[Hadoop] Lab02 Sqoop를 이해하기

학습 목표

가. Sqoop은 무엇일까?

나. sqoop의 tool를 이용하여 MySQL의 Table를 HDFS로 데이터를 옮기는 것에 대해 이해한다.

01 스쿱(Sqoop)은 무엇일까?

- 스쿱은 **하둡 생태계에 속하는 아파치 프로젝트**이다.
- 스쿱은 클러스터 간에 데이터 이동하는 대신 **JDBC드라이버를 사용**한다.
- JDBC 드라이버를 이용한 관계형 데이터베이스(RDBMS)에 데이터를 내보내거나 가져오도록 설계되었다.

02 작업을 위한 사전준비

(가) Cloudera를 시작한다.

(나) mysql 의 기본 암호 설정 및 root의 localhost를 확인한다.

```
# 암호 재설정
$ mysql -u root -p
Enter password:[cloudera]
UPDATE mysql.user
    SET Password=PASSWORD('1234')
WHERE User='root';
FLUSH PRIVILEGES;
```

```
# localhost 확인
MySql 5.7미만
mysql> SELECT Host, User FROM mysql.user;
```

[결과 내용]

```
mysql> select host, user from mysql.user;
+-----------------------+----------+
| host                  | user     |
+-----------------------+----------+
| %                     | hiveuser |
| 127.0.0.1             | root     |
| localhost             |          |
| localhost             | hive     |
| localhost             | hiveuser |
| localhost             | hue      |
| localhost             | root     |
| localhost             | training |
| localhost.localdomain |          |
| localhost.localdomain | root     |
+-----------------------+----------+
10 rows in set (0.00 sec)
```

도움말

sqoop help

sqoop list-tables --> 엔터를 치면 상세한 내용이 나온다.

03 Mysql 전체 테이블 리스트 확인해보기

[사용법]

```
sqoop list-tables
      --connect  <jdbc-uri>       # JDBC connect 문자열을 지정
      --username  dbuser          # user(사용자를 지정한다.)
     --password   dbpassword  # db의 password를 지정한다.
```

[실행 명령어]

```
sqoop list-tables --connect jdbc:mysql://localhost/dbname --username  root --password  1234
```

[실행결과]

```
[training@localhost Desktop]$ sqoop list-tables --connect jdbc:mysql://localhost/training --username root --password 1234
18/05/13 08:30:17 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
18/05/13 08:30:17 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
Movies
bible_freq
cityByCountry
countries
shake_freq
```

04 전체 테이블 임포트하기

```
sqoop import-all-tables \
  --connect jdbc:mysql://localhost/training  \
  --username  root  \
  --password   1234
```

05 하나의 테이블 임포트

Movies의 테이블을 임포트한다.

```
sqoop import  --table  Movies \
    --connect  jdbc:mysql://localhost/training \
    --username  root  --password  1234  \
    --fields-terminated-by   "\t"
```

[실행결과]

```
[training@localhost Desktop]$ sqoop import  --table  Movies \
> --connect  jdbc:mysql://localhost/training \
> --username  root  --password  1234  \
> --fields-terminated-by   "\t"
18/05/13 08:40:29 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
18/05/13 08:40:29 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
18/05/13 08:40:29 INFO tool.CodeGenTool: Beginning code generation
18/05/13 08:40:30 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `Movies` AS t LIMIT 1
18/05/13 08:40:30 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `Movies` AS t LIMIT 1
18/05/13 08:40:30 INFO orm.CompilationManager: HADOOP_HOME is /usr/lib/hadoop
Note: /tmp/sqoop-training/compile/94861fefc5b305b7642ab8ebd7b520f4/Movies.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
18/05/13 08:40:34 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-training/compile/94861fefc5b305b7642ab8ebd7b520f4/Movies.jar
18/05/13 08:40:34 WARN manager.MySQLManager: It looks like you are importing from mysql.
18/05/13 08:40:34 WARN manager.MySQLManager: This transfer can be faster! Use the --direct
18/05/13 08:40:34 WARN manager.MySQLManager: option to exercise a MySQL-specific fast path.
18/05/13 08:40:34 INFO manager.MySQLManager: Setting zero DATETIME behavior to convertToNull (mysql)
18/05/13 08:40:34 INFO mapreduce.ImportJobBase: Beginning import of Movies
18/05/13 08:40:37 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. Applications should implement Tool for the
same.
18/05/13 08:40:39 INFO db.DataDrivenDBInputFormat: BoundingValsQuery: SELECT MIN(`movieid`), MAX(`movieid`) FROM `Movies`
18/05/13 08:40:39 INFO mapred.JobClient: Running job: job_201805091052_0005
18/05/13 08:40:40 INFO mapred.JobClient:  map 0% reduce 0%
18/05/13 08:40:58 INFO mapred.JobClient:  map 50% reduce 0%
18/05/13 08:41:10 INFO mapred.JobClient:  map 75% reduce 0%
18/05/13 08:41:11 INFO mapred.JobClient:  map 100% reduce 0%
18/05/13 08:41:14 INFO mapred.JobClient: Job complete: job_201805091052_0005
18/05/13 08:41:14 INFO mapred.JobClient: Counters: 23
18/05/13 08:41:14 INFO mapred.JobClient:   File System Counters
18/05/13 08:41:14 INFO mapred.JobClient:     FILE: Number of bytes read=0
18/05/13 08:41:14 INFO mapred.JobClient:     FILE: Number of bytes written=797564
```

```
18/05/13 08:41:14 INFO mapred.JobClient:     FILE: Number of read operations=0
18/05/13 08:41:14 INFO mapred.JobClient:     FILE: Number of large read operations=0
18/05/13 08:41:14 INFO mapred.JobClient:     FILE: Number of write operations=0
18/05/13 08:41:14 INFO mapred.JobClient:     HDFS: Number of bytes read=450
18/05/13 08:41:14 INFO mapred.JobClient:     HDFS: Number of bytes written=234674
18/05/13 08:41:14 INFO mapred.JobClient:     HDFS: Number of read operations=4
18/05/13 08:41:14 INFO mapred.JobClient:     HDFS: Number of large read operations=0
18/05/13 08:41:14 INFO mapred.JobClient:     HDFS: Number of write operations=4
18/05/13 08:41:14 INFO mapred.JobClient:   Job Counters
18/05/13 08:41:14 INFO mapred.JobClient:     Launched map tasks=4
18/05/13 08:41:14 INFO mapred.JobClient:     Total time spent by all maps in occupied slots (ms)=54156
18/05/13 08:41:14 INFO mapred.JobClient:     Total time spent by all reduces in occupied slots (ms)=0
18/05/13 08:41:14 INFO mapred.JobClient:     Total time spent by all maps waiting after reserving slots (ms)=0
18/05/13 08:41:14 INFO mapred.JobClient:     Total time spent by all reduces waiting after reserving slots (ms)=0
18/05/13 08:41:14 INFO mapred.JobClient:   Map-Reduce Framework
18/05/13 08:41:14 INFO mapred.JobClient:     Map input records=1682
18/05/13 08:41:14 INFO mapred.JobClient:     Map output records=1682
18/05/13 08:41:14 INFO mapred.JobClient:     Input split bytes=450
18/05/13 08:41:14 INFO mapred.JobClient:     Spilled Records=0
18/05/13 08:41:14 INFO mapred.JobClient:     CPU time spent (ms)=5660
18/05/13 08:41:14 INFO mapred.JobClient:     Physical memory (bytes) snapshot=198574080
18/05/13 08:41:14 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=1552662528
18/05/13 08:41:14 INFO mapred.JobClient:     Total committed heap usage (bytes)=63963136
18/05/13 08:41:14 INFO mapreduce.ImportJobBase: Transferred 0 bytes in 39.0823 seconds (0 bytes/sec)
18/05/13 08:41:14 INFO mapreduce.ImportJobBase: Retrieved 1682 records.
```

```
[HDFS 확인]
[training@localhost Desktop]$ hadoop fs -ls -R
drwxr-xr-x   - training supergroup          0 2018-05-13 08:41 Movies
-rw-r--r--   1 training supergroup          0 2018-05-13 08:41 Movies/_SUCCESS
drwxr-xr-x   - training supergroup          0 2018-05-13 08:40 Movies/_logs
drwxr-xr-x   - training supergroup          0 2018-05-13 08:40 Movies/_logs/history
-rw-r--r--   1 training supergroup      87015 2018-05-13 08:40 Movies/_logs/history/0.0.0.0_1525877582151_job_201805091052_0005_conf.xml
-rw-r--r--   1 training supergroup      22177 2018-05-13 08:41 Movies/_logs/history/job_201805091052_0005_1526215239637_training_Movies.jar
-rw-r--r--   1 training supergroup      58800 2018-05-13 08:40 Movies/part-m-00000
-rw-r--r--   1 training supergroup      58986 2018-05-13 08:40 Movies/part-m-00001
-rw-r--r--   1 training supergroup      58458 2018-05-13 08:41 Movies/part-m-00002
-rw-r--r--   1 training supergroup      58430 2018-05-13 08:41 Movies/part-m-00003
```

# 데이터 확인

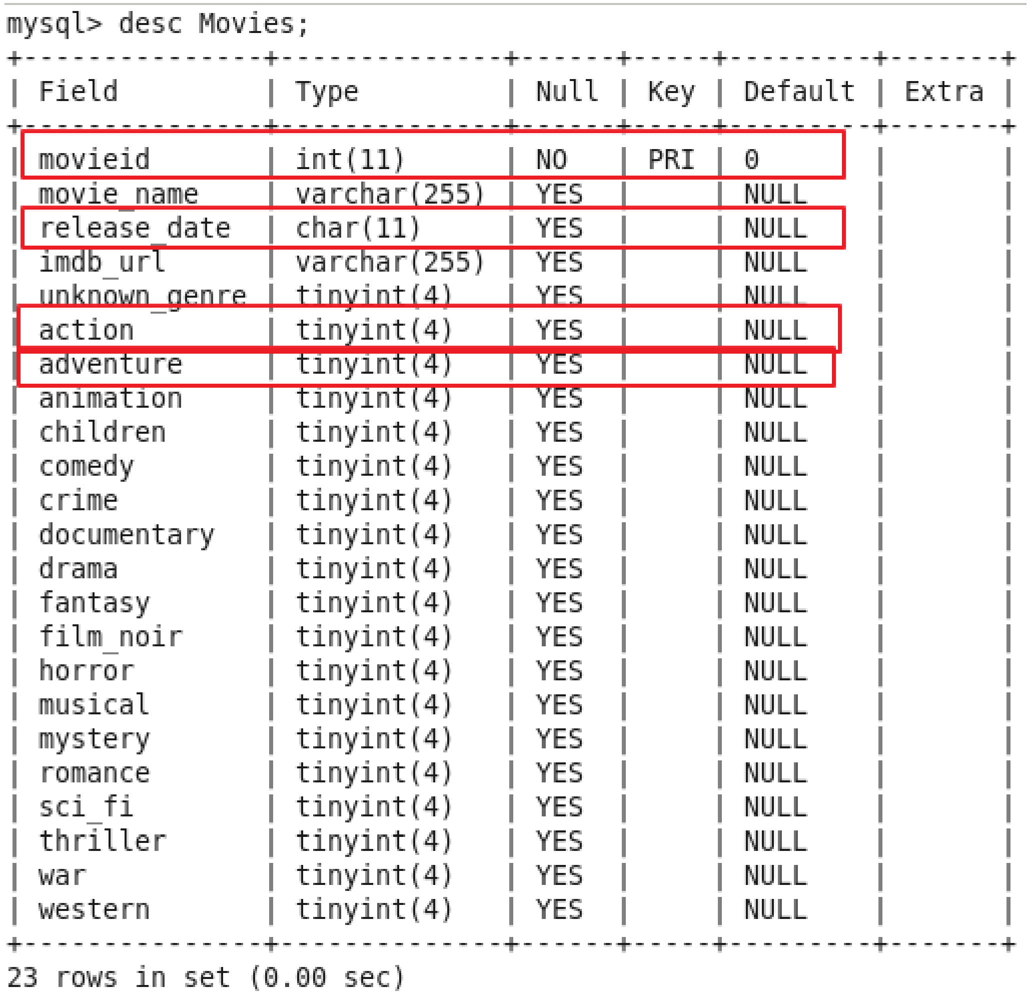
```
[training@localhost Desktop]$ hadoop fs -cat Movies/part-m-00000
...
```

(참고 mysql 확인시)

```
mysql> desc Movies;
+---------------+--------------+------+-----+---------+-------+
| Field         | Type         | Null | Key | Default | Extra |
+---------------+--------------+------+-----+---------+-------+
| movieid       | int(11)      | NO   | PRI | 0       |       |
| movie_name    | varchar(255) | YES  |     | NULL    |       |
| release_date  | char(11)     | YES  |     | NULL    |       |
| imdb_url      | varchar(255) | YES  |     | NULL    |       |
| unknown_genre | tinyint(4)   | YES  |     | NULL    |       |
| action        | tinyint(4)   | YES  |     | NULL    |       |
| adventure     | tinyint(4)   | YES  |     | NULL    |       |
| animation     | tinyint(4)   | YES  |     | NULL    |       |
| children      | tinyint(4)   | YES  |     | NULL    |       |
| comedy        | tinyint(4)   | YES  |     | NULL    |       |
| crime         | tinyint(4)   | YES  |     | NULL    |       |
| documentary   | tinyint(4)   | YES  |     | NULL    |       |
| drama         | tinyint(4)   | YES  |     | NULL    |       |
| fantasy       | tinyint(4)   | YES  |     | NULL    |       |
| film_noir     | tinyint(4)   | YES  |     | NULL    |       |
| horror        | tinyint(4)   | YES  |     | NULL    |       |
| musical       | tinyint(4)   | YES  |     | NULL    |       |
| mystery       | tinyint(4)   | YES  |     | NULL    |       |
| romance       | tinyint(4)   | YES  |     | NULL    |       |
| sci_fi        | tinyint(4)   | YES  |     | NULL    |       |
| thriller      | tinyint(4)   | YES  |     | NULL    |       |
| war           | tinyint(4)   | YES  |     | NULL    |       |
| western       | tinyint(4)   | YES  |     | NULL    |       |
+---------------+--------------+------+-----+---------+-------+
23 rows in set (0.01 sec)
```

[실습 1-1] training database안에 cityByCountry를 HDFS 시스템으로 임포트 해 보자.

06 부분적인 테이블 임포트



```
mysql> desc Movies;
+---------------+--------------+------+-----+---------+-------+
| Field         | Type         | Null | Key | Default | Extra |
+---------------+--------------+------+-----+---------+-------+
| movieid       | int(11)      | NO   | PRI | 0       |       |
| movie_name    | varchar(255) | YES  |     | NULL    |       |
| release_date  | char(11)     | YES  |     | NULL    |       |
| imdb_url      | varchar(255) | YES  |     | NULL    |       |
| unknown_genre | tinyint(4)   | YES  |     | NULL    |       |
| action        | tinyint(4)   | YES  |     | NULL    |       |
| adventure     | tinyint(4)   | YES  |     | NULL    |       |
| animation     | tinyint(4)   | YES  |     | NULL    |       |
| children      | tinyint(4)   | YES  |     | NULL    |       |
| comedy        | tinyint(4)   | YES  |     | NULL    |       |
| crime         | tinyint(4)   | YES  |     | NULL    |       |
| documentary   | tinyint(4)   | YES  |     | NULL    |       |
| drama         | tinyint(4)   | YES  |     | NULL    |       |
| fantasy       | tinyint(4)   | YES  |     | NULL    |       |
| film_noir     | tinyint(4)   | YES  |     | NULL    |       |
| horror        | tinyint(4)   | YES  |     | NULL    |       |
| musical       | tinyint(4)   | YES  |     | NULL    |       |
| mystery       | tinyint(4)   | YES  |     | NULL    |       |
| romance       | tinyint(4)   | YES  |     | NULL    |       |
| sci_fi        | tinyint(4)   | YES  |     | NULL    |       |
| thriller      | tinyint(4)   | YES  |     | NULL    |       |
| war           | tinyint(4)   | YES  |     | NULL    |       |
| western       | tinyint(4)   | YES  |     | NULL    |       |
+---------------+--------------+------+-----+---------+-------+
23 rows in set (0.00 sec)
```

(가) Mysql의 Movies 테이블의 일부 컬럼 임포트

우리는 movieid, movie_name, release_date, action, adventure를 임포트해보자.

```
sqoop  import  --table  Movies \
  --connect jdbc:mysql://localhost/training \
  --username root --password 1234 \
  --columns "movieid, movie_name, release_date, action, adventure"
```

[실행결과]

```
[training@localhost Desktop]$ sqoop  import  --table  Movies \
>   --connect jdbc:mysql://localhost/training \
>   --username root --password 1234 \
>   --columns "movieid, movie_name, release_date, action, adventure"
18/05/13 12:28:48 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
18/05/13 12:28:49 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
18/05/13 12:28:49 INFO tool.CodeGenTool: Beginning code generation
18/05/13 12:28:49 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `Movies` AS t LIMIT 1
18/05/13 12:28:49 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `Movies` AS t LIMIT 1
18/05/13 12:28:49 INFO orm.CompilationManager: HADOOP_HOME is /usr/lib/hadoop
Note: /tmp/sqoop-training/compile/92c1fc3258c9ee22643bc5a9f58a25e6/Movies.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
18/05/13 12:28:52 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-training/compile/92c1fc3258c9ee22643bc5a9f58a25e6/Movies.jar
18/05/13 12:28:52 WARN manager.MySQLManager: It looks like you are importing from mysql.
18/05/13 12:28:52 WARN manager.MySQLManager: This transfer can be faster! Use the --direct
18/05/13 12:28:52 WARN manager.MySQLManager: option to exercise a MySQL-specific fast path.
18/05/13 12:28:52 INFO manager.MySQLManager: Setting zero DATETIME behavior to convertToNull (mysql)
18/05/13 12:28:52 INFO mapreduce.ImportJobBase: Beginning import of Movies
18/05/13 12:28:55 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. Applications should implement Tool for the same.
18/05/13 12:28:57 INFO db.DataDrivenDBInputFormat: BoundingValsQuery: SELECT MIN(`movieid`), MAX(`movieid`) FROM `Movies`
18/05/13 12:28:58 INFO mapred.JobClient: Running job: job_201805131018_0015
18/05/13 12:28:59 INFO mapred.JobClient:  map 0% reduce 0%
18/05/13 12:29:20 INFO mapred.JobClient:  map 50% reduce 0%
18/05/13 12:29:34 INFO mapred.JobClient:  map 100% reduce 0%
18/05/13 12:29:37 INFO mapred.JobClient: Job complete: job_201805131018_0015
18/05/13 12:29:37 INFO mapred.JobClient: Counters: 23
18/05/13 12:29:37 INFO mapred.JobClient:   File System Counters
18/05/13 12:29:37 INFO mapred.JobClient:     FILE: Number of bytes read=0
18/05/13 12:29:37 INFO mapred.JobClient:     FILE: Number of bytes written=796816
18/05/13 12:29:37 INFO mapred.JobClient:     FILE: Number of read operations=0
18/05/13 12:29:37 INFO mapred.JobClient:     FILE: Number of large read operations=0
18/05/13 12:29:37 INFO mapred.JobClient:     FILE: Number of write operations=0
18/05/13 12:29:37 INFO mapred.JobClient:     HDFS: Number of bytes read=450
18/05/13 12:29:37 INFO mapred.JobClient:     HDFS: Number of bytes written=75356
18/05/13 12:29:37 INFO mapred.JobClient:     HDFS: Number of read operations=4
18/05/13 12:29:37 INFO mapred.JobClient:     HDFS: Number of large read operations=0
18/05/13 12:29:37 INFO mapred.JobClient:     HDFS: Number of write operations=4
18/05/13 12:29:37 INFO mapred.JobClient:   Job Counters
18/05/13 12:29:37 INFO mapred.JobClient:     Launched map tasks=4
18/05/13 12:29:37 INFO mapred.JobClient:     Total time spent by all maps in occupied slots (ms)=65796
18/05/13 12:29:37 INFO mapred.JobClient:     Total time spent by all reduces in occupied slots (ms)=0
```

```
18/05/13 12:29:37 INFO mapred.JobClient:     Total time spent by all maps waiting after reserving slots (ms)=0
18/05/13 12:29:37 INFO mapred.JobClient:     Total time spent by all reduces waiting after reserving slots (ms)=0
18/05/13 12:29:37 INFO mapred.JobClient:   Map-Reduce Framework
18/05/13 12:29:37 INFO mapred.JobClient:     Map input records=1682
18/05/13 12:29:37 INFO mapred.JobClient:     Map output records=1682
18/05/13 12:29:37 INFO mapred.JobClient:     Input split bytes=450
18/05/13 12:29:37 INFO mapred.JobClient:     Spilled Records=0
18/05/13 12:29:37 INFO mapred.JobClient:     CPU time spent (ms)=7150
18/05/13 12:29:37 INFO mapred.JobClient:     Physical memory (bytes) snapshot=198578176
18/05/13 12:29:37 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=1552662528
18/05/13 12:29:37 INFO mapred.JobClient:     Total committed heap usage (bytes)=63963136
18/05/13 12:29:37 INFO mapreduce.ImportJobBase: Transferred 0 bytes in 44.062 seconds (0 bytes/sec)
18/05/13 12:29:37 INFO mapreduce.ImportJobBase: Retrieved 1682 records.
```

만약 다른 조건으로 **다른 디렉터리로 임포트(보내기)**하고 싶다면 아래를 참조하자.

```
sqoop  import  --table  Movies \
  --connect jdbc:mysql://localhost/training \
  --username root --password 1234 \
  --columns "movieid, action"  \
  --target-dir  training/MovieTbl
```

[결과 확인]

```
[training@localhost Desktop]$ hadoop fs -ls -R Movies
-rw-r--r--   1 training supergroup          0 2018-05-13 11:09 Movies/_SUCCESS
drwxr-xr-x   - training supergroup          0 2018-05-13 11:08 Movies/_logs
drwxr-xr-x   - training supergroup          0 2018-05-13 11:08 Movies/_logs/history
-rw-r--r--   1 training supergroup      86828 2018-05-13 11:08 Movies/_logs/history/0.0.0.0_1526221129783_job_201805131018_0004_conf.xml
-rw-r--r--   1 training supergroup      22177 2018-05-13 11:09 Movies/_logs/history/job_201805131018_0004_1526224117409_training_Movies.jar
-rw-r--r--   1 training supergroup      43644 2018-05-13 11:08 Movies/part-m-00000
-rw-r--r--   1 training supergroup      43866 2018-05-13 11:08 Movies/part-m-00001
-rw-r--r--   1 training supergroup      43338 2018-05-13 11:09 Movies/part-m-00002
-rw-r--r--   1 training supergroup      43274 2018-05-13 11:09 Movies/part-m-00003

[training@localhost Desktop]$ hadoop fs -ls -R training/MovieTbl
...
-rw-r--r--   1 training supergroup      22167 2018-05-13 12:12
training/MovieTbl/_logs/history/job_201805131018_0013_1526227932030_training_Movies.jar
-rw-r--r--   1 training supergroup       2418 2018-05-13 12:12 training/MovieTbl/part-m-00000
-rw-r--r--   1 training supergroup       2520 2018-05-13 12:12 training/MovieTbl/part-m-00001
```

```
-rw-r--r--   1 training supergroup       2782 2018-05-13 12:12 training/MovieTbl/part-m-00002
-rw-r--r--   1 training supergroup       2947 2018-05-13 12:12 training/MovieTbl/part-m-00003
...
```

[training@localhost Desktop]$ hadoop fs -cat Movies/part-m-00000

```
[training@localhost Desktop]$ hadoop fs -cat  Movies/part-m-00000
```

[실습 1-2] training database안에 cityByCountry의 city, lat, lng를 HDFS 시스템으로 다른 디렉터리로 training/cityByCountry1 로 임포트 해 보자.

07 HDFS에서 RDBMS로 데이터 보내기

```
sqoop export \
--connect jdbc:mysql://localhost/training \
--username root \
--password 1234 \
--table sample_tbl \
--export-dir training/MovieTbl
```

에러 발생 한다면.

```
18/05/13 11:44:19 ERROR manager.SqlManager: Error executing statement: com.mysql.jdbc.exceptions.jdbc4.MySQLSyntaxErrorException: Table 'training.sample_tbl' doesn't exist
com.mysql.jdbc.exceptions.jdbc4.MySQLSyntaxErrorException: Table 'training.sample_tbl' doesn't exist
    at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
```

```
CREATE TABLE sample_tbl  (
    movieid int(11),
    action       tinyint(4)
);
```

[만약 잘못되어 삭제할때는 Drop table 이용]

```
Drop TABLE sample_tbl ;
```

[결과 확인]

```
mysql> show tables;
+----------------------+
| Tables_in_training |
+--------------------+
| Movies             |
| bible_freq         |
| cityByCountry      |
| countries          |
| sample_tbl         |
| shake_freq         |
+--------------------+
6 rows in set (0.00 sec)


mysql> select * from sample_tbl;
+---------+--------+
| movieid | action |
+---------+--------+
|       1 |      0 |
|       2 |      1 |
|       3 |      0 |
|       4 |      1 |
|       5 |      0 |
|       6 |      0 |
|       7 |      0 |
|       8 |      0 |
.....
|     412 |      0 |
|     413 |      0 |
|     414 |      0 |
|     415 |      0 |
|     416 |      0 |
|     417 |      0 |
|     418 |      0 |
|     419 |      0 |
|     420 |      0 |
|     421 |      0 |
+---------+--------+
421 rows in set (0.00 sec)
```

[실습 1-3] 도전과제

cityByCountry1의 HDFS의 파일을 sample_tbl2의 테이블로 옮겨보자.

### REF

(1) 하둡과 빅데이터 분석 실무

### Last : 2019.02.13