하이브 테이블 관리

1 하이브 테이블

① 데이터를 하이브 테이블로 가져오면?

다양한 도구들을 사용해 데이터를 처리할 수 있음

② 하이브가 지원하는 테이블 종류

내부 테이블

- 하이브가 관리하는 테이블
- 하이브 데이터 웨어하우스에 저장됨(/hive/warehouse/)
- 내부 테이블을 삭제하면 하이브의 테이블 메타정의와 테이블에 들어 있는 모든 데이터도 같이 삭제됨
- ORC 같은 최적화된 형식으로 저장될 수 있어 비교적 빠른 성능을 낼 수 있음

하이브 테이블 관리

1 하이브 테이블

2 하이브가 지원하는 테이블 종류

외부 테이블

- 하이브가 직접 관리하지 않음
- 하이브의 메타 정의만 사용해 원시 형태로 저장된 텍스트 데이터에 접근함
- 외부 테이블의 데이터를 삭제하면 하이브의 테이블 메타 정의만 삭제되고 실제 데이터는 그대로 유지됨
- 해당 데이터가 하이브 외부에 적재되어 있거나 테이블이 삭제되더라도 원본 데이터가 본래 위치에 남아야 할 때 주로 사용함

하이브 테이블 관리

2 데이터를 하이브 테이블로 가져오기

① CSV 파일을 하이브 테이블로 가져오기

1) 샘플 CSV 파일(names.csv)

```
10, Andrew, Manager, DE, PC
11, Arun, Manager, NJ, PC
12, Harish, Sales, NJ, MAC
13, Robert, Manager, PA, MAC
14, Laura, Engineer, PA, MAC
```

2) 샘플 CSV 파일을 HDFS에 복사함

```
$ hdfs dfs -mkdir names
$ hdfs dfs -put names.csv names
```

3) 외부 하이브 테이블 생성하기

- 필드 사이의 구분자(delimiter)는 쉼표(,)를 사용했음
- LOCATION 문 : 테이블이 사용할 입력 파일의 경로를 지정함

```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS Names_text(
    > EmployeeID INT, FirstName STRING, Title STRING,
    > State STRING, Laptop STRING)
    > COMMENT 'Employee Names'
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE
    > LOCATION '/user/username/names';
```

하이브 테이블 관리

2 데이터를 하이브 테이블로 가져오기

① CSV 파일을 하이브 테이블로 가져오기

4) 외부 하이브 테이블의 데이터 확인하기

```
hive> Select * from Names_text limit 5;
OK
10      Andrew   Manager   DE      PC
11      Arun      Manager   NJ      PC
12      Harish    Sales      NJ      MAC
13      Robert    Manager   PA      MAC
14      Laura     Engineer   PA      MAC
```

5) ORC 형식을 사용하는 내부 하이브 테이블 생성

- STORED AS : 내부 테이블의 파일 형식을 지정할 수 있음

```
hive> CREATE TABLE IF NOT EXISTS Names(
    > EmployeeID INT, FirstName STRING, Title STRING,
    > State STRING, Laptop STRING)
    > COMMENT 'Employee Names'
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > STORED AS ORC;
```

하이브 테이블 관리

2 데이터를 하이브 테이블로 가져오기

① CSV 파일을 하이브 테이블로 가져오기

6) 데이터 형식

텍스트 파일	모든 데이터가 유니코드 표준을 사용한 원시 텍스트로 저장됨
시퀀스 파일	데이터가 이진 키-값 쌍으로 저장됨
RC 파일	모든 데이터가 로우(행, row) 기반 최적화 대신 컬럼(열, column) 기반 최적화 형식으로 저장됨
ORC 형식	Optimized Row Columnar의 약자로 하이브 성능을 크게 높임
Parquet 형식	하이브, 임팔라(Impala), 피그 등, 다양한 하둡 도구와 호환되는 컬럼 기반 형식

하이브 테이블 관리

2 데이터를 하이브 테이블로 가져오기

① CSV 파일을 하이브 테이블로 가져오기

7) 외부 하이브 테이블의 데이터를 내부 하이브 테이블로 복사하기

```
hive> INSERT OVERWRITE TABLE Names SELECT * FROM Names_text
```

8) 내부 하이브 테이블의 내용 확인하기

```
hive> Select * from Names limit 5;
OK
10      Andrew   Manager   DE      PC
11      Arun      Manager   NJ      PC
12      Harish    Sales      NJ      MAC
13      Robert    Manager   PA      MAC
14      Laura     Engineer   PA      MAC
```

하이브 테이블 관리

3 하이브의 테이블 파티션

① 파티션 기능 지원

- 대용량 테이블을 논리적으로 나누어 효율적인 쿼리가 가능하도록 함
- State 필드에 파티션이 적용된 내부 테이블 생성하기

```
hive> CREATE TABLE IF NOT EXISTS Names_part(
    > EmployeeID INT,
    > FirstName STRING,
    > Title STRING,
    > Laptop STRING)
    > COMMENT 'Employee Names partitioned by state'
    > PARTITIONED BY (State STRING)
    > STORED AS ORC;
```

- 펜실베니아주 출신 직원들의 데이터를 외부 테이블에서 내부 테이블로 복사하기

```
hive> INSERT INTO TABLE Names_part PARTITION(state='PA')
    > SELECT EmployeeID, FirstName, Title, Laptop FROM Names_text WHERE state='PA';
```