00 목차

2018년 7월 3일 화요일 오전 9:58

Table of Contents - Pages

01 하둡 살펴보기 (하둡의 정의)
02 하둡 개발 준비 (하둡 설치)
03 하둡 분산 파일 시스템 명령어
04 하이브(Hive) 설치 및 사용방법 총정리
05 스쿱으로 오라클과 연동
06 타조 설치 및 사용방법 총정리
07 Pig 설치 및 사용방법 총정리
08 MongoDB in Ubuntu 설치 및 사용방법 총정리
09 R과 하둡 연동하기
10 자바를 이용해서 HDFS의 데이터를 SELECT 하기
11 MySQL in CentOS
12 실제 리눅스 서버에 하둡 멀티노드 구성

01 하둡 살펴보기 (하둡의 정의)

2018년 7월 3일 화요일 오전 9:58

Table of Contents

- 하둡이란 ?
- 하둡이 나온 배경
- 하둡의 장점
- 하둡 에코 시스템
- 하이브 사용 예
- 하둡의 단점
- 주요 하둡 배포판

하둡이란 ?

대용량 데이터를 분산 처리할 수 있는 자바 기반의 오픈소스 프레임워크

하둡이 나온 배경

구글에서 구글에 쌓여지는 수많은 빅데이터들을 구글에서도 처음에는 RDBMS(오라클)에 입력하고 데이터를 저장하고 처리하려고 시도했으나 너무 데이터가 많아 실패했다.
그래서 자체적으로 빅데이터를 저장할 기술을 개발하고 대외적으로 이 기술에 대한 논문을 발표했다.
그 논문을 더그 커팅(하둡을 만든 사람)이 읽고 자바로 구현했다.

구현한 것을 뭐라고 이름 지을까 하던 더그 커팅이 애기가 노란 코끼리 장난감을 들고 놀면서 Hadoop 이라고 한 것을 듣고 그렇게 지었다.

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

데이터 분석하려는 개발자들이 자바 기반의 Hadoop을 사용하기 어려웠기 때문에
Hive(벌떼), Pig(돼지), Tajo 를 개발해 조금 쉽게 다루고자 했다.

위 Hive(벌떼), Pig(돼지), Tajo 를 NoSQL 이라고 하는데,
SQL과 비슷한 언어로 하둡의 데이터를 분석하기 위해 쓰인다.
(NoSQL = Not only SQL)

하둡의 장점

1. 무료
2. 분산 처리가 가능 - 여러 대의 노드를 묶어서 마치 하나의 서버처럼 보이게 하고 여러 노드의 자

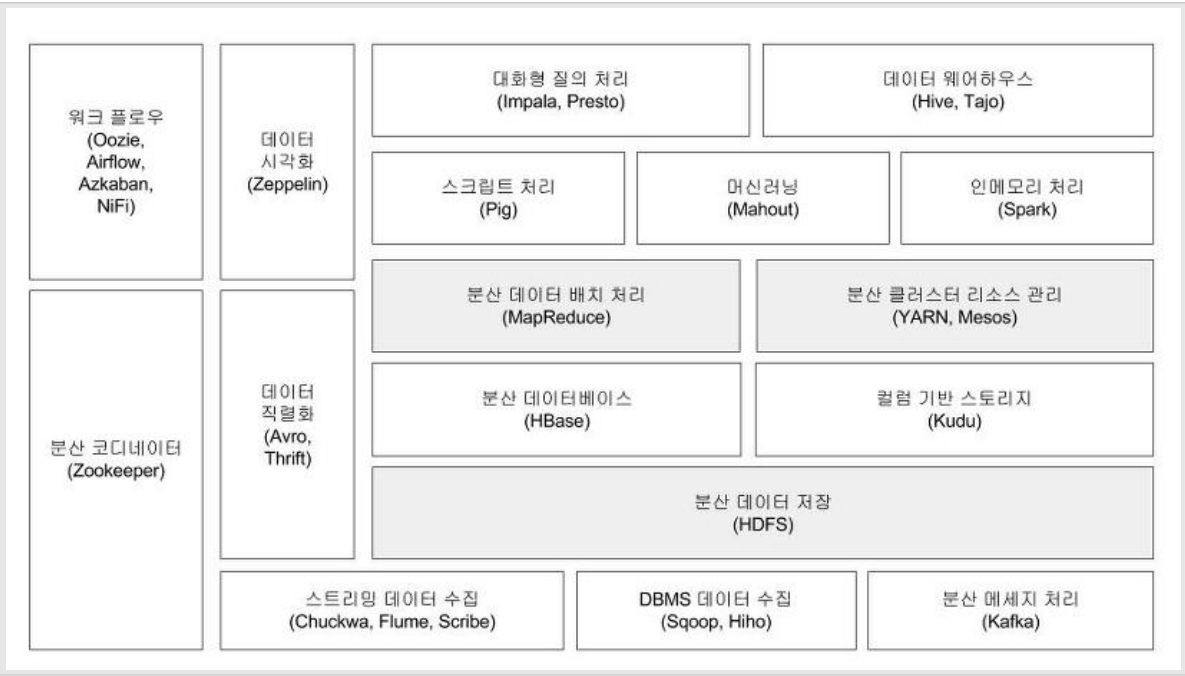
원을 이용해서 데이터를 처리하기 때문에 처리하는 속도가 빠른 장점이 있다.

예)
한 대의 서버로 1TB의 데이터를 처리하는데 걸리는 시간이 2시간 반이라고 하면
하둡으로 여러 대의 서버로 병렬로 작업 한다면 2분 내에 데이터를 읽을 수 있다.

실제 예)
2008년 뉴욕타임즈의 130년 분량의 신문기사 1100만 페이지를 하둡을 이용해서 하루 만에 pdf로 변환을 했고 비용이 200만 원 밖에 들지 않았다.
만약, 하둡이 아닌 일반 서버로 처리했을 경우 14년이 걸렸을 것으로 예상한다.

3. 데이터 복구 가능 (여러 컴퓨터에 복제해놓으면 된다)

하둡 에코 시스템



MapReduce와 HDFS를 더그 커팅이 자바로 만들었고
그것을 쉽게 하기 위한 것이 Hive Pig Sqoop Zookeeper 이다.

빅데이터 분석	R, Python 등을 이용해 분석
↑	↑
빅데이터 저장	Hbase, MongoDB, Cassandra, CouchDB
↑	↑
분산처리 지원	Hive, Pig, Sqoop, Zookeeper
↑	↑
분산 배치 관리	하둡(Hadoop) - MapReduce
분산 파일 관리	하둡(Hadoop) - HDFS

실제 예) 모 생명사에서 보험 상품에 대해 관심이 있는 고객에 대해서 분석을 시도

분석 목표: 관심 있는 고객에 대해서만 상품 소개 및 기존 고객 관리

관심 있는 고객을 선별하기 위해 하둡에서 데이터를 필터링 한다. (40대 만 추출)

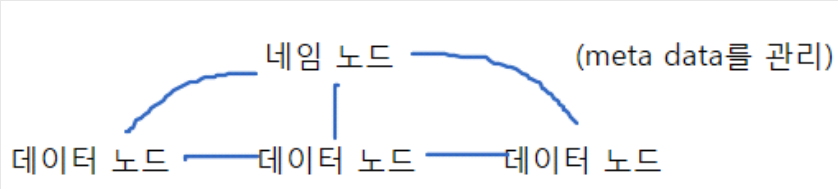
40대 고객층을 상담 내용과 내용의 단어들을 활용해 k-means 로 군집분석을 한다.
크게 상품에 관심이 있는 고객과 관심없는 고객으로 군집화 한다.

차후과제: 고객의 통화내용이 녹음이 되는데 음성을 텍스트화 하여 그 텍스트를 단어로 정리해서 matrix로 만들어서 군집 분석을 하자

하이브 사용 예

hive> select * from movies limit 5;

위와 같이 select를 하면 네임노드에 가서 movies 라는 데이터가 어느 데이터 노드에 있는지 물어본다



야후의 경우는 약 5만 대를 연결해서 하둡을 사용하고, 페이스북은 약 1만 대 이상의 하둡 클러스터를 이용하고 있다

아마존 서버에 하둡 클러스터를 구성해보면 가장 하둡 설치와 관리에 대해서 아주 많이 배울 수 있다.
저렴한 구축 비용과 비용 대비 빠른 데이터 처리

하둡의 단점

1. 무료이다 보니 유지보수가 어렵다
2. 네임노드가 다운되면 고가용성이 지원되지 않는다.
3. 한번 저장된 파일은 수정할 수 없다. (기존 데이터에 append는 되는데 update(수정)은 안됨)

주요 하둡 배포판

리눅스도 CentOS, redhat, Ubunto 처럼 배포판이 있듯이 하둡도 있다.

1. Cloudera 업체에서 나온 CDH : 가장 높은 신뢰
2. Hortonworks 에서 나온 HDP
3. 아마존에서 나온 EMR (Elastic MapReduce)
4. Hstreaming 에서 나온 Hstreaming

---

02 하둡 개발 준비 (하둡 설치)

2018년 7월 3일 화요일 오전 9:58

Table of Contents

- 1. java 설치
- 2. java 환경 설정
- 3. keygen 생성
- 4. 하둡 설치 파일을 다운받아 압축을 푼다
- 5. 하둡 홈 디렉토리를 설정한다
- 6. 하둡 환경설정을 하기 위해 아래의 4개의 파일을 세팅해야 한다
- 7. 하둡 네임노드를 포맷한다
- 8. 하둡을 시작시킨다
- 9. 하둡이 잘 시작되었는지 확인한다
- Hive 설치
- 하둡 시스템이 정상인지 확인하는 명령어

하둡 홈페이지 : http://hadoop.apache.org/

하둡 설치

```
1. java 설치
2. java 환경 설정
3. keygen 생성
4. 하둡 설치 파일을 다운받아 압축을 푼다
5. 하둡 홈 디렉토리를 설정한다
6. 하둡을 운영하기 위한 xml 파일 3개를 수정한다
7. 하둡 네임노드를 포맷한다
8. 하둡을 시작시킨다
9. 하둡이 잘 시작되었는지 확인한다
```

첨부파일 실습_00.Hadoop install.txt

1. java 설치

```
자바 설치 이유
하둡이 자바로 개발되었고 데몬을 구동할 때에 jar 파일을 수정하기 때문에 반드시 자바가 필요하다.
(자바는 jdk 1.6 버전 이상 설치를 권장)

버전 확인 (Xshell 에서)
$ java -version
java version "1.6.0_18"
```

2. java 환경 설정

$ su -
Password: oracle
[root@edydr1p2 ~]# mkdir -p /u01/app/java
[root@edydr1p2 ~]# mv /home/oracle/jdk-7u60-linux-i586.gz /u01/app/java/
[root@edydr1p2 ~]# cd /u01/app/java/
[root@edydr1p2 java]# tar xvzf jdk-7u60-linux-i586.gz

[root@edydr1p2 java]# chown -R root:root jdk1.7.0_60
[root@edydr1p2 java]# exit

[oracle@edydr1p2 ~]$ vi ~/.bash_profile

```
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/bin

export PATH
#export JAVA_HOME=/usr/java/jdk1.6.0_18
export JAVA_HOME=/u01/app/java/jdk1.7.0_60
export PATH=/u01/app/java/jdk1.7.0_60/bin:$PATH
export CLASSPATH=.:/usr/java/jdk1.7.0_60/lib:$CLASSPATH
export PATH=$JAVA_HOME/bin:$PATH
export PS1='[`echo $ORACLE_SID`:\W]$ '

export ORACLE_BASE=/u01/app/oracle
export ORACLE_SID=orcl
export ORACLE_HOME=/u01/app/oracle/product/11.2.0/dbhome_1
export PATH=/home/oracle/R-3.2.3/bin:$PATH
export PATH=$ORACLE_HOME/bin:$PATH
unset LANG
echo -e "여기는 운영서버입니다. 조심히 작업하세요~ "
echo -e "저한테 왜 그러시는 건데요 ㅡㅡ"
```

주석처리 해주고 그 아래
export JAVA_HOME=/u01/app/java/jdk1.7.0_60
export PATH=/u01/app/java/jdk1.7.0_60/bin:$PATH
export CLASSPATH=.:/usr/java/jdk1.7.0_60/lib:$CLASSPATH
이 세 줄 복붙

[orcl:~]$ . .bash_profile
[orcl:~]$ java -version

3. keygen 생성

하둡은 SSH 프로토컬을 이용해 하둡 클러스트 간의 내부 통신을 수행한다

하둡을 다 설치하고 나서 하둡을 시작하는 명령어인 start-all.sh 쉘 스크립트를 수행하면 네임노드가 설치된 서버에서 데이터 노드가 설치된 서버로 접근해 데이터 노드와 테스크 트래커를 구동하게 된다.
그런데 이때 ssh를 이용할 수 없다면 하둡을 실행할 수 없다.

네임노드에서 공개키를 설정하고 이 공개키(authorized_keys)를 다른 데이터 노드에 다 복사해줘야 한다.

이 키를 가지고 있으면 ssh로 다른 노드에 접속할 때 패스워드 없이 접속할 수 있다.

```
[oracle@edydr1p2 ~]$ cd
[oracle@edydr1p2 ~]$ rm -rf .ssh
[oracle@edydr1p2 ~]$ ssh-keygen -t rsa

cat /home/oracle/.ssh/id_rsa.pub >> /home/oracle/.ssh/authorized_keys
```

아래의 3개의 접속 명령어를 수행했을때 패스워드 물어보면 안되고 바로 접속되는지 확인하시오
1. $ ssh edydr1p0.us.oracle.com
2. $ ssh edydr1p0
3. $ ssh localhost

4. 하둡 설치 파일을 다운받아 압축을 푼다

```
[oracle@edydr1p2 ~]$ mkdir -p /u01/app/hadoop
[oracle@edydr1p2 ~]$ cd /u01/app/hadoop/

[oracle@edydr1p2 hadoop]$ wget http://mirror.apache-kr.org/hadoop/common/hadoop-1.2.1/hadoop-1.2.1.tar.gz
[oracle@edydr1p2 hadoop]$ tar xvzf hadoop-1.2.1.tar.gz

[orcl:hadoop]$ ls -l hadoop-1.2.1.tar.gz
-rw-r--r-- 1 oracle oinstall 63851630 Jul  3 13:58 hadoop-1.2.1.tar.gz

[oracle@edydr1p2 hadoop]$ rm hadoop-1.2.1.tar.gz    # 압축파일 날려버린다
```

5. 하둡 홈 디렉토리를 설정한다

```
[oracle@edydr1p2 hadoop]$ cd
[oracle@edydr1p2 ~]$ vi .bash_profile
==================================================
export HADOOP_HOME=/u01/app/hadoop/hadoop-1.2.1
                                # 오른쪽의 절대경로를 HADOOP_HOME이라는 이름으로 지정해준 것
export PATH=$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$PATH
```

```
=====================================================
[oracle@edydr1p2 ~]$ . .bash_profile
```

6. 하둡 환경설정을 하기 위해 아래의 4개의 파일을 세팅해야 한다

1. hadoop-env.sh : 자바 홈 디렉토리와 hadoop 홈디렉토리가 어딘지 지정한다
2. core-site.xml : 하둡의 네임노드가 어느 서버인지를 지정한다
3. mapred-site.xml : java 로 만들어진 MapReduce 프레임워크와 관련된 정보를 지정하는 파일
4. hdfs-site.xml : 하둡 파일 시스템인 HDFS(Hadoop Distributed File System)와 관련된 정보를 저장하는 파일

6_1. hadoop-env.sh

```
[oracle@edydr1p2 ~]$ cd $HADOOP_HOME/conf
[orcl:hadoop-1.2.1]$ pwd
/u01/app/hadoop/hadoop-1.2.1

[oracle@edydr1p2 conf]$ vi hadoop-env.sh
=====================================================
# The java implementation to use. Required.
# export JAVA_HOME=/usr/lib/j2sdk1.5-sun              << 여기 아래에 밑에 3줄을 복붙해주면 됨
export JAVA_HOME=/u01/app/java/jdk1.7.0_60
export HADOOP_HOME=/u01/app/hadoop/hadoop-1.2.1
export HADOOP_HOME_WARN_SUPPRESS=1
=====================================================
```

6_2. core-site.xml

```
지금 single 이기 때문에 내 컴퓨터가 네임노드이자 데이터 노드이다

[oracle@edydr1p2 conf]$ vi core-site.xml
=====================================================
<configuration>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://localhost:9000</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/u01/app/hadoop/hadoop-1.2.1/hadoop-${user.name}</value>
  </property>
</configuration>
=====================================================
```

6_3. mapred-site.xml

```
[oracle@edydr1p2 conf]$ vi mapred-site.xml
=====================================================
<configuration>
  <property>
    <name>mapred.job.tracker</name>
    <value>localhost:9001</value>
  </property>
  <property>
    <name>mapred.local.dir</name>
    <value>${hadoop.tmp.dir}/mapred/local</value>
  </property>
  <property>
    <name>mapred.system.dir</name>
    <value>${hadoop.tmp.dir}/mapred/system</value>
  </property>
</configuration>
=====================================================

[oracle@edydr1p2 conf]$ mkdir /u01/app/hadoop/hadoop-1.2.1/dfs
[oracle@edydr1p2 conf]$ mkdir /u01/app/hadoop/hadoop-1.2.1/dfs/name
[oracle@edydr1p2 conf]$ mkdir /u01/app/hadoop/hadoop-1.2.1/dfs/data
```

6_4. hdfs-site.xml

```
[oracle@edydr1p2 conf]$ vi hdfs-site.xml
=====================================================
<configuration>
  <property>
    <name>dfs.name.dir</name>
    <value>/u01/app/hadoop/hadoop-1.2.1/dfs/name</value>
  </property>
  <property>
    <name>dfs.name.edits.dir</name>
    <value>${dfs.name.dir}</value>
  </property>
  <property>
    <name>dfs.data.dir</name>
    <value>/u01/app/hadoop/hadoop-1.2.1/dfs/data</value>
  </property>
</configuration>
=====================================================
```

7. 하둡 네임노드를 포맷한다

```
[oracle@edydr1p2 conf]$ cd
```

```
[oracle@edydr1p2 ~]$ hadoop namenode -format
```

8. 하둡을 시작시킨다

```
[oracle@edydr1p2 ~]$ start-all.sh

하둡 중지
[oracle@edydr1p2 ~]$ stop-all.sh
```

9. 하둡이 잘 시작되었는지 확인한다

```
[oracle@edydr1p2 ~]$ jps
```

21332 JobTracker
20970 NameNode
21248 SecondaryNameNode
21585 Jps
21100 DataNode
21474 TaskTracker

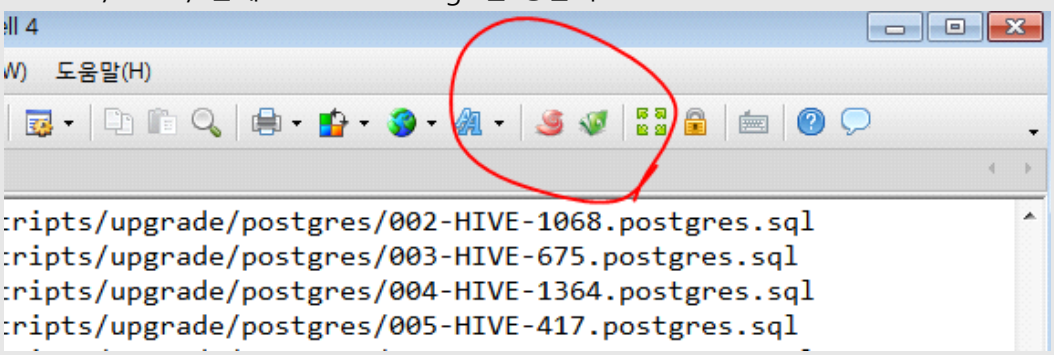
Hive 설치

" 자바를 몰라도 RDBMS에 익숙한 데이터 분석가들을 위해서 SQL을 이용해서 하둡의 맵리듀싱 프로그램을 지원하는 프로그램 "

페이스북에서 만든 오픈소스

HiveQL
1. SELECT는 되는데 update와 delete 명령어는 지원 안함
2. from 절의 서브쿼리는 사용 가능
3. select 문 사용 시 having 절은 사용 불가능
4. PL/SQL의 프로시저는 hive2.0 부터 가능

1. home/oracle/ 밑에 hive-0.12.0.tar.gz 를 넣는다



2. hive 설치 파일의 압축을 푼다
$ tar xvzf hive-0.12.0.tar.gz

```
3. hive 로 접속한다
$ cd /home/oracle/hive-0.12.0/bin
$ ./hive

hive> show tables;
OK
Time taken: 6.525 seconds
```

문제 1) hive 에서 emp 테이블을 생성하시오

답)

```
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 ;
```

문제 2) 하둡 파일 시스템에 emp2.csv를 올리시오

답)

```
hive> exit;
[orcl:bin]$ cd
[orcl:~]$ hadoop fs -put emp2.csv emp2.csv
[orcl:~]$ hadoop fs -ls emp2.csv
Found 1 items
-rw-r--r--   3 oracle supergroup        644 2018-07-03 15:20 /user/oracle/emp2.csv
```

문제 3) emp2.csv 를 emp에 로드하시오

답)

```
emp2.csv를 /home/oracle에 올린다

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

hive> select * from emp;
```

하둡 시스템이 정상인지 확인하는 명령어

```
hive> exit;
[orcl:bin]$ cd
[orcl:~]$ jps
23598 JobTracker
24583 Jps
23208 NameNode
23733 TaskTracker
23501 SecondaryNameNode
23338 DataNode
```

JobTracker	하둡 클러스터에 등록된 전체 잡의 스케쥴링을 관리하고 모니터링하는 데몬
Jps	현재 jps 명령어 수행한 프로세서 (나)
NameNode	HDFS의 모든 메타 데이터(data의 위치정보)를 관리하고 클라이언트가 HDFS에 저장된 파일에 접근할 수 있게 해준다.
TaskTracker	사용자가 설정한 맵리듀스 프로그램을 실행하는 역활을 하며 하둡 데이터 노드에서 실행되는 데몬
SecondaryNameNode	하둡 보조 네임노드로 네임노드의 파일 시스템 이미지 파일을 주기적으로 갱신하는 역할을 갖는 노드
DataNode	HDFS의 데이터를 입력하면 입력 데이터는 32MB의 블록으로 나눠져서 여러 대의 데이터 노드에 분산되어 저장된다. 그 데이터를 저장하는 노드

하둡 운영에 문제가 생겼을 때 조치 방법

```
$ jps
를 수행했을때 위의 6개의 데몬 말고 RunJar 라는 프로세서가 뜨면 문제가 있으므로 반드시 KILL 시킨다
$ kill -9 숫자(RunJar 앞의)
```

문제 4) 자동화 스크립트의 위의 하둡 파일 관리 명령어를 자동화하시오

보기)

```
$ sh m2.sh

1. sar 그래프를 ...
2. ..
.
.
7. 하둡 파일 시스템을 중단 시키려면 7번을
8. 하둡 파일 시스템을 시작 시키려면 8번을
9. 하둡 파일 시스템의 상태를 확인하려면 9번을
```

답)

```
7) stop-all.sh
8) start-all.sh
9) jps
```

---

03 하둡 분산 파일 시스템 명령어

2018년 7월 3일 화요일 오전 9:58

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 형태로 화면을 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 합친 후 하나의 파일로 복사하는 명령어
12. mv : 파일을 이동하는 명령어
13. moveFromLocal : 복사 후 원본 파일 삭제
14. rm : 파일 삭제
15. rmr : 디렉토리 삭제
16. count : 지정된 디렉토리의 파일의 개수 확인 (리눅스에선 wc)
17. tail : 파일의 마지막 내용 확인
18. chmod : 권한 변경
19. chown : 소유자 변경
20. tochz : 0 바이트의 파일 생성
21. stat : 통계정보 조회
22. expuge : 휴지통 비우기
23. grep : 파일에서 특정 문자의 라인을 검색
25. awk : 파일의 특정 컬럼을 검색

문제 5) 하둡 파일 시스템 명령어의 help를 조회하시오

답)

```
[orcl:~]$ hadoop fs -help
[orcl:~]$ hadoop fs -help du
```

문제 6) 겨울왕국 대본 winter.txt를 하둡 파일 시스템에 올리시오

답)

```
[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     114548 2018-07-03 16:29 /user/oracle/winter.txt
```

문제 7) 하둡 파일 시스템에 있는 emp2.csv 파일을 올리세요

답)

```
[orcl:~]$ hadoop fs -put emp2.csv emp2.csv
```

문제 8) 하둡 파일 시스템에 emp2.csv를 cat으로 조회하시오

답)

```
[orcl:~]$ hadoop fs -cat emp2.csv
```

문제 9) 직업이 SALESMAN인 사원들의 모든 데이터를 출력하시오

답)

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

결과)

```
[orcl:~]$ hadoop fs -cat emp2.csv | grep -i "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 10) alias를 만들어서 아래와 같이 하둡 명령어를 쓸 수 있도록 하시오

보기)

```
$ h -cat emp2.csv
```

답)

```
[orcl:~]$ alias h='hadoop fs'
[orcl:~]$ h -cat emp2.csv
```

---

04 하이브(Hive) 설치 및 사용방법 총정리

2018년 7월 3일 화요일 오전 9:58

Table of Contents

- Hive에서 조인 (1999 ANSI 조인 문법)
- Hive에서 data 분석 함수 사용해보기 (sum, rank 등)
- Hive에서 지원하는 내장 함수 (문자, 날짜, 숫자함수 등)
- Hive에서 from절의 서브쿼리
- Hive에서 with절
- Hive에서 CTAS 문법

문제 11) Hive에 접속할 때 어느 디렉토리에서든 편하게 hive라고 하면 접속되게 하시오

답)

```
[orcl:~]$ cd hive-0.12.0
[orcl:hive-0.12.0]$ pwd
/home/oracle/hive-0.12.0
위를 복사한 뒤
[orcl:hive-0.12.0]$ cd
[orcl:~]$ vi .bash_profile
==================================================
export HIVE_HOME=/home/oracle/hive-0.12.0
export PATH=$HIVE_HOME/bin:$PATH
==================================================
를 추가하고
[orcl:~]$ . .bash_profile
[orcl:~]$ hive
```

문제 12) dept.csv 를 하둡 파일 시스템에 로드하고 dept 테이블을 hive에서 생성한 후 dept.csv를 dept 테이블에 입력하시오

답)

1. 테이블 생성

```
[orcl:~]$ hive
hive>  create table dept
(deptno    int,
dname     string,
```

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

2. 하둡 파일 시스템에 dept2.csv를 put함

```
hive> exit;
[orcl:~]$ cd
[orcl:~]$ h -put dept2.csv dept2.csv
[orcl:~]$ h -cat dept2.csv
```

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

3. hive에서 dept 테이블에 로드

```
[orcl:~]$ hive
hive> load data inpath '/user/oracle/dept2.csv'
    > overwrite into table dept;
hive> select * from dept;
```

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

문제 13) hive에서 월급이 3000인 사원의 사원번호와 이름과 월급을 출력하세요

답)

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

```
Total MapReduce CPU Time Spent: 1 seconds 80 msec
OK
7902    FORD    3000
7788    SCOTT   3000
```

문제 14) 직업, 직업 별 토탈 월급을 출력하는데 직업 별 토탈 월급이 높은 것부터 출력하시오

답)

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

```
    > order by sumsal desc;
MANAGER 8275
ANALYST 6000
SALESMAN        5600
PRESIDENT       5000
CLERK   4150
Time taken: 40.086 seconds, Fetched: 5 row(s)
```

문제 15) 부서번호, 부서번호 별 평균 월급을 출력하는데 부서번호 별 평균 월급이 낮은 것부터 출력하시오

답)

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

문제 16) jps 명령어를 수행했을 때 RunJar 프로세서의 프로세서 번호만 출력하시오

보기)

```
hive에 들어간 뒤 세션 복제하고 jps를 쳐보면 RunJar가 있다.
```

답)

```
jps | grep -i 'runjar' | awk '{print $1}'
```

문제 17) RunJar 프로세서를 Kill 시키는 쉘을 생성하시오

보기)

```
$ sh runjar.sh
```

답)

```
$ vi runjar.sh

#!/bin/bash

pid=`jps | grep -i 'runjar' | awk '{print $1}'

kill -9 $pid
```

문제 18) RunJar 프로세서 kill 시키는 쉘을 자동화 스크립트의 10번으로 추가하시오

답)

```
$ vi m2.sh

10. RunJar 프로세서를 kill 시키려면 10번을

10)
/home/oracle/runjar.sh ;;

위 두 문장 추가
```

Hive에서 조인 (1999 ANSI 조인 문법)

문제 19) dept 테이블이 존재하는지 확인해보고 emp와 dept를 조인해서 이름과 부서위치를 출력하시오

보기)

```
hive에서는 1999 ansi 조인문법만 지원된다

하이브에서 컬럼명 같이 보는 명령어
set hive.cli.print.header=true;
```

답)

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

문제 20) 부서위치, 부서위치 별 토탈 월급을 출력하시오

답)

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

```
BOSTON          NULL
CHICAGO         9400
```

DALLAS 10875
NEW YORK 8750

문제 21) full outer join 이 가능한지 확인해 보시오

답)

```
위에서 full outer join을 씀..
```

Hive에서 data 분석 함수 사용해보기 (sum, rank 등)

문제 22) 이름, 월급, 순위를 출력하는데 순위가 월급이 높은 사원부터 출력되게 하시오

답)

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

문제 23) 부서번호, 이름, 월급, 순위를 출력하는데 순위가 부서번호 별로 각각 월급이 높은 사원 순으로 순위를 부여하시오

답)

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

문제 24) 부서위치, 이름, 월급, 순위 출력하는데 순위가 부서 위치 별로 각각 월급이 높은 사원 순으로 순위를 부여하시오

답)

```
select d.loc, e.ename, e.sal, rank () over (partition by d.loc order by e.sal desc) rnk
        from emp e join dept d
        on ( e.deptno = d.deptno );
```

```
CHICAGO BLAKE   2850    1
CHICAGO ALLEN   1600    2
CHICAGO TURNER  1500    3
CHICAGO MARTIN  1250    4
CHICAGO WARD    1250    4
CHICAGO JAMES   950     6
DALLAS  SCOTT   3000    1
DALLAS  FORD    3000    1
DALLAS  JONES   2975    3
DALLAS  ADAMS   1100    4
DALLAS  SMITH   800     5
NEW YORK        KING    5000    1
NEW YORK        CLARK   2450    2
NEW YORK        MILLER  1300    3
```

문제 25) 사원번호, 이름, 월급, 월급의 누적치가 출력되게 하시오

답)

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

```
SMITH   800     800
ALLEN   1600    2400
WARD    1250    3650
JONES   2975    6625
MARTIN  1250    7875
BLAKE   2850    10725
CLARK   2450    13175
SCOTT   3000    16175
KING    5000    21175
TURNER  1500    22675
ADAMS   1100    23775
JAMES   950     24725
FORD    3000    27725
MILLER  1300    29025
```

Hive에서 지원하는 내장 함수 (문자, 날짜, 숫자함수 등)

1. concat
2. substr
3. upper
4. lower
5. trim
6. rtrim, ltrim
7. regexp_replace
8. to_date
9. round
10. floor
11. ceil
12. year

```
13. month
14. day
```

문제 26) 이름을 출력하는데 이름의 첫 글자만 출력하고 그 첫글자를 소문자로 출력하시오

답)

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

문제 27) 이름, 입사한 년도 4자리를 출력하시오

답)

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

답 2)

```
hive> select ename, substr(hiredate, 1, 4)
    > from emp;
```

문제 28) 이름, 입사한 달을 출력하시오

답)

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

문제 29) 부서번호, 부서번호 별 토탈 월급을 가로로 출력하시오

답)

```
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) dept30
    > from emp;
```

지정해준 컬럼 별칭을 뽑기 위해 첫줄의 세팅을 해줘야 한다

문제 30) 직업, 부서번호, 직업 별 부서번호 별 평균 월급을 출력하시오

답)

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

결과)

```
job             dept10      dept20      dept30
ANALYST         NULL        3000.0      NULL
CLERK           1300.0      950.0       950.0
MANAGER         2450.0      2975.0      2850.0
PRESIDENT       5000.0      NULL        NULL
SALESMAN        NULL        NULL        1400.0
```

문제 31) 직업, 직업 별 인원 수를 출력하시오

답)

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

문제 32) 직업, 부서번호, 직업 별 부서번호 별 인원 수를 출력하시오

답)

```
hive> select job, count(case when deptno=10 then empno end) dept10,
    > count(case when deptno=20 then empno end) dept20,
    > count(case when deptno=30 then empno end) dept30
    > from emp
    > group by job;
```

결과)

```
job             dept10      dept20      dept30
```

```
ANALYST         0       2       0
CLERK           1       2       1
MANAGER         1       1       1
PRESIDENT       1       0       0
SALESMAN        0       0       4
```

Hive가 서브쿼리는 지원되지 않지만, from절의 서브쿼리는 가능

Hive에서 from절의 서브쿼리

문제 33) 사원 테이블에서 월급의 순위가 1등인 사원의 이름과 월급과 순위를 출력하시오

답)

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

설명)
Hive에서는 from절의 서브쿼리 alias를 사용해야 한다

문제 34) 아래의 오라클 SQL을 Hive로 구현하시오

답)

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

```
NULL    29025
10      8750
20      10875
30      9400
```

답 2)

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

```
10      8750
20      10875
30      9400
NULL    29025
```

문제 35) 아래의 오라클 SQL을 Hive로 구현하시오

Oracle)

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

답)

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

```
NULL    29025
10      8750
20      10875
30      9400
```

문제 36) 아래의 오라클 SQL을 Hive로 구현하시오

Oracle)

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

답)

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

```
NULL    NULL    29025
NULL    ANALYST 6000
NULL    CLERK   4150
NULL    MANAGER 8275
NULL    PRESIDENT       5000
NULL    SALESMAN        5600
10      NULL    8750
20      NULL    10875
30      NULL    9400
```

Hive에서 with절

문제 37) 오라클 with 절이 Hive에서도 가능한지 확인하시오

Oracle)

```
with job_sumsal as ( select job, sum(sal) sumsal
                       from emp
```

```
                              group by job )
select job, sumsal
  from job_sumsal
  where sumsal > ( select avg(sumsal)
                             from job_sumsal );
```

답)
안된다.

문제 38) 사원 테이블의 직업의 종류가 몇 가지인지 출력하시오

답)

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

설명)

```
Hive는 update, delete 는 지원 안함
insert는 append로 가능하다
```

문제 39) scott의 월급을 9000으로 변경하시오

답)

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

결과)
안된다.

문제 40) 사원 테이블을 전부 삭제하시오

답)

```
hive> delete from emp;
```

결과)
안된다.

Hive에서 CTAS 문법

문제 41) emp 테이블을 CTAS로 emp_backup으로 생성하시오

답)

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

hive> select * from emp_backup;
```

문제 42) emp 테이블의 데이터를 emp_backup에 insert 하시오

답)

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

hive> select * from emp_backup;
```

```
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
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 43) emp_backup의 내용을 emp 테이블의 내용으로 overwrite하시오

답)

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

hive> select * from emp_backup;
```

```
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
```

---

05 스쿱으로 오라클과 연동

2018년 7월 3일 화요일 오전 9:58

Table of Contents

- 스쿱의 정의
- 스쿱 설치
- 스쿱으로 Oracle에서 Hive로 import 하는 법
- 오라클에서 Hive로 데이터 로드하는 쉘 스크립트 생성하기
- Hive에서 Oracle로 데이터를 export 하는 법

스쿱의 정의

```
오라클과 hive 와의 데이터 연동 또는 오라클과 hdfs 와의 데이터 연동을 위한 툴

오라클의 emp 테이블을 hive로 바로 로드할 수 있다
```

스쿱 설치

1. 스쿱 설치 파일을 올린다

```
ojdbc6.jar
sqoop-1.4.6.bin__hadoop-1.0.0.tar.gz
```

2. 스쿱 설치 파일의 압축을 푼다

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

3. ojdbc6.jar 를 sqoop 라이브러리로 이동한다

```
[orcl:~]$ ls -l ojdbc6.jar
[orcl:~]$ cp /home/oracle/ojdbc6.jar /home/oracle/sqoop/lib
[orcl:~]$ cd ./sqoop/lib
[orcl:lib]$ ls -l ojdbc6.jar
```

4. 스쿱 디렉토리의 bin 디렉토리로 가서 스쿱을 실행한다

```
[orcl:~]$ mv sqoop-1.4.6.bin__hadoop-1.0.0 sqoop
[orcl:~]$ cd ./sqoop/bin
[orcl:bin]$ ./sqoop                    # 스쿱 실행
```

```
Warning: /home/oracle/sqoop/bin/../../hbase does not exist! HBase imports will fa
Please set $HBASE_HOME to the root of your HBase installation.
Warning: /home/oracle/sqoop/bin/../../hcatalog does not exist! HCatalog jobs will
Please set $HCAT_HOME to the root of your HCatalog installation.
Warning: /home/oracle/sqoop/bin/../../accumulo does not exist! Accumulo imports w
l.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
Warning: /home/oracle/sqoop/bin/../../zookeeper does not exist! Accumulo imports
il.
Please set $ZOOKEEPER_HOME to the root of your Zookeeper installation.
Try 'sqoop help' for usage.
```

위 처럼 경고 4개가 나오면 정상적으로 실행된 것이다

4. .bash_profile 에 sqoop 홈디렉토리를 지정한다.

```
$ vi .bash_profile

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

$ . .bash_profile
```

하이브에서 dept 테이블 제거

```
hive> drop  table  dept;
hive> exit;
```

5. 오라클의 dept 테이블을 hive로 로드 (옮기는 작업)

```
[orcl:~]$ 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

[orcl:~]$ hive
```

```
hive> select * from dept;
```

```
OK
10.0    ACCOUNTING      NEW YORK
20.0    RESEARCH        DALLAS
30.0    SALES   CHICAGO
40.0    OPERATIONS      BOSTON
Time taken: 5.036 seconds, Fetched: 4 row(s)
```

스쿱으로 Oracle에서 Hive로 import 하는 법

문제 75) hr 계정의 employees 테이블을 hive로 로드하시오

답)

```
[orcl:~]$ pwd
/home/oracle
[orcl:~]$ sqlplus hr/hr

SQL> select * from employees;

[orcl:~]$ sh table_import.sh hr hr EMPLOYEES

[orcl:~]$ hive
hive> select count(*) from employees;
```

```
Total MapReduce CPU Time Spent: 2 seconds 20 msec
OK
107
Time taken: 19.474 seconds, Fetched: 1 row(s)
```

오라클에서 Hive로 데이터 로드하는 쉘 스크립트 생성하기

문제 76) 오라클에서 Hive로 데이터를 로드하는 쉘을 생성하시오

보기)

```
$ sh hadoop_manage.sh

유저명 입력: scott
패스워드 입력: tiger
테이블명 입력: salgrade
```

답)

```
$ vi hadoop_manage.sh

============================================================
echo -n "유저명 입력: "
```

```
read username
echo -n "패스워드 입력: "
read password
echo -n "테이블명 입력: "
read table

sh table_import.sh $username $password $table
================================================================

$ sh hadoop_manage.sh
유저명 입력: scott
패스워드 입력: tiger
테이블명 입력: salgrade

[orcl:~]$ hive
hive> select * from salgrade;
```

```
OK
1.0     700.0   1200.0
2.0     1201.0  1400.0
3.0     1401.0  2000.0
4.0     2001.0  3000.0
5.0     3001.0  9999.0
Time taken: 0.071 seconds, Fetched: 5 row(s)
```

문제 77) 위에서 만든 hadoop_manage.sh 를 자동화 스크립트 13번에 추가하시오

답)

```
[orcl:~]$ vi m2.sh

================================================================
13. 스쿱으로 오라클에서 Hive로 데이터를 로드하려면 13번을
13)
sh /home/oracle/hadoop_manage.sh ;;
================================================================

위 두줄 적절한 위치에 추가
```

Hive에서 Oracle로 데이터를 export 하는 법

문제 78) Hive에서 Oracle로 데이터를 로드하는 스크립트를 생성하시오

답)

우선 Hive에 dept 테이블 있는 확인

```
hive> select * from dept;
```

오라클(scott 계정)에서 dept 테이블 제거

```
SQL> drop table dept cascade constraints;
SQL> exit;
```

Table dropped.

하이브에서 dept 테이블 생성

```
hive> show create table dept;
```

```
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  'hdfs://localhost:9000/user/hive/warehouse/dept'
TBLPROPERTIES (
  'numPartitions'='0'
```

위에 생성한 거 있나 확인

```
[orcl:~]$ hadoop fs -lsr /user
```

```
drwxr-xr-x   - oracle supergroup          0 2018-07-03 16:55 /user/hive
drwxr-xr-x   - oracle supergroup          0 2018-07-05 16:34 /user/hive/warehouse
drwxr-xr-x   - oracle supergroup          0 2018-07-05 14:58 /user/hive/warehouse/dept
-rw-r--r--   3 oracle supergroup          0 2018-07-05 14:58 /user/hive/warehouse/dept/
```

오라클에서 비어있는 테이블 먼저 만든다

```
SQL> create  table  dept
    ( deptno  number(2),
      dname   varchar2(14),
      loc     varchar2(13) );

SQL> exit;
```

Table created.

Hive 에서 Oracle 로 데이터 export하는 쉘 스크립트 작성

```
$ vi  table_export.sh

================================================================
#!/bin/bash

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

sqoop export --username $1 ₩
--password $2 ₩
--connect jdbc:oracle:thin:@localhost:1521:orcl ₩
--table $oracle_table ₩
--export-dir /user/hive/warehouse/$3 ₩
--input-fields-terminated-by '₩001' ₩
-m 1
```

```
================================================================


$ sh  table_export.sh scott tiger dept
```

Oracle 에서 확인해본다

```
[orcl:~]$ sqlplus scott/tiger
SQL> select * from dept;
```

```
    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON
```

문제 79) 자동화 스크립트 14번에 Hive -> Oracle 로 데이터 export 하는 스크립트를 추가하시오

답)

```
[orcl:~]$ vi hadoop_manage.sh


================================================================
echo -n "Oracle에서 Hive로 옮길거면 import를, Hive에서 Oracle로 옮길거면 export 입력: "
read transport
echo -n "유저명 입력: "
read username
echo -n "패스워드 입력: "
read password
echo -n "테이블명 입력: "
read table

sh table_$transport.sh $username $password $table
================================================================
```

```
[orcl:~]$ vi m2.sh


================================================================
      12. 타조를 중단 시키려면 12번을
      13. 스쿱으로 데이터를 import/export 하려면 13번을
      "
cho "   "
cho -n "번호를 입력하세요: "

       sh /home/oracle/tajo/tajo/bin/stop-tajo.sh ;;
       13)
       sh /home/oracle/hadoop_manage.sh ;;
sac
```

```
==========================================================
```

---

06 타조 설치 및 사용방법 총정리

2018년 7월 3일 화요일 오전 9:58

Table of Contents

- 타조(Tajo)의 정의
- 타조 설치
- 타조에서 emp 테이블 생성 및 조회
- 타조에서 JOIN

타조(Tajo)의 정의

최현식 박사는 타조를 하둡 기반의 분산데이터웨어하우스 시스템이라고 설명했다. SQL을 사용해 빅데이터를 분석할 때 맵리듀스를 사용하지 않고 HDFS 파일을 바로 읽어내는 기술이다.

그는 "아파치 타조는 하둡의 하이브를 대체하는 기술을 만들자는 목표로 시작됐고, 효율성과 low-latency (낮은 반응 시간) 시스템을 향해 발전하고 있다"라고 말했다.

그는 "맵리듀스 프로그램 대신 자체 설계한 처리 엔진을 통해 하이브보다 훨씬 빠른 처리 시간을 보여준다"라며 "비공유클러스터와도 호환되고, 네트워크 보틀넥이 없는 설계구조를 갖는다"라고 설명했다.

타조 개발진은 비공유 구조 상에서 발생하는 네트워크 보틀넥을 보완하기 위해 셔플 횟수를 줄일 수 있는 실행모델을 고안했다. DB에 널리 사용되는 코스트 기반 쿼리 최적화, 컬럼 익스큐션 엔진 등의 각종 기법을 적용하면서, 레거시 시스템(낡은 컴퓨터 시스템, 소프트웨어 등을 말한다)까지 연동할 수 있는 방안도 고민이었다.

타조는 무엇보다 온라인 상에서 SQL 입력 데이터 처리를 완료하기 전에도 결과값을 볼 수 있다는 특징을 갖고 있다.

그는 "큰 그림 위주로 결과값을 확인해야 하는 필요가 있는데, 타조는 중간 결과값을 확인하고 정확하다고 판단되면 중간에 중단하는 것도 가능하다"라고 설명했다.

특히 타조는 하나의 SQL 쿼리처리를 위해 필요한 잡의 오버헤드를 줄이는데 역점을 둔다. 그 결과 비용을 줄이면서, 아무리 큰 쿼리라도 하나의 잡으로 처리해 성능을 높인다.

그에 따르면, 작년 8월 TPC-H 테스트 당시 타조는 하이브 보다 모든 쿼리에서 앞선 성능을 보였다.

그는 타조와 임팔라를 비교하기도 했다. 드릴은 맵R을 중심으로 개발되고 있던 유사기술이다.

그는 "타조는 노드의 메모리보다 큰 데이터라고 해도 처리할 수 있는 성능을 보이지만, 임팔라는 작은 데이터만 효과적으로 처리하는 한계를 갖고 있다"라며 "또한 임팔라가 소스코드만 공개될 뿐 외부 개발자가 참여할 수 있는 부분이 적다는 것도 타조와 대비되는 점"이라고 말했다.

타조는 2010년 고려대학교 데이터베이스랩에서 시작됐다. 올해 3월 7일로 아파치재단 인큐페이터로 선정되면서 그 세를 불렸다. 그루터 인력이 적극 참여해 타조 고도화에 기여하고 있으며, 개발 속도에 힘을 붙이고 있다.

아키텍처는 마스터 워커 모델을 따른다. 마스터 한 대와 다수의 워커로 구성되고, 하둡의 얀(YARN)을 리소스 플랫폼으로 사용해 맵리듀스나 하이브, 피그 등의 하둡 클러스터도 공유할 수 있다. 각 워커는 로컬 쿼리엔진이 HDFS나 로컬 파일시스템, 다른 데이터 소스 테이블을 처리할 수 있게 설계됐다.

그는 "막 인큐베이팅된 만큼 빠른 시일 내 아파치 이름으로 첫 번째 버전을 내놓을 것"이라며 "이후 API나 기능을 논의과정을 거쳐 표준화하고, SQL 풀스펙 지원, 코스트 기반 최적화 개선, 네이티브 컬럼 익스큐션 엔진 등을 개선해나갈 계획이다"라고 강조했다.

타조 설치

1. 타조 설치 파일을 /home/oracle 밑에 가져다 둡니다.

2. 압축 해제 합니다.

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

3. 작업을 쉽게 하기 위해서 리눅스 심볼릭 링크 이용

```
ln -s tajo-0.11.1-desktop-r1 tajo
```

- 심볼릭 링크 지우는 법
rm -rf [심볼릭 링크명]

4. tajo 디렉토리로 들어갑니다.

```
[orcl:~]$ cd tajo
[orcl:tajo]$ cd tajo
[orcl:tajo]$ pwd
/home/oracle/tajo/tajo
```

압축해제 후 Tajo 셋팅 부분
자바 path는 자동으로 잡아줍니다.

```
[orcl:tajo]$ sh bin/configure.sh
Enter JAVA_HOME [default: /u01/app/java/jdk1.7.0_60]
```

추가적으로 옵션을 주기위한 부분이기 때문에 N으로 설정합니다

```
Would you like advanced configure? [y/N] N
```

5. 타조 실행 명령어

```
[orcl:tajo]$ sh bin/startup.sh
```

```
starting master, logging to /home/oracle/tajo/tajo/bin/../logs/tajo-oracle-master-edydr1p0.us.oracle.com.out
Tajo master started
starting worker, logging to /home/oracle/tajo/tajo/bin/../logs/tajo-oracle-worker-edydr1p0.us.oracle.com.out
Tajo worker started
Tajo master web UI: http://edydr1p0.us.oracle.com:xxxxx
Tajo Client Service: edydr1p0.us.oracle.com:xxxxxx
```

#타조 실행 중단 명령어

```
[orcl:tajo]$ pwd
/home/oracle/tajo/tajo
[orcl:tajo]$ sh bin/stop-tajo.sh
```

6. 타조 쉘 작동 확인

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

TSQL은 HDFS와 같은 실행기능을 제공하며, ₩dfs 옵션으로 HDFS 명령어를 실행할 수 있습니다.

타조 실행 화면
welcome to

```
   ____ __  ____ __
  /_  _/ _ |/_  _/   /
   / // /_| |_/ // / /
  /_//_/ /_/___/ ₩__/  0.11.1-p1

Try ₩? for help.
```

7. orcl 용 데이터베이스를 생성한 후 접속을 변경합니다.
DB 생성 명령어 이름은 orcl 로 생성

```
default> create database orcl;
default> ₩c orcl;
```

8. emp table 생성 방법

```
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';
```

9. emp 테이블의 메타 데이터 조회하는 방법
(테이블 및 함수의 메타 데이터는 ₩d 옵션으로 조회할 수 있습니다)

```
emp> ₩d emp;
```

```
table name: orcl.emp
table uri: file:///home/oracle/emp2.csv
store type: TEXT
number of rows: unknown
volume: 644 B
Options:
        'text.delimiter'=','

schema:
empno   INT4
ename   TEXT
job     TEXT
mgrno   INT4
hiredate        TEXT
sal     INT4
comm    INT4
deptno  INT4
```

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

```
empno,  ename,  job,  mgrno,  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
(14 rows, 0.684 sec, 0 B selected)
```

이상 타조 비행 성공~~~~

타조에서 emp 테이블 생성 및 조회

문제 44) 월급이 3000 이상인 사원들의 이름과 월급과 직업을 출력하시오

답)

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

```
Progress: 100%, response time: 2.487 sec
ename,  sal,  job
-------------------------------
KING,  5000,  PRESIDENT
FORD,  3000,  ANALYST
SCOTT,  3000,  ANALYST
(3 rows, 2.487 sec, 120 B selected)
```

문제 45) 이름의 첫글자가 S로 시작하는 사원들의 이름과 월급을 출력하시오

답)

```
orcl> select ename, sal
from emp
where ename like 'S%';
```

```
Progress: 100%, response time: 0.069 sec
ename,  sal
-------------------------------
SMITH,  800
SCOTT,  3000
(2 rows, 0.069 sec, 50 B selected)
```

문제 46) 월급이 1000 에서 3000 사이인 사원들의 이름과 월급을 출력하시오

답)

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

```
Progress: 100%, response time: 0.068 sec
ename,  sal
-------------------------------
BLAKE,  2850
CLARK,  2450
JONES,  2975
MARTIN,  1250
ALLEN,  1600
TURNER,  1500
WARD,  1250
FORD,  3000
SCOTT,  3000
ADAMS,  1100
MILLER,  1300
(11 rows, 0.068 sec, 276 B selected)
```

문제 47) 커미션이 null인 사원들의 이름과 커미션을 출력하시오

답)

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

문제 48) 직업, 직업 별 토탈 월급을 출력하시오

답)

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

```
Progress: 100%, response time: 0.275 sec
job,  ?sum
------------------------------
MANAGER,  8275
CLERK,  4150
PRESIDENT,  5000
SALESMAN,  5600
ANALYST,  6000
(5 rows, 0.275 sec, 156 B selected)
```

문제 49) having 절이 지원되는지 확인하시오

직업, 직업 별 토탈 월급을 출력하는데 직업 별 토탈월급이 4000 이상인 것만 출력하시오

답)

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

문제 50) 부서번호, 부서번호 별 토탈 월급을 출력하는데 가로로 출력하시오

답)

```
orcl> 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;
```

```
Progress: 100%, response time: 0.375 sec
dept10,  dept20,  dept30
------------------------------
8750,  10875,  9400
(1 rows, 0.375 sec, 40 B selected)
```

문제 51) dept 테이블을 생성하고 데이터를 로드하시오

답)

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

orcl> select * from dept;
```

문제 52) 이름과 부서위치를 출력하시오

답)

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

답 2)

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

타조에서 JOIN

문제 53) 이름과 부서위치를 출력하는데 right outer join을 써서 수행하시오

답)

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

설명)
ANSI 조인 문법만 된다

문제 54) 타조에서 full outer join을 지원하는지 확인하시오

답)

```
orcl> select e.ename, d.loc
> from emp e full outer join dept d
> on e.deptno = d.deptno
```

```
Progress: 100%, response time: 0.148 sec
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
,  BOSTON
```

문제 55) 타조에서 rollup이 지원되는지 확인하시오

답)

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

결과)

```
orcl> select deptno, sum(sal) sumsal
> from emp
> group by rollup(deptno);
ERROR: unsupported feature: Rollup
```

답)

```
orcl> select deptno, sum(sal) sumsal
> from emp
> group by deptno
> union all
> select 1, sum(sal)
> from emp;
```

결과)

```
Progress: 100%, response time: 0.193 sec
deptno,  sumsal
------------------------------
1,  29025
20,  10875
30,  9400
10,  8750
(1 rows, 0.193 sec, 96 B selected)
```

문제 56) Hive 에서 지원하지 않았던 서브쿼리 지원되는지 확인해 보시오

답)

```
orcl> select ename, sal
>   from emp
>   where sal > ( select sal from emp where ename = 'JONES' ) ;
ERROR: internal error: ExprAnnotator cannot take NamedExpr
```

문제 57) 이름, 월급, 순위를 출력하는데 순위가 월급이 높은 순서에 대한 순위를 출력하시오

답)

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

문제 58) 위의 결과를 다시 출력하는데 순위가 1등인 사원만 출력하시오

답)

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

```
Progress: 100%, response time: 0.325 sec
ename,  sal,  rnk
------------------------------
KING,  5000,  1
(1 rows, 0.325 sec, 36 B selected)
```

문제 59) (마지막 문제) 타조 가장 최신 버전 설치 파일을 다운로드하시오

OK

---

07 Pig 설치 및 사용방법 총정리

2018년 7월 3일 화요일 오전 9:58

Table of Contents

- Pig의 정의
- Pig 설치
- Pig 에서 emp 테이블 생성 및 조회 (filter, order)
- Pig 에서 그룹함수(count, sum)

Pig의 정의

```
1. 야후에서 만든 NoSQL
(야후에서는 40% 이상의 JOB을 Pig로 처리)

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

3. 사용자 정의함수 (오라클의 프로시저와 같은 언어)를 지원한다.

4. 매뉴얼
https://www.tutorialspoint.com/apache_pig/apache_pig_grunt_shell.htm
```

Pig 설치

1. pig 설치파일을 /home/oracle 에 올린다

```
pig-0.12.0.tar.gz
```

2. pig 설치 파일의 압축을 푼다

```
$ tar xvf pig-0.12.0.tar.gz
$ ls -ld pig*
```

```
drwxr-xr-x 15 oracle oinstall     4096 Oct  8  2013 pig-0.12.0
-rw-r--r--  1 oracle oinstall 59941262 Jul  5 09:58 pig-0.12.0.tar.gz
```

3. pig 환경설정을 한다

```
[orcl:~]$ mv pig-0.12.0 pig
[orcl:~]$ ls -ld pig*
```

```
drwxr-xr-x 15 oracle oinstall     4096 Oct  8  2013 pig
-rw-r--r--  1 oracle oinstall 59941262 Jul  5 09:58 pig-0.12.0.tar.gz
```

```
[orcl:~]$ cd pig
[orcl:pig]$ cd conf
[orcl:conf]$ vi pig.properties
```

제일 아래 줄에

==========

fs.default.name=hfdf://localhost:9000

mapred.job.tracker=localhost:9001

==========

위 두 문장 추가

4. .bash_profile 에 PIG_HOME 디렉토리를 지정한다

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

==========

export PIG_HOME=/home/oracle/pig

export PATH=$PIG_HOME/bin:$PATH

==========

5. . .bash_profile을 실행한다

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

6. pig에 접속한다

```
[orcl:~]$ pig -x local
```

Pig 에서 emp 테이블 생성 및 조회 (filter, order)

문제 60) pig에서 emp 테이블을 생성하시오

답)

```
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;
```

이름, 월급을 뽑아라~

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

문제 61) 월급이 3000 이상인 사원들의 이름과 월급을 출력하시오

답)

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

```
(KING,5000)
(FORD,3000)
(SCOTT,3000)
```

문제 62) 직업이 SALESMAN인 사원들의 이름과 월급과 직업을 출력하시오

답)

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

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

설명)

문자열 쓸 때 작은 따옴표 쓰도록!

문제 63) 이름과 월급을 출력하는데 월급이 높은 사원부터 출력하시오

답)

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

```
(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 64) 부서번호가 30번인 사원들의 이름, 월급, 부서번호를 출력하는데 월급이 높은 사원부터 출력하시오

답)

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

```
(BLAKE,2850,30)
(ALLEN,1600,30)
(TURNER,1500,30)
(MARTIN,1250,30)
(WARD,1250,30)
(JAMES,950,30)
```

Pig 에서 그룹함수(count, sum)

문제 65) 직업과 직업 별 인원 수를 출력하시오

Oracle)

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

답)

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

```
(CLERK,4)
(ANALYST,2)
(MANAGER,3)
(SALESMAN,4)
(PRESIDENT,1)
```

문제 66) 직업을 출력하는데 중복 제거해서 출력하시오

답)

```
data = foreach emp generate job;
data2 = group data by job;
data3 = foreach data2 generate group;
dump data3;
```

```
(CLERK)
(ANALYST)
(MANAGER)
(SALESMAN)
(PRESIDENT)
```

문제 67) 직업과 직업 별 토탈 월급을 출력하시오

답)

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

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

문제 68) 부서번호, 부서번호 별 토탈월급을 출력하는데 높은 것부터 출력되게 하시오

답)

```
data = foreach emp generate deptno, sal;
data2 = group data by deptno;
data3 = foreach data2 generate group, SUM(data.sal) as sumsal;
data4 = order data3 by sumsal desc;
dump data4;
```

```
(20,10875)
(30,9400)
(10,8750)
```

문제 69) 직업과 직업 별 토탈월급을 출력하는데 직업이 SALESMAN 인 사원은 제외하고 토탈 월급이 4000 이상인 것만 출력하고 높은 것부터 출력하시오

답)

```
data = foreach emp generate job, sal;
data2 = filter data by job != 'SALESMAN' ;
data3 = group data2 by job;
data4 = foreach data3 generate group, SUM(data2.sal) as sumsal;
data5 = order data4 by sumsal desc;
data6 = filter data5 by sumsal >= 5000;
```

```
dump data6;
```

```
(MANAGER,8275)
(ANALYST,6000)
(PRESIDENT,5000)
```

문제 70) dept2.csv 를 이용해서 dept 테이블을 pig에서 생성하시오

답)

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

DUMP dept;
```

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

문제 71) 이름과 부서위치를 출력하시오

답)

```
empdept = JOIN emp by deptno RIGHT, dept BY deptno;
data = foreach empdept generate ename, loc;
dump data;
```

```
(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)
(,BOSTON)
```

문제 72) 부서위치, 부서위치 별 토탈 월급을 출력하는데 높은 것부터 출력하시오

답)

```
empdept = JOIN emp by deptno RIGHT, dept BY deptno;
data = foreach empdept generate loc, sal;
```

```
data2 = group data by loc;
data3 = foreach data2 generate group, SUM(data.sal) as sumsal ;
data4 = order data3 by sumsal desc;
dump data4;
```

```
(DALLAS,10875)
(CHICAGO,9400)
(NEW YORK,8750)
(BOSTON,)
```

문제 73) dept 테이블을 hdfs 에 올려서 쿼리하시오

답)

```
[orcl:~]$ hadoop fs -put dept2.csv dept2.csv
[orcl:~]$ hadoop fs -ls dept2.csv
```

```
[orcl:~]$ hadoop fs -ls dept2.csv
Found 1 items
-rw-r--r--   3 oracle supergroup         84 2018-07-05 14:05 /user/oracle/dept2.csv
```

```
[orcl:~]$ pig -x local
dept = LOAD '/home/oracle/dept2.csv'  USING PigStorage(',') as (deptno:int, dname:chararray,
loc:chararray);
DUMP dept;
data = foreach dept generate deptno, loc;
dump data
```

```
(10,NEW YORK)
(20,DALLAS)
(30,CHICAGO)
(40,BOSTON)
```

문제 74) emp2.csv 도 하둡 파일 시스템에 올려서 pig 에서 하둡 파일 시스템에 있는 emp2.csv 로 emp 테이블을 생성하시오

답)

```
[orcl:~]$ hadoop fs -put emp2.csv emp2.csv
[orcl:~]$ hadoop fs -ls emp2.csv
[orcl:~]$ pig -x local
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);
DUMP emp;
emp = foreach emp generate empno, ename, sal;
dump emp
```

---

08 MongoDB in Ubuntu 설치 및 사용방법 총정리

2018년 7월 3일 화요일 오전 9:58

Table of Contents

- MongoDB 설치
- MongoDB 에서 emp 테이블 생성 및 쿼리 실행
- 비교 연산자
- 논리 연산자
- 산술 연산자 & 문자 연산자
- MongoDB 에서 GROUP
- MongoDB 에서 emp 테이블을 drop하고 다시 생성하는 방법
- MongoDB 에서의 JOIN
- MongoDB 에서 DML 문 - UPDATE
- MongoDB 에서 DML 문 - DELETE
- MongoDB 에서 DML 문 - INSERT
- MongoDB 에서 DDL 문 - DROP
- MongoDB 에서 DDL 문 - ALTER (컬럼명 변경, 컬럼 추가, 컬럼 삭제)
- MongoDB 에서 index 생성하는 방법

MongoDB 설치

https://velopert.com/436 에 들어가서 아래로 내려가면 "리눅스" 라고 있다.

리눅스

이 포스트에서는 Ubuntu 에서 MongoDB 설치하는 과정을 살펴보도록하겠습니다.
다른 리눅스 디스트로를 사용하시는분들은 https://docs.mongodb.org/manual/installation/ 을 참조해주세요.

주의: Cloud9 를 사용하신다면 1번과 2번은 생략하셔도 됩니다.

1. MongoDB Pulbic GPG Key 등록

1. MongoDB Public GPG key 등록

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

2. MongoDB를 위한 list file 생성 (자신의 Ubuntu 버전에 맞게 입력할 것)

```
# Ubuntu 14.04
$ echo "deb http://repo.mongodb.org/apt/ubuntu trusty/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. 서버 실행

```
$ sudo service mongod start
# 서버가 제대로 실행됐는지 확인


$ cat /var/log/mongodb/mongod.log
# [initandlisten] waiting for connections on port <port>
```

기본 포트는 27017 이며, 방화벽을 사용하시는분들은 해당 포트를 Open 해주세요.
포트는 /etc/mongod.conf 에서 변경 가능합니다.

5. MongoDB 서버 접속

```
$ mongo
```

```
MongoDB shell version: 3.2.20
connecting to: test
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
        http://docs.mongodb.org/
Questions? Try the support group
        http://groups.google.com/group/mongodb-user
```

MongoDB 에서 emp 테이블 생성 및 쿼리 실행

문제80. mongodb 에 emp 테이블을 생성하시오

답)

```
# 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,dpetno:20})
db.emp.save({empno:7654,ename:"MARTIN",job:"SALESMAN",mgr:7698,hiredate:"1981-09-28",sal:1250,comm:1400,dpetno:30})
db.emp.save({empno:7698,ename:"BLAKE",job:"MANAGER",mgr:7839,hiredate:"1981-05-01",sal:285
```

```
0,comm:0,deptno:30})
db.emp.save({empno:7782,ename:"CLARK",job:"MANAGER",mgr:7839,hiredate:"1981-06-09",sal:24
50,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,co
mm:0,deptno:10})
db.emp.save({empno:7844,ename:"TURNER",job:"SALESMAN",mgr:7698,hiredate:"1981-09-08",sal:1
500,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,co
mm: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})
```

문제 81) 아래의 select 문을 MongoDB 로 구현하시오

답)

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

```
> db.emp.aggregate([{$group:{_id:null, count:{$sum:1}}}])
{ "_id" : null, "count" : 14 }
```

문제 82) 부서번호가 10번 인 사원의 사원번호, 이름, 월급을 조회하시오

답)

```
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 83) 월급이 3000인 사원의 이름, 월급, 직업을 출력하시오

답)

```
db.emp.find({sal:{$all:[3000]}},
```

```
            {_id:0, ename:1,sal:1,job:1})
```

```
{ "ename" : "SCOTT", "job" : "ANALYST", "sal" : 3000 }
```

비교 연산자

문제 84) 월급이 2000 이상인 사원의 이름, 월급, 직업을 출력하시오

답)

```
db.emp.find({sal:{$gte: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 }
```

설명)

비교(Comparison) 연산자

operator	설명
$eq	(equals) 주어진 값과 일치하는 값
$gt	(greater than) 주어진 값보다 큰 값
$gte	(greather than or equals) 주어진 값보다 크거나 같은 값
$lt	(less than) 주어진 값보다 작은 값
$lte	(less than or equals) 주어진 값보다 작거나 같은 값
$ne	(not equal) 주어진 값과 일치하지 않는 값
$in	주어진 배열 안에 속하는 값
$nin	주어빈 배열 안에 속하지 않는 값

SQL	MongoDB
select * from emp where sal > 2000;	db.emp.find({sal:{$gt:2000}}, {_id:0})

문제 85) 직업이 SALESMAN이 아닌 사원들의 이름과 직업을 출력하시오

답)

```
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 86) 월급이 3000 이하 인 사원들의 이름과 월급을 출력하시오

SQL)

```
select ename, sal
  from emp
  where sal <= 3000 ;
```

MongoDB)

```
db.emp.find({sal:{$lte:3000}},
            {_id:0, ename:1, sal:1})
```

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

문제 87) 이름, 월급을 출력하는데 월급이 높은 사원부터 출력하시오

SQL)

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

MongoDB)

```
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) 직업이 SALESMAN인 사원들의 이름, 월급, 직업을 출력하는데 월급이 낮은 사원부터 출력하시오

SQL)

```
select ename, sal, job
  from emp
  where job = "SALESMAN"
  order by sal asc ;
```

MongoDB)

```
db.emp.find( {job:"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 }
```

설명)

db.[테이블명].find({조건}, {뽑을 컬럼}).sort({조건})

sort	
1	오름차순
-1	내림차순

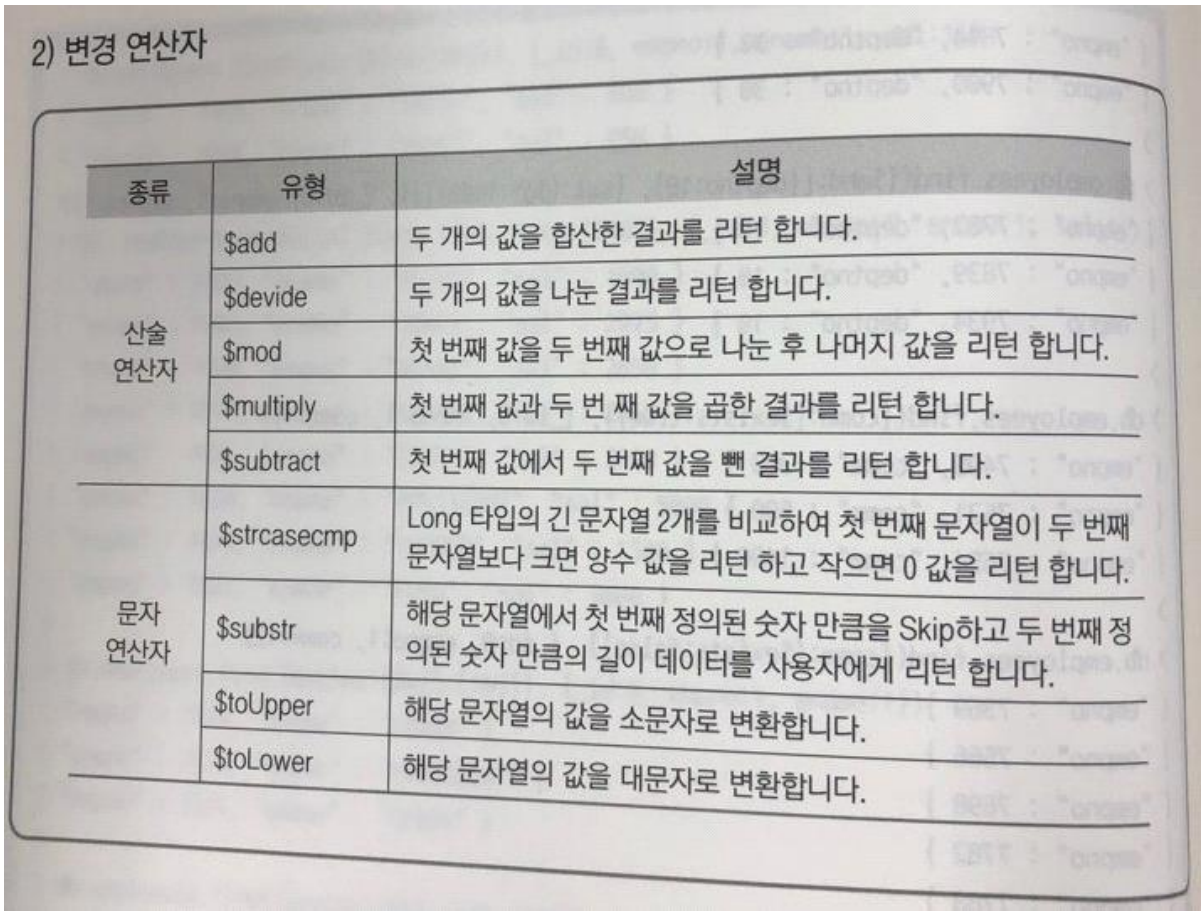
논리 연산자

논리 연산자

이 부분은 프로그래머라면 따로 설명이 필요 없을 듯 합니다.

operator	설명
$or	주어진 조건중 하나라도 true 일 때 true
$and	주어진 모든 조건이 true 일 때 true
$not	주어진 조건이 false 일 때 true
$nor	주어진 모든 조건이 false 일때 true

산술 연산자 & 문자 연산자



2) 변경 연산자

종류	유형	설명
산술 연산자	$add	두 개의 값을 합산한 결과를 리턴 합니다.
	$devide	두 개의 값을 나눈 결과를 리턴 합니다.
	$mod	첫 번째 값을 두 번째 값으로 나눈 후 나머지 값을 리턴 합니다.
	$multiply	첫 번째 값과 두 번 째 값을 곱한 결과를 리턴 합니다.
	$subtract	첫 번째 값에서 두 번째 값을 뺀 결과를 리턴 합니다.
문자 연산자	$strcasecmp	Long 타입의 긴 문자열 2개를 비교하여 첫 번째 문자열이 두 번째 문자열보다 크면 양수 값을 리턴 하고 작으면 0 값을 리턴 합니다.
	$substr	해당 문자열에서 첫 번째 정의된 숫자 만큼을 Skip하고 두 번째 정의된 숫자 만큼의 길이 데이터를 사용자에게 리턴 합니다.
	$toUpper	해당 문자열의 값을 소문자로 변환합니다.
	$toLower	해당 문자열의 값을 대문자로 변환합니다.

문제 89) 월급이 1000 에서 3000 사이 인 사원들의 이름, 월급을 출력하시오

SQL)

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

MongoDB)

```
db.emp.find( {$and: [{sal:{$gte:1000}}, {sal:{$lte:3000}}]},
```

```
  {_id:0,ename:1, sal:1} )
```

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

다른 코드)

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

문제 90) 사원번호가 7788번 또는 7902 인 사원들의 사원번호, 이름, 월급을 출력하시오

SQL)

```
select ename, sal
  from emp
  where empno = 7788 or empno = 7902
```

MongoDB)

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

```
{ "empno" : 7788, "ename" : "SCOTT", "sal" : 3000 }
{ "empno" : 7902, "ename" : "FORD", "sal" : 3500 }
```

문제 91) 부서번호를 출력하는데 중복제거해서 출력하시오

SQL)

```
select distinct deptno
  from emp;
```

MongoDB)

```
db.emp.distinct( "deptno" )
```

```
[ 20, 30, 10 ]
```

문제 92) 사원 테이블의 전체 건수를 출력하시오

SQL)

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

MongoDB)

```
db.emp.count()
```

14

문제 93) 직업이 SALESMAN 인 사원들의 인원 수를 출력하세요

SQL)

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

MongoDB)

```
db.emp.find( {job:"SALESMAN"} ).count()
```

4

다른 방법)

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

문제 94) 월급이 3000 이상 인 사원들의 인원 수를 출력하시오

SQL)

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

MongoDB)

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

3

다른 방법)

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

문제 95) 사원 테이블 전체에서 최대 월급을 출력하시오

MongoDB)

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

```
{ "sal" : 5000 }
```

문제 96) 직업이 SALESMAN 인 사원들의 최대 월급을 출력하시오

MongoDB)

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

```
{ "sal" : 1600 }
```

문제 97) 사원 테이블의 토탈 월급을 출력하시오

MongoDB)

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

```
{ "_id" : null, "total" : 30525 }
```

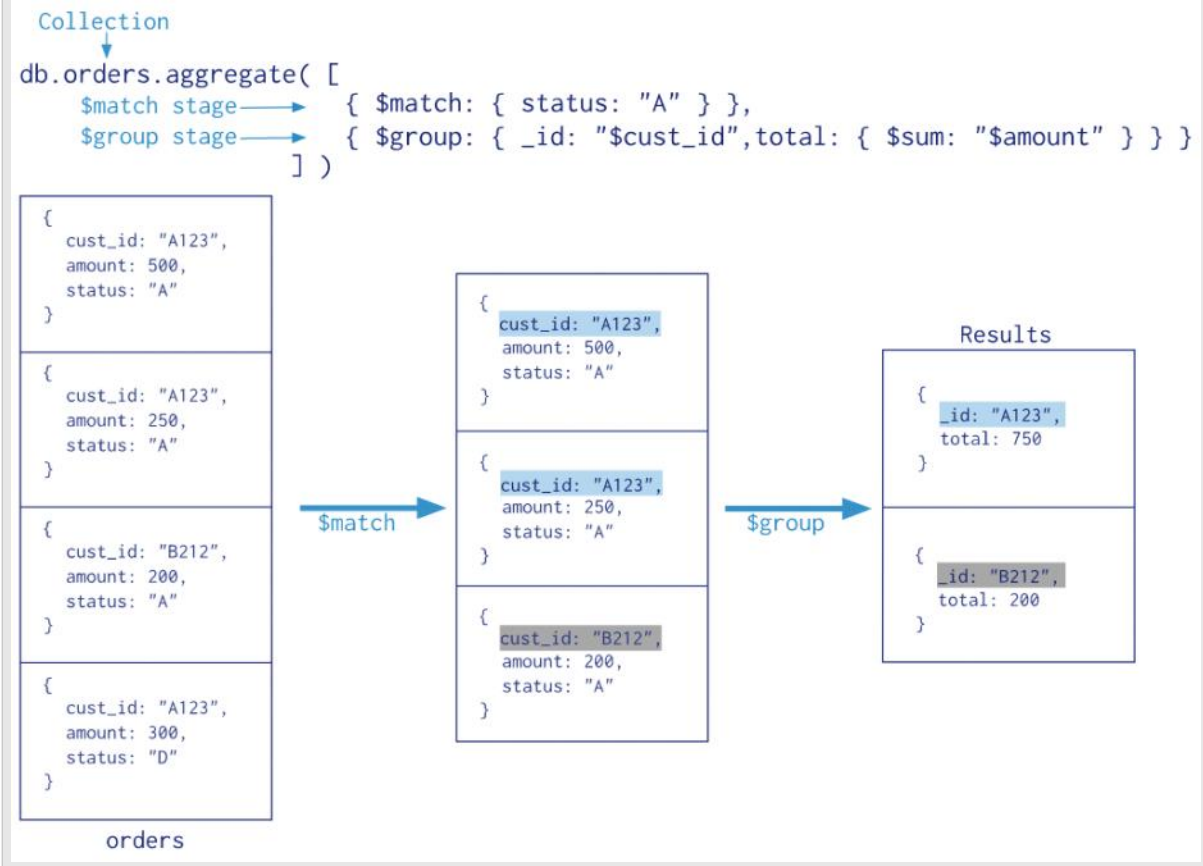
문제 98) 직업이 SALESMAN 인 사원의 토탈 월급을 출력하시오

MongoDB)

```
db.emp.aggregate([{$match: {job:"SALESMAN"}}, {$group:{_id:null, total:{$sum:"$sal"}}}])
```

```
{ "_id" : null, "total" : 5600 }
```

설명)



문제 99) 부서번호가 30번 인 사원들의 최대 월급을 출력하시오

MongoDB)

```
db.emp.find( {deptno:30}, {_id:0, sal:1} ).sort( {sal:-1} ).limit(1)
```

```
{ "sal" : 2850 }
```

다른 방법)

```
db.emp.aggregate([{$match: {job:"SALESMAN"}}, {$group:{_id:null, maxsal:{$max:"$sal"}}}])
```

MongoDB 에서 GROUP

문제 100) 부서번호, 부서번호 별 토탈 월급을 출력하시오

SQL)

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

MongoDB)

```
db.emp.aggregate( [{$group:{_id:"$deptno", total:{$sum:"$sal"}}}] )
```

```
{ "_id" : 10, "total" : 7450 }
{ "_id" : 30, "total" : 7800 }
{ "_id" : 20, "total" : 15275 }
```

MongoDB 에서 emp 테이블을 drop하고 다시 생성하는 방법

위에서 emp 테이블을 생성할 때 JONES와 MARTIN의 deptno가 dpetno로 잘못 들어가있었다.

```
# emp 테이블 삭제

> db.emp.drop()
true

# 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})
```

문제 101) 직업을 뽑고 직업 별 최저 월급을 출력하시오

MongoDB)

```
db.emp.aggregate( [{$group:{_id:"$job", minsal:{$min:"$sal"}}}] )
```

```
{ "_id" : "PRESIDENT", "minsal" : 5000 }
{ "_id" : "ANALYST", "minsal" : 3000 }
{ "_id" : "MANAGER", "minsal" : 2450 }
{ "_id" : "SALESMAN", "minsal" : 1250 }
{ "_id" : "CLERK", "minsal" : 950 }
```

문제 102) 직업, 직업 별 토탈 월급을 출력하는데 직업이 SALESMAN은 제외하고 출력하시오

SQL)

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

MongoDB)

```
db.emp.aggregate( [{$match: {job:{$ne:"SALESMAN"}}}, {$group:{_id:"$job",
sumsal:{$sum:"$sal"}}}] )
```

```
{ "_id" : "PRESIDENT", "sumsal" : 5000 }
{ "_id" : "ANALYST", "sumsal" : 6500 }
{ "_id" : "MANAGER", "sumsal" : 8275 }
{ "_id" : "CLERK", "sumsal" : 5150 }
```

문제 103) 직업, 직업 별 토탈 월급을 출력하는데 직업이 SALESMAN 인 것은 제외하고 직업 별 토탈 월급이 높은 것 부터 출력하시오

SQL)

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

MongoDB)

```
db.emp.aggregate( [{$match: {job:{$ne:"SALESMAN"}}},
{$group:{_id:"$job", sumsal:{$sum:"$sal"}}},
{$sort:{"sumsal":-1}}] )
```

```
{ "_id" : "MANAGER", "sumsal" : 8275 }
{ "_id" : "ANALYST", "sumsal" : 6500 }
{ "_id" : "CLERK", "sumsal" : 5150 }
{ "_id" : "PRESIDENT", "sumsal" : 5000 }
```

문제 104) 이름의 첫번째 글자가 A로 시작하는 사원들의 이름과 월급을 출력하시오

SQL)

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

MongoDB)

```
db.emp.find( {ename:/^A/}, {_id:0, ename:1, sal:1} )
```

```
{ "ename" : "ALLEN", "sal" : 1600 }
{ "ename" : "ADAMS", "sal" : 1100 }
```

문제 105) 이름의 끝 글자가 T로 끝나는 사원들의 이름, 월급을 출력하시오

SQL)

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

MongoDB)

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

```
{ "ename" : "SCOTT", "sal" : 3000 }
```

문제 106) 이름의 철자 M을 포함하는 사원들의 이름과 월급을 출력하시오

SQL)

```
select ename, sal
  from emp
  where ename like '%M%' ;
```

MongoDB)

```
db.emp.find( {ename:/M/}, {_id:0, ename:1, sal:1} )
```

```
{ "ename" : "SMITH", "sal" : 1800 }
{ "ename" : "MARTIN", "sal" : 1250 }
{ "ename" : "ADAMS", "sal" : 1100 }
{ "ename" : "JAMES", "sal" : 950 }
{ "ename" : "MILLER", "sal" : 1300 }
```

문제 107) 이름의 두번째 철자가 M 인 사원들의 이름과 월급을 출력하시오

SQL)

```
select ename, sal
  from emp
  where ename like '_M%' ;
```

MongoDB)

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

```
{ "ename" : "SMITH", "sal" : 1800 }
```

MongoDB 에서의 JOIN

문제 108) emp 테이블 생성 스크립트를 참고해서 dept 테이블을 생성하시오

답)

```
# dept 테이블 생성

db.dept.save({deptno:10,dname:"ACCOUNTING",loc:"NEWYORK"})
db.dept.save({deptno:20,dname:"RESEARCH",loc:"DALLAS"})
db.dept.save({deptno:30,dname:"SALES",loc:"CHICAGO"})
db.dept.save({deptno:40,dname:"OPERATIONS",loc:"BOSTON"})

db.dept.find({}, {_id:0})
```

```
{ "deptno" : 10, "dname" : "ACCOUNTING", "loc" : "NEWYORK" }
{ "deptno" : 20, "dname" : "RESEARCH", "loc" : "DALLAS" }
{ "deptno" : 30, "dname" : "SALES", "loc" : "CHICAGO" }
{ "deptno" : 40, "dname" : "OPERATIONS", "loc" : "BOSTON" }
```

문제 109) 이름과 부서위치를 출력하세요

SQL)

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

MongoDB)

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

문제 110) 이름, 월급, 부서명을 출력하시오

MongoDB)

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

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

문제 111) 위의 결과를 다시 출력하는데 월급이 3000 이상인 사원들만 출력하시오

MongoDB)

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

```
{$match:{sal:{$gte:3000}}}
 ] )
```

```
{ "ename" : "SCOTT", "sal" : 3000, "aa" : [ { "dname" : "RESEARCH" } ] }
{ "ename" : "KING", "sal" : 5000, "aa" : [ { "dname" : "ACCOUNTING" } ] }
{ "ename" : "FORD", "sal" : 3500, "aa" : [ { "dname" : "RESEARCH" } ] }
```

MongoDB 에서 DML 문 - UPDATE

문제 112) SCOTT의 월급을 5600으로 변경하시오

SQL)

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

MongoDB)

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


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

```
{ "ename" : "SMITH", "sal" : 1800 }
{ "ename" : "ALLEN", "sal" : 1600 }
{ "ename" : "WARD", "sal" : 1250 }
{ "ename" : "JONES", "sal" : 2975 }
{ "ename" : "MARTIN", "sal" : 1250 }
{ "ename" : "BLAKE", "sal" : 2850 }
{ "ename" : "CLARK", "sal" : 2450 }
{ "ename" : "SCOTT", "sal" : 5600 }
{ "ename" : "KING", "sal" : 5000 }
{ "ename" : "TURNER", "sal" : 1500 }
{ "ename" : "ADAMS", "sal" : 1100 }
{ "ename" : "JAMES", "sal" : 950 }
{ "ename" : "FORD", "sal" : 3500 }
{ "ename" : "MILLER", "sal" : 1300 }
```

문제 113) 월급이 3000 이상 인 사원들의 comm을 7000으로 수정하시오

SQL)

```
update emp
  set comm = 7000
  where sal >= 3000 ;
```

MongoDB)

```
db.emp.update( {sal:{$gte:3000}}, {$set:{comm:7000}}, {multi:true} )


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

```
{ "ename" : "SCOTT", "sal" : 5600, "comm" : 7000 }
{ "ename" : "KING", "sal" : 5000, "comm" : 7000 }
{ "ename" : "FORD", "sal" : 3500, "comm" : 7000 }
{ "ename" : "MARTIN", "sal" : 1250, "comm" : 1400 }
{ "ename" : "SMITH", "sal" : 1800, "comm" : 800 }
{ "ename" : "ALLEN", "sal" : 1600, "comm" : 800 }
{ "ename" : "WARD", "sal" : 1250, "comm" : 500 }
{ "ename" : "JONES", "sal" : 2975, "comm" : 0 }
{ "ename" : "BLAKE", "sal" : 2850, "comm" : 0 }
{ "ename" : "CLARK", "sal" : 2450, "comm" : 0 }
{ "ename" : "TURNER", "sal" : 1500, "comm" : 0 }
{ "ename" : "ADAMS", "sal" : 1100, "comm" : 0 }
{ "ename" : "JAMES", "sal" : 950, "comm" : 0 }
{ "ename" : "MILLER", "sal" : 1300, "comm" : 0 }
```

MongoDB 에서 DML 문 - DELETE

문제 114) 직업이 ANALYST 인 사원들을 삭제하시오

SQL)

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

MongoDB)

```
db.emp.remove( {job:"ANALYST"} )


db.emp.find( {}, {_id:0, ename:1, job:1} )
```

```
{ "ename" : "SMITH", "job" : "CLERK" }
{ "ename" : "ALLEN", "job" : "SALESMAN" }
{ "ename" : "WARD", "job" : "SALESMAN" }
{ "ename" : "JONES", "job" : "MANAGER" }
{ "ename" : "MARTIN", "job" : "SALESMAN" }
{ "ename" : "BLAKE", "job" : "MANAGER" }
{ "ename" : "CLARK", "job" : "MANAGER" }
{ "ename" : "KING", "job" : "PRESIDENT" }
{ "ename" : "TURNER", "job" : "SALESMAN" }
{ "ename" : "ADAMS", "job" : "CLERK" }
{ "ename" : "JAMES", "job" : "CLERK" }
{ "ename" : "MILLER", "job" : "CLERK" }
```

문제 115) emp 테이블 전체를 지우시오

SQL)

```
delete from emp ;
```

MongoDB)

```
db.emp.remove( {} )


db.emp.find()
```

MongoDB 에서 DML 문 - INSERT

문제 116) 아래의 insert를 MongoDB로 구현하시오

SQL)

```
insert into emp (empno, ename, sal)
  values(7788, 'SCOTT', 3000) ;
```

MongoDB)

```
db.emp.insert( {empno:7788, ename:"SCOTT", sal:3000} )


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

```
{ "empno" : 7788, "ename" : "SCOTT", "sal" : 3000 }
```

MongoDB 에서 DDL 문 - DROP

문제 117) 아래의 drop table 명령어를 몽고디비로 구현하시오

SQL)

```
drop table emp ;
```

MongoDB)

```
db.emp.drop()
```

MongoDB 에서 DDL 문 - ALTER (컬럼명 변경, 컬럼 추가, 컬럼 삭제)

문제 118) emp 테이블을 다시 생성하고 컬럼명 sal을 salary로 변경하시오

SQL)

```
alter table emp
  rename column sal to salary ;
```

MongoDB)

```
위의 노트 필기 참고하여 emp 테이블 생성
db.emp.update( {}, {$rename:{"sal":"salary"} }, {multi:true} )
db.emp.find( {}, {_id:0, empno:1, ename:1, salary:1} )
```

```
{ "empno" : 7499, "ename" : "SMITH", "salary" : 1800 }
{ "empno" : 7369, "ename" : "ALLEN", "salary" : 1600 }
{ "empno" : 7521, "ename" : "WARD", "salary" : 1250 }
{ "empno" : 7566, "ename" : "JONES", "salary" : 2975 }
{ "empno" : 7654, "ename" : "MARTIN", "salary" : 1250 }
{ "empno" : 7698, "ename" : "BLAKE", "salary" : 2850 }
{ "empno" : 7782, "ename" : "CLARK", "salary" : 2450 }
{ "empno" : 7788, "ename" : "SCOTT", "salary" : 3000 }
{ "empno" : 7839, "ename" : "KING", "salary" : 5000 }
{ "empno" : 7844, "ename" : "TURNER", "salary" : 1500 }
{ "empno" : 7876, "ename" : "ADAMS", "salary" : 1100 }
{ "empno" : 7900, "ename" : "JAMES", "salary" : 950 }
{ "empno" : 7902, "ename" : "FORD", "salary" : 3500 }
{ "empno" : 7934, "ename" : "MILLER", "salary" : 1300 }
```

문제 119) emp 테이블에 email 컬럼을 추가하시오

SQL)

```
alter table emp
  add email varchar2(50) ;
```

MongoDB)

```
db.emp.update( {}, {$set:{"email":' '}}, {upsert:false, multi:true} )


db.emp.find( {}, {_id:0, empno:1, ename:1, salary:1, email:1} )
```

설명)

```
upsert : insert 와 update 를 한번에 수행하는 게 upsert
```

```
{ "empno" : 7499, "ename" : "SMITH", "salary" : 1800, "email" : " " }
{ "empno" : 7369, "ename" : "ALLEN", "salary" : 1600, "email" : " " }
{ "empno" : 7521, "ename" : "WARD", "salary" : 1250, "email" : " " }
{ "empno" : 7566, "ename" : "JONES", "salary" : 2975, "email" : " " }
{ "empno" : 7654, "ename" : "MARTIN", "salary" : 1250, "email" : " " }
{ "empno" : 7698, "ename" : "BLAKE", "salary" : 2850, "email" : " " }
{ "empno" : 7782, "ename" : "CLARK", "salary" : 2450, "email" : " " }
{ "empno" : 7788, "ename" : "SCOTT", "salary" : 3000, "email" : " " }
{ "empno" : 7839, "ename" : "KING", "salary" : 5000, "email" : " " }
{ "empno" : 7844, "ename" : "TURNER", "salary" : 1500, "email" : " " }
{ "empno" : 7876, "ename" : "ADAMS", "salary" : 1100, "email" : " " }
{ "empno" : 7900, "ename" : "JAMES", "salary" : 950, "email" : " " }
{ "empno" : 7902, "ename" : "FORD", "salary" : 3500, "email" : " " }
{ "empno" : 7934, "ename" : "MILLER", "salary" : 1300, "email" : " " }
```

문제 120) email 컬럼을 삭제하시오

SQL)

```
alter table emp
  drop column email ;
```

MongoDB)

```
db.emp.update( {}, {$unset:{email:1}}, {multi:true} )


db.emp.find( {}, {_id:0, empno:1, ename:1, salary:1, email:1} )
```

```
{ "empno" : 7499, "ename" : "SMITH", "salary" : 1800 }
{ "empno" : 7369, "ename" : "ALLEN", "salary" : 1600 }
{ "empno" : 7521, "ename" : "WARD", "salary" : 1250 }
{ "empno" : 7566, "ename" : "JONES", "salary" : 2975 }
{ "empno" : 7654, "ename" : "MARTIN", "salary" : 1250 }
{ "empno" : 7698, "ename" : "BLAKE", "salary" : 2850 }
{ "empno" : 7782, "ename" : "CLARK", "salary" : 2450 }
{ "empno" : 7788, "ename" : "SCOTT", "salary" : 3000 }
{ "empno" : 7839, "ename" : "KING", "salary" : 5000 }
{ "empno" : 7844, "ename" : "TURNER", "salary" : 1500 }
{ "empno" : 7876, "ename" : "ADAMS", "salary" : 1100 }
{ "empno" : 7900, "ename" : "JAMES", "salary" : 950 }
{ "empno" : 7902, "ename" : "FORD", "salary" : 3500 }
{ "empno" : 7934, "ename" : "MILLER", "salary" : 1300 }
```

MongoDB 에서 index 생성하는 방법

index 란?

```
대용량 DB 환경에서 데이터의 검색 속도를 향상 시켜주는 DB Object
```

문제 121) emp 테이블에 월급을 검색할 때 속도를 높일 수 있는 인덱스를 생성하시오

SQL)

```
create index emp_sal
  on emp(sal) ;
```

MongoDB)

```
db.emp.ensureIndex( {salary:1} )


db.emp.find( {}, {_id:0, ename:1, salary:1} )
```

문제 122) 아래의 index를 MongoDB 에서 생성하시오

SQL)

```
create index emp_deptno_sal
  on emp( deptno, sal desc ) ;


select deptno, sal
  from emp
```

```
  where deptno > 0 ;
```

MongoDB)

```
db.emp.ensureIndex( {deptno:1, salary:-1} )

db.emp.find( {}, {_id:0, ename:1, deptno:1, salary:1} )
```

문제 123) 아래의 SQL 을 MongoDB 에서 검색하시오

SQL)

```
select ename, sal, deptno
  from emp
  where deptno = 20 and sal = 1600 ;
```

MongoDB)

```
db.emp.ensureIndex( {deptno:1, sal:-1} )

db.emp.find( {$and: [{deptno:20}, {salary:1600}]}, {_id:0, ename:1, salary:1, deptno:1} )
```

```
{ "ename" : "ALLEN", "deptno" : 20, "salary" : 1600 }
```

문제 124) 부서위치, 부서위치 별 토탈 월급을 출력하시오

위에서 바꿨던 salary 부터 sal 로 바꾸고 시작한다

```
db.emp.update( {}, {$rename:{"salary":"sal"} }, {multi:true} )

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

MongoDB)

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

```
{ "_id" : [ "NEWYORK" ], "sumsal" : 7450 }
{ "_id" : [ "CHICAGO" ], "sumsal" : 7800 }
{ "_id" : [ "DALLAS" ], "sumsal" : 15275 }
```

---

09 R과 하둡 연동하기

2018년 7월 3일 화요일 오전 9:58

1. 먼저 루트에서 R 을 실행한다.

```
[orcl:~]$ su -
Password: oracle
[root@edydr1p0 ~]# cd /home/oracle/R-3.2.3
[root@edydr1p0 R-3.2.3]# cd bin
[root@edydr1p0 bin]# ./R
```

2. 하둡을 이용할수 있도록 R 에서 환경설정

```
Sys.setenv (JAVA_HOME="/u01/app/java/jdk1.7.0_60")
Sys.setenv (HADOOP_CMD="/u01/app/hadoop/hadoop-1.2.1/bin/hadoop")
```

3. rhdfs 패키지를 다운로드 받는다.

https://github.com/RevolutionAnalytics/RHadoop/wiki/Downloads

rhdfs_1.0.8.tar.gz 다운로드 파일을 아래의 위치에 둔다.

```
[orcl:~]$ cd ./R-3.2.3/library/
[orcl:library]$ pwd
/home/oracle/R-3.2.3/library
```

4. R 에서 아래와 같이 설치한다.

```
install.packages("/home/oracle/R-3.2.3/library/rhdfs_1.0.8.tar.gz", repos=NULL, type="source")
```

5. 아래와 같이 library 하고 hdfs.init() 를 불러온다.

```
library(rhdfs)
library(rJava)
```

```
HADOOP_CMD=/u01/app/hadoop/hadoop-1.2.1/bin/hadoop
Be sure to run hdfs.init()
```

```
hdfs.init()
```

6. 하둡 파일 시스템에 emp.csv 를 올리고 하둡 R 에서 읽어온다.

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


hdfs.ls("/user/oracle/emp2.csv")
tmp <- hdfs.cat("/user/oracle/emp2.csv")
emp <- read.csv(textConnection(tmp),head=F)
```

```
colnames(emp) <- c("empno", "ename", "job", "mgr", "hiredate", "sal", "comm", "deptno")
```

7. 이름과 월급을 조회하시오 !

```
emp[emp$ename=='SCOTT',c("ename","sal")]
```

```
   empno  ename       job  mgr   hiredate  sal comm deptno
1   7839   KING PRESIDENT    0 1981-11-17 5000    0     10
2   7698  BLAKE   MANAGER 7839 1981-05-01 2850    0     30
3   7782  CLARK   MANAGER 7839 1981-05-09 2450    0     10
4   7566  JONES   MANAGER 7839 1981-04-01 2975    0     20
5   7654 MARTIN  SALESMAN 7698 1981-09-10 1250 1400     30
6   7499  ALLEN  SALESMAN 7698 1981-02-11 1600  300     30
7   7844 TURNER  SALESMAN 7698 1981-08-21 1500    0     30
8   7900  JAMES     CLERK 7698 1981-12-11  950    0     30
9   7521   WARD  SALESMAN 7698 1981-02-23 1250  500     30
10  7902   FORD   ANALYST 7566 1981-12-11 3000    0     20
11  7369  SMITH     CLERK 7902 1980-12-09  800    0     20
12  7788  SCOTT   ANALYST 7566 1982-12-22 3000    0     20
13  7876  ADAMS     CLERK 7788 1983-01-15 1100    0     20
14  7934 MILLER     CLERK 7782 1982-01-11 1300    0     10
```

---

10 자바를 이용해서 HDFS의 데이터를 SELECT 하기

2018년 7월 3일 화요일 오전 9:58

Table of Contents

- 하둡의 중요한 2가지 기능
- Java 를 이용한 HDFS 입출력 (P 82 ~ 84)
- 두 개의 파일을 하나로 합쳐서 하둡 파일 시스템에 올리는 실습
- WordCount 예제

하둡의 중요한 2가지 기능

```
1. HDFS                 ----> java
2. MapReduce            ----> java
        1. map 기능 (함수)
        2. Reduce 기능 (함수)
```

예)

원본 문서	map 함수	reduce 함수
감자는 가지과의 여러 해살이 풀이다	감자 1 가지 1	map 함수에서 뽑은 걸 합침
안데스 산맥 일대가 감자의 원산지이다	안데스 산맥 1 감자 1 원산지 1	ex) 감자 3, 가지 1, 안데스 산맥 1, 원산지 1, 기후 1
감자는 서늘한 기후를 좋아한다	감자 1 기후 1	

Java 를 이용한 HDFS 입출력 (P 82 ~ 84)

#1. hadoop 홈디렉토리에 자바 실행 파일인 jar 파일의 위치가 어디인지 설정하는 환경설정 부분

```
[oracle@edydr1p2 ~]$ cat >> .bash_profile << EOF
export CLASSPATH=.:$HADOOP_HOME/hadoop-ant-1.2.1.jar:$HADOOP_HOME/hadoop-client-1.2.1.jar:$HADOOP_HOME/hadoop-core-1.2.1.jar:$HADOOP_HOME/hadoop-examples-1.2.1.jar:$HADOOP_HOME/hadoop-minicluster-1.2.1.jar:$HADOOP_HOME/hadoop-test-1.2.1.jar:$HADOOP_HOME/hadoop-tools-1.2.1.jar:$HADOOP_HOME/lib/asm-3.2.jar:$HADOOP_HOME/lib/aspectjrt-1.6.11.jar:$HADOOP_HOME/lib/aspectjtools-1.6.11.jar:$HADOOP_HOME/lib/commons-beanutils-1.7.0.jar:$HADOOP_HOME/lib/commons-beanutils-core-1.8.0.jar:$HADOOP_HOME/lib/commons-cli-1.2.jar:$HADOOP_HOME/lib/commons-
```

```
codec-1.4.jar:$HADOOP_HOME/lib/commons-collections-3.2.1.jar:
$HADOOP_HOME/lib/commons-configuration-1.6.jar:$HADOOP_HOME/lib/commons-
daemon-1.0.1.jar:$HADOOP_HOME/lib/commons-digester-1.8.jar:
$HADOOP_HOME/lib/commons-el-1.0.jar:$HADOOP_HOME/lib/commons-httpclient-3.0.1.jar:
$HADOOP_HOME/lib/commons-io-2.1.jar:$HADOOP_HOME/lib/commons-lang-2.4.jar:
$HADOOP_HOME/lib/commons-logging-1.1.1.jar:$HADOOP_HOME/lib/commons-logging-
api-1.0.4.jar:$HADOOP_HOME/lib/commons-math-2.1.jar:$HADOOP_HOME/lib/commons-
net-3.1.jar:$HADOOP_HOME/lib/core-3.1.1.jar:$HADOOP_HOME/lib/hadoop-capacity-
scheduler-1.2.1.jar:$HADOOP_HOME/lib/hadoop-fairscheduler-1.2.1.jar:
$HADOOP_HOME/lib/hadoop-thriftfs-1.2.1.jar:$HADOOP_HOME/lib/hsqldb-1.8.0.10.jar:
$HADOOP_HOME/lib/jackson-core-asl-1.8.8.jar:$HADOOP_HOME/lib/jackson-mapper-
asl-1.8.8.jar:$HADOOP_HOME/lib/jasper-compiler-5.5.12.jar:$HADOOP_HOME/lib/jasper-
runtime-5.5.12.jar:$HADOOP_HOME/lib/jdeb-0.8.jar:$HADOOP_HOME/lib/jersey-core-1.8.jar:
$HADOOP_HOME/lib/jersey-json-1.8.jar:$HADOOP_HOME/lib/jersey-server-1.8.jar:
$HADOOP_HOME/lib/jets3t-0.6.1.jar:$HADOOP_HOME/lib/jetty-6.1.26.jar:
$HADOOP_HOME/lib/jetty-util-6.1.26.jar:$HADOOP_HOME/lib/jsch-0.1.42.jar:
$HADOOP_HOME/lib/junit-4.5.jar:$HADOOP_HOME/lib/kfs-0.2.2.jar:
$HADOOP_HOME/lib/log4j-1.2.15.jar:$HADOOP_HOME/lib/mockito-all-1.8.5.jar:
$HADOOP_HOME/lib/oro-2.0.8.jar:$HADOOP_HOME/lib/servlet-api-2.5-20081211.jar:
$HADOOP_HOME/lib/slf4j-api-1.4.3.jar:$HADOOP_HOME/lib/slf4j-log4j12-1.4.3.jar:
$HADOOP_HOME/lib/xmlenc-0.52.jar:$CLASSPATH
EOF

[oracle@edydr1p2 ~]$ source .bash_profile
```

#2. 하둡 홈디렉토리 밑 labs 라는 디렉토리를 만들고 거기에 SingleFileWriteRead.java 파일을 생성한다.
SingleFileWriteRead.java : 로컬에 있는 파일을 하둡 파일 시스템에 올리는 자바 파일

```
[oracle@edydr1p2 ~]$ cd $HADOOP_HOME
[oracle@edydr1p2 hadoop-1.2.1]$ mkdir labs
[oracle@edydr1p2 hadoop-1.2.1]$ cd labs

[oracle@edydr1p2 labs]$ vi SingleFileWriteRead.java
```

```
==========================================================================
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FSDataInputStream;
import org.apache.hadoop.fs.FSDataOutputStream;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;

public class SingleFileWriteRead {
  public static void main(String[] args) {
    if (args.length != 2) {
      System.err.println("Usage: SingleFileWriteRead <filename> <contents>");
```

```
      System.exit(2);
    }

    try {
      Configuration conf = new Configuration();
      conf.set("fs.default.name","hdfs://localhost:9000");
      FileSystem hdfs = FileSystem.get(conf);

      Path path = new Path(args[0]);
      if (hdfs.exists(path)) {
        hdfs.delete(path, true);
      }

      FSDataOutputStream outStream = hdfs.create(path);
      outStream.writeUTF(args[1]);
      outStream.close();

      FSDataInputStream inputStream = hdfs.open(path);
      String inputString = inputStream.readUTF();
      inputStream.close();

      System.out.println("## inputString:" + inputString);

    } catch (Exception e) {
      e.printStackTrace();
    }
  }
}
```

```
=============================================================
// 자바에서 하둡 파일 시스템에 접근가능하다.
```

#3. java 파일을 컴파일해서 class 파일을 생성해야 한다.
(컴퓨터가 알아 먹을 수 있는 바이너리 코드로 바꿔준다)

```
[oracle@edydr1p2 labs]$ javac SingleFileWriteRead.java
```

#4. 임의의 first.txt 를 만든다

```
$ vi first.txt
```

```
=======================================
aaaaaa
bbbbb
ccccccc
=======================================
```

#5. 하둡 파일 시스템에 SingleFileWriteRead 클래스 파일을 이용해서 first.txt 를 올린다

```
$ hadoop -cp $CLASSPATH:. SingleFileWriteRead first.txt firstText
```

inputString:firstText

#6. 확인 후 제거

```
$ hadoop fs -ls
$ hadoop fs -cat first.txt

$ hadoop fs -rm first.txt
```

두 개의 파일을 하나로 합쳐서 하둡 파일 시스템에 올리는 실습

PutMerge

```
$ vi PutMerge.java
```

```
================================================================
import java.io.IOException;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FSDataInputStream;
import org.apache.hadoop.fs.FSDataOutputStream;
import org.apache.hadoop.fs.FileStatus;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;

public class PutMerge {

  public static void main(String[] args) throws IOException {

    Configuration conf = new Configuration();
    conf.set("fs.default.name","hdfs://localhost:9000");

    FileSystem hdfs  = FileSystem.get(conf);
    FileSystem local = FileSystem.getLocal(conf);

    Path inputDir = new Path(args[0]);
    Path hdfsFile = new Path(args[1]);

    try {
      FileStatus[] inputFiles = local.listStatus(inputDir);
      FSDataOutputStream out = hdfs.create(hdfsFile);
```

```
      for (int i=0; i<inputFiles.length; i++) {
        System.out.println(inputFiles[i].getPath().getName());
        FSDataInputStream in = local.open(inputFiles[i].getPath());
        byte buffer[] = new byte[256];
        int bytesRead = 0;
        while( (bytesRead = in.read(buffer)) > 0) {
                    out.write(buffer, 0, bytesRead);
        }
        in.close();
      }
      out.close();
      } catch (IOException e) {
        e.printStackTrace();
    }
  }
}
```

==========

```
[orcl:labs]$ javac PutMerge.java
[orcl:labs]$ mkdir inputtext
[orcl:labs]$ cp /etc/hosts inputtext/
[orcl:labs]$ cp /etc/group inputtext/
[orcl:labs]$ ls -l inputtext/
total 8
-rw-r--r-- 1 oracle oinstall 783 Jul  9 16:14 group
-rw-r--r-- 1 oracle oinstall 253 Jul  9 16:14 hosts
```

```
[orcl:labs]$ hadoop -cp $CLASSPATH:. PutMerge inputtext result.txt
hosts
group
```

```
[orcl:labs]$  hadoop fs -ls
Found 5 items
-rw-r--r--   3 oracle supergroup         84 2018-07-05 14:05 /user/oracle/dept2.csv
-rw-r--r--   3 oracle supergroup       1815 2018-07-09 14:17 /user/oracle/emp.csv
-rw-r--r--   3 oracle supergroup        644 2018-07-03 16:37 /user/oracle/emp2.csv
-rw-r--r--   3 oracle supergroup       1036 2018-07-09 16:14 /user/oracle/result.txt
-rw-r--r--   3 oracle supergroup     114548 2018-07-03 16:29 /user/oracle/winter.txt
```

```
[orcl:labs]$ hadoop fs -cat result.txt
```

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1       edydr1p0.us.oracle.com edydr1p0 localhost.localdomain localhost
10.0.2.128      edydr1p0.us.oracle.com edydr1p0 localhost.localdomain localhost
root:x:0:root
bin:x:1:root,bin,daemon
daemon:x:2:root,bin,daemon
sys:x:3:root,bin,adm
adm:x:4:root,adm,daemon
tty:x:5:
disk:x:6:root
lp:x:7:daemon,lp
mem:x:8:
kmem:x:9:
wheel:x:10:root
mail:x:12:mail
news:x:13:news
uucp:x:14:uucp
man:x:15:
games:x:20:
gopher:x:30:
dip:x:40:
```

```
[orcl:labs]$ hadoop fs -rm result.txt
```

Deleted hdfs://localhost:9000/user/oracle/result.txt

```
[orcl:labs]$ rm -rf inputtext/
```

WordCount 예제

#1. 하둡 파일 시스템에 input 이라는 디렉토리를 만든다

```
hadoop fs -mkdir input
```

#2. 하둡 파일 시스템에 input 디렉토리에 로컬에 etc 밑에 hosts 라는 파일을 input 에 올린다.
또 하둡 홈 디렉토리 밑에 README.txt 를 input 에다가 올린다

```
hadoop fs -put /etc/hosts input
hadoop fs -lsr
```

#3. 하둡 홈 디렉토리에 기본적으로 내장되어 있는 맵리듀스 함수를 이용해서 지금 올린 1개의 파일을 워드 카운드 한다

```
hadoop jar $HADOOP_HOME/hadoop-examples-*.jar wordcount input/hosts output1
```

wordcount 라는 클래스 이름을 써줘서 output1 이라는 이름으로 워드카운트 한다

#4. 확인

```
hadoop fs -lsr
hadoop fs -cat output1/part-r-00000
```

```
#       2
10.0.2.128      1
127.0.0.1       1
Do      1
edydr1p0        2
edydr1p0.us.oracle.com  2
fail.   1
following       1
functionality   1
line,   1
localhost       2
localhost.localdomain   2
network 1
not     1
or      1
programs        1
remove  1
require 1
that    1
the     1
various 1
will    1
```

#5. 삭제

```
hadoop fs -rmr output*
```

Deleted hdfs://localhost:9000/user/oracle/output1

문제 125) 겨울왕국 대본 winter.txt 를 하둡 파일 시스템에 올리고 하둡의 맵리듀스 함수로 워드 카운트 하시오

답)

#1. 하둡 파일 시스템에 input2 이라는 디렉토리를 만든다

```
hadoop fs -mkdir input2
```

#2. 하둡 파일 시스템에 input 디렉토리에 로컬에 home/oracle 밑에 winter.txt 라는 파일을 input2 에 올린다.

```
hadoop fs -put /home/oracle/winter.txt input2
hadoop fs -lsr
```

#3. 하둡 홈 디렉토리에 기본적으로 내장되어 있는 맵리듀스 함수를 이용해서 지금 올린 1개의 파일을 워드 카운드 한다

```
hadoop jar $HADOOP_HOME/hadoop-examples-*.jar wordcount input2/winter.txt output2
```

wordcount 라는 클래스 이름을 써줘서 output1 이라는 이름으로 워드카운트 한다

#4. 확인

```
hadoop fs -lsr
hadoop fs -cat output2/part-r-00000
```

깨지기 때문에

#5. get으로 로컬로 받아준다

```
hadoop fs -get output2/part-r-00000 /home/oracle/result_winter.txt


cat result_winter.txt
```

#6. 삭제

```
hadoop fs -rmr output*
```

Deleted hdfs://localhost:9000/user/oracle/output1

---

11 MySQL in CentOS

2018년 7월 3일 화요일 오전 9:58

Table of Contents

- MySQL 에 emp 테이블 생성
- MySQL 에서의 문자함수, 날짜함수
- MySQL 에서의 그룹함수
- MySQL 에서의 JOIN
- MySQL 에서의 DML 문
- MySQL 에서의 DDL 문
- MySQL 에서 외부 테이블 가져오기

설치 (그대로 따라할 것)
https://shas15.github.io/install-centos-mysql/

MySQL 에 emp 테이블 생성

```
create database orcl;

use orcl;

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;
```

MySQL 에서의 문자함수, 날짜함수

문제 126) 직업이 SALESMAN 인 사원들이 이름, 월급, 직업을 출력하는데 월급이 높은 사원부터 출력하시오

답)

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

```
+--------+------+----------+
| ename  | sal  | job      |
+--------+------+----------+
| ALLEN  | 1600 | SALESMAN |
| TURNER | 1500 | SALESMAN |
| MARTIN | 1250 | SALESMAN |
| WARD   | 1250 | SALESMAN |
+--------+------+----------+
4 rows in set (0.00 sec)
```

설명)

오라클과 다르게 salesman 을 소문자로 써도 된다

문제 127) 이름, 커미션을 출력하는데 커미션이 null 인 사원들은 0으로 출력하시오

답)

```
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)
```

설명)

오라클	vs	MySQL
nvl		ifnull

문제 128) 오늘 날짜를 출력하시오

답)

```
select sysdate()
  from dual ;
```

문제 129) 이름, 입사한 날짜부터 오늘까지 총 몇 일 근무했는지 출력하시오

Oracle)

```
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   |                                  13384 |
| BLAKE  |                                  13584 |
| CLARK  |                                  13576 |
| JONES  |                                  13614 |
| MARTIN |                                  13452 |
| ALLEN  |                                  13663 |
| TURNER |                                  13472 |
| JAMES  |                                  13360 |
| WARD   |                                  13651 |
| FORD   |                                  13360 |
| SMITH  |                                  13727 |
| SCOTT  |                                  12984 |
| ADAMS  |                                  12960 |
| MILLER |                                  13329 |
+--------+----------------------------------------+
14 rows in set (0.00 sec)
```

문제 130) 이름, 입사한 날짜부터 오늘까지 총 몇 달 근무했는지 출력하시오

Oracle)

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

MySQL)

```
select ename, period_diff( date_format(now(), '%Y%m'), date_format(hiredate, '%Y%m') ) as
months
  from emp ;
```

```
+--------+--------+
| ename  | months |
+--------+--------+
| KING   |    440 |
| BLAKE  |    446 |
| CLARK  |    446 |
| JONES  |    447 |
| MARTIN |    442 |
| ALLEN  |    449 |
| TURNER |    443 |
| JAMES  |    439 |
| WARD   |    449 |
| FORD   |    439 |
| SMITH  |    451 |
| SCOTT  |    427 |
| ADAMS  |    426 |
| MILLER |    438 |
+--------+--------+
14 rows in set (0.00 sec)
```

문제 131) 오늘부터 100달 뒤에 돌아오는 날짜가 어떻게 되는가 ?

Oracle)

```
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) |
+-------------------------------------------------+
|                                          202910 |
+-------------------------------------------------+
1 row in set (0.00 sec)
```

또는

```
select sysdate() + interval 100 month;
```

```
+---------------------------------+
| sysdate() + interval 100 month |
+---------------------------------+
| 2026-11-10 14:15:35            |
+---------------------------------+
1 row in set (0.00 sec)
```

문제 132) 오늘부터 이번 달 말일까지 총 몇 일 남았는지 출력하시오

Oracle)

```
select last_day(sysdate) - sysdate
  from dual ;
```

MySQL)

```
select to_days(last_day(sysdate())) - to_days(sysdate()) ;
```

```
+-----------------------------------------------------+
| to_days(last_day(sysdate())) - to_days(sysdate()) |
+-----------------------------------------------------+
|                                                  21 |
+-----------------------------------------------------+
1 row in set (0.00 sec)
```

문제 133) 오늘이 무슨 요일인지 출력하시오

Oracle)

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

MySQL)

```
select date_format(sysdate(), '%W') ;
```

문제 134) 이름, 입사일, 입사한 요일을 출력하시오

Oracle)

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

MySQL)

```
select ename, hiredate, date_format(hiredate, '%W') as hiredate_day
  from emp ;
```

```
+--------+------------+--------------+
| ename  | hiredate   | hiredate_day |
+--------+------------+--------------+
| 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 135) 이름, 월급을 출력하는데 월급을 출력할 때에 천 단위를 붙이시오

Oracle)

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

MySQL)

```
select ename, format(sal, 0)
  from emp ;
```

문제 136) 이름, 커미션을 출력하는데 커미션을 출력할 때 커미션이 null 인 사원들은 no comm 이라고 출력하시오

Oracle)

```
select nvl(to_char(comm), 'no comm')
  from emp;
```

MySQL)

```
select ename, ifnull(comm, 'no comm')
  from emp ;
```

```
+--------+--------------------------+
| ename  | ifnull(comm, 'no comm') |
+--------+--------------------------+
| KING   | no comm                  |
| BLAKE  | no comm                  |
| CLARK  | no comm                  |
| JONES  | no comm                  |
| MARTIN | 1400                     |
| ALLEN  | 300                      |
| TURNER | 0                        |
| JAMES  | no comm                  |
| WARD   | 500                      |
| FORD   | no comm                  |
| SMITH  | no comm                  |
| SCOTT  | no comm                  |
| ADAMS  | no comm                  |
| MILLER | no comm                  |
+--------+--------------------------+
14 rows in set (0.00 sec)
```

MySQL 에서의 그룹함수

문제 137) 이름, 직업, 보너스를 출력하는데 직업이 SALESMAN 이면 보너스를 6000 을 출력하고 아니면 0 을 출력하시오

Oracle)

```
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 138) 이름, 부서번호, 보너스를 출력하는데 부서번호가 10번이면 보너스를 7000 으로,
부서번호가 20번이면 보너스를 9000 으로, 부서번호가 30번이면 보너스를 4000 으로 출력하시오

MySQL)

```
select ename, job, if( deptno=10, 7000, if( deptno=20, 9000, 4000 ) ) as bonus
  from emp ;
```

또는

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

```
+--------+--------+-------+
| ename  | deptno | bonus |
+--------+--------+-------+
| 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 (0.00 sec)
```

문제 139) 아래와 같이 결과를 출력하시오

보기)

```
+------+-------+------+
| 10   | 20    | 30   |
+------+-------+------+
| 8750 | 10875 | 9400 |
+------+-------+------+
1 row in set (0.00 sec)
```

Oracle)

```
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;
```

문제 140) 아래와 같이 결과를 출력하시오

보기)

```
+-----------+------+------+------+
| job       | 10   | 20   | 30   |
+-----------+------+------+------+
| ANALYST   | NULL | 6000 | NULL |
| CLERK     | 1300 | 1900 |  950 |
| MANAGER   | 2450 | 2975 | 2850 |
| PRESIDENT | 5000 | NULL | NULL |
| SALESMAN  | NULL | NULL | 5600 |
+-----------+------+------+------+
5 rows in set (0.00 sec)
```

Oracle)

```
select job, 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
group by job;
```

MySQL)

```
select job, 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
```

```
  group by job;
```

문제 141) 아래의 SQL을 MySQL로 구현하시오

Oracle)

```
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 (0.00 sec)
```

문제 142) 아래의 SQL 을 MySQL 로 구현하시오

Oracle)

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

MySQL)

```
select 'total' as deptno, sum(sal)
  from emp
union all
select deptno, sum(sal)
  from emp
  group by deptno with rollup;
```

설명)

```
MySQL 은 grouping sets 와 cube 를 지원하지 않는다
```

rollup 만 with rollup 으로 지원한다

문제 143) 이름, 입사일, 순위를 출력하는데 순위가 최근에 입사한 사원 순으로 순위를 부여하시오

Oracle)

```
select ename, hiredate,
       dense_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 순위 asc ;
```

```
+--------+------------+--------+
| ename  | hiredate   | 순위   |
+--------+------------+--------+
| ADAMS  | 1983-01-15 |      1 |
| SCOTT  | 1982-12-22 |      2 |
| MILLER | 1982-01-11 |      3 |
| JAMES  | 1981-12-11 |      4 |
| FORD   | 1981-12-11 |      4 |
| KING   | 1981-11-17 |      6 |
| MARTIN | 1981-09-10 |      7 |
| TURNER | 1981-08-21 |      8 |
| CLARK  | 1981-05-09 |      9 |
| BLAKE  | 1981-05-01 |     10 |
| JONES  | 1981-04-01 |     11 |
| WARD   | 1981-02-23 |     12 |
| ALLEN  | 1981-02-11 |     13 |
| SMITH  | 1980-12-09 |     14 |
+--------+------------+--------+
14 rows in set (0.00 sec)
```

설명)

스칼라 서브쿼리 설명 : 자기 자신보다 먼저 입사한 사람이 몇 명인지 카운트

문제 144) 이름, 월급, 순위를 출력하는데 순위가 월급이 높은 사원에 순으로 출력하시오

Oracle)

```
select ename, sal, dense_rank() over (order by sal desc) 순위
  from emp ;
```

MySQL)

```
select ename, sal, ( select count(*) + 1
                     from emp
                     where sal > e.sal ) 순위
  from emp e
  order by 순위 asc ;
```

```
+--------+------+--------+
| 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 |
| MARTIN | 1250 |     10 |
| WARD   | 1250 |     10 |
| ADAMS  | 1100 |     12 |
| JAMES  |  950 |     13 |
| SMITH  |  800 |     14 |
+--------+------+--------+
14 rows in set (0.00 sec)
```

문제 145) 부서번호, 이름, 월급, 순위를 출력하는데 순위가 부서번호 별로 각각 월급이 높은 순서대로 순위를 출력하시오

Oracle)

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

MySQL)

```
select deptno, ename, sal, ( select count(*) + 1
                             from emp
                             where sal > e.sal and deptno = e.deptno ) 순위
  from emp e
  order by deptno, 순위 asc ;
```

```
+--------+--------+------+--------+
| 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 | MARTIN | 1250 |      4 |
|     30 | WARD   | 1250 |      4 |
|     30 | JAMES  |  950 |      6 |
+--------+--------+------+--------+
14 rows in set (0.00 sec)
```

문제 146) 부서번호, 해당 부서번호에 속한 사원들의 이름을 가로로 출력하시오

Oracle)

```
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 (0.01 sec)
```

MySQL 에서의 JOIN

문제 147) 이름, 부서위치를 출력하시오

MySQL)

```
select e.ename, d.loc
  from emp e, dept d
  where 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 |
+--------+----------+
14 rows in set (0.00 sec)
```

문제 148) 이름, 부서위치를 출력하는데 outer join 도 되는지 확인하시오

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 (0.00 sec)
```

문제 149) 아래의 full outer join 을 MySQL 에서 구현하시오

보기)

```
+--------+----------+
| ename  | loc      |
+--------+----------+
| KING   | NEW YORK |
| CLARK  | NEW YORK |
| MILLER | NEW YORK |
| JONES  | DALLAS   |
| FORD   | DALLAS   |
| SMITH  | DALLAS   |
| SCOTT  | DALLAS   |
| ADAMS  | DALLAS   |
| BLAKE  | CHICAGO  |
| MARTIN | CHICAGO  |
| ALLEN  | CHICAGO  |
| TURNER | CHICAGO  |
| JAMES  | CHICAGO  |
| WARD   | CHICAGO  |
| NULL   | BOSTON   |
+--------+----------+
15 rows in set (0.00 sec)
```

Oracle)

```
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 ) ;
```

설명)

union all을 쓰지 않고 union 을 쓴 이유는 중복된 값을 출력하지 않기 위해서

문제 150) JONES의 월급보다 더 많은 월급을 받는 사원들의 이름과 월급을 출력하시오

Oracle)

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

MySQL)

```
select ename, sal
  from emp
```

```
where sal > ( select sal from emp where ename = 'jones' ) ;
```

```
+-------+------+
| ename | sal  |
+-------+------+
| KING  | 5000 |
| FORD  | 3000 |
| SCOTT | 3000 |
+-------+------+
3 rows in set (0.00 sec)
```

MySQL 에서의 DML 문

문제 151) SCOTT의 월급을 0으로 변경하시오

Oracle)

```
update emp
  set sal = 0
  where ename = 'SCOTT' ;
```

MySQL)

```
update emp
  set sal = 0
  where ename = 'SCOTT' ;

rollback ;
```

설명)

```
rollback이 안된다. 왜냐하면 auto commit 기능이 있기 때문이다.
따라서 그 기능을 꺼줘야 한다

set autocommit=false ;
select @@autocommit ;
```

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

문제 152) 사원 테이블을 전체 delete 하고 rollback 되는지 확인하시오

MySQL)

```
delete from emp;
select * from emp;
rollback;
select * from emp;
```

문제 153) dept 테이블 스크립트를 이용해서 dept2 테이블을 생성하고 dept2를 select 한 후에 rollback을 하면 dept2 테이블이 어떻게 되는지 확인해 보시오

MySQL)

```
CREATE TABLE dept3
 (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');

select * from dept2;
rollback;
select * from dept2;
```

Empty set (0.00 sec)

설명)

```
DDL문(create, alter, drop)은 자동 COMMIT 된다
```

MySQL 에서의 DDL 문

문제 154) salgrade 테이블을 생성하시오

MySQL)

```
create table salgrade
(grade    int,
 losal    int,
 hisal    int);



insert into salgrade values(1,700,1200);
```

```
insert into salgrade values(2,1201,1400);
insert into salgrade values(3,1401,2000);
insert into salgrade values(4,2001,3000);
insert into salgrade values(5,3001,9999);

commit;
```

MySQL 에서 외부 테이블 가져오기

문제 155) emp2.csv 를 가지고 external 테이블을 생성하시오

MySQL)

```
CREATE TABLE emp2 (
empno int NOT NULL,
ename VARCHAR(10),
job VARCHAR(9),
mgr int,
hiredate DATE,
sal int,
comm int,
deptno int );

load data local infile '/root/emp2.csv'
  into table emp2 fields terminated by ',' ;

select * from emp2 ;
```

설명)

```
오라클과 다른점은 테이블이 실제로 생성된 거라서 DML 다 가능하고 인덱스도 생성된다
```

문제 156) price.csv 를 가지고 price 테이블을 생성하시오

MySQL)

```
alter database orcl default character set utf8 COLLATE utf8_general_ci;
quit

[root@KRKim ~]# sudo service mysql restart;

create  table price
```

```
(
P_SEQ    int,
M_SEQ    int,
M_NAME   varchar(80),
A_SEQ    int,
A_NAME    varchar(60),
A_UNIT    varchar(40),
A_PRICE    int,
P_YEAR_MONTH    varchar(30),
ADD_COL          varchar(180),
M_TYPE_CODE    varchar(20),
M_TYPE_NAME    varchar(20),
M_GU_CODE    varchar(10),
M_GU_NAME    varchar(30) );

load data local infile '/root/price.txt'
  into table price character set utf8 fields terminated by '₩t';

select * from price ;
```

---

12 실제 리눅스 서버에 하둡 멀티노드 구성

2018년 7월 11일 수요일 오전 10:50

따로 실습해볼 예정임
