|  |  |
| --- | --- |
| A picture of a winding road and trees  MID tERM pROJECT  Chicago Taxi Data | Abstract  Marketix, a Market Research firm, has hired you as a big data engineer. You are asked to prepare the datafor performing market research on the trends and patterns of Taxi services in Chicago. The report shouldequip to serve the following target audience:• Executives and Leadership team from any of the existing taxi service provider or aggregator thatoperates already in the region.• Executives and Leadership team from any of the taxi service providers operating in otherregions, who are trying to enter the market for the first time.• Any potential Entrepreneur or investor who is trying to invest in the business of taxi service oraggregator service.The outcomes and findings must cater to the needs of diversified target audience, who have differentgoals in their mind.  Debmalya Ray  Project Submission |

**Database Creation for the Mid – Term Project**

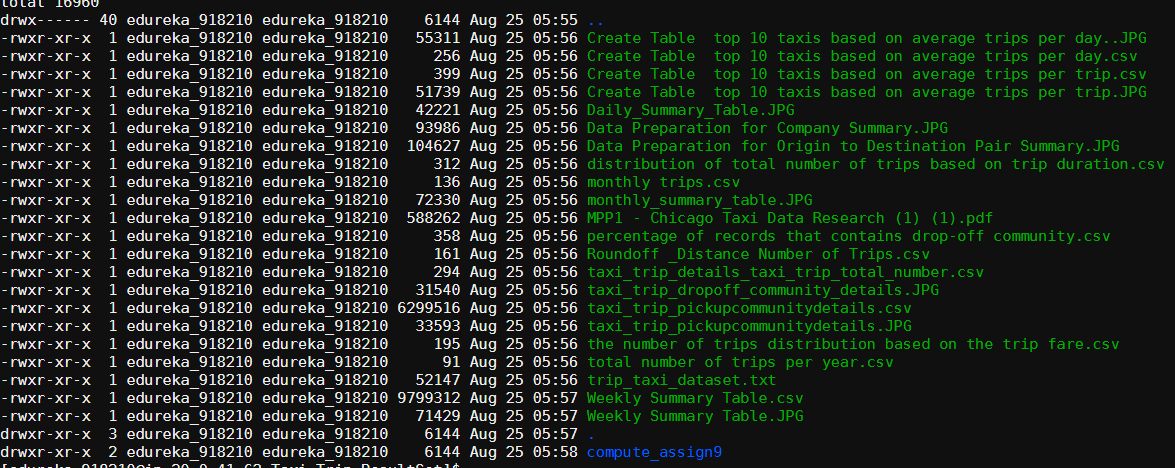
create database taxi\_trip\_database

use taxi\_trip\_database

**Raw File Location – taxi trip dataset**

http://bdlabs.edureka.co:50003/filebrowser/#/bigdatapgp/common\_folder/midproject/taxi\_trip\_dataset

**RESULT DATA**



[edureka\_918210@ip-20-0-41-164 ~]$ hadoop fs -ls /bigdatapgp/common\_folder/midproject/taxi\_trip\_dataset

Found 1 items

-rw-r--r-- 3 evaluationuser01 supergroup 2589084507 2020-05-17 13:54 /bigdatapgp/common\_folder/midproject/taxi\_trip\_dataset/taxi\_trip.csv

[edureka\_918210@ip-20-0-41-164 ~]$

hadoop fs -ls /user/edureka\_918210

hadoop fs -mkdir /user/edureka\_918210/TaxiTrip

hadoop fs -put taxi\_trip.csv /user/edureka\_918210/TaxiTrip

**Cleaning, Loading and Pre-processing Data**

**1. Creating taxi\_details\_str table & loading the whole row as a string**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_details\_str (

taxi\_trip\_details\_str String)

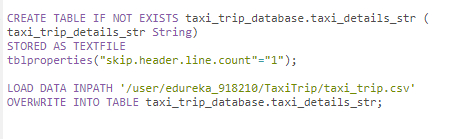
STORED AS TEXTFILE

tblproperties("skip.header.line.count"="1");

LOAD DATA INPATH '/user/edureka\_918210/TaxiTrip/taxi\_trip.csv'

OVERWRITE INTO TABLE taxi\_trip\_database.taxi\_details\_str;

**Table - taxi\_trip\_database.taxi\_details\_str**



**2. Splitting one column in multiple columns and creating taxi\_trip\_details table**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_details

AS

select split(taxi\_trip\_details\_str, ",")[0] as trip\_id,

split(taxi\_trip\_details\_str, ",")[1] as taxi\_id,

split(taxi\_trip\_details\_str, ",")[2] as trip\_start\_time,

split(taxi\_trip\_details\_str, ",")[3] as trip\_end\_time,

cast(split(taxi\_trip\_details\_str, ",")[4] as int) as trip\_seconds,

cast(split(taxi\_trip\_details\_str, ",")[5] as float) as trip\_miles,

cast(split(taxi\_trip\_details\_str, ",")[6] as bigint) as pickup\_tract,

cast(split(taxi\_trip\_details\_str, ",")[7] as bigint) as dropoff\_tract,

cast(split(taxi\_trip\_details\_str, ",")[8] as tinyint) as pickup\_community,

cast(split(taxi\_trip\_details\_str, ",")[9] as tinyint) as dropoff\_community,

cast(split(taxi\_trip\_details\_str, ",")[10] as float) as trip\_fare,

cast(split(taxi\_trip\_details\_str, ",")[11] as float) as tip\_amt,

cast(split(taxi\_trip\_details\_str, ",")[12] as float) as toll\_amt,

cast(split(taxi\_trip\_details\_str, ",")[13] as float) as extra\_amt,

cast(split(taxi\_trip\_details\_str, ",")[14] as float) as trip\_total\_amt,

split(taxi\_trip\_details\_str, ",")[15] as payment\_type,

split(taxi\_trip\_details\_str, ",")[16] as company,

cast(split(taxi\_trip\_details\_str, ",")[17] as double) as pickup\_latitude,

cast(split(taxi\_trip\_details\_str, ",")[18] as double) as pickup\_longitude,

split(taxi\_trip\_details\_str, ",")[19] as pickup\_location,

cast(split(taxi\_trip\_details\_str, ",")[20] as double) as dropoff\_latitude,

cast(split(taxi\_trip\_details\_str, ",")[21] as double) as dropoff\_longitude,

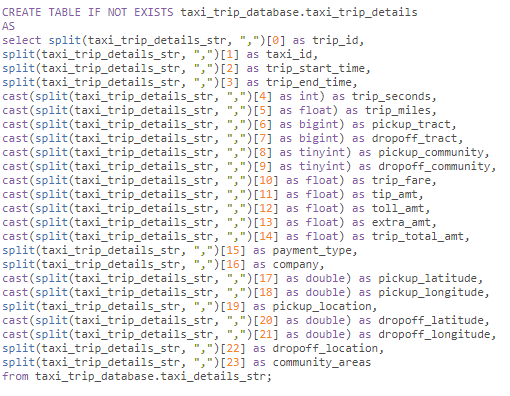
split(taxi\_trip\_details\_str, ",")[22] as dropoff\_location,

split(taxi\_trip\_details\_str, ",")[23] as community\_areas

from

taxi\_trip\_database.taxi\_details\_str;

**Table – taxi\_trip\_details**



select split(taxi\_trip\_details\_str, ",")[0] as trip\_id,

split(taxi\_trip\_details\_str, ",")[1] as taxi\_id,

split(taxi\_trip\_details\_str, ",")[2] as trip\_start\_time,

split(taxi\_trip\_details\_str, ",")[3] as trip\_end\_time,

cast(split(taxi\_trip\_details\_str, ",")[4] as int) as trip\_seconds,

cast(split(taxi\_trip\_details\_str, ",")[12] as float) as toll\_amt

from

taxi\_trip\_database.taxi\_details\_str limit 5;

**3. Numerical Mapping of taxi\_id & trip\_id to reduce the data volume**

select count(distinct taxi\_id) from taxi\_trip\_database.taxi\_trip\_details;

**RESULT :::::::::::::::::::::: 5396**

There are only 8287 distinct values. But as we can see the uuid is a really large:

c1305c4490085b703eed20e95ab0c479c954ae3735a963578627d563fadbf2cc859e12ebcd12c4b8f34b7eb

2d6c4782b17b56ba8ddfe896fff5763105e81e050

Keeping this column would result in significant performance implications.

Hence, we can either

• Drop this field

• Or move it to a separate table and create a numerical mapping of the same into the master table.

This would reduce the data volume significantly as we are going to store only around 8K distinct

values of taxi ids, which are repeating for over 90M trips.

Second option makes sense, as it would help us perform per taxi-wise analysis.

Similarly, we can drop the trip\_id, which is a uuid and create a row\_number which can be an integer in its

place.

**3.a. Creating a separate table with distinct taxi\_id values**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_id\_mapping

AS

select distinct taxi\_id from taxi\_trip\_database.taxi\_trip\_details

describe taxi\_trip\_database.taxi\_id\_mapping

select \* from taxi\_trip\_database.taxi\_id\_mapping limit 5;

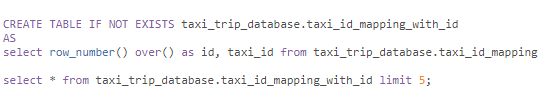
CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_id\_mapping\_with\_id

AS

select row\_number() over() as id, taxi\_id from taxi\_trip\_database.taxi\_id\_mapping

select \* from taxi\_trip\_database.taxi\_id\_mapping\_with\_id limit 5;

**Table - taxi\_id\_mapping\_with\_id**



**3.b. Joining taxi\_id\_mapping\_with\_id table to master table to replace the current taxi\_id i.e uuid with a**

**numerical id.**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_details\_taxi\_id\_removed

AS

SELECT

trip\_id,

id as taxi\_id\_int,

trip\_start\_time,

trip\_end\_time,

trip\_seconds,

trip\_miles,

pickup\_tract,

dropoff\_tract,

pickup\_community,

dropoff\_community,

trip\_fare,

tip\_amt,

toll\_amt,

extra\_amt,

trip\_total\_amt,

payment\_type,

company,

pickup\_latitude,

pickup\_longitude,

pickup\_location,

dropoff\_latitude,

dropoff\_longitude,

dropoff\_location,

community\_areas

from

taxi\_trip\_database.taxi\_trip\_details as a

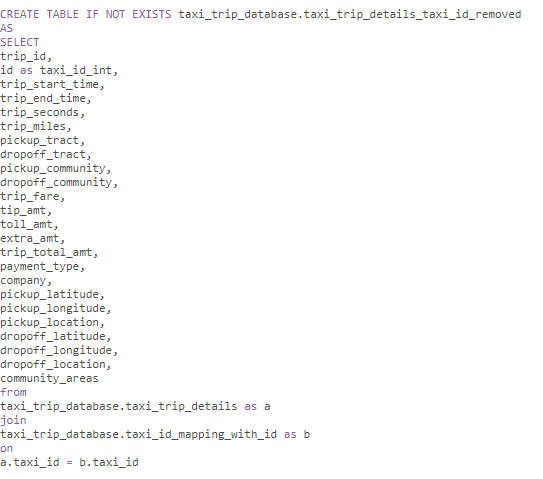
join

taxi\_trip\_database.taxi\_id\_mapping\_with\_id as b

on

a.taxi\_id = b.taxi\_id

**Table - taxi\_trip\_database.taxi\_trip\_details\_taxi\_id\_removed**



**3.c. Removing trip\_id (uuid) and adding an int id instead**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_details\_taxi\_trip\_id\_removed

STORED AS ORC

AS

SELECT

row\_number() over() as trip\_id\_int,

taxi\_id\_int,

trip\_start\_time,

trip\_end\_time,

trip\_seconds,

trip\_miles,

pickup\_tract,

dropoff\_tract,

pickup\_community,

dropoff\_community,

trip\_fare,

tip\_amt,

toll\_amt,

extra\_amt,

trip\_total\_amt,

payment\_type,

company,

pickup\_latitude,

pickup\_longitude,

pickup\_location,

dropoff\_latitude,

dropoff\_longitude,

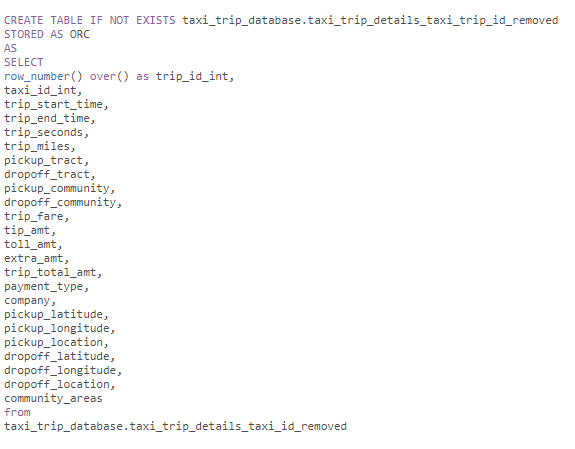
dropoff\_location,

community\_areas

from

taxi\_trip\_database.taxi\_trip\_details\_taxi\_id\_removed

**Table – taxi\_trip\_database.taxi\_trip\_details\_tai\_trip\_id\_removed**



**4. Cleaning up the temp tables**

drop table chicago\_taxis.taxi\_details\_str

drop table chicago\_taxis.taxi\_trip\_details

drop table chicago\_taxis.taxi\_trip\_details\_taxi\_id\_removed

**5. Casting date fields**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_details\_taxi\_trip\_id\_removed\_ts

STORED AS ORC

AS

SELECT

taxi\_id\_int,

trip\_start\_time,

trip\_end\_time,

to\_date(from\_unixtime(unix\_timestamp(split(trip\_start\_time, " ")[0], 'MM/dd/yyyy'), 'yyyy-MM-dd')) as

trip\_start\_date,

to\_date(from\_unixtime(unix\_timestamp(split(trip\_end\_time, " ")[0], 'MM/dd/yyyy'), 'yyyy-MM-dd')) as

trip\_end\_date,

trip\_seconds,

trip\_miles,

pickup\_tract,

dropoff\_tract,

pickup\_community,

dropoff\_community,

trip\_fare,

tip\_amt,

toll\_amt,

extra\_amt,

trip\_total\_amt,

payment\_type,

company,

pickup\_latitude,

pickup\_longitude,

pickup\_location,

dropoff\_latitude,

dropoff\_longitude,

dropoff\_location,

community\_areas

from

taxi\_trip\_database.taxi\_trip\_details\_taxi\_trip\_id\_removed

select \* from taxi\_trip\_database.taxi\_trip\_details\_taxi\_trip\_id\_removed\_ts

**6. Adding two fields for the trip start & end day of the week**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_details\_processed\_with\_dayofweek

AS

SELECT

\*,

from\_unixtime(unix\_timestamp(split(trip\_start\_time, " ")[0], 'MM/dd/yyyy'), 'u') as start\_dayofweek,

from\_unixtime(unix\_timestamp(split(trip\_end\_time, " ")[0], 'MM/dd/yyyy'), 'u') as end\_dayofweek

from

taxi\_trip\_database.taxi\_trip\_details\_taxi\_trip\_id\_removed

select \* from taxi\_trip\_database.taxi\_trip\_details\_processed\_with\_dayofweek limit 5

**7. Adding a weekend field to store whether a day is weekday or weekend**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded

STORED AS ORC

AS

SELECT

\*,

CASE

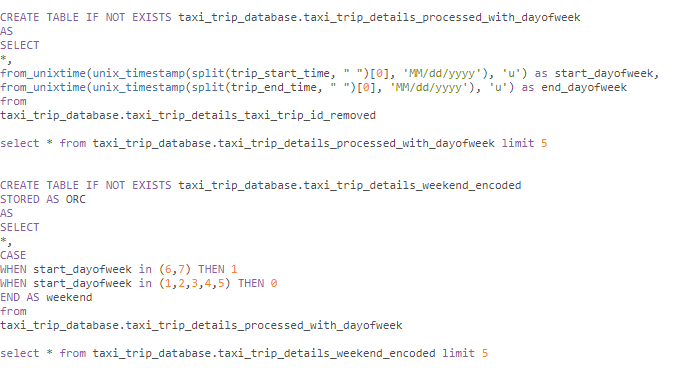
WHEN start\_dayofweek in (6,7) THEN 1

WHEN start\_dayofweek in (1,2,3,4,5) THEN 0

END AS weekend

from

taxi\_trip\_database.taxi\_trip\_details\_processed\_with\_dayofweek



select \* from taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded limit 5

**TABLE DESCRIPTION :**

hive> describe taxi\_trip\_database.taxi\_details\_str;

OK

taxi\_trip\_details\_str string

Time taken: 0.493 seconds, Fetched: 1 row(s)

> describe taxi\_trip\_database.taxi\_trip\_details;

OK

trip\_id string

taxi\_id string

trip\_start\_time string

trip\_end\_time string

trip\_seconds int

trip\_miles float

pickup\_tract bigint

dropoff\_tract bigint

pickup\_community tinyint

dropoff\_community tinyint

trip\_fare float

tip\_amt float

toll\_amt float

extra\_amt float

trip\_total\_amt float

payment\_type string

company string

pickup\_latitude double

pickup\_longitude double

pickup\_location string

dropoff\_latitude double

dropoff\_longitude double

dropoff\_location string

community\_areas string

Time taken: 0.068 seconds, Fetched: 24 row(s)

> describe taxi\_trip\_database.taxi\_id\_mapping

> ;

OK

taxi\_id string

> describe taxi\_trip\_database.taxi\_id\_mapping\_with\_id;

OK

id int

taxi\_id string

hive>

> describe taxi\_trip\_database.taxi\_trip\_details\_taxi\_id\_removed

> ;

OK

trip\_id string

taxi\_id\_int int

trip\_start\_time string

trip\_end\_time string

trip\_seconds int

trip\_miles float

pickup\_tract bigint

dropoff\_tract bigint

pickup\_community tinyint

dropoff\_community tinyint

trip\_fare float

tip\_amt float

toll\_amt float

extra\_amt float

trip\_total\_amt float

payment\_type string

company string

pickup\_latitude double

pickup\_longitude double

pickup\_location string

dropoff\_latitude double

dropoff\_longitude double

dropoff\_location string

community\_areas string

Time taken: 0.047 seconds, Fetched: 24 row(s)

hive>

> describe taxi\_trip\_database.taxi\_trip\_details\_taxi\_trip\_id\_removed\_ts;

OK

taxi\_id\_int int

trip\_start\_time string

trip\_end\_time string

trip\_start\_date string

trip\_end\_date string

trip\_seconds int

trip\_miles float

pickup\_tract bigint

dropoff\_tract bigint

pickup\_community tinyint

dropoff\_community tinyint

trip\_fare float

tip\_amt float

toll\_amt float

extra\_amt float

trip\_total\_amt float

payment\_type string

company string

pickup\_latitude double

pickup\_longitude double

pickup\_location string

dropoff\_latitude double

dropoff\_longitude double

dropoff\_location string

community\_areas string

Time taken: 0.052 seconds, Fetched: 25 row(s)

hive>

>

> describe taxi\_trip\_database.taxi\_trip\_details\_processed\_with\_dayofweek;

OK

trip\_id\_int int

taxi\_id\_int int

trip\_start\_time string

trip\_end\_time string

trip\_seconds int

trip\_miles float

pickup\_tract bigint

dropoff\_tract bigint

pickup\_community tinyint

dropoff\_community tinyint

trip\_fare float

tip\_amt float

toll\_amt float

extra\_amt float

trip\_total\_amt float

payment\_type string

company string

pickup\_latitude double

pickup\_longitude double

pickup\_location string

dropoff\_latitude double

dropoff\_longitude double

dropoff\_location string

community\_areas string

start\_dayofweek string

end\_dayofweek string

Time taken: 0.045 seconds, Fetched: 26 row(s)

hive>

> describe taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded;

OK

trip\_id\_int int

taxi\_id\_int int

trip\_start\_time string

trip\_end\_time string

trip\_seconds int

trip\_miles float

pickup\_tract bigint

dropoff\_tract bigint

pickup\_community tinyint

dropoff\_community tinyint

trip\_fare float

tip\_amt float

toll\_amt float

extra\_amt float

trip\_total\_amt float

payment\_type string

company string

pickup\_latitude double

pickup\_longitude double

pickup\_location string

dropoff\_latitude double

dropoff\_longitude double

dropoff\_location string

community\_areas string

start\_dayofweek string

end\_dayofweek string

weekend int

Time taken: 0.046 seconds, Fetched: 27 row(s)

hive>

**3.2.1 Data Summary**

**What are the total number of trips per year? Present the findings in the below format**.

select count(trip\_id\_int), cast(substr(trip\_end\_time, 7, 4) as int) from taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded group by cast(substr(trip\_end\_time, 7, 4) as int)



**Create the same summary for number of trips at monthly level. Present the findings in the below format.**

select count(trip\_id\_int), cast(substr(trip\_end\_time, 1, 2) as int) from taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded group by cast(substr(trip\_end\_time, 1, 2) as int)



**Calculate the percentage of records that contains drop-off community value. Excluding all the NULL records, find out the top 10 communities, where people travel to, based on the drop-off**

**community field and also find its percentage to the total number of trips. Present the findings in**

**the below format.**

select

sum(case when dropoff\_community is not null then 1 else 0 end) dropoff\_community,

(sum(case when dropoff\_tract is not null then 1 else 0 end)/count(trip\_id\_int))\*100 dropoff\_tract,

(sum(case when dropoff\_community is not null then 1 else 0 end)/count(dropoff\_community))\*100 dropoff\_community

from

taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded

group by dropoff\_community

order by dropoff\_community desc

limit 10;



**Create a table which contains the total number of trips for each drop-off community across each**

**year. Using the above table, find the top 10 records based on number of trips with year and**

**drop\_off community. Remove the null record while creating the table to remove inconsistencies.**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_details\_taxi\_trip\_total\_number

STORED AS ORC

AS

SELECT

dropoff\_community,

cast(substr(trip\_end\_time, 7, 4) as int) trip\_end\_time,

sum(case when pickup\_location is not null then 1 else 0 end) pickup\_location

from

taxi\_trip\_database.taxi\_trip\_details\_taxi\_trip\_id\_removed

group by cast(substr(trip\_end\_time, 7, 4) as int) ,dropoff\_community

order by cast(substr(trip\_end\_time, 7, 4) as int) desc

limit 10;

select \* from taxi\_trip\_database.taxi\_trip\_details\_taxi\_trip\_total\_number

**5. Create a table which contains total number of trips for each drop-off communities across**

**weekdays & weekends to check if there is any sort of pattern visible. After creating the table, find**

**the top 10 drop off communities based on number of trips where people travel on weekdays.**

**Find the same for the weekends. Also find the total number of trips taken on weekdays &**

**weekends and their ratio.**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_details\_taxi\_trip\_id\_weekend\_weekdays

STORED AS ORC

AS

SELECT

dropoff\_community,

sum(CASE WHEN start\_dayofweek in (6,7) then 1 else 0 end) start\_dayofweek,

sum(CASE WHEN start\_dayofweek in (1,2,3,4,5) then 1 else 0 end) start\_dayofweek,

(sum(CASE WHEN start\_dayofweek in (1,2,3,4,5) then 1 else 0 end) start\_dayofweek) / (sum(CASE WHEN start\_dayofweek in (6,7) then 1 else 0 end) start\_dayofweek)

from

taxi\_trip\_database.taxi\_trip\_details\_processed\_with\_dayofweek

group by dropoff\_community

order by dropoff\_community

limit 10;

**6. Find the distribution of total number of trips based on trip duration, like <1 hr, 1 to 2 hr, 2 to 3, …**

**22 to 23 hr. Note that this requires converting trip\_seconds into trip\_hours as pre-processing.**

**Remove the trips that do not contain trip duration.**

select trip\_range, count(\*) cnt

from (select

case when (trip\_seconds/3600 >=0 and trip\_seconds/3600 <1) then '(0-1)'

when (trip\_seconds/3600 >=1 and trip\_seconds/3600 <2) then '(1-2)'

when (trip\_seconds/3600 >=2 and trip\_seconds/3600 <3) then '(2-3)'

when (trip\_seconds/3600 >=3 and trip\_seconds/3600 <4) then '(3-4)'

when (trip\_seconds/3600 >=4 and trip\_seconds/3600 <5) then '(4-5)'

when (trip\_seconds/3600 >=5 and trip\_seconds/3600 <6) then '(5-6)'

when (trip\_seconds/3600 >=6 and trip\_seconds/3600 <7) then '(6-7)'

when (trip\_seconds/3600 >=7 and trip\_seconds/3600 <8) then '(7-8)'

when (trip\_seconds/3600 >=8 and trip\_seconds/3600 <9) then '(8-9)'

when (trip\_seconds/3600 >=9 and trip\_seconds/3600 <10) then '(9-10)'

when (trip\_seconds/3600 >=10 and trip\_seconds/3600 <11) then '(10-11)'

when (trip\_seconds/3600 >=11 and trip\_seconds/3600 <12) then '(11-12)'

when (trip\_seconds/3600 >=12 and trip\_seconds/3600 <13) then '(12-13)'

when (trip\_seconds/3600 >=13 and trip\_seconds/3600 <14) then '(13-14)'

when (trip\_seconds/3600 >=14 and trip\_seconds/3600 <15) then '(14-15)'

when (trip\_seconds/3600 >=15 and trip\_seconds/3600 <16) then '(15-16)'

when (trip\_seconds/3600 >=16 and trip\_seconds/3600 <17) then '(16-17)'

when (trip\_seconds/3600 >=17 and trip\_seconds/3600 <18) then '(17-18)'

when (trip\_seconds/3600 >=18 and trip\_seconds/3600 <19) then '(18-19)'

when (trip\_seconds/3600 >=19 and trip\_seconds/3600 <20) then '(19-20)'

when (trip\_seconds/3600 >=20 and trip\_seconds/3600 <21) then '(20-21)'

when (trip\_seconds/3600 >=21 and trip\_seconds/3600 <22) then '(21-22)'

when (trip\_seconds/3600 >=22 and trip\_seconds/3600 <23) then '(22-23)' end trip\_range

from taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded) T

group by trip\_range

order by cnt desc;



**7. Find the top 10 buckets of the number of trips distribution based on the distance covered. Also**

**round off the trip miles to the nearest integer. Remove the trips that do not contain distance.**

trip\_id\_int int

taxi\_id\_int int

trip\_start\_time string

trip\_end\_time string

trip\_seconds int

trip\_miles float

pickup\_tract bigint

dropoff\_tract bigint

pickup\_community tinyint

dropoff\_community tinyint

trip\_fare float

tip\_amt float

toll\_amt float

extra\_amt float

trip\_total\_amt float

payment\_type string

company string

pickup\_latitude double

pickup\_longitude double

pickup\_location string

dropoff\_latitude double

dropoff\_longitude double

dropoff\_location string

community\_areas string

start\_dayofweek string

end\_dayofweek string

weekend int

Roundoff | Distance Number of Trips

SELECT trip\_miles, count(\*) AS Total

FROM taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded

WHERE isnotnull(trip\_miles) and trip\_miles > 0

GROUP BY trip\_miles

ORDER BY trip\_miles desc limit 10;



**8. Find top 10 buckets of the number of trips distribution based on the trip fare. Also round off the**

**trip fare to the nearest integer. Remove the trips that do not contain trip fare.**

**Roundoff Fare Number of Trips**

SELECT trip\_fare, count(\*) AS Total

FROM taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded

WHERE isnotnull(trip\_fare) and trip\_fare > 0

GROUP BY trip\_fare

ORDER BY trip\_fare desc limit 10;



**9. Compute the average trip fare per day. Also compute the average trip fare per trip. Compute the**

**same based on weekdays and weekend days. Find out if there is any substantial difference**

**observed.**

**Normal Days :**

SELECT

cast(substr(trip\_end\_time, 7, 4) as int) trip\_end\_time,

avg(trip\_fare)

from

taxi\_trip\_database.taxi\_trip\_details\_taxi\_trip\_id\_removed

group by cast(substr(trip\_end\_time, 7, 4) as int)

order by cast(substr(trip\_end\_time, 7, 4) as int) desc

limit 10;



SELECT

dropoff\_tract ,

avg(trip\_fare)

from

taxi\_trip\_database.taxi\_trip\_details\_taxi\_trip\_id\_removed

group by dropoff\_tract

order by dropoff\_tract desc

limit 10;



**Weekend Days :**

SELECT

cast(substr(trip\_end\_time, 7, 4) as int) trip\_end\_time,

avg(trip\_fare)

from

taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded

where weekend > 0

group by cast(substr(trip\_end\_time, 7, 4) as int)

order by cast(substr(trip\_end\_time, 7, 4) as int) desc

limit 10;



SELECT

dropoff\_tract ,

avg(trip\_fare)

from

taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded

where weekend > 0

group by dropoff\_tract

order by dropoff\_tract desc

limit 10;



**10. Create a table to store the taxi wise total fare & total number of trips for each day. Find the**

**following insights from the table:**

**a. Find the top 10 taxis based on average trips per day.**

**b. Find the top 10 taxis based on average fare per day.**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_details\_avg\_trip\_fare\_per\_day

STORED AS ORC

AS

SELECT

cast(substr(trip\_end\_time, 7, 4) as int) trip\_end\_time,

avg(trip\_fare)

from

taxi\_trip\_database.taxi\_trip\_details\_taxi\_trip\_id\_removed

group by cast(substr(trip\_end\_time, 7, 4) as int)

order by cast(substr(trip\_end\_time, 7, 4) as int) desc

limit 10;

describe taxi\_trip\_database.taxi\_trip\_details\_avg\_trip\_fare\_per\_day

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_details\_avg\_trip\_fare\_per\_trip

STORED AS ORC

AS

SELECT

dropoff\_tract ,

avg(trip\_fare)

from

taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded

where weekend > 0

group by dropoff\_tract

order by dropoff\_tract desc

limit 10;

describe taxi\_trip\_database.taxi\_trip\_details\_avg\_trip\_fare\_per\_trip

**3.2.2 Data Preparation for Forecasting**

**DAILY SUMMARY TABLE**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_details\_summary

STORED AS ORC

AS

SELECT

\*,

day(trip\_start\_time) AS day,

month(trip\_start\_time) AS month,

year(trip\_start\_time) AS year,

CASE

WHEN start\_dayofweek in (6,7) THEN 1

WHEN start\_dayofweek in (1,2,3,4,5) THEN 0

END AS sat\_sun

from taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded;

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_daily\_details\_summary

STORED AS ORC

AS

SELECT

day

month,

year,

CASE

WHEN start\_dayofweek in (6,7) THEN 1

WHEN start\_dayofweek in (1,2,3,4,5) THEN 0

END AS satsun,

sum(case when trip\_end\_time is not null then 1 else 0 end) trip\_end\_time,

trip\_total\_amt,

sum(trip\_seconds/60),

avg(trip\_fare),

avg(trip\_miles),

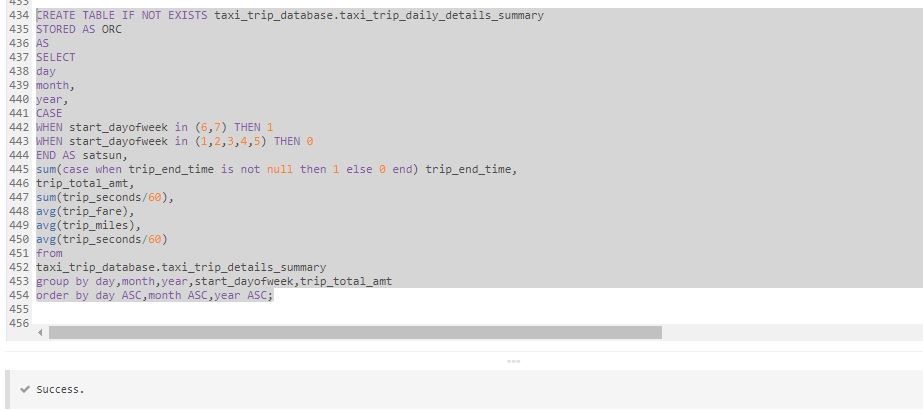
avg(trip\_seconds/60)

from

taxi\_trip\_database.taxi\_trip\_details\_summary

group by day,month,year,start\_dayofweek,trip\_total\_amt

order by day ASC,month ASC,year ASC;



**WEEKLY SUMMARY TABLE**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_weekly\_details\_summary

STORED AS ORC

AS

SELECT

CASE

WHEN start\_dayofweek in (1,2,3,4,5,6,7) THEN 1 ELSE 0 END AS Week\_No,

substr(trip\_start\_time, 1, 11),

substr(trip\_end\_time, 1, 11),

substr(trip\_end\_time, 4, 2),

count(trip\_id\_int),

sum(trip\_fare),

sum(trip\_miles),

sum(trip\_seconds/60),

avg(trip\_fare),

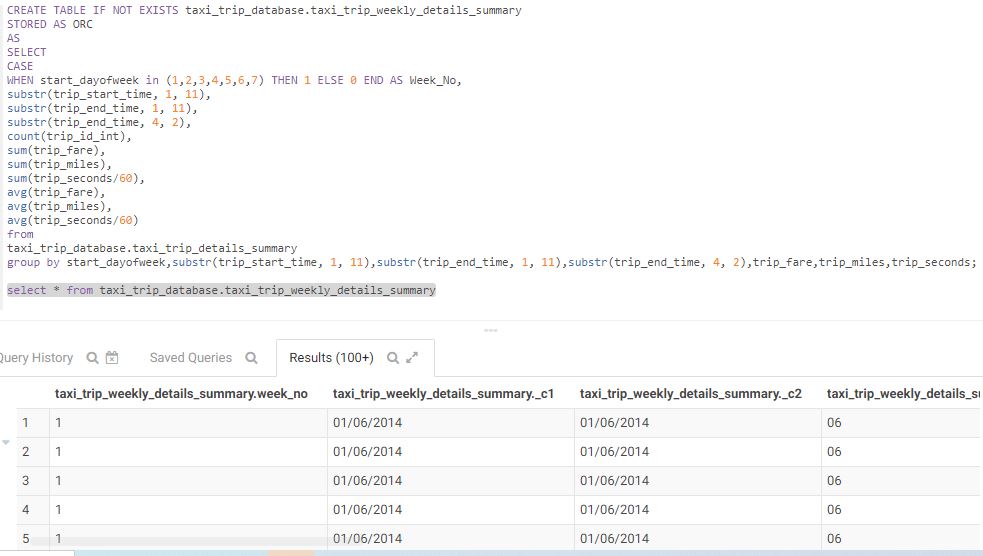
avg(trip\_miles),

avg(trip\_seconds/60)

from

taxi\_trip\_database.taxi\_trip\_details\_summary

group by start\_dayofweek,substr(trip\_start\_time, 1, 11),substr(trip\_end\_time, 1, 11),substr(trip\_end\_time, 4, 2),trip\_fare,trip\_miles,trip\_seconds;



**MONTHLY SUMMARY TABLE**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_monthly\_details\_summary

STORED AS ORC

AS

SELECT

CASE

WHEN substr(trip\_end\_time, 4, 2) IS NULL THEN 0 ELSE 1 END AS Month\_No,

substr(trip\_start\_time, 1, 11),

substr(trip\_end\_time, 1, 11),

cast(substr(trip\_end\_time, 7, 4) as int) ,

count(trip\_id\_int),

sum(trip\_fare),

sum(trip\_miles),

sum(trip\_seconds/60),

avg(trip\_fare),

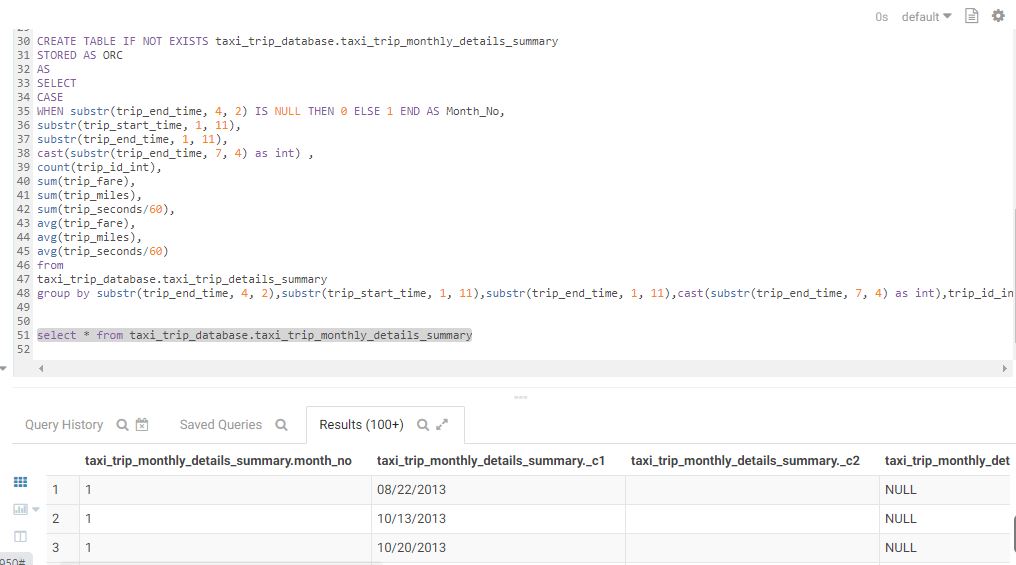
avg(trip\_miles),

avg(trip\_seconds/60)

from

taxi\_trip\_database.taxi\_trip\_details\_summary

group by substr(trip\_end\_time, 4, 2),substr(trip\_start\_time, 1, 11),substr(trip\_end\_time, 1, 11),cast(substr(trip\_end\_time, 7, 4) as int),trip\_id\_int,trip\_fare,trip\_miles,trip\_seconds;



**3.2.3 Data Preparation for Community Summary**

**Pickup Communities**

**Attributes Used by Pickup Communities**

PickupCommunityID,

DailyTripCount,

DailyTotalFare,

DailyTotalDistance,

DailyTotalDuration,

DailyAverageAmount,

DailyAverageDistance,

DailyAverageDuration

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_pickupcommunitydetails

STORED AS ORC

AS

SELECT \* FROM

(select

CASE

WHEN pickup\_community IS NULL THEN 0 ELSE pickup\_community END AS pickup\_community\_id,

CASE

WHEN trip\_end\_time IS NULL THEN 0 ELSE 1 END AS Trip\_Count,

sum(trip\_fare),

sum(trip\_miles),

sum(trip\_seconds/60),

avg(trip\_fare),

avg(trip\_miles),

avg(trip\_seconds/60)

from

taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded

group by pickup\_community,trip\_end\_time,trip\_fare,trip\_miles,trip\_seconds) b;



**Dropoff Communities**

**Attributes Used by Dropoff Communities**

DropoffCommunityID,

DailyTripCount,

DailyTotalFare,

DailyTotalDistance,

DailyTotalDuration,

DailyAverageAmount,

DailyAverageDistance,

DailyAverageDuration

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_dropoffcommunitydetails

STORED AS ORC

AS

SELECT \* FROM

(select dropoff\_community,

CASE

WHEN trip\_end\_time IS NULL THEN 0 ELSE 1 END AS Trip\_Count,

sum(trip\_fare),

sum(trip\_miles),

sum(trip\_seconds/60),

avg(trip\_fare),

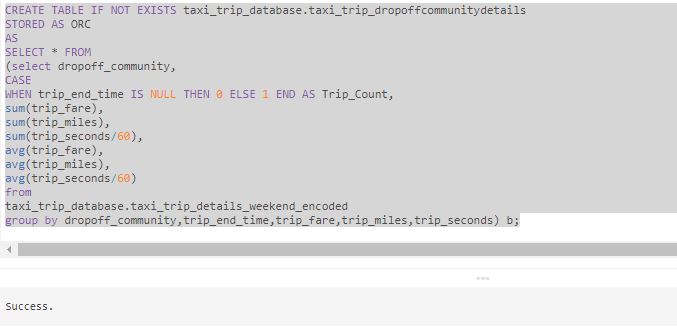
avg(trip\_miles),

avg(trip\_seconds/60)

from

taxi\_trip\_database.taxi\_trip\_details\_weekend\_encoded

group by dropoff\_community,trip\_end\_time,trip\_fare,trip\_miles,trip\_seconds) b;

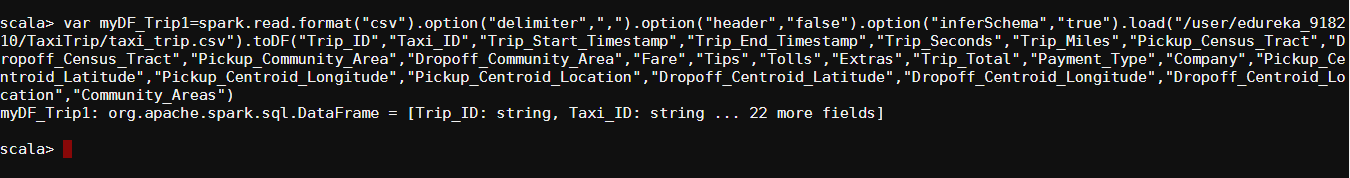


**3.2.4 Data Preparation for Origin to Destination Pair Summary**

IN SPARK

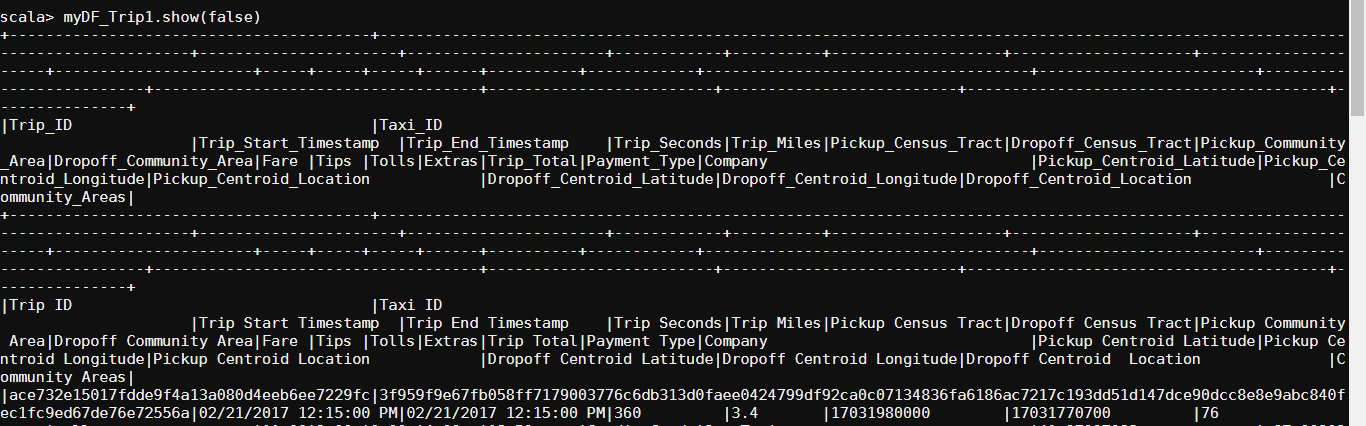
**1) Mapping the Raw Data with RDD(Resilent Distributed Data) and converstion of DF - Single Code**

var myDF\_Trip1=spark.read.format("csv").option("delimiter",",").option("header","false").option("inferSchema","true").load("/user/edureka\_918210/TaxiTrip/taxi\_trip.csv").toDF("Trip\_ID","Taxi\_ID","Trip\_Start\_Timestamp","Trip\_End\_Timestamp","Trip\_Seconds","Trip\_Miles","Pickup\_Census\_Tract","Dropoff\_Census\_Tract","Pickup\_Community\_Area","Dropoff\_Community\_Area","Fare","Tips","Tolls","Extras","Trip\_Total","Payment\_Type","Company","Pickup\_Centroid\_Latitude","Pickup\_Centroid\_Longitude","Pickup\_Centroid\_Location","Dropoff\_Centroid\_Latitude","Dropoff\_Centroid\_Longitude","Dropoff\_Centroid\_Location","Community\_Areas")



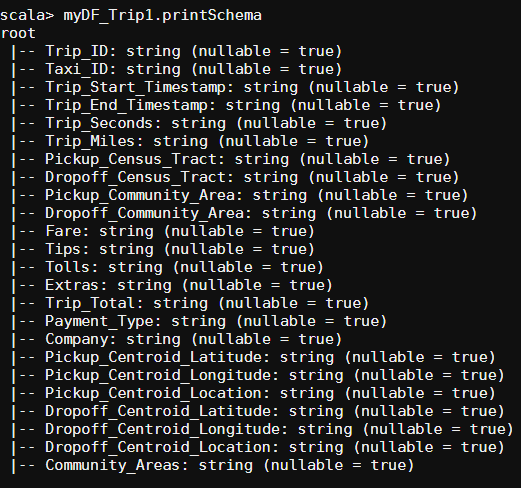
**2) DF definition . Please Note, we used InferSchema Option for automatic conversion in data definition in above command**

myDF\_Trip1.show(false)



**3) DF data definition**

myDF\_Trip1.printSchema



root

|-- Trip\_ID: string (nullable = true)

|-- Taxi\_ID: string (nullable = true)

|-- Trip\_Start\_Timestamp: string (nullable = true)

|-- Trip\_End\_Timestamp: string (nullable = true)

|-- Trip\_Seconds: string (nullable = true)

|-- Trip\_Miles: string (nullable = true)

|-- Pickup\_Census\_Tract: string (nullable = true)

|-- Dropoff\_Census\_Tract: string (nullable = true)

|-- Pickup\_Community\_Area: string (nullable = true)

|-- Dropoff\_Community\_Area: string (nullable = true)

|-- Fare: string (nullable = true)

|-- Tips: string (nullable = true)

|-- Tolls: string (nullable = true)

|-- Extras: string (nullable = true)

|-- Trip\_Total: string (nullable = true)

|-- Payment\_Type: string (nullable = true)

|-- Company: string (nullable = true)

|-- Pickup\_Centroid\_Latitude: string (nullable = true)

|-- Pickup\_Centroid\_Longitude: string (nullable = true)

|-- Pickup\_Centroid\_Location: string (nullable = true)

|-- Dropoff\_Centroid\_Latitude: string (nullable = true)

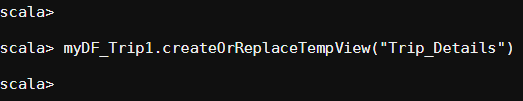
|-- Dropoff\_Centroid\_Longitude: string (nullable = true)

|-- Dropoff\_Centroid\_Location: string (nullable = true)

|-- Community\_Areas: string (nullable = true)

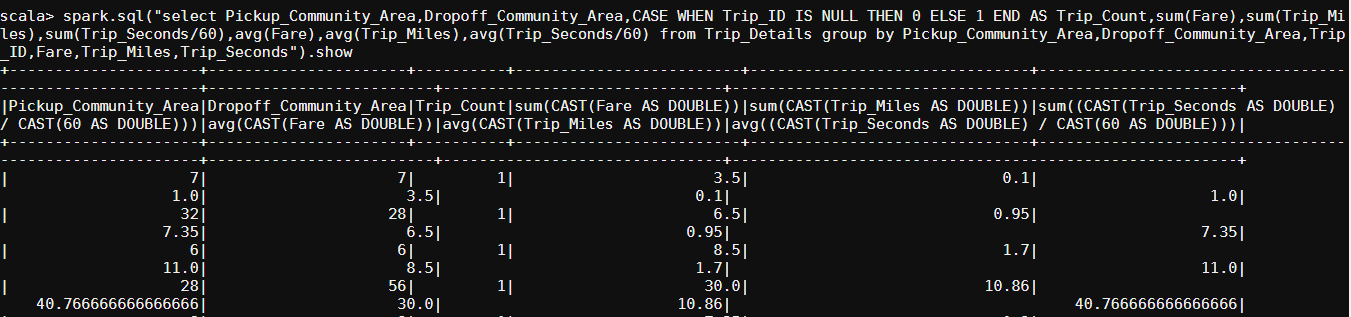
**4) Registering orCreation of Temp View for using Spark SQL**

myDF\_Trip1.createOrReplaceTempView("Trip\_Details")



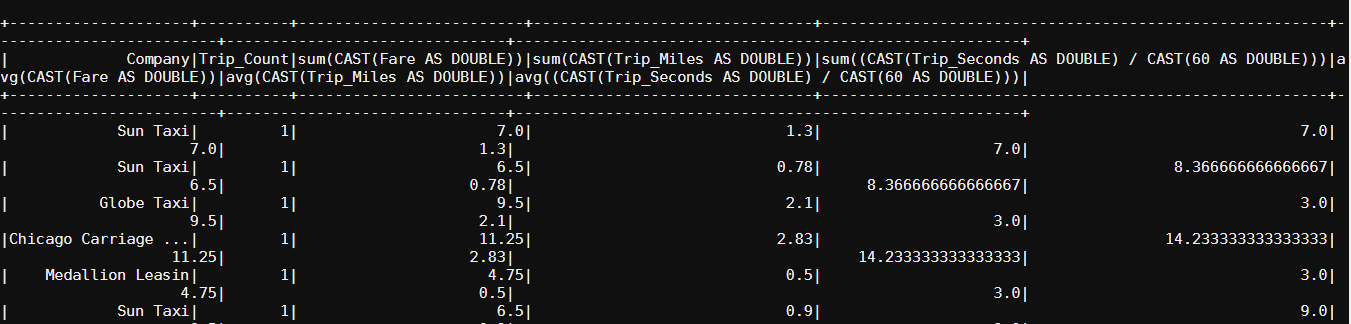
**5) Using Spark SQL**

spark.sql("select Pickup\_Community\_Area,Dropoff\_Community\_Area,CASE WHEN Trip\_ID IS NULL THEN 0 ELSE 1 END AS Trip\_Count,sum(Fare),sum(Trip\_Miles),sum(Trip\_Seconds/60),avg(Fare),avg(Trip\_Miles),avg(Trip\_Seconds/60) from Trip\_Details group by Pickup\_Community\_Area,Dropoff\_Community\_Area,Trip\_ID,Fare,Trip\_Miles,Trip\_Seconds").show



**3.2.5 Data Preparation for Company Summary**

spark.sql("select Company,CASE WHEN Trip\_ID IS NULL THEN 0 ELSE 1 END AS Trip\_Count,sum(Fare),sum(Trip\_Miles),sum(Trip\_Seconds/60),avg(Fare),avg(Trip\_Miles),avg(Trip\_Seconds/60) from Trip\_Details group by Company,Dropoff\_Community\_Area,Trip\_ID,Fare,Trip\_Miles,Trip\_Seconds").show



**3.2.6 Data Summary in RDBMS**

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_pickupcommunitydetails (

Pickup\_community\_id TINYINT NOT NULL,

daily\_trip\_count TINYINT NOT NULL,

total\_trip\_fare INT AUTO\_INCREMENT PRIMARY KEY,

total\_trip\_miles TINYINT NOT NULL,

total\_trip\_mins TINYINT NOT NULL,

avg\_trip\_fare TINYINT NOT NULL,

avg\_trip\_miles TINYINT NOT NULL,

avg\_trip\_mins TINYINT NOT NULL

) ENGINE=INNODB;

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_dropoffcommunitydetails (

dropoff\_community\_id TINYINT NOT NULL,

daily\_trip\_count TINYINT NOT NULL,

total\_trip\_fare INT AUTO\_INCREMENT PRIMARY KEY,

total\_trip\_miles TINYINT NOT NULL,

total\_trip\_mins TINYINT NOT NULL,

avg\_trip\_fare TINYINT NOT NULL,

avg\_trip\_miles TINYINT NOT NULL,

avg\_trip\_mins TINYINT NOT NULL

) ENGINE=INNODB;

CREATE TABLE IF NOT EXISTS taxi\_trip\_database.taxi\_trip\_companydetails (

company TINYINT NOT NULL,

daily\_trip\_count TINYINT NOT NULL,

total\_trip\_fare INT AUTO\_INCREMENT PRIMARY KEY,

total\_trip\_miles TINYINT NOT NULL,

total\_trip\_mins TINYINT NOT NULL,

avg\_trip\_fare TINYINT NOT NULL,

avg\_trip\_miles TINYINT NOT NULL,

avg\_trip\_mins TINYINT NOT NULL

) ENGINE=INNODB;

**LOADING THE DATA**

LOAD DATA INFILE '/mnt/home/edureka\_918210/Taxi\_Trip\_ResultSet/taxi\_trip\_pickupcommunitydetails.csv'

INTO TABLE taxi\_trip\_database.taxi\_trip\_pickupcommunitydetails

FIELDS TERMINATED BY ','

LINES TERMINATED BY '/n'

IGNORE 1 ROWS;

LOAD DATA INFILE '/mnt/home/edureka\_918210/Taxi\_Trip\_ResultSet/percentageofrecordsthat contains drop-off community.csv'

INTO TABLE taxi\_trip\_database.taxi\_trip\_dropoffcommunitydetails

FIELDS TERMINATED BY ','

LINES TERMINATED BY '/n'

IGNORE 1 ROWS;

LOAD DATA INFILE '/mnt/home/edureka\_918210/Taxi\_Trip\_ResultSet/taxi\_trip\_pickupcommunitydetails.csv'

INTO TABLE taxi\_trip\_database.taxi\_trip\_companydetails

FIELDS TERMINATED BY ','

LINES TERMINATED BY '/n'

IGNORE 1 ROWS;

**3.2.7 Summary Data Mart**

CREATING TABLES FROM HIVE TO HBASE :

**PICKUP COMMUNITY DETAILS :**

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.hbase.HBaseConfiguration;

import org.apache.hadoop.hbase.HColumnDescriptor;

import org.apache.hadoop.hbase.HTableDescriptor;

import org.apache.hadoop.hbase.client.HBaseAdmin;

public class CreateHBaseTable

{

public static void main(String[] args) throws IOException

{

HBaseConfiguration hconfig = new HBaseConfiguration(new Configuration());

HTableDescriptor htable = new HTableDescriptor("taxi\_trip\_pickupcommunitydetails");

htable.addFamily( new HColumnDescriptor("pickup\_community\_id"));

htable.addFamily( new HColumnDescriptor("Trip\_Count"));

htable.addFamily( new HColumnDescriptor("Total\_trip\_fare"));

htable.addFamily( new HColumnDescriptor("Total\_trip\_miles"));

htable.addFamily( new HColumnDescriptor("Total\_trip\_duration\_mins"));

htable.addFamily( new HColumnDescriptor("Average\_trip\_fare"));

htable.addFamily( new HColumnDescriptor("Average\_trip\_miles"));

htable.addFamily( new HColumnDescriptor("Average\_trip\_duration\_mins"));

System.out.println( "Connecting..." );

HBaseAdmin hbase\_admin = new HBaseAdmin( hconfig );

System.out.println( "Creating Table..." );

hbase\_admin.createTable( htable );

System.out.println("Done!");

}

}

**DROPOFF COMMUNITY DETAILS :**

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.hbase.HBaseConfiguration;

import org.apache.hadoop.hbase.HColumnDescriptor;

import org.apache.hadoop.hbase.HTableDescriptor;

import org.apache.hadoop.hbase.client.HBaseAdmin;

public class CreateHBaseTable

{

public static void main(String[] args) throws IOException

{

HBaseConfiguration hconfig = new HBaseConfiguration(new Configuration());

HTableDescriptor htable = new HTableDescriptor("taxi\_trip\_dropoffcommunitydetails");

htable.addFamily( new HColumnDescriptor("dropoff\_community\_id"));

htable.addFamily( new HColumnDescriptor("Trip\_Count"));

htable.addFamily( new HColumnDescriptor("Total\_trip\_fare"));

htable.addFamily( new HColumnDescriptor("Total\_trip\_miles"));

htable.addFamily( new HColumnDescriptor("Total\_trip\_duration\_mins"));

htable.addFamily( new HColumnDescriptor("Average\_trip\_fare"));

htable.addFamily( new HColumnDescriptor("Average\_trip\_miles"));

htable.addFamily( new HColumnDescriptor("Average\_trip\_duration\_mins"));

System.out.println( "Connecting..." );

HBaseAdmin hbase\_admin = new HBaseAdmin( hconfig );

System.out.println( "Creating Table..." );

hbase\_admin.createTable( htable );

System.out.println("Done!");

}

}

**COMPANY SUMMARY TABLE**

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.hbase.HBaseConfiguration;

import org.apache.hadoop.hbase.HColumnDescriptor;

import org.apache.hadoop.hbase.HTableDescriptor;

import org.apache.hadoop.hbase.client.HBaseAdmin;

public class CreateHBaseTable

{

public static void main(String[] args) throws IOException

{

HBaseConfiguration hconfig = new HBaseConfiguration(new Configuration());

HTableDescriptor htable = new HTableDescriptor("taxi\_trip\_dropoffcommunitydetails");

htable.addFamily( new HColumnDescriptor("Company"));

htable.addFamily( new HColumnDescriptor("Trip\_Count"));

htable.addFamily( new HColumnDescriptor("Total\_trip\_fare"));

htable.addFamily( new HColumnDescriptor("Total\_trip\_miles"));

htable.addFamily( new HColumnDescriptor("Total\_trip\_duration\_mins"));

htable.addFamily( new HColumnDescriptor("Average\_trip\_fare"));

htable.addFamily( new HColumnDescriptor("Average\_trip\_miles"));

htable.addFamily( new HColumnDescriptor("Average\_trip\_duration\_mins"));

System.out.println( "Connecting..." );

HBaseAdmin hbase\_admin = new HBaseAdmin( hconfig );

System.out.println( "Creating Table..." );

hbase\_admin.createTable( htable );

System.out.println("Done!");

}

}

**3.2.7 SUMMARY DATA MART**

package HBASECRUD.HBASE;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.hbase.Cell;

import org.apache.hadoop.hbase.CellUtil;

import org.apache.hadoop.hbase.HColumnDescriptor;

import org.apache.hadoop.hbase.HTableDescriptor;

import org.apache.hadoop.hbase.TableName;

import org.apache.hadoop.hbase.client.Connection;

import org.apache.hadoop.hbase.client.ConnectionFactory;

import org.apache.hadoop.hbase.client.Delete;

import org.apache.hadoop.hbase.client.HBaseAdmin;

import org.apache.hadoop.hbase.client.Put;

import org.apache.hadoop.hbase.client.Result;

import org.apache.hadoop.hbase.client.ResultScanner;

import org.apache.hadoop.hbase.client.Scan;

import org.apache.hadoop.hbase.client.Table;

import org.apache.hadoop.hbase.util.Bytes;

public class HBaseCustomClient {

private HBaseAdmin admin;

private Connection connection = null;

public HBaseCustomClient(Configuration conf) throws IOException {

connection = ConnectionFactory.createConnection(conf);

admin = (HBaseAdmin) connection.getAdmin();

}

public void createTable(String tableName, String[] CFs) {

try {

if (admin.tableExists(tableName)) {

System.out.println(tableName + "Already Exists");

} else {

HTableDescriptor tableDescriptor = new HTableDescriptor(TableName.valueOf(tableName));

for (String CFName : CFs) {

tableDescriptor.addFamily(new HColumnDescriptor(CFName));

}

admin.createTable(tableDescriptor);

}

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public void deleteTable(String tableName) {

try {

if (admin.tableExists(tableName)) {

admin.disableTable(tableName);

admin.deleteTable(tableName);

} else {

System.out.println(tableName + " Doesn't exist");

}

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public void insertRecord(String tableName, String rowKey, String family, String qualifier, String value) {

try {

Table table = connection.getTable(TableName.valueOf(tableName));

Put p = new Put(Bytes.toBytes(rowKey));

p.addColumn(Bytes.toBytes(family), Bytes.toBytes(qualifier), Bytes.toBytes(value));

table.put(p);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public void deleteRecord(String tableName, String rowKey) {

try {

Table table = connection.getTable(TableName.valueOf(tableName));

Delete d = new Delete(Bytes.toBytes(rowKey));

table.delete(d);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public void printTable(String tablename) {

ResultScanner rsObj = null;

try {

Table table = connection.getTable(TableName.valueOf(tablename));

Scan s = new Scan();

rsObj = table.getScanner(s);

for (Result result : rsObj) {

System.out.println(Bytes.toString(result.getRow()));

for (Cell c : result.rawCells()) {

System.out.println("Family: " + Bytes.toString(CellUtil.cloneFamily(c)));

System.out.println("Qualifiers: " + Bytes.toString(CellUtil.cloneQualifier(c)));

System.out.println("Values: " + Bytes.toString(CellUtil.cloneValue(c)));

}

}

rsObj.close();

} catch (IOException e) {

// TODO Auto-generated catch block

rsObj.close();

e.printStackTrace();

}

}

}