Practicum I CS5200

Code

Balakrishna, Dhanush

Summer Full 2023

##Connect to Database

Hide

```
library(DBI)
library(RMySQL)

db_name_fh <- "sql9629190"
db_user_fh <- "sql9629190"
db_host_fh <- "sql9.freemysqlhosting.net"
db_pwd_fh <- "1JaewqfKIc"
db_port_fh <- 3306

mydb <-  dbConnect(RMySQL::MySQL(), user = db_user_fh, password = db_pwd_fh, dbname =
db_name_fh, host = db_host_fh, port = db_port_fh)
```

##Create Database

Hide

```
create table if not exists airports(aid INTEGER NOT NULL PRIMARY KEY,
                                    airportState TEXT,
                                    airportCode TEXT)
```

Hide

```
create table if not exists flights(
                  fid INTEGER NOT NULL,
                  dates DATE,
                  origin INTEGER,
                  airline TEXT,
                  aircraft TEXT,
                  altitude NUMERIC unsigned CHECK (altitude>0),
                  heavy BIT,
                  PRIMARY KEY(fid),
                  FOREIGN KEY(origin) REFERENCES airports(aid))
```

Hide

```
create table if not exists conditions(cid INTEGER,
                        sky_condition TEXT,
                        explanation TEXT,
                        PRIMARY KEY (cid))
```

Hide

```
create table if not exists strikes(sid INTEGER,
                      fid INTEGER,
                      numbirds INTEGER,
                      impact TEXT,
                      damage BIT,
                      altitude INTEGER CHECK (altitude>=0),
                      conditions INTEGER,
                      FOREIGN KEY (fid) REFERENCES flights(fid),
                      FOREIGN KEY (conditions) REFERENCES conditions(cid))
```

##Populating Database

Hide

```
bds.raw = read.csv("BirdStrikesData-V2.csv", header=T)
bds.raw$heavy_flag <- ifelse(bds.raw$heavy_flag == "Yes", 1, 0)
bds.raw$flight_date <- as.Date(bds.raw$flight_date, format = "%m/%d/%Y")

n.airports <- nrow(bds.raw)
df.airports <- data.frame(aid = 100 + match(bds.raw$origin, unique(bds.raw$origin)),
                          airportState = bds.raw$origin)

n.flights <- nrow(bds.raw)
df.flights <- data.frame(fid = 100 + seq(1,n.flights),
                         dates = bds.raw$flight_date,
                         airline = bds.raw$airline,
                         origin = df.airports$aid,
                         aircraft = bds.raw$aircraft,
                         altitude = bds.raw$altitude_ft,
                         heavy = bds.raw$heavy_flag)

n.conditions <- nrow(bds.raw)
df.conditions <- data.frame(cid = 100 + match(bds.raw$sky_conditions,unique(bds.raw$s
ky_conditions)),
                            sky_condition = bds.raw$sky_conditions)

n.strikes <- nrow(bds.raw)
df.strikes <- data.frame(sid = 10 + seq(1,n.strikes),
                         fid = df.flights$fid,
                         numbirds = bds.raw$wildlife_struck,
                         damage = ifelse(bds.raw$damage=="Caused damage", TRUE, FALSE
),
                         altitude = bds.raw$altitude_ft,
                         conditions = df.conditions$cid)
```

##Setting FK PK Relations

Hide

```
for (r in 1:n.flights) {
  f <- df.airports$aid[which(df.airports$airportState == bds.raw$origin[r])]
  df.flights$origin[r] <- f
}

for (r in 1:n.strikes) {
  s <- df.conditions$cid[which(df.conditions$sky_condition==bds.raw$sky_conditions[r]
)]
  df.strikes$conditions[r] <- s
}

for (r in 2:n.strikes) {
  p <- df.flights$fid[which(df.flights$dates==bds.raw$flight_date[r] &
                            df.flights$origin==bds.raw$origin[r] &
                            df.flights$airline==bds.raw$airline[r] &
                            df.flights$aircraft==bds.raw$aircraft[r] &
                            df.flights$altitude==bds.raw$altitude_ft[r])]
  df.strikes$fid[r] <- p
}
```

##Writing to the Database

Hide

```
dbWriteTable(mydb, "airports", df.airports, overwrite = F, append = T, row.names=F)
dbWriteTable(mydb, "flights", df.flights, overwrite = F, append = T, row.names=F)
dbWriteTable(mydb, "conditions", df.conditions, overwrite = F, append = T, row.names=
F)
dbWriteTable(mydb, "strikes", df.strikes, overwrite = F, append = T, row.names=F)
```

##Viewing the tables in the Database

Hide

```
select * from flights;
```

Hide

```
select * from airports;
```

Hide

```
select * from conditions;
```

Hide

```
select * from strikes;
```

##Question 8

Hide

```
select airportState as States, count(*) as incidents from strikes left join flights o
n strikes.fid = flights.fid left join airports on flights.origin=airports.aid group b
y airportState order by incidents desc;
```

##Question 9

Hide

```
select f.airline, count(*) as num_incidents from strikes s left join flights f on
s.fid = f.fid group by f.airline having count(*) > (select avg(num_incidents) from (s
elect count(*) as num_incidents from strikes s left join flights f on s.fid = f.fid g
roup by f.airline) as subquery);
```

##Question 10

Hide

```
rs <- dbGetQuery(mydb, "select month(f.dates) as month, year(f.dates) as year, count(
*) as total_strikes
from strikes s left join flights f on s.fid = f.fid group by year(f.dates), month(f.d
ates)
order by year(f.dates), month(f.dates);")
rs
```

##Question 11

Hide

```
library(plotly)
fig = plot_ly(x = rs$month, y = rs$total_strikes, type="scatter", mode = "heatmap")
fig
plot(rs$year, rs$total_strikes, type = "l", xlab = "year", ylab = "Strikes", main = "
Number of Birds Striking Aircraft by Year")
plot(rs$month, rs$total_strikes, type = "l", xlab = "month", ylab = "Strikes", main =
"Number of Birds Striking Aircraft by Month")
```

##Question 12

Hide

```
DELIMITER //

CREATE PROCEDURE AddStrike(
   airportState TEXT,
   dates DATE,
   origin TEXT,
   airline TEXT,
   aircraft TEXT,
   conditions TEXT,
   num_strikes INT
)
BEGIN
    -- Insert the strike incident
    INSERT INTO airports
    VALUES (airportState);

    INSERT INTO flights
    VALUES (dates, origin, airline, aircraft);

    INSERT INTO conditions
    VALUES (conditions)

    INSERT INTO strikes
    VALUES (num_strikes)
)
END //

DELIMITER ;
```