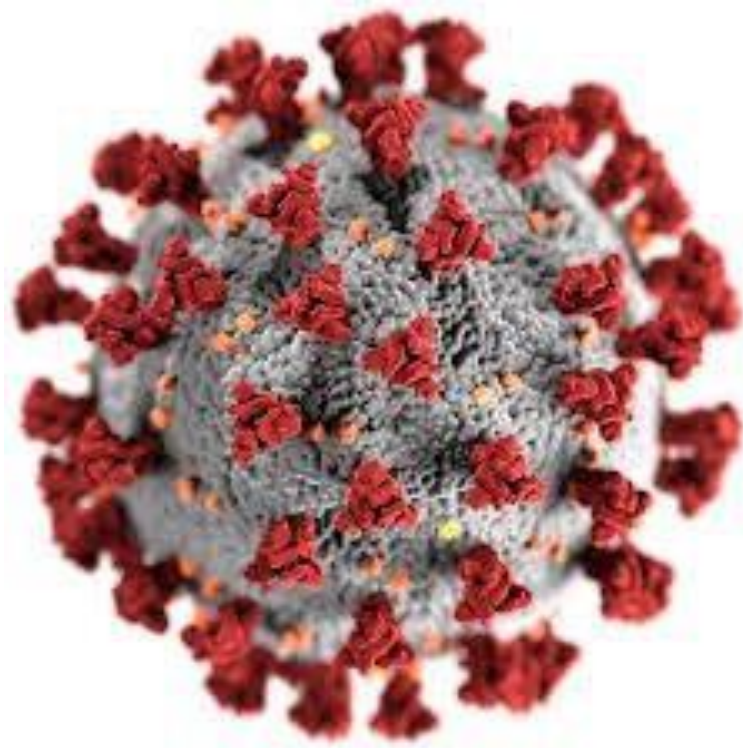




COVID-19

Group-6





Group Members

Name	CWID	HAWK ID
Sibo Bai	A20434344	sbai8@hawk.iit.edu
Sarthak Agrawal	A20444355	sagrawal4@hawk.iit.edu
Shivani Agrawal	A20443602	sagrawal3@hawk.iit.edu
Vaibhavi kulkarni	A20452251	vkulkarni3@hawk.iit.edu
Shreeya Deshpande	A20438928	sdeshpande4@hawk.iit.edu
Ramakant Chaudhari	A20446183	rchaudhari1@hawk.iit.edu

Index

Serial No	Topic	Page No
1	Introduction	03
2	Data Preparation	03
3	Provisioning & Transformation	08
4	Dashboard Implementation	18
5	Analysis	20

Abstract

Nowadays, COVID-19 is a global pandemic according to the assessment. In this report, our group is trying to find out how much impact and casualty the United States have faced state-wise till now. Here, with the help of data warehousing, SQL, and Pentaho transformation knowledge we are building a cumulative study model and displaying it with the help of some visualization graphs using tableau. Here, we discuss the design of data warehouses for COVID-19. We have collected an extensive set of data on COVID-19 from the Kaggle website. We have done all the necessary pre-processing on this data. We have implemented ETL to simplify source code conversion to realize the diversity of data points for reporting. Then we design the type of visualization to attract audiences by using interactive elements. We believe that our prediction and analysis of the data is very useful for COVID-19.



Introduction

A new coronavirus designated 2019-nCoV was first identified in Wuhan, the capital of China's Hubei province. People developed pneumonia without a clear cause and for which existing vaccines or treatments were not effective. The virus has shown evidence of human-to-human transmission. In the main page of Our World in Data, we can find a daily updated comprehensive overview of the data and research on the pandemic of COVID-19. The visualizations of confirmed cases and confirmed deaths are updated daily and published by the European centers for disease control, the best global data set on the epidemic.

We started by collecting the data and then Cleaning it so that we could proceed with transforming it. The data cleaning process involved the following steps:

Data Preparation

```

1 setwd("C:\\Users\\13128\\Downloads\\us-counties-covid-19-dataset", header = T)
2 MyData = read.csv("us-counties.csv", header = T)
3 str(MyData)
4

```

```

C:/Users/13128/Downloads/us-counties-covid-19-dataset/
> setwd("C:/Users/13128/Downloads/us-counties-covid-19-dataset")
> MyData = read.csv("us-counties.csv", header = T)
> str(MyData)
'data.frame':  53848 obs. of  6 variables:
 $ date : Factor w/ 83 levels "1/21/2020","1/22/2020",...: 1 2 3 4 4 5 5 6 6 ...
 $ county: Factor w/ 1608 levels "Abbeville","Acadia",...: 1325 1325 1325 348 1325 1066 348 1325 886 850 ...
 $ state : Factor w/ 55 levels "Alabama","Alaska",...: 52 52 52 15 52 5 15 52 3 5 ...
 $ fips  : int  53061 53061 53061 17031 53061 6059 17031 53061 4013 6037 ...
 $ cases : Factor w/ 1579 levels "0","1","10","100",...: 2 2 2 2 2 2 2 2 2 2 ...
 $ deaths: int  0 0 0 0 0 0 0 0 0 0 ...

```

Using “setwd” to get the path and working directory of our folder in which our data is present.

Using “MyData” we are loading our .csv file into the data frame.

Using “str” displaying various columns with their attributes on which we have to work.



```

1 setwd("C:\\Users\\13128\\Downloads\\us-counties-covid-19-dataset", header = T)
2 MyData = read.csv("us-counties.csv", header = T)
3 str(MyData)
4 View(MyData)
5

```

5:18 (Top Level) R

Console Terminal Jobs

C:/Users/13128/Downloads/us-counties-covid-19-dataset/

```

> setwd("C:/Users/13128/Downloads/us-counties-covid-19-dataset")
> MyData = read.csv("us-counties.csv", header = T)
> str(MyData)
'data.frame':  53848 obs. of  6 variables:
 $ date   : Factor w/ 83 levels "1/21/2020","1/22/2020",...: 1 2 3 4 5 5 5 6 6 ...
 $ county : Factor w/ 1608 levels "Abbeville","Acadia",...: 1325 1325 1325 348 1325 1066 348 1325 886 850 ...
 $ state  : Factor w/ 55 levels "Alabama","Alaska",...: 52 52 52 15 52 5 15 52 3 5 ...
 $ fips   : int  53061 53061 53061 17031 53061 6059 17031 53061 4013 6037 ...
 $ cases  : Factor w/ 1579 levels "0","1","10","100",...: 2 2 2 2 2 2 2 2 2 2 ...
 $ deaths : int  0 0 0 0 0 0 0 0 0 0 ...
> View(MyData)
>

```

Using “**View**” function, we find whether the correct file is loaded or not.

Our data looks like this:

	date	county	state	fips	cases	deaths
1	1/21/2020	Snohomish	Washington	53061	1	0
2	1/22/2020	Snohomish	Washington	53061	1	0
3	1/23/2020	Snohomish	Washington	53061	1	0
4	1/24/2020	Cook	Illinois	17031	1	0
5	1/24/2020	Snohomish	Washington	53061	1	0
6	1/25/2020	Orange	California	6059	1	0
7	1/25/2020	Cook	Illinois	17031	1	0
8	1/25/2020	Snohomish	Washington	53061	1	0
9	1/26/2020	Maricopa	Arizona	4013	1	0
10	1/26/2020	Los Angeles	California	6037	1	0
11	1/26/2020	Orange	California	6059	1	0
12	1/26/2020	Cook	Illinois	17031	1	0

Showing 1 to 14 of 53,848 entries, 6 total columns



```

4 View(MyData)
5 library (janitor)
6 DATE = MyData$date
7 COUNTY = MyData$county
8 STATE = MyData$state
9 FIPS = MyData$fips
10 CASES = MyData$cases
11 DEATHS = MyData$deaths
12 summary(MyData$fips)
13
13:1 (Top Level)

```

Console Terminal Jobs

C:/Users/13128/Downloads/us-counties-covid-19-dataset/

```

> library (janitor)
> DATE = MyData$date
> COUNTY = MyData$county
> STATE = MyData$state
> FIPS = MyData$fips
> CASES = MyData$cases
> DEATHS = MyData$deaths
> summary(MyData$fips)
  Min. 1st Qu.  Median    Mean 3rd Qu.   Max.   NA's
1001  17161  28135  29529  42125  56043    716

```

Using the library “Janitor” we load the attributes for the pre-processing that needs to be performed.

Using “Summary” we get the various factors with null values present in the dataset.

We can see that in the “fips” column there are 700+ missing values.

```

9 FIPS = MyData$fips
10 CASES = MyData$cases
11
14:1 (Top Level)

```

Console Terminal Jobs

C:/Users/13128/Downloads/us-counties-covid-19-dataset/

```

> summary(MyData$cases)
 1    2    3    4    5    6    7    8   10    9   11   12   13
11355 5946 4386 2972 2304 1887 1527 1290 1067 1056 922 792 715
14    15   17   16   18   19   20   21   22   23   24   25   26
640   595  508  504  469  417  406  395  331  325  295  283  269
27    28   31   32   30   29   33   37   34   36   35   40   42
266   258  247  206  199  198  189  189  181  171  162  147  145
39    38   41   43   44   50   45   46   47   49   54   51   53
143   142  132  132  121  120  112  111  108  107  104  96   95
57    61   48   55   56   52   59   65   67   79   66   58   69
95    92   88   88   87   78   77   77   77   77   76   75   72
81    64   68   71   62   72   60   77   101  63   75   95   82
72    70   67   67   62   62   60   59   58   58   58   58   54
83    90   76   70   74   73   84   80   107  87   88   0    110
54    54   53   52   52   50   49   47   45   44   44   42   42
78    91   100  105  94   85   89   93   (other) 5295
42    42   41   41   41   39   39   39   5295

```

Performing the “summary” function on the attributes and finding if any null values.

In cases and death attributes there were no missing values and hence no pre-processing has been performed on them.



```

14 summary(MyData$deaths)
15 View(MyData)
16 library(dummies)
17 df=dummy.data.frame(MyData, names = c(""))
18 head(df)
19
19:1 (Top Level)

```

Console Terminal x Jobs x

C:/Users/13128/Downloads/us-counties-covid-19-dataset/

	70	91	100	105	94	85	89	95 (Owner)
	42	42	41	41	41	39	39	39 5295

```

> DEATHS = MyData$deaths
> View(MyData)
> df=dummy.data.frame(MyData, names = c(""))
> head(df)
  date      county      state  fips  cases  deaths
1 1/21/2020 Snohomish Washington 53061     1      0
2 1/22/2020 Snohomish Washington 53061     1      0
3 1/23/2020 Snohomish Washington 53061     1      0
4 1/24/2020      Cook  Illinois 17031     1      0
5 1/24/2020 Snohomish Washington 53061     1      0
6 1/25/2020   Orange California 6059     1      0
>

```

Here, we have created a dummy frame of the dataset to remove or to perform cleaning on the dataset.

“df” is the name of the variable set for the functions to be performed in the MyData dataset.

```

18 head(df)
19 MyData=MyData[,(-4)]
20 View(MyData)
21 write.csv(MyData, "C:\\Users\\13128\\Downloads\\newfile_us-counties-covid-19-dataset.csv", row.names = FALSE)
22
22:1 (Top Level)

```

Console Terminal x Jobs x

C:/Users/13128/Downloads/us-counties-covid-19-dataset/

```

> view(MyData)
> df=dummy.data.frame(MyData, names = c(""))
> head(df)
  date      county      state  fips  cases  deaths
1 1/21/2020 Snohomish Washington 53061     1      0
2 1/22/2020 Snohomish Washington 53061     1      0
3 1/23/2020 Snohomish Washington 53061     1      0
4 1/24/2020      Cook  Illinois 17031     1      0
5 1/24/2020 Snohomish Washington 53061     1      0
6 1/25/2020   Orange California 6059     1      0
> MyData=MyData[,(-4)]
> View(MyData)
> write.csv(MyData, "C:\\Users\\13128\\Downloads\\newfile_us-counties-covid-19-dataset.csv", row.names = FALSE)
>

```

Using the “head” function our first 6 records are displayed.

As we went through our data, column “fips” made no contribution in the further process, and also was not contributing to this process. Hence we have removed it from the dataset using “-4” as it is the fourth column in our dataset.



Using “*write.csv*” we have created our new dataset file which is ready for the further process of transformation.

	date	county	state	cases	deaths
1	1/21/2020	Snohomish	Washington	1	0
2	1/22/2020	Snohomish	Washington	1	0
3	1/23/2020	Snohomish	Washington	1	0
4	1/24/2020	Cook	Illinois	1	0
5	1/24/2020	Snohomish	Washington	1	0
6	1/25/2020	Orange	California	1	0
7	1/25/2020	Cook	Illinois	1	0
8	1/25/2020	Snohomish	Washington	1	0
9	1/26/2020	Maricopa	Arizona	1	0
10	1/26/2020	Los Angeles	California	1	0
11	1/26/2020	Orange	California	1	0
12	1/26/2020	Cook	Illinois	1	0

Showing 1 to 14 of 53,848 entries, 5 total columns

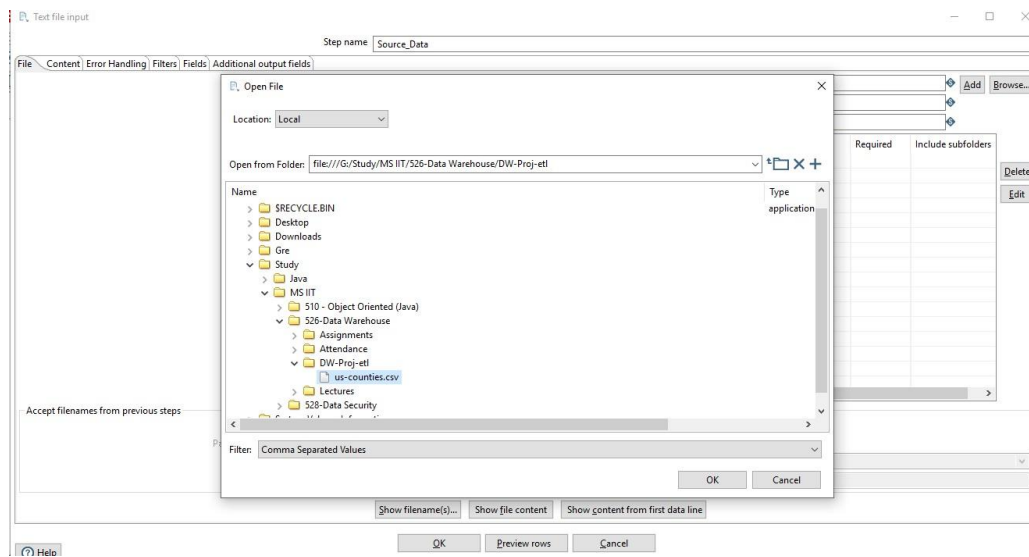
Using “*view*” we checked whether the column has been eliminated or not.

So, finally the fourth column of attribute “*fips*” has been removed and our preprocessed and cleaning a new .csv file has been completed.

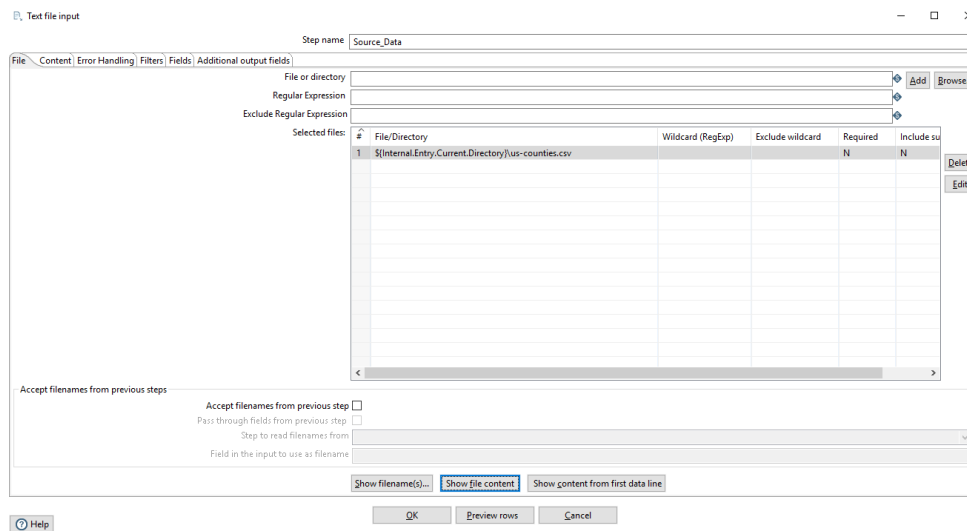


Provisioning and Transformation

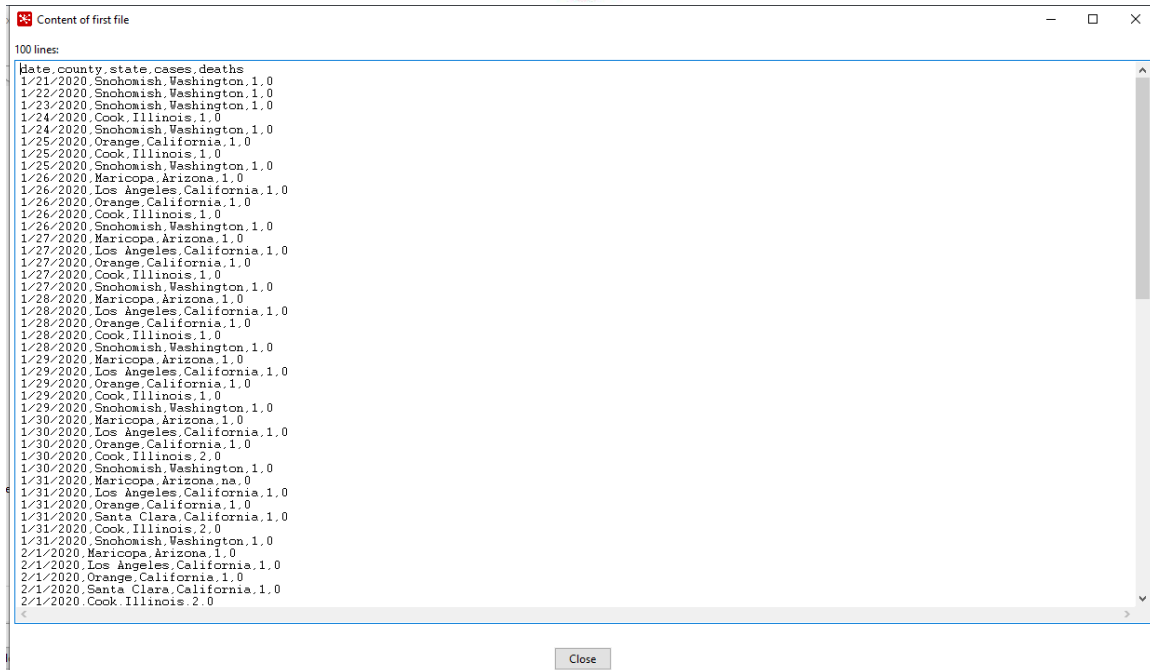
Steps performed for the Provisioning and Transformation of COVID-19 data.



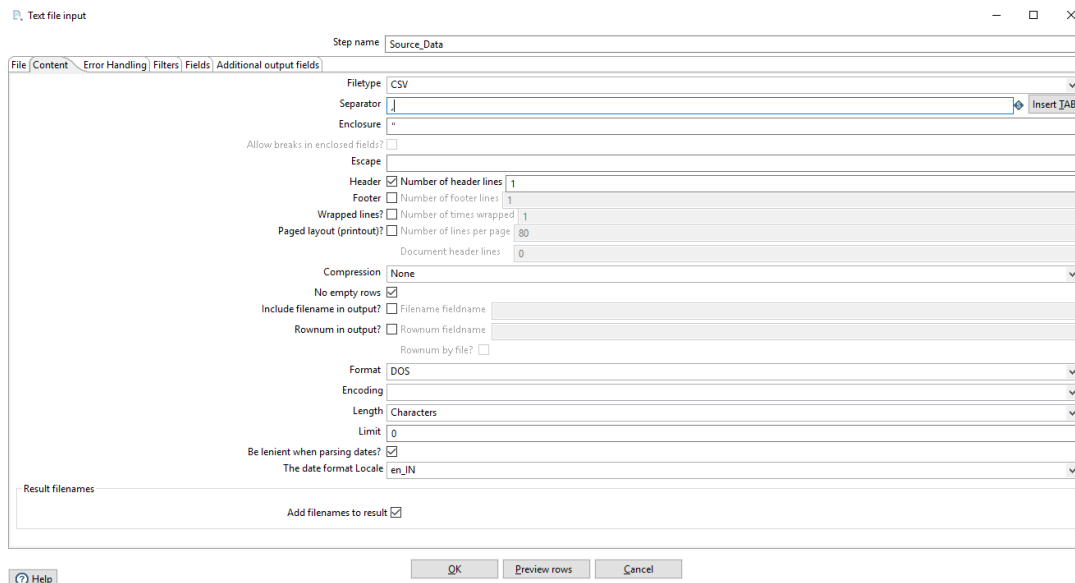
Text file input i.e. Source_Data



File Path replaced with an internal variable



Preview if the file is loaded properly.



As the file is CSV format need “,” separator.



Text file input

Scan results

Result after scanning 100 lines.

#	Name	Type	Format	Position
1	date	Date	MM/dd/yyyy	
2	county	String		
3	state	String		
4	cases	String		
5	deaths	Integer	#	

Field nr. 1 : date
Field name : Date
Field type : Date
Maximum length : MM/dd/yyyy
Date format : MM/dd/yyyy
WARNING: More than 1 date format seems to match all sampled records:
Date format : yyyy/MM/dd HH:mm:ss SSS
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900
Date format : yyyy/MM/dd HH:mm:ss SSS XXX
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900
Date format : yyyy/MM/dd HH:mm:ss
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900
Date format : yyyy/MM/dd HH:mm:ss XXX
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900
Date format : yyyyMMddHHmmss
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900
Date format : yyyy/MM/dd
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900
Date format : yyyyMMdd
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900
Date format : yyyy-MM-dd
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900
Date format : yyyy-MM-dd HH:mm:ss
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900
Date format : yyyy-MM-dd HH:mm:ss XXX
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900
Date format : yyyyMMdd
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900
Date format : MM/dd/yyyy
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900
Date format : MM/dd/yyyy HH:mm:ss
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900
Date format : MM-dd-yyyy
Minimum value : Wed Jan 01 04:29:59 IST 2200
Maximum value : Mon Jan 01 04:30:00 IST 1900

Help Close

Default Field types of columns.

Text file input

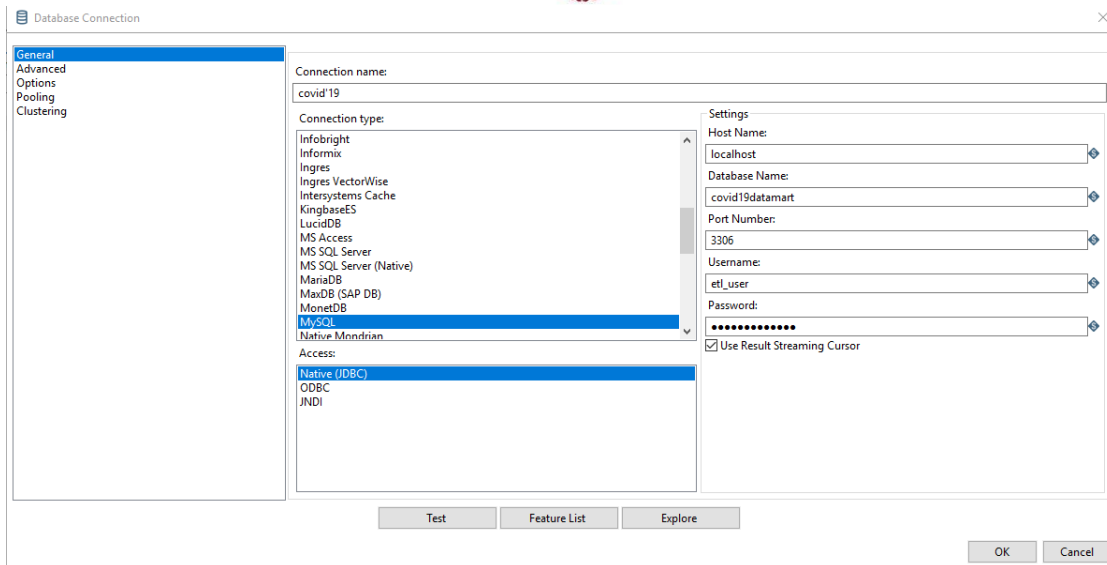
Step name Source_Data

#	Name	Type	Format	Position	Length	Precision	Currency	Decimal	Group	Null if	Default	Trim type	Repeat
1	date	Date	MM/dd/yyyy				Rs.	-	-	-		none	N
2	county	String					Rs.	-	-	-		none	N
3	state	String					Rs.	-	-	-		none	N
4	cases	String					Rs.	-	-	-		none	N
5	deaths	String					Rs.	-	-	-		none	N
6													
7													

Get Fields Minimal width

OK Preview rows Cancel

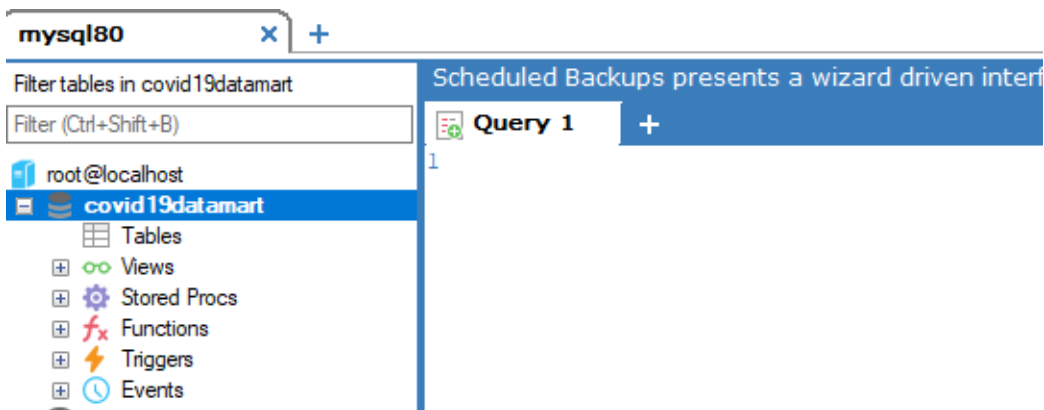
Changed type to string and removed other parameters w.r.t position, length, and precision.



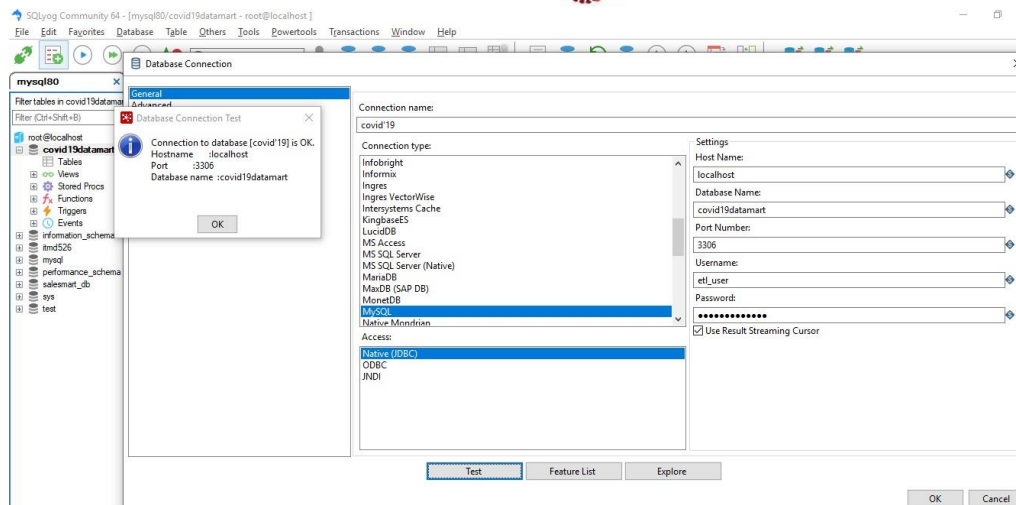
Then Created a new MySQL database connection for table output and covid19datamart in SQLYog.

Connection name: COVID'19

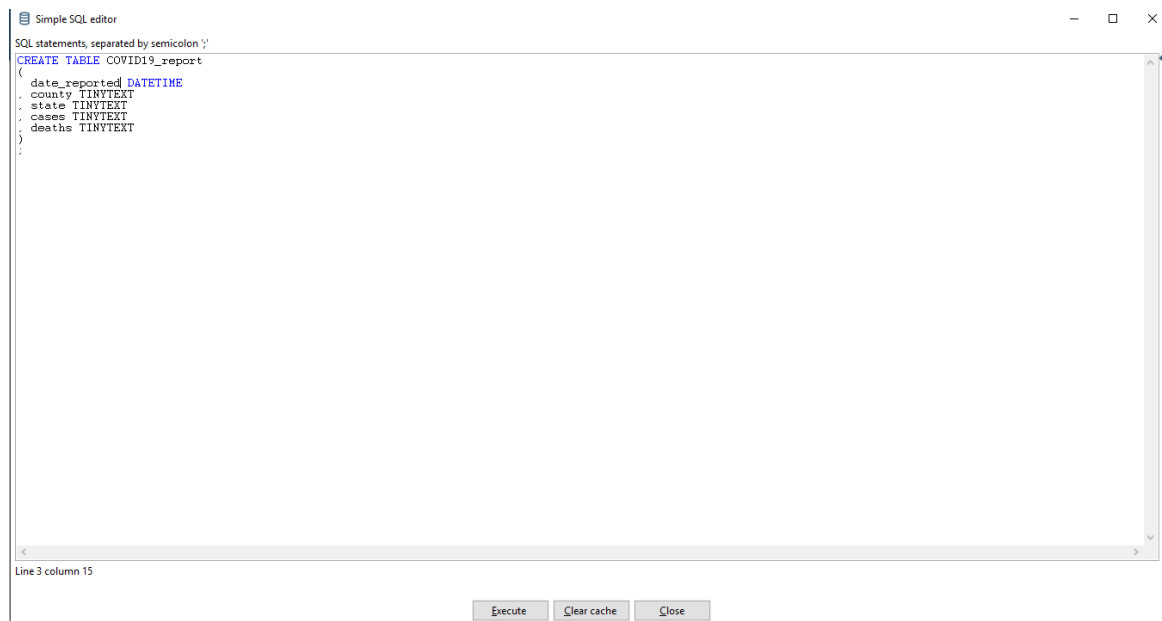
Database name: covid19datamart database.



Before creating a DB connection, we need DB in SQLyog as below:

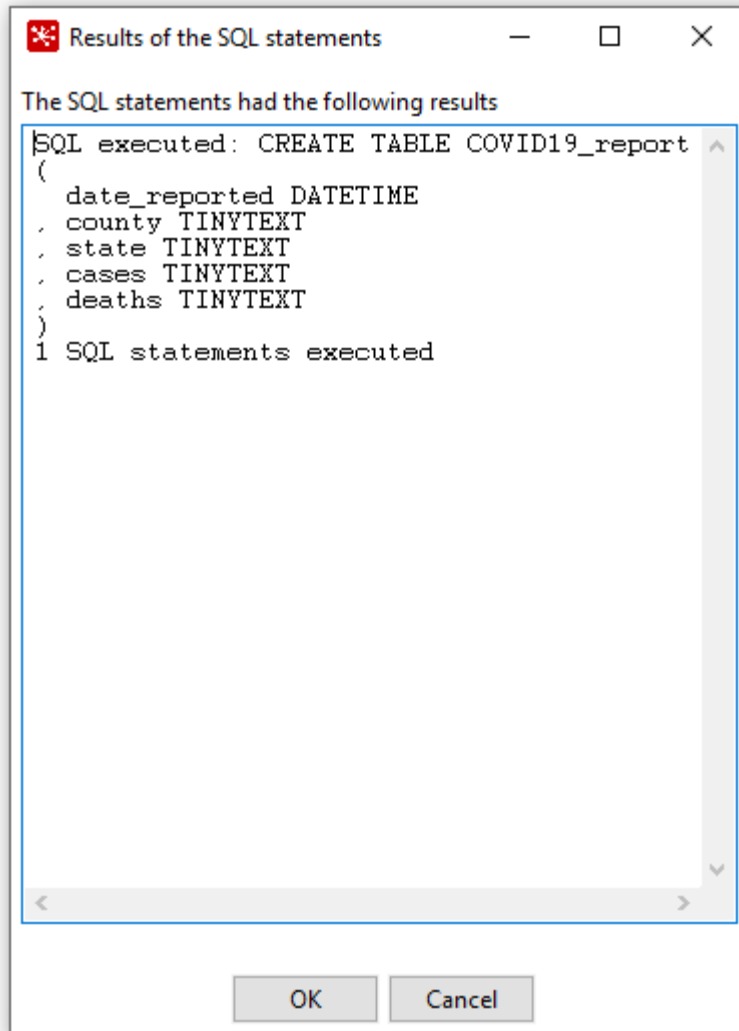


Tested Connection: OK



Creation of Table: covid19_report

Here, date to date_reported to avoid sql_query error.



SQL query executed successfully.



Table output

Step name: Table output

Connection: covid'19 [Edit... New... Wizard...]

Target schema: [Browse...]

Target table: COVID19_report [Browse...]

Commit size: 1000

Truncate table: ☒

Ignore insert errors: ☐

Specify database fields: ☒

Main options Database fields

Partition data over tables: ☐

Partitioning field: []

Partition data per month: ☒

Partition data per day: ☐

Use batch update for inserts: ☒

Is the name of the table defined in a field?: ☐

Field that contains name of table: []

Store the tablename field: ☒

Return auto-generated key: ☐

Name of auto-generated key field: []

[Help] [OK] [Cancel] [SQL]

Check the Truncate Table so that every run will have fresh data.

Spoon - export (changed)

File Edit View Action Tools Help

View Design

Search

Transformations

- export
 - Run configurations
 - Database connections
 - Steps
 - Hops
 - Partition schemas
 - Slave server
 - Kettle cluster schemas
 - Data Services
 - Hadoop clusters

Table output

Step name: Table output

Connection: covid'19 [Edit... New... Wizard...]

Target schema: []

Target table: COVID19_report [Browse...]

Commit size: 1000

Truncate table: ☒

Ignore insert errors: ☐

Specify database fields: ☒

Enter Mapping

Source fields: []

Target fields: []

Mappings:

- date --> date_reported
- county --> county
- state --> state
- cases --> cases
- deaths --> deaths

Add Delete

Auto target selection? ☒ Auto source selection? ☐

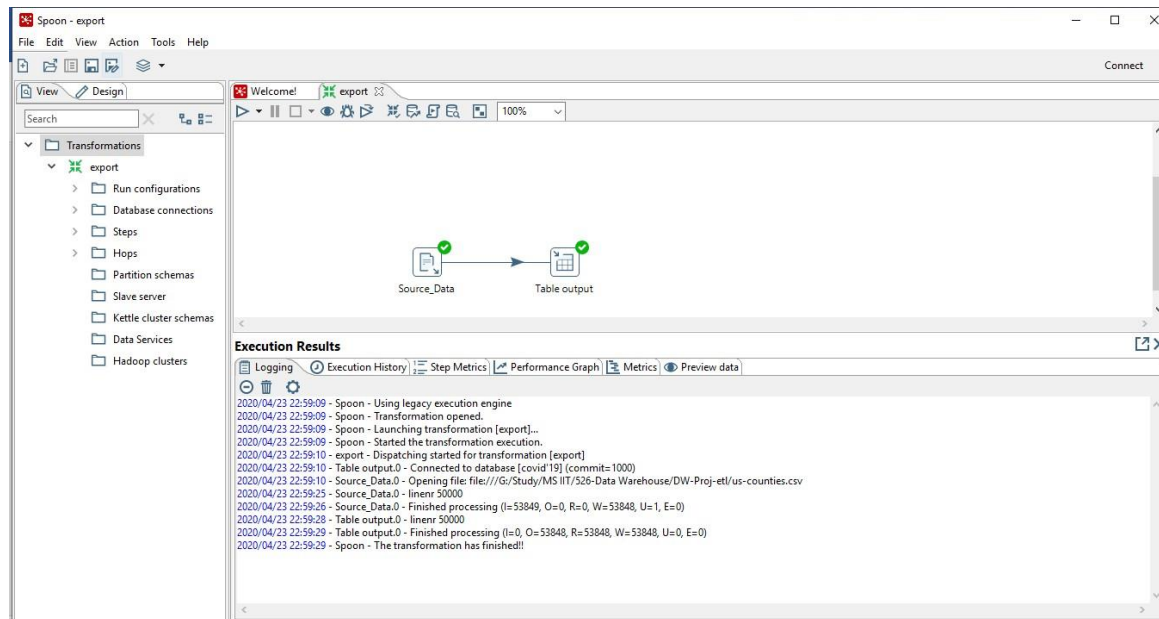
Hide assigned source fields? ☒ Hide assigned target fields? ☒

OK Guess Cancel

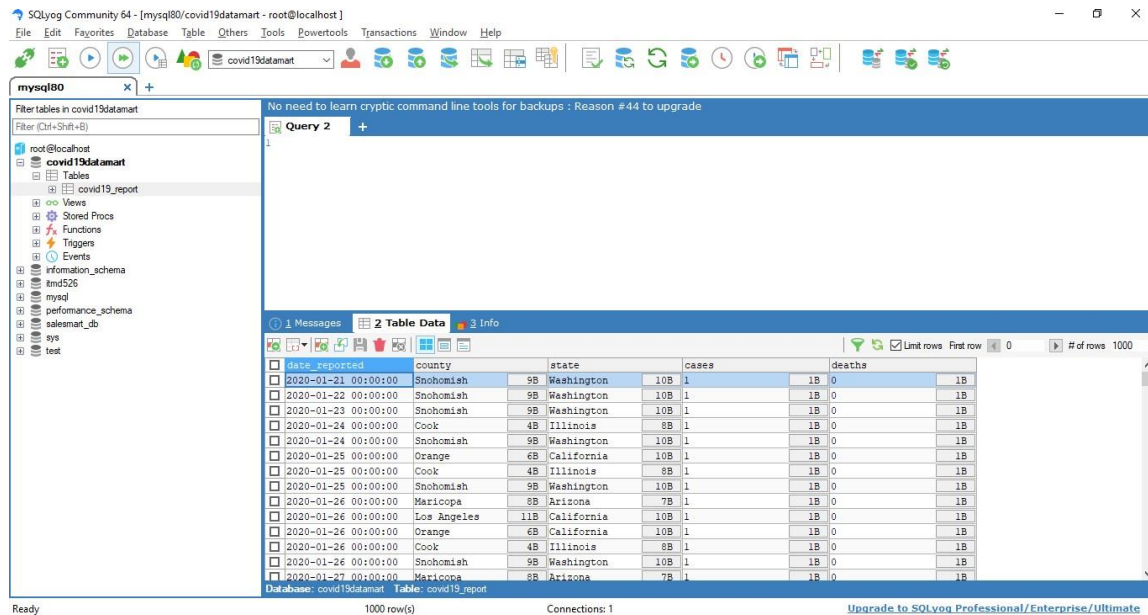
Source	Target
4 cases	cases
5 deaths	deaths

[Help] [OK] [Cancel] [SQL]

Mapped Source fields and Sql_Table fields i.e. target table "COVID19_report"



After running successful transformation source data is pushed into sql table.



Data pushed into table using KTR file.



SQLyog Community 64 - [mysql80/covid19datamart - root@localhost*]

File Edit Favorites Database Table Others Tools PowerTools Transactions Window Help

mysql80

Filter tables in covid19datamart

root@localhost

covid19datamart

Tables

covid19_report

covid19_state_daily_report

Views

Stored Procs

Functions

Triggers

Events

information_schema

mysql

performance_schema

sys

test

Use HTTP/SSH Tunneling to connect to MySQL even if direct connection is disallowed : Reason #36 to upgrade

Query 1

```

1 DROP TABLE IF EXISTS COVID19_state_daily_report;
2
3 CREATE TABLE COVID19_state_daily_report
4 AS
5 SELECT TRIM(state) AS state, date_reported, IFNULL(SUM(cases),0) AS tot_cases, IFNULL(SUM(deaths),0) AS tot_deaths
6 FROM covid19_report
7 WHERE 1=1
8 AND state IS NOT NULL
9 GROUP BY state,date_reported;

```

1 Messages 2 Table Data 3 Info

state	date_reported	tot_cases	tot_deaths
Alabama	2020-03-13 00:00:00	6	0
Alabama	2020-03-14 00:00:00	12	0
Alabama	2020-03-15 00:00:00	23	0
Alabama	2020-03-16 00:00:00	29	0
Alabama	2020-03-17 00:00:00	39	0
Alabama	2020-03-18 00:00:00	51	0
Alabama	2020-03-19 00:00:00	78	0
Alabama	2020-03-20 00:00:00	106	0
Alabama	2020-03-21 00:00:00	131	0
Alabama	2020-03-22 00:00:00	157	0
Alabama	2020-03-23 00:00:00	196	0
Alabama	2020-03-24 00:00:00	242	0
Alabama	2020-03-25 00:00:00	386	1
Alabama	2020-03-26 00:00:00	538	3

Database: covid19datamart Table: covid19_state_daily_report

ObjectBrowser Refreshed 1000 row(s) Connections: 1 Upgrade to SQLyog Professional/Enterprise/Ultimate

Above query will create new table “COVID19_state_daily_report” and grouped cases according to state and date_reported to give total count of cases/deaths confirmed on single day.

SQLyog Community 64 - [mysql80/covid19datamart - root@localhost*]

File Edit Favorites Database Table Others Tools PowerTools Transactions Window Help

mysql80

Filter tables in covid19datamart

root@localhost

covid19datamart

Tables

covid19_report

covid19_state_daily_report_seq

covid19_state_daily_report

Views

Stored Procs

Functions

Triggers

Events

information_schema

mysql

performance_schema

sys

test

Query Profiler reports the exact time spent in each step of query execution : Reason #56 to upgrade

Query 1

```

1 DROP TABLE IF EXISTS covid19_state_daily_report_seq;
2
3 CREATE TABLE covid19_state_daily_report_seq
4 AS
5 SELECT TRIM(a.state) AS state, a.date_reported, a.tot_cases,a.tot_deaths, COUNT(*) state_seq
6 FROM covid19_state_daily_report a, covid19_state_daily_report b
7 WHERE a.state= b.state
8 AND a.date_reported >= b.date_reported
9 GROUP BY a.state, a.date_reported;

```

1 Messages 2 Table Data 3 Info

state	date_reported	tot_cases	tot_deaths	state_seq
Alabama	2020-03-13 00:00:00	6	0	1
Alabama	2020-03-14 00:00:00	12	0	2
Alabama	2020-03-15 00:00:00	23	0	3
Alabama	2020-03-16 00:00:00	29	0	4
Alabama	2020-03-17 00:00:00	39	0	5
Alabama	2020-03-18 00:00:00	51	0	6
Alabama	2020-03-19 00:00:00	78	0	7
Alabama	2020-03-20 00:00:00	106	0	8
Alabama	2020-03-21 00:00:00	131	0	9
Alabama	2020-03-22 00:00:00	157	0	10
Alabama	2020-03-23 00:00:00	196	0	11
Alabama	2020-03-24 00:00:00	242	0	12
Alabama	2020-03-25 00:00:00	386	1	13
Alabama	2020-03-26 00:00:00	538	3	14

Database: covid19datamart Table: covid19_state_daily_report_seq

ObjectBrowser Refreshed 1000 row(s) Connections: 1 Upgrade to SQLyog Professional/Enterprise/Ultimate

Above query will generate state_seq in covid19_state_daily_report_seq table.



SQLyog Community 64 - [mysql80/covid19datamart - root@localhost*]

File Edit Favorites Database Table Others Tools PowerTools Transactions Window Help

mysql80

Filter tables in covid19datamart

root@localhost

covid19datamart

Tables

- covid19_report
- covid19_state_daily_report_seq
- covid19_state_cumulative_report
- covid19_state_daily_report

Views

Stored Procs

Functions

Triggers

Events

information_schema

mysql526

mysql

performance_schema

salesmart_db

sys

test

Write queries 10x faster using Smart Autocomplete : Reason #14 to upgrade

Query 1

```

1 DROP TABLE IF EXISTS covid19_state_cumulative_report;
2
3 CREATE TABLE covid19_state_cumulative_report
4 AS
5 SELECT a.state,a.date_reported,a.state_seq, a.tot_cases AS newly_confirmed_cases,SUM(b.tot_cases) AS tot_cases_till_date,a.tot_deaths AS newly_co
6 FROM covid19_state_daily_report_seq AS a
7 INNER JOIN covid19_state_daily_report_seq AS b ON (a.state_seq=b.state_seq) AND (a.state=b.state)
8 GROUP BY a.state,a.tot_cases,a.state_seq;

```

1 Messages 2 Table Data Info

state	date_reported	state_seq	newly_confirmed_cases	tot_cases_till_date	newly_confirmed_deaths	tot_deaths_till_date
Alabama	2020-03-13 00:00:00	1	6	6	0	0
Alabama	2020-03-14 00:00:00	2	12	18	0	0
Alabama	2020-03-15 00:00:00	3	23	41	0	0
Alabama	2020-03-16 00:00:00	4	29	70	0	0
Alabama	2020-03-17 00:00:00	5	39	109	0	0
Alabama	2020-03-18 00:00:00	6	51	160	0	0
Alabama	2020-03-19 00:00:00	7	78	238	0	0
Alabama	2020-03-20 00:00:00	8	106	344	0	0
Alabama	2020-03-21 00:00:00	9	131	475	0	0
Alabama	2020-03-22 00:00:00	10	157	632	0	0
Alabama	2020-03-23 00:00:00	11	196	828	0	0
Alabama	2020-03-24 00:00:00	12	242	1070	0	0
Alabama	2020-03-25 00:00:00	13	396	1466	1	1
Alabama	2020-03-26 00:00:00	14	538	1994	3	4

Database: covid19datamart Table: covid19_state_cumulative_report

ObjectBrowser Refreshed 1000 row(s) Connections: 1 Upgrade to SQLyog Professional/Enterprise/Ultimate

With help of seq generated above query will help to give cumulative_report per state. Which is helpful for visualization purpose.

SQLyog Community 64 - [mysql80/covid19datamart - root@localhost*]

File Edit Favorites Database Table Others Tools PowerTools Transactions Window Help

mysql80

Filter tables in covid19datamart

root@localhost

covid19datamart

Tables

- covid19_report
- covid19_state_daily_report
- covid19_state_seq
- covid19_state_total_cases_deaths

Views

Stored Procs

Functions

Triggers

Events

information_schema

mysql526

mysql

performance_schema

salesmart_db

sys

test

Query Profiler reports the exact time spent in each step of query execution : Reason #56 to upgrade

Query 1

```

1 DROP TABLE IF EXISTS COVID19_state_total_cases_deaths;
2
3 CREATE TABLE COVID19_state_total_cases_deaths
4 AS
5 SELECT TRIM(state) AS state,IFNULL(SUM(tot_cases),0) AS tot_cases_per_state, IFNULL(SUM(tot_deaths),0) AS tot_deaths_per_state
6 FROM COVID19_state_daily_report
7 WHERE 1=1
8 AND state IS NOT NULL
9 GROUP BY state;

```

1 Messages 2 Table Data Info

state	tot_cases_per_state	tot_deaths_per_state
Alabama	32909	757
Alaska	3278	60
Arizona	36765	934
Arkansas	15980	268
California	258570	6315
Colorado	82599	2456
Connecticut	105188	3661
Delaware	12830	269
District of Columbia	17798	345
Florida	204200	3879
Georgia	125449	4376
Guam	5000	59
Hawaii	6161	59
Idaho	16310	232

Database: covid19datamart Table: covid19_state_total_cases_deaths

Ready 55 row(s) Connections: 1 Upgrade to SQLyog Professional/Enterprise/Ultimate

Above query will give total number of cases/deaths for each state till date.

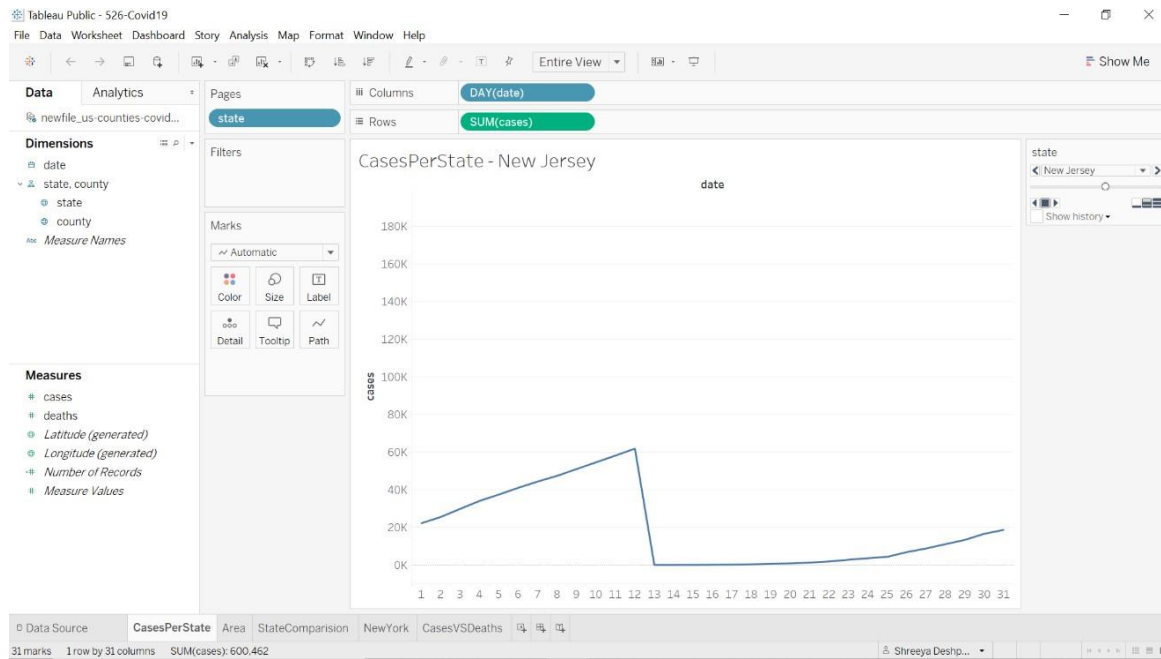
This, data will also be required for Visualization purpose.



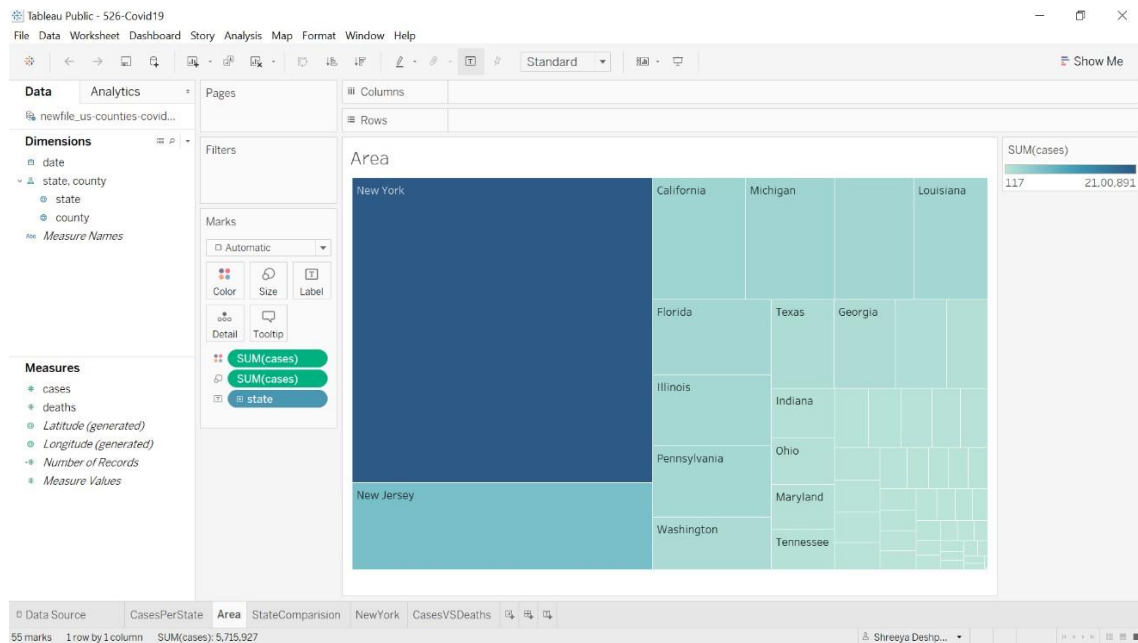
Dashboard Implementation (Visualization)

The visualization has been done in tableau. The CSV file generated after preprocessing was used for visualizing. The different graphs generated are shown below:

- Sum of Cases day-wise in different states:

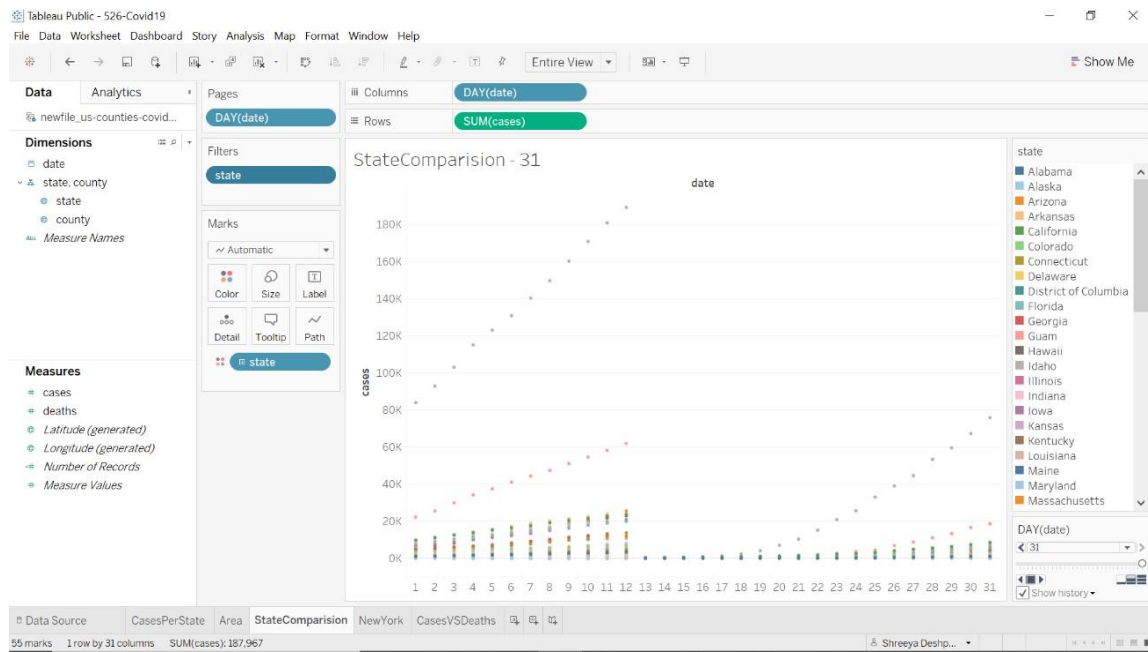


- Area Graph for comparing state-wise total cases:

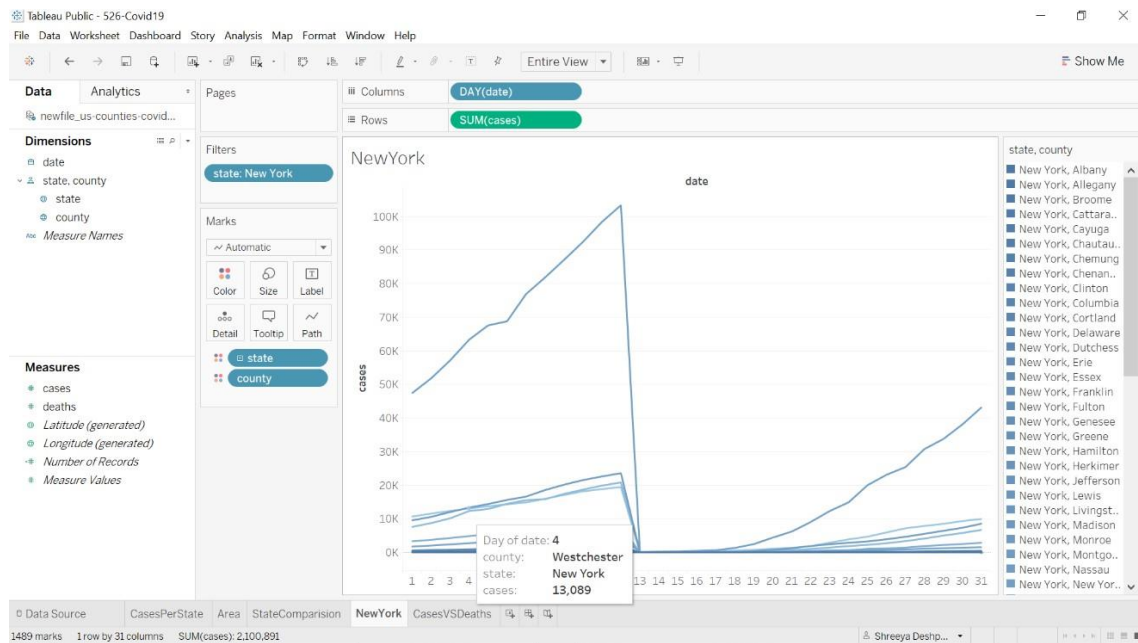




- Comparison of cases in all States day-wise:

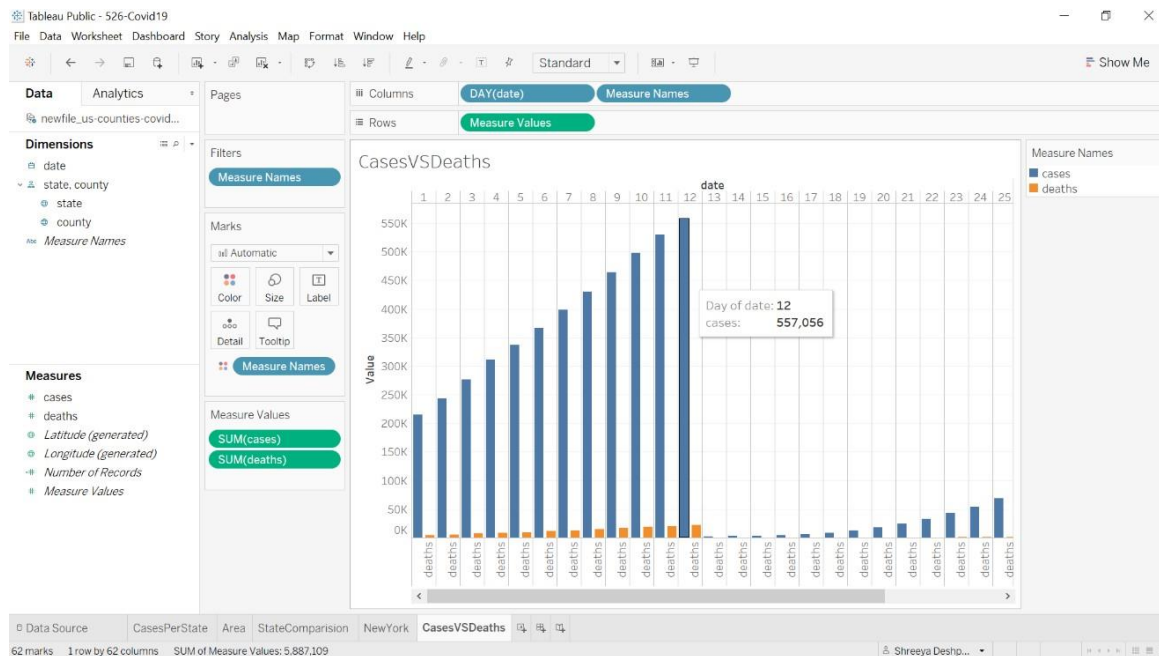


- Countywise cases in New York State:





- Comparison of total cases and deaths day-wise:



Analysis of data

- Deaths across America spiked as Covid-19 began its spread, and many were never attributed to the new coronavirus. Our analysis includes the data between 21st January to 12th April 2020. The data is from Kaggle website.
- Notable increases in deaths were seen in March and early April. This was especially true in New York and New Jersey, states hard-hit by the pandemic. Using data from this website we found that 15,000 excess deaths from March 1 to April 4.
- In the state of New York in county Westchester on day 4, we could see total 13,089 cases which are highest in comparison with all other county.
- After visualizing the data, we could see the number of cases day by day.
- We could measure and compare number of deaths per identified positive cases which is not too much.
- That means people are recovering from this covid 19 even though the cases are increasing on daily basis.