Connecting databases with R

While csv or excel files are convenient for handling smaller datasets, to store large amounts of data in a structured manner we would require the help of relational data bases.

A database where the values stored are related to each other are termed as **Relational**.

R can connect easily to many such relational databases (MySQL, Oracle, SQL server etc.) and fetch records from them as a data frame. The dataset can be manipulated or analyzed using packages and functions.

We will use the open-source RDBMS called **MySQL** for establishing the connection.

NOTE: If MySQL is not installed in your system, kindly install it from here.

In order to establish a connection, we need a package known as **RMysql** which is a Database Interface and 'MySQL' Driver for R.

The process of establishing connection with MySQL using R:

1. Install and call the package '**RMySQL**'

```
2. install.packages("RMySQL")
3. library(RMySQL)
```

2. Set the settings for the connection.

```
1. db_user <- 'root'
2. db_password <- '<your password>'
3. db_name <- 'sakila'
4. db_host <- '127.0.0.1' # local access
```

NOTE: The default user in your MySQL local instance is root and the database 'sakila' is readily available in MySQL.

3. Establishing Connection

```
1. drv <- dbDriver("MySQL")
2. mydb <-  dbConnect(drv, user = db_user, password = db_password,
3.                    dbname = db_name, host = db_host)
```

Reading Data:

Listing the tables available in the database '**sakila**':

```
1.  dbListTables(mydb)
```

```
> dbListTables(mydb)
 [1] "actor"                      "actor_info"                 "address"
 [4] "category"                   "city"                       "country"
 [7] "customer"                   "customer_list"              "film"
[10] "film_actor"                 "film_category"              "film_list"
[13] "film_text"                  "inventory"                  "language"
[16] "mtcars"                     "nicer_but_slower_film_list" "payment"
[19] "rental"                     "sales_by_film_category"     "sales_by_store"
[22] "staff"                      "staff_list"                 "store"
```

Viewing first 5 values in '**actor**' table:

```
1. result = dbSendQuery(mydb, "select * from actor")
```

The '*' in the query indicates all the values.

The values given should be stored in a R based data frame as seen below.
The fetch() function is for getting the last computed value from the database.

```
1. data.frame = fetch(result, n = 5) #fetching first 5 rows
2. print(data.frame)
```

```
> print(data.frame)
  actor_id first_name    last_name         last_update
1        1   PENELOPE      GUINESS 2006-02-15 04:34:33
2        2       NICK     WAHLBERG 2006-02-15 04:34:33
3        3         ED        CHASE 2006-02-15 04:34:33
4        4   JENNIFER        DAVIS 2006-02-15 04:34:33
5        5     JOHNNY LOLLOBRIGIDA 2006-02-15 04:34:33
```

Clear the result in the data frame object to avoid pending connection with rows error which appear while running new queries.

```
1. dbClearResult(result)
```

Applying Filters using '**where**' clause in the queries:

'where' is a conditional clause which helps in filtering the data.

```
1. result = dbSendQuery(mydb, "select * from film_actor where actor_id =
   '1'")
2.  data.frame = fetch(result, n = -1)
3. print(data.frame)
4. dbClearResult(result)
```

```
> print(data.frame)
   actor_id film_id         last_update
1         1       1 2006-02-15 05:05:03
2         1      23 2006-02-15 05:05:03
3         1      25 2006-02-15 05:05:03
4         1     106 2006-02-15 05:05:03
5         1     140 2006-02-15 05:05:03
6         1     166 2006-02-15 05:05:03
```

Manipulating data:

Creating new tables:

Before performing data manipulation, run the code below to be able load local R data in MySQL.

```
1. dbSendQuery(mydb, "SET GLOBAL local_infile = true;")
```

Loading the local R data frame "mtcars" into MySQL environment as table using the function dbWriteTable.

```
1. dbWriteTable(mydb, "mtcars", mtcars[, ], overwrite = TRUE)
```

The overwrite statement is to overwrite the existing table. Now, all the rows of "mtcars" are taken into MySQL as seen here in MySQL Workspace.

```
1  show databases;
2  use sakila;
3  show tables;
4
```

Result Grid | Filter Rows:

Tables_in_sakila
film_list
film_text
inventory
language
mtcars
nicer_but_slow...
payment

Result 2

Updating new rows:

```
1. dbSendQuery(mydb, "update mtcars set drat = 4.5 where gear = 4")
```

Update the rows in a MySQL table by passing the update query using dbSendQuery(). The values are all updated as seen here in MySQL environment as seen here.

```
4
5  Select * from mtcars;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

row_names	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21	6	160	110	4.5	2.62	16.46	0	1	4	4
Mazda RX4 Wag	21	6	160	110	4.5	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	4.5	2.32	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.44	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.46	20.22	1	0	3	1
Duster 360	14.3	8	360	245	3.21	3.57	15.84	0	0	3	4

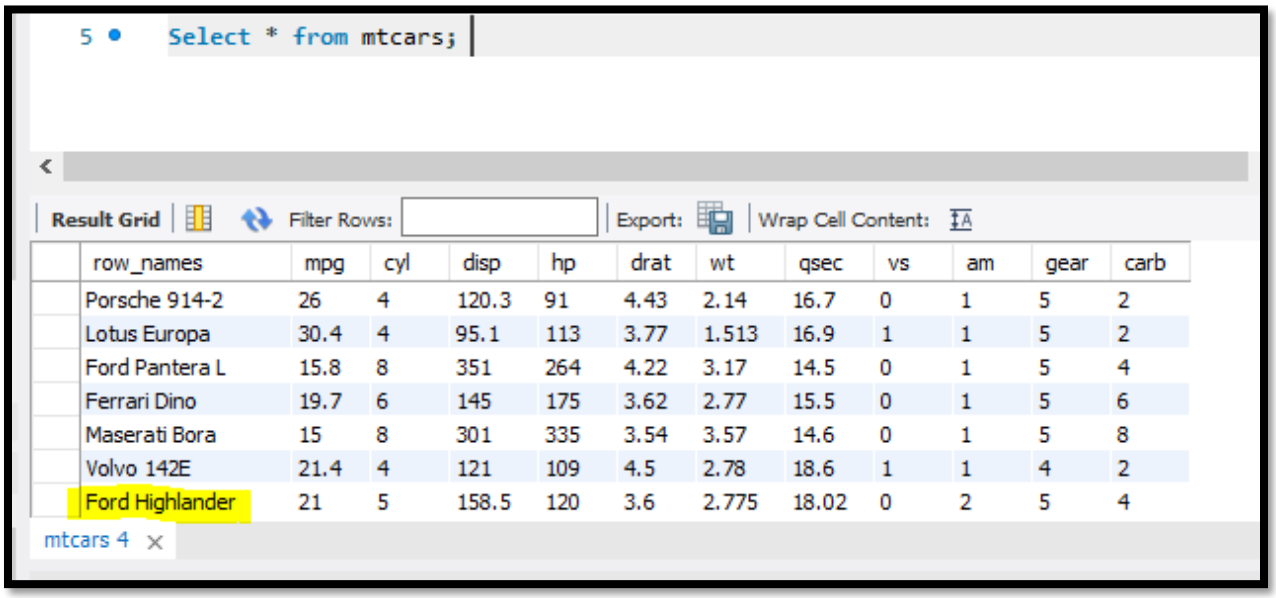
mtcars 3

Inserting new rows:

```
1. dbSendQuery(mydb,
2.              "insert into mtcars(row_names, mpg, cyl, disp, hp, drat,
   wt, qsec, vs, am, gear, carb)
3.     values('Ford Highlander', 21, 5, 158.5, 120, 3.6, 2.775, 18.02, 0,
   2, 5, 4)"
4. )
```

Insert a new row in a MySQL table by passing the update query using dbSendQuery().

The new row is inserted as seen here in MySQL environment as seen here.



row_names	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Porsche 914-2	26	4	120.3	91	4.43	2.14	16.7	0	1	5	2
Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.9	1	1	5	2
Ford Pantera L	15.8	8	351	264	4.22	3.17	14.5	0	1	5	4
Ferrari Dino	19.7	6	145	175	3.62	2.77	15.5	0	1	5	6
Maserati Bora	15	8	301	335	3.54	3.57	14.6	0	1	5	8
Volvo 142E	21.4	4	121	109	4.5	2.78	18.6	1	1	4	2
Ford Highlander	21	5	158.5	120	3.6	2.775	18.02	0	2	5	4

Dropping Tables:

```
1. dbSendQuery(mydb, 'drop table if exists mtcars')
```

Now the ‘mtcars’ table will be dropped from MySQL environment will dropped.

Disconnecting the MySQL database:

```
1. on.exit(dbDisconnect(mydb))
```