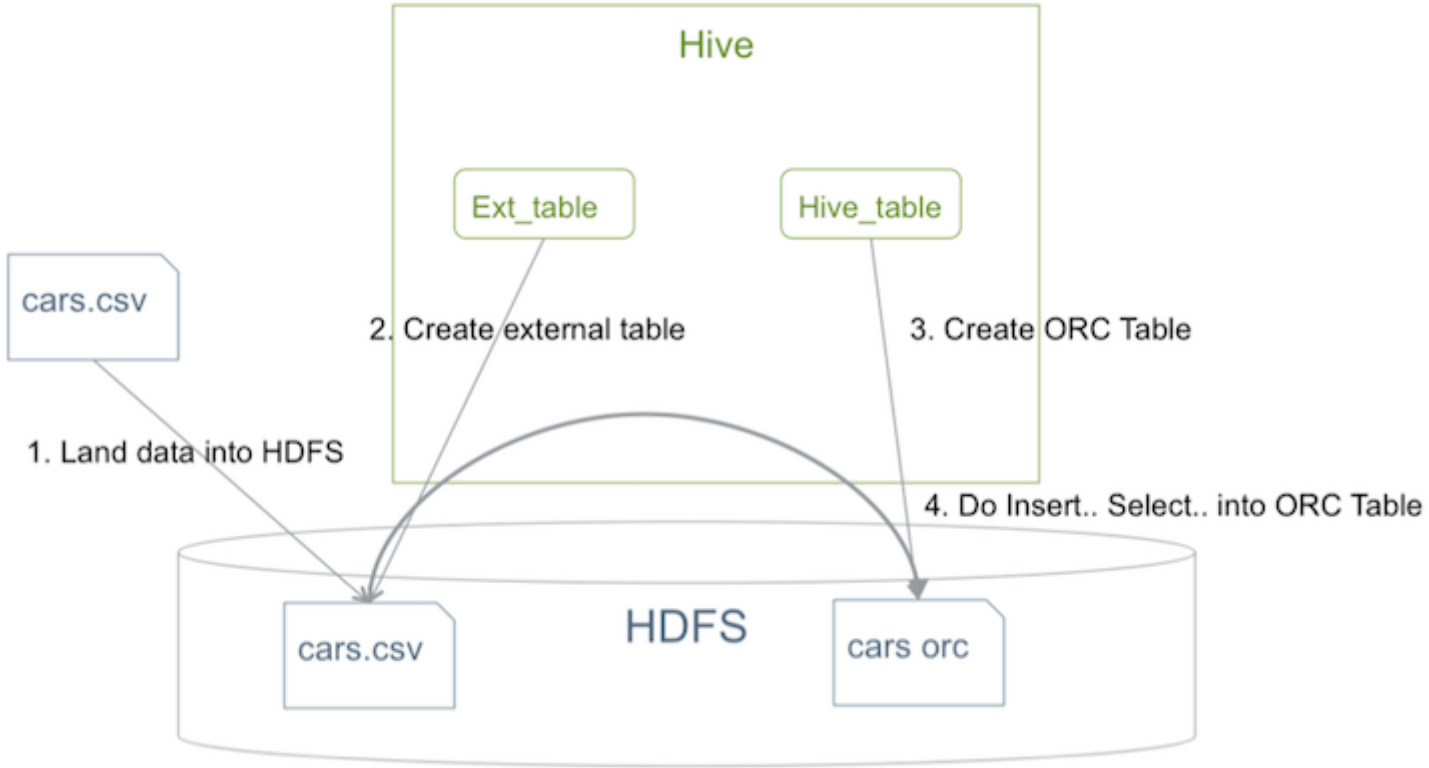
3.1. Moving Data from HDFS to Hive Using an External Table

This is the most common way to move data into Hive when the ORC file format is required as the target data format. Then Hive can be used to perform a fast parallel and distributed conversion of your data into ORC. The process is shown in the following diagram:

Figure 1.1. Example: Moving .CSV Data into Hive



Moving .CSV Data into Hive

The following steps describe moving .CSV data into Hive using the method illustrated in the above diagram with command-line operations.

1. **Move .CSV data into HDFS**:

   a. The following is a .CSV file which contains a header line that describes the fields and subsequent lines that contain the data:

```
[<username>@cn105-10 ~]$ head cars.csv
Name,Miles_per_Gallon,Cylinders,Displacement,Horsepower,Weight_in_lbs,Acceleration,Year,Origin
"chevrolet chevelle malibu",18,8,307,130,3504,12,1970-01-01,A
"buick skylark 320",15,8,350,165,3693,11.5,1970-01-01,A
"plymouth satellite",18,8,318,150,3436,11,1970-01-01,A
"amc rebel sst",16,8,304,150,3433,12,1970-01-01,A
"ford torino",17,8,302,140,3449,10.5,1970-01-01,A
...
[<username>@cn105-10 ~]$
```

   <username> is the user who is performing the operation. To test this example, run with a user from your environment.

   b. First, use the following command to remove the header line from the file because it is not part of the data for the table:

```
[<username>@cn105-10 ~]$ sed -i 1d cars.csv
```

   c. Move the data to HDFS:

```
[<username>@cn105-10 ~]$ hdfs dfs -copyFromLocal cars.csv /user/<username>/visdata
[<username>@cn105-10 ~]$ hdfs dfs -ls /user/<username>/visdata
Found 1 items
-rwxrwxrwx   3 <username> hdfs      22100 2015-08-12 16:16 /user/<username>/visdata/cars.csv
```

2. **Create an external table**.

   An *external table* is a table for which Hive does not manage storage. If you delete an external table, only the definition in Hive is deleted. The data remains. An *internal table* is a table that Hive manages. If you delete an internal table, both the definition in Hive *and* the data are deleted.

   The following command creates an external table:

```
CREATE EXTERNAL TABLE IF NOT EXISTS Cars(
        Name STRING,
        Miles_per_Gallon INT,
        Cylinders INT,
```

```
        Displacement INT,
        Horsepower INT,
        Weight_in_lbs INT,
        Acceleration DECIMAL,
        Year DATE,
        Origin CHAR(1))
    COMMENT 'Data about cars from a public database'
    ROW FORMAT DELIMITED
    FIELDS TERMINATED BY ','
    STORED AS TEXTFILE
    location '/user/<username>/visdata';
```

3. **Create the ORC table**.

   Now, create a table that is managed by Hive with the following command:

```
CREATE TABLE IF NOT EXISTS mycars(
        Name STRING,
        Miles_per_Gallon INT,
        Cylinders INT,
        Displacement INT,
        Horsepower INT,
        Weight_in_lbs INT,
        Acceleration DECIMAL,
        Year DATE,
        Origin CHAR(1))
    COMMENT 'Data about cars from a public database'
    ROW FORMAT DELIMITED
    FIELDS TERMINATED BY ','
    STORED AS ORC;
```

4. **Insert the data from the external table to the Hive ORC table**.

   Now, use an SQL statement to move the data from the external table that you created in Step 2 to the Hive-managed ORC table that you created in Step 3:

```
INSERT OVERWRITE TABLE mycars SELECT * FROM cars;
```

Note

Using Hive to convert an external table into an ORC file format is very efficient because the conversion is a parallel and distributed action, and no standalone ORC conversion tool is necessary.

5. **Verify that you imported the data into the ORC-formatted table correctly**:

```
hive> select * from mycars limit 3;
OK
"chevrolet chevelle malibu"     18      8       307     130     3504    12      1970-01-01      A
"buick skylark 320"     15      8       350     165     3693    12      1970-01-01      A
"plymouth satellite"    18      8       318     150     3436    11      1970-01-01      A
Time taken: 0.144 seconds, Fetched: 3 row(s)
```