Getting Started with the Web of Science PostgreSQL Database for University of Toronto Using R Language

Accessing the Database via High Performance Computing Environment Available on SciNet

Introduction

This tutorial will help querying the Web of Science (WoS) PostgreSQL database for University of Toronto users. The tutorial closely follows SQL and Python tutorials outlined by University of Toronto Map and Data Library. Users will need access to the high performance computing environment on SciNet for querying the WoS database. Instructions for accessing SciNet are outlined by University of Toronto Library.

Login to SciNet, install modules and open R To install needed modules and open R there are two options as shown below:

```
cd $HOME
module load gcc
module load postgresql
module load r/4.1.2
R

# OR

cd $HOME
module load gcc
module load r/4.1.2
singularity pull docker://rocker/tidyverse:4.1.3
singularity exec tidyverse_4.1.3.sif R
```

Download R packages Now that R is opened, download the needed R packages. Once downloaded, attach each package to current session using command library.

```
install.packages(c("DBI", "dplyr", "dbplyr",
                   "RSQLite", "RPostgres",
                   "magrittr", "stringr"))
library("DBI")
library("dplyr")
library("dbplyr")
library("RSQLite")
library("RPostgres")
library("magrittr")
library("stringr")
```

Connecting to database on R We will connect to WoS database using R package `DBI` and using function dbConnect(). Anything followed by a hashtag is a comment in R.

```
db <- 'wos'  # provide the name of database
hostdb <- 'idb1' # host name
dbWoS <- DBI::dbConnect(RPostgres::Postgres(),
                  dbname = db,
                  host = hostdb)
# Let's take a closer look at the database
dbplyr::src_dbi(dbWoS)
```

Getting help with R If you are unclear of any function used, you may type ? followed by function name to pull up the help documentation. Another option is to use help() function. Both options are shown below.

```
?DBI::dbConnect

# OR

help(dbConnect, package = "DBI")
```

a. Search by Title Let's find publications that have the word "visualization". Type

```
searchWords <- c("visualization") # search a word
pubSearchA1 <- dplyr::tbl(dbWoS, "publication") %>%
  dplyr::select(title) %>%
  dplyr::filter(grepl(searchWords, title))
```

Let's find publications that have the words "visualization", and "library" OR "libraries" OR "librarian" in the title. Type

```
searchWords <- c("visualization", "library", "libraries", "librarian") # search multiple words
pubSearchA2 <- dplyr::tbl(dbWoS, "publication") %>%
  dplyr::select(title) %>%
  dplyr::filter(grepl(stringr::str_flatten(searchWords, collapse="|"), title))
```

b. Search by Title words and Year We can also limit searches based on multiple criteria for different fields. Let's run the same search as above, but limit it to only publications published later than 2015. Type

```
searchWords <- c("visualization", "library", "libraries", "librarian")
pubSearchB <- dplyr::tbl(dbWoS, "publication") %>%
  dplyr::select(title, year) %>%
  dplyr::filter(grepl(stringr::str_flatten(searchWords, collapse="|"), title)) %>%
  dplyr::filter(year > "2015")
```

c. Search by Title words and Year, return specific fields only We have been selecting few fields in the publication table, but we can instead select several fields (type, year, title, ref_count) from the publication table. Type

```
searchWords <- c("visualization", "library", "libraries", "librarian")
pubSearchC1 <- dplyr::tbl(dbWoS, "publication") %>%
  dplyr::select(type, year, title, ref_count) %>%
  dplyr::filter(grepl(stringr::str_flatten(searchWords, collapse="|"), title)) %>%
  dplyr::filter(year > "2015")
```

We have been selecting few fields in the publication table, but we can instead select all fields in the publication table. Type

```
searchWords <- c("visualization", "library", "libraries", "librarian")
pubSearchC2 <- dplyr::tbl(dbWoS, "publication") %>%
  dplyr::select(edition, source_id, type, year, month, day, vol,
                issue, page_begin, page_end, page_count, title, ref_count) %>%
  dplyr::filter(grepl(stringr::str_flatten(searchWords, collapse="|"), title)) %>%
  dplyr::filter(year > "2015")
```

d. Search by Title words and Year, but return Author information as well So far these queries have focused on returning data from one table, but you can join tables to get information from multiple

tables, such as publication and author. Let's run the same search from above, but also get author names included in the results. Type

```
searchWords <- c("visualization", "library", "libraries", "librarian")
pubSearchD <- dplyr::tbl(dbWoS, c("publication", "author")) %>%
  dplyr::select(year, title, full_name) %>%
  dplyr::filter(grepl(stringr::str_flatten(searchWords, collapse="|"), title)) %>%
  dplyr::filter(year > "2015")

# Note: This will result in publication titles being duplicated if there are multiple authors to list.
```

e. Search by Title words and Year, but return Author and Source information as well You can join one table to more than one other table to pull in more information into your results. Let's run the query from example d, but add the journal information as well. Type

```
searchWords <- c("visualization", "library", "libraries", "librarian")
pubSearchE <- dplyr::tbl(dbWoS, c("publication", "author", "source")) %>%
  dplyr::select(year, title, full_name, name) %>%
  dplyr::filter(grepl(stringr::str_flatten(searchWords, collapse="|"), title)) %>%
  dplyr::filter(year > "2015")
```

f. Search by Title words, Year and Author name You can also limit searches based on information in these multiple tables. Let's run the same search from above, but also limit to only authors with the last name "Reid". Type

```
searchWords <- c("visualization", "library", "libraries", "librarian")
pubSearchF <- dplyr::tbl(dbWoS, c("publication", "author", "source")) %>%
  dplyr::select(year, title, full_name, name) %>%
  dplyr::filter(grepl(stringr::str_flatten(searchWords, collapse="|"), title)) %>%
  dplyr::filter(grepl("Reid", full_name)) %>%
  dplyr::filter(year > "2015")
```

g. Search by Year and Source name This can continue to get more complicated. You might want to join a table in order to query a field, but aren't interested in including the data from that table in the final results. Note you could construct a query similar to example f above, the only difference is that the columns from the source table are not included here, but are included in example f. For this example, let's query the database to find all recent publications with author information for publications from the journal called "Scientometrics". This query finds all the source IDs where the source name is "Scientometrics" then filters publications that have that source ID, plus the other criteria outlined below. Type

```
pubSearchG <- dplyr::tbl(dbWoS, c("publication", "author", "source")) %>%
  dplyr::select(year, title, full_name, name) %>%
  dplyr::filter(name == toupper("Scientometrics")) %>%
  dplyr::filter(year > "2019") %>%
  dplyr::select(-name)
```

h. Search by Title words, Year and Author institution Some tables in the database are bridging tables, where there are many-to-one relationships, such as an author having many addresses. Let's query the database to find all publications with the word "visualization" in the title, published in the last couple of years from authors from the University of Toronto. First you find all the address IDs that are for the University of Toronto, then you find all the author IDs that have those address IDs, and then filter by those authors, plus the other criteria outlined below. (Note: Just to simplify the query and make it run faster for this example, we're just looking for addresses with "Univ Toronto". Type

```
searchWords <- c("visualization")
pubSearchH <- dplyr::tbl(dbWoS,
  c("publication", "author", "address")) %>%
  dplyr::select(year, title, full_name, address) %>%
  dplyr::filter(grepl(searchWords, title)) %>%
  dplyr::filter(year > "2019") %>%
  dplyr::filter(grepl("Univ Toronto", address))
```

i. Search by Keywords and Year Here we are using another bridging table, this time to find publications based on a particular descriptor, such as a subject or keyword. This example is similar to the one above except searching by Keywords Plus (standardized keywords in the Web of Science dataset) instead of author affiliation. Let's query the database to find all publications from 2020 that have a Keywords Plus field roughly equal to "Artificial Intelligence". Type

```
pubSearchI <- dplyr::tbl(dbWoS, c("publication", "descriptor")) %>%
  dplyr::select(year, title, text) %>%
  dplyr::filter(year == "2020") %>%
  dplyr::filter(text == "Artificial Intelligence")
```

j. Search by Title words and Year, returning only publication title and abstract One useful field for text analysis that we haven't seen in our examples yet would be to obtain abstracts for the items found. Let's run a search with similar search parameters to example b, but return titles and abstracts only. Type

```
searchWords <- c("visualization", "library", "libraries", "librarian")
pubSearchJ <- dplyr::tbl(dbWoS, c("abstract", "publication")) %>%
  dplyr::select(title, year, text) %>%
  dplyr::filter(grepl(stringr::str_flatten(searchWords, collapse="|"), title)) %>%
  dplyr::filter(year > "2019")
```

k. Search for articles that cite a subset of articles The Web of Science dataset is very valuable to analyze citation networks. For example, we can use another bridging table called references to find all publication IDs that cited or are cited by other publication IDs. Let's query the database to find all the articles that cite a (very small) subset of items. The subset is similar to example b above, find all articles that have the words "visualization", and "library" OR "libraries" OR "librarian" in the title, but this time only published after 2019. These types of queries are intensive and can take a while to run, so this is a very simple and small example to get you started. Type

```
searchWords <- c("visualization", "library", "libraries", "librarian")
# First create the subset
pubSuset <- dplyr::tbl(dbWoS, c("publication")) %>%
  dplyr::select(title, year, id) %>%
  dplyr::filter(grepl(stringr::str_flatten(searchWords, collapse="|"), title)) %>%
  dplyr::filter(year > "2019") %>%
  dplyr::pull(id)
# Search the subset publication id in citing_id
pubSearchK <- dplyr::tbl(dbWoS, c("publication", "reference")) %>%
  dplyr::select(title, year, citing_id) %>%
  dplyr::filter(.data[["citing_id"]] %in% pubSuset)
```

l. Search for articles that are cited by a subset of articles We can also query this the opposite way to find articles cited by a subset of articles. Let's query the database to find all the articles that are cited by

a (very small) subset of items. The subset is the same as in example k, and the modifications to the query in example k are minimal. Type

```
searchWords <- c("visualization", "library", "libraries", "librarian")
# First create the subset
pubSuset <- dplyr::tbl(dbWoS, c("publication")) %>%
  dplyr::select(title, year, id) %>%
  dplyr::filter(grepl(stringr::str_flatten(searchWords, collapse="|"), title)) %>%
  dplyr::filter(year > "2019") %>%
  dplyr::pull(id)
# Search the subset publication id in cited_ids
pubSearchL <- dplyr::tbl(dbWoS, c("publication", "reference")) %>%
  dplyr::select(title, year, cited_id) %>%
  dplyr::filter(.data[["cited_id"]] %in% pubSuset)
```

To save results and quit R These files will be saved to $HOME

```
# To save a specific object, pubSearchJ, to a file rds
saveRDS(pubSearchJ, file = paste0("pubSearchJDate", Sys.Date(), ".rds"))

# To save a specific object, pubSearchJ, to a file csv
save.csv(pubSearchJ, file = paste0("pubSearchJDate", Sys.Date(), ".csv"))

# To save the entire workspace image
save.image(file = paste0("WoSQueryDate", Sys.Date(), ".RData"))

q() # Enter q() at prompt to quit R

# [END]
```

If you would like to 'Save workspace image?', press 'y'.

Maintainer

Anjali Silva (a.silva@utorontoca). Last updated 8 April 2022.

Contributions

This tutorial welcomes issues, enhancement requests, and other contributions. To submit an issue, use the GitHub issues.

Acknowledgments

SciNet HPC Consortium, University of Toronto, ON, Canada for all the SciNet setup support.