Database Exemplar

Will Doyle

4/2/2019

1. Using the nyclfights13 library, open all of the data tables and turn them into a databse. Make sure to include flights, airlines, airports, weather and planes.

```
# Mostly taken from : http://cran.r-project.org/web/packages/dplyr/vignettes/databases.html circa 2014

# Will need: nycflights13 RSQLite,

#Get libraries
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.5.3
```

```
## -- Attaching packages ----------------------------------------------------------------- tidyverse 1.2.1
```

```
## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.1
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## Warning: package 'ggplot2' was built under R version 3.5.3
```

```
## Warning: package 'tibble' was built under R version 3.5.3
```

```
## Warning: package 'tidyr' was built under R version 3.5.3
```

```
## Warning: package 'purrr' was built under R version 3.5.3
```

```
## Warning: package 'dplyr' was built under R version 3.5.3
```

```
## Warning: package 'stringr' was built under R version 3.5.3
```

```
## Warning: package 'forcats' was built under R version 3.5.3
```

```
## -- Conflicts -------------------------------------------------------------------- tidyverse_conflicts()
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(nycflights13)
```

```
## Warning: package 'nycflights13' was built under R version 3.5.3
```

```
library(RSQLite)
```

```
## Warning: package 'RSQLite' was built under R version 3.5.3
```

```
data(flights)
data(airlines)
data(airports)
data(weather)
data(planes)
```

```
con <- dbConnect(RSQLite::SQLite(), ":memory:")

#Write flights tables to database (you won't usually do this)
dbWriteTable(con,
```

```
             "flights",
             as.data.frame(flights))

dbWriteTable(con,
             "planes",
             as.data.frame(planes))

dbWriteTable(con,
             "airlines",
             as.data.frame(airlines))

dbWriteTable(con,
             "weather",
             as.data.frame(weather))

dbWriteTable(con,
             "airports",
             as.data.frame(airports))
```

Tidyverse

Create a data frame from a SQL pull from the database that consists only of flights that took off from JFK in Ma

```
req_text<-"Select * from flights"

#Send query through connection
req<-dbSendQuery(con,req_text)

#Generate dataframe from results
req_df<-dbFetch(req,n=-1)

#Good practice: clear request
dbClearResult(req)

req_df%>%filter(origin=="JFK",month==5)->req_df
```

SQL Way

```
req_text<-"Select * from flights
           Where origin=='JFK' AND month==5"


#Send query through connection
req<-dbSendQuery(con,req_text)

#Generate dataframe from results
req_df<-dbFetch(req,n=-1)

#Good practice: clear request
dbClearResult(req)
```

Create a data frame from a SQL pull from the database that consists only of flights that took off on-time (a delay of less than 10 minutes) from Newark at temperatures of less than 40 degrees F.

```
req_text<-"Select * from flights"

#Send query through connection
req<-dbSendQuery(con,req_text)

#Generate dataframe from results
req_df<-dbFetch(req,n=-1)

flights<-req_df

#Good practice: clear request
dbClearResult(req)

req_text<-"Select * from weather"

#Send query through connection
req<-dbSendQuery(con,req_text)

#Generate dataframe from results
req_df<-dbFetch(req,n=-1)

weather<-req_df

#Good practice: clear request
dbClearResult(req)

combined<-left_join(flights,weather,by=c("origin","year", "month" ,"day", "hour"  ))

combined<-combined%>%
  filter(origin=="EWR"&dep_delay<10&temp<40)
```

```
req_text<-"Select f.dep_delay, f.origin, w.temp
          FROM flights f
          JOIN weather w
          WHERE f.origin=w.origin AND f.year=w.year AND f.month=w.month AND f.day=w.day AND f.hour=w.hou
          "


#Send query through connection
req<-dbSendQuery(con,req_text)

#Generate dataframe from results
req_df<-dbFetch(req,n=-1)

flights<-req_df

#Good practice: clear request
dbClearResult(req)
```

#Create data frame from a SQL pull from the database that consists of #planes flown by United.

```
req_text<-"Select * from flights"

#Send query through connection
req<-dbSendQuery(con,req_text)

#Generate dataframe from results
req_df<-dbFetch(req,n=-1)

flights<-req_df

#Good practice: clear request
dbClearResult(req)

req_text<-"Select * from planes"

#Send query through connection
req<-dbSendQuery(con,req_text)

#Generate dataframe from results
req_df<-dbFetch(req,n=-1)

planes<-req_df

flights<-flights%>%select(carrier,tailnum)%>%filter(carrier=="UA")

combined<-left_join(flights,planes, by="tailnum")

combined<-combined%>%
  group_by(tailnum,year,model)%>%
  summarize(total=n())%>%
  arrange(year)
```

```
req_text<-"SELECT p.tailnum, p.model, p.year,f.carrier
           FROM planes p
           JOIN flights f
           WHERE p.tailnum=f.tailnum AND f.carrier='UA'
           ORDER BY p.year
           "

#Send query through connection
req<-dbSendQuery(con,req_text)
```

```
## Warning: Closing open result set, pending rows
```

```
#Generate dataframe from results
req_df<-dbFetch(req,n=-1)

combined<-req_df

#Good practice: clear request
dbClearResult(req)
```