DATA420-19S2 (C)

Assignment 1 GHCN Data Analysis using Spark

Processing

Q1.

(a) How is the data structured? Draw a directory tree to represent this in a sensible way.

The GHCN data are divided into two parts. One is the directory of *daily* which describes climates summaries. In this directory, there are 255 csv files in gzip that show the record from year of 1763 to 2017. And the other parts are metadata, containing four files like *countries, inventory, states* and *station*. We can draw the schema of files. The directory tree shows as below,

```
└──/data/ghcnd/
   ├──countries
   ├──daily
     ├──1763.csv.gz
                 ├──1764.csv.gz
                 ├──1765.csv.gz
                      …
                 ├──2017.csv.gz
   ├──inventory
   ├──states
   ├──station
```

(b) How many years are contained in *daily*, and how does the size of the data change?

There are 255 years contained in *daily*. The file size increased from 3,367 to 194,390,036, which means data size in 2016 increased dramatically by 57,733 times compared with that in 1763.

(c) What is the total size of all of the data? How much of that is *daily*?

The actual total size of all of the data is 13.4GB, the backup size is 107.2GB. *Daily* take up almost entire size. All of the metadata combined only contributes 36.3 MB to the total. As the daily data was compressed, and the actual size of the uncompressed data will be significantly higher. The gzip compression ratio for csv files is usually around 10:1 so this could be between 100 GB and 1 TB.

Q2.

(a) Define schemas for each of *daily, stations, states, countries* and *inventory*.

Please see the attached code.

(b) Load 1000 rows of ///data/ghcnd/daily/2017.csv.gz. Was the description of the data accurate? Was there anything unexpected?

The top 10 rows of daily2017 showed as the result 1.

```
In [4]: daily2017.show(10, False)
+-----------+--------+-------+-----+----------------+------------+-----------+----------------+
|ID         |DATE    |ELEMENT|VALUE|MEASUREMENT FLAG|QUALITY FLAG|SOURCE FLAG|OBSERVATION TIME|
+-----------+--------+-------+-----+----------------+------------+-----------+----------------+
|CA1MB000296|20170101|PRCP   |0.0  |null            |null        |N          |null            |
|US1NCBC0113|20170101|PRCP   |5.0  |null            |null        |N          |null            |
|ASN00015643|20170101|TMAX   |274.0|null            |null        |a          |null            |
|ASN00015643|20170101|TMIN   |218.0|null            |null        |a          |null            |
|ASN00015643|20170101|PRCP   |2.0  |null            |null        |a          |null            |
|US1MTMH0019|20170101|PRCP   |43.0 |null            |null        |N          |null            |
|US1MTMH0019|20170101|SNOW   |28.0 |null            |null        |N          |null            |
|US1MTMH0019|20170101|SNWD   |178.0|null            |null        |N          |null            |
|ASN00085296|20170101|TMAX   |217.0|null            |null        |a          |null            |
|ASN00085296|20170101|TMIN   |127.0|null            |null        |a          |null            |
+-----------+--------+-------+-----+----------------+------------+-----------+----------------+
only showing top 10 rows
```

Result 1

Not all description is accurate data type. For example, when I defined the value *Date* as *Datetype* in schema definition for daily2017 schema, there were all null values after loading. However, I can load data correctly when I changed *Datetype* as *Stringtype*. Also, *Timestamptype* was not fit for the value of *OBSERVATION TIME* because it would give us a current time which was confused.

(c) How many rows are in each metadata table? How many stations do not have a WMO ID?

The *countries* data has 218 rows in total, representing 218 distinct countries or territories from around the world.

The *inventories* data has 595,699 rows in total, representing 595,699 sets of elements were recorded by each station, during the time period.

The *states* data has 74 rows in data, representing 74 states from around the world.

The *stations* data has 103,656 rows in total, representing 103,656 records associated with states and countries.

Metadata Table	Rows
Countries	218
Inventories	595,699
States	74
Stations	103,656

According to the readme text, the null fields of all metadata table represented by whitespace and *WMO ID* was string type value, which means we cannot regard it as null values. I used function *trim* instead of function *isNull*. Finally, I found that there were 95,595 stations do not have a *WMO ID*.

Q3.

(a) Extract the two characters country code from each station code in *stations* and store the output as a new column using the *withColumn* command.

The new stations with new column called *COUNTRYCODE* showed as the result 2. It is important that I found that all country name in *countries* were matched with the country name in station.

```
In [31]: stations_new.show(5,False)
+-----------+--------+---------+---------+-----+---------------------+-------+-----------+-----+-----------+
|ID         |LATITUDE|LONGITUDE|ELEVATION|STATE|STATIONNAME          |GSNFLAG|HCN/CRNFLAG|WMOID|COUNTRYCODE|
+-----------+--------+---------+---------+-----+---------------------+-------+-----------+-----+-----------+
|ACW00011604|17.1167 |-61.7833 |10.1     |     |ST JOHNS COOLIDGE FLD|       |           |     |AC         |
|ACW00011647|17.1333 |-61.7833 |19.2     |     |ST JOHNS             |       |           |     |AC         |
|AE000041196|25.333  |55.517   |34.0     |     |SHARJAH INTER. AIRP  |GSN    |           |41196|AE         |
|AEM00041194|25.255  |55.364   |10.4     |     |DUBAI INTL           |       |           |41194|AE         |
|AEM00041217|24.433  |54.651   |26.8     |     |ABU DHABI INTL       |       |           |41217|AE         |
+-----------+--------+---------+---------+-----+---------------------+-------+-----------+-----+-----------+
only showing top 5 rows
```

Result 2

(b) LEFT JOIN *stations* with *countries* using the output from part a

The new stations that left join *countries* showed as the result 3.

```
In [32]: join_station_country.show(5, False)
+-----------+-----------+--------+---------+---------+-----+---------------------+-------+-----------+-----+--------------------+
|COUNTRYCODE|ID         |LATITUDE|LONGITUDE|ELEVATION|STATE|STATIONNAME          |GSNFLAG|HCN/CRNFLAG|WMOID|COUNTRYNAME         |
+-----------+-----------+--------+---------+---------+-----+---------------------+-------+-----------+-----+--------------------+
|AC         |ACW00011604|17.1167 |-61.7833 |10.1     |     |ST JOHNS COOLIDGE FLD|       |           |     |Antigua and Barbuda |
|AC         |ACW00011647|17.1333 |-61.7833 |19.2     |     |ST JOHNS             |       |           |     |Antigua and Barbuda |
|AE         |AE000041196|25.333  |55.517   |34.0     |     |SHARJAH INTER. AIRP  |GSN    |           |41196|United Arab Emirates|
|AE         |AEM00041194|25.255  |55.364   |10.4     |     |DUBAI INTL           |       |           |41194|United Arab Emirates|
|AE         |AEM00041217|24.433  |54.651   |26.8     |     |ABU DHABI INTL       |       |           |41217|United Arab Emirates|
+-----------+-----------+--------+---------+---------+-----+---------------------+-------+-----------+-----+--------------------+
only showing top 5 rows
```

Result 3.

(c) LEFT JOIN *stations* and *states*, allowing for the fact that states codes are only provided for *stations* in the US.

The new stations that left join *states* showed as the result 4. There were only 65,920 stations have *state name* matched. However, the total records of *stations* were 103,656, which was due to the fact that stated codes are only provided for *stations* in the US.

```
   ...:
   ...: station_country_states.show(5, False)
+-----+-----------+-----------+--------+---------+---------+---------------------+-------+-----------+-----+--------------------+---------+
|STATE|COUNTRYCODE|ID         |LATITUDE|LONGITUDE|ELEVATION|STATIONNAME          |GSNFLAG|HCN/CRNFLAG|WMOID|COUNTRYNAME         |STATENAME|
+-----+-----------+-----------+--------+---------+---------+---------------------+-------+-----------+-----+--------------------+---------+
|     |AC         |ACW00011604|17.1167 |-61.7833 |10.1     |ST JOHNS COOLIDGE FLD|       |           |     |Antigua and Barbuda |null     |
|     |AC         |ACW00011647|17.1333 |-61.7833 |19.2     |ST JOHNS             |       |           |     |Antigua and Barbuda |null     |
|     |AE         |AE000041196|25.333  |55.517   |34.0     |SHARJAH INTER. AIRP  |GSN    |           |41196|United Arab Emirates|null     |
|     |AE         |AEM00041194|25.255  |55.364   |10.4     |DUBAI INTL           |       |           |41194|United Arab Emirates|null     |
|     |AE         |AEM00041217|24.433  |54.651   |26.8     |ABU DHABI INTL       |       |           |41217|United Arab Emirates|null     |
+-----+-----------+-----------+--------+---------+---------+---------------------+-------+-----------+-----+--------------------+---------+
only showing top 5 rows
```

Result 4.

(d) Based on *inventory*, what was the first and last year that each station was active and collected any element at all?

The *inventory* with details of first year and last year showed as the result 5. First, I grouped *ID* to get a range of first year and last year of each station. Then *aggregating the* minimum of *FRISTYEAR* as the first year and the maximum *LASTYEAR* as the last year of each station.

```
   ...: year_element.show(10,False)
+-----------+---------+----------+
|ID         |FIRSTYEAR|LASTERYEAR|
+-----------+---------+----------+
|ACW00011647|1957     |1970      |
|AEM00041217|1983     |2017      |
|AG000060590|1892     |2017      |
|AGE00147706|1893     |1920      |
|AGE00147708|1879     |2017      |
|AGE00147709|1879     |1938      |
|AGE00147710|1909     |2009      |
|AGE00147711|1880     |1938      |
|AGE00147714|1896     |1938      |
|AGE00147719|1888     |2017      |
+-----------+---------+----------+
only showing top 10 rows
```

Result 5.

How many different elements has each stations collected overall?

The stations with different elements counts showed as the result 6. I just need *group* the *ID* and get unique count of *ELEMENT* that each station had.

```
   ...: distinct_element.show(10, False)
+-----------+-------------+
|ID         |DISTINCTCOUNT|
+-----------+-------------+
|ACW00011647|7            |
|AG000060590|4            |
|AGE00147708|5            |
|AGE00147709|3            |
|AGE00147710|4            |
|AGE00147714|3            |
|AGE00147719|4            |
|AGM00060351|4            |
|AGM00060360|4            |
|AGM00060445|5            |
+-----------+-------------+
only showing top 10 rows
```

Result 6.

Count separately the number of core elements and the number of "other" elements that each station has collected overall.

The stations with the count of core, other and distinct elements showed as result 7. According to the readme text, the core element includes that *PRCP, SNOW, SNWD, TMAX* and *TMIN*. First, I counted the stations when they have core elements. Then *OTHERCOUNT* can be achieved by subtracting the *DISTRINCTCOUNT*,

```
   ...:
   ...: number_core_other_element.show(10,False)
+-----------+---------+-------------+----------+
|ID         |CORECOUNT|DISTINCTCOUNT|OTHERCOUNT|
+-----------+---------+-------------+----------+
|ACW00011647|5        |7            |2         |
|AEM00041217|3        |4            |1         |
|AG000060590|3        |4            |1         |
|AGE00147706|3        |3            |0         |
|AGE00147708|4        |5            |1         |
|AGE00147709|3        |3            |0         |
|AGE00147710|3        |4            |1         |
|AGE00147711|3        |3            |0         |
|AGE00147714|3        |3            |0         |
|AGE00147719|3        |4            |1         |
+-----------+---------+-------------+----------+
only showing top 10 rows
```

Result 7.

How many stations collect all five core elements?

There are total 20,224 stations that collected all five core elements. And the stations that have all five core elements showed as the result 8. To get this result, I *filtered* the stations that the *CORECOUNT* number equals to five and count them.

```
   ...: five_core_element.show(10, False)
   ...: five_core_element.count()
+-----------+---------+
|ID         |CORECOUNT|
+-----------+---------+
|ACW00011647|5        |
|AQC00914021|5        |
|AQC00914873|5        |
|AYM00089664|5        |
|AYW00067402|5        |
|AYW00067601|5        |
|AYW00068701|5        |
|BFW00012716|5        |
|CA001011500|5        |
|CA001011922|5        |
+-----------+---------+
only showing top 10 rows

Out[81]: 20224
```

Result 8.

How many only collection precipitations?

The number of stations only collect precipitation are 15,970 stations, which showed as result 9. I just need *filter* the *ELEMENT* containing *PRCP* and count the result.

```
   ...: precipitation_element.show()
   ...: precipitation_element.count()
+-----------+-------+---------+----------+
|         ID|ELEMENT|CORECOUNT|OTHERCOUNT|
+-----------+-------+---------+----------+
|AJ000037679|   PRCP|        1|         0|
|AJ000037831|   PRCP|        1|         0|
|AJ000037912|   PRCP|        1|         0|
|AJ000037981|   PRCP|        1|         0|
|AM000037683|   PRCP|        1|         0|
|AM000037698|   PRCP|        1|         0|
|AM000037786|   PRCP|        1|         0|
|AM000037802|   PRCP|        1|         0|
|AM000037873|   PRCP|        1|         0|
|AM000037954|   PRCP|        1|         0|
|AR000000002|   PRCP|        1|         0|
|AR000000004|   PRCP|        1|         0|
|AR000000012|   PRCP|        1|         0|
|AR000000013|   PRCP|        1|         0|
|ASN00001003|   PRCP|        1|         0|
|ASN00002004|   PRCP|        1|         0|
|ASN00002027|   PRCP|        1|         0|
|ASN00002033|   PRCP|        1|         0|
|ASN00002034|   PRCP|        1|         0|
|ASN00002039|   PRCP|        1|         0|
+-----------+-------+---------+----------+
only showing top 20 rows

Out[82]: 15970
```

Result 9.

(e) Left join stations and output from d

The new stations that left join the previous output showed as result 10. I chose Parquet format to save it. Because this format is designed to bring efficient columnar storage of data compared the row-based files like CSV. And queries of Parquet files does not need reading all raw data which means we can save money and get better performance.

```
In [43]: stations2.show(10,False)
```

ID	STATE	COUNTRYCODE	LATITUDE	LONGITUDE	ELEVATION	STATIONNAME	GSNFLAG	HCN/CRNFLAG	WMOID	COUNTRYNAME	STATENAME	CORECOUNT	DISTINCTCOUNT	OTHERCOUNT	FIRSTYEAR	LASTERYEAR
ACW00011647		AC	17.1333	-61.7833	19.2	ST JOHNS				Antigua and Barbuda	null	5	7	2	1957	1970
AEM00041217		AE	24.433	54.651	26.8	ABU DHABI INTL			41217	United Arab Emirates	null	3	4	1	1983	2017
AG000060590		AG	30.5667	2.8667	397.0	EL-GOLEA	GSN		60590	Algeria	null	3	4	1	1892	2017
AGE00147706		AG	36.8	3.03	344.0	ALGIERS-BOUZAREAH				Algeria	null	3	3	0	1893	1920
AGE00147708		AG	36.72	4.05	222.0	TIZI OUZOU			60395	Algeria	null	4	5	1	1879	2017
AGE00147709		AG	36.63	4.2	942.0	FORT NATIONAL				Algeria	null	3	3	0	1879	1938
AGE00147710		AG	36.75	5.1	9.0	BEJAIA-BOUGIE (PORT)			60401	Algeria	null	3	4	1	1909	2009
AGE00147711		AG	36.3697	6.62	660.0	CONSTANTINE				Algeria	null	3	3	0	1880	1938
AGE00147714		AG	35.77	0.8	78.0	ORAN-CAP FALCON				Algeria	null	3	3	0	1896	1938
AGE00147719		AG	33.7997	2.89	767.0	LAGHOUAT			60545	Algeria	null	3	4	1	1888	2017

only showing top 10 rows

Result 10.

(f) How expensive do you think it would be to LEFT JOIN all of daily and stations? Could you determine if there are any stations in daily that are not in stations without using LEFT JOIN.

The new daily that limited with 1000 rows and left join the stations showed as result 11.

```
In [53]: station_daily.show(10,False)
```

ID	DATE	ELEMENT	VALUE	MEASUREMENT FLAG	QUALITY FLAG	SOURCE FLAG	OBSERVATION TIME	STATE	COUNTRYCODE	LATITUDE	LONGITUDE	ELEVATION	STATIONNAME	GSNFLAG	HCN/CRNFLAG	WMOID	COUNTRYNAME	STATENAME	CORECOUNT	DISTINCTCOUNT	OTHERCOUNT	FIRSTYEAR	LASTERYEAR
ASN00002069	20170101	PRCP	0.0	null	null	a	null		AS	-16.4433	127.7836	270.0	KACHANA				Australia	null	1	4	[illegible]	1993	2017
ASN00014167	20170101	PRCP	166.0	null	null	a	null		AS	-12.4648	130.8495	41.0	STOKES HILL				Australia	null	1	4	[illegible]	1968	2017
ASN00015005	20170101	PRCP	24.0	null	null	a	null		AS	-20.0298	137.4906	205.0	AVON DOWNS				Australia	null	3	6	[illegible]	1909	2017
ASN00024580	20170101	TMAX	229.0	null	null	a	null		AS	-35.2836	138.8934	58.0	STRATHALBYN RACECOURSE			94814	Australia	null	3	4	[illegible]	1996	2017
ASN00024580	20170101	TMIN	137.0	null	null	a	null		AS	-35.2836	138.8934	58.0	STRATHALBYN RACECOURSE			94814	Australia	null	3	4	[illegible]	1996	2017
ASN00024580	20170101	PRCP	0.0	null	null	a	null		AS	-35.2836	138.8934	58.0	STRATHALBYN RACECOURSE			94814	Australia	null	3	4	[illegible]	1996	2017
ASN00031051	20170101	PRCP	44.0	null	null	a	null		AS	-17.3442	144.9306	470.0	PETFORD				Australia	null	1	4	[illegible]	1920	2017
ASN00031181	20170101	PRCP	0.0	null	null	a	null		AS	-17.545	145.7156	580.0	BARTLE VIEW ALERT				Australia	null	1	1	[illegible]	2000	2017
ASN00032172	20170101	PRCP	0.0	null	null	a	null		AS	-18.4606	145.7383	685.8	WALLAMAN ALERT				Australia	null	1	1	[illegible]	2000	2017
ASN00056194	20170101	PRCP	44.0	null	null	a	null		AS	-29.2716	151.8612	930.0	TENTERFIELD (KOOKYNIE)				Australia	null	1	4	[illegible]	1995	2017

only showing top 10 rows

Result 11.

There are no stations in the subset of daily 2017 that are not in stations at all.

It would be at high cost to left join all of *daily* and *stations*. There are 24 variables in the new daily that left join *daily* and *stations*. the cost of join operation of 1000 rows of *2017daily* would be O(1000 * 24) which costs at least 3 seconds, however, there are 21,904,999 rows in *2017daily* and total 255 years of all *daily* data, leading the cost of join whole daily with station would be O(m*24), even the complexity can be reduced to O(m+n), it is still an expensive operation.

Another way is to do subtraction. The code as follows,

count_null = station_daily.select([“ID”]).subtract(stations2.select([“ID”]))

Analysis

Q1

(a) How many *stations* are there in total? How many *stations* have been active in 2017?

First I got the distinct count of station *ID* and then filtered them by the *FIRSTYEAR* that less than 2017 and the *LASTYEAR* that large than 2017. Next, I counted the number of stations by filtering them with different network. The result as followed.
There are total 103,656 stations. 37,546 stations have been active in 2017.
There are 991 stations in GCOS Surface Network(GSN).
There are 1,218 stations in the US Historical Climatology Network(HCN).
There are 230 stations in the US Climate Reference Network(CRN).
There are 14 stations that are in both GSN and HCN.

(b) Count the total number of stations in each country, and store the output in countires using withColumnRenamed command. Do the same for states.

The total number of *stations* in each country showed as the result 12. I counted the station *ID* after grouping them with *COUNTRYCODE.*

```
   ...: countries2.show(5,False)
+-----------+------------+
|COUNTRYCODE|STATIONCOUNT|
+-----------+------------+
|AQ         |21          |
|AU         |13          |
|AY         |102         |
|BF         |6           |
|BK         |6           |
+-----------+------------+
only showing top 5 rows
```

Result 12.

The same operation on states. The result showed as the result 13.

```
   ...: states2.show(5,False)
+-----+------------+
|STATE|STATIONCOUNT|
+-----+------------+
|NT   |137         |
|SK   |758         |
|MB   |697         |
|ON   |1730        |
|NB   |264         |
+-----+------------+
only showing top 5 rows
```

Result 13.

(c) How many stations are there in the Southern Hemisphere only?

There are 25,337 stations in the Southern Hemisphere, which showed as the result 14. I counted the station *ID* after filtering the *LATITUDE* below the zero.

```
   ...: station_southern_hemisphere.show(5,False)
+-----------+--------+
|ID         |LATITUDE|
+-----------+--------+
|AO000066390|-14.933 |
|AO000066422|-15.2   |
|AQC00914021|-14.2667|
|AQC00914138|-14.2833|
|AQC00914424|-14.2333|
+-----------+--------+
only showing top 5 rows
```

Result 14.

How many stations are there in total in the territories of the United States around the world?

There were 57,227 stations in the territories of the United States around the world. I selected the *COUNTRYNAME* containing United Stated and counted the station *ID*, which showed as the result 15.

```
   ...: station_USA.show(20,False)
   ...: station_USA.count()
+-----------+---------------------------------------------+
|ID         |COUNTRYNAME                                  |
+-----------+---------------------------------------------+
|AQC00914021|American Samoa [United States]               |
|AQC00914138|American Samoa [United States]               |
|AQC00914424|American Samoa [United States]               |
|AQC00914873|American Samoa [United States]               |
|CQC00914800|Northern Mariana Islands [United States]|
|CQW00041408|Northern Mariana Islands [United States]|
|CQW00041412|Northern Mariana Islands [United States]|
|GQC00914468|Guam [United States]                         |
|GQW00041407|Guam [United States]                         |
|GQW00041414|Guam [United States]                         |
|JQW00021602|Johnston Atoll [United States]               |
|LQW00020603|Palmyra Atoll [United States]                |
|LQW00020604|Palmyra Atoll [United States]                |
|RQ1PRAC0002|Puerto Rico [United States]                  |
|RQ1PRCG0002|Puerto Rico [United States]                  |
|RQ1PRCL0001|Puerto Rico [United States]                  |
|RQ1PRCY0001|Puerto Rico [United States]                  |
|RQ1PRGB0003|Puerto Rico [United States]                  |
|RQ1PRHM0001|Puerto Rico [United States]                  |
|RQ1PRIS0001|Puerto Rico [United States]                  |
+-----------+---------------------------------------------+
only showing top 20 rows

Out[94]: 57227
```

Result 15.

Q2

(a) Computes the geographical distance between two stations using their latitude and longitude as arguments.

The geographical distance between two stations showed as the result 16. First I made the function called *compute_distance* to compute the geographical distance of two points. Then wrapping this function by pyspark function *udf that allows* me to lambda using functions on multiple column values. The important process is to *CROSS JOIN* stations and finally filtered different station ID.

```
   ...: test_distance.show(5,False)
+-----------+---------+----------+-----------+---------+----------+--------+
|ID1        |LATITUDE1|LONGITUDE1|ID2        |LATITUDE2|LONGITUDE2|distance|
+-----------+---------+----------+-----------+---------+----------+--------+
|AQC00914021|-14.2667 |-170.5833 |AQC00914138|-14.2833 |-170.6833 |11.27   |
|AQC00914021|-14.2667 |-170.5833 |AQC00914424|-14.2333 |-169.5167 |118.66  |
|AQC00914021|-14.2667 |-170.5833 |AQC00914873|-14.35   |-170.7667 |22.35   |
|AQC00914021|-14.2667 |-170.5833 |CQC00914800|14.1333  |145.15    |5757.83 |
|AQC00914021|-14.2667 |-170.5833 |CQW00041408|15.1167  |145.7     |5763.81 |
+-----------+---------+----------+-----------+---------+----------+--------+
only showing top 5 rows
```

Result 16.

(b) Compute the pairwise distances between all stations in New Zealand. And what two stations are the geographically closest in New Zealand?

The pairwise distances between all stations in New Zealand showed as the result 17. I selected all observations of New Zealand by filtering the *COUNTRYNAME* including New Zealand, and then did the same operation as previous.

```
In [68]: distance_stations_newzealand.show(5,False)
+-----------+---------+----------+-----------+---------+----------+--------+
|ID1        |LATITUDE1|LONGITUDE1|ID2        |LATITUDE2|LONGITUDE2|distance|
+-----------+---------+----------+-----------+---------+----------+--------+
|NZ000093012|-35.1    |173.267   |NZM00093110|-37.0    |174.8     |270.57  |
|NZM00093110|-37.0    |174.8     |NZ000093012|-35.1    |173.267   |270.57  |
|NZ000093012|-35.1    |173.267   |NZ000936150|-42.717  |170.983   |876.51  |
|NZ000093012|-35.1    |173.267   |NZ000939450|-52.55   |169.167   |1970.45 |
|NZ000093012|-35.1    |173.267   |NZM00093678|-42.417  |173.7     |809.78  |
+-----------+---------+----------+-----------+---------+----------+--------+
only showing top 5 rows
```

Result 17.

The closes station is *NZM00093439* and *NZ000093417*, which is *52.09km*. The result as below,

```
   ...:
   ...: distance_stations_newzealand.show(5,False)
+-----------+---------+----------+-----------+---------+----------+--------+
|ID1        |LATITUDE1|LONGITUDE1|ID2        |LATITUDE2|LONGITUDE2|distance|
+-----------+---------+----------+-----------+---------+----------+--------+
|NZM00093439|-41.333  |174.8     |NZ000093417|-40.9    |174.983   |52.09   |
|NZM00093678|-42.417  |173.7     |NZM00093439|-41.333  |174.8     |171.3   |
|NZM00093678|-42.417  |173.7     |NZM00093781|-43.489  |172.532   |175.7   |
|NZ000936150|-42.717  |170.983   |NZM00093781|-43.489  |172.532   |192.05  |
|NZM00093678|-42.417  |173.7     |NZ000093417|-40.9    |174.983   |220.29  |
+-----------+---------+----------+-----------+---------+----------+--------+
only showing top 5 rows
```

Result 18.

Q3

(a) How many blocks are required for the daily climate summaries for the year 2017? What about the year 2010? What are the individual block sizes for the year 2010?

The default block size of HDFS is 134,217,728 Byte (about 134MB). Two number of blocks for year 2010 are required and the average block size is 103,590,865 Byte (about 103MB). The result showed as the result 19. It is possible for spark to load and apply transformation in parallel for year 2010 as it was allocated two blocks.

```
Connecting to namenode via http://node0:9870/fsck?ugi=jpe116&path=%2Fdata%2Fghcnd%2Fdaily%2F2010.csv.gz
FSCK started by jpe116 (auth:SIMPLE) from /192.168.40.10 for path /data/ghcnd/daily/2010.csv.gz at Thu Aug 29 09:55:12 NZST 2019

Status: HEALTHY
 Number of data-nodes:  32
 Number of racks:               1
 Total dirs:                    0
 Total symlinks:                0

Replicated Blocks:
 Total size:    207181730 B
 Total files:   1
 Total blocks (validated):      2 (avg. block size 103590865 B)
 Minimally replicated blocks:   2 (100.0 %)
 Over-replicated blocks:        0 (0.0 %)
 Under-replicated blocks:       0 (0.0 %)
 Mis-replicated blocks:         0 (0.0 %)
 Default replication factor:    4
 Average block replication:     8.0
 Missing blocks:                0
 Corrupt blocks:                0
 Missing replicas:              0 (0.0 %)
```

Result 19.

However, only one block is required for 2017, as the file size is smaller than the default block size. We can see the output from the result 20. So, that means that it is impossible to do parallel operation.

```
WARNING: Use of this script to execute fsck is deprecated.
WARNING: Attempting to execute replacement "hdfs fsck" instead.

Connecting to namenode via http://node0:9870/fsck?ugi=jpe116&path=%2Fdata%2Fghcnd%2Fdaily%2F2017.csv.gz
FSCK started by jpe116 (auth:SIMPLE) from /192.168.40.10 for path /data/ghcnd/daily/2017.csv.gz at Thu Aug 29 09:58:15 NZST 2019

Status: HEALTHY
 Number of data-nodes:  32
 Number of racks:               1
 Total dirs:                    0
 Total symlinks:                0

Replicated Blocks:
 Total size:    125257391 B
 Total files:   1
 Total blocks (validated):      1 (avg. block size 125257391 B)
 Minimally replicated blocks:   1 (100.0 %)
 Over-replicated blocks:        0 (0.0 %)
 Under-replicated blocks:       0 (0.0 %)
 Mis-replicated blocks:         0 (0.0 %)
 Default replication factor:    4
 Average block replication:     8.0
 Missing blocks:                0
 Corrupt blocks:                0
 Missing replicas:              0 (0.0 %)
```

Result 20

(b) Load the count the number of rows in daily for each of the years 2010 and 2017. How many tasks were executed by each stage of each job? Did the number of tasks executed correspond to the number of blocks in each input?

The number of rows in *daily* for 2017 and 2010 are 21,904,999 and 36,946,080 respectively.

One task executed in each stage of each job. The details as below. After counting the number of rows in daily for each of the years 2010 and 2017, there were two jobs showed as the figure 1. One for operations on 2010 and the other for 2017.

▾ Completed Jobs (2)

Job Id ▾	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
1	count at NativeMethodAccessorImpl.java:0 count at NativeMethodAccessorImpl.java:0	2019/08/29 10:11:52	10 s	2/2	2/2
0	count at NativeMethodAccessorImpl.java:0 count at NativeMethodAccessorImpl.java:0	2019/08/29 10:10:36	8 s	2/2	2/2

Figure 1

When did operations on year 2017, there were two stage for job0.

▾ Completed Stages (2)

Stage Id ▾	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
1	count at NativeMethodAccessorImpl.java:0 +details	2019/08/29 10:10:44	0.1 s	1/1			59.0 B	
0	count at NativeMethodAccessorImpl.java:0 +details	2019/08/29 10:10:36	8 s	1/1	119.5 MB			59.0 B

Figure 2

There was one task for each stage. One task for stage0 and one task for stage 1.

Tasks (1)

Index	ID	Attempt	Status	Locality Level	Executor ID	Host	Launch Time	Duration	GC Time	Input Size / Records	Write Time	Shuffle Write Size / Records	Errors
0	0	0	SUCCESS	ANY	0	192.168.40.120 stdout stderr	2019/08/29 10:10:36	8 s	0.2 s	119.5 MB / 21904999	4 ms	59.0 B / 1	

Figure 3

Tasks (1)

Index	ID	Attempt	Status	Locality Level	Executor ID	Host	Launch Time	Duration	GC Time	Shuffle Read Size / Records	Errors
0	1	0	SUCCESS	NODE_LOCAL	0	192.168.40.120 stdout stderr	2019/08/29 10:10:44	0.1 s		59.0 B / 1	

Figure 4

When did operations for year 2010 there were also two stage for job 1

Completed Stages (2)

Stage Id	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
3	count at NativeMethodAccessorImpl.java:0 +details	2019/08/29 10:12:02	23 ms	1/1			59.0 B	
2	count at NativeMethodAccessorImpl.java:0 +details	2019/08/29 10:11:52	10 s	1/1	197.6 MB			59.0 B

Figure 5

One task for stage 2, and one task for stage 3.

Tasks (1)

Index	ID	Attempt	Status	Locality Level	Executor ID	Host	Launch Time	Duration	GC Time	Input Size / Records	Write Time	Shuffle Write Size / Records	Errors
0	2	0	SUCCESS	ANY	0	192.168.40.120 stdout stderr	2019/08/29 10:11:52	10 s	55 ms	197.6 MB / 36946080	1 ms	59.0 B / 1	

Figure 6

Tasks (1)

Index	ID	Attempt	Status	Locality Level	Executor ID	Host	Launch Time	Duration	GC Time	Shuffle Read Size / Records	Errors
0	3	0	SUCCESS	NODE_LOCAL	0	192.168.40.120 stdout stderr	2019/08/29 10:12:02	22 ms		59.0 B / 1	

Figure 7

In conclusion, I found that the number of tasks executed does not matter the number of blocks in each inputs.

(c) Load and count the number of rows in daily from 2010 to 2015. How many tasks were executed by each stage, and how does this number correspond to your input. How spark partitions input files that are compressed.

The rows in daily from 2010 to 2015 is 207,716,098.

For stage 4 there are 6 tasks

▾ Tasks (6)

Index ▴	ID	Attempt	Status	Locality Level	Executor ID	Host	Launch Time	Duration	GC Time	Input Size / Records	Write Time	Shuffle Write Size / Records	Errors
0	4	0	SUCCESS	ANY	0	192.168.40.120 stdout stderr	2019/08/29 10:49:55	11 s	86 ms	197.6 MB / 36946080		59.0 B / 1	
1	5	0	SUCCESS	ANY	1	192.168.40.158 stdout stderr	2019/08/29 10:49:55	18 s	0.5 s	187.4 MB / 34450636	6 ms	59.0 B / 1	
2	6	0	SUCCESS	ANY	0	192.168.40.120 stdout stderr	2019/08/29 10:50:07	10 s	91 ms	187.3 MB / 33928984		59.0 B / 1	
3	7	0	SUCCESS	ANY	1	192.168.40.158 stdout stderr	2019/08/29 10:50:13	13 s	85 ms	186.9 MB / 34352055		59.0 B / 1	
4	8	0	SUCCESS	ANY	0	192.168.40.120 stdout stderr	2019/08/29 10:50:17	10 s	79 ms	186.8 MB / 34279588		59.0 B / 1	
5	9	0	SUCCESS	ANY	1	192.168.40.158 stdout stderr	2019/08/29 10:50:26	14 s	67 ms	184.2 MB / 33758755	1 ms	59.0 B / 1	

Figure 8

For stage 5 there are 1 task

▾ Tasks (1)

Index ▴	ID	Attempt	Status	Locality Level	Executor ID	Host	Launch Time	Duration	GC Time	Shuffle Read Size / Records	Errors
0	10	0	SUCCESS	NODE_LOCAL	1	192.168.40.158 stdout stderr	2019/08/29 10:50:40	0.2 s		354.0 B / 6	

Figure 9

When files were stored as gzip that are not splitting, there are no way to avoid reading the file in its entirely on one core. In order to parallelize work, the nth chunk depends on the n-1-th chunk's position. But it is impossible to read gzip stream. So, Spark decompresses the file first in its entirety before it can shuffle it to increase parallelism.

(d) Based on parts(b) and (c) what level of parallelism can you achieve when loading and applying transformations to daily? Can you think of any way you could increase this level of parallelism ether in Spark or by additional pre-processing?

There are 255 files in daily. As the number of tasks may equal the input of files, so Spark can reach at most 255 tasks at every stage.

Q4

(a) Count the number of rows in *daily.*

There are 2,624,027,105 rows in *daily.*

(b) Filter daily using the filter command to obtain the subset of observations containing the five core elements described in inventory. How many observations are there for each of five core numbers. Which elements has the most observations? Is this element consistent with Processing Q3?

The number of observations for each of the five core elements

The number of observations for each of five core numbers showed as the result 21. There are 918,490,401 observations recording *Precipitation* values. I filtered *stations* that have core elements and grouped by *ELEMENT*. The next step was aggregating the count of observations.

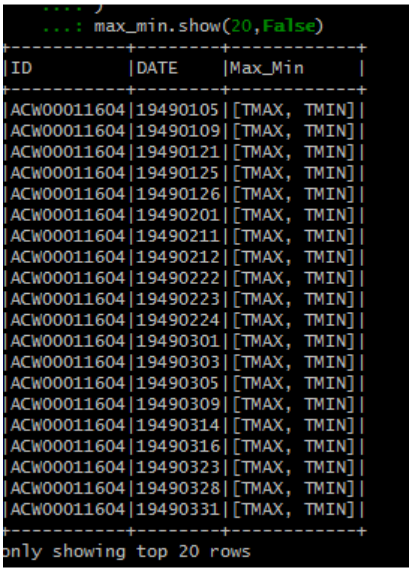
The result is consistent with processing Q3. In processing Q3, the number of stations only collect precipitation are 15,970 stations, accounting for 78% of total core elements.

```
    ...: max_element.show()
+-------+--------------+
|ELEMENT|count(ELEMENT)|
+-------+--------------+
|   PRCP|     918490401|
|   TMAX|     362528096|
|   TMIN|     360656728|
|   SNOW|     322003304|
|   SNWD|     261455306|
+-------+--------------+
```

Result 21.

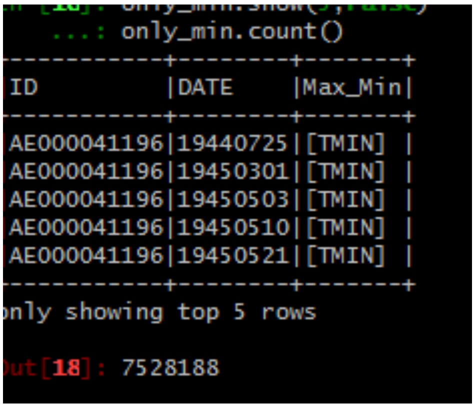
(c) How many observations of *TMIN* do not have a corresponding observation of *TMAX.*

There are 7,528,188 observations of *TMIN* do not have a corresponding observation of *TMAX* showed as the result 23. I filtered the stations that have maximum temperature and minimum temperature and *withcolumn* a new column to store this output as a set. This showed as the result 22. Finally, I filtered the set only contains the minimum temperature to get the result.



```
   ...: max_min.show(20,False)
+-----------+--------+------------+
|ID         |DATE    |Max_Min     |
+-----------+--------+------------+
|ACW00011604|19490105|[TMAX, TMIN]|
|ACW00011604|19490109|[TMAX, TMIN]|
|ACW00011604|19490121|[TMAX, TMIN]|
|ACW00011604|19490125|[TMAX, TMIN]|
|ACW00011604|19490126|[TMAX, TMIN]|
|ACW00011604|19490201|[TMAX, TMIN]|
|ACW00011604|19490211|[TMAX, TMIN]|
|ACW00011604|19490212|[TMAX, TMIN]|
|ACW00011604|19490222|[TMAX, TMIN]|
|ACW00011604|19490223|[TMAX, TMIN]|
|ACW00011604|19490224|[TMAX, TMIN]|
|ACW00011604|19490301|[TMAX, TMIN]|
|ACW00011604|19490303|[TMAX, TMIN]|
|ACW00011604|19490305|[TMAX, TMIN]|
|ACW00011604|19490309|[TMAX, TMIN]|
|ACW00011604|19490314|[TMAX, TMIN]|
|ACW00011604|19490316|[TMAX, TMIN]|
|ACW00011604|19490323|[TMAX, TMIN]|
|ACW00011604|19490328|[TMAX, TMIN]|
|ACW00011604|19490331|[TMAX, TMIN]|
+-----------+--------+------------+
only showing top 20 rows
```

Result 22.



```
   ...: only_min.count()
+-----------+--------+-------+
|ID         |DATE    |Max_Min|
+-----------+--------+-------+
|AE000041196|19440725|[TMIN] |
|AE000041196|19450301|[TMIN] |
|AE000041196|19450503|[TMIN] |
|AE000041196|19450510|[TMIN] |
|AE000041196|19450521|[TMIN] |
+-----------+--------+-------+
only showing top 5 rows

Out[18]: 7528188
```

Result 23

How many different stations contributed to these observations? Do they belong to the GSN, HCN, or CRN?

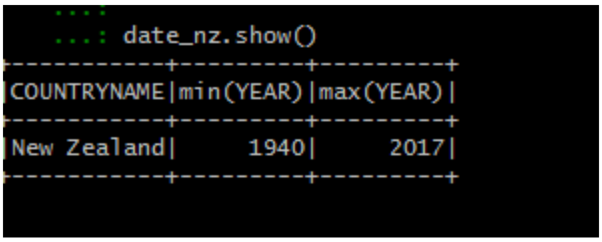
These observations contains 26,625 stations. And there are 2,111 stations belong to GSN, HCN or CRN.

(d) How many observations are there, and how many years are covered by the observations.

There are 447,017 observations in New Zealand that have minimum temperature and maximum temperature records. And those observations cover from year 1940 to 2017. The results showed as below,

```
In [22]: daily_NZ.show(5,False)
|ID         |STATE|COUNTRYCODE|LATITUDE|LONGITUDE|ELEVATION|STATIONNAME|GSNFLAG|HCN/CRNFLAG|WMOID|COUNTRYNAME|STATENAME|CORECOUNT|DISTINCTCOUNT|OTHERCOUNT|FIRSTYEAR|LASTERYEAR|DATE    |ELEMENT|VALUE|MEASUREMENT FLAG|QUALITY FLAG|SOURCE FLAG|OBSERVATION TIME|
|NZ000093012|     |NZ         |-35.1   |173.267  |54.0     |KAITAIA    |       |           |93119|New Zealand|null     |3        |4            |1         |1965     |2017      |20080810|TMAX   |190.0|null            |null        |S          |null            |
|NZ000093012|     |NZ         |-35.1   |173.267  |54.0     |KAITAIA    |       |           |93119|New Zealand|null     |3        |4            |1         |1965     |2017      |20070610|TMAX   |160.0|null            |D           |S          |null            |
|NZ000093012|     |NZ         |-35.1   |173.267  |54.0     |KAITAIA    |       |           |93119|New Zealand|null     |3        |4            |1         |1965     |2017      |20070710|TMAX   |150.0|null            |null        |S          |null            |
|NZ000093012|     |NZ         |-35.1   |173.267  |54.0     |KAITAIA    |       |           |93119|New Zealand|null     |3        |4            |1         |1965     |2017      |20070910|TMAX   |160.0|null            |D           |S          |null            |
|NZ000093012|     |NZ         |-35.1   |173.267  |54.0     |KAITAIA    |       |           |93119|New Zealand|null     |3        |4            |1         |1965     |2017      |20040101|TMAX   |234.0|null            |null        |G          |null            |
only showing top 5 rows
```

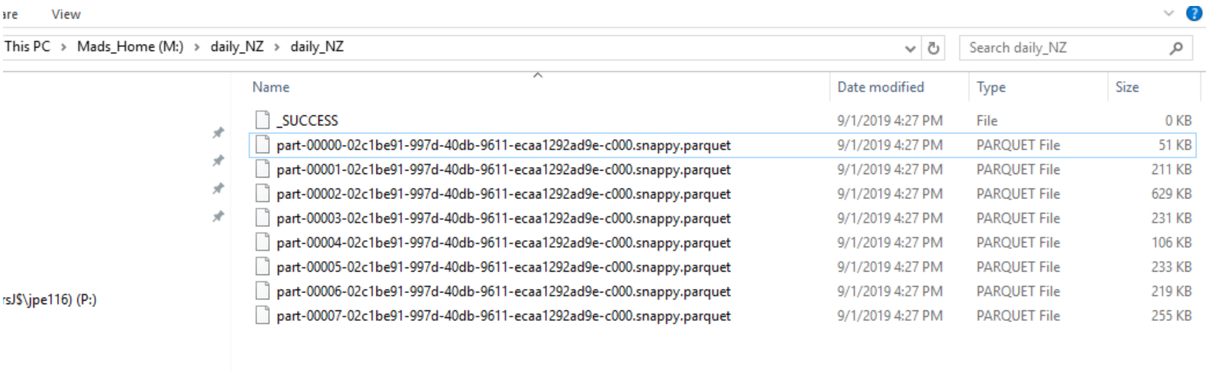
Result 24.



```
   ...: date_nz.show()
+-----------+---------+---------+
|COUNTRYNAME|min(YEAR)|max(YEAR)|
+-----------+---------+---------+
|New Zealand|     1940|     2017|
+-----------+---------+---------+
```

Result 25.

The output in local directory as below.

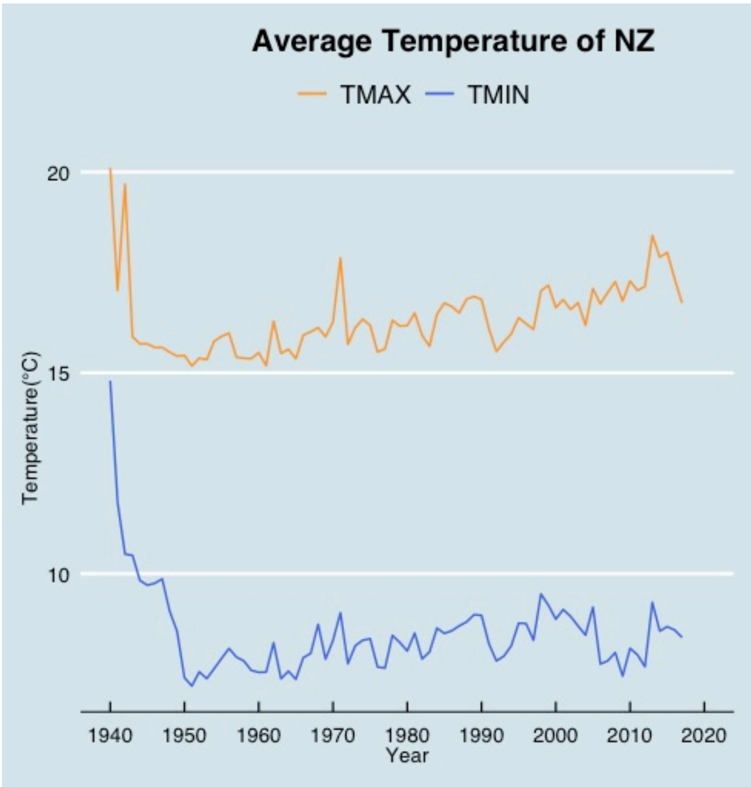


Figures 10

The number of rows in the part files of CSV files matched the number of observations using Spark.

Plot the time series of TMIN and TMAX on the same axis for each station in New Zealand. And plot the average time series for the entire country.

The average time series for the entire country as the result 26. And the other plots for each station as appendix.



Result 26

(e) Group the precipitation observations by year and country. Compute the average rainfall in each year for each country.

Which country has the highest average rainfall in a single year across the entire dataset.

Plot the cumulative rainfall for each country.

The precipitation observations by year and country showed as the result 27. The highest average rainfall happens in Equatorial Guinea in year 2000.

```
   ...: average_rainfall.cache()
   ...: average_rainfall.show()
+----+--------------------+-----------+------------------+
|YEAR|         COUNTRYNAME|COUNTRYCODE|       AVG_RAINFALL|
+----+--------------------+-----------+------------------+
|2000|   Equatorial Guinea|         EK|            4361.0|
|1975|  Dominican Republic|         DR|            3414.0|
|1974|                Laos|         LA|            2480.5|
|1978|              Belize|         BH| 2244.714285714286|
|1974|          Costa Rica|         CS|            1820.0|
|1979|              Belize|         BH|1755.5454545454545|
|1973|            Suriname|         NS|            1710.0|
|1977|              Belize|         BH|1541.7142857142858|
|1978|Netherlands Antil...|         NT|1495.6530612244899|
|1979|Netherlands Antil...|         NT|           1487.72|
|1978|            Honduras|         HO|1469.6122448979593|
|1977|Netherlands Antil...|         NT| 1332.378787878788|
|2009|                Iraq|         IZ|            1300.0|
|1977|            Honduras|         HO| 1284.138888888889|
|1978| Trinidad and Tobago|         TD|            1265.0|
|1976|              Guyana|         GY|1213.3333333333333|
|1973|             Tunisia|         TS|            1162.0|
|2006|               Burma|         BM|            1152.0|
|2001|   Equatorial Guinea|         EK|            1100.0|
|1977| Martinique [France]|         MB|           1045.32|
+----+--------------------+-----------+------------------+
only showing top 20 rows
```

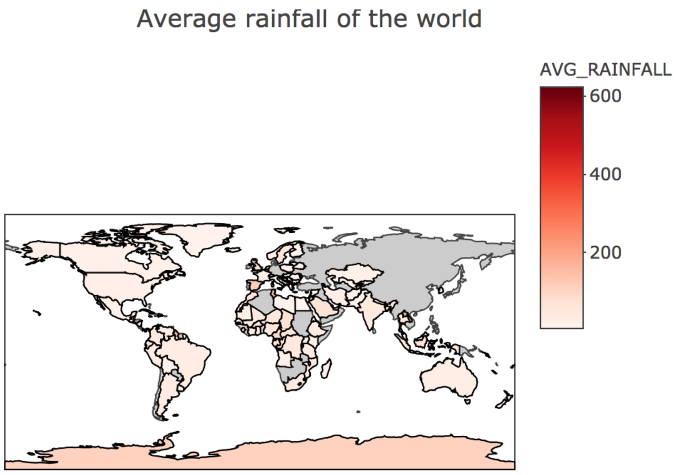
Result 27.

The cumulative rainfall for each country showed as the result 28.

```
   ...: cummu_rainfall_country.show(5,False)
+-----------+--------------------+------------------+
|COUNTRYCODE|COUNTRYNAME         |AVG_RAINFALL      |
+-----------+--------------------+------------------+
|EK         |Equatorial Guinea   |623.6077594289269 |
|BH         |Belize              |192.37566641740898|
|CS         |Costa Rica          |182.55250413781388|
|TL         |Tokelau [New Zealand] |179.44281097228026|
|NC         |New Caledonia [France]|172.84295899940219|
+-----------+--------------------+------------------+
only showing top 5 rows
```

Result 28

The plot as below,



Result 29.

Challenges

Q1

I am interested in research wind speed and wind direction value of the USA. First, I can obtain the overall information that observations including *ELEMENT* of *AWND* and *AWDR* in the territory of United States. Then the years that these observations covered can be found. Finally, I visualized the data of wind speed and wind direction of two stations in 2017. So, the overall information showed as followed. Then the year data showed as the result 31.

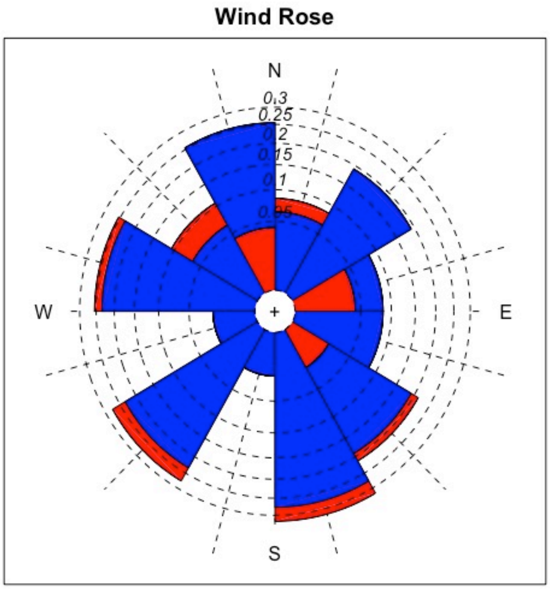
```
...: wind_USA.cache()
...: wind_USA.show()
+-----------+--------+-------+-----+----+-------------+-----------+
|         ID|    DATE|ELEMENT|VALUE|YEAR|  COUNTRYNAME|COUNTRYCODE|
+-----------+--------+-------+-----+----+-------------+-----------+
|USW00094626|20100101|   AWND| 36.0|2010|United States|         US|
|USW00014719|20100101|   AWND| 10.0|2010|United States|         US|
|USW00024061|20100101|   AWND| 20.0|2010|United States|         US|
|USW00024229|20100101|   AWND| 58.0|2010|United States|         US|
|USW00012876|20100101|   AWND| 34.0|2010|United States|         US|
|USW00012842|20100101|   AWND| 38.0|2010|United States|         US|
|USW00003967|20100101|   AWND| 30.0|2010|United States|         US|
|USW00003889|20100101|   AWND| 32.0|2010|United States|         US|
|USW00093721|20100101|   AWND| 30.0|2010|United States|         US|
|USW00093990|20100101|   AWND| 38.0|2010|United States|         US|
|USW00023110|20100101|   AWND| 13.0|2010|United States|         US|
|USW00012935|20100101|   AWND| 53.0|2010|United States|         US|
|USW00003030|20100101|   AWND| 45.0|2010|United States|         US|
|USW00053909|20100101|   AWND| 18.0|2010|United States|         US|
|USW00093009|20100101|   AWND| 38.0|2010|United States|         US|
|USW00053855|20100101|   AWND| 47.0|2010|United States|         US|
|USW00014770|20100101|   AWND| 20.0|2010|United States|         US|
|USW00012844|20100101|   AWND| 40.0|2010|United States|         US|
|USW00004839|20100101|   AWND| 23.0|2010|United States|         US|
|USW00004842|20100101|   AWND| 62.0|2010|United States|         US|
+-----------+--------+-------+-----+----+-------------+-----------+
only showing top 20 rows
```

Result 30

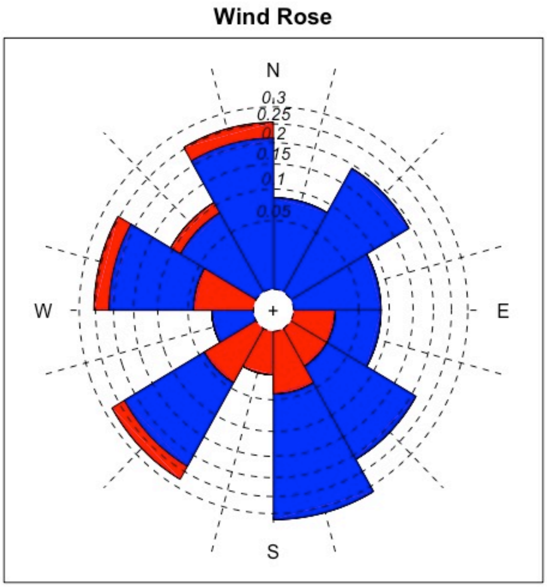
```
...: wind_USA_year.show()
+--------------------+---------+---------+
|         COUNTRYNAME|min(YEAR)|max(YEAR)|
+--------------------+---------+---------+
|American Samoa [U...|     1984|     2017|
|Puerto Rico [Unit...|     1984|     2017|
|       United States|     1982|     2017|
|Midway Islands [U...|     1986|     1995|
|Northern Mariana ...|     2006|     2017|
|Virgin Islands [U...|     1994|     2017|
|Wake Island [Unit...|     1984|     1997|
|Guam [United States]|     1984|     2017|
|Johnston Atoll [U...|     1984|     1995|
+--------------------+---------+---------+
```

Result 31.

After choosing the stations that have the top 2 daily wind speed in 2017, my aim is to visual data of station USS0006N04S, USS0006P10S. The plot showed as the result 32 and 33.



Result32- USS0006N04S



Result 33 - USS0006P10S

Q2

There are 7,688,694 rows in *daily* marked with failed flag. The result showed as the result 34.

In my opinion, the quality of the dataset is acceptable. There are main reasons. First, the data is complete and continuous. *Daily* covers the data of 255 years without missing any year. Second, the redundant of data is limit. In term of this case, all field is considered as important items and long data frame is better than wide data frame. At last, the data is worth trusty. There are total 2,624,027,105 observations in *daily*, which means 99.7% of the observations passed the quality assurance check.

```
   ...: fail_quality_check.count()
+-----------+--------+-------+-----+----------------+------------+-----------+----------------+
|ID         |DATE    |ELEMENT|VALUE|MEASUREMENT FLAG|QUALITY FLAG|SOURCE FLAG|OBSERVATION TIME|
+-----------+--------+-------+-----+----------------+------------+-----------+----------------+
|NOE00109640|20100101|PRCP   |0.0  |null            |I           |E          |null            |
|NOE00109640|20100101|SNWD   |100.0|null            |I           |E          |null            |
|USR0000AHLM|20100101|TMAX   |0.0  |H               |D           |U          |null            |
|USR0000AHLM|20100101|TMIN   |0.0  |H               |D           |U          |null            |
|RSM00031961|20100101|SNWD   |79.0 |null            |I           |S          |null            |
|USR0000ALIR|20100101|TMAX   |39.0 |H               |I           |U          |null            |
|FR069029001|20100101|TMIN   |35.0 |null            |I           |E          |null            |
|USW00094814|20100101|WESD   |0.0  |null            |I           |0          |null            |
|USW00094010|20100101|SNOW   |51.0 |null            |I           |0          |null            |
|USW00094010|20100101|SNWD   |152.0|null            |I           |0          |null            |
+-----------+--------+-------+-----+----------------+------------+-----------+----------------+
only showing top 10 rows

Out[32]: 7688694
```

Result 34.

I am interested in finding the relationship between wind speed and season in the USA, and I wonder know if we can predict season by the wind speed value. So, I used the wind_USA data frame from previous then sampling randomly them by the size of 500. Next, I chose the top 300 records as training data and the rest as testing data. Following training dataset were ready, I built the model by K-nearest neighbour method. When I chose K = 3, the output showed as followed. And the mean error rate is 45%.

```
knn.pred  1  2  3  4
       1 31  9  4 42
       2  0  0  0  1
       3  0  0  0  0
       4 37 11  5 60
> # get the mean square of error
> mean(knn.pred == test$sn)
[1] 0.455
```

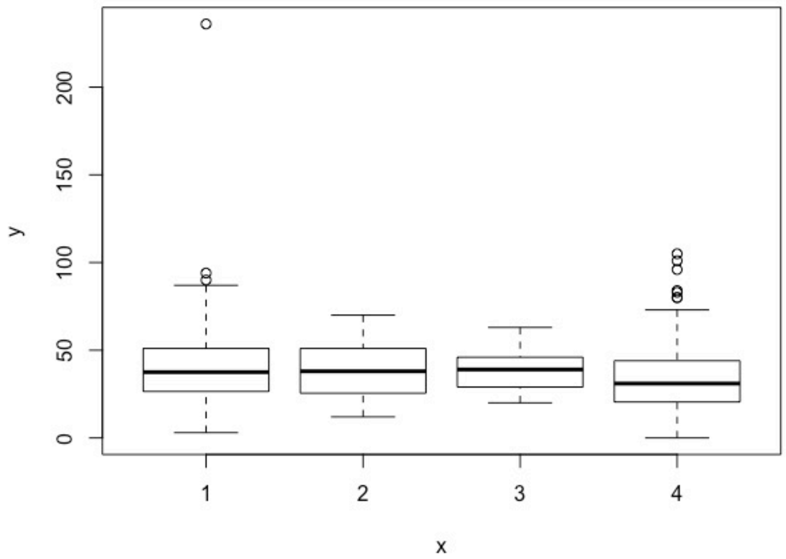
Result 35.

However, when k = 1, the output showed as followed. The mean error rate is still 43% which means the prediction is very poor.

```
knn.pred  1  2  3  4
       1 29  9  2 37
       2  1  0  3  3
       3  1  1  0  6
       4 37 10  4 57
> # get he mean square of error
> mean(knn.pred == test$sn)
[1] 0.43
```

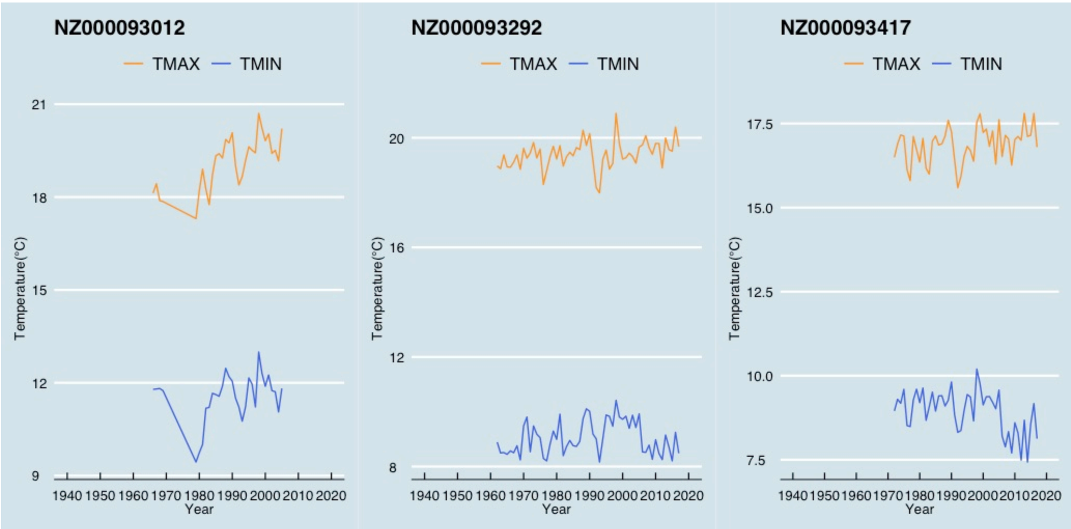
Result 36.

Actually, I can see no patterns when I plot the data of wind speed and season directly. This showed as followed. I also try to find the relationship between wind speed and day or month, as a result, the relationship both are weak. Maybe the prediction would be better with the multiple variables such as temperature and pressure.

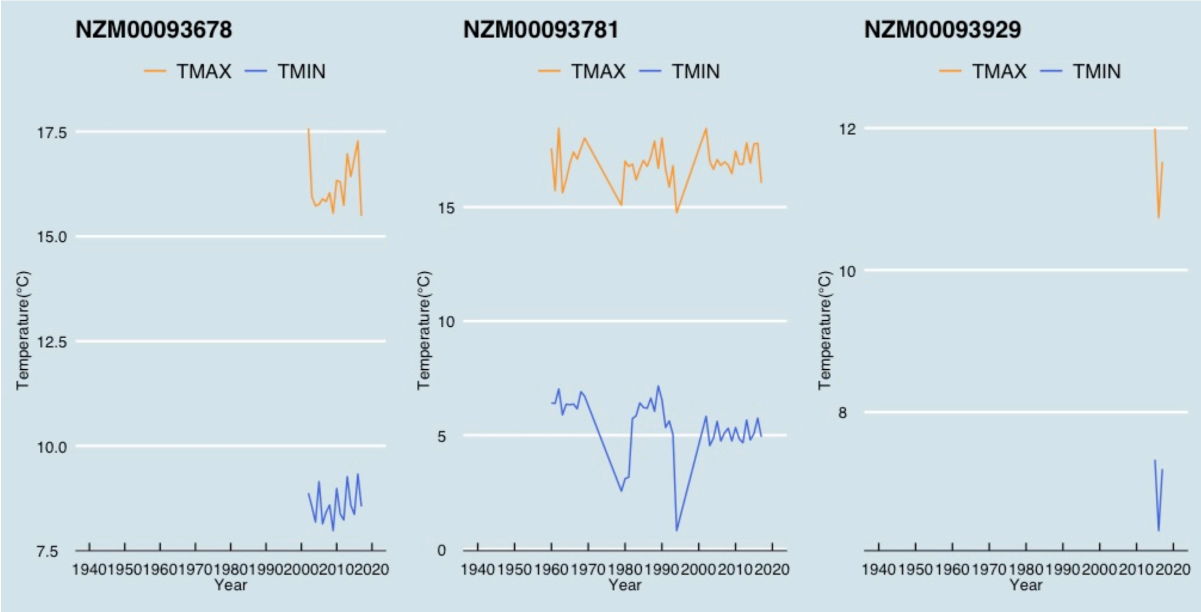


Result 37.

Appendix




NZ000093844
TMAX TMIN
Temperature(°C)
Year
NZ000093994
TMAX TMIN
Temperature(°C)
Year
NZ000933090
TMAX TMIN
Temperature(°C)
Year
NZ000936150
TMAX TMIN
Temperature(°C)
Year
NZ000937470
TMAX TMIN
Temperature(°C)
Year
NZ000939450
TMAX TMIN
Temperature(°C)
Year
NZ000939870
TMAX TMIN
Temperature(°C)
Year
NZM00093110
TMAX TMIN
Temperature(°C)
Year
NZM00093439
TMAX TMIN
Temperature(°C)
Year


NZM00093678
TMAX
TMIN
17.5
15.0
12.5
10.0
7.5
Temperature(°C)
1940 1950 1960 1970 1980 1990 2000 2010 2020
Year
NZM00093781
TMAX
TMIN
15
10
5
0
Temperature(°C)
1940 1950 1960 1970 1980 1990 2000 2010 2020
Year
NZM00093929
TMAX
TMIN
12
10
8
Temperature(°C)
1940 1950 1960 1970 1980 1990 2000 2010 2020
Year