seconds 20 msec
OK
k
b
c
j
m
a
t
j
w
f
s
s
a
m
Time taken: 12.967 seconds, Fetched: 14 row(s)
```

문제 24. 이름, 입사한 년도 4자리를 출력하시오!

```
hive> select ename, from_unixtime(unix_timestamp(hiredate, 'yyyy-mm-dd'),'yyyy') as yearmm
    > from emp;

Total MapReduce CPU Time Spent: 1 seconds 280 msec
OK
KING    1981
BLAKE   1981
CLARK   1981
JONES   1981
MARTIN  1981
ALLEN   1981
TURNER  1981
JAMES   1981
WARD    1981
FORD    1981
SMITH   1980
SCOTT   1982
ADAMS   1983
MILLER  1982
Time taken: 7.458 seconds, Fetched: 14 row(s)
```

문제 25. 이름, 입사한 달을 출력하시오 !

```
hive> select ename, from_unixtime(unix_timestamp(hiredate, 'yyyy-mm-dd'),'mm') as mm
    > from emp;

Total MapReduce CPU Time Spent: 1 seconds 20 msec
OK
KING    11
BLAKE   05
CLARK   05
JONES   04
MARTIN  09
ALLEN   02
TURNER  08
JAMES   12
WARD    02
FORD    12
SMITH   12
SCOTT   12
ADAMS   01
MILLER  01
Time taken: 11.855 seconds, Fetched: 14 row(s)
```

문제 26. 부서번호, 부서번호별 토탈월급을 가로로 출력하시오 !

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

※ hive에서는 case문을 사용해야 한다.

```
hive> 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)
    > from emp;

Total MapReduce CPU Time Spent: 2 seconds 600 msec
OK
8750    10875   9400
Time taken: 20.233 seconds, Fetched: 1 row(s)
```

★ 컬럼명이 나오게 하고 싶다면 ??

```
hive> set hive.cli.print.header=true;
hive> 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)
    > from emp;

Total MapReduce CPU Time Spent: 2 seconds 570 msec
OK
dept10  dept20  _c2
8750    10875   9400
Time taken: 20.277 seconds, Fetched: 1 row(s)
```

문제 27. (오늘의 마지막 문제) 아래의 SQL을 Hive로 구현하시오 !

```
SQL> select job,
           sum ( decode(deptno,10,sal) ) "dept10",
           sum ( decode(deptno,20,sal) ) "dept20",
           sum ( decode(deptno,30,sal) ) "dept30"
     from emp
     group by job;

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

Total MapReduce CPU Time Spent: 2 seconds 590 msec
OK
job     dept10  dept20  dept30
ANALYST NULL    6000    NULL
CLERK   1300    1900    950
MANAGER 2450    2975    2850
PRESIDENT       5000    NULL    NULL
SALESMAN        NULL    NULL    5600
Time taken: 16.755 seconds, Fetched: 5 row(s)
```

문제 28. 사원 테이블에서 월급의 순위가 1등인 사원의 이름과 월급과 순위를 출력하시오 !

```
hive> select *
        from  (  select  ename, sal,
                    rank()  over (order by sal desc) rnk
                  from  emp
               )  aaa
        where rnk = 1;
```

※ hive 에서는 from 절의 서브쿼리에 alias 를 사용해야한다.

문제 29. 아래의 오라클 SQL 을 Hive 로 구현하시오 !

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

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

hive> select  *
        from   (  select  deptno, sum(sal)
                     from emp
                     group by deptno
                   union  all
                  select null  as deptno, sum(sal)
                     from emp )  aaa;
```

※ union all 과 같은 집합 연산자를 hive 에서 사용할때는
반드시 from 절의 서브쿼리에 사용해야한다.

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

문제 30. 아래의 오라클 SQL 을 hive 에서 구현하시오 !

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

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

문제 31. 아래의 오라클 SQL 을 Hive 로 구현하시오 !

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

```
hive>  select  deptno, sum(sal)
         from  emp
         group  by  deptno  gruoping  sets( deptno, () );
```

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

문제 32. 아래의 오라클 SQL 을 Hive 로 구현하시오 !

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

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

문제 33. 사원 테이블의 직업의 종류가 몇가지인지 출력하시오 !

```
hive> select  count( distinct  job)  from  emp;
```

문제 34. 오라클의 with 절의 hive 에서도 가능하지 확인하시오 !

```
SQL> with  q1  as  ( select  job, sum(sal)  sumsal
                     from  emp
                     group  by  job )
     select  job, sumsal
        from  q1;

hive>   with  q1   ( select  job, sum(sal)  sumsal
                     from  emp
                     group  by  job )
         select  job, sumsal
           from  q1;
```

※ hive 에서는 update, delete 는 지원 안함 .
   insert 는 append 는 가능하다.

문제 35. scott 의 월급을 9000 으로 변경하시오 !

```
hive>  update  emp
          set  sal = 9000
          where  ename='SCOTT';
```

문제 36. 사원 테이블을 전부 삭제 하시오 ! (delete)

```
  delete  from   emp;
```

문제 37. emp 테이블을 emp_backup 으로 생성하시오 !

```
hive> create  table  emp_backup
      as
         select  * from  emp;

hive> select * from  emp_backup;
```

문제 38. emp 테이블의 데이터를 emp_backup 에 insert 하시오

```
hive>  insert  into  table  emp_backup
       select *
        from  emp;

hive> select count(*)  from  emp_backup;

28
```

문제 39. emp_backup 의 내용을 emp 테이블의 내용으로 overwrite
하시오 !

```
hive> insert overwrite  table  emp_backup
      select *
       from  emp;

hive> select count(*) from emp_backup;

14
```

문제 40. 월급이 3000 인 사원들의 이름과 월급을 출력하시오 !

문제 41. (점심시간 문제) 월급이 1000 에서 3000 사이인 사원들의 이름과 월급을 출력하시오 !

문제 42. 이름의 첫글자가 S 로 시작하는 사원들의 이름과 월급을 출력하시오 !

```
tajo>  select ename
        from emp
       where ename like 'S%';
```

문제 43. 커미션이 null 인 사원들의 이름과 커미션을 출력하시오 !

```
 select ename, comm
  from  emp
  where  comm is null;
```

문제 44. 직업, 직업별 토탈월급을 출력하시오 !

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

문제 45. having 절이 지원되는지 확인하시오 !
직업, 직업별 토탈월급을 출력하는데 직업별 토탈월급이 4000 이상인것만 출력하시오 !

```
select  job, sum(sal)
  from emp
  group by job
  having sum(sal) >= 4000;
```

문제 46. select 문의 6가지절이 다 지원되는지 확인하시오 !
직업, 직업별 토탈월급을 출력하는데 직업이 SALESMAN 인 사원들은 제외하고 출력하고 직업별 토탈월급이 4000 이상인 것만 출력하고 직업별 토탈월급이 높은것부터 출력하시오 !

```
select  job, sum(sal)
  from  emp
  where  job !='SALESMAN'
  group  by  job
  having  sum(sal) >= 4000
  order by  sum(sal) desc;
```

문제 47.부서번호, 부서번호별 토탈월급을 출력하는데 가로로 출력하시오 !

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

tajo>  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;
```

문제 48. dept2.csv 로 dept 테이블을 생성하시오 !
(emp 테이블 생성했던 스크립트 가져와서 수정하세요)

```
CREATE EXTERNAL TABLE  dept (
 deptno INT ,
 dname text ,
 loc TEXT
) USING CSV WITH ('text.delimiter'=',') LOCATION 'file:///home/oracle/dept2.csv';
```

문제 49. 이름과 부서위치를 출력하시오 !

```
select  e.ename, d.loc
  from  emp  e, dept  d
  where  e.deptno = d.deptno ;
```

문제 50. 이름과 부서위치를 출력하는데 right outer join 을 써서 출력하시오 !

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

문제 51. 타조에서 full outer join 도 지원되는지 확인하시오 !

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

문제 52. 타조에서는 rollup 이 지원되지는 않지만 union all 은 지원되므로 아래의 SQL 을 union all 로 구현하시오 !

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

tajo> select  deptno, sum(sal)
        from  emp
        group  by  deptno
      union  all
      select  0  as deptno, sum(sal)
        from emp;
```

문제 53. 이름과 월급과 순위를 출력하는데 순위를 월급이 높은 사원 순으로 순위를 부여하시오 !

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

문제 54. 사원번호, 이름과 월급과 월급의 누적치를 출력하시오 !

```
select  empno, ename, sal,
              sum(sal) over (order by sal asc rows
                             between unbounded preceding
                             and  current  row)  sumsal
  from  emp;
```

※ 타조는 누적 데이터 출력을 지원 안함

문제 55. lead 와 lag 데이터 분석함수가 지원이 되는지 확인하시오!

```
select  ename, sal,
        lead(sal,1) over ( order by  sal  asc)  lead_sal,
         lag(sal,1) over ( order by  sal  asc)  lag_sal
   from  emp;
```

문제 56. 우리반 테이블을 타조에 올려서 테이블로 생성하시오!

```
CREATE EXTERNAL TABLE  emp8 (
ename         text,
age       int,
brith    text,
major    text,
email     text,
telecom    text,
address     text
) USING CSV WITH ('text.delimiter'=',') LOCATION 'file:///home/oracle/emp8.csv';


ename
age
brith
major
email
teleco
address
```

문제 57. 전공이 심리학과인 학생들이 이름과 나이와 전공을 출력하시오 !

```
select ename, age, major
     from emp8
    where  major like '%심리%';
```

문제 58. 서울에서 살지 않는 학생들의 이름과 나이와 주소를 출력하시오!

```
 select  ename, age, address
    from  emp8
   where address not like '%서울%'
```

문제 59. 리눅스 자동화 쉘에 아래의 내용을 추가하시오 !
"타조로 접속하려면 6번을" 을 추가하시오 !

```
$ sh total.sh

       1. 두파일의 차이를 확인하려면 1번을
       2. 특정 파일을 검색하려면 2번을
       3. 찾고자하는 단어의 갯수를 검색하려면 3번을
       4. 오라클 데이터를 csv 로 생성하려면 4번을
       5. emp.csv 에서 이름과 월급을 검색하려면 5번을
       6. 타조로 접속하려면 6번을
```

답 : $ cd

```
$ vi .bash_profile

export TAJO_HOME=/home/oracle/tajo/tajo
export PATH=$TAJO_HOME/bin:$PATH

$ . .bash_profile

$ tsql
```

문제 60. 리눅스 자동화 쉘에 아래의 내용을 추가하시오 !
"하이브로 접속하려면 7번을" 을 추가하시오 !

```
$ sh total.sh

      1. 두파일의 차이를 확인하려면 1번을
      2. 특정 파일을 검색하려면 2번을
      3. 찾고자하는 단어의 갯수를 검색하려면 3번을
      4. 오라클 데이터를 csv 로 생성하려면 4번을
      5. emp.csv 에서 이름과 월급을 검색하려면 5번을
      6. 타조로 접속하려면 6번을
      7. 하이브로 접속하려면 7번을
```

문제 61. 타조를 시작시키는 명령어와 타조를 중지시키는 명령어를 자동화 스크립트에 추가하시오 !
(오늘의 마지막 문제)

타조 시작시키는 명령어 : startup.sh
타조 중지시키는 명령어 : stop-tajo.sh

```
$ sh total.sh

      1. 두파일의 차이를 확인하려면 1번을
      2. 특정 파일을 검색하려면 2번을
      3. 찾고자하는 단어의 갯수를 검색하려면 3번을
      4. 오라클 데이터를 csv 로 생성하려면 4번을
      5. emp.csv 에서 이름과 월급을 검색하려면 5번을
      6. 타조로 접속하려면 6번을
      7. 하이브로 접속하려면 7번을
      8. 타조를 시작시키려면 8번을
      9. 타조를 중지시키려면 9번을
```

문제 62. pig에 emp테이블을 생성하시오 !

```
grunt> emp = LOAD '/home/oracle/emp2.csv'
              USING PigStorage(',')
             as (empno:int, ename:chararray, job:chararray,
                 mgr:int,hiredate:chararray,
                  sal:int, comm:int,deptno:int);

grunt> dump emp;

ths to process : 1
```

```
(7839,KING,PRESIDENT,0,1981-11-17,5000,0,10)
(7698,BLAKE,MANAGER,7839,1981-05-01,2850,0,30)
(7782,CLARK,MANAGER,7839,1981-05-09,2450,0,10)
(7566,JONES,MANAGER,7839,1981-04-01,2975,0,20)
(7654,MARTIN,SALESMAN,7698,1981-09-10,1250,1400,30)
(7499,ALLEN,SALESMAN,7698,1981-02-11,1600,300,30)
(7844,TURNER,SALESMAN,7698,1981-08-21,1500,0,30)
(7900,JAMES,CLERK,7698,1981-12-11,950,0,30)
(7521,WARD,SALESMAN,7698,1981-02-23,1250,500,30)
(7902,FORD,ANALYST,7566,1981-12-11,3000,0,20)
(7369,SMITH,CLERK,7902,1980-12-09,800,0,20)
(7788,SCOTT,ANALYST,7566,1982-12-22,3000,0,20)
(7876,ADAMS,CLERK,7788,1983-01-15,1100,0,20)
(7934,MILLER,CLERK,7782,1982-01-11,1300,0,10)
```

문제 63. 이름과 월급을 출력하시오 !

```
grunt> data = foreach emp generate ename, sal ;
grunt> dump data;

ths to process : 1
(KING,5000)
(BLAKE,2850)
(CLARK,2450)
(JONES,2975)
(MARTIN,1250)
(ALLEN,1600)
(TURNER,1500)
(JAMES,950)
(WARD,1250)
(FORD,3000)
(SMITH,800)
(SCOTT,3000)
(ADAMS,1100)
(MILLER,1300)
```

문제 64. 월급이 3000이상인 사원들의 이름과 월급을 출력하시오 !

```
grunt> data = foreach emp generate ename, sal;
grunt> data2 = filter data by sal >= 3000;
grunt> dump data2;

ths to process : 1
(KING,5000)
(FORD,3000)
(SCOTT,3000)
```

문제 65. 직업이 SALESMAN인 사원들의 이름과 월급과 직업을 출력하시오 !

```
grunt> data = foreach emp generate ename, sal , job;
grunt> data2 = filter data by job == 'SALESMAN';
grunt> dump data;

ths to process : 1
```

```
(MARTIN,1250,SALESMAN)
(ALLEN,1600,SALESMAN)
(TURNER,1500,SALESMAN)
(WARD,1250,SALESMAN)
```

문제 66. 직업이 SALESMAN인 사원들의 이름과 월급을 출력하시오 !

```
grunt> data = foreach emp generate ename, sal, job;
grunt> data2 = filter data by job =='SALESMAN';
grunt> data3 = foreach data2 generate ename, sal;
grunt> dump data3;

ths to process : 1
(MARTIN,1250)
(ALLEN,1600)
(TURNER,1500)
(WARD,1250)
```

문제 67. 이름과 월급을 출력하는데 월급이 높은 사원부터 출력하시오 !

```
grunt> data = foreach emp generate ename, sal;
grunt> data2 = order data by sal desc;
grunt> dump data2;

ths to process : 1
(KING,5000)
(FORD,3000)
(SCOTT,3000)
(JONES,2975)
(BLAKE,2850)
(CLARK,2450)
(ALLEN,1600)
(TURNER,1500)
(MILLER,1300)
(WARD,1250)
(MARTIN,1250)
(ADAMS,1100)
(JAMES,950)
(SMITH,800)
```

문제 68. 부서번호가 30번인 사원들의 이름과 월급과 부서번호를 출력하는데 월급이 높은 사원부터 출력하시오 !

```
grunt> data = foreach emp generate ename, sal, deptno;
grunt> data2 = order data by sal desc;
grunt> data3 = filter data by deptno == 30;
grunt> dump data3;

ths to process : 1
(BLAKE,2850,30)
(MARTIN,1250,30)
(ALLEN,1600,30)
(TURNER,1500,30)
(JAMES,950,30)
(WARD,1250,30)
```

문제 69. 직업과 직업별 인원수를 출력하시오 !

```
SQL> select job, count(*)
       from emp
       group by job;

grunt> data = foreach emp generate job;
grunt> data2 = group data by job;
grunt> data3 = foreach data2 generate group, COUNT(data);
grunt> dump data3;


ths to process : 1
(CLERK,4)
(ANALYST,2)
(MANAGER,3)
(SALESMAN,4)
(PRESIDENT,1)
```

문제 70. 부서번호, 부서번호별 토탈월급을 출력하시오 !

```
grunt> data = foreach emp generate deptno, sal;
grunt> data2 = group data by deptno;
grunt> data3 = foreach data2 generate group, SUM(data.sal);

ths to process : 1
(10,8750)
(20,10875)
(30,9400)
```

문제 71. 직업, 직업별 최대월급을 출력하시오 !

```
grunt> data = foreach emp generate job, sal;
grunt> data2 = group data by job;
grunt> data3 = foreach data2 generate group, MAX(data.sal);
grunt> dump data3;

ths to process : 1
(CLERK,1300)
(ANALYST,3000)
(MANAGER,2975)
(SALESMAN,1600)
(PRESIDENT,5000)
```

문제 72. 직업, 직업별 토탈월급을 출력하는데 직업이 SALESMAN인 사원은 제외하고 출력하시오 !

```
grunt> data = foreach emp generate job, sal;
grunt> data2 = group data by job;
grunt> data3 = foreach data2 generate group, SUM(data.sal);
grunt> data4 = filter data3 by group != 'SALESMAN';
grunt> dump data4;
```

```
ths to process : 1
(CLERK,4150)
(ANALYST,6000)
(MANAGER,8275)
(PRESIDENT,5000)
```

문제 73. 직업, 직업별 토탈월급을 출력하는데 직업이 SALESMAN 인 사원은 제외하고 출력하고 직업별 토탈월급이 4000 이상인 것만 출력하고 직업별 토탈월급이 높은것부터 출력하시오 !

```
grunt> data = foreach emp generate job, sal;
grunt> data1 = filter data by job != 'SALESMAN';
grunt> data2 = group data1 by job;
grunt> data3 = foreach data2 generate group, SUM(data1.sal) as sumsal;
grunt> data4 = filter data3 by sumsal >= 5000;
grunt> data5 = order data4 by sumsal desc;
grunt> dump data5;

ths to process : 1
(MANAGER,8275)
(ANALYST,6000)
(PRESIDENT,5000)
```

문제 74. 부서번호, 부서번호별 평균월급을 출력하는데 부서번호가 20번은 제외하고 출력하고 부서번호별 평균월급이 2000 이상인 것만 출력하고 부서번호별 평균월급이 높은것부터 출력하시오 ! (점심시간 문제)

```
grunt> data = foreach emp generate deptno, sal;
grunt> data1 = filter data by deptno != 20;
grunt> data2 = group data1 by deptno;
grunt> data3 = foreach data2 generate group, AVG(data1.sal) as avgsal;
grunt> data4 = filter data3 by avgsal >= 2000;
grunt> data5 = order data4 by avgsal desc;
grunt> dump data5;

ths to process : 1
(10,2916.6666666666665)
```

문제 75. dept2.csv르 ㄹ이용해서 dept테이블을 pig에서 생성하시오 !

```
grunt> dept = LOAD '/home/oracle/dept2.csv'
               USING PigStorage(',')
              as (deptno:int, dname:chararray, loc:chararray);

grunt> dump dept;

(10,ACCOUNTING,NEW YORK)
(20,RESEARCH,DALLAS)
(30,SALES,CHICAGO)
(40,OPERATIONS,BOSTON)
```

문제 76. 이름과 부서위치를 출력하시오 !

```
grunt> empdept = JOIN emp BY deptno, dept BY deptno;
```

```
grunt> data = foreach empdept generate ename, loc;
grunt> dump data;

ths to process : 1
(MILLER,NEW YORK)
(CLARK,NEW YORK)
(KING,NEW YORK)
(JONES,DALLAS)
(FORD,DALLAS)
(SMITH,DALLAS)
(SCOTT,DALLAS)
(ADAMS,DALLAS)
(MARTIN,CHICAGO)
(ALLEN,CHICAGO)
(TURNER,CHICAGO)
(JAMES,CHICAGO)
(WARD,CHICAGO)
(BLAKE,CHICAGO)
```

문제 77. DALLAS에서 근무하는 사원들의 이름과 부서워치를 출력하시오 !

```
grunt> empdept = JOIN emp BY deptno, dept BY deptno;
grunt> data = foreach empdept generate ename, loc;
grunt> data1 = filter data by loc == 'DALLAS';
grunt> dump data1;

ths to process : 1
(JONES,DALLAS)
(FORD,DALLAS)
(SMITH,DALLAS)
(SCOTT,DALLAS)
(ADAMS,DALLAS)
```

문제 78. 부서위치, 부서위치별 토탈월급을 출력하시오 !

```
grunt> empdept = JOIN emp BY deptno, dept BY deptno;
grunt> data = foreach empdept generate loc, sal;
grunt> data1 = group data by loc;
grunt> data2 = foreach data1 generate group, SUM(data.sal);
grunt> dump data2;

ths to process : 1
(DALLAS,10875)
(CHICAGO,9400)
(NEW YORK,8750)
```

문제 79. 부서위치, 부서위치별 인원수를 출력하시오 !

```
grunt> empdept = JOIN emp BY deptno, dept BY deptno;
grunt> data = foreach empdept generate loc;
grunt> data1 = group data by loc;
grunt> data2 = foreach data1 generate group, COUNT(data);
grunt> dump data2;
```

```
ths to process : 1
(DALLAS,5)
(CHICAGO,6)
(NEW YORK,3)
```

문제81. mongodb 에 emp 테이블을 생성하시오 !

```
# emp 테이블 drop

db.emp.drop()

# emp 테이블 생성

db.emp.save({empno:7499,ename:"SMITH",job:"CLERK",mgr:7902,hiredate:"1980-12-
17",sal:1800,comm:800,deptno:20})
db.emp.save({empno:7369,ename:"ALLEN",job:"SALESMAN",mgr:7698,hiredate:"1981-02-
20",sal:1600,comm:800,deptno:20})
db.emp.save({empno:7521,ename:"WARD",job:"SALESMAN",mgr:7698,hiredate:"1981-02-
22",sal:1250,comm:500,deptno:30})
db.emp.save({empno:7566,ename:"JONES",job:"MANAGER",mgr:7839, hiredate:"1981-04-
02",sal:2975,comm:0,deptno:20})
db.emp.save({empno:7654,ename:"MARTIN",job:"SALESMAN",mgr:7698,hiredate:"1981-09-
28",sal:1250,comm:1400,deptno:30})
db.emp.save({empno:7698,ename:"BLAKE",job:"MANAGER",mgr:7839,hiredate:"1981-05-
01",sal:2850,comm:0,deptno:30})
db.emp.save({empno:7782,ename:"CLARK",job:"MANAGER",mgr:7839,hiredate:"1981-06-
09",sal:2450,comm:0,deptno:10})
db.emp.save({empno:7788,ename:"SCOTT",job:"ANALYST",mgr:7566,hiredate:"1987-04-
19",sal:3000,comm:0,deptno:20})
db.emp.save({empno:7839,ename:"KING",job:"PRESIDENT",mgr:0,hiredate:"1981-11-
17",sal:5000,comm:0,deptno:10})
db.emp.save({empno:7844,ename:"TURNER",job:"SALESMAN",mgr:7698,hiredate:"1981-09-
08",sal:1500,comm:0,deptno:30})
db.emp.save({empno:7876,ename:"ADAMS",job:"CLERK",mgr:7788,hiredate:"1987-05-23",sal:1100,
comm:0,deptno:20})
db.emp.save({empno:7900,ename:"JAMES",job:"CLERK",mgr:7698,hiredate:"1981-12-03",sal:950,comm:0,deptno:30})
db.emp.save({empno:7902,ename:"FORD",job:"ANALYST",mgr:7566,hiredate:"1981-12-
03",sal:3500,comm:0,deptno:20})
db.emp.save({empno:7934,ename:"MILLER",job:"CLERK",mgr:7782,hiredate:"1982-01-23",sal:1300,
comm:0,deptno:20})
```

문제82. 아래의 select 문을 mongodb 로 구현하시오 !

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

> db.emp.aggregate([{$group:{_id:null, count:{$sum:1}}}])
```

문제 83. 부서번호가 10번인 사원들의 사원번호와 이름 월급을 조회하시오 !

```
SQL> select empno, ename, sal
       from emp
       where deptno = 10;
```

```
> db.emp.find( {deptno:{$all:[10]}}, {_id:0, empno:1, ename:1, sal:1} )
{ "empno" : 7782, "ename" : "CLARK", "sal" : 2450 }
{ "empno" : 7839, "ename" : "KING", "sal" : 5000 }
```

문제 84. 월급이 3000인 사원들의 이름과 월급과 직업을 출력하시오 !

```
> db.emp.find( {sal:{$all:[3000]}}, {_id:0, ename:1, sal:1, job:1} )
{ "ename" : "SCOTT", "job" : "ANALYST", "sal" : 3000 }
```

문제 85. 월급이 2000이상인 사원들의 이름과 월급과 직업을 출력하시오 !

```
SQL> select ename, sal,job
       from emp
       where sal >= 2000;

>db.emp.find( {sal:{$gt:2000}}, {_id:0, ename:1, sal:1, job:1} )

{ "ename" : "JONES", "job" : "MANAGER", "sal" : 2975 }
{ "ename" : "BLAKE", "job" : "MANAGER", "sal" : 2850 }
{ "ename" : "CLARK", "job" : "MANAGER", "sal" : 2450 }
{ "ename" : "SCOTT", "job" : "ANALYST", "sal" : 3000 }
{ "ename" : "KING", "job" : "PRESIDENT", "sal" : 5000 }
{ "ename" : "FORD", "job" : "ANALYST", "sal" : 3500 }
```

문제 86. 직업이 SALESMAN이 아닌 사원들의 이름과 직업을 출력하시오 !

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

>db.emp.find( {job:{$ne:'SALESMAN'}}, {_id:0, ename:1, job:1} )

{ "ename" : "SMITH", "job" : "CLERK" }
{ "ename" : "JONES", "job" : "MANAGER" }
{ "ename" : "BLAKE", "job" : "MANAGER" }
{ "ename" : "CLARK", "job" : "MANAGER" }
{ "ename" : "SCOTT", "job" : "ANALYST" }
{ "ename" : "KING", "job" : "PRESIDENT" }
{ "ename" : "ADAMS", "job" : "CLERK" }
{ "ename" : "JAMES", "job" : "CLERK" }
{ "ename" : "FORD", "job" : "ANALYST" }
{ "ename" : "MILLER", "job" : "CLERK" }
```

문제 87. 이름과 월급을 출력하는데 월급이 높은 사원부터 출력하시오 !

```
SQL> select ename, sal
       from emp
       order by sal desc;

>db.emp.find( {}, {_id:0, ename:1, sal:1} ).sort( {sal:-1} )
```

```
{ "ename" : "KING", "sal" : 5000 }
{ "ename" : "FORD", "sal" : 3500 }
{ "ename" : "SCOTT", "sal" : 3000 }
{ "ename" : "JONES", "sal" : 2975 }
{ "ename" : "BLAKE", "sal" : 2850 }
{ "ename" : "CLARK", "sal" : 2450 }
{ "ename" : "SMITH", "sal" : 1800 }
{ "ename" : "ALLEN", "sal" : 1600 }
{ "ename" : "TURNER", "sal" : 1500 }
{ "ename" : "MILLER", "sal" : 1300 }
{ "ename" : "WARD", "sal" : 1250 }
{ "ename" : "MARTIN", "sal" : 1250 }
{ "ename" : "ADAMS", "sal" : 1100 }
{ "ename" : "JAMES", "sal" : 950 }
```

문제 88. (오늘의 마지막 문제) 직업이 SALESMAN인 사원들의 이름과 월급과 직업을 출력하는데 월급이 낮은 사원부터
출력하시오 !

```
>db.emp.find( {job:{$eq:'SALESMAN'}}, {_id:0, ename:1, sal:1, job:1} ).sort( {sal:1} )

{ "ename" : "WARD", "job" : "SALESMAN", "sal" : 1250 }
{ "ename" : "MARTIN", "job" : "SALESMAN", "sal" : 1250 }
{ "ename" : "TURNER", "job" : "SALESMAN", "sal" : 1500 }
{ "ename" : "ALLEN", "job" : "SALESMAN", "sal" : 1600 }
```

문제 89. 월급이 1250 인 사원들의 이름과 월급을 출력하시오 !

```
SQL> select ename, sal
       from emp
       where sal = 1250;

> db.emp.find( {sal:{$eq:1250}}, {_id:0, ename:1, sal:1})

{ "ename" : "WARD", "sal" : 1250 }
{ "ename" : "MARTIN", "sal" : 1250 }
```

문제 90. 월급이 1000에서 3000 사이인 사원들의 이름과 월급을 출력하시오 !

```
SQL> select ename, sal
       from emp
       where sal >= 1000 and sal <= 3000;

> db.emp.find( {$and: [ {sal:{$gte:1000, $lte:3000}} ] }, {_id:0, ename:1, sal:1})

> db.emp.find( {$and: [ {sal:1000}, {sal:3000} ] }, {_id:0, ename:1, sal:1})
```

문제 91. 사원번호가 7788 또는 7902인 사원들의 사원번호와 이름과 월급을 출력하시오 !

```
> db.emp.find( {$or: [ {empno:{$eq:7788}}, {empno:{$eq:7902}} ] }, {_id:0, empno:1, ename:1, sal:1})
{ "empno" : 7788, "ename" : "SCOTT", "sal" : 3000 }
```

```
{ "empno" : 7902, "ename" : "FORD", "sal" : 3500 }
```

문제 92. 부서번호를 출력하는데 중복제거해서 출력하시오 !

```
SQL> select distinct deptno
        from emp;

> db.emp.distinct('deptno')
[ 20, 30, 10 ]
```

문제 93. 사원 테이블의 전체 건수를 출력하시오 !

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

> db.emp.count()
14
```

문제 94. 직업이 SALESMAN인 사원들의 인원수를 출력하시오 !

```
SQL> select count(*)
        from emp
        where job = 'SALESMAN';

> db.emp.find( {job:{$eq:"SALESMAN"}} ).count()

> db.emp.count( {job:'SALESMAN'} )
```

문제 95. 월급이 3000이상인 사원들의 인원수를 출력하시오 !

```
SQL> select count(*)
       from emp
       where sal >= 3000;


> db.emp.count( {sal:{$gte:3000}})
3
```

문제 96. 사원테이블 전체의 최대월급을 출력하시오 !

```
SQL> select max(sal)
       from emp;

> db.emp.find( {} , { _id:0, sal:1 } ).sort( {sal:-1} ).limit(1)
{ "sal" : 5000 }
```

문제 97. 직업이 SALESMAN인 사원들의 최대 월급을 출력하시오 !

```
SQL> select max(sal)
       from emp
```

```
        where job = 'SALESMAN';

> db.emp.find( {job:'SALESMAN'} , { _id:0, sal:1 } ).sort( {sal:-1} ).limit(1)
{ "sal" : 1600 }
```

문제 98. 사원테이블의 토탈월급을 출력하시오 !

```
SQL> select sum(sal)
        from emp;

> db.emp.aggregate( [ {$group: {_id:null, total:{$sum:'$sal'}}}])
```

문제 99. 최대월급을 출력하시오 !

```
SQL> select max(sal)
        from emp;

> db.emp.aggregate( [ {$group: {_id:0, max:{$max:'$sal'} } } ] )
{ "_id" : 0, "max" : 5000 }
```

문제 100. 최소월급을 출력하시오 !

```
SQL> select min(sal)
        from emp;

> db.emp.aggregate( [ {$group: {_id:0, min:{$min:'$sal'} } } ] )
{ "_id" : 0, "min" : 950 }
```

문제 101. 직업이 SALESMAN인 사원들의 최대 월급을 출력하시오 !

```
SQL> select max(sal)
        from emp
        where job = 'SALESMAN';

> db.emp.aggregate( [{$match: {job:'SALESMAN'}}, {$group: {_id:0, max:{$max:'$sal'} } } ] )
{ "_id" : 0, "max" : 1600 }
```

문제 102. 부서번호가 30번인사원들의 최대 월급을 출력하시오 1

```
SQL> select max(sal)
        from emp
        where deptno = 30;

> db.emp.aggregate( [{$match: {deptno:30}}, {$group: {_id:0, max:{$max:'$sal'} } } ] )
{ "_id" : 0, "max" : 2850 }
```

문제 103. 부서번호, 부서번호별 토탈월급을 출력하시오 !

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

```
        from emp
        group by deptno;

> db.emp.aggregate( [ {$group: {_id:'$deptno' , total:{$sum:'$sal'} } } ] )
{ "_id" : 10, "total" : 7450 }
{ "_id" : 30, "total" : 7800 }
{ "_id" : 20, "total" : 15275 }
```

문제 104. 직업, 직업별 최대월급을 출력하시오 !

```
SQL> select job, max(sal)
        from emp
        group by job;

> db.emp.aggregate( [ {$group: {_id:'$job', max:{$max:'$sal'} } } ] )
{ "_id" : "PRESIDENT", "max" : 5000 }
{ "_id" : "ANALYST", "max" : 3500 }
{ "_id" : "MANAGER", "max" : 2975 }
{ "_id" : "SALESMAN", "max" : 1600 }
{ "_id" : "CLERK", "max" : 1800 }
```

문제 105. 직업, 직업별 토탈월급을 출력하는데 직업이 SALESMAN인 사원을 제외하고 출력하시오 !

```
SQL> select job, sum(sal)
        from emp
        where job != 'SALESMAN';

> db.emp.aggregate({$match:{job:{ $ne:'SALESMAN'}}}, {$group:{_id:'$job', total:{$sum:'$sal'}}})
{ "_id" : "PRESIDENT", "total" : 5000 }
{ "_id" : "ANALYST", "total" : 6500 }
{ "_id" : "MANAGER", "total" : 8275 }
{ "_id" : "CLERK", "total" : 5150 }
```

문제 106. 직업, 직업별 토탈월급을 출력하는데 직업이 SALESMAN은 제외하고 출력하고 직업별 토탈월급이 높은것부터
출력하시오 !

```
SQL> select job, sum(sal) sumsal
        from emp
        where job != 'SALESMAN'
        order by sumsal desc;

> db.emp.aggregate({$match:{job:{$ne:'SALESMAN'}}},
                 {$group:{_id:'$job', total:{$sum:'$sal'}}},
                 {$sort:{'total':-1}})
{ "_id" : "MANAGER", "total" : 8275 }
{ "_id" : "ANALYST", "total" : 6500 }
{ "_id" : "CLERK", "total" : 5150 }
{ "_id" : "PRESIDENT", "total" : 5000 }
```

문제 107. 직업, 직업별 토탈월급을 출력하는데 직업이 SALESMAN인 사원은 제외하고 출력하고 직업별 토탈월급이
6000 이상인 것만 출력하고 직업별 토탈월급이 높은것 부터 출력하시오 !

```
> db.emp.aggregate({$match:{job:{$ne:'SALESMAN'}}},
```

```
                    {$group:{_id:'$job', total:{$sum:'$sal'}}},
                    {$match:{total:{$gte:6000}}},
                    {$sort:{'total':-1}})

{ "_id" : "MANAGER", "total" : 8275 }
{ "_id" : "ANALYST", "total" : 6500 }
```

문제 108. 이름의 첫번째 철자가 A로 시작하는 사람들의 이름과 월급을 출력하시오

```
SQL> select ename, sal
       from emp
       where ename like 'A%';

> db.emp.find({ename:/^A/}, {_id:0, ename:1, sal:1})
{ "ename" : "ALLEN", "sal" : 1600 }
{ "ename" : "ADAMS", "sal" : 1100 }
```

문제 109. 이름의 끝 글자가 T로 끝나는 사원들의 이름과 월급을 출력하시오 !

```
SQL> select ename, sal
        from emp
        where ename like '%T';

> db.emp.find({ename:/T$/}, {_id:0, ename:1, sal:1})
{ "ename" : "SCOTT", "sal" : 3000 }
```

< 점심시간 문제 두 문제 검사>

문제 110. 이름에 두번째 철자가 M인 사원들의 이름과 월급을 출력하시오 !

```
> db.emp.find({ename:/^.M/}, {_id:0, ename:1, sal:1})
{ "ename" : "SMITH", "sal" : 1800 }
```

문제 111. 이름에 A를 포함하고 있는 사원들의 이름과 월급을 출력하시오 !

```
> db.emp.find({ename:/.*A.*/}, {_id:0, ename:1, sal:1})
{ "ename" : "ALLEN", "sal" : 1600 }
{ "ename" : "WARD", "sal" : 1250 }
{ "ename" : "MARTIN", "sal" : 1250 }
{ "ename" : "BLAKE", "sal" : 2850 }
{ "ename" : "CLARK", "sal" : 2450 }
{ "ename" : "ADAMS", "sal" : 1100 }
{ "ename" : "JAMES", "sal" : 950 }
```

문제 113. 부서위치가 DALLAS의 부서번호를 출력하시오 !

```
> db.dept.find({loc:'DALLAS'}, {_id:0, deptno:1})
20.0
```

문제 114. 이름과 부서위치를 출력하시오 !

```
db.emp.aggregate([
{
    $lookup:
    {
        from:"dept",
        localField:"deptno",
        foreignField:"deptno",
        as:"aa"
        }
    },
    { $project:
        {
            _id:0,
            ename:1,
            aa:{loc:1}
        }
    }
])
```

문제 115. 이름과 월급과 부서명을 출력하시오 !

```
db.emp.aggregate([
{
    $lookup:
    {
        from:"dept",
        localField:"deptno",
        foreignField:"deptno",
        as:"aa"
        }
    },
    { $project:
        {
            _id:0,
            ename:1,
            sal:1,
            aa:{dname:1}
        }
    }
    ])
```

문제 116. SCOTT의 월급을 5600으로 변경하시오 !

```
SQL> update emp set sal =5600 where ename = 'SCOTT';

db.emp.update(
                {ename:'SCOTT'},
                {$set:{sal:5600} },
                {multi:true}
              )
SCOTT 5600.0
```

문제 117. 월급이 3000이상인 사원들의 comm을 7000으로 수정하시오 !

```
db.emp.update( {sal:{$gte:3000}},
               {$set:{comm:7000}},
               {multi:true}
             )
Updated 3 existing record(s) in 3ms
```

문제 118. 직업이 ANALYST인 사원들을 삭제하시오 !

```
SQL> delete from emp where job = 'ANALYST';

db.emp.remove( {job:'ANALYST'}
             )
Removed 2 record(s) in 28ms
```

문제 119. emp 테이블을 전체를 지우시오 !

```
db.emp.remove( {empno:{$ne:0}}
              )
Removed 12 record(s) in 19ms
```

문제 120. 직업이 SALESMAN인 사원들의 이름과 월급과 직업을 출력하는데 월급이 높은 사원부터 출력하시오 !

```
select ename, sal, job
  from emp
  where job = 'salesman'
  order by sal desc;
```

※ 오라클과는 다르게 salesman을 소문자로 써도 된다.

문제 121. 이름과 커미션을 출력하는데 커미션이 null인 사원들은 0으로 출력하시오 !

※ 오라클 vs MySQL
nvl ifnull

```
mysql> select ename, ifnull(comm, 0)
    ->   from emp;
+--------+-----------------+
| ename  | ifnull(comm, 0) |
+--------+-----------------+
| KING   |               0 |
| BLAKE  |               0 |
| CLARK  |               0 |
| JONES  |               0 |
| MARTIN |            1400 |
```

```
| ALLEN  |             300 |
| TURNER |               0 |
| JAMES  |               0 |
| WARD   |             500 |
| FORD   |               0 |
| SMITH  |               0 |
| SCOTT  |               0 |
| ADAMS  |               0 |
| MILLER |               0 |
+--------+-----------------+
14 rows in set (0.00 sec)
```

문제122. 오늘 날짜를 출력하시오.

```
오라클 vs mySQL
sysdate   sysdate()

mysql> select sysdate() from dual;

+---------------------+
| sysdate()           |
+---------------------+
| 2019-01-03 17:28:11 |
+---------------------+
1 row in set (0.00 sec)
```

문제123. 이름, 입사한 날짜부터 오늘까지 총 며칠 근무했는지 출력하시오.

```
SQL> select ename, round(sysdate - hiredate)
       from emp;

mySQL> select ename, to_days(sysdate())-to_days(hiredate)
        from emp;

+--------+---------------------------------------+
| ename  | to_days(sysdate())-to_days(hiredate) |
+--------+---------------------------------------+
| KING   |                                 13561 |
| BLAKE  |                                 13761 |
| CLARK  |                                 13753 |
| JONES  |                                 13791 |
| MARTIN |                                 13629 |
| ALLEN  |                                 13840 |
| TURNER |                                 13649 |
| JAMES  |                                 13537 |
| WARD   |                                 13828 |
| FORD   |                                 13537 |
| SMITH  |                                 13904 |
| SCOTT  |                                 13161 |
| ADAMS  |                                 13137 |
| MILLER |                                 13506 |
+--------+---------------------------------------+
```

문제124. 이름, 입사한 날짜부터 오늘까지 총 몇 달 근무했는지 출력하시오.

```
SQL> select ename, months_between(sysdate,hiredate)
       from emp;

mySQL> select ename, period_diff(date_format(now(),'%Y%m') ,date_format(hiredate,'%Y%m')) month
         from emp;

+--------+-------+
| ename  | month |
+--------+-------+
| KING   |   446 |
| BLAKE  |   452 |
| CLARK  |   452 |
| JONES  |   453 |
| MARTIN |   448 |
| ALLEN  |   455 |
| TURNER |   449 |
| JAMES  |   445 |
| WARD   |   455 |
| FORD   |   445 |
| SMITH  |   457 |
| SCOTT  |   433 |
| ADAMS  |   432 |
| MILLER |   444 |
+--------+-------+
14 rows in set (0.01 sec)
```

문제 125. 오늘부터 100달 뒤에 돌아오는 날짜가 어떻게 되는가?

```
SQL> select add_months(sysdate,100)
       from dual;

mysql> select period_add(date_format(sysdate(), '%y%m'),100) from dual;
+-----------------------------------------------+
| period_add(date_format(sysdate(), '%y%m'),100) |
+-----------------------------------------------+
|                                        202705 |
+-----------------------------------------------+
1 row in set (0.00 sec)


mysql> select substring(aa,1,4) as year,
    ->        substring(aa,5,2) as month
    -> from ( select period_add(date_format(sysdate(), '%y%m'),100) aa
    ->          from dual ) aaa;
+------+-------+
| year | month |
+------+-------+
| 2027 | 05    |
+------+-------+
1 row in set (0.00 sec)
```

문제 126. 이번달의 마지막 날짜가 어떻게 되는가?

```
SQL> select last_day(sysdate)
       from dual;

mysql> select last_day( sysdate() )
    ->   from dual;
+----------------------+
| last_day( sysdate() ) |
+----------------------+
| 2019-01-31           |
+----------------------+
1 row in set (0.01 sec)
```

문제 127. (오늘의 마지막 문제) 오늘부터 이번달 말일까지 총 몇일 남았는지 출력하시오 ! (to_date 함수 활용)

```
mysql> select to_days(last_day(sysdate()))-to_days(sysdate()) from dual;
+-----------------------------------------------+
| to_days(last_day(sysdate()))-to_days(sysdate()) |
+-----------------------------------------------+
|                                            28 |
+-----------------------------------------------+
1 row in set (0.00 sec)
```

문제 128. 오늘의 무슨요일인지 출력하시오 !

```
SQL> select to_char ( sysdate, 'day' )
       from dual;


mysql> select date_format( sysdate(), '%W' )
    ->   from dual;
+--------------------------------+
| date_format( sysdate(), '%W' ) |
+--------------------------------+
| Friday                         |
+--------------------------------+
1 row in set (0.00 sec)
```

문제 129. 이름, 입사일, 입사한 요일을 출력하시오 !

```
SQL> select ename, hiredate, to_char( hiredate, 'day')
       from emp;

mysql> select ename, hiredate, date_format(hiredate,'%w')
    -> from emp;
+--------+------------+-----------------------------+
| ename  | hiredate   | date_format( hiredate, '%W') |
+--------+------------+-----------------------------+
| KING   | 1981-11-17 | Tuesday                     |
| BLAKE  | 1981-05-01 | Friday                      |
| CLARK  | 1981-05-09 | Saturday                    |
| JONES  | 1981-04-01 | Wednesday                   |
```

```
| MARTIN | 1981-09-10 | Thursday                    |
| ALLEN  | 1981-02-11 | Wednesday                   |
| TURNER | 1981-08-21 | Friday                      |
| JAMES  | 1981-12-11 | Friday                      |
| WARD   | 1981-02-23 | Monday                      |
| FORD   | 1981-12-11 | Friday                      |
| SMITH  | 1980-12-09 | Tuesday                     |
| SCOTT  | 1982-12-22 | Wednesday                   |
| ADAMS  | 1983-01-15 | Saturday                    |
| MILLER | 1982-01-11 | Monday                      |
+--------+------------+-----------------------------+
14 rows in set (0.00 sec)
```

문제 130. 이름과 월급을 출력하는데 월급을 출력할때에 천단위를 붙이시오 !

```
SQL> select ename, to_char(sal, '999,999')
       from emp;

mysql> select ename, format(sal, 0)
    ->   from emp;
+--------+---------------+
| ename  | format(sal, 0) |
+--------+---------------+
| KING   | 5,000          |
| BLAKE  | 2,850          |
| CLARK  | 2,450          |
| JONES  | 2,975          |
| MARTIN | 1,250          |
| ALLEN  | 1,600          |
| TURNER | 1,500          |
| JAMES  | 950            |
| WARD   | 1,250          |
| FORD   | 3,000          |
| SMITH  | 800            |
| SCOTT  | 3,000          |
| ADAMS  | 1,100          |
| MILLER | 1,300          |
+--------+---------------+
14 rows in set (0.00 sec)
```

문제 131. 이름, 직업, 보너스를 출력하는데 직업이 SALESMAN 이면 보너스를 6000을 출력하고 아니면 0을 출력하시오 !

```
SQL> select ename, job, decode(job, 'SALESMAN', 6000, 0) as bonus
       from emp;

mysql> select ename, job, if(job='salesman', 6000, 0) as bonus
    ->   from emp;
+--------+-----------+-------+
| ename  | job       | bonus |
+--------+-----------+-------+
| KING   | PRESIDENT |     0 |
| BLAKE  | MANAGER   |     0 |
| CLARK  | MANAGER   |     0 |
| JONES  | MANAGER   |     0 |
| MARTIN | SALESMAN  |  6000 |
```

```
| ALLEN  | SALESMAN  |  6000 |
| TURNER | SALESMAN  |  6000 |
| JAMES  | CLERK     |     0 |
| WARD   | SALESMAN  |  6000 |
| FORD   | ANALYST   |     0 |
| SMITH  | CLERK     |     0 |
| SCOTT  | ANALYST   |     0 |
| ADAMS  | CLERK     |     0 |
| MILLER | CLERK     |     0 |
+--------+-----------+-------+
14 rows in set (0.00 sec)
```

문제 132. 이름, 직업, 보너스를 출력하는데 직업이 SALESMAN이면 보너스를 6000을 출력하고 직업이 ANALYST이면 보너스를 3000을 출력하고 나머지는 0을 출력하시오 !

```
SQL> select ename, job, decode( job, 'SALESMAN', 6000, job, 'ANALYST', 3000, 0 ) as bonus
       from emp;

mysql> select ename, job, if ( job='salesman', 6000, if(job='analyst', 3000, 0)) as bonus
    ->   from emp;
+--------+-----------+-------+
| ename  | job       | bonus |
+--------+-----------+-------+
| KING   | PRESIDENT |     0 |
| BLAKE  | MANAGER   |     0 |
| CLARK  | MANAGER   |     0 |
| JONES  | MANAGER   |     0 |
| MARTIN | SALESMAN  |  6000 |
| ALLEN  | SALESMAN  |  6000 |
| TURNER | SALESMAN  |  6000 |
| JAMES  | CLERK     |     0 |
| WARD   | SALESMAN  |  6000 |
| FORD   | ANALYST   |  3000 |
| SMITH  | CLERK     |     0 |
| SCOTT  | ANALYST   |  3000 |
| ADAMS  | CLERK     |     0 |
| MILLER | CLERK     |     0 |
+--------+-----------+-------+
14 rows in set (0.00 sec)
```

문제 133. 이름, 부서번호, 보너스를 출력하는데 부서번호가 10번이면 보너스를 7000으로, 부서번호가 20번이면 9000으로, 부서번호가 30번이면 보너스를 4000으로 출력하시오 !

```
SQL> select ename, deptno,
            case deptno when 10 then 6000
                        when 20 then 9000
            else 4000
            end as bones
       from emp;

mysql> select ename, deptno,
    ->             case deptno when 10 then 6000
    ->                         when 20 then 9000
    ->             else 4000
    ->             end as bones
    ->        from emp;
```

```
+--------+--------+-------+
| ename  | deptno | bones |
+--------+--------+-------+
| KING   |     10 |  6000 |
| BLAKE  |     30 |  4000 |
| CLARK  |     10 |  6000 |
| JONES  |     20 |  9000 |
| MARTIN |     30 |  4000 |
| ALLEN  |     30 |  4000 |
| TURNER |     30 |  4000 |
| JAMES  |     30 |  4000 |
| WARD   |     30 |  4000 |
| FORD   |     20 |  9000 |
| SMITH  |     20 |  9000 |
| SCOTT  |     20 |  9000 |
| ADAMS  |     20 |  9000 |
| MILLER |     10 |  6000 |
+--------+--------+-------+
14 rows in set (0.17 sec)
```

문제 134. 이름, 월급, 보너스를 출력하는데 월급이 2000 이상이면 보너스를 900을 출력하고 1000이상이면 보너스를 800을 출력하고 나머지는 0을 출력하시오

```
SQL>
select ename, sal,
       case when sal >= 2000 then 900
            when sal >= 1000 then 800
       else 0 end as bonus
  from emp;

mysql> select ename, sal,
    ->        case when sal >= 2000 then 900
    ->             when sal >= 1000 then 800
    ->        else 0 end as bonus
    ->   from emp;
+--------+------+-------+
| ename  | sal  | bonus |
+--------+------+-------+
| KING   | 5000 |   900 |
| BLAKE  | 2850 |   900 |
| CLARK  | 2450 |   900 |
| JONES  | 2975 |   900 |
| MARTIN | 1250 |   800 |
| ALLEN  | 1600 |   800 |
| TURNER | 1500 |   800 |
| JAMES  |  950 |     0 |
| WARD   | 1250 |   800 |
| FORD   | 3000 |   900 |
| SMITH  |  800 |     0 |
| SCOTT  | 3000 |   900 |
| ADAMS  | 1100 |   800 |
| MILLER | 1300 |   800 |
+--------+------+-------+
14 rows in set (0.00 sec)
```

문제 135. 아래와 같이 결과를 출력하시오 !

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

mysql> select sum( if( deptno = 10 , sal, null ) ) as "10",
    ->        sum( if( deptno = 20 , sal, null ) ) as "20",
    ->        sum( if( deptno = 30 , sal, null ) ) as "30"
    ->   from emp;
+------+-------+------+
| 10   | 20    | 30   |
+------+-------+------+
| 8750 | 10875 | 9400 |
+------+-------+------+
1 row in set (0.00 sec)
```

문제 136. 아래의 SQL을 MySQL로 구현하시오 !

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

mysql> select deptno, sum(sal)
    ->   from emp
    ->   group by deptno with rollup;
+--------+----------+
| deptno | sum(sal) |
+--------+----------+
|     10 |     8750 |
|     20 |    10875 |
|     30 |     9400 |
|   NULL |    29025 |
+--------+----------+
4 rows in set (0.02 sec)
```

※ MySQL은 rollup은 with rollup으로 지원하는데 cube는 지원하지 않는다.

문제 137. 아래의 SQL을 MySQL로 구현하시오 !

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

mysql> select null as deptno, sum(sal)
    ->   from emp
    -> union all
    -> select deptno, sum(sal)
    ->   from emp
    ->   group by deptno;
+--------+----------+
| deptno | sum(sal) |
+--------+----------+
|   NULL |    29025 |
|     10 |     8750 |
|     20 |    10875 |
```

```
|     30 |     9400 |
+--------+----------+
4 rows in set (0.04 sec)
```

문제 138. 이름, 입사일, 순위를 출력하는데 순위가 먼저 입사한 사원순으로 순위를 부여하시오 !

```
SQL> select ename, hiredate, rank() over (order by hiredate asc) 순위
       from emp;

mysql> select ename, hiredate, (select count(*) + 1 from emp where hiredate < e.hiredate ) 순위
    ->   from emp e
    ->   order by hiredate asc;
+--------+------------+--------+
| ename  | hiredate   | 순위   |
+--------+------------+--------+
| SMITH  | 1980-12-09 |      1 |
| ALLEN  | 1981-02-11 |      2 |
| WARD   | 1981-02-23 |      3 |
| JONES  | 1981-04-01 |      4 |
| BLAKE  | 1981-05-01 |      5 |
| CLARK  | 1981-05-09 |      6 |
| TURNER | 1981-08-21 |      7 |
| MARTIN | 1981-09-10 |      8 |
| KING   | 1981-11-17 |      9 |
| FORD   | 1981-12-11 |     10 |
| JAMES  | 1981-12-11 |     10 |
| MILLER | 1982-01-11 |     12 |
| SCOTT  | 1982-12-22 |     13 |
| ADAMS  | 1983-01-15 |     14 |
+--------+------------+--------+
14 rows in set (0.00 sec)
```

문제 139. 이름, 월급, 순위를 출력하시오 ! (순위는 월급 높은 순)

```
SQL> select ename, sal, rank() over (order by sal desc)
       from emp;

mysql> select ename, sal, (select count(*) + 1 from emp where sal > e.sal) 순위
    ->   from emp e
    ->   order by sal desc;
+--------+------+--------+
| ename  | sal  | 순위   |
+--------+------+--------+
| KING   | 5000 |      1 |
| SCOTT  | 3000 |      2 |
| FORD   | 3000 |      2 |
| JONES  | 2975 |      4 |
| BLAKE  | 2850 |      5 |
| CLARK  | 2450 |      6 |
| ALLEN  | 1600 |      7 |
| TURNER | 1500 |      8 |
| MILLER | 1300 |      9 |
| WARD   | 1250 |     10 |
| MARTIN | 1250 |     10 |
| ADAMS  | 1100 |     12 |
| JAMES  |  950 |     13 |
```

```
| SMITH  |  800 |     14 |
+--------+------+--------+
14 rows in set (0.00 sec)
```

문제 140. 부서번호, 이름, 월급, 순위를 출력하는데 순위가 부서번호별로 각각 월급이 높은 순서대로 순위를 출력하시오 !

```
SQL> select deptno, ename, sal, rank() over (partition by deptno order by sal desc)
       from emp;
```

```
mysql> select deptno, ename, sal,
    ->        (select count(*) + 1 from emp where sal > e.sal and deptno = e.deptno) 순     위
    ->   from emp e
    ->   order by deptno, sal desc;
+--------+--------+------+--------+
| deptno | ename  | sal  | 순위   |
+--------+--------+------+--------+
|     10 | KING   | 5000 |      1 |
|     10 | CLARK  | 2450 |      2 |
|     10 | MILLER | 1300 |      3 |
|     20 | SCOTT  | 3000 |      1 |
|     20 | FORD   | 3000 |      1 |
|     20 | JONES  | 2975 |      3 |
|     20 | ADAMS  | 1100 |      4 |
|     20 | SMITH  |  800 |      5 |
|     30 | BLAKE  | 2850 |      1 |
|     30 | ALLEN  | 1600 |      2 |
|     30 | TURNER | 1500 |      3 |
|     30 | WARD   | 1250 |      4 |
|     30 | MARTIN | 1250 |      4 |
|     30 | JAMES  |  950 |      6 |
+--------+--------+------+--------+
14 rows in set (0.00 sec)
```

문제 141. 부서번호, 해당 부서번호에 속한 사원들의 이름을 가로로 출력하시오 !

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

```
mysql> select deptno, group_concat(ename order by ename asc separator ',')
    ->   from emp
    ->   group by deptno;
+--------+-----------------------------------------------------+
| deptno | group_concat(ename order by ename asc separator ',') |
+--------+-----------------------------------------------------+
|     10 | CLARK,KING,MILLER                                   |
|     20 | ADAMS,FORD,JONES,SCOTT,SMITH                        |
|     30 | ALLEN,BLAKE,JAMES,MARTIN,TURNER,WARD                |
+--------+-----------------------------------------------------+
3 rows in set (0.00 sec)
```

문제 142. 아래와 같이 결과를 출력하시오 !

```
+--------+-----------------------------------------------------------------------+
| deptno | group_concat(enamesal order by ename asc separator ',')               |
+--------+-----------------------------------------------------------------------+
|     10 | CLARK(2450),KING(5000),MILLER(1300)                                   |
|     20 | ADAMS(1100),FORD(3000),JONES(2975),SCOTT(3000),SMITH(800)             |
|     30 | ALLEN(1600),BLAKE(2850),JAMES(950),MARTIN(1250),TURNER(1500),WARD(1250) |
+--------+-----------------------------------------------------------------------+
3 rows in set (0.00 sec)

       mysql> select deptno, group_concat(enamesal order by ename asc separator ',')
           ->   from (select deptno, ename, concat( ename, '(', sal, ')') as enamesal
           ->           from emp) ee
           ->   group by deptno;
```

문제 143. 이름과 부서위치를 출력하시오 ! (MySQL에서는 1999ansi 문법으로 수행한다)

※ MySQL 에서의 조인은 1999 ansi 문법으로 수행한다

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

문제 144. 아래의 오라클 조인문장을 MySQL에서 구현하시오 !

```
SQL> select e.ename, d.loc
       from emp e, dept d
       where e.deptno (+) = d.deptno;

mysql> select e.ename, d.loc
    ->   from emp e right outer join dept d
    ->   on ( e.deptno = d.deptno);
+--------+----------+
| ename  | loc      |
+--------+----------+
| KING   | NEW YORK |
| BLAKE  | CHICAGO  |
| CLARK  | NEW YORK |
| JONES  | DALLAS   |
| MARTIN | CHICAGO  |
| ALLEN  | CHICAGO  |
| TURNER | CHICAGO  |
| JAMES  | CHICAGO  |
| WARD   | CHICAGO  |
| FORD   | DALLAS   |
| SMITH  | DALLAS   |
| SCOTT  | DALLAS   |
| ADAMS  | DALLAS   |
| MILLER | NEW YORK |
| NULL   | BOSTON   |
+--------+----------+
15 rows in set (0.03 sec)
```

문제 145. 아래의 오라클의 full outer join 을 MySQL에서 구현하시오 !

```
----------------------------------------
insert into emp(empno, ename, sal, deptno)
  values (2929, 'jack', 4500, 70);
----------------------------------------
```

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

mysql> select e.ename, d.loc
    ->   from emp e left outer join dept d
    ->   on (e.deptno = d.deptno)
    -> union
    -> select e.ename, d.loc
    ->   from emp e right outer join dept d
    ->   on (e.deptno = d.deptno);
+--------+----------+
| ename  | loc      |
+--------+----------+
| KING   | NEW YORK |
| BLAKE  | CHICAGO  |
| CLARK  | NEW YORK |
| JONES  | DALLAS   |
| MARTIN | CHICAGO  |
| ALLEN  | CHICAGO  |
| TURNER | CHICAGO  |
| JAMES  | CHICAGO  |
| WARD   | CHICAGO  |
| FORD   | DALLAS   |
| SMITH  | DALLAS   |
| SCOTT  | DALLAS   |
| ADAMS  | DALLAS   |
| MILLER | NEW YORK |
| jack   | NULL     |
| NULL   | BOSTON   |
+--------+----------+
16 rows in set (0.00 sec)
```

* 점심시간 문제

문제 146. (서브쿼리) JONES의 월급보다 더 많은 월급을 받는 사원들의 이름과 월급을 출력하시오 !

```
mysql> select ename, sal
    ->   from emp
    ->   where sal > (select sal from emp where ename = 'JONES');
+-------+------+
| ename | sal  |
+-------+------+
| KING  | 5000 |
| FORD  | 3000 |
| SCOTT | 3000 |
| jack  | 4500 |
+-------+------+
4 rows in set (0.00 sec)
```

문제 147. SCOOTT의 월급을 0으로 변경하시오 !

```
mysql> update emp set sal = 0 where ename = 'scott';
Query OK, 1 row affected (0.11 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> rollback;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from emp;
+-------+--------+-----------+------+------------+------+------+--------+
| empNO | ENAME  | JOB       | MGR  | HIREDATE   | SAL  | COMM | deptNO |
+-------+--------+-----------+------+------------+------+------+--------+
|  7839 | KING   | PRESIDENT | NULL | 1981-11-17 | 5000 | NULL |     10 |
|  7698 | BLAKE  | MANAGER   | 7839 | 1981-05-01 | 2850 | NULL |     30 |
|  7782 | CLARK  | MANAGER   | 7839 | 1981-05-09 | 2450 | NULL |     10 |
|  7566 | JONES  | MANAGER   | 7839 | 1981-04-01 | 2975 | NULL |     20 |
|  7654 | MARTIN | SALESMAN  | 7698 | 1981-09-10 | 1250 | 1400 |     30 |
|  7499 | ALLEN  | SALESMAN  | 7698 | 1981-02-11 | 1600 |  300 |     30 |
|  7844 | TURNER | SALESMAN  | 7698 | 1981-08-21 | 1500 |    0 |     30 |
|  7900 | JAMES  | CLERK     | 7698 | 1981-12-11 |  950 | NULL |     30 |
|  7521 | WARD   | SALESMAN  | 7698 | 1981-02-23 | 1250 |  500 |     30 |
|  7902 | FORD   | ANALYST   | 7566 | 1981-12-11 | 3000 | NULL |     20 |
|  7369 | SMITH  | CLERK     | 7902 | 1980-12-09 |  800 | NULL |     20 |
|  7788 | SCOTT  | ANALYST   | 7566 | 1982-12-22 |    0 | NULL |     20 |
|  7876 | ADAMS  | CLERK     | 7788 | 1983-01-15 | 1100 | NULL |     20 |
|  7934 | MILLER | CLERK     | 7782 | 1982-01-11 | 1300 | NULL |     10 |
|  2929 | jack   | NULL      | NULL | NULL       | 4500 | NULL |     70 |
+-------+--------+-----------+------+------------+------+------+--------+
15 rows in set (0.00 sec)
```

※ 자동 커밋이 default가 켜있기 때문에 rollback이 안된다.

```
mysql> select @@autocommit;
+--------------+
| @@autocommit |
+--------------+
|            1 |
+--------------+
1 row in set (0.02 sec)

mysql> set autocommit = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> select @@autocommit;
+--------------+
| @@autocommit |
+--------------+
|            0 |
+--------------+
1 row in set (0.00 sec)
```

문제 148. 사원 테이블을 전체 delete 하시오 그리고 rollback되는지 확인하시오 !

```
mysql> delete from emp;
Query OK, 15 rows affected (0.00 sec)
```

```
mysql> rollback;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from emp;
+-------+--------+-----------+------+------------+------+------+--------+
| empNO | ENAME  | JOB       | MGR  | HIREDATE   | SAL  | COMM | deptNO |
+-------+--------+-----------+------+------------+------+------+--------+
|  7839 | KING   | PRESIDENT | NULL | 1981-11-17 | 5000 | NULL |     10 |
|  7698 | BLAKE  | MANAGER   | 7839 | 1981-05-01 | 2850 | NULL |     30 |
|  7782 | CLARK  | MANAGER   | 7839 | 1981-05-09 | 2450 | NULL |     10 |
|  7566 | JONES  | MANAGER   | 7839 | 1981-04-01 | 2975 | NULL |     20 |
|  7654 | MARTIN | SALESMAN  | 7698 | 1981-09-10 | 1250 | 1400 |     30 |
|  7499 | ALLEN  | SALESMAN  | 7698 | 1981-02-11 | 1600 |  300 |     30 |
|  7844 | TURNER | SALESMAN  | 7698 | 1981-08-21 | 1500 |    0 |     30 |
|  7900 | JAMES  | CLERK     | 7698 | 1981-12-11 |  950 | NULL |     30 |
|  7521 | WARD   | SALESMAN  | 7698 | 1981-02-23 | 1250 |  500 |     30 |
|  7902 | FORD   | ANALYST   | 7566 | 1981-12-11 | 3000 | NULL |     20 |
|  7369 | SMITH  | CLERK     | 7902 | 1980-12-09 |  800 | NULL |     20 |
|  7788 | SCOTT  | ANALYST   | 7566 | 1982-12-22 |    0 | NULL |     20 |
|  7876 | ADAMS  | CLERK     | 7788 | 1983-01-15 | 1100 | NULL |     20 |
|  7934 | MILLER | CLERK     | 7782 | 1982-01-11 | 1300 | NULL |     10 |
|  2929 | jack   | NULL      | NULL | NULL       | 4500 | NULL |     70 |
+-------+--------+-----------+------+------------+------+------+--------+
15 rows in set (0.00 sec)
```

문제 149. dept 테이블 생성 스크립트를 이용해서 dept2 테이블을 생성하고 dept2 테이블을 select한 후에 rollback을 하면 dept2 테이블이 어떻게 되는지 확인해 보시오 !

```
-----------------------------------------------------
CREATE TABLE dept2
(deptNO int,
DNAME VARCHAR(14),
LOC VARCHAR(13) );

INSERT INTO dept2 VALUES (10, 'ACCOUNTING', 'NEW YORK');
INSERT INTO dept2 VALUES (20, 'RESEARCH', 'DALLAS');
INSERT INTO dept2 VALUES (30, 'SALES', 'CHICAGO');
INSERT INTO dept2 VALUES (40, 'OPERATIONS', 'BOSTON');
-----------------------------------------------------

mysql> select * from dept2;
+--------+------------+----------+
| deptNO | DNAME      | LOC      |
+--------+------------+----------+
|     10 | ACCOUNTING | NEW YORK |
|     20 | RESEARCH   | DALLAS   |
|     30 | SALES      | CHICAGO  |
|     40 | OPERATIONS | BOSTON   |
+--------+------------+----------+
4 rows in set (0.00 sec)

mysql> rollback;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from dept2;
Empty set (0.00 sec)
```

문제 150. hive에서 emp 테이블을 drop하고 오라클의 emp를 hive로 이행하시오 !

```
hive> drop table emp;

[orcl:~]$ sh table_import.sh scott tiger emp
                      :
                      :
19/01/04 16:05:46 INFO hive.HiveImport: OK
19/01/04 16:05:46 INFO hive.HiveImport: Time taken: 0.427 seconds
19/01/04 16:05:46 INFO hive.HiveImport: Hive import complete.

hive> select * from emp;
OK
7369.0  SMITH   CLERK   7902.0  1980-12-17 00:00:00.0   800.0   NULL    20.0
7499.0  ALLEN   SALESMAN        7698.0  1981-02-20 00:00:00.0   1600.0  300.0   30.0
7521.0  WARD    SALESMAN        7698.0  1981-02-22 00:00:00.0   1250.0  500.0   30.0
7566.0  JONES   MANAGER 7839.0  1981-04-02 00:00:00.0   2975.0  NULL    20.0
7654.0  MARTIN  SALESMAN        7698.0  1981-09-28 00:00:00.0   1250.0  1400.0  30.0
7698.0  BLAKE   MANAGER 7839.0  1981-05-01 00:00:00.0   2850.0  NULL    30.0
7782.0  CLARK   MANAGER 7839.0  1981-06-09 00:00:00.0   2450.0  NULL    10.0
7788.0  SCOTT   ANALYST 7566.0  1987-04-19 00:00:00.0   3000.0  NULL    20.0
7839.0  KING    PRESIDENT       NULL    1981-11-17 00:00:00.0   5000.0  NULL    10.0
7844.0  TURNER  SALESMAN        7698.0  1981-09-08 00:00:00.0   1500.0  0.0     30.0
7876.0  ADAMS   CLERK   7788.0  1987-05-23 00:00:00.0   1100.0  NULL    20.0
7900.0  JAMES   CLERK   7698.0  1981-12-03 00:00:00.0   950.0   NULL    30.0
7902.0  FORD    ANALYST 7566.0  1981-12-03 00:00:00.0   3000.0  NULL    20.0
7934.0  MILLER  CLERK   7782.0  1982-01-23 00:00:00.0   1300.0  NULL    10.0
Time taken: 4.03 seconds, Fetched: 14 row(s)
```