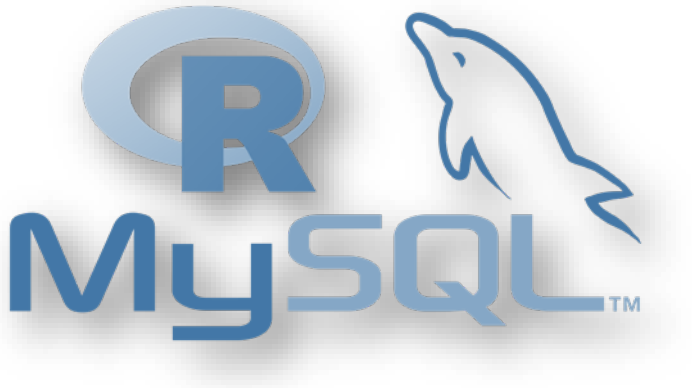
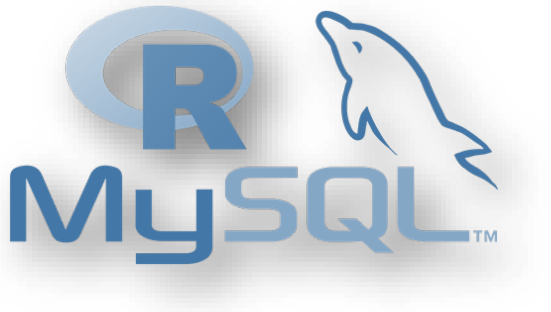
# R and MySQL



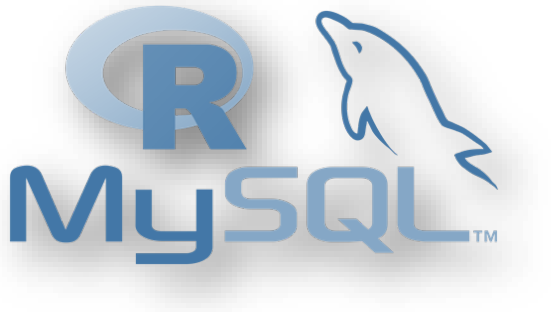
# Content

- Connect
- Query



# Content

- Connect
- Query



# Connect

## RMySQL

To be able to connect to a MySQL database in R, the package **RMySQL** is required.

```
> install.packages("RMySQL")
```

R Script – Client

R Package


Database – Server

# Connect

The first thing to do in R to access a MySQL database is to establish a connection to the server.

```
# import RMySQL package
library(RMySQL)

# establish a connection to the MySQL database
database <- dbConnect(MySQL(),
                      user     = 'username',
                      password = 'password',
                      host     = 'hostname',
                      dbname   = 'databasename')

# close connection
dbDisconnect(database)
```

# Connect

## List content

```
# list the tables
dbListTables(database)

# list columns of a table
dbListFields(database,'tablename')
```

# Connect

## config.cnf

config.cnf

```
[groupname]
user=username
password=password
host=hostname or IP
database=databasename
```

myCode.R

```
# configuration file
configfile <- 'path/filename'

# create a new connection to the database
database <- dbConnect(MySQL(),
                      default.file=configfile,
                      group='groupname')
```

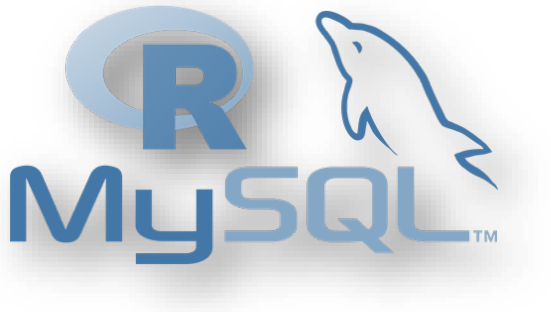
# Connect

## Errors

Nothing to do here… we leave error handling to the functions defined in the RMySQL-package.

# Content

- Connect
- Query



# Query

## dbSendQuery

To execute a query, the function **dbSendQuery** in the RMySQL-package is used:

```
# statement
query <- "some valid sql statement;"

# execute statement
dbSendQuery(database,query)
```

# Query

## INSERT

Inserting records into a database does not require an explicit *commit* call.

```
# statement
query <- "INSERT INTO tablename (attribute1, attribute2)
                         VALUES (value1, value2);"

# execute statement
dbSendQuery(database,query)
```

# Query

## INSERT

To use variables dynamically, the query statement can be composed using the function **paste0**:

```
# set values
value1 <- 10
value2 <- "shrubbery"

# statement
query <- paste0("INSERT INTO tablename (attribute1, attribute2)
                                 VALUES (", value1, ",", value2, ");"

# execute statement
dbSendQuery(database,query)
```

# Query

## SELECT

After sending a SELECT query, the function **dbFetch** is used to load the data:

```
# statement
query <- "SELECT * FROM tablename;"

# execute statement
result <- dbSendQuery(database,query)

# fetch the data
data <- dbFetch(result, n=-1)
```

Number of records to retrieve
Default: n = 500
All: n = -1

# Query

## SELECT

To use variables dynamically, the query statement can be composed using the function **paste0**:

```
# set value
value <- 10

# statement
query <- paste0("SELECT * FROM tablename
                 WHERE attribute = ", value, ";"

# execute statement
result <- dbSendQuery(database,query)

# get the data
data <- dbFetch(result, n=-1)
```

# Query

## dbReadTable

Read a database table and store it in a local data frame.

```
# get the data
data = dbReadTable(database,"tablename")
```

# Query

## dbWriteTable

Write a local data frame as a table into the database.

```
# write to database
dbWriteTable(database,"tablename", data)

# overwrite if table already exists
dbWriteTable(database,"tablename", data, overwrite=TRUE)

# append data if table already exists
dbWriteTable(database,"tablename", data, append=TRUE)
```