RStudio makes Oracle accessibility from R easier via odbc and connections Pane.

Personally, I find it’s not so easy.

As it finally works for me, I will detail some snippets here.

After tens of try it seems good to share some tricks. This blog post is also a notepad for me. Oracle and R configuration is a step where we potentially waste a lot of time.

Many things can cause oracle and R not to work at all:

it depends on which client is installed (32b, 64b ?) wether odbc driver is correctly installed or not

you have to dissect tnsnames.ora investigate on many ORA error’s maybe try to clean install Oracle client

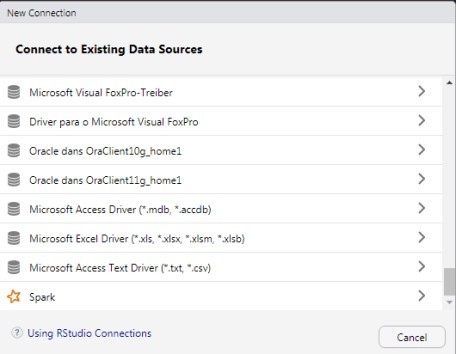
Often ROracle is used and it works well, sometimes it doesn’t (some oci.dll not found3, etc.). But it doesn’t work with dplyr/dbplyr at the moment.

After several years with ROracle, I’m happy to have both possibilities for query writing and collecting (SQL, and now dplyr)

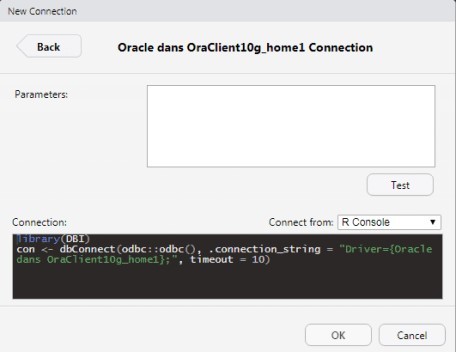
Here we are:

# RStudio connection Pane

From connection Pane we take Oracle odbc driver name, we have two here for two Oracle client versions:



And then:



We now have a big component of the connection string.

# 32b or 64b

If your Oracle client is 32bit, you have to switch to R 32bits, otherwhise it doesn’t work (at least for me).

# Connection string

Then stackoverflow history helped me4 to structure the entire string:

library(odbc) library(dplyr) library(dbplyr) library(lubridate)

my\_oracle <- dbConnect(odbc::odbc(),

.connection\_string = "Driver={Oracle dans OraClient10g\_home1};DBQ=host:port/db\_name;UID=woo;PWD=hoo",

timeout = 10)

You will find all these informations in tnsnames.ora. Port is probably 1521.

# Some dplyr/dbplyr statements

## Simple one

dplyr::tbl(my\_oracle, dbplyr::in\_schema('SCHEMA\_ONE', 'TB\_ONE'))

<SQL> SELECT \*

FROM SCHEMA\_ONE.TB\_ONE

## dplyr and dblink

If you have another oracle database with dblinks it may also works like this:

dplyr::tbl(my\_oracle, dbplyr::in\_schema('SCHEMA\_B', 'TC\_TWO@MYDBTWOLINK'))

<SQL> SELECT \*

FROM SCHEMA\_B.TC\_TWO@MYDBTWOLINK

### List dblinks

DBI::dbGetQuery(my\_oracle, "SELECT \* FROM ALL\_DB\_LINKS")

<SQL> SELECT \*

FROM ALL\_DB\_LINKS

### Catalog of all columns5

<SQL> SELECT \*

FROM ALL\_TAB\_COLUMNS

# Decomposing the connection string

In order to ask for password, we split the connection parts:

library(odbc) library(dplyr) library(dbplyr) library(lubridate)

my\_oracle <- dbConnect(odbc::odbc(),

Driver = "Oracle dans OraClient10g\_home1", DBQ = "host:port/db\_name",

SVC = "DB\_SCHEMA", # schema when connection

opens

UID = "woo", PWD = "hoo")

And then:

library(odbc) library(dplyr) library(dbplyr) library(lubridate)

my\_oracle <- dbConnect(odbc::odbc(),

Driver = "Oracle dans OraClient10g\_home1", DBQ = "host:port/db\_name",

SVC = "DB\_SCHEMA",

UID = rstudioapi::askForPassword('woo

(username)'),

(password)'))

PWD = rstudioapi::askForPassword('Open, Sesame