Task 6: Data Loading Tasks

James Mbewu

14/06/2021

Overview

This R Markdown document serves as the report and code for Task 6: Data Loading Tasks for the City of Cape Town Data Science Unit Code Challenge. The code by itself can be found in the file further_data_transformations.R.

Brief

Select a subset of columns (including the H3 index column) from the `sr_hex.csv` or the anonymised file created in the task above, and write it to the write-only S3 bucket.

Be sure to name your output file something that is recognisable as your work, and unlikely to collide with the names of others.

Report

The goal of this task is to upload a file containing a subset of the columns of the `sr_hex.csv` file to the AWS S3 bucket provided. To do this we again use the package paws. We will also verify that the upload has completed successfully by comparing the eTag returned from AWS S3 with the MD5 checksum of the file we uploaded.

Load Credentials

We first load the credentials we will use later for the upload.

```
rm(list = ls())


# Setup AWS S3 access
aws_credentials_url <- "https://cct-ds-code-challenge-input-data.s3.af-south-1.amazonaws.com/ds_code_cha
secrets <- fromJSON(aws_credentials_url)

Sys.setenv(
  "AWS_ACCESS_KEY_ID" = secrets$s3$access_key,
  "AWS_SECRET_ACCESS_KEY" = secrets$s3$secret_key,
  "AWS_DEFAULT_REGION" = "af-south-1",
  "AWS_S3_ENDPOINT" = "",
  "AWS_REGION" = "af-south-1"
)
```

Prepare Data

Next we load the data from file and choose a subset of the columns (NotificationNumber, h3_level_8_index, Latitude and Longitude) to upload. The resulting data frame is then saved to file.

```
# PREPARE DATA ============================================================

start <- Sys.time()

# Download data if not present
# TODO: using local data for now

end_download <- Sys.time()
download_time <- difftime(end_download, start)
print(paste("download_time =",download_time))

start_load <- Sys.time()

# Read in data
# For testing purposes we are only going to read in a subset of the data
sr_hex_data <- read_csv("data/sr_hex.csv",n_max = 1000,skip = 0, col_types = cols())

end_load <- Sys.time()
load_time <- difftime(end_load, start_load)
print(paste("load_data_time =",load_time))


start_save_subset <- Sys.time()

# Choose a subset of the columns
cols <- c("NotificationNumber","h3_level8_index","Latitude","Longitude")
#cols <- c("NotificationNumber")
sr_hex_data <- sr_hex_data[ , cols]

# Write the data to file
sr_sub_filename <- "data/sr_hex_sub_james_mbewu.csv"
write.table(sr_hex_data,file=sr_sub_filename,
            row.names = FALSE, sep=",",quote=FALSE)

end_save_subset <- Sys.time()
save_subset_time <- difftime(end_save_subset, start_save_subset)
print(paste("save_subset_time =",save_subset_time))
```

Upload data

Next we upload the file using paws and give it the name `sr_hex_sub_james_mbewu.csv` in the S3 bucket.

```
# UPLOAD DATA ============================================================

start_upload <- Sys.time()

# Setup connection to AWS S3
s3 <- paws::s3()

# # Upload file
# result <- s3$put_object(
#   Body = sr_sub_filename,
#   Bucket = "testbucketmbewu",
#   Key = "sr_hex_sub_james_mbewu.csv"
```

```
# )

result <- s3$put_object(
  Body = sr_sub_filename,
  Bucket = "cct-ds-code-challenge-input-data",
  Key = "sr_hex_sub_james_mbewu.csv"
)

end_upload <- Sys.time()
upload_time <- difftime(end_upload, start_upload)
print(paste("upload_time =",upload_time))
```

Verification

Finally we verify that the file uploaded was the file we sent to the S3 bucket. This can be done by comparing the eTag returned by S3 with the MD5 checksum of the file. If the upload was not a multipart upload, then they should be identical. If the upload was a multipart upload, then we would need to do some calculations to figure out the MD5 checksum of the merged file in the S3 bucket and verify it is indeed the same as what we sent.

```
# VERIFICATION ========================================================================

start_verification <- Sys.time()
# compare the eTag and the md5sum
eTag <- str_extract(result$ETag,"[:alnum:]+")
num_parts <- as.numeric(str_match(string = eTag,
                          pattern = '-(\\d+)$')[2])

# Check it wasn't a multipart upload
if(is.na(num_parts)){
  md5 <- as.character(md5sum(sr_sub_filename))
  if(md5 == eTag) {
    print("File successfully uploaded. :)")
  }
  else{
    print("File was NOT successfully uploaded. :(")
  }
} else{
  print("Multipart upload verification not supported yet.")
}


end_verification <- Sys.time()
verification_time <- difftime(end_verification, start_verification)
print(paste("verification_time =",verification_time))


total_time <- difftime(end_verification, start)
print(paste("total_time =",total_time))
```

Performance

The time taken to perform these operations was logged to the command line and is presented in the table below.

```
# Table showing the logged times for each subsection and the total
log_times <- c(download_time,load_time,save_subset_time,
               upload_time,verification_time,total_time)
log_times_df <- data.frame(Times = log_times)
colnames(log_times_df) <- c("Times")
rownames(log_times_df) <- c("Download","Load","Save Subset",
                            "Upload","Verification","Total")
kable(log_times_df, caption = "Log times for Data Loading Tasks Task")
```

Conclusions

Unfortunately, when I ran this on Monday morning I got a `Error: AccessDenied (HTTP 403). Access Denied` error. I'm not too sure what the issue was because I was able to successfully query data in the bucket. So it isn't a problem of connecting to it. In addition I was able to upload data when testing it with my own bucket.

In any case, when testing on my own bucket the I could verify that the file in the bucket is indeed the same file that we sent and the upload was successful.