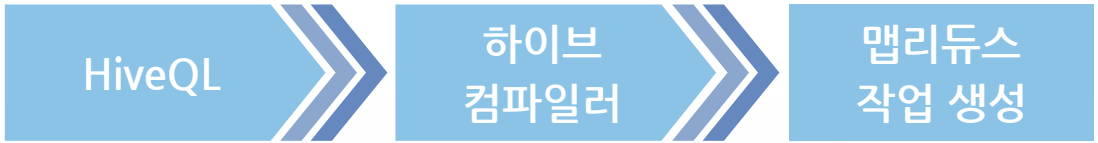
# LESSON 1

# 하이브(Hive)

# 하이브(Hive)

## Apache Hive

- 클라우데라에서 개발
- SQL-like HiveQL 사용
- 커스텀 맵/리듀스 작업 플러그인 가능




- 메타스토어(Metastore)로 MySQL을 사용
  - MySQL이 설치되어 있어야 함

MySQL설치 동영상

# 하이브(Hive)

## Hive 설치

다운로드

$ wget http://apache.mirror.cdnetworks.com/hive/hive-2.1.0/apache-hive-2.1.0-bin.tar.gz

압축해제

$ tar xvfz /apache-hive-2.1.0-bin.tar.gz

환경변수(~/.profile 이나 ~/.bashrc)

export HIVE_HOME=/apache-hive-2.1.0-bin

export PATH=$PATH:/$HIVE_HOME/bin

Source ~/.profile or source ~/.bashrc

# 하이브(Hive)

## Hive 설치

초기화

```
$ bin/schematool -initSchema -dbType derby
```

에러 발생시

```
$ mv metastore_db metastore_db.tmp
```

하이브 실행

```
$ start-dfs.sh
$ hive
```

데이터베이스 조회

```
show databases;
Show tables;
```

박사님 여기서는 대문자를 쓰는 것이 맞는지 확인 부탁드립니다.

# 하이브(Hive)

## 하이브 메타스토어 설정

```
$ nano conf/hive-site.xml
<property>
  <name>hive.metastore.local</name>
  <value>true</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://localhost:3306/hive?createDatabaseIfNotExist=true</value>
  <description>JDBC connect string for a JDBC metastore</description>
</property>

<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>com.mysql.jdbc.Driver</value>
  <description>Driver class name for a JDBC metastore</description>
</property>
```

# 하이브(Hive)

## 하이브 메타스토어 설정

```
<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value>hiveid</value>
  <description>username to use against metastore database</description>
</property>

<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value>hivepass</value>
  <description>password to use against metastore database</description>
</property>
```

warehouse

# 하이브(Hive)

## 하이브 메타스토어 설정

MySQL 서버 시작

service mysql start

mysql>grant all privileges on *.* to hiveid@localhost identified by 'hivepass' with grant option

hive> create database hive;

# 하이브(Hive)

## 하이브 메타스토어 설정

MySQL JDBC 드라이버 설치

$ apt-get install libmysql-java

드라이버,복사/

$ cp /usr/share/java/mysql-connector-java.jar $HIVE_HOME/lib

# 하이브(Hive)

## 하이브 실습

HDFS 권한설정
hadoop fs -mkdir /tmp
hadoop fs -mkdir /user/hive/warehouse
hadoop fs -chmod g+w /tmp
hadoop fs -chmod g+w /user/hive/warehousew

테이블 생성
CREATE TABLE bigmark (id int, mark STRING) ROW FORMAT DELIMITED FIELDS ₩
TERMINATED BY ',' LINES TERMINATED BY '₩n';
테이블에 파일 내용 읽어오기
LOAD DATA LOCAL INPATH '/home/bigmark/localfiles/bigmark.csv' ₩
OVERWRITE INTO TABLE bigmark;
/user/hive/warehouse

하이브 실습

# LESSON 2

## Apache HBase

# Apache HBase

- http://hbase.apache.org
- 자바기반의 HDFS에 위에서 동작하는 NoSQL 데이터스토어
- HDFS의 특징(분산파일시스템, 고가용성)을 공유
- 주키퍼(Zookeeper) 사용해 HA(High Availability) 제공
- 기존의 하둡시스템과 동시에 운영이 가능

# Apache HBase

## NoSQL

- NoSQL(No! SQL? Not only SQL?)
- NoSQL은 빠른 쓰기를 목표로 한다.
  - RDBMS가 빠른 읽기에 최적화
- NoSQL은 주로 키/밸류 형태의 구조를 띄고 있다.
- 데이터 모델에 따른 분류
  - KV Store/Ordered KV Store/Document KV store
  - CAP(Consistency, Availability, Partition Tolerence)
- 기본적은 대량의 분산처리를 목표로 하는 빅데이터와는 성격이 다름
- 적(RDBMS/오라클)의 적(BigData, NoSQL)은 동지

# Apache HBase

## HBase의 구조

- 컬럼 : 기본 단위
- 로우 : 여러 개의 컬럼이 모여서 하나의 로우를 이룸
- 테이블 : 여러 개의 로우가 모여 하나의 테이블이 됨
- 각 컬럼은 여러 개의 버전을 가질 수 있음
  - 별도의 셀(CELL)에 저장
- 로우 키를 기준으로 사전식으로 정렬된다.

# Apache HBase

## HBase의 구조

- 컬럼은 컬럼패밀리로 그룹화된다.
- 모든 컬럼 값(셀)은 타임스탬프가 있음
  - 시스템 내부적으로/명시적으로 설정가능
  - 동일한 값이 여러 개가 저장 가능함
- 셀 단위로는 타임스탬프의 내림차순으로 저장
  - 최신값을 먼저 읽을 수 있도록
- (Table, RowKey, Family, Column, Timestamp) → Value
- SortedMap<RowKey, List<SortedMap<Column, List <Value, TimeStamp>>>>

# Apache HBase

## HBase의 특징

- 자동 샤딩(Auto Sharding) 지원
- 리전(Region)
  - HBase의 확장성 및 로드밸런싱의 기본단위
  - 함께 저장된 인접한 범위의 로우들
  - 리전이 커지면 시스템에 의해 동적으로 분할/통합되기도 함
    - 서버가 고장났을 때 재빨리 복구해주고 부하가 심할 때 이동 가능함
  - 서버당 리전 개수는 10-1000 사이의 값
  - 리전의 크기는 1-2GB정도의 사이즈

# Apache HBase

## HBase의 특징

- 데이터는 HFile에 저장
  - 연속적인 블럭, 블럭에 대한 색인이 마지막에 저장
- 색인은 HFile이 열릴 때 메모리에 로드
  - 바이너리서치(이진 탐색) 사용
- 블록 크기는 64KB가 기본
- Memstore(메모리상의 저장소)
- Write-Ahead Log
  - 데이터의 변경사항 memstore에 저장
    → 일정 크기가 되면 HFile에 저장

# Apache HBase

## HBase의 특징

- 테이블 스캔
  - O(N)
- 로우키 탐색/데이터 변경
  - O(logN)
  - 블룸피터사용할 경우는 O(1)
- 컬럼 지향 구조에서는 NULL을 저장하는 공간이 필요 없음

# Apache HBase

## Hbase 설정

- HBase
  - 로컬파일시스템
  - HDFS
  - hdfs://<namenode>:<port>:<path>
  - Amazon S3
- 스탠드 얼론(Standalone)
- 의사분산(Pseudo Distributed Mode)
- 완전분산(Full Distributed Mode)
  - sudo nano conf/hbase-site.xml
  - hadoop과 동일
  - <property>
    - <name>hbase.cluster.distributed</name>
    - <value>true</value>
  - conf/regionservers에 파일 지정

# Apache HBase

## Hbase 설치

다운로드

wget http://apache.mirror.cdnetworks.com/hbase/stable/hbase-1.2.2-bin.tar.gz

압축해제

tar xvfz hbase-1.2.2-bin.tar.gz

설정파일 수정

hbase-env.sh

export JAVA_HOME=자바 디렉토리(/usr/lib/jvm/..)

# Apache HBase

## Hbase 설치

hbase-site.xml

```
<configuration>
<property>
<name>hbase.cluster.distributed</name>
<value>true</value>
</property>
<property>
<name>hbase:rootdir</name>
<value>hdfs://localhost:9000/hbase</value>
</property>
</configuration>
```

# Apache HBase

## HBase 실행

$ start-dfs.sh/ $ start-yarn.sh
$ start-hbase.sh
$ jps
...
3800 HQuorumPeer
3981 HRegionServer
3855 HMaster

$ hadoop fs -ls /tmp/hbase-root/hbase
$ hbase shell
실행상태 확인
http://localhost:16010/master-status

# Apache HBase

## HBase 실습

```
$ hbase shell
create 'Contacts', 'Personal', 'Office'
List
put 'Contacts', '1000', 'Personal:Name', 'John Dole'
put 'Contacts', '1000', 'Personal:Phone', '1-425-000-0001'
put 'Contacts', '1000', 'Office:Phone', '1-425-000-0002'
put 'Contacts', '1000', 'Office:Address', '1111 San Gabriel Dr.'
scan 'Contacts'
ROW          COLUMN+CELL
1000         column=Office:Address, timestamp=1469692701344, value=1111 San Gabriel Dr.
1000         column=Office:Phone, timestamp=1469692699480, value=1-425-000-0002
1000         column=Personal:Name, timestamp=1469692699212, value=John Dole
1000         column=Personal:Phone, timestamp=1469692699320, value=1-425-000-0001
1 row(s) in 0.1300 seconds

get 'Contacts', '1000'
```