

## Air China RPK Analysis

Goal: Using data scraping tools in Python and R to perform ETL process, then compute year over year growth and deploy visualization using Tableau.

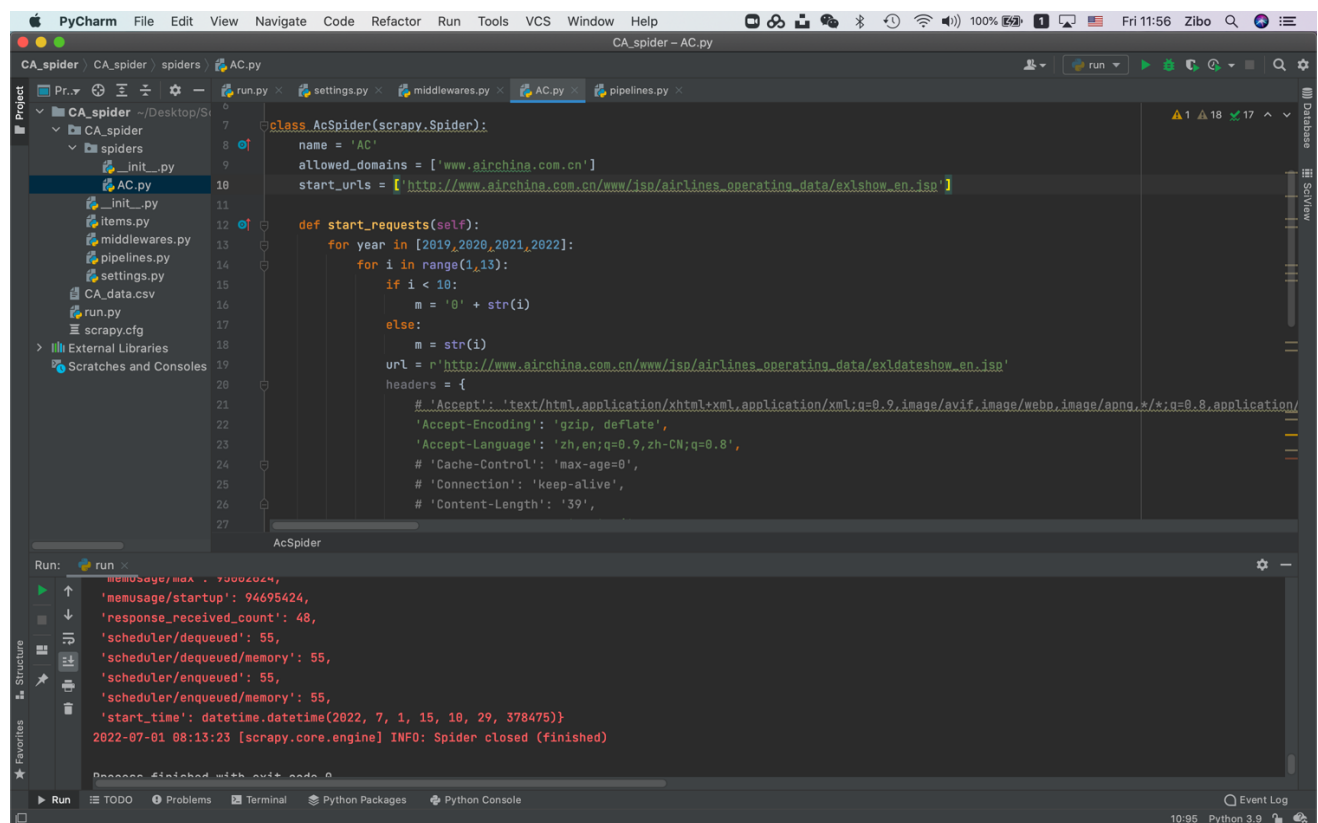
Raw Data storage type:

- Air China: html

[http://www.airchina.com.cn/en/investor\\_relations/traffic\\_data.shtml](http://www.airchina.com.cn/en/investor_relations/traffic_data.shtml)

Steps:

1. Scraping needed data from Air China official website by using Python Scrapy.



The screenshot displays the PyCharm IDE interface. The main editor shows the code for a Scrapy spider named `AcSpider` in `AC.py`. The code defines the spider's name, allowed domains, and start URLs. It includes a `start_requests` method that iterates over years from 2019 to 2022 and generates requests for each year. The spider is configured with various headers, including `Accept`, `Accept-Encoding`, `Accept-Language`, `Cache-Control`, `Connection`, and `Content-Length`.

The Run console at the bottom shows the execution output of the spider. It includes statistics such as `memusage/max`, `memusage/startup`, `response_received_count`, `scheduler/dequeued`, `scheduler/dequeued/memory`, `scheduler/enqueued`, and `scheduler/enqueued/memory`. The output also shows the start time and a message indicating that the spider closed successfully.

```
class AcSpider(scrapy.Spider):
    name = 'AC'
    allowed_domains = ['www.airchina.com.cn']
    start_urls = ['http://www.airchina.com.cn/www/isp/airlines_operating_data/exlshow_en.jsp']

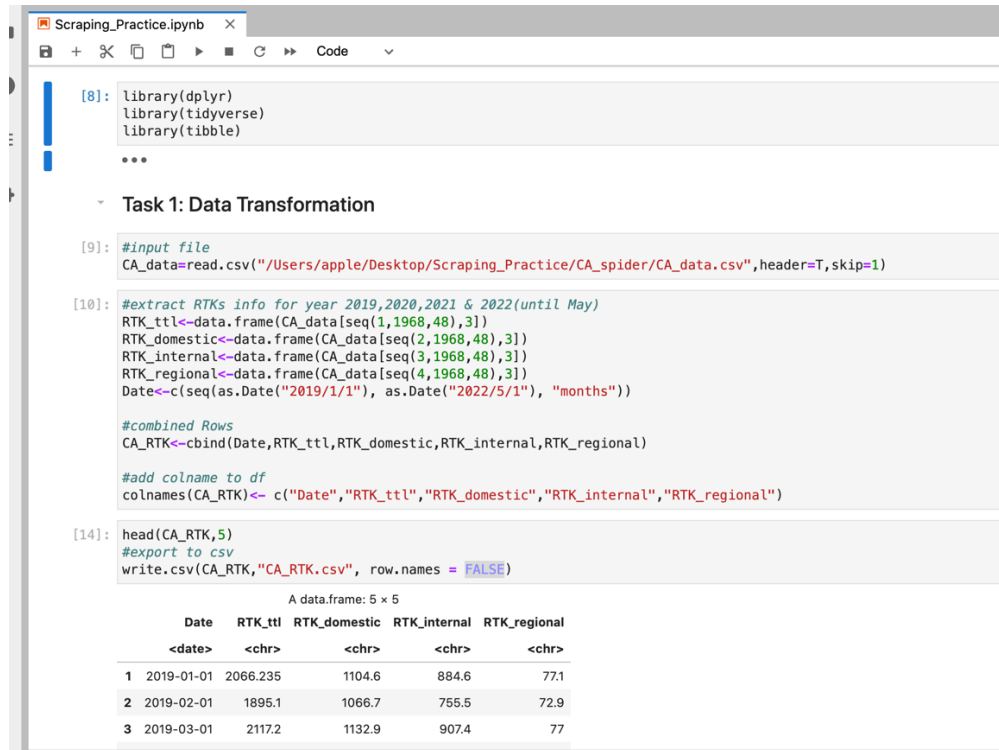
    def start_requests(self):
        for year in [2019, 2020, 2021, 2022]:
            for i in range(1, 13):
                if i < 10:
                    m = '0' + str(i)
                else:
                    m = str(i)
                url = r'http://www.airchina.com.cn/www/isp/airlines_operating_data/exlshow_en.jsp'
                headers = {
                    # 'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/
                    'Accept-Encoding': 'gzip, deflate',
                    'Accept-Language': 'zh,en;q=0.9,zh-CN;q=0.8',
                    # 'Cache-Control': 'max-age=0',
                    # 'Connection': 'keep-alive',
                    # 'Content-Length': '39',
                }
```

```
memusage/max: 100000000,
memusage/startup: 94695424,
response_received_count: 48,
scheduler/dequeued: 55,
scheduler/dequeued/memory: 55,
scheduler/enqueued: 55,
scheduler/enqueued/memory: 55,
start_time: datetime.datetime(2022, 7, 1, 15, 10, 29, 378475)}
2022-07-01 08:13:23 [scrapy.core.engine] INFO: Spider closed (finished)
Process finished with quit code 0
```

Data from 2019 to 2022 May has been scraped, all the columns included output as “.csv”. Preview as below:

A1								
	A	B	C	D	E	F	G	H
1		Unnamed: 0	Jan~†2019	% change vs	% change vs	Cumulative	% cumulative change	
2	0	1.Traffic	1.Traffic	1.Traffic	1.Traffic	1.Traffic	1.Traffic	
3	1	1.RTKs (in m	2066.235	7.804	2.833	2066.235	7.8	
4	2	Domestic	1104.6	7.1	4.5	1104.6	7.1	
5	3	International	884.6	7.9	0.8	884.6	7.9	
6	4	Regional	77.1	18	3.1	77.1	18	
7	5	2.RPKs (in m	18930.6	10.9	6.1	18930.6	10.9	
8	6	Domestic	10960.5	8.9	6.2	10960.5	8.9	
9	7	International	7216.3	13	6.2	7216.3	13	
10	8	Regional	753.7	20.8	4.9	753.7	20.8	
11	9	3.RFTKs(in m	404.2	-0.3	-3.6	404.2	-0.3	
12	10	Domestic	144	0.9	5.1	144	0.9	
13	11	International	252.7	-1	-8	252.7	-1	
14	12	Regional	43466	-0.6	-3	7.5	-0.6	
15	13	4.Number o	9156.5	11.3	6.1	9156.5	11.3	
16	14	Domestic	7243.8	9.4	5.5	7243.8	9.4	
17	15	International	1441.1	18.7	9.5	1441.1	18.7	
18	16	Regional	471.6	20.7	4.6	471.6	20.7	
19	17	5.Total Carg	128179.4	2	0.3	128179.4	2	
20	18	Domestic	88339.6	1.1	3.5	88339.6	1.1	
21	19	International	35245.6	4.9	-6.7	35245.6	4.9	
22	20	Regional	4594.1	-2.9	-2.2	4594.1	-2.9	
23	21	2.Capacity	2.Capacity	2.Capacity	2.Capacity	2.Capacity	2.Capacity	
24	22	1.ATKs (in m	3131.8	9.9	5.5	3131.8	9.9	
25	23	Domestic	1647	10.2	6.4	1647	10.2	
26	24	International	1372.5	8.9	4.3	1372.5	8.9	
27	25	Regional	112.2	17	6	112.2	17	
28	26	2.ASKs (in m	23688.7	9.4	4.2	23688.7	9.4	
29	27	Domestic	13761.8	9.6	4.9	13761.8	9.6	
30	28	International	9006.6	8.4	3.2	9006.6	8.4	
31	29	Regional	920.4	16.1	4.2	920.4	16.1	
32	30	3.AFTKs (in r	994.7	11	8.4	994.7	11	
33	31	Domestic	408.8	12.3	11.6	408.8	12.3	
34	32	International	560.6	9.8	6	560.6	9.8	
35	33	Regional	25.3	18.9	12.7	25.3	18.9	
36	34	3.Load Factor	3.Load Factor	3.Load Factor	3.Load Factor	3.Load Factor	3.Load Factor	
37	35	1.Passenger	79.9	1.1	1.4	79.9	1.1	
38	36	Domestic	79.6	-0.5	1	79.6	-0.5	
39	37	International	80.1	3.2	2.3	80.1	3.2	
40	38	Regional	81.9	3.2	0.5	81.9	3.2	
41	39	2.Cargo load	40.6	-4.6	-5.1	40.6	-4.6	
42	40	Domestic	35.2	-4	-2.2	35.2	-4	

2. By using R Jupyter, extracting RTK, RPK for both domestic, international, and regional in each month in 2019, 2020, 2021 & 2022(until May).



```
[8]: library(dplyr)
library(tidyverse)
library(tibble)
...

Task 1: Data Transformation

[9]: #input file
CA_data=read.csv("/Users/apple/Desktop/Scraping_Practice/CA_spider/CA_data.csv",header=T,skip=1)

[10]: #extract RTKs info for year 2019,2020,2021 & 2022(until May)
RTK_ttl<-data.frame(CA_data[seq(1,1968,48),3])
RTK_domestic<-data.frame(CA_data[seq(2,1968,48),3])
RTK_internal<-data.frame(CA_data[seq(3,1968,48),3])
RTK_regional<-data.frame(CA_data[seq(4,1968,48),3])
Date<-c(seq(as.Date("2019/1/1"), as.Date("2022/5/1"), "months"))

#combined Rows
CA_RTK<-cbind(Date,RTK_ttl,RTK_domestic,RTK_internal,RTK_regional)

#add colname to df
colnames(CA_RTK)<- c("Date","RTK_ttl","RTK_domestic","RTK_internal","RTK_regional")

[14]: head(CA_RTK,5)
#export to csv
write.csv(CA_RTK,"CA_RTK.csv", row.names = FALSE)
```

A data.frame: 5 × 5

	Date	RTK_ttl	RTK_domestic	RTK_internal	RTK_regional
	<date>	<chr>	<chr>	<chr>	<chr>
1	2019-01-01	2066.235	1104.6	884.6	77.1
2	2019-02-01	1895.1	1066.7	755.5	72.9
3	2019-03-01	2117.2	1132.9	907.4	77

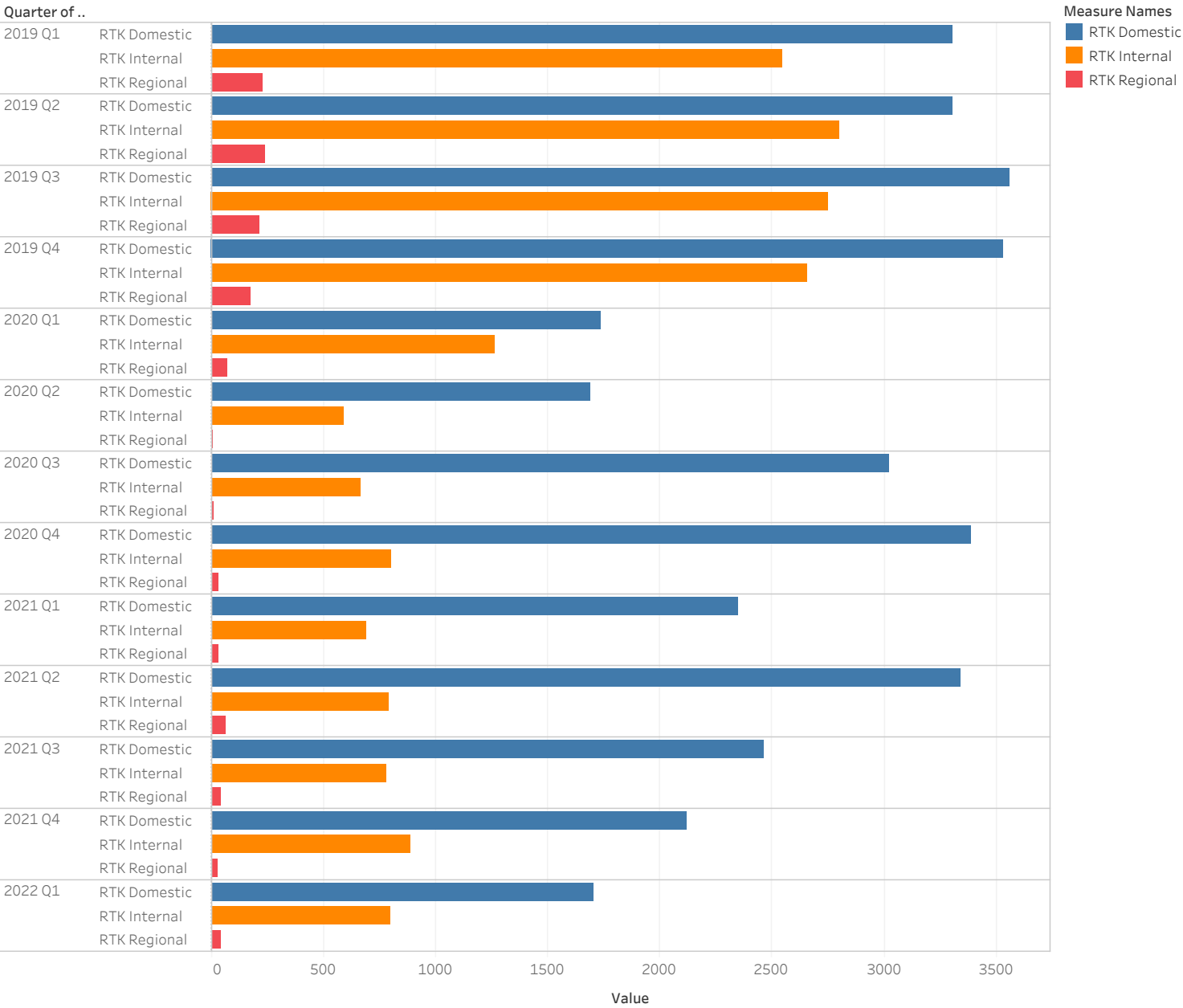
Then output the processed data to “.csv” file.

### CA\_RTK

Date	RTK_ttl	RTK_domestic	RTK_internal	RTK_regional
2019-01-01	2066.235	1104.6	884.6	77.1
2019-02-01	1895.1	1066.7	755.5	72.9
2019-03-01	2117.2	1132.9	907.4	77
2019-04-01	2123.7	1101	941.3	81.4
2019-05-01	2151.8	1120.1	950.8	80.9
2019-06-01	2071.3	1085.2	907.2	78.9
2019-07-01	2189	1182.2	925.6	81.3
2019-08-01	2238	1219.7	945.1	73.1
2019-09-01	2097.1	1161	880.3	55.8
2019-10-01	2176	1208.6	907.4	59.9
2019-11-01	2070.6	1152.9	861.4	56.3
2019-12-01	2117.8	1172	885.6	60.2
2020-01-01	1999.2	1098	841.7	59.5
2020-02-01	439.4	195.2	238.2	6
2020-03-01	630.6	442.7	183.9	4
2020-04-01	557.1	419.7	135.5	1.9
2020-05-01	833.5	605.1	226.4	2
2020-06-01	895	664.8	227.6	2.6
2020-07-01	1062.7	833.2	227.1	2.5
2020-08-01	1239.1	1027.1	208.9	3.1
2020-09-01	1398.1	1165.2	228.8	4.1
2020-10-01	1505.6	1232.4	263.4	9.8
2020-11-01	1441.9	1128.3	302.4	11.2
2020-12-01	1278.3	1028.3	236.7	13.3
2021-01-01	910.9	683.8	217.6	9.6
2021-02-01	741.1	533.1	201.6	6.3

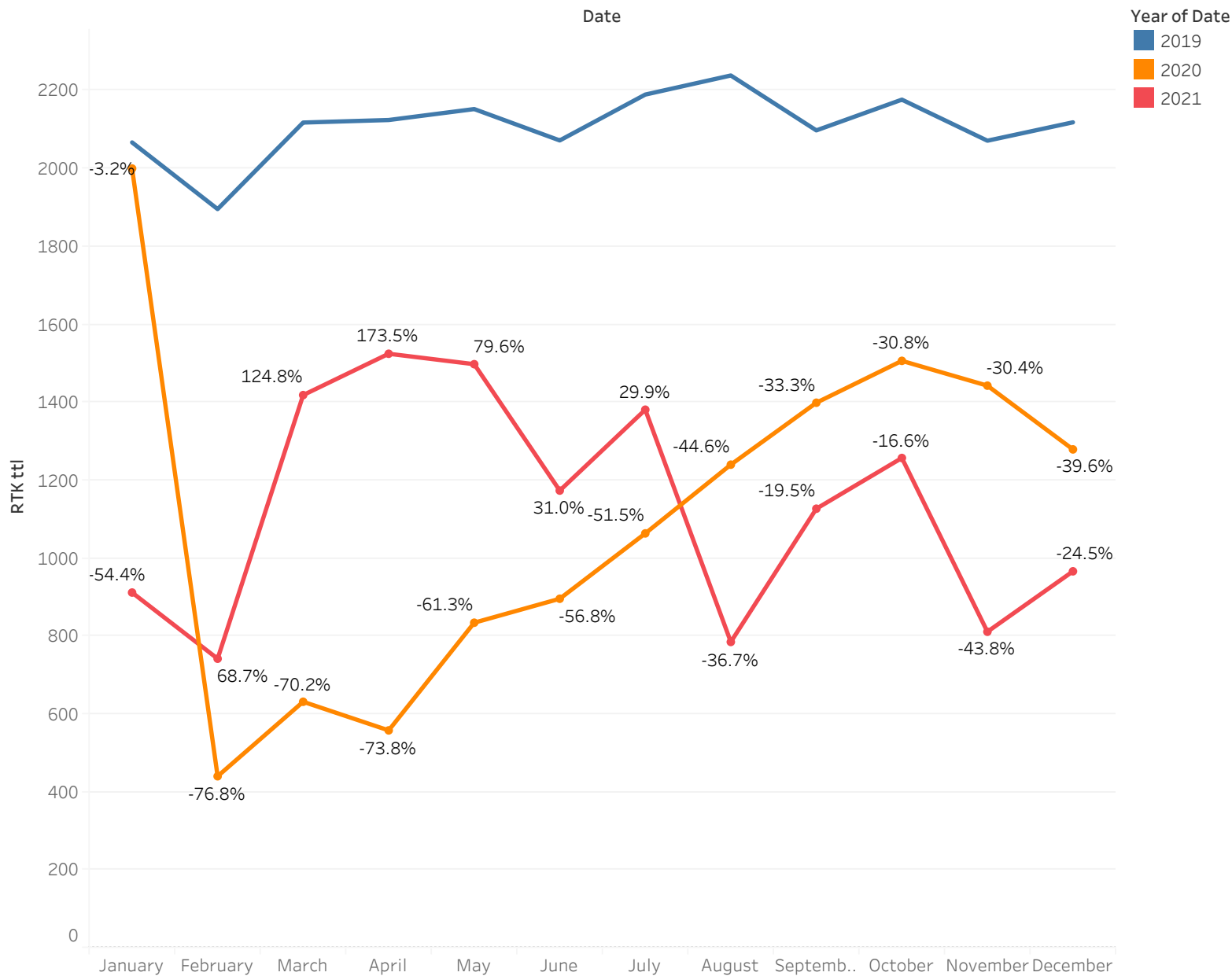
3. Visualization using Tableau: Quantile RTK Bar Chart and Year Over year growth line graph.

AirChina Quartile RTK 2019~2021



RTK Domestic, RTK Internal and RTK Regional for each Date Quarter. Color shows details about RTK Domestic, RTK Internal and RTK Regional. The view is filtered on Date Quarter, which excludes 2022 Q2.

AirChina Total RTK Year Over Year Growth 2019~2021



The trend of sum of RTK ttl for Date Month. Color shows details about Date Year. The marks are labeled by % Difference in RTK ttl. The view is filtered on Date Year, which keeps 2019, 2020 and 2021.