Getting Data: Exercises

Exercises

These exercises accompany the Getting Data tutorial: http://rpubs.com/NateByers/GettingData.

1. To use the `raqdm` package you must have a user name and password from EPA. If you don't have a user name and pass word, you can register for an account here: https://aqs.epa.gov/api. Assuming you had a user name and account, write R code that would request ozone data from all Illinois monitors between June 1 and September 30, 2015. For a list of parameter and state codes, go to the AQS codes website: http://www.epa.gov/aqs/aqs-code-list.

Solution 1

2. Write code that uses `raqdm` to send and receive a request for all criteria pollutant data for Cook County, Illinois between June 1 and August 31, 2015. **Hint**: You can use a `pc` parameter in `getAQDMdata` that will take a text value indicating a group of paramters ("CRITERIA"). You can also find county codes in the AQS codes website: http://www.epa.gov/aqs/aqs-code-list.

Solution 2

3. Use the `read.csv()` function to create a data frame of unit measurement information from the AQS codes website: http://www.epa.gov/aqs/aqs-code-list.

Solution 3

Advanced Exercise

4. Create an `SQLite` database. Read in the parameters and units data from the AQS codes website to load the database with two tables: Parameters and Units. Query the database and pull out a single table with parameters and unit information in it. **Hint**: In your SQL statement, you will need to use a JOIN statement joining on the "Standard Unit" column in the Paramters table and the "Unit" column in the Unit table.

Solution 4

Solutions

Solution 1 First, library the package and set your user name and password if you have not done so.

```
library(raqdm)
setAQDMuser(user = "me@email.com", password = "secret", save = TRUE)
```

On the AQS codes website we can find the number for Illinois (17) and the number for ozone (44201). So we send a request using the `getAQDMdata()` function, assigning the `synchronous` parameter a value of `FALSE`.

```
ozone_req <- getAQDMdata(state = "17", bdate = "20150601", edate = "20150930",
                         param = "44201", synchronous = FALSE)
```

Then wait for the email notification. Once it arrives, run the `getAQDMrequest()` function.

```
ozone <- getAQDMrequest(ozone_req)
head(ozone)
```

```
##    Latitude Longitude Datum Horizontal.Accuracy State.Code County.Code
## 1 39.396075 -89.80974 WGS84                    5         17         117
## 2 39.831522 -89.64093 WGS84                    3         17         167
## 3 39.396075 -89.80974 WGS84                    5         17         117
## 4 39.831522 -89.64093 WGS84                    3         17         167
## 5 39.396075 -89.80974 WGS84                    5         17         117
## 6 39.831522 -89.64093 WGS84                    3         17         167
```

Back to exercises

Solution 2 On the AQS codes website we can find the number for Cook County (031). We send a request using the `getAQDMdata()` function, assigning the `synchronous` parameter a value of `FALSE` and assigning the `pc` parameter a value of `"CRITERIA"`.

```
criteria_req <- getAQDMdata(state = "17", county = "031", bdate = "20150601",
                            edate = "20150831", pc = "CRITERIA",
                            synchronous = FALSE)
```

Then wait for the email notification. Once it arrives, run the `getAQDMrequest()` function.

```
criteria <- getAQDMrequest(criteria_req)
head(criteria)
```

```
##   Latitude Longitude Datum Horizontal.Accuracy State.Code County.Code
## 1 41.66812 -87.99057 WGS84                    5         17          31
## 2 41.66812 -87.99057 WGS84                    5         17          31
## 3 41.66812 -87.99057 WGS84                    5         17          31
## 4 41.66812 -87.99057 WGS84                    5         17          31
## 5 41.66812 -87.99057 WGS84                    5         17          31
## 6 41.66812 -87.99057 WGS84                    5         17          31
```

Back to exercises

Solution 3 On the AQS codes website we can find the url for units by clicking on "Units" and then clicking on "Download Delimited Version of the Code Table". Copy and paste the url into `read.csv()` and assign the `skip` parameter a value of `1`.

```
aqs_units <- read.csv("https://aqs.epa.gov/aqsweb/codes/data/Units.csv",
                      skip = 1, stringsAsFactors = FALSE)
head(aqs_units)
```

```
##   Unit                      Unit_Desc
## 1    1 Micrograms/cubic meter (25 C)
## 2    2  Micrograms/cubic meter (0 C)
## 3    3  Nanograms/cubic meter (25 C)
## 4    4   Nanograms/cubic meter (0 C)
## 5    5 Milligrams/cubic meter (25 C)
## 6    6  Milligrams/cubic meter (0 C)
```

Back to exercises

Solution 4 Read the Parameters and Units into R from the AQS code website, if you have not done so already.

```
aqs_params <- read.csv("https://aqs.epa.gov/aqsweb/codes/data/ParametersByDesc.csv",
                       stringsAsFactors = FALSE, skip = 1)
aqs_units <- read.csv("https://aqs.epa.gov/aqsweb/codes/data/Units.csv",
                      skip = 1, stringsAsFactors = FALSE)
```

You will need to replace the . between the words in the column names with _, or you won't be able to refer to field names in your SQL statement.

```
names(aqs_params) <- sub("\\.", "_", names(aqs_params))
names(aqs_units) <- sub("\\.", "_", names(aqs_units))
```

Create the database in your working directory and load the two tables.

```
library(RSQLite)
db_par_unit <- dbConnect(SQLite(), "par_unit.sqlite")
dbWriteTable(db_par_unit, "Parameters", aqs_params)
```

```
## [1] TRUE
```

```
dbWriteTable(db_par_unit, "Units", aqs_units)
```

```
## [1] TRUE
```

```
dbListTables(db_par_unit)
```

```
## [1] "Parameters" "Units"
```

Now we write a SQL statement that joins the "Standard_Unit" column of the Parameters table on the "Unit" column of the Units table.

```
query <- dbSendQuery(db_par_unit,
                     "SELECT Parameters.Parameter_Desc,
                             Parameters.Standard_Unit,
                             Units.Unit_Desc
                      FROM Parameters
                      INNER JOIN Units
                      ON Parameters.Standard_Unit = Units.Unit")
param_units <- dbFetch(query)
head(param_units)
```

```
param_units
```

```
##                                     Parameter_Desc Standard_Unit
## 1                       1,1,1,2,2-PENTAFLUOROETHANE           078
## 2                         1,1,1,2-TETRACHLOROETHANE           078
## 3 1,1,1-TRICHLORO-2,2-BIS (P-CHLOROPHENYL) ETHANE           003
## 4                         1,1,2,2-TETRACHLOROETHANE           078
## 5           1,1,2-TRICHLORO-1,2,2-TRIFLUOROETHANE           078
## 6                             1,1,2-TRICHLOROETHANE           078
##                       Unit_Desc
## 1     Parts per billion Carbon
## 2     Parts per billion Carbon
## 3 Nanograms/cubic meter (25 C)
## 4     Parts per billion Carbon
## 5     Parts per billion Carbon
## 6     Parts per billion Carbon
```

Now we clear the query result, disconnect from the database, and delete it.

```
dbClearResult(query)
```

```
## [1] TRUE
```

```
dbDisconnect(db_par_unit)
```

```
## [1] TRUE
```

```
unlink("par_unit.sqlite")
```

Back to exercises