**Analyze Historical Weather Data with SQL Server**

By: [Rick Dobson](https://www.mssqltips.com/sqlserverauthor/57/rick-dobson/)   |   Updated: 2020-08-06

FROM: <https://www.mssqltips.com/sqlservertip/6504/analyze-historical-weather-data-with-sql-server/?utm_source=dailynewsletter&utm_medium=email&utm_content=headline&utm_campaign=20200806>

**Problem**

Our company wants to develop and evaluate applications for daily historical weather trends. However, we do not have a process for collecting daily historical weather from a quality source. Please present how to collect, insert, and make available historical weather data from SQL Server.

**Solution**

The National Weather Service (NWS) Cooperative Observer Program (COOP) is sometimes referred to as the nation's weather and climate observing network of, by and for the people. The COOP is comprised of volunteer weather observers who use standardized practices to collect daily surface weather, such as temperature, rainfall, snowfall, and wind. The historical observations by COOP volunteers comprise a quality data source for daily historical weather trends.

The formal origins of the COOP date back to the Organic act of Congress in 1890 that established the NWS. A core set of weather stations were in operation prior to the act. Notable historical figures serving as weather observers include Thomas Jefferson, Benjamin Franklin, and George Washington.

Weather stations are added to and dropped from the COOP network from time to time. There are currently around nine thousand COOP weather stations dispersed across the United States and its territories. Daily weather observations are transferred from the COOP station operators to the NWS. Metadata for the weather stations as well as weather observations are available from [web sites](https://www.ncdc.noaa.gov/) maintained by the National Oceanic and Atmospheric Administration (NOAA) of the US Department of Commerce. There are also other government web pages specifically on the history and general support roles for American agriculture and business of the COOP network (such as [here](https://www.weather.gov/coop/overview) and [here](https://www.weather.gov/pah/coop)).

An introduction to COOP weather data

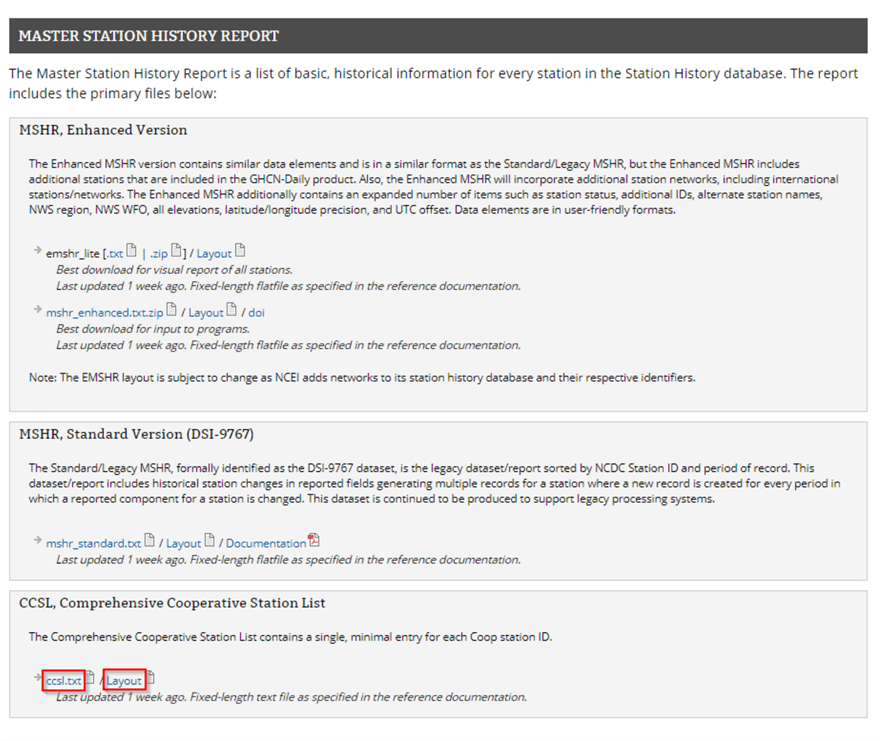
There are two key components to COOP weather data observations.

The base component is a list of COOP weather stations. This list maintains station metadata, such as station name, the latitude and longitude of each COOP station, as well as the end date for daily observations from each station. You can think of this list as a source for metadata about stations.

Each COOP station can generate multiple time series. When dealing with daily observations, the period for each time series will be one day. There will typically be more than one time series for a station. For example, regarding daily temperature, there are three time series for daily minimum, maximum, and average temperature.

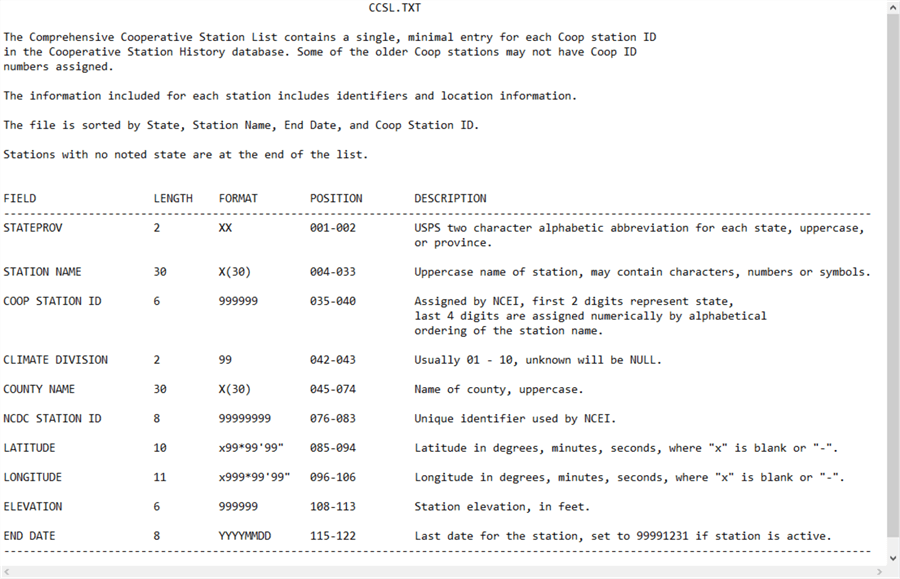
Collecting metadata for stations

The most recent COOP station metadata is maintained in a pair of txt files that can be downloaded from the [NOAA Historical Observing Metadata Repository](https://www.ncdc.noaa.gov/homr/reports). An excerpt from the NOAA web page with COOP station metadata appears below. The NOAA maintains three lists of weather observing stations. The third section on the page excerpt below is for COOP station metadata. Within this section. there are two red box markups that each highlights a hyperlink. One box, ccsl.txt, is for the metadata contents. The contents reside in a fixed-length txt file. The second box embraces Layout, which is for a txt file named CCSL\_TABLE.txt. The contents of the second file denote the name, length, format, position, and description for each column in the ccsl.txt file.



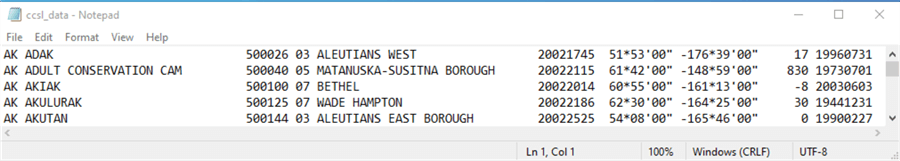
Here’s the contents of the CCSL\_table.txt file.

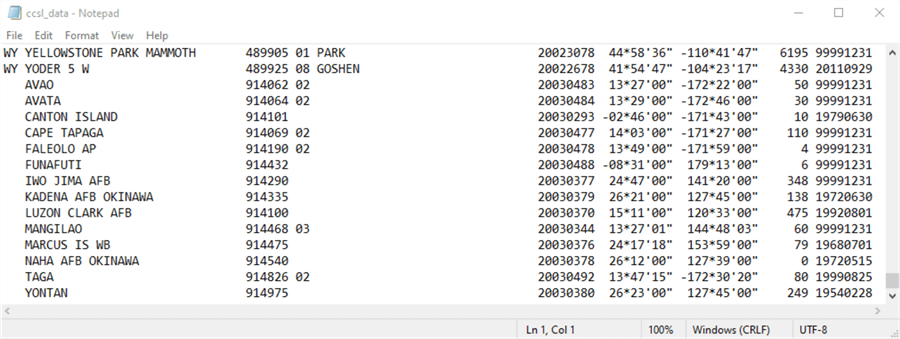
* There are data for ten columns in the ccsl.txt file.
* The first column has the name STATEPROV, which indicates a state or territory abbreviation associated with a weather station’s locale.
* The last column has the name END DATE, which denotes the last date during which there was an observation at a station. An END DATE value of 99991231 denotes that a station is still being observed on a daily basis.



Here are two excerpts from the ccsl.txt file.

* The first excerpt is for the first five weather stations from the state of Alaska (AK).
* The second excerpt is for 16 weather stations at the end of the file.
  + The first two rows are for locales in the state of Wyoming.
  + The last fourteen rows do not have a STATEPROV identifier value. These last fourteen rows are for locales that do not correspond to places for which the United States Postal Service (USPS) has a two-character state/province abbreviation.
  + There is a list of the USPS abbreviations for states and territories at this [web page](https://faq.usps.com/s/article/What-are-the-USPS-abbreviations-for-U-S-states-and-territories).





The following T-SQL script reads a subset of the contents from the ccsl.txt file into a SQL Server table.

* The code begins with a use statement to reference the noaa\_data database as the default database. You could use any other database of your choice as the default database.
* The next code block creates a fresh version of the ImportedFileTable object. This table will be used to store the unparsed line values from the ccsl.txt file downloaded from the NOAA website.
* Then, a bulk insert statement reads the data from the ccsl.txt file in the noaa\_data folder of the C drive.
* After populating the ImportedFileTable object, the code parses each line and filters the resulting rows with a select…into statement.
  + The select list items specify expressions for parsing the values on each row in the ImportedFileTable object into columns within the ccsl\_still\_in\_use table within the dbo schema of the default database.
  + The where clause filters the results set in the target table so that only rows with an END DATE column value of ‘99991231’ are retained. This is a NOAA code to denote that a weather station is still in operation. Other END DATE values denote the date in YYYYMMDD format for the last observation from a station. The where clause excludes rows not still in operation from the ccsl\_still\_in\_use table.

use noaa\_data

go

-- Collect info about ccsl weather stations for

-- those still in use

-- ccsl.txt file downloaded on‎May‎12,‎2020

-- Remove prior ImportedFileTable

if EXISTS(SELECT \* FROM sys.objects WHERE name = 'ImportedFileTable' and TYPE = 'u')

DROP TABLE ImportedFileTable

-- Create ImportedFileTable

CREATE TABLE ImportedFileTable(

textvalue varchar(max)

)

GO

-- Import text file for Comprehensive Cooperative Station List (ccsl)

BULK INSERT ImportedFileTable

FROM 'C:\noaa\_data\ccsl\_data.txt'

begin try

drop table dbo.ccsl\_still\_in\_use

end try

begin catch

print 'ccsl\_still\_in\_use table not available to drop'

end catch

-- select all stations from 'C:\noaa\_data\ccsl\_data.txt'

-- convert latitude and longitude from original string (LATITUDE and LONGITUDE)

-- to string representation in dd.mmss (lat\_dd.mmss and long\_dd.mmss)

-- station name starting with 'UTQIA' omitted

-- because of lack of collation support for character in station name

-- no explicit criterion to omit the station because it is no longer operational

-- and the result set from this query is only for currently operating

-- weather stations

select

left(textvalue,2) STATEPROV

,substring(textvalue,4,30) [STATION NAME]

,substring(textvalue,35,6) [COOP STATION ID]

,substring(textvalue,42,2) [CLIMATE DIVISION]

,substring(textvalue,45,30) [COUNTY NAME]

,substring(textvalue,76,8) [NCDC STATION ID]

,substring(textvalue,85,10) [LATITUDE]

,LEFT(substring(textvalue,85,10),3) + '.'

+ LEFT(substring(textvalue,89,3),2)

+ LEFT(substring(textvalue,92,3),2) [lat\_dd.mmss]

,substring(textvalue,96,11) [LONGITUDE]

,LEFT(substring(textvalue,96,4),4) + '.'

+ LEFT(substring(textvalue,101,2),2)

+ LEFT(substring(textvalue,104,2),2) [long\_dd.mmss]

,substring(textvalue,108,5) [ELEVATION]

,substring(textvalue,115,8) [END DATE]

into dbo.ccsl\_still\_in\_use

from ImportedFileTable

where substring(textvalue,115,8) = '99991231' -- station is still in operation

Once you have populated the ccsl\_still\_in\_use table, you can query it to report data about the weather stations. Here’s a simple script to display all the rows in the ccsl\_still\_in\_use table. It orders its results set by stateprov in desccending order and then by station name and coop station id in ascending order.

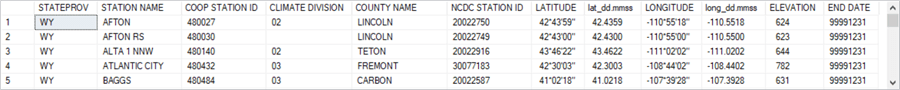
-- sorted by stateprov, [station name], [end date], [coop station id]

-- stations with no stateprov value are at the end of the list

select \* from dbo.ccsl\_still\_in\_use

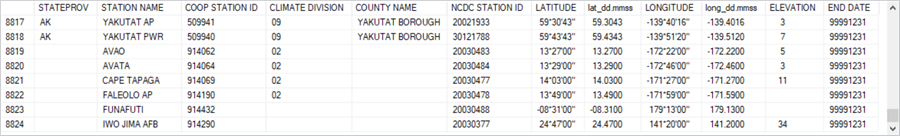
order by stateprov desc, [station name], [coop station id]

Here are the first five rows from the results set from the preceding script. All five rows are for weather stations in Wyoming. The first two rows are for weather stations in Lincoln county. The next three rows are for weather stations in Teton, Fremont, and Carbon counties.



The next screen shot is for the last eight rows in the results set from the preceding script.

* The first two rows are for weather stations within Yakutat Borough in the state of Alaska (AK).
* The next six rows are for weather stations that are not located in a locale with a USPS two-character state or territory abbreviation. You can, however, specify the location of these weather stations by STATION NAME, COOP STATION ID, or the latitude and longitude of the station location.



The next query counts the number of weather stations by USPS two-character state/territory identifier.

-- count of stations by USPS two-character state/territory identifier

select stateprov, count(\*) count\_of\_stations\_by\_stateprov

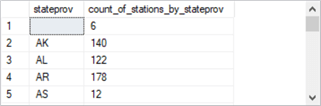
from dbo.ccsl\_still\_in\_use

group by stateprov

order by stateprov

Here’s the first five rows from the results set of the preceding query.

* The count for the first row confirms the prior listing by showing there are six weather stations not associated with a USPS two-character state/territory abbreviation.
* The next three rows report the count of weather stations in Alaska (AK), Alabama (AL), and Arkansas (AR).
* The last row is for the count of weather stations in American Somoa (AS).



You may sometimes seek information about one or more weather stations that cannot be directly obtained from a weather station metadata column. In such cases, you may need an ad hoc query based on station name(s), COOP station id values, or even longitude and latitude values.

The following query lists metadata for weather stations in the Washington, DC area. The where clause uses STATION NAME with like operators. The like operators are used twice -- once for an embedded name of ‘Washington’ and a second time for the inclusion of ‘AP’ in STATION NAME.

-- weather stations in airports within Washington, DC area

-- no metadata column directly supports the query

select \* from dbo.ccsl\_still\_in\_use where([STATION NAME] like '%washington%') and [STATION NAME] like '%AP%'

The following results set shows the three rows from the preceding query. The weather station name column values reflect the names of three major airports relatively near Washington, DC.

collecting_weather_data_fig_8

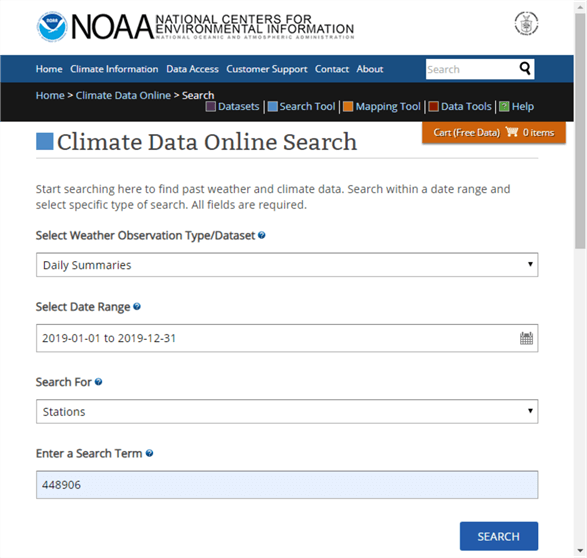
Collecting weather time series for a weather station

After identifying one or more weather stations for collecting weather time series values, you can return to the NOAA website to collect historical weather observations for those stations. This section demonstrates how to collect weather for a station and export the weather observations from the NOAA website in a csv file. The prior section showed how to download and import a fixed-width text file from NOAA (that demonstration was for weather station metadata).

This section walks you through the steps for specifying and extracting a set of daily weather observations for a COOP station from the NOAA website. The example is for the COOP weather station at Washington Reagan National Airport. The remainder of this section presents the sequence of website screens you can encounter for collecting historical weather observations.

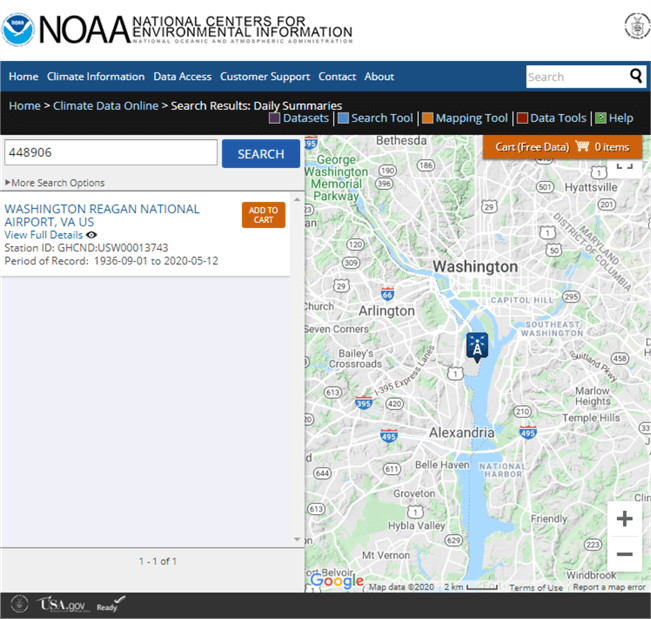
First, go to the internet address at ncdc.noaa.gov/cdo-web/search with your browser. This opens a blank form. The example in the following screen shows the request for weather data.

* The selection of Daily Summaries indicates a request for daily data.
* By clicking the calendar icon in the Select Date Rage box, you can specify a date range for the returned daily weather observations.
* One way to specify a COOP weather station from which to collect observations is by
  + Selecting Stations in the Search For box and
  + Typing a COOP Station ID number in the Enter a search term box
  + The prior screen shot indicates that the COOP Station ID number is 448906, which denotes the weather station at Washington Reagan National Airport.

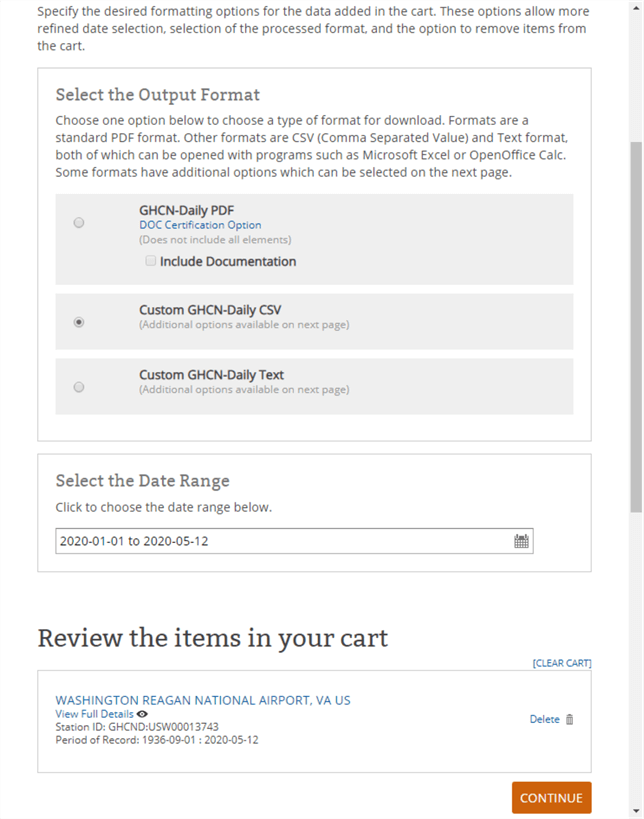


Clicking the Search button on the preceding screen shot opens the ncdc\_noaa\_gov/cdo-web/results page (next screen shot).

* From this page, you can select the weather station designated in the preceding screen by clicking the Add to Cart button. You can designate multiple weather stations if your application requires them.
* Advance to the next screen by clicking the Cart (Free Data) button with one or more selected weather stations.

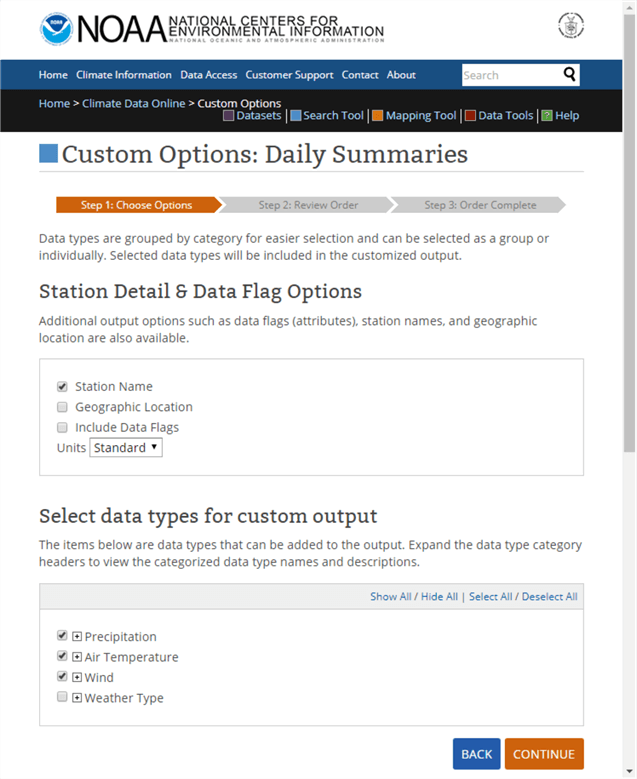


The next screen has a web address of ncdc.noaa.gov/cdo-web/cart. One key function supported by the screen is the specification of a file type for the return weather data. This tip demonstrates the use of the csv file format for output because it is relatively easy to read into SQL Server after you master a few details explained and demonstrated in the section.



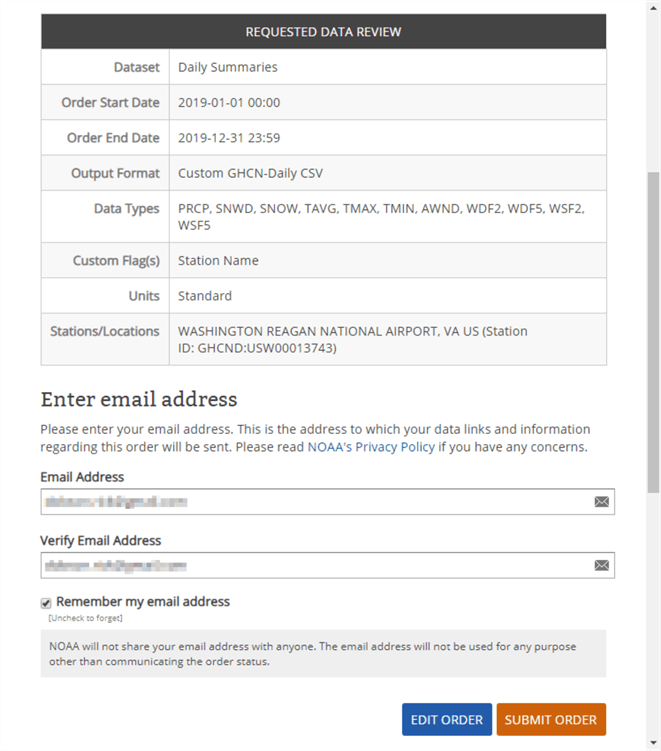
By clicking Continue on the preceding screen, you open what NOAA calls the custom options screen (see the next screen shot).

* The most important set of options allows you to select different types of weather data. In the screen shot below, weather for precipitation, air temperature, and wind are selected.
* You can also choose units as standard English units or metric units. The sample screen below shows the selection of standard English units (for example, inches instead of centimeters).



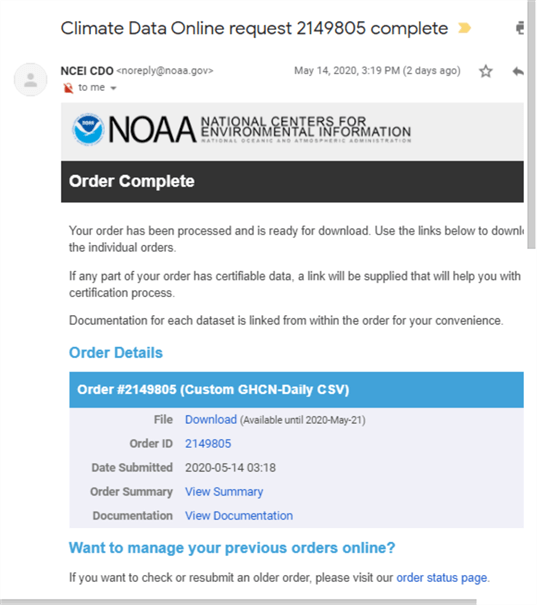
Clicking Continue towards the bottom of the preceding page opens a review web page (ncdc.noaa.gov/cdo-web/review) from which you can examine the weather order and submit the request to NOAA for processing and shipment to an email address.

* You can use the page for navigating through prior pages to modify your weather order or to submit your order as is.
  + The Dataset, Order Start Date, and Order End Date correspond to entries made on the ncdc.noaa.gov/cdo-web/search page.
  + The Output Format corresponds to this tip’s selection in the ncdc.noaa.gov/cdo-web/cart page. In this tip, the csv output format is selected, but two alternate formats are available instead of a csv file.
  + The Data Types names refer to different types of weather, such as rainfall, snowfall, wind, and air temperature.
  + The station name is not a COOP station name or station id. Instead, it is a NOAA name for the weather station(s) specified on the ncdc.noaa.gov/cdo-web/search and ncdc.noaa.gov/cdo-web/cart web pages.
* Before clicking the SUBMIT ORDER button, you must enter and verify an email address for the NOAA to send notices about the status of an order. For a small order, such as one year of daily weather observations for a single weather station, the processing can take noticeably less than one minute.
* The email address that you specify will receive two messages per order.
  + The first message is to confirm that processing has commenced.
  + The second message is to confirm that order request is completed. You can download the weather data file by clicking a link in the second message.



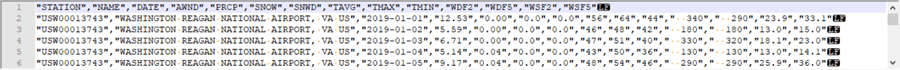
Importing the downloaded NOAA csv formatted weather data file

The following screen shot is an email message excerpt from NOAA to the email address in the preceding screen shot. By clicking the Download link for Order ID 2149805, you can download the csv file created with the NOAA website selections from the prior section.



I copied the csv file (2149805.csv) to the C:\noaa\_data directory. Here is a Notepad++ image of the first six rows from the file. There are 366 rows in the file – one for each day in 2019 and an additional header row.

* There are fourteen data columns from STATION through WSF5.
* Notice that column header designation and column data values are delimited by double quotes. This allows the representation of commas in field values as well as for field delimiters.



When working with transferring data via csv files between Microsoft products, such as from Excel to SQL Server, it is common to use a specification that appears in the following code window.

* Notice that the code excerpt early on creates a fresh version of a table (#daily\_weather\_from\_station\_txt\_flds) to hold the data from the file (C:\noaa\_data\2149805.csv).
* The bulk insert statement has three parameters.
  + The FIRSTROW parameter value of 2 instructs SQL Server to populate the target table with data starting from the second row of the file.
  + The FIELDTERMINATOR parameter value of ‘,’ indicates column values in the source data file are separated by columns.
  + The ROWTERMINATOR parameter value of ‘\n’ tells SQL Server to expect rows in the data file to end with a request for a new line (\n). Windows automatically treats \n as \r\n, which is equivalent to a carriage return followed by a new line.

use noaa\_data

go

begin try

drop table #daily\_weather\_from\_station\_txt\_flds

end try

begin catch

print '#daily\_weather\_from\_station\_txt\_flds table not available to drop'

end catch

go

-- for string fields without fix-ups and transformations

create table #daily\_weather\_from\_station\_txt\_flds

(

noaa\_station\_id nvarchar(15) --USW00013743

,noaa\_station nvarchar(50) -- select len('WASHINGTON REAGAN NATIONAL AIRPORT, VA US')

,[date] nvarchar(50) -- year of the record (4 digits) followed by month (2 digits) and day (2 digits)

,awnd nvarchar(50) -- average wind speed (mph)

,prcp nvarchar(50) -- precipitation (to hundredths of an inch)

,snow nvarchar(50) -- snowfall (to tenths of an inch)

,snwd nvarchar(50) -- snowdepth (to tenths of an inch)

,tavg nvarchar(50) -- represents average of hourly temperatures (Fahrenheit) during an observation period

,tmax nvarchar(50) -- represents maximum of hourly temperatures (Fahrenheit) during an observation period

,tmin nvarchar(50) -- represents minimum of hourly temperatures (Fahrenheit) during an observation period

,wdf2 nvarchar(50) -- direction of fastest 2-minute wind (degrees)

,wdf5 nvarchar(50) -- direction of fastest 5-second wind (degrees)

,wsf2 nvarchar(50) -- fastest 2-minute wind speed (mph)

,wsf5 nvarchar(50) -- fastest 5-second wind speed (mph)

)

go

------------------------------------------------------------------------------------------------------

-- bulk insert from 2149805.csv

-- into #daily\_weather\_from\_station\_txt\_flds

-- with routine settings for

-- fieldterminator and rowterminator

BULK INSERT #daily\_weather\_from\_station\_txt\_flds

FROM 'C:\noaa\_data\2149805.csv'

WITH

(

FIRSTROW = 2,

FIELDTERMINATOR = ',',

ROWTERMINATOR = '\n'

)

-- display #daily\_weather\_from\_station\_txt\_flds table after

-- an attempt to populate it with routine bulk insert settings

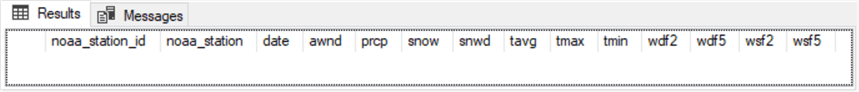
SELECT

\*

FROM #daily\_weather\_from\_station\_txt\_flds

The preceding script ends with a select statement for all the columns in the #daily\_weather\_from\_station\_txt\_flds table. This is the table to which the bulk insert statement attempts to copy data from the 2149805.csv file. Here is the results set from the select statement.

* Notice there are fourteen column headers.
* However, there are no row values.
* The bulk insert statement failed to populate the #daily\_weather\_from\_station\_txt\_flds table with values from 2149805.csv.



Here is a revised version of the preceding script that handles several issues.

* It specifies a special row terminator (0x0a) that is common in Unix and Linux csv files. It turns out that NOAA uses this alternative format to the one used by Microsoft among its own products in a Windows environment.
* Next, the revision adapts a suggestion from a SQL Server professional (Jingyang Li) for handling column field values that are delimited by double quotes, such as 2149805.csv. Li’s [original solution](https://jingyangli.wordpress.com/2014/10/31/import-double-quotes-qualified-csv-file-with-bulk-insert/) needed a slight adjustment for the data from NOAA. The tweak is described in the code review below.
* Finally, the revision below transforms the character data from csv file to different SQL Server data types based on the type of values in the target table’s columns.

There are three major sections to the script. The sections are separated from one another by lines of dashes (-).

* The first section
  + This section declares noaa\_data as the default database and creates a fresh version of the #daily\_weather\_from\_station\_txt\_flds table.
  + This section of the beginning code is the same as the comparable code in the preceding script.
* The second section
  + This section populates the #daily\_weather\_from\_station\_txt\_flds table with the fixes described for the fieldterminator and rowterminator bulk insert parameters.
  + This fix generates a non-empty results set. However, it is not the one required by the application.
  + This sections ends with a select statement for all columns in the #daily\_weather\_from\_station\_txt\_flds table.
* The third section
  + This section starts by creating a fresh copy of the #daily\_weather\_from\_station table. The new #daily\_weather\_from\_station table has typed columns depending on the contents of a column. The #daily\_weather\_from\_station\_txt\_flds table has nvarchar data type settings for all columns.
  + Next, the section executes an insert…select statement to populate the #daily\_weather\_from\_station table from the #daily\_weather\_from\_station\_txt\_flds table. The select statement in the insert…select statement
    - Removes leading and trailing double quote marks
    - Has cast functions to change the assigned data type for a column if that is required
  + This section ends with a select statement for all columns in the #daily\_weather\_from\_station table.

use noaa\_data

go

begin try

drop table #daily\_weather\_from\_station\_txt\_flds

end try

begin catch

print '#daily\_weather\_from\_station\_txt\_flds table not available to drop'

end catch

go

-- for string fields without fix-ups and transformations

create table #daily\_weather\_from\_station\_txt\_flds

(

noaa\_station\_id nvarchar(15) --USW00013743

,noaa\_station nvarchar(50) -- select len('WASHINGTON REAGAN NATIONAL AIRPORT, VA US')

,[date] nvarchar(50) -- year of the record (4 digits) followed by month (2 digits) and day (2 digits)

,awnd nvarchar(50) -- average wind speed (mph)

,prcp nvarchar(50) -- precipitation (to hundredths of an inch)

,snow nvarchar(50) -- snowfall (to tenths of an inch)

,snwd nvarchar(50) -- snowdepth (to tenths of an inch)

,tavg nvarchar(50) -- represents average of hourly temperatures (Fahrenheit) during an observation period

,tmax nvarchar(50) -- represents maximum of hourly temperatures (Fahrenheit) during an observation period

,tmin nvarchar(50) -- represents minimum of hourly temperatures (Fahrenheit) during an observation period

,wdf2 nvarchar(50) -- direction of fastest 2-minute wind (degrees)

,wdf5 nvarchar(50) -- direction of fastest 5-second wind (degrees)

,wsf2 nvarchar(50) -- fastest 2-minute wind speed (mph)

,wsf5 nvarchar(50) -- fastest 5-second wind speed (mph)

)

go

---------------------------------------------------------------------------------------------------

-- bulk insert from 2149805.csv

-- into #daily\_weather\_from\_station\_txt\_flds

-- with special settings for fieldterminator

-- and rowterminator

BULK INSERT #daily\_weather\_from\_station\_txt\_flds

FROM 'C:\noaa\_data\2149805.csv'

WITH

(

FIRSTROW = 2,

FIELDTERMINATOR = '","',

ROWTERMINATOR = '0x0a'

)

-- display #daily\_weather\_from\_station\_txt\_flds table row values

-- without any editing

SELECT

\*

FROM #daily\_weather\_from\_station\_txt\_flds

---------------------------------------------------------------------------------------------------

begin try

drop table dbo.daily\_weather\_from\_station

end try

begin catch

print '#dbo.daily\_weather\_from\_station table not available to drop'

end catch

go

-- for typed fields after fix-ups and transformations

create table dbo.daily\_weather\_from\_station

(

noaa\_station\_id nvarchar(15) --USW00013743

,noaa\_station nvarchar(50) -- select len('WASHINGTON REAGAN NATIONAL AIRPORT, VA US')

,[date] date -- year of the record (4 digits) followed by month (2 digits) and day (2 digits)

,awnd real -- average wind speed (mph)

,prcp real -- precipitation (to hundredths of an inch)

,snow real -- snowfall (to tenths of an inch)

,snwd real -- snowdepth (to tenths of an inch)

,tavg real -- represents average of hourly temperatures (Fahrenheit) during an observation period

,tmax real -- represents maximum of hourly temperatures (Fahrenheit) during an observation period

,tmin real -- represents minimum of hourly temperatures (Fahrenheit) during an observation period

,wdf2 real -- direction of fastest 2-minute wind (degrees)

,wdf5 real -- direction of fastest 5-second wind (degrees)

,wsf2 real -- fastest 2-minute wind speed (mph)

,wsf5 real -- fastest 5-second wind speed (mph)

)

go

-- beginning and ending double quote (") removal

-- as well as data typing for bulk inserted string type field values

insert into dbo.daily\_weather\_from\_station

SELECT

substring(noaa\_station\_id,2,len(noaa\_station\_id)-1) noaa\_station\_id

,noaa\_station

,date date

,cast(awnd as real) awnd

,cast(prcp as real) prcp

,cast(snow as real) snow

,cast(snwd as real) snwd

,cast(tavg as real) tavg

,cast(tmax as real) tmax

,cast(tmin as real) tmin

,cast(wdf2 as real) wdf2

,cast(wdf5 as real) wdf5

,cast(wsf2 as real) wsf2

,cast(rtrim(ltrim(substring(wsf5,1,len(wsf5)-1))) as real) wsf5

FROM #daily\_weather\_from\_station\_txt\_flds

-- display #daily\_weather\_from\_station table row values

-- after any editing

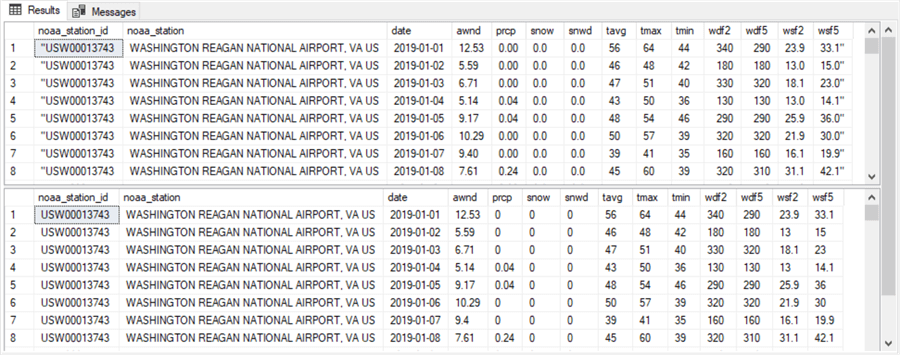
select

\*

from dbo.daily\_weather\_from\_station

Here are excerpts from the results sets in the preceding script.

* The top pane shows the first eight rows from the results set for the #daily\_weather\_from\_station\_txt\_flds table.
  + Notice the leading double quote for the noaa\_station\_id column.
  + Also notice the trailing double quote for the wsf5 column.
* The bottom pane shows the first eight rows from the results set for the #daily\_weather\_from\_station table.
  + Notice the leading and trailing double quote marks are removed.
  + The numeric columns also have fresh data types; they are not character representations of numeric values.



**Next Steps**

There are four steps for re-creating the contents reported in this tip and confirming your grasp of how to make daily weather observations available from a SQL Server instance.

1. Review the process for downloading the COOP weather station metadata file (ccsl.txt) and its descriptor file (CCSL\_table.txt). The "Collecting metadata for stations" section includes the NOAA web page from which to download the files. For your convenience, the ccsl.txt file is also included among the [download files from this tip](https://www.mssqltips.com/tipimages2/6504_scripts.zip).
2. Run the T-SQL code for transferring a subset of the contents of ccsl.txt to a SQL Server table named ccsl\_still\_in\_use. Then, run the queries in the "Collecting metadata for stations" section to confirm you are able to return results sets that match those reported for the ccsl\_still\_in\_use table.
3. Follow the steps in the "Collecting weather time series for a weather station" and "Importing the downloaded NOAA csv formatted weather data file" sections to download a csv file with daily weather for the COOP weather station at Washington Reagan National Airport. For your convenience, the csv file used in this tip is also included among the download files from this tip.
4. Run the T-SQL code for importing the downloaded csv file into a SQL Server table and query the table to display the contents of the downloaded csv file.

After you confirm that the download process and code are generating the same results as in this tip, you can adapt the process described in the preceding steps to collect weather data for whatever other locales your needs require.

Next, consider staying on the lookout for follow-up tips to this one that dive into more detail on how to process the metadata for weather stations and how to load weather data for multiple weather stations into a SQL Server data warehouse. Also, examine a forthcoming tip on how to assess seasonality with T-SQL code for weather data like those downloaded with the steps in this tip.

Last Updated: 2020-08-06