RMySQL tutorial

(Access MySQL from R)

Ang Sherpa

National Center for Atmospheric Research

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

What is MySQL?

It is a database management system.

- It provides multiple-user access to a number of databases.
- It allows users to add, access, and process data stored in a computer database.

The SQL part stands for "Standard Query Language."
SQL is the most common standardized language used to access databases since 1986.

MySQL Database

MySQL database is organized into categories:
$\cdot$ *Database* $\rightarrow$ *Tables* $\rightarrow$ *Fields*

- Datas in Database are stored in seperate tables.
  A table can be data frame, data table, lists, e.t.c
- Fields are the column/attributes of a table.

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

RMySQL

- Created by Jeffrey Horner.
- Maintained by Jeroen Ooms and Hadley Wickham.
- It helps R users to interface between R and MySQL.
- Advantages of using RMySQL
  - Convenient to access database from MySQL to R.
  - Modify MySQL data from R.
  - Easy to export data to MySQL.
  - Use MySQL commands in R.

Installation

- Install RMySQL package

$$install.packages(\text{"}RMySQL\text{"})$$

- Load the package

$$library(RMySQL)$$

- Install DBI package.
  DBI provides a common interface for (most of) R's database packages

$$install.packages(\text{"}DBI\text{"})$$

- Load the package

$$library(DBI)$$

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

Connect To MySQL

- MySQL connection object:

$$con <- dbConnect(RMySQL::MySQL(), user = "streamflow\_admin", password = "*****", dbname = "streamflow", host = "****")$$

Database Tables

- Check the tables in the database:

$$dbListTables(con)$$

$$[1]"dataDaily""dataInst""metadata"$$

- Lets see whats in metadata:

$$dbListFields(con,"data\_daily")$$

- Import database to R:

$$dbReadTable(con,"data\_daily")$$

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

Querying Data

Queries can be executed by supplying the SQL statement.
Two ways to perform query:

- Retrieve all the datas at once:

$$dbGetQuery(con, "MySQL\ Statement")$$

- Retrieve chunks of data(s) at a time:

$$dbSendQuery(con, "MySQL\ Statement")$$

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

Querying Data

To retrieve results a chunk at a time:

1. Specify what datas we need.
   $res <- dbSendQuery(con, "SELECT * FROM data\_daily")$
2. Fetch a chunk of data at a time:
   $while(!dbHasCompleted(res)){}$
   $chunk <- dbFetch(res, n = 5)$
   $dosomething$
   $print(nrow(chunk))$
   $\}$
   Here n = maximum number of records to retrieve per fetch
3. Clear the result to free all the resources.
   $dbClearResult(res)$

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

Querying Data

When to use dbSendQuery

Use dbSendQuery when the table is too big.

$$
\begin{aligned}
&result < -data.frame() \\
&res < -dbSendQuery(con, "SELECT * FROM data\_inst") \\
&while(!dbHasCompleted(res)){ \\
&chunk < -dbFetch(res, n = 19510233) \\
&sampleInds < -sample(1 : nrow(chunk), 100) \\
&sample < -chunk[sampleInds,] \\
&result = rbind(result, sample) \\
&print(nrow(chunk)) \\
&\}
\end{aligned}
$$

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

Querying Data

To retrieve all the results:

$$res <- dbGetQuery(con, "SELECT * FROM metadata")$$

dbGetQuery calls dbSendQuery, dbFetch, dbClearResult at once
*Not recommended on large amount of data.

Querying Data

MySQL Statement

To select particular rows:

$$SELECT * FROM\ table\_name\ CONDITION$$

- To get first 100 rows:

$$SELECT * FROM\ metadata\ LIMIT\ 400$$

- To select rows with site _no = ’01010000’ and AND lat_va = ’464202’
  $SELECT * FROM\ metadata\ WHERE\ site\_no = '01010000'\ AND\ lat\_va = '\ 464202'$

MySQL Statement

To select particular column(s):

SELECT column_name FROM table_name

To get column station _id and lat _va from metadata:

SELECT station_id, *lat_va FROM metadata*

To select particular row(s) and column(s):

SELECT column_name FROM table_name CONDITION

To get POSIXct from 2016-01-01 to 2016-12-30 from data _daily:

SELECT POSIXct FROM data_daily WHERE POSIXct $>$
$"2016-01-01"$ *AND POSIXct* $<$ $"2016-12-30"$

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

MySQL Statement

Get number of records in a table.

$$SELECT\ COUNT(*)FROM\ table\_name$$

Get average of stream flow from data _daily

$$select\ avg(q\_cms)\ from\ data\_daily$$

To get information about the table

$$DESCRIBE\ table\_name$$

To clear data from table

$$DELETE\ FROM\ table\_name\ CONDITION$$

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

DataTypes in MySQL

The datatypes interface between R and MySQL is not precise.

Lets see what kind of datatypes is in data _daily:
$res <- dbGetQuery(con, "DESCRIBE\ data\_daily")$

Import data _daily to R:
$data <- dbGetQuery(con, "select * from\ data\_daily\ LIMIT\ 100")$

Structure of data:
$str(data)$
As you can see the datatypes are not same.

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

DataTypes in MySQL

The datatypes interface between R and MySQL is not precise.

Lets see what kind of datatypes is in data _daily:
$res <- dbGetQuery(con, "DESCRIBE\ data\_daily")$

Import data _daily to R:
$data <- dbGetQuery(con, "select * from\ data\_daily\ LIMIT\ 100")$

Structure of data:
$str(data)$
As you can see the datatypes are not same.

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

DataTypes in MySQL

The datatypes interface between R and MySQL is not precise.

Lets see what kind of datatypes is in data _daily:
$res <- dbGetQuery(con, "DESCRIBE\ data\_daily")$

Import data _daily to R:
$data <- dbGetQuery(con, "select * from\ data\_daily\ LIMIT\ 100")$

Structure of data:
$str(data)$
As you can see the datatypes are not same.

Data mappings between R and MySQL

R → MySQL

If you overwrite a table, all the data types will change.

- character → text
- POSIXct → text
- numeric → double

When appending data to MySQL from R, the only differnece float:

numeric → *float*

0.08778222 → 0.0877822

0.09344559 → 0.0934456

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

Data mappings between R and MySQL

MySQL → R

- varchar(n <= 8000) → character
- timestamp → character
- float → integer
- tinyint(1 byte) → integer
- bigint(8 byte) → numeric

Write Table

Write data(s) to database.

dbWriteTable can be used for the following:

- Create new table
- Overwrite existing data
- Append data to table

Write Table

Create a new table

Lets create a data frame "demo" with column "x" and "y"

$$x <- c(1, 2, 3, 4)$$

$$y <- c(letters[1 : 4])$$

$$demo <- data.frame(x, y)$$

Now export it to the database:
$dbWriteTable(con, "demo", demo)$

Write Table

Overwrite a table

Overwrite "demo" with "demo1"

$$a <- c(5, 6, 7, 8, 9)$$

$$b <- c(e, f, g, h, i)$$

$$demo1 <- data.frame(a, b)$$

$$dbWriteTable(con, "demo", demo1, overwrite = TRUE)$$

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

Write Table

Append data to table

Append "demo2" on "demo"

$$c < -c(2, 4, 6, 8, 10)$$

$$d < -c(j, k, l, m, n)$$

$$demo2 < -data.frame(c, d)$$

$$dbWriteTable(con, "demo", demo2, append = TRUE)$$

Remove table "demo":

$$dbRemoveTable(con, "demo")$$

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

Disconnect

Disconnect from the server:

$$dbDisconnect(con)$$

The best practice is to disconnect on exit

$$on.exit(dbDisconnect(con))$$

There is a limitation of 15 connections at a time for MySQL.
To look up active connections:

$$dbGetQuery(con, "show\ processlist")$$

To disconnect the connection:

$$dbGetQuery(con, "kill\ Id")$$

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

References

- https://cran.r-project.org/web/packages/RMySQL/RMySQL.pdf
- http://dev.mysql.com/doc/refman/8.0/en/
- https://cran.r-project.org/web/packages/DBI/DBI.pdf
- http://stackoverflow.com
- https://msdn.microsoft.com/en-us/library/mt590948.aspx