Tutorial 4b: Extending your analysis with more datasets

This tutorial was built for BDCS-CE version 17.4.1 as part of the New Data Lake User Journey: here (https://github.com/oracle/learning-library/tree/master/workshops/journey2-new-data-lake). Questions and feedback about the tutorial: david.bayard@oracle.com (mailto:david.bayard@oracle.com)

```
Be sure you previously ran the Tutorial: "Working with the Spark Interpreter"
```

This tutorial will add some additional datasets (weather and holidays) to combine with our bike trip data.

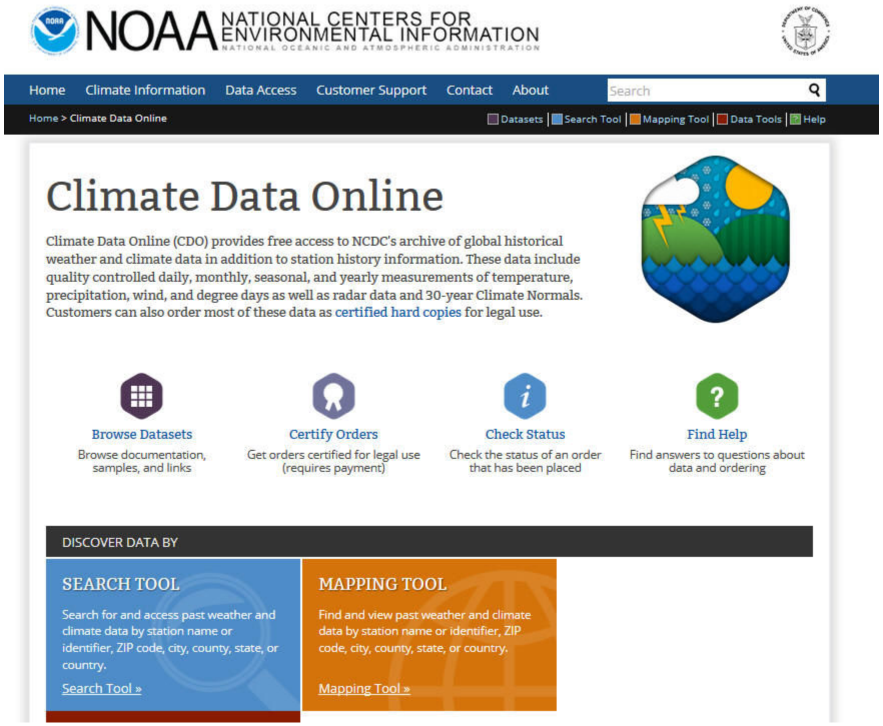
Contents

- How to get daily weather data from the US Government
- Creating a temporary table on weather data
- Creating a temporary table on calendar/holiday data
- Joining our bike_trips with weather and calendar data
- Creating a permanent hive parquet table on the combined data

As a reminder, the documentation for BDCS-CE can be found: here (http://docs.oracle.com/cloud/latest/big-data-compute-cloud/index.html)

Requesting free Weather Data from the Government

In this tutorial, we will add some weather data to our data lake. We will use the US Government's National Center for Environmental Information website to request the daily weather summary for New York City for December 2016 (the same time frame as our bike data). We will use the website to request/order the weather data and then we download the data once the request is ready.


NOAA NATIONAL CENTERS FOR ENVIRONMENTAL INFORMATION
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION
Home
Climate Information
Data Access
Customer Support
Contact
About
Search
Home > Climate Data Online
Datasets
Search Tool
Mapping Tool
Data Tools
Help

Climate Data Online

Climate Data Online (CDO) provides free access to NCDC's archive of global historical weather and climate data in addition to station history information. These data include quality controlled daily, monthly, seasonal, and yearly measurements of temperature, precipitation, wind, and degree days as well as radar data and 30-year Climate Normals. Customers can also order most of these data as certified hard copies for legal use.

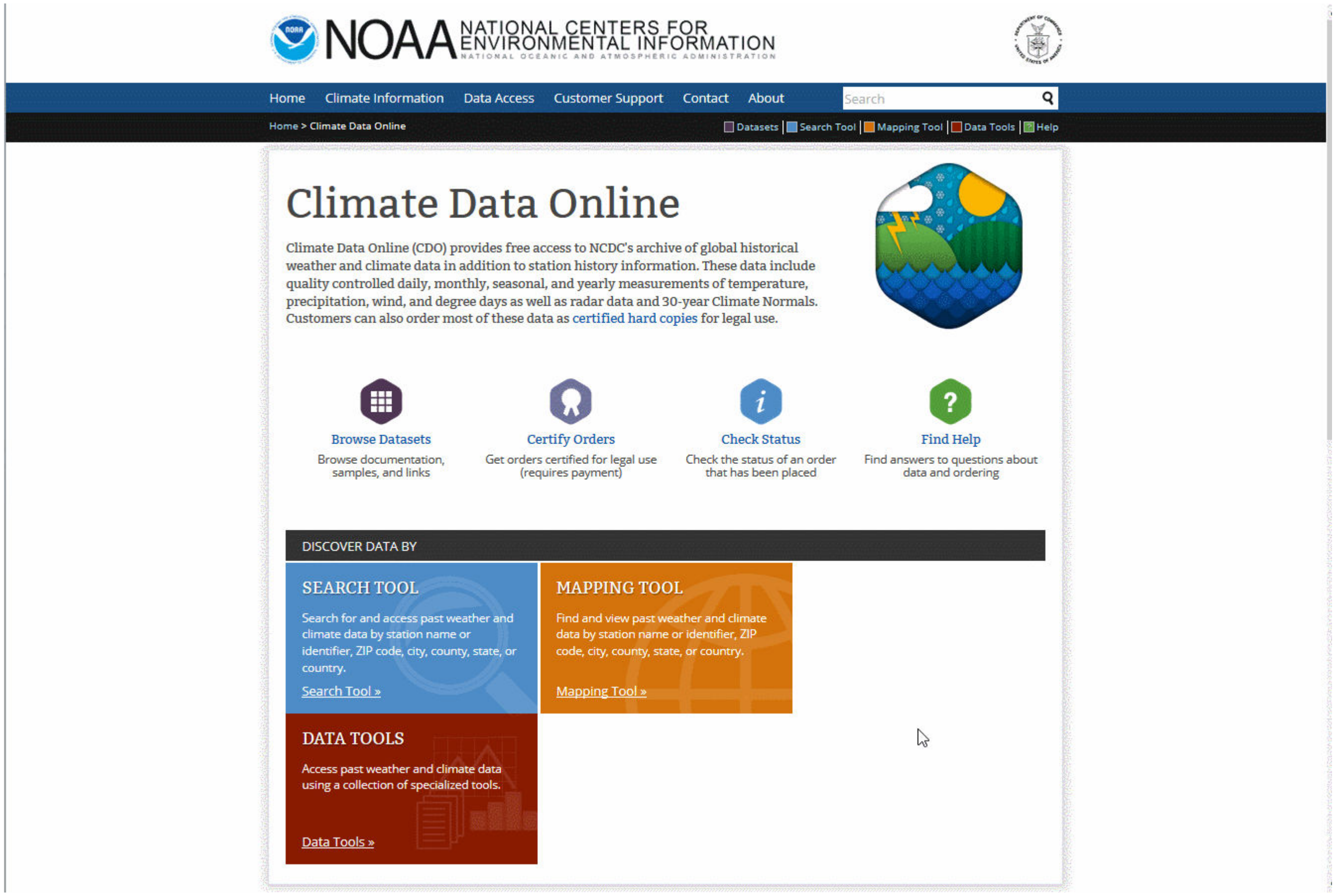
Browse Datasets
Browse documentation, samples, and links
Certify Orders
Get orders certified for legal use (requires payment)
Check Status
Check the status of an order that has been placed
Find Help
Find answers to questions about data and ordering
DISCOVER DATA BY
SEARCH TOOL

Search for and access past weather and climate data by station name or identifier, ZIP code, city, county, state, or country.

Find and view past weather and climate data by station name or identifier, ZIP code, city, county, state, or country.

To get our weather data, follow these steps: (if you are in a hurry, you can skip below to the paragraph titled “If you are in a hurry…”)

- Open a browser to the Climate Data Online website, run by the US Government: https ://www.ncdc.noaa.gov/cdo-web/ (https://www.ncdc.noaa.gov/cdo-web/)
- Click on Search Tool
- Choose:
  Daily Summaries
  2016-12-01 to 2016-12-31
  Cites
  New York City
- Click the “ADD TO CART” button for the “New York, NY US” row.
- Navigate to the “Cart (Free Data) 1 item” area and click “View All Items(1)”
- Choose Custom GHCN-Daily CSV and click Continue
- In the “Select data types for custom output” section, click “Show All”. Then select
  PRCP - Precipitation
  SNWD - Snow depth
  SNOW - Snowfall
  TAVG - Average Temperature.
  TMAX - Maximum temperature
  TMIN - Minimum temperature
  AWND - Average wind speed
- Click Continue
- Enter your email address and click Submit Order


NOAA NATIONAL CENTERS FOR ENVIRONMENTAL INFORMATION
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION
Home
Climate Information
Data Access
Customer Support
Contact
About
Search
Home > Climate Data Online
Datasets
Search Tool
Mapping Tool
Data Tools
Help

Climate Data Online

Climate Data Online (CDO) provides free access to NCDC's archive of global historical weather and climate data in addition to station history information. These data include quality controlled daily, monthly, seasonal, and yearly measurements of temperature, precipitation, wind, and degree days as well as radar data and 30-year Climate Normals. Customers can also order most of these data as certified hard copies for legal use.

Browse Datasets
Browse documentation, samples, and links

Certify Orders
Get orders certified for legal use (requires payment)

Check Status
Check the status of an order that has been placed

Find Help
Find answers to questions about data and ordering

DISCOVER DATA BY

SEARCH TOOL
Search for and access past weather and climate data by station name or identifier, ZIP code, city, county, state, or country.
Search Tool »

MAPPING TOOL
Find and view past weather and climate data by station name or identifier, ZIP code, city, county, state, or country.
Mapping Tool »

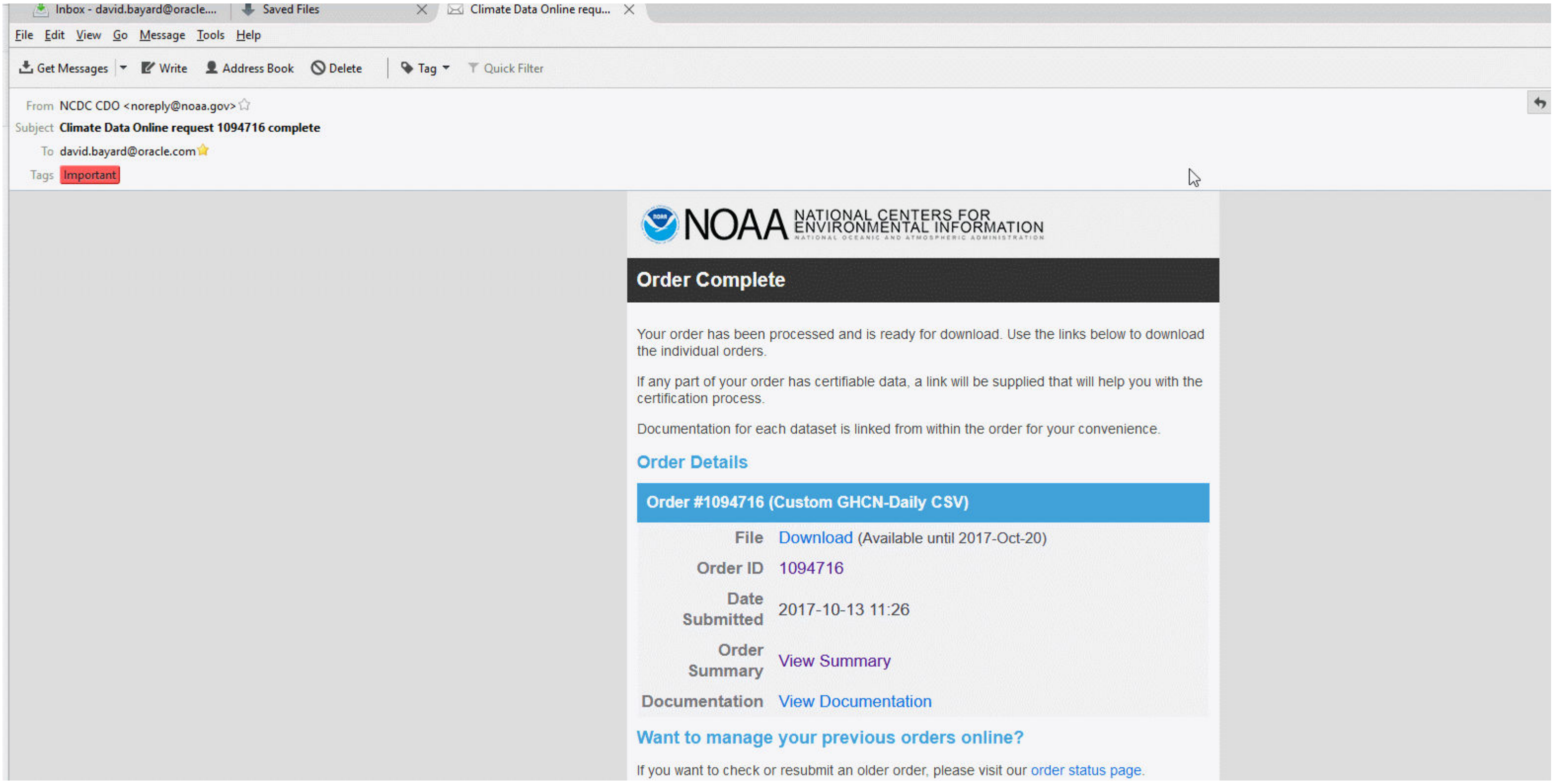
DATA TOOLS
Access past weather and climate data using a collection of specialized tools.
Data Tools »

Good job! Now get yourself a cup a coffee as it may take a few minutes for your order to work through the queue.

Downloading, Uploading the weather data when the request is ready

A few minutes after submiting your free order for weather data, you should receive an email indicating that your Order is complete.

- Click on the Order ID link in the email. This will take you to the Order History page
- Right-click on the download button and choose Copy Link Location
- Paste the value into the weather data URL input field in the paragraph below, then run the paragraph below


Inbox - david.bayard@oracle....
Saved Files
Climate Data Online requ...
File Edit View Go Message Tools Help
Get Messages
Write
Address Book
Delete
Tag
Quick Filter

From NCDC CDO <noreply@noaa.gov>
Subject **Climate Data Online request 1094716 complete**
To david.bayard@oracle.com
Tags Important

NOAA NATIONAL CENTERS FOR ENVIRONMENTAL INFORMATION
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION

Order Complete

Your order has been processed and is ready for download. Use the links below to download the individual orders.

If any part of your order has certifiable data, a link will be supplied that will help you with the certification process.

Documentation for each dataset is linked from within the order for your convenience.

Order Details

Order #1094716 (Custom GHCN-Daily CSV)

File	Download (Available until 2017-Oct-20)
Order ID	1094716
Date Submitted	2017-10-13 11:26
Order Summary	View Summary
Documentation	View Documentation

Want to manage your orders online?

If you want to check or resubmit an older order, please visit our order status page.

If you are in a hurry... (and don't want to order the government weather data yourself)

Copy this value and paste it into the Weather_Data_URL input field below and run the paragraph.

```
https://raw.githubusercontent.com/millerhoo/journey2-new-data-lake/master/workshops/journey2-new-data-lake/files/1090166.csv
```

Script to download weather data and upload to Object Store

Weather_Data_URL

https://raw.githubusercontent.com/millerhoo/journey2-new-data-lake/master/workshops/journey2-new-data-lake/files/1090166.csv

```
Downloading 201612-weather.csv.  This may take a minute.
rm: cannot remove `201612-weather.csv': No such file or directory
--2017-11-15 20:08:13--  https://raw.githubusercontent.com/millerhoo/journey2-new-data-lake/master/workshops/journey2-new-data-lake/files/1090166.csv
Resolving raw.githubusercontent.com... 151.101.40.133
Connecting to raw.githubusercontent.com|151.101.40.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 259900 (254K) [text/plain]
Saving to: “201612-weather.csv”
     0K .......... .......... .......... .......... .......... 19%  155K 1s
```

```
    50K .......... .......... .......... .......... .......... 39%  438K 1s
   100K .......... .......... .......... .......... .......... 59% 1012K 0s
   150K .......... .......... .......... .......... .......... 78% 1.34M 0s
   200K .......... .......... .......... .......... .......... 98%  726K 0s
   250K ...                                                   100% 7264G=0.6s
2017-11-15 20:08:15 (428 KB/s) - “201612-weather.csv” saved [259900/259900]
.
.
.
-rw-rw-r-- 1 zeppelin zeppelin 259900 Nov 15 20:08 201612-weather.csv
.
.
.
Storing file to Object Storage.  This may take a few minutes.
List the directory. directory should be empty or missing
ls: `swift://journeyC.default/weather': No such file or directory
Make the raw directory in Object Store
Copy First File to Object Store. May take a minute
Validate by listing the csv file that got copied to Object Store
Found 1 items
-rw-rw-rw-   1     259900 2017-11-15 20:08 swift://journeyC.default/weather/raw/201612-weather.csv
.
.
Quick glance at first few lines of weather file...
"STATION","NAME","LATITUDE","LONGITUDE","ELEVATION","DATE","AWND","PRCP","SNOW","SNWD","TAVG","TMAX","TMIN"
"US1NYWC0003","WHITE PLAINS 3.1 NNW, NY US","41.0639","-73.7722","71.0","2016-12-01",,"1.14",,,,,
"US1NYWC0003","WHITE PLAINS 3.1 NNW, NY US","41.0639","-73.7722","71.0","2016-12-02",,"0.00","0.0",,,,
"US1NYWC0003","WHITE PLAINS 3.1 NNW, NY US","41.0639","-73.7722","71.0","2016-12-03",,"0.00","0.0",,,,
"US1NYWC0003","WHITE PLAINS 3.1 NNW, NY US","41.0639","-73.7722","71.0","2016-12-04",,"0.00","0.0",,,,
"US1NYWC0003","WHITE PLAINS 3.1 NNW, NY US","41.0639","-73.7722","71.0","2016-12-05",,"0.12",,,,,
"US1NYWC0003","WHITE PLAINS 3.1 NNW, NY US","41.0639","-73.7722","71.0","2016-12-06",,"0.00","0.0",,,,
"US1NYWC0003","WHITE PLAINS 3.1 NNW, NY US","41.0639","-73.7722","71.0","2016-12-07",,"0.32",,,,,
"US1NYWC0003","WHITE PLAINS 3.1 NNW, NY US","41.0639","-73.7722","71.0","2016-12-08",,"0.00","0.0",,,,
"US1NYWC0003","WHITE PLAINS 3.1 NNW, NY US","41.0639","-73.7722","71.0","2016-12-09",,"0.00","0.0",,,,
.
.
done
```

Reading the weather data and registering as a Spark SQL table

The next step is to use Spark to read our weather data CSV file that we uploaded to the Object Store. Once we read the CSV into a Spark Data Frame, we will register the data frame as a Spark SQL temp table.

You can review the Spark SQL programming guide for a refresher about Data Frames and Temporary Tables: Spark SQL Programming Guide (https://spark.apache.org/docs/2.1.0/sql-programming-guide.html)

Spark Scala to read CSV and register as a temp view

```
%spark


val Container = "journeyC"
val Directory = "weather"

sqlContext.setConf("spark.sql.shuffle.partitions", "4")

//val df = sqlContext.read.format("com.databricks.spark.csv").option("header", "true").load("swift://"+Container+".default/"+Directory+"/raw/201612-weather.csv")
//We will use the bdfs (alluxio) cached file system to access our object store data...
val wdf = sqlContext.read.format("com.databricks.spark.csv").option("header", "true").option("inferSchema","true").load("bdfs://localhost:19998/"+Directory+"/raw/201612-weather.csv")

// If you get this error message:
// java.lang.IllegalStateException: Cannot call methods on a stopped SparkContext.
// Then go to the Settings tab, then click on Notebook.  Then restart the Notebook.  This will restart your SparkContext


println("Here is the schema detected from the CSV")
wdf.printSchema()
println("..")

println("# of rows: %s".format(
  wdf.count()
))
println("..")

wdf.createOrReplaceTempView("weather_temp")
println("done")
```

```
Container: String = journeyC
Directory: String = weather
wdf: org.apache.spark.sql.DataFrame = [STATION: string, NAME: string ... 11 more fields]
Here is the schema detected from the CSV
root
 |-- STATION: string (nullable = true)
 |-- NAME: string (nullable = true)
 |-- LATITUDE: double (nullable = true)
 |-- LONGITUDE: double (nullable = true)
 |-- ELEVATION: double (nullable = true)
 |-- DATE: timestamp (nullable = true)
 |-- AWND: double (nullable = true)
 |-- PRCP: double (nullable = true)
 |-- SNOW: double (nullable = true)
 |-- SNWD: integer (nullable = true)
 |-- TAVG: integer (nullable = true)
 |-- TMAX: integer (nullable = true)
 |-- TMIN: integer (nullable = true)
..
# of rows: 2576
..
done
```

SQL to experiment with our Weather data

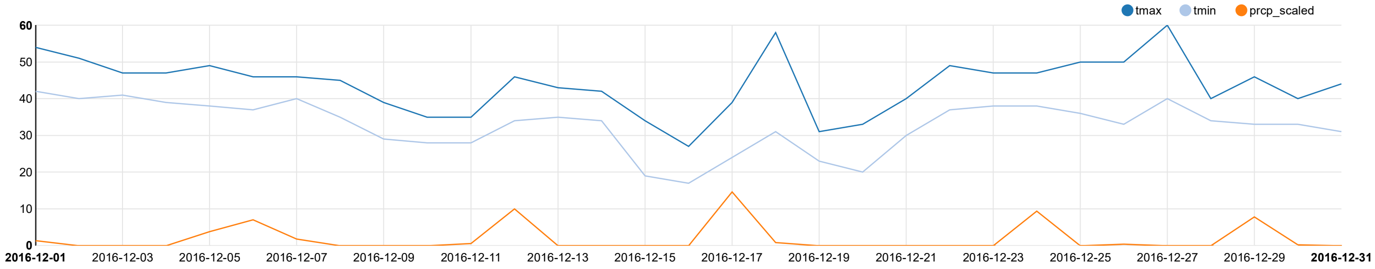
```
%sql
select * from weather_temp
where NAME like '%CENTRAL%'
```

STATION ▾	NAME ▾	LATITUDE ▾	LONGITUDE ▾	ELEVATION ▾	DATE ▾	AWND ▾	PRCP ▾	SNOW ▾	SNWD ▾	TAVG ▾	TMAX ▾	TMIN ▾
USW00094728	NY CITY CENTRAL PARK, NY US	40.77898	-73.96925	42.7	2016-12-01 00:00:00.0	7.61	0.07	0.0	0	null	54	42
USW00094728	NY CITY CENTRAL PARK, NY US	40.77898	-73.96925	42.7	2016-12-02 00:00:00.0	8.72	0.0	0.0	0	null	51	40
USW00094728	NY CITY CENTRAL PARK, NY US	40.77898	-73.96925	42.7	2016-12-03 00:00:00.0	8.72	0.0	0.0	0	null	47	41
USW00094728	NY CITY CENTRAL PARK, NY US	40.77898	-73.96925	42.7	2016-12-04 00:00:00.0	4.92	0.0	0.0	0	null	47	39
USW00094728	NY CITY CENTRAL PARK, NY US	40.77898	-73.96925	42.7	2016-12-05 00:00:00.0	5.59	0.19	0.0	0	null	49	38
USW00094728	NY CITY CENTRAL PARK, NY US	40.77898	-73.96925	42.7	2016-12-06 00:00:00.0	5.37	0.35	0.0	0	null	46	37
USW00094728	NY CITY CENTRAL PARK, NY US	40.77898	-73.96925	42.7	2016-12-07 00:00:00.0	3.8	0.09	0.0	0	null	46	40
USW00094728	NY CITY CENTRAL PARK, NY US	40.77898	-73.96925	42.7	2016-12-08 00:00:00.0	null	0.0	0.0	0	null	45	35
USW00094728	NY CITY CENTRAL PARK, NY US	40.77898	-73.96925	42.7	2016-12-09 00:00:00.0	null	0.0	0.0	0	null	39	29

More SQL to experiment with our Weather data

```
%sql
select cast(date as date) day_of_month, tmax, tmin, prcp*20 prcp_scaled
from weather_temp
where station='USW00094728'
order by day_of_month
```

settings ▾



Creating a calendar dataset

In addition to weather, let’s add some calendar data such as holidays and attributes about workdays and weekends. Since there are not many holidays, it seems like overkill to create a new file and upload it just to keep track. Instead, we’ll programatically create a new holiday dataset in the next paragraph.

Create a temp view for Holidays

```
%spark
```

```
hdf: org.apache.spark.sql.Dataset[String] = [value: string]
+----------+
|     value|
+----------+
|2016-12-24|
|2016-12-25|
|2016-12-26|
|2016-12-31|
+----------+
..
# of rows: 4
..
done
```

Basic SQL to experiment with joining Weather and Calendar

```
%sql
select cast(date as date) day_of_month, tmax, tmin, h.value ,
   date_format(date,"E") day_of_week,
   IF(isnull(h.value), 'N', 'Holiday') holiday,
   IF(isnull(h.value) and date_format(date,"E") in ("Mon",'Tue','Wed','Thu','Fri'), 'Workday', 'N') workday
from weather_temp w LEFT OUTER JOIN holidays_temp h
ON (w.date = h.value)
where station='USW00094728'
order by day_of_month
```

day_of_month ▾	tmax ▾	tmin ▾	value ▾	day_of_week ▾	holiday ▾	workday ▾
2016-12-01	54	42	null	Thu	N	Workday
2016-12-02	51	40	null	Fri	N	Workday
2016-12-03	47	41	null	Sat	N	N
2016-12-04	47	39	null	Sun	N	N
2016-12-05	49	38	null	Mon	N	Workday
2016-12-06	46	37	null	Tue	N	Workday
2016-12-07	46	40	null	Wed	N	Workday
2016-12-08	45	35	null	Thu	N	Workday
2016-12-09	39	29	null	Fri	N	Workday

SQL to experiment with joining Weather and Calendar and Bike Trips

```
%sql
/* if bike_trips_temp is not found, then you need to go back and run Tutorial 4 Working with Spark Interpreter. */
SELECT  `Start Station Name` STARTSTATIONNAME,
`Start Station ID` STARTSTATIONID,
 cast(`Start Station Latitude` as string) STARTSTATIONLATITUDE,
 cast(`Start Station Longitude` as string) STARTSTATIONLONGITUDE,
`End Station ID` ENDSTATIONID,
`End Station Name` ENDSTATIONNAME,
 cast(`End Station Latitude` as string) ENDSTATIONLATITUDE,
 cast(`End Station Longitude` as string) ENDSTATIONLONGITUDE,
`Start Time` STARTTIME,
`Stop Time` STOPTIME,
 cast(`Start Time` as date) STARTDATE,
 cast(date_format(`Start Time`,"H") as integer) STARTHOUR,
 date_format(date,"E") STARTDAY_OF_WEEK,
 IF(isnull(h.value), 'Not_Holiday', 'Holiday') HOLIDAY,
 IF(isnull(h.value) and date_format(date,"E") in ("Mon",'Tue','Wed','Thu','Fri'), 'Workday', 'Not_Workday') WORKDAY,
`Trip Duration` TRIPDURATION,
`Bike ID` BIKEID,
`User Type` USERTYPE
```

STARTSTATIONNAME	STARTSTATIONID	STARTSTATIONLATITUDE	STARTSTATIONLONGITUDE	ENDSTATIONID	ENDSTATIONNAME	ENDSTATIONLATITUDE	ENDSTATIONLONG
Broadway & W 60 St	499	40.76915505	-73.98191841	228	E 48 St & 3 Ave	40.7546011026	-73.971878855
Plaza St West & Flatbush Ave	3418	40.6750207	-73.97111473	3358	Garfield Pl & 8 Ave	40.6711978	-73.97484126
E 15 St & 3 Ave	297	40.734232	-73.986923	345	W 13 St & 6 Ave	40.73649403	-73.99704374
Washington St & Gansevoort St	405	40.739323	-74.008119	358	Christopher St & Greenwich St	40.73291553	-74.00711384
Peck Slip & Front St	279	40.707873	-74.00167	279	Peck Slip & Front St	40.707873	-74.00167
Myrtle Ave & St Edwards St	245	40.69327018	-73.97703874	372	Franklin Ave & Myrtle Ave	40.694528	-73.958089
W 20 St & 8 Ave	470	40.74345335	-74.00004031	453	W 22 St & 8 Ave	40.74475148	-73.99915362
1 Ave & E 94 St	3312	40.7817212	-73.94594	3325	E 95 St & 3 Ave	40.7849032	-73.950503

Create a permanent table of bike trips joined with weather and calendar (will not have output)

```
%sql
create table bike_trips_weather_parquet
stored as parquet
SELECT  `Start Station Name` STARTSTATIONNAME,
`Start Station ID` STARTSTATIONID,
 cast(`Start Station Latitude` as string) STARTSTATIONLATITUDE,
 cast(`Start Station Longitude` as string) STARTSTATIONLONGITUDE,
`End Station ID` ENDSTATIONID,
`End Station Name` ENDSTATIONNAME,
 cast(`End Station Latitude` as string) ENDSTATIONLATITUDE,
 cast(`End Station Longitude` as string) ENDSTATIONLONGITUDE,
`Start Time` STARTTIME,
`Stop Time` STOPTIME,
 cast(`Start Time` as date) STARTDATE,
 cast(date_format(`Start Time`,"H") as integer) STARTHOUR,
 date_format(date,"E") STARTDAY_OF_WEEK,
 IF(isnull(h.value), 'Not_Holiday', 'Holiday') HOLIDAY,
 IF(isnull(h.value) and date_format(date,"E") in ("Mon",'Tue','Wed','Thu','Fri'), 'Workday', 'Not_Workday') WORKDAY,
`Trip Duration` TRIPDURATION,
`Bike ID` BIKEID,
`User Type` USERTYPE,
`Birth Year` BIRTHYEAR,
 2017-`Birth Year` RIDERAGE,
`Gender` GENDER_CODE,
 case when gender=1 then 'Male' when gender=2 then 'Female' else 'unknown' end GENDER ,
 w.AWND AVERAGEWIND,
 w.PRCP PRECIPITATION,
 w.SNOW SNOW,
 w.SNWD SNOW_ON_GROUND,
 w.TMAX MAXTEMPERATURE,
 w.TMIN MINTEMPERATURE
FROM weather_temp w
  LEFT OUTER JOIN holidays_temp h
    ON (w.date = h.value),
  bike_trips_temp t
WHERE to_date(w.date) = to_date(t.`Start Time`)
AND w.station='USW00094728'
```

Querying the combined bike trip, weather, and calendar information

Now we can show some examples of querying our combined table.

Trips by type of day and weather

```
%sql
select cast(starttime as date) day, workday, precipitation, count(*)
from bike_trips_weather_parquet
group by cast(starttime as date), workday, precipitation
order by day
```

day	workday	precipitation	count(1)
2016-12-01	Workday	0.07	43242
2016-12-02	Workday	0.0	42371
2016-12-03	Not_Workday	0.0	29854
2016-12-04	Not_Workday	0.0	27455
2016-12-05	Workday	0.19	35932
2016-12-06	Workday	0.35	33332
2016-12-07	Workday	0.09	38021
2016-12-08	Workday	0.0	39626
2016-12-09	Workday	0.0	33547

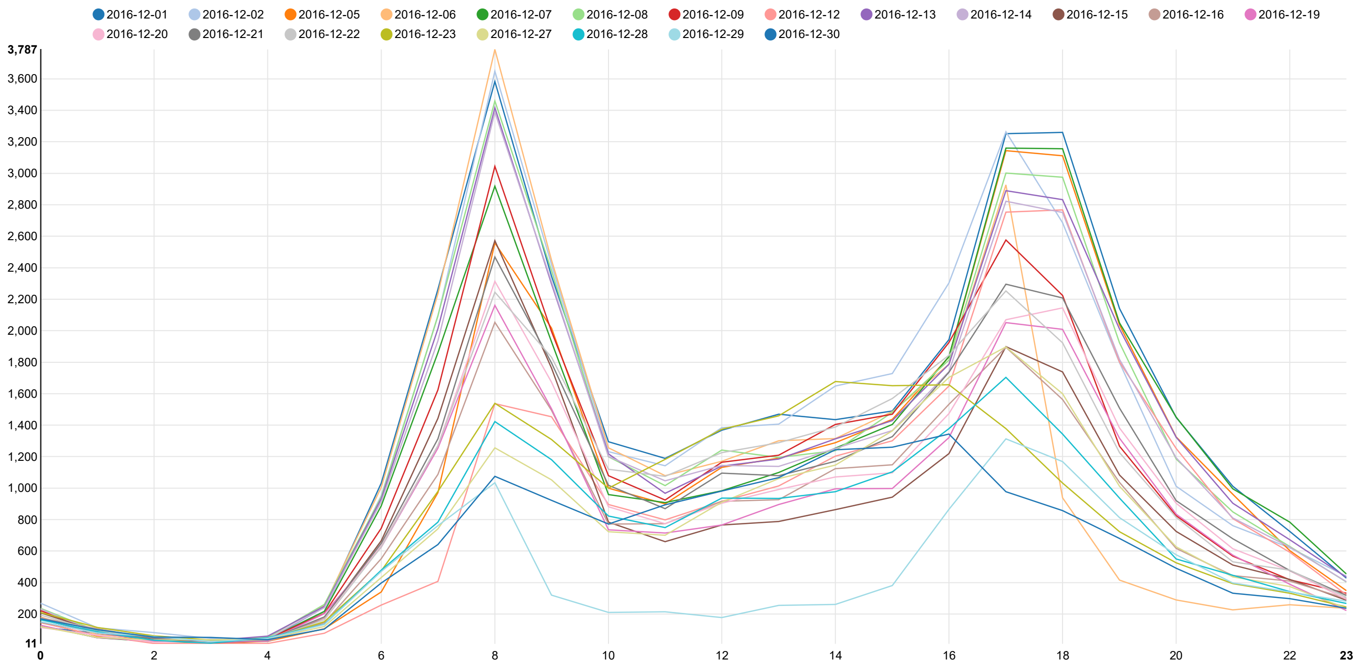
Workday Bike Trips by Hour

```
%sql
select startdate,  starthour, count(*)
from (select starthour,
 startday_of_week,
 startdate
from bike_trips_weather_parquet
where workday="Workday"
and (gender="${gender=Male,Male|Female|unknown}" )) bike_times
group by startdate,  starthour
```

gender

Male

settings ▾



Next Steps

So far in the journey, we have downloaded Citi Bike data and stored it into the Object Store. Then, we configured Spark to be able to work with CSV files. Then, we read in the data and defined a Spark SQL temporary table with it.

In this tutorial, we added some additional datasets. We showed how to download daily weather data and use it as a Spark table. And we showed how to create a simple holiday table. Then we combined the bike trips, weather, and holiday data into a single table and saved it as a permanent hive table called bike_trips_weather_parquet.

To continue, proceed to the next tutorial where we will show you how you can connect Oracle Data Visualization Desktop to BDCS-CE and begin to visualize our combined bike_trips_weather_parquet dataset.

Change Log

October 13, 2017 - Created and tested with 17.3.5.

```
%md
```