Connecting R-Studio to Azure SQL Database

Jedi Jeremy O
SMU – Doing Data Science

High Level Steps

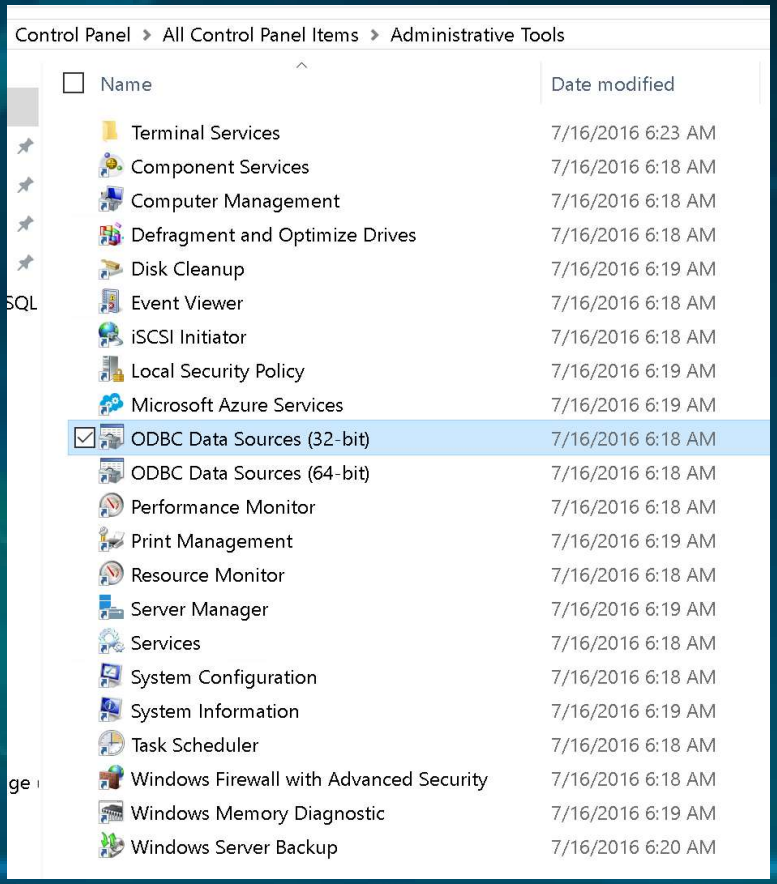
NOTE: Assuming you have a SQL DB in Azure. This process can also be adopted for a SQL DB on premise.

- ✓ Verify ODBC connector version in Windows 10
- ✓ Configure ODBC connector
- ✓ In R-Studio install RODBC package
- ✓ Configure connection + credentials
- ✓ Import data as data frame

Find the ODBC Connector

NOTE: This is for Windows OS. Mac users see this URL:
https://docs.microsoft.com/en-us/sql/connect/odbc/linux-mac/installing-the-microsoft-odbc-driver-for-sql-server?view=sql-server-2017

1. In the search icon type **Control Panel**
2. Within the Control Panel locate & select **Administrative Tools**
3. Within Administrative Tools click on **ODBC Data Sources 32-bit**

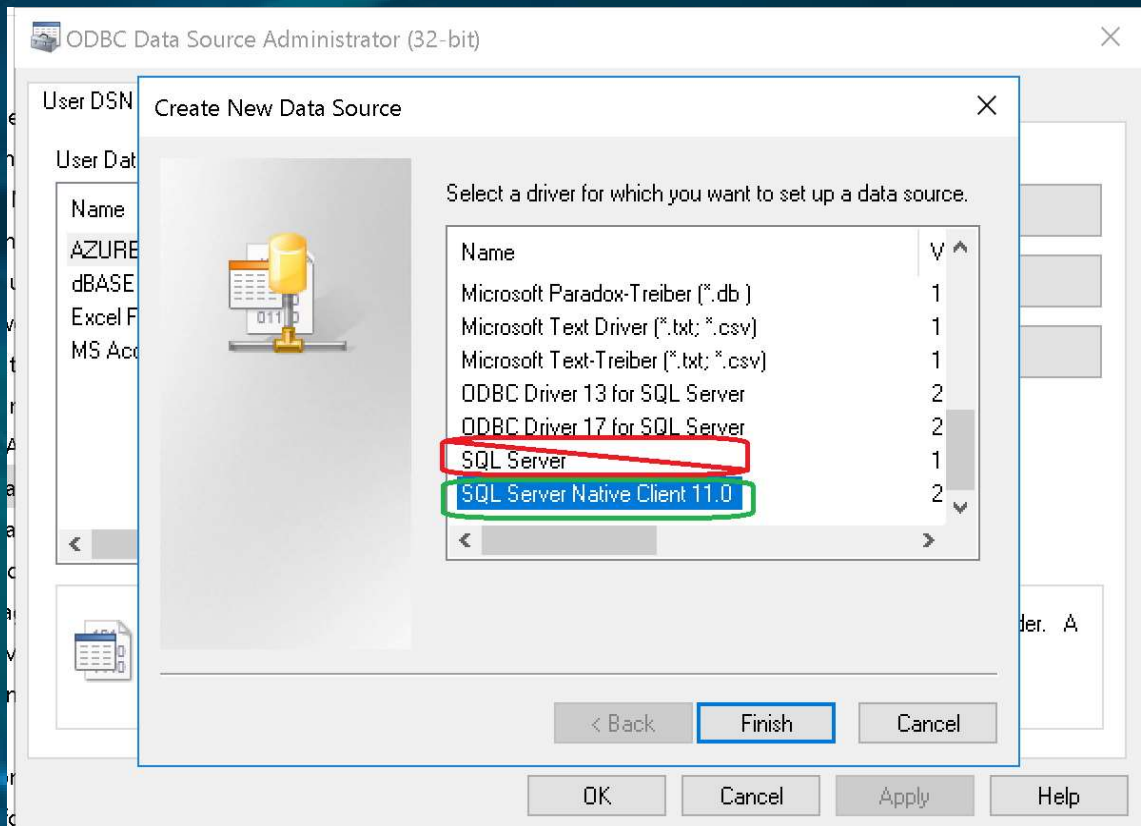


Verify Correct ODBC Connector Type

Download the updated ODBC connector:
https://docs.microsoft.com/en-us/sql/connect/odbc/microsoft-odbc-driver-for-sql-server

1. In the **User DSN** tab click **Add**
2. In the new window that opens look for an option that says **SQL Server Native Client**.

NOTE: the option that says only **SQL Server** is **NOT** the correct option. If you do not see **SQL Server Native Client** you need to download the correct ODBC connector



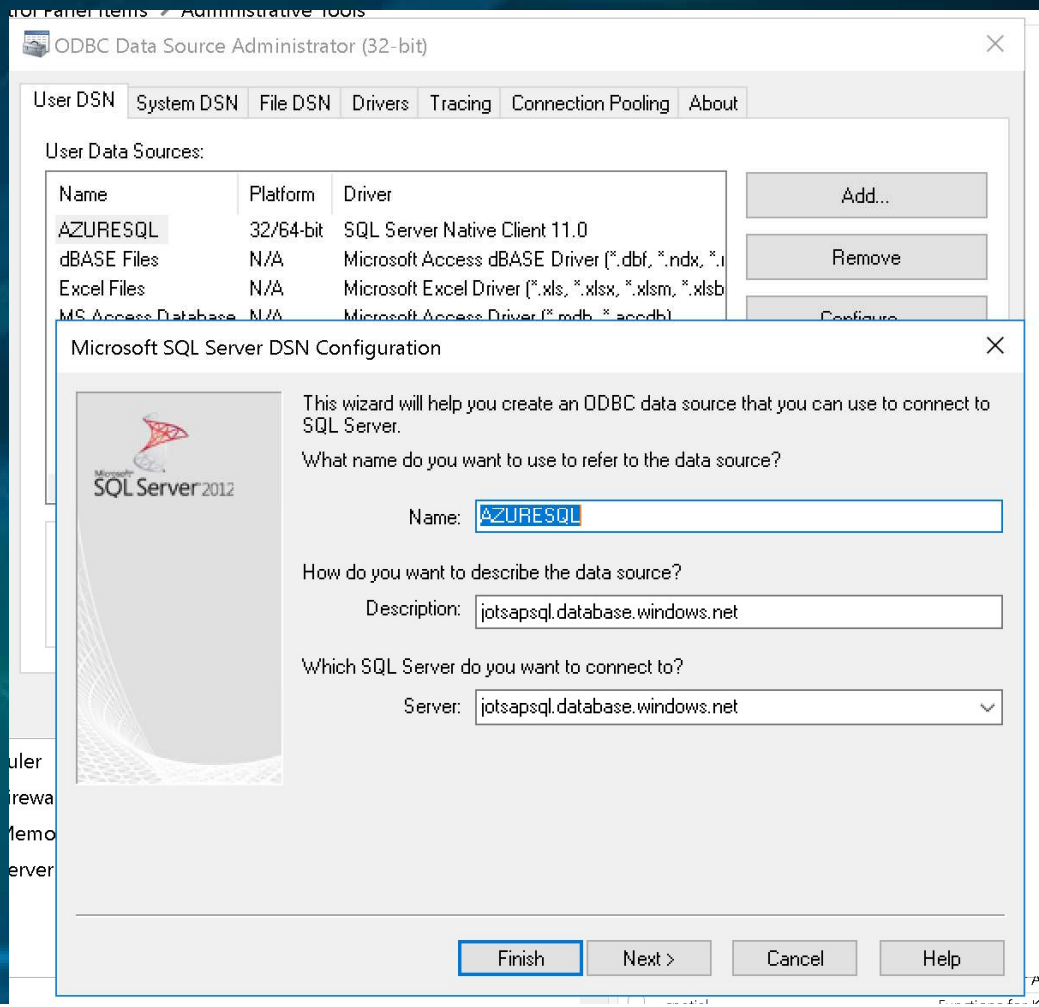
Configure ODBC Connection

You will need the following information at a minimum to configure the ODBC connection:

- ✓ Server's DNS name or IP address
- ✓ Authorized account / username
- ✓ Credential / password

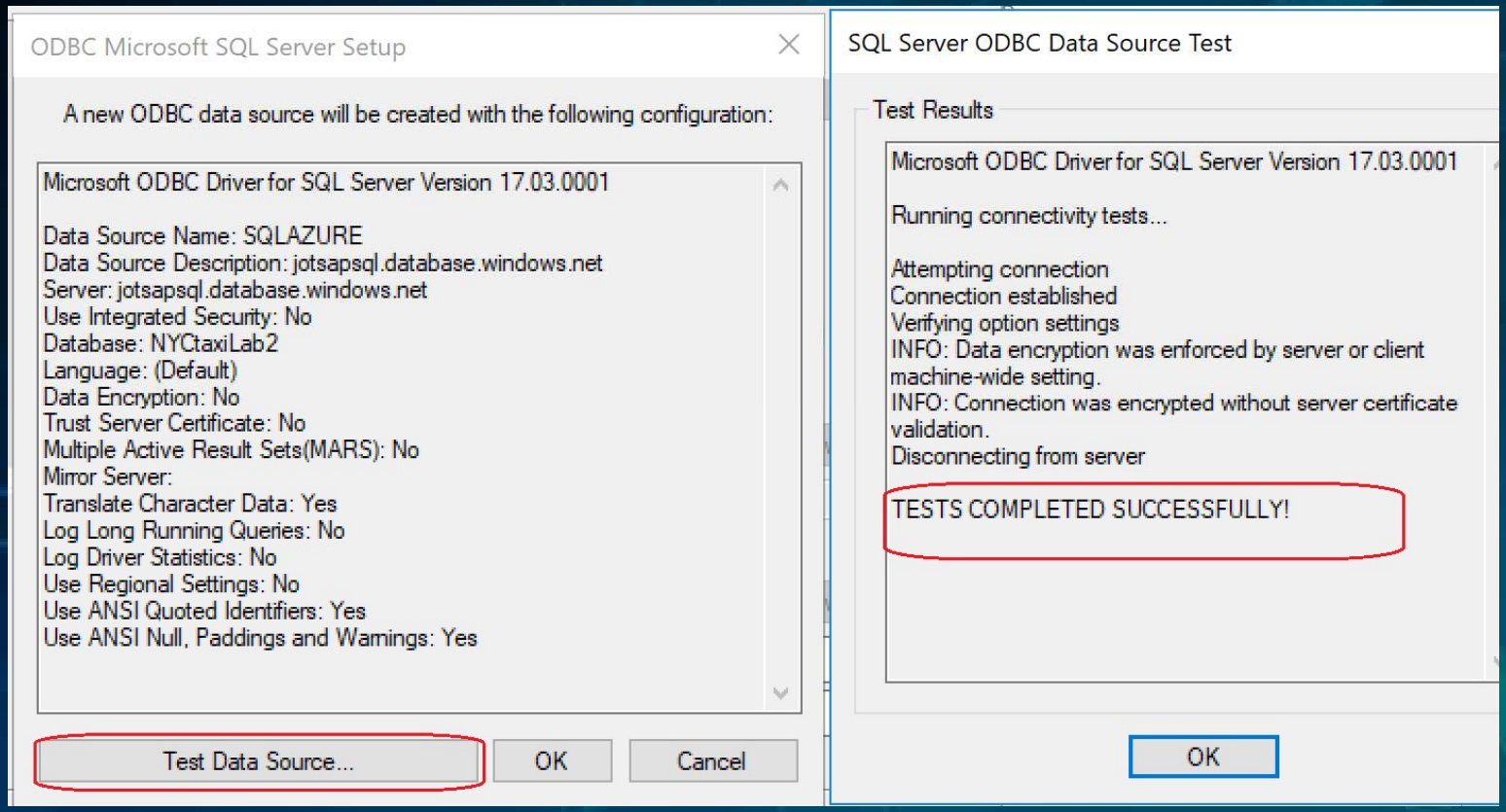
Note the name you give. In this case I named mine **AZURESQL**.

Optionally you may want to set the default database to the one you are interested in.
Ex) Sales, Inventory, Customers, etc



Test ODBC Connection

Validate the ODBC connection



The R Code To Connect To SQL

```
### SQL CONNECTION ###

# load RODBC library
library(RODBC)

# call previously configured ODBC connection "AZURESQL"
# note you will need to configure credentials
# NOT SECURE FOR PRODUCTION
odbcConnect("AZURESQL", uid = "sqladmin", pwd = "*******") -> azureodbc

# Imbeds following SQL statement: select top(10000) from trips_all
# Takes top 10000 rows w/ all columns from "trips_all" data base
# then outputs those results to taxi.df data frame
sqlQuery(azureodbc, "select top(10000) * from dbo.trips_all;" ) -> taxi.df

# verify taxi.df
head(taxi.df)
str(taxi.df)
```

Screenshot of Results

SQL_Connection.R | taxi.df

Filter

	trip_type	trip_year	trip_month	taxi_type	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	trip_distance	rate_code_id	store_and_fwd_flag	pickup_location_id
978	0	2016	1	yellow	1	2016-01-02 03:10:38	2016-01-02 03:17:50	1	1.40	1	N	0
979	0	2016	1	yellow	1	2016-01-02 03:10:51	2016-01-02 03:15:10	1	0.80	1	N	0
980	0	2016	1	yellow	1	2016-01-02 03:11:08	2016-01-02 03:17:48	1	0.90	1	N	0
981	0	2016	1	yellow	1	2016-01-02 03:11:19	2016-01-02 03:20:01	2	3.10	1	N	0
982	0	2016	1	yellow	1	2016-01-02 03:11:29	2016-01-02 03:14:33	1	0.70	1	N	0
983	0	2016	1	yellow	1	2016-01-02 03:11:36	2016-01-02 03:19:18	2	2.70	1	N	0
984	0	2016	1	yellow	1	2016-01-02 03:11:47	2016-01-02 03:15:43	1	0.90	1	N	0
985	0	2016	1	yellow	1	2016-01-02 03:11:57	2016-01-02 03:25:23	2	7.00	1	N	0
986	0	2016	1	yellow	1	2016-01-02 03:12:06	2016-01-02 03:24:02	3	3.10	1	N	0
987	0	2016	1	yellow	1	2016-01-02 03:12:15	2016-01-02 03:21:16	1	2.70	1	N	0
988	0	2016	1	yellow	2	2016-01-02 03:12:26	2016-01-02 03:26:20	3	3.89	1	N	0
989	0	2016	1	yellow	2	2016-01-02 03:12:38	2016-01-02 03:18:44	2	2.05	1	N	0
990	0	2016	1	yellow	2	2016-01-02 03:12:51	2016-01-02 03:15:11	1	0.59	1	N	0
991	0	2016	1	yellow	1	2016-01-02 03:13:00	2016-01-02 03:20:04	1	1.80	1	Y	0
992	0	2016	1	yellow	2	2016-01-02 03:13:14	2016-01-02 03:27:23	4	4.77	1	N	0
993	0	2016	1	yellow	1	2016-01-02 03:13:24	2016-01-02 03:21:11	1	1.90	1	N	0
994	0	2016	1	yellow	1	2016-01-02 03:13:37	2016-01-02 03:19:20	1	0.80	1	N	0
995	0	2016	1	yellow	2	2016-01-02 03:13:51	2016-01-02 03:47:06	1	18.36	2	N	0
996	0	2016	1	yellow	2	2016-01-02 03:14:06	2016-01-02 03:31:54	1	5.28	1	N	0
997	0	2016	1	yellow	1	2016-01-02 03:14:18	2016-01-02 03:34:27	2	11.50	1	N	0
998	0	2016	1	yellow	1	2016-01-02 03:14:27	2016-01-02 03:19:16	1	2.00	1	N	0
999	0	2016	1	yellow	1	2016-01-02 03:14:42	2016-01-02 03:21:53	1	1.10	1	N	0
1000	0	2016	1	yellow	2	2016-01-02 03:14:54	2016-01-02 03:19:35	3	0.89	1	N	0

Looking at Ingested Data

Notice that datetime attribute is automatically categorized

[very convenient for timestamps on logs, sensor data, etc]

```
>
> str(taxi.df)
'data.frame':   1000 obs. of  26 variables:
 $ trip_type            : int  0 0 0 0 0 0 0 0 0 0 ...
 $ trip_year            : int  2016 2016 2016 2016 2016 2016 2016 2016 2016 2016 ...
 $ trip_month           : int  1 1 1 1 1 1 1 1 1 1 ...
 $ taxi_type            : Factor w/ 1 level "yellow": 1 1 1 1 1 1 1 1 1 1 ...
 $ vendor_id            : int  2 1 1 1 1 2 1 1 1 1 ...
 $ pickup_datetime      : POSIXct, format: "2016-01-01 00:00:00" "2016-01-01 00:00:07" "2016-01-01 00:00:20"
 $ dropoff_datetime     : POSIXct, format: "2016-01-01 00:00:00" "2016-01-01 00:09:49" "2016-01-01 00:29:05"
 $ passenger_count      : int  2 1 1 1 1 4 1 1 1 1 ...
 $ trip_distance        : num  1.1 1.8 13 1.2 4.9 1.24 5.1 5.6 10.1 1 ...
 $ rate_code_id         : int  1 1 1 1 1 1 1 1 1 1 ...
 $ store_and_fwd_flag   : Factor w/ 2 levels "N","Y": 1 1 1 1 1 1 1 1 1 1 ...
 $ pickup_location_id   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ dropoff_location_id  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ pickup_longitude     : num  -74 -74 -74 -74 -74 ...
 $ pickup_latitude      : num  40.7 40.7 40.8 40.7 40.7 ...
 $ dropoff_longitude    : num  -74 -74 -73.9 -74 -74 ...
 $ dropoff_latitude     : num  40.7 40.7 40.9 40.8 40.7 ...
 $ payment_type         : int  2 2 2 1 1 2 2 1 2 1 ...
 $ fare_amount          : num  7.5 9 36.5 6.5 19 14.5 17 22.5 33 9 ...
 $ extra                : num  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0 ...
 $ mta_tax              : num  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 ...
 $ tip_amount           : num  0 0 0 2.3 1.7 0 0 4.75 0 1 ...
 $ tolls_amount         : num  0 0 0 0 0 0 0 0 0 0 ...
 $ improvement_surcharge: num  0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 ...
 $ ehail_fee            : num  0 0 0 0 0 0 0 0 0 0 ...
 $ total_amount         : num  8.8 10.3 37.8 10.1 22 ...
>
```