1일차. 아파치 스쿱을 통한 테이블 수집 실습 - Apache Sqoop

> *아파치 스쿱을 통해 다양한 수집 예제를 실습합니다*

- 목차

아파치 스쿱을 통한 테이블 수집

1. 스쿱 도커 기동 확인

- 최신 버전 코드를 내려 받습니다

```
cd /home/ubuntu/work/data-engineer-intermediate-training
git pull
```

- 아래의 명령을 확인합니다

```
docker ps --filter name=sqoop
```

- 기동되지 않았다면 Network, MySQL, Sqoop 순서대로 생성합니다

```
cd /home/ubuntu/work/data-engineer-intermediate-training/day1/sbin
./docker-create-network.sh
./docker-run-mysql.sh
./docker-run-sqoop.sh
```

아파치 스쿱 테이블 수집 예제 테이블

- 이전에 생성된 가비지 정보들을 모두 삭제하기 위해 하둡 및 MySQL 삭제 작업을 수행합니다 (처음이라면 스킵합니다)
  - userid: user, password: pass

```
bash>
./hadoop.sh fs -rm -r /user/sqoop
```

```
docker exec -it mysql mysql -uuser -p
mysql>
use testdb;
drop table inc_table;
drop table users;
drop table seoul_popular_stg;
drop table seoul_popular_exp;
```

- 로컬 환경에서 서울인기여행(testdb.seoul_popular_trip) 테이블을 수집합니다

```
cd /home/ubuntu/work/data-engineer-intermediate-training/day1/sbin
docker exec -it sqoop sqoop import -jt local -fs local -m 1 --connect
jdbc:mysql://mysql:3306/testdb --username user --password pass --table
seoul_popular_trip --target-dir /tmp/sqoop/seoul_popular_trip
docker exec -it sqoop ls /tmp/sqoop/seoul_popular_trip
docker exec -it sqoop cat /tmp/sqoop/seoul_popular_trip/part-m-00000
```

- 클러스터 환경에서 서울인기여행(testdb.seoul_popular_trip) 테이블을 수집합니다

```
docker exec -it sqoop sqoop import -m 1 --connect
jdbc:mysql://mysql:3306/testdb --username user --password pass --table
seoul_popular_trip --target-dir /user/sqoop/target/seoul_popular_trip
docker exec -it sqoop hadoop fs -ls /user/sqoop/target/seoul_popular_trip
docker exec -it sqoop hadoop fs -cat
/user/sqoop/target/seoul_popular_trip/part-m-00000
```

유형별 테이블 수집

- 하나의 테이블을 4개의 맵작업으로 병렬 수행 (sqoop-import.sh 반복적인 "접속정보, 계정, 패스워드"를 저장해두고 편리하기 쓰기 위한 배시스크립트입니다)

```
./sqoop-import.sh -m 4 --split-by id --table seoul_popular_trip --target-dir
/user/sqoop/target/seoul_popular_trip_v1 --fields-terminated-by '\t'
```

- 위의 sqoop-import.sh 명령 대신 아래와 같이 직접 모두 입력해도 됩니다

```
docker exec -it sqoop sqoop import -m 4 --split-by id --connect
jdbc:mysql://mysql:3306/testdb --table seoul_popular_trip --target-dir
/user/sqoop/target/seoul_popular_trip_v2 --fields-terminated-by '\t' --delete-
target-dir --verbose --username user --password pass
```

파티션 테이블 수집

- 하나의 테이블을 조건에 따라 분리해서 저장히기 위해 id 값의 범위를 확인해 봅니다 (Min(id), Max(id))

```
./sqoop-eval.sh "select min(id), max(id) from seoul_popular_trip"
./sqoop-eval.sh "select count(1) from seoul_popular_trip"
```

- 테이블을 파티션 단위로 저장하기 위해서는 루트 경로를 생성해 두어야만 합니다 (그래야 하위에 key=value 형식의 경로로 저장할 수 있습니다)

```
./hadoop.sh fs -mkdir -p /user/sqoop/target/seoul_popular_partition
```

- 확인한 값의 범위를 이용하여 조건을 달리하여 하나의 테이블을 3번으로 나누어 수집을 수행합니다

```
docker exec -it sqoop sqoop import --connect jdbc:mysql://mysql:3306/testdb --
username user --password pass --delete-target-dir -m 1 --table
seoul_popular_trip --where "id < 10000" --target-dir
/user/sqoop/target/seoul_popular_partition/part=0
docker exec -it sqoop sqoop import --connect jdbc:mysql://mysql:3306/testdb --
username user --password pass --delete-target-dir -m 1 --table
seoul_popular_trip --where "id > 10001 and id < 20000" --target-dir
/user/sqoop/target/seoul_popular_partition/part=10000
docker exec -it sqoop sqoop import --connect jdbc:mysql://mysql:3306/testdb --
```

```
username user --password pass --delete-target-dir -m 1 --table
seoul_popular_trip --where "id > 20001" --target-dir
/user/sqoop/target/seoul_popular_partition/part=20000
```

- 테이블 수집이 정상적으로 수행 되었는지 하둡 명령어를 통해 확인해 봅니다

```
./hadoop.sh fs -ls /user/sqoop/target/seoul_popular_partition
```

증분 테이블 수집

- 증분 테이블 수집을 위해 MySQL 접속 후, 예제 테이블을 생성합니다 (inc_table)
  - userid: user, password: pass

```
docker exec -it mysql mysql -uuser -p
create table inc_table (id int not null auto_increment, name
varchar(30), salary int, primary key (id));
insert into inc_table (name, salary) values ('suhyuk', 10000);
```

- 증분 테이블 초기 수집은 --last-value 값을 0으로 두고 수집합니다
  - 증분 테이블 수집 후 마지막에 --last-value 값이 1인 점을 확인해 둡니다 (다음 수집 시에 사용할 예정입니다)
  - 수집 이후에 하둡 명령어로 파티션 파일이 잘 생성되었는지 확인합니다

```
./sqoop-import.sh --table inc_table --incremental append --check-column
id --last-value 0 --target-dir /user/sqoop/target/seoul_popular_inc
./hadoop.sh fs -ls /user/sqoop/target/seoul_popular_inc
./hadoop.sh fs -cat /user/sqoop/target/seoul_popular_inc/part-m-00000
```

- 초기 수집 이후에 데이터가 추가되었(증분)다는 것을 테스트하기 위해 데이터를 추가합니다

```
docker exec -it mysql mysql -uuser -p
insert into inc_table (name, salary) values ('psyoblade', 20000);
```

- 증분 테이블 수집을 위해 이전 --last-value 1 을 입력하고 다시 수집합니다

```
./sqoop-import.sh --table inc_table --incremental append --check-column id --
last-value 1 --target-dir /user/sqoop/target/seoul_popular_inc
```

- 수집 된 테이블의 최종 결과 테이블에 파티션 파일이 어떻게 생성되고 있는지 확인합니다

```
./hadoop.sh fs -ls /user/sqoop/target/seoul_popular_inc
./hadoop.sh fs -cat /user/sqoop/target/seoul_popular_inc/part-m-00001
```

모든 테이블 수집

- 모든 테이블 수집을 위한 데이터베이스(testdb) 경로를 하나 생성합니다

```
./hadoop.sh fs -mkdir /user/sqoop/target/testdb
```

- 생성된 경로를 루트로 하위로 모든 테이블을 수집합니다 (반드시 --autoreset-to-one-mapper 를 지정해 주어야 primary key 가 없는 경우에도 -m 1 으로 수집이 됩니다)

```
docker exec -it sqoop sqoop import-all-tables --connect
jdbc:mysql://mysql:3306/testdb --autoreset-to-one-mapper --warehouse-dir
/user/sqoop/target/testdb --username user --password pass
```

아파치 스쿱을 통한 테이블 적재

새로운 테이블을 생성하고 적재

- 테이블 익스포트를 위해 적재 테이블(seoul_popular_exp)을 생성합니다
    - userid: user, password: pass

```
bash>
docker exec -it mysql mysql -uuser -p
```

mysql> create table testdb.seoul_popular_exp (category int not null, id int not null, name varchar(100), address varchar(100), naddress varchar(100), tel varchar(20), tag varchar(500)) character set utf8 collate utf8_general_ci;

bash> ./sqoop-eval.sh "show tables" ./sqoop-export.sh -m 1 --table seoul_popular_exp --export-dir /user/sqoop/target/seoul_popular_trip

```
* 위의 익스포트 명령어를 수행하면 오류가 발생하는데 리소스매니저(http://student#.lgebigdata.com:8088) 사이트를 통해 디버깅을 해봅니다
  * application\_number 링크를 클릭하고 logs 경로를 클릭하면 http://9e48393c5f39:8042/ 와 같이 docker-container 아이로 redirect 되는데 이 값을 student#.lgebigdata.com 값으로 변경해 주면 디버깅이 가능합니다
  * 해당 로그 페이지에서 "syslog : Total file length is 49301 bytes." 링크를 클릭하고 "here" 링크를 클릭하면 전체 로그를 한 번에 확인할 수 있습니다
  * 초기에 수집한 테이블의 경우 콤마를 기준으로 수집했는데, 필드에 콤마(,)가 들어가 있어서 export 시에 필드의 개수가 맞지 않는다는 오류를 확인합니다
* 탭을 구분자로 테이블 임포트를 다시 수행합니다
```bash
./sqoop-import.sh -m 1 --table seoul_popular_trip --fields-terminated-by '\t' --delete-target-dir --target-dir /user/sqoop/target/seoul_popular_exp
```

- 탭 구분자로 익스포트된 경로의 파일을 이용하여 다시 익스포트를 수행합니다

```
./sqoop-export.sh -m 1 --table seoul_popular_exp --fields-terminated-by '\t' --export-dir /user/sqoop/target/seoul_popular_exp
./sqoop-eval.sh "select count(1) from seoul_popular_exp"
```

테이블 익스포트 시에 스테이징을 통한 적재

- 적재 시에 도중에 실패하는 경우 대상 테이블에 일부 데이터만 적재되는 경우가 있는데 이러한 경우를 회피하기 위해 스테이징 테이블을 사용할 수 있습니다.
- 스테이징 테이블은 원본 테이블을 그대로 두고 별도의 스테이징 테이블에 적재 후 완전 export 가 성공하면 원본 테이블을 clear 후 적재합니다
- 아래와 같이 임시 스테이징 테이블을 동일한 스키마로 생성하고 익스포트를 수행합니다
    - userid: user, password: pass

```
bash>
docker exec -it mysql mysql -uuser -p
```

mysql> create table testdb.seoul_popular_stg (category int not null, id int not null, name varchar(100), address varchar(100), naddress varchar(100), tel varchar(20), tag varchar(500)) character set utf8 collate
```

utf8_general_ci;

```
* 이미 적재된 테이블에 다시 적재하는 경우는 중복 데이터가 생성되므로  삭제 혹은 truncate 는 수작업으로 수행되어야만 합니다
```

bash> ./sqoop-eval.sh "truncate testdb.seoul_popular_exp" ./sqoop-export.sh -m 4 --table seoul_popular_exp --fields-terminated-by '\t' --staging-table seoul_popular_stg --clear-staging-table --export-dir /user/sqoop/target/seoul_popular_exp

```
### 컨테이너 정리
* 테스트 작업이 완료되었으므로 모든 컨테이너를 종료합니다 (한번에 실행중인 모든 컨테이너를 종료합니다)
```bash
docker stop `docker ps -q`
docker rm `docker ps -aq`
docker rm -f `docker ps -aq` # 위의 두 가지 작업을 한 번에 수행
```
```