Assignment 10Sql A: Data Retrieval via SQL

Joshua Conte

July 16, 2017

Data Retrieval via SQL

For this assignment, I need to revisit the Bird Strike database created in the last assignment (assignment 9). Using SQL SELECT statements, write R programs that retrieve data from the database. I need to submit this assignment as an extension to week 9 assignment i.e. add this assignment at the bottom of assignment 9.

Reviewing Assignment 9

Setting up R

First I configure R studio with the parameters below:

```
# clears the console in RStudio
cat("\014")
```

```
# clears environment
rm(list = ls())

# Set working directory
setwd("C:/R/DA5020/Week_10/")

# Load required packages
require(RSQLite)
```

```
## Loading required package: RSQLite
```

```
require(sqldf)
```

```
## Loading required package: sqldf
```

```
## Loading required package: gsubfn
```

```
## Loading required package: proto
```

```
require(lubridate)
```

```
## Loading required package: lubridate
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(tidyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:lubridate':
##
##     intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Get the data

Then I load in the data as a data frame below:

```
# Load the data
# I also added NA to all blank cells to make it easier to analyze and
# stringsAsFactors = FALSE so I can remove levels from the data.
if (!exists("bird.strikes")) {
  bird.strikes <-
    read.csv(
      unz("Bird Strikes.zip", "Bird Strikes.csv"),
      header = TRUE,
      na.strings = c("", "NA"),
      stringsAsFactors = FALSE,
      sep = ","
    )
}
```

Make a database

Next I will create a new, empty SQLite database where I can store the bird strike data. SQLite has a rather simple data storage mechanism, all data for a database is installed within a single file. The name of this file must be specified when the database is created, and a connection to this database is returned and used in subsequent commands to access and manipulate the data and data structures within the database.

```
birdstrike <- dbConnect(SQLite(),'birdstrike.db')
```

This command creates a file named "birdstike.db".

Next I will take the information needed from the bird strike data and put it into different data frames to make different tables. I need to do this so the database will have the correct relationships in 3NF form. The primary key for the tables is Record ID and the remaining are not unique. Dates are tough in SQL, so I converted the date to the default in R and put it back as a character. This way the date is ready to be converted in SQL if needed (using CAST) and the non-utilized time stamp is removed.

```
# Create a table.  I created the table as described above in the introduction to
# databse section.  I defined the primary key and specified the data types.
dbSendQuery(
  birdstrike,
  "CREATE TABLE table1 (
  RecordID INT,
  AircraftType CHAR,
  AircraftAirlineOperator CHAR,
```

```
  AirportName CHAR,
  AircraftMakeModel CHAR,
  FlightDate DATE,
  PRIMARY KEY (RecordID))"
  )
```

```
## <SQLiteResult>
##   SQL  CREATE TABLE table1 (
##   RecordID INT,
##   AircraftType CHAR,
##   AircraftAirlineOperator CHAR,
##   AirportName CHAR,
##   AircraftMakeModel CHAR,
##   FlightDate DATE,
##   PRIMARY KEY (RecordID))
##   ROWS Fetched: 0 [complete]
##        Changed: 0
```

```
# The code below works by first listing all column data I want to use, then
# processing each column at a time to be written to the database in the required
# format.
RecordID <- bird.strikes$Record.ID
AircraftType <- bird.strikes$Aircraft..Type
AircraftAirlineOperator <-
  bird.strikes$Aircraft..Airline.Operator
AirportName <- bird.strikes$Airport..Name
AircraftMakeModel <- bird.strikes$Aircraft..Make.Model
FlightDate <- as.character(as.Date(bird.strikes$FlightDate, format = "%m/%d/%Y"))

# I put the data above into a data frame called data
data = cbind.data.frame(
  RecordID,
  AircraftType,
  AircraftAirlineOperator,
  AirportName,
  AircraftMakeModel,
  FlightDate
)

# This puts the data frame into the table and into the database
dbWriteTable(
  conn = birdstrike,
  name = "table1",
  data,
  append = T,
  row.names = F
)
```

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

To have proper relationships, a second table is needed. For the second table, I need to break up "Conditions: Precipitation without having it comma separated. Below is an easier way of putting data frames into a database:

```
# This section requires tidyr and dplyr
```

```
makeTable2 <- function(x) {
  # This function will take Record ID and Conditions..Precipitation and split up the
  # comma separated values and put it with the proper Record ID.

  # This puts the required info into a data frame
  df <- data.frame(x$Record.ID,
                   x$Conditions..Precipitation)
  # This gives it proper names
  names(df) <-
    c("RecordID",
      "ConditionsPrecipitation")

  # This removes the comma separated values in the conditions column
  df2 <- df %>%
    mutate(ConditionsPrecipitation =
             strsplit(as.character(ConditionsPrecipitation), ",")) %>%
    unnest(ConditionsPrecipitation)

  return(df2)
}

table_2 <- makeTable2(bird.strikes)
```

Now that the data is extracted I can add it to a database.

With the data loaded, and an active database connection to the SQLite database, I can write the data by specifying the connection, the name of the table, and the name of the data frame that contains the data to be persisted.

```
# Add the second table to the database
dbWriteTable(birdstrike, "table2", table_2)

# confirm tables
dbListTables(birdstrike)
```

```
## [1] "table1" "table2"
```

Now the database birdstrike has all of the required information in it with proper relationships and normalized to 3NF.

Assignment 9 Tasks

Before I begin the remaining tasks, I will clear the environment and load only the database to free up memory:

```
# clears environment
rm(list = ls())

#Load database
birdstrike <- dbConnect(SQLite(),'birdstrike.db')
```

Assignment 9, Task 2:

Write a SQL SELECT statement that counts the number of incidents where the incident reported fog during the incident.

```
# This uses basic SQL syntax, I selected ConditionsPrecipitation and count. I
# defined the count as BirdStrikes and this information is in table2.  Since this is
# text, I used where and like to count.
dbGetQuery(birdstrike, "SELECT ConditionsPrecipitation, count(*)
           AS BirdStrikes
           FROM table2
           WHERE ConditionsPrecipitation
           LIKE 'fog'")
```

```
##   ConditionsPrecipitation BirdStrikes
## 1                     Fog        1061
```

Assignment 9, Task 3:

Write a function called CountIncidents(AircraftType) that accepts an aircraft type and returns the number of birdstrikes incidents for the aircraft type.

```
CountIncidents <- function(x) {
  # This function is the same as task 2, however, I needed to use the fn$ function of
  # package "sqldf" to get the variable "x" to work.
  fn$dbGetQuery(
    birdstrike,
    "SELECT AircraftType, count(*)
    AS BirdStrikes
    FROM table1
    WHERE AircraftType
    LIKE ('$x')"
  )
}

# This calls the function with argument in quotes
CountIncidents("Airplane")
```

```
##   AircraftType BirdStrikes
## 1     Airplane       73521
```

Assignment 9, Task 4:

Write a function called Incidents(Airline) that accepts an airline and returns a data frame that contains all incidents for that airline. Limit the columns to: AirportName, AircraftModel, and Flight Date.

```
Incidents <- function(x) {
  # This is similar to the other tasks, but I selected 3 columns and did not use
  # count. I also ordered the results by date from oldest to newest.
  fn$dbGetQuery(
    birdstrike,
    "SELECT AirportName, AircraftMakeModel, FlightDate
    FROM table1
    WHERE AircraftAirlineOperator
    LIKE ('$x')
    ORDER BY FlightDate"
  )
}
```

```r
# This calls the function with argument in quotes
airlineIncidents.df <- Incidents("CONTINENTAL AIRLINES")

# This shows the summary and the first six rows of the data
summary(airlineIncidents.df)
```

```
##  AirportName        AircraftMakeModel   FlightDate
##  Length:833         Length:833         Length:833
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
```

```r
head(airlineIncidents.df)
```

```
##                AirportName AircraftMakeModel FlightDate
## 1 NEWARK LIBERTY INTL ARPT         B-757-200 2000-01-01
## 2  LAFAYETTE REGIONAL (LA)            ATR-42 2000-01-04
## 3                  UNKNOWN         B-737-800 2000-01-27
## 4                  UNKNOWN         B-737-500 2000-02-17
## 5 NEWARK LIBERTY INTL ARPT             MD-82 2000-03-06
## 6                  UNKNOWN             B-737 2000-03-08
```

Assignment 9, Task 5:

Write a function called CountIncidentsByAirline() that creates a data frame where the first column is a name of an Aircraft and the second column is the total number of incidents the Airline had.

```r
CountIncidentsByAirline <- function(x) {
  # Function to group airlines and count the number of incidents for each
  # Query that extracts airline and its count by grouping
  query4 <-
    dbGetQuery(
      birdstrike,
      paste(
        "Select AircraftMakeModel, count(AircraftMakeModel) as BirdStrikes
        from table1 group by AircraftMakeModel
        ORDER BY BirdStrikes DESC"
      )
    )
  # Putting results into data frame
  e <- data.frame(query4)
  # Returning results
  return(e)
}
# This calls the function with argument in quotes
airlineIncidentsCount.df <-
  CountIncidentsByAirline()

# I use head to show the first six rows and summary to show the data info
head(airlineIncidentsCount.df)
```

```
##   AircraftMakeModel BirdStrikes
## 1           UNKNOWN       24637
## 2         B-737-300        5524
## 3             A-320        4654
```

```
## 4      CL-RJ100/200        4262
## 5         B-737-700        4046
## 6         B-757-200        3945
```

```
summary(airlineIncidentsCount.df)
```

```
##  AircraftMakeModel   BirdStrikes
##  Length:526         Min.   :    1.0
##  Class :character   1st Qu.:    2.0
##  Mode  :character   Median :   10.5
##                     Mean   :  189.0
##                     3rd Qu.:   63.0
##                     Max.   :24637.0
```

Begin Assignment 10Sql A

Assignment 10Sql A Tasks

Assignment 10Sql A, Task 1:

How many bird strikes occurred for American Airlines? This is a single number.

```
# I use SQL to select AircraftAirlineOperator and count, where
# AircraftAirlineOperator is American Airlines.
dbGetQuery(
  birdstrike,
  "SELECT AircraftAirlineOperator, count(*)
  AS BirdStrikes
  FROM table1
  WHERE AircraftAirlineOperator
  LIKE 'American Airlines'"
)
```

```
##   AircraftAirlineOperator BirdStrikes
## 1       AMERICAN AIRLINES        3851
```

Assignment 10Sql A, Task 2:

How many bird strikes were there for each airline? Show the airline name, including UNKNOWN, and the number of strikes?

```
# This gets all of the airlines in a data frame with the birdstrike count.  I added
# ORDER BY DESC to sort the data from largest to smallest make the data more
# presentable.
df.airline <- dbGetQuery(birdstrike,
            "SELECT AircraftAirlineOperator, COUNT(AircraftAirlineOperator) as
            BirdStrikes FROM table1
            group by AircraftAirlineOperator
            ORDER BY BirdStrikes DESC
            ")
# This prints the first 6 rows and summary
head(df.airline)
```

```
##   AircraftAirlineOperator BirdStrikes
```

```
## 1                 UNKNOWN       21441
## 2                MILITARY        9193
## 3         UNITED AIRLINES        8721
## 4      SOUTHWEST AIRLINES        7485
## 5                BUSINESS        7167
## 6           FEDEX EXPRESS        4423
```

```
summary(df.airline)
```

```
##  AircraftAirlineOperator  BirdStrikes
##  Length:372              Min.   :    1.0
##  Class :character        1st Qu.:    1.0
##  Mode  :character        Median :    5.0
##                          Mean   :  267.2
##                          3rd Qu.:   20.0
##                          Max.   :21441.0
```

Assignment 10Sql A, Task 3:

Which airline had the most bird strikes, excluding military and unknown?

```
# This uses MAX to get the airline with the most BirdStrikes and this uses WHERE NOT
# IN to remove military and unknown
dbGetQuery(birdstrike,
           "SELECT AircraftAirlineOperator, MAX (BirdStrikes) AS BirdStrikes
            FROM (SELECT AircraftAirlineOperator,
            COUNT(AircraftAirlineOperator) BirdStrikes
            FROM table1
            WHERE AircraftAirlineOperator NOT IN ('UNKNOWN', 'MILITARY')
            GROUP BY AircraftAirlineOperator)
           ")
```

```
##   AircraftAirlineOperator BirdStrikes
## 1         UNITED AIRLINES        8721
```

Assignment 10Sql A, Task 4:

Which bird strikes occurred for Helicopters? List all the bird strike incidents with date.

```
# This stores a data frame with helicopter and date.  This selects flightdate and
# counts the number for each date as birdstrikes.  Then filters by helicopter and
# returns the count from largest to smallest.
df.Aircraft <- dbGetQuery(
    birdstrike,
    "SELECT FlightDate, count(FlightDate)
    AS BirdStrikes
    FROM table1
    WHERE AircraftType
    LIKE ('Helicopter')
    GROUP BY FlightDate
    ORDER BY BirdStrikes DESC"
  )

# This is the first 6 lines of a data frame with all the bird strike incidents with
```

```r
# date
head(df.Aircraft)
```

```
##   FlightDate BirdStrikes
## 1 2011-09-23           5
## 2 2009-05-06           4
## 3 2010-05-16           4
## 4 2010-05-17           4
## 5 2010-09-13           4
## 6 2011-09-15           4
```

```r
# This is the summary of the data
summary(df.Aircraft)
```

```
##   FlightDate         BirdStrikes
##  Length:810         Min.   :1.000
##  Class :character   1st Qu.:1.000
##  Mode  :character   Median :1.000
##                     Mean   :1.247
##                     3rd Qu.:1.000
##                     Max.   :5.000
```

Assignment 10Sql A, Task 5:

Which airlines had more than 10 bird strikes? List the airline names only excluding military and unknown.

```r
# This is very similar to task 2, but with WHERE NOT IN to remove military and
# unknown and HAVING COUNT > 10.  I kept the data sorted with the order statement.
df.airline.greater <- dbGetQuery(birdstrike,
            "SELECT AircraftAirlineOperator, COUNT(AircraftAirlineOperator) as
            BirdStrikes FROM table1
            WHERE AircraftAirlineOperator NOT IN ('UNKNOWN', 'MILITARY')
            group by AircraftAirlineOperator
            HAVING COUNT (AircraftAirlineOperator) > 10
            ORDER BY BirdStrikes DESC
            ")

# This prints the first 3 rows then the last 3 to show military and unknown have
# been removed and there is nothing smaller than 11. I also showed the summary.
head(df.airline.greater, 3)
```

```
##   AircraftAirlineOperator BirdStrikes
## 1         UNITED AIRLINES        8721
## 2      SOUTHWEST AIRLINES        7485
## 3                BUSINESS        7167
```

```r
tail(df.airline.greater, 3)
```

```
##     AircraftAirlineOperator BirdStrikes
## 122         BIGSKY AIRLINES          11
## 123      CAPITAL CARGO INTL          11
## 124                 WESTJET          11
```

```r
summary(df.airline.greater)
```

```
##  AircraftAirlineOperator  BirdStrikes
```

```
##  Length:124             Min.   :  11.0
##  Class :character       1st Qu.:  20.0
##  Mode  :character       Median :  67.0
##                         Mean   : 548.4
##                         3rd Qu.: 401.5
##                         Max.   :8721.0
```