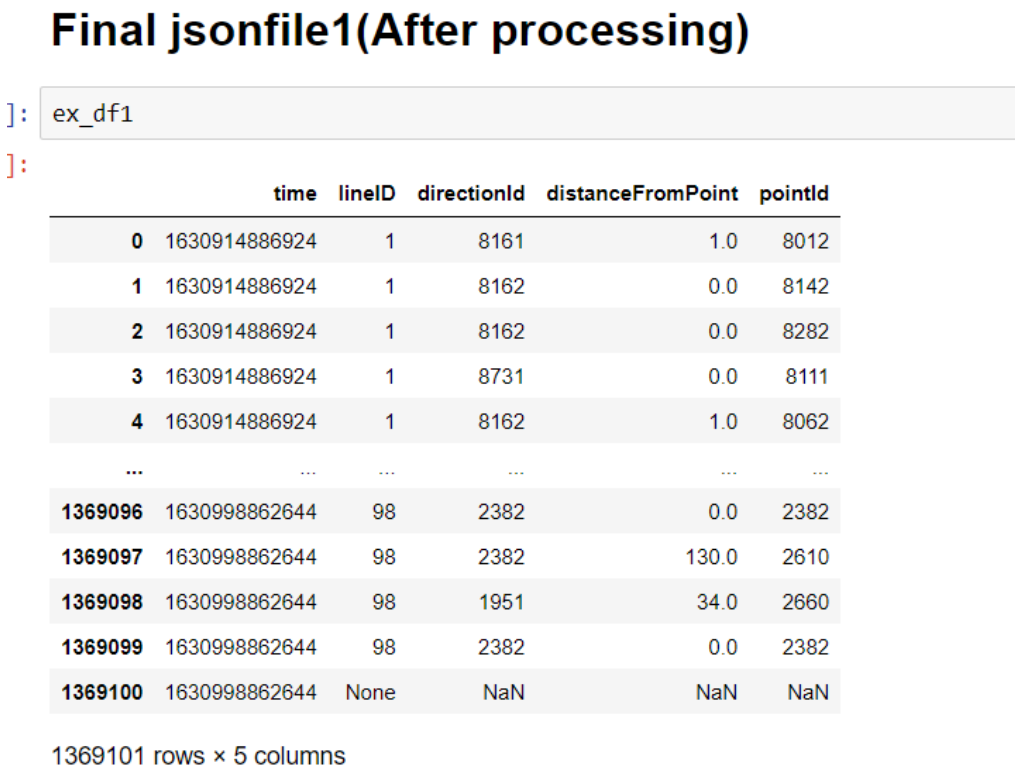
Task 1: Segregate the data based on line id and day

1. Preprocessing:

1. As the data is a nested json file I have to normalize it first.
2. I used pd.json_normalize() for normalizing the dictionary values and explode() to extract all the elements in a list.
3. I dropped all the unnecessary columns which had been created after applying pd.json_normalize().
4. Finally I renamed the column to avoid large column names.



Final jsonfile1(After processing)

```
]: ex_df1
```

]:

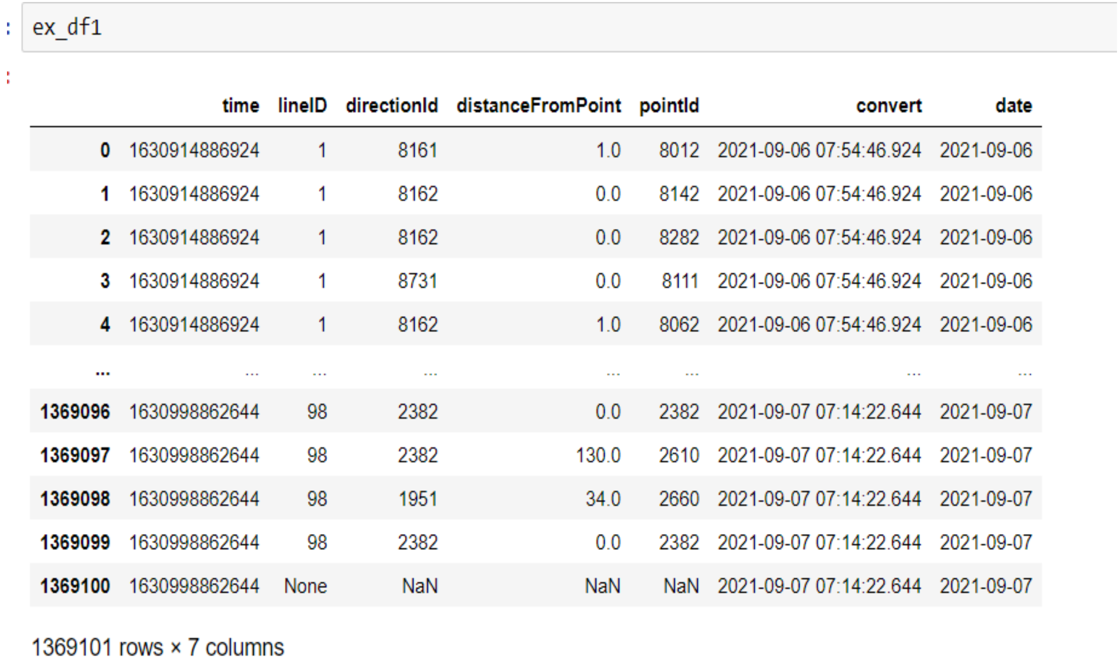
	time	lineID	directionId	distanceFromPoint	pointId
0	1630914886924	1	8161	1.0	8012
1	1630914886924	1	8162	0.0	8142
2	1630914886924	1	8162	0.0	8282
3	1630914886924	1	8731	0.0	8111
4	1630914886924	1	8162	1.0	8062
...	...	...	...	...	...
1369096	1630998862644	98	2382	0.0	2382
1369097	1630998862644	98	2382	130.0	2610
1369098	1630998862644	98	1951	34.0	2660
1369099	1630998862644	98	2382	0.0	2382
1369100	1630998862644	None	NaN	NaN	NaN

1369101 rows × 5 columns

Fig-1 : Dataframe after all necessary pre processing

2. Extract Date and Time from the above time column

1. I used pd.to_datetime() to extract the date from the unix timestamp value.
2. I saved the date value in the date column



```
ex_df1
```

	time	lineID	directionId	distanceFromPoint	pointId	convert	date
0	1630914886924	1	8161	1.0	8012	2021-09-06 07:54:46.924	2021-09-06
1	1630914886924	1	8162	0.0	8142	2021-09-06 07:54:46.924	2021-09-06
2	1630914886924	1	8162	0.0	8282	2021-09-06 07:54:46.924	2021-09-06
3	1630914886924	1	8731	0.0	8111	2021-09-06 07:54:46.924	2021-09-06
4	1630914886924	1	8162	1.0	8062	2021-09-06 07:54:46.924	2021-09-06
...	...	...	...	...	...	...	...
1369096	1630998862644	98	2382	0.0	2382	2021-09-07 07:14:22.644	2021-09-07
1369097	1630998862644	98	2382	130.0	2610	2021-09-07 07:14:22.644	2021-09-07
1369098	1630998862644	98	1951	34.0	2660	2021-09-07 07:14:22.644	2021-09-07
1369099	1630998862644	98	2382	0.0	2382	2021-09-07 07:14:22.644	2021-09-07
1369100	1630998862644	None	NaN	NaN	NaN	2021-09-07 07:14:22.644	2021-09-07

1369101 rows × 7 columns

Fig-2: Dataframe after extracting date column

3. Segregating the data based on line ID and Date

1. At first I applied groupby() on the date column. I used the date value as the parent directory.
2. Then I applied the groupby (‘lineID’) on the resultant rows that had been retrieved in the previous step. I created a separate csv file for each lineID value under the parent directory (date). The excel files can be find in the Results folder in the Github link (https://github.com/afsanamimii/Ques1-Vehicle_Data_analysis)

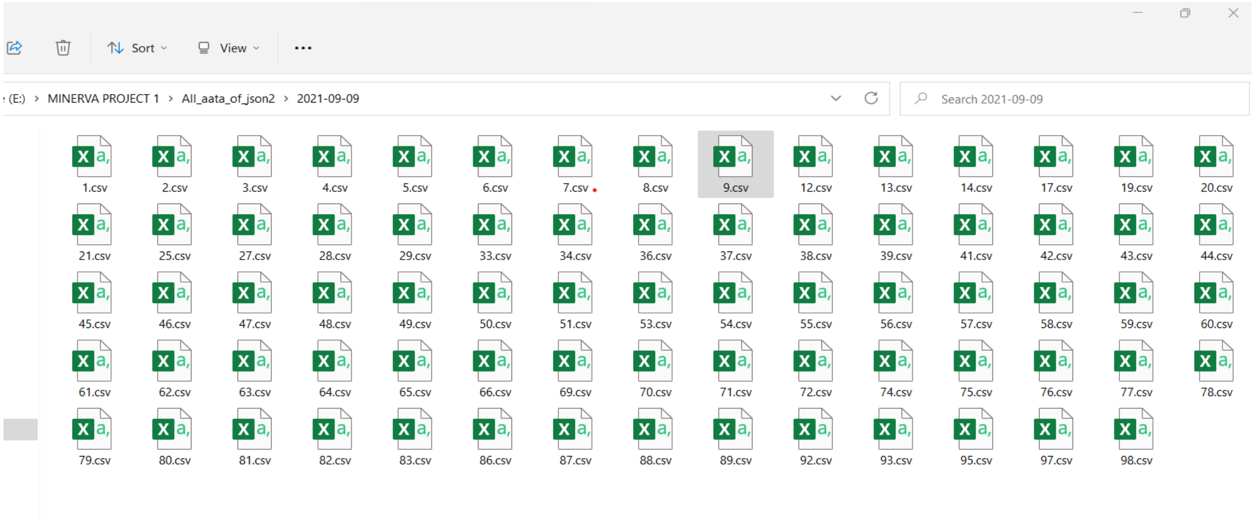
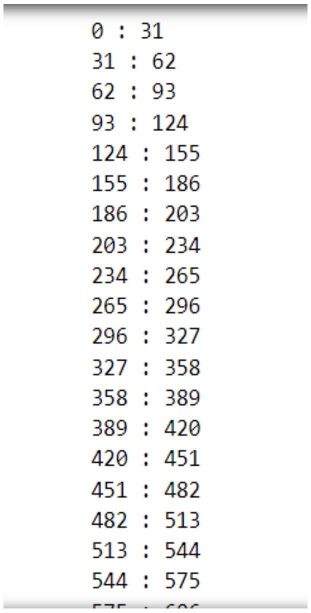

Sort
View
(E:) › MINERVA PROJECT 1 › All_aata_of_json2 › 2021-09-09
Search 2021-09-09
1.csv 2.csv 3.csv 4.csv 5.csv 6.csv 7.csv 8.csv 9.csv 12.csv 13.csv 14.csv 17.csv 19.csv 20.csv
21.csv 25.csv 27.csv 28.csv 29.csv 33.csv 34.csv 36.csv 37.csv 38.csv 39.csv 41.csv 42.csv 43.csv 44.csv
45.csv 46.csv 47.csv 48.csv 49.csv 50.csv 51.csv 53.csv 54.csv 55.csv 56.csv 57.csv 58.csv 59.csv 60.csv
61.csv 62.csv 63.csv 64.csv 65.csv 66.csv 69.csv 70.csv 71.csv 72.csv 74.csv 75.csv 76.csv 77.csv 78.csv
79.csv 80.csv 81.csv 82.csv 83.csv 86.csv 87.csv 88.csv 89.csv 92.csv 93.csv 95.csv 97.csv 98.csv

Fig-3 : Final output

Task2: Identify the vehicle id which is missing here

Steps:

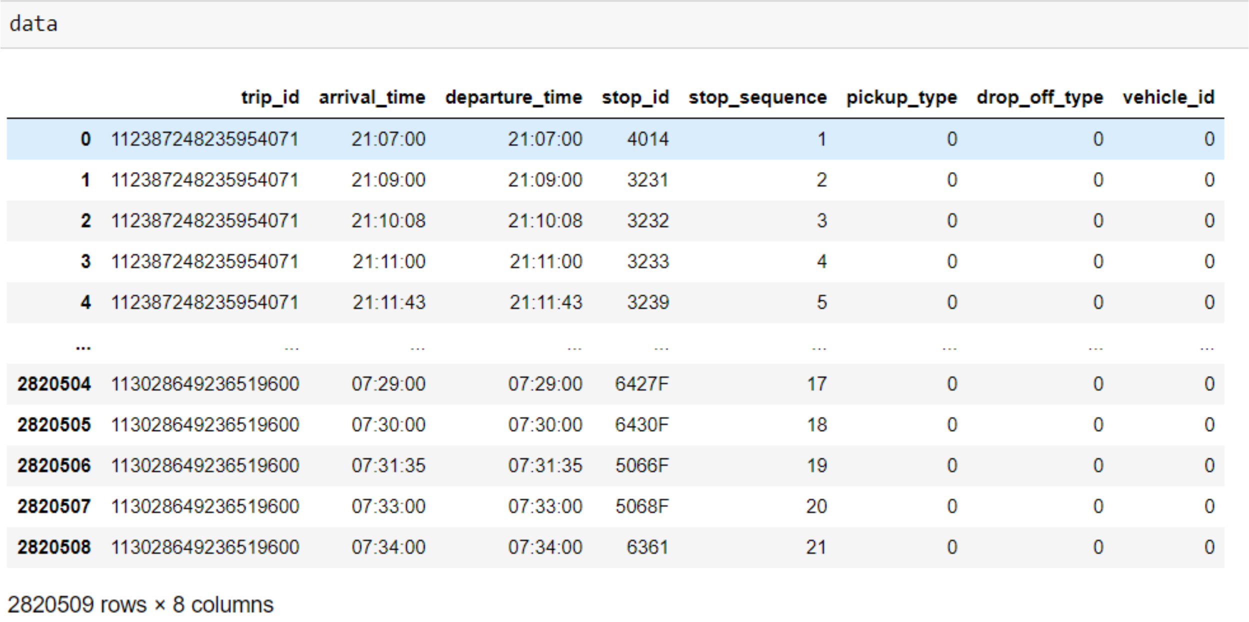
1. In the stop_sequnce column the sequence of each unique vehicle is given. For example, In the first 31 rows the value of stop_sequnce is 1 to 31. For the 32th row the value of stop_sequnce value again starts from 1. So I am assuming for the 1st 31th row a particular vehicle will be stopped in different stops for 31 times. So I will assign a unique Vehicle Id for those 31 rows assuming that those 31 rows contain the information of the same vehicle.
2. Using the same logic I sliced the main dataframe by finding out the start index and end index of a dataframe.



```
0 : 31
31 : 62
62 : 93
93 : 124
124 : 155
155 : 186
186 : 203
203 : 234
234 : 265
265 : 296
296 : 327
327 : 358
358 : 389
389 : 420
420 : 451
451 : 482
482 : 513
513 : 544
544 : 575
575 : 606
```

Fig-4 : slice of Dataframe(Start index: End index)

3. For finding the index I loop through the whole dataset and pick the index number of those where “stop_sequnce”==1.
4. I created a Vehicle_ID column where all values will be 0 initially.



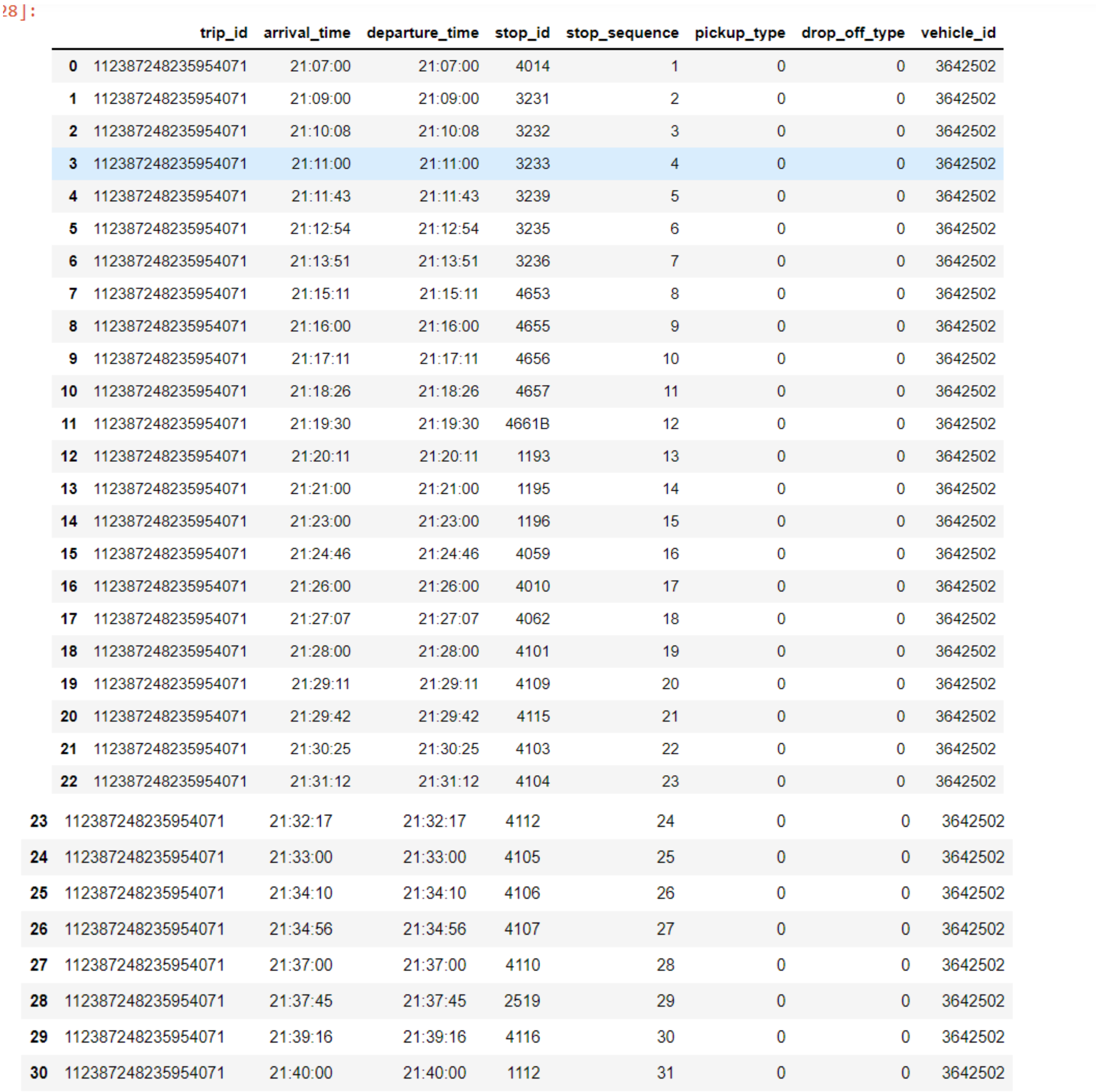
```
data
```

	trip_id	arrival_time	departure_time	stop_id	stop_sequence	pickup_type	drop_off_type	vehicle_id
0	112387248235954071	21:07:00	21:07:00	4014	1	0	0	0
1	112387248235954071	21:09:00	21:09:00	3231	2	0	0	0
2	112387248235954071	21:10:08	21:10:08	3232	3	0	0	0
3	112387248235954071	21:11:00	21:11:00	3233	4	0	0	0
4	112387248235954071	21:11:43	21:11:43	3239	5	0	0	0
...	...	...	...	...	...	...	...	...
2820504	113028649236519600	07:29:00	07:29:00	6427F	17	0	0	0
2820505	113028649236519600	07:30:00	07:30:00	6430F	18	0	0	0
2820506	113028649236519600	07:31:35	07:31:35	5066F	19	0	0	0
2820507	113028649236519600	07:33:00	07:33:00	5068F	20	0	0	0
2820508	113028649236519600	07:34:00	07:34:00	6361	21	0	0	0

2820509 rows × 8 columns

Fig-5: Initial value of Vehicle_id

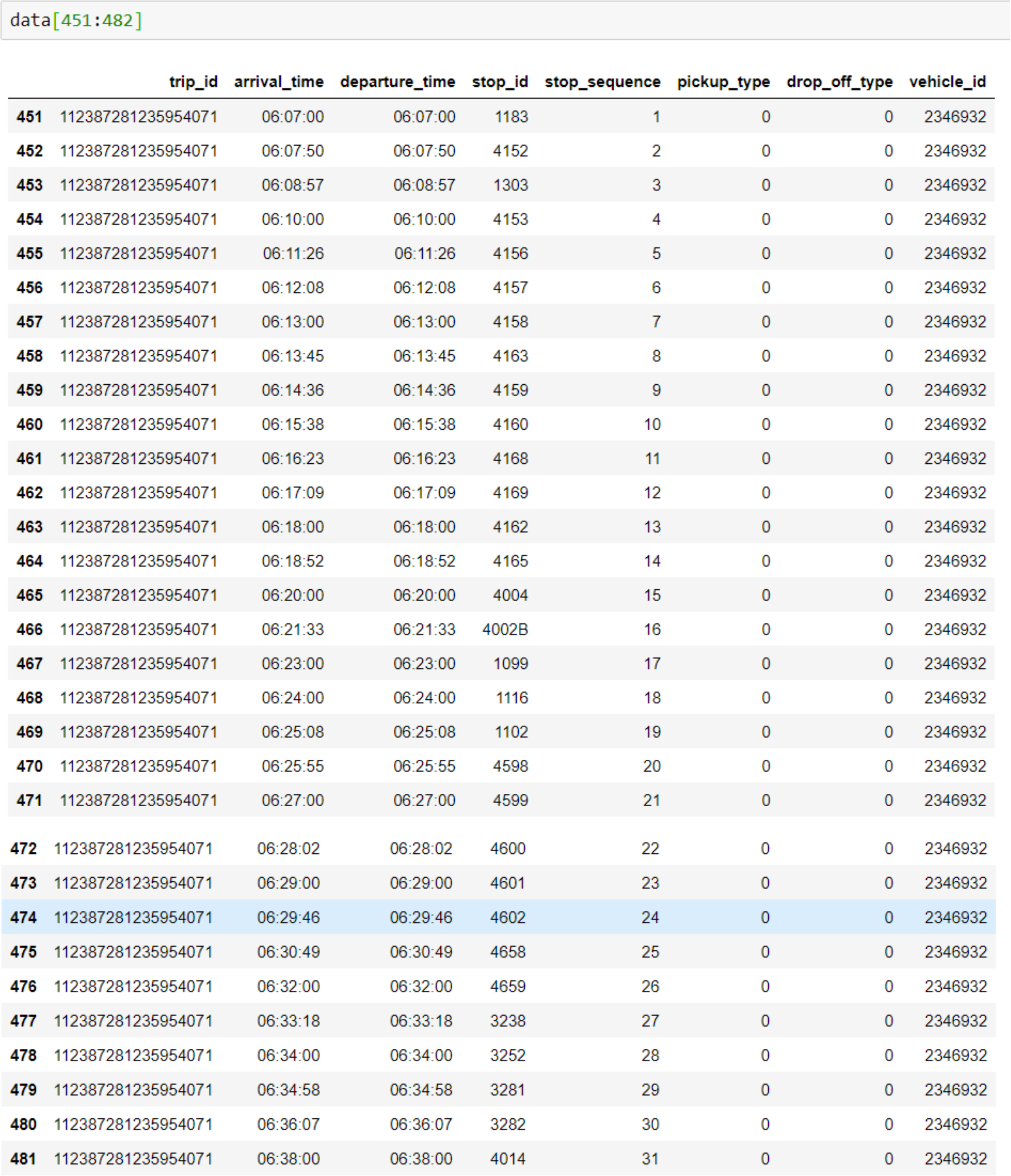
5. After that I put all the indexes values retrieved from step3 in a list and loop through the length of list and assign a Unique random value for a particular slice (For example Assign a random value in DATA[0] TO DATA[31] as they means the same vehicle according to my logic) in the Vehicle_ID column. So these 31 rows will have the same value as they define one vehicle.



	trip_id	arrival_time	departure_time	stop_id	stop_sequence	pickup_type	drop_off_type	vehicle_id
0	112387248235954071	21:07:00	21:07:00	4014	1	0	0	3642502
1	112387248235954071	21:09:00	21:09:00	3231	2	0	0	3642502
2	112387248235954071	21:10:08	21:10:08	3232	3	0	0	3642502
3	112387248235954071	21:11:00	21:11:00	3233	4	0	0	3642502
4	112387248235954071	21:11:43	21:11:43	3239	5	0	0	3642502
5	112387248235954071	21:12:54	21:12:54	3235	6	0	0	3642502
6	112387248235954071	21:13:51	21:13:51	3236	7	0	0	3642502
7	112387248235954071	21:15:11	21:15:11	4653	8	0	0	3642502
8	112387248235954071	21:16:00	21:16:00	4655	9	0	0	3642502
9	112387248235954071	21:17:11	21:17:11	4656	10	0	0	3642502
10	112387248235954071	21:18:26	21:18:26	4657	11	0	0	3642502
11	112387248235954071	21:19:30	21:19:30	4661B	12	0	0	3642502
12	112387248235954071	21:20:11	21:20:11	1193	13	0	0	3642502
13	112387248235954071	21:21:00	21:21:00	1195	14	0	0	3642502
14	112387248235954071	21:23:00	21:23:00	1196	15	0	0	3642502
15	112387248235954071	21:24:46	21:24:46	4059	16	0	0	3642502
16	112387248235954071	21:26:00	21:26:00	4010	17	0	0	3642502
17	112387248235954071	21:27:07	21:27:07	4062	18	0	0	3642502
18	112387248235954071	21:28:00	21:28:00	4101	19	0	0	3642502
19	112387248235954071	21:29:11	21:29:11	4109	20	0	0	3642502
20	112387248235954071	21:29:42	21:29:42	4115	21	0	0	3642502
21	112387248235954071	21:30:25	21:30:25	4103	22	0	0	3642502
22	112387248235954071	21:31:12	21:31:12	4104	23	0	0	3642502
23	112387248235954071	21:32:17	21:32:17	4112	24	0	0	3642502
24	112387248235954071	21:33:00	21:33:00	4105	25	0	0	3642502
25	112387248235954071	21:34:10	21:34:10	4106	26	0	0	3642502
26	112387248235954071	21:34:56	21:34:56	4107	27	0	0	3642502
27	112387248235954071	21:37:00	21:37:00	4110	28	0	0	3642502
28	112387248235954071	21:37:45	21:37:45	2519	29	0	0	3642502
29	112387248235954071	21:39:16	21:39:16	4116	30	0	0	3642502
30	112387248235954071	21:40:00	21:40:00	1112	31	0	0	3642502

Fig-6: Assign a vehicle Id for a particular sequence

6. In fig-7 I’m attaching another seq where the vehicle Id will be different. Here the vehicle_Id value is different from fig-6 as it is a different sequence.



```
data[451:482]
```

	trip_id	arrival_time	departure_time	stop_id	stop_sequence	pickup_type	drop_off_type	vehicle_id
451	112387281235954071	06:07:00	06:07:00	1183	1	0	0	2346932
452	112387281235954071	06:07:50	06:07:50	4152	2	0	0	2346932
453	112387281235954071	06:08:57	06:08:57	1303	3	0	0	2346932
454	112387281235954071	06:10:00	06:10:00	4153	4	0	0	2346932
455	112387281235954071	06:11:26	06:11:26	4156	5	0	0	2346932
456	112387281235954071	06:12:08	06:12:08	4157	6	0	0	2346932
457	112387281235954071	06:13:00	06:13:00	4158	7	0	0	2346932
458	112387281235954071	06:13:45	06:13:45	4163	8	0	0	2346932
459	112387281235954071	06:14:36	06:14:36	4159	9	0	0	2346932
460	112387281235954071	06:15:38	06:15:38	4160	10	0	0	2346932
461	112387281235954071	06:16:23	06:16:23	4168	11	0	0	2346932
462	112387281235954071	06:17:09	06:17:09	4169	12	0	0	2346932
463	112387281235954071	06:18:00	06:18:00	4162	13	0	0	2346932
464	112387281235954071	06:18:52	06:18:52	4165	14	0	0	2346932
465	112387281235954071	06:20:00	06:20:00	4004	15	0	0	2346932
466	112387281235954071	06:21:33	06:21:33	4002B	16	0	0	2346932
467	112387281235954071	06:23:00	06:23:00	1099	17	0	0	2346932
468	112387281235954071	06:24:00	06:24:00	1116	18	0	0	2346932
469	112387281235954071	06:25:08	06:25:08	1102	19	0	0	2346932
470	112387281235954071	06:25:55	06:25:55	4598	20	0	0	2346932
471	112387281235954071	06:27:00	06:27:00	4599	21	0	0	2346932
472	112387281235954071	06:28:02	06:28:02	4600	22	0	0	2346932
473	112387281235954071	06:29:00	06:29:00	4601	23	0	0	2346932
474	112387281235954071	06:29:46	06:29:46	4602	24	0	0	2346932
475	112387281235954071	06:30:49	06:30:49	4658	25	0	0	2346932
476	112387281235954071	06:32:00	06:32:00	4659	26	0	0	2346932
477	112387281235954071	06:33:18	06:33:18	3238	27	0	0	2346932
478	112387281235954071	06:34:00	06:34:00	3252	28	0	0	2346932
479	112387281235954071	06:34:58	06:34:58	3281	29	0	0	2346932
480	112387281235954071	06:36:07	06:36:07	3282	30	0	0	2346932
481	112387281235954071	06:38:00	06:38:00	4014	31	0	0	2346932

Fig-7: New random value for a new data slice