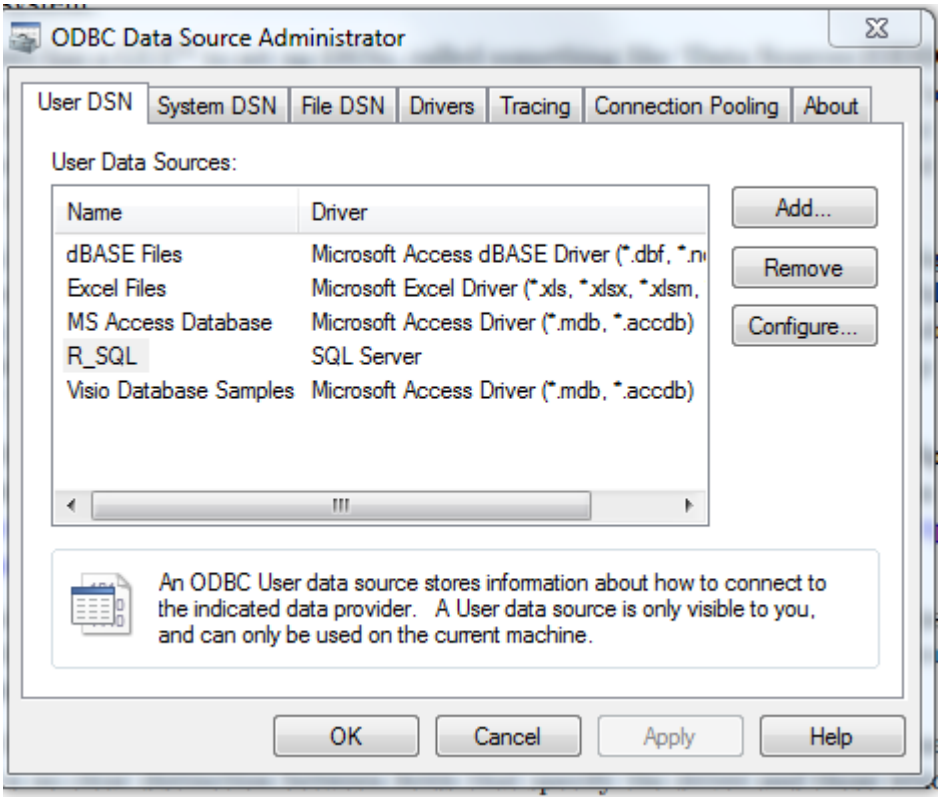
BRIEF INTRODUCTION to INSTALLING R STUDIO and CREATING A DATA FRAME in R from a SQL SERVER TABLE

The 1st step in using the R integrated package for ODBC database connectivity, which provides a common API access for DBMSs such as MySQL, PostgresSQL, Microsoft Access, SQL Server, DB2, Oracle and SQLite, is to establish the connection to the Data Source Name (DSN) for Windows Users. Navigate to **Control Panel\System and Security\Administrative Tools**. Select **Data Sources (ODBC)**.

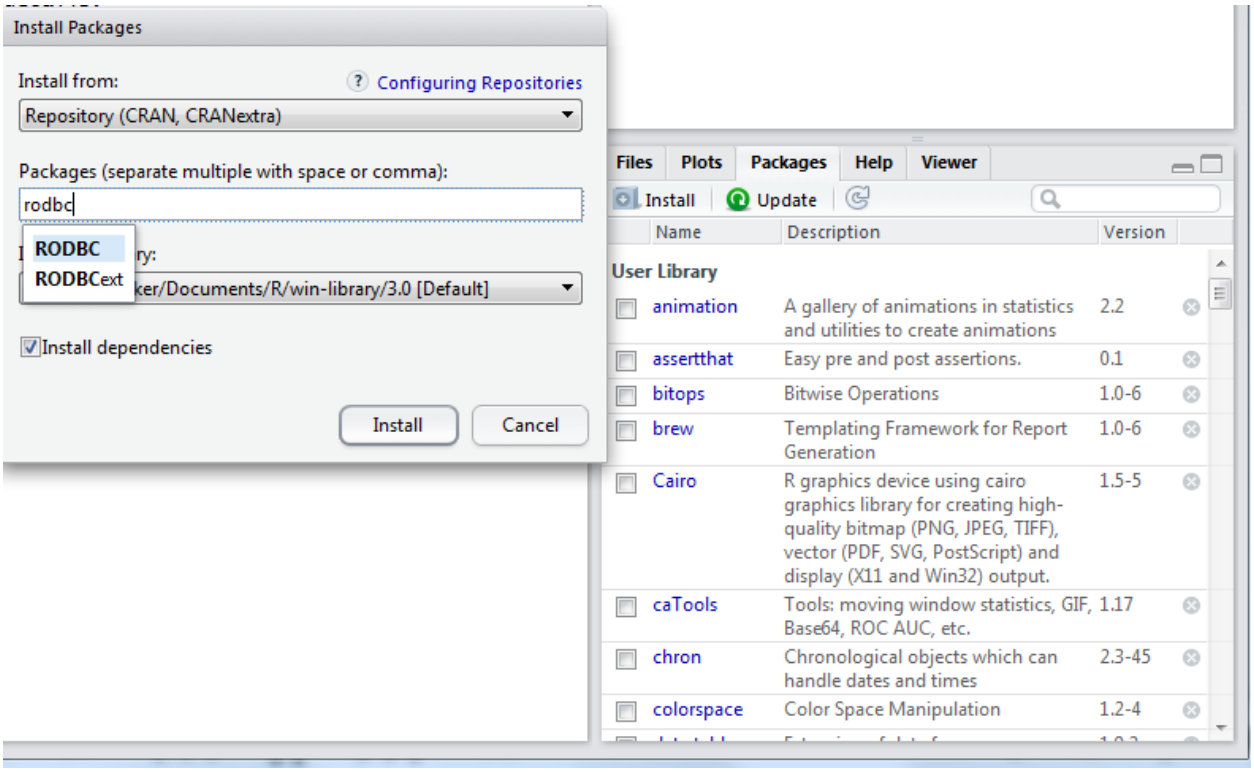


You need to add a new 'User Data Source'; mine is named "R_SQL". When you click Add, you will see an option to select a driver for the relevant data source (SQL Server). You will want to go through the setup and ensure you give it a name you can use going forward; and then select to connect to the appropriate SQL Server instance. When you see, 'How should SQL Server verify the authenticity of the login ID?', select 'With Windows NT authentication using the network login ID'. After walking through the steps you should be able to 'Test Data Source' and verify that the ODBC Microsoft SQL Server Setup is successful.

Next you will want to download R, itself. It is open source and I recommend R Studio (a slight GUI-friendly version rather plain prompt-based R), which is a slightly more visual dashboard based version of R. An installer for your platform (e.g., Windows 7 64bit) should correctly install RStudio. Here is the link…http://www.rstudio.com/products/rstudio/download/

A brief introduction to R can be found here… http://www.ats.ucla.edu/stat/r/seminars/intro.htm

Once you've opened R Studio, you can navigate to the 'Packages' tab and click 'Install' to install packages (over 4,000 publicly available open source); typing "RODBC" into the packages prompt will locate the ODBC package for selection. This package will allow use of SQL Server connectivity.



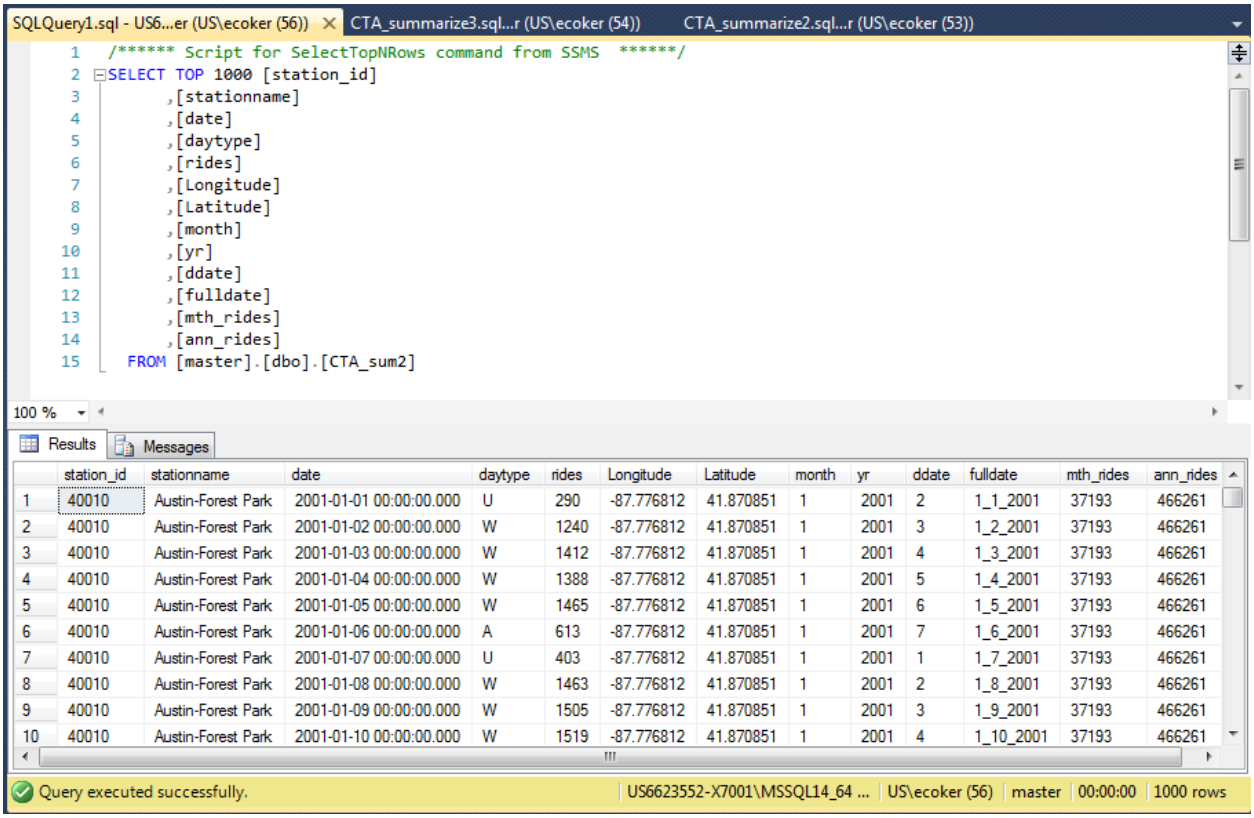
After installing, you will see the following confirmation in the R Console. Then type into the console, "library(RODBC)" in order to activate the ODBC package for R.

```
> install.packages("RODBC")
Installing package into 'C:/Users/ecoker/Documents/R/win-library/3.0'
(as 'lib' is unspecified)
trying URL 'http://cran.revolutionanalytics.com/bin/windows/contrib/3.0/RODBC_1.3-10.zip'
Content type 'application/zip' length 829130 bytes (809 Kb)
opened URL
downloaded 809 Kb

package 'RODBC' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
 C:\Users\ecoker\AppData\Local\Temp\RtmpoVebkD\downloaded_packages
> library(RODBC)
>
```

Then using R, I can pull data from a SQL Server table into a data frame of similar format and structure, within R. Using R, advanced data mining, statistical modeling, data cleaning or formatting, and visualization can be performed; which is not possible within SQL Server. Please note the SQL Server script.



SQLQuery1.sql - US6...er (US\ecoker (56)) | CTA_summarize3.sql...r (US\ecoker (54)) | CTA_summarize2.sql...r (US\ecoker (53))

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [station_id]
      ,[stationname]
      ,[date]
      ,[daytype]
      ,[rides]
      ,[Longitude]
      ,[Latitude]
      ,[month]
      ,[yr]
      ,[ddate]
      ,[fulldate]
      ,[mth_rides]
      ,[ann_rides]
  FROM [master].[dbo].[CTA_sum2]
```

100 %

Results | Messages

	station_id	stationname	date	daytype	rides	Longitude	Latitude	month	yr	ddate	fulldate	mth_rides	ann_rides
1	40010	Austin-Forest Park	2001-01-01 00:00:00.000	U	290	-87.776812	41.870851	1	2001	2	1_1_2001	37193	466261
2	40010	Austin-Forest Park	2001-01-02 00:00:00.000	W	1240	-87.776812	41.870851	1	2001	3	1_2_2001	37193	466261
3	40010	Austin-Forest Park	2001-01-03 00:00:00.000	W	1412	-87.776812	41.870851	1	2001	4	1_3_2001	37193	466261
4	40010	Austin-Forest Park	2001-01-04 00:00:00.000	W	1388	-87.776812	41.870851	1	2001	5	1_4_2001	37193	466261
5	40010	Austin-Forest Park	2001-01-05 00:00:00.000	W	1465	-87.776812	41.870851	1	2001	6	1_5_2001	37193	466261
6	40010	Austin-Forest Park	2001-01-06 00:00:00.000	A	613	-87.776812	41.870851	1	2001	7	1_6_2001	37193	466261
7	40010	Austin-Forest Park	2001-01-07 00:00:00.000	U	403	-87.776812	41.870851	1	2001	1	1_7_2001	37193	466261
8	40010	Austin-Forest Park	2001-01-08 00:00:00.000	W	1463	-87.776812	41.870851	1	2001	2	1_8_2001	37193	466261
9	40010	Austin-Forest Park	2001-01-09 00:00:00.000	W	1505	-87.776812	41.870851	1	2001	3	1_9_2001	37193	466261
10	40010	Austin-Forest Park	2001-01-10 00:00:00.000	W	1519	-87.776812	41.870851	1	2001	4	1_10_2001	37193	466261

Query executed successfully. | US6623552-X7001\MSSQL14_64 ... | US\ecoker (56) | master | 00:00:00 | 1000 rows

Then, running the following code from within the R script window in RStudio will create a table using SQL, which is actually referred to as a data frame from R. The last line of script code, “head(CTA)”, will display a header preview of the new data frame in R; which looks similar to how it did in SQL Server.

```
odbcChannel <- odbcConnect("R_SQL")

CTA <- as.data.frame(sqlQuery(odbcChannel, "SELECT
       [station_id]
      ,[stationname]
      ,[rides]
      ,[Longitude]
      ,[Latitude]
      ,[month]
      ,[yr]
      ,[ddate]
      ,[fulldate]
      ,[mth_rides]
      ,[ann_rides]
FROM [master].[dbo].[CTA_sum2]"))
head(CTA)

19:1  (Top Level)                                   R Script
```

Note the console below, where results (of which include the preview of the newly named ‘CTA’ data frame from the SQL Server table) of an R script in RStudio will appear.

```
Console ~/
> odbcChannel <- odbcConnect("R_SQL")
>
> CTA <- as.data.frame(sqlQuery(odbcChannel, "SELECT
+        [station_id]
+       ,[stationname]
+       ,[rides]
+       ,[Longitude]
+       ,[Latitude]
+       ,[month]
+       ,[yr]
+       ,[ddate]
+       ,[fulldate]
+       ,[mth_rides]
+       ,[ann_rides]
+ FROM [master].[dbo].[CTA_sum2]"))
> head(CTA)
  station_id        stationname    rides Longitude Latitude month       yr ddate
1  40,010.00 Austin-Forest Park   290.00    -87.78    41.87     1 2,001.00     2
2  40,010.00 Austin-Forest Park 1,240.00    -87.78    41.87     1 2,001.00     3
3  40,010.00 Austin-Forest Park 1,412.00    -87.78    41.87     1 2,001.00     4
4  40,010.00 Austin-Forest Park 1,388.00    -87.78    41.87     1 2,001.00     5
5  40,010.00 Austin-Forest Park 1,465.00    -87.78    41.87     1 2,001.00     6
6  40,010.00 Austin-Forest Park   613.00    -87.78    41.87     1 2,001.00     7
    fulldate mth_rides  ann_rides
1 1_1_2001   37,193.00 466,261.00
2 1_2_2001   37,193.00 466,261.00
3 1_3_2001   37,193.00 466,261.00
4 1_4_2001   37,193.00 466,261.00
5 1_5_2001   37,193.00 466,261.00
6 1_6_2001   37,193.00 466,261.00
>
```

**Iterative commands can be placed from the R console window, in addition to the script window; much like interactive programming in Python for those familiar.

Now, there are an extremely large (4,000) number of packages that can be installed into RStudio in order to perform diagnostic, descriptive, or predictive analytics of our new data frame. But importantly, you have moved your data from SQL Server to R, and in the future can reformat this R script as you see fit, if performing different actions in SQL Server.