NETFLIX USE CASE:

Let us assume we have Netflix users data from India and USA.

The data is as follows: users_data.csv

Note : This will be a CSV file for better understanding. I created it as a table.

USER_ID	USER_NAME	EMAIL	STATUS	Country	State
1	Virat	a@b.com	Active	USA	California
2	Kohli	c@d.com	Active	USA	Florida
3	Mahi	a@b.com	Active	USA	Texas
4	Dravid	dba@b.com	Inactive	India	Karnataka
5	Vvs	abd@c.com	Active	India	Telangana
6	Sourav	bad@v.com	Active	India	Bengal
7	Sachin	vvs@k.com	Inactive	India	Mumbai
8	Khan	kh@z.com	Active	USA	Arizona
9	Patel	pa@m.com	Inactive	India	Delhi
10	Gautam	ga@g.com	Active	USA	Colorado

The data is in some source location and we load this data from source to HDFS.

Load data from source machine to HDFS:

Step 1 : Create a folder called data and inside create another folder called users.

```
hadoop fs -mkdir /data
hadoop fs -mkdir /data/users
```

Step 2: Now into this folder we need to load the data.

```
hadoop fs -put users_data.csv /data/users
```

Now in HDFS /data/users we have a file called users_data.csv

Now we need to load this data into HIVE.

As we can see we have a very small amount of data. But in the real world scenario it can be very huge and may go up to more than 1TB.

So for Optimization we will do a partition. In the real world we can't do static partitioning, we will do dynamic partitioning.

Now first we create a non-partition table by loading data from our HDFS location.

Step 3 : Load data from HDFS location

CREATE external TABLE non_part_users(user_id int, user_name string, email string, status string, country string, state string) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION '**/data/users**';

Step 4 : Create a Partitioned table Schema

CREATE TABLE **partition_users**(user_id int, user_name string, email string, status string) **partitioned by** (country string, state string);

Step 5: Load data into the partition table from the non-partition table.

set hive.exec.dynamic.partition.mode=nonstrict;

insert overwrite table partition_users partition(user_id,user_name,email,status) select user_id,user_name,email,status,country,state from non_part_users;

Now when we do this the data from table will be stored in hdfs as :

/data/hive/warehouse/db/partition_users/USA/California
/data/hive/warehouse/db/partition_users/USA/Florida
/data/hive/warehouse/db/partition_users/USA/Texas
/data/hive/warehouse/db/partition_users/USA/Arizona
/data/hive/warehouse/db/partition_users/USA/Colorado

Similarly the folders will be created for India and its corresponding States.

When we write:

SELECT * FROM partition_users WHERE (country = 'USA' AND state = 'California') AND (status = 'Active');

This will not iterate through entire data, as we created this as partition table it will know that we have partitioned this on country and state it will go hive default hdfs directory and will

check if the particular db directory, table directory, country directory and state directory is there or not and will return the data.

Now let us assume we subscribe to Netflix without email and some users use the subscribed email id for watching Netflix and now Netflix wants to delete those users from both USA and INDIA in all states.

WITH CTE AS (SELECT *, ROW_NUMBER() OVER(PARTITION BY email ORDER BY user_id) AS RNK FROM partition_users WHERE country = ‘USA’ or Country = ‘INDIA’)
DELETE * FROM CTE WHERE RNK > 1;

This will go to /data/hive/warehouse and check for India and USA directories and get data from all the directories having India and USA and delete the duplicate users.