Practicum I CS5200

Pramatha Bhat(bhat.pra@northeastern.edu) & Harshitha Prabhu(prabhu.h@northeastern.edu)

Spring 2023

Connect to Database

```
library(RMySQL)
```

```
## Loading required package: DBI
```

```
db_user <- 'root'
db_password <- 'Appleharshi123$'
db_name <- 'MYSQL'
db_host <- 'localhost'
db_port <- 3306
mydb <-  dbConnect(MySQL(), user = db_user, password = db_password,
                 dbname = db_name, host = db_host, port = db_port)
mydb
```

```
## <MySQLConnection:0,0>
```

Create Database

drop incidents table

```
DROP TABLE IF EXISTS incidents
```

Create incdidents table

```
CREATE TABLE incidents (
    rid INTEGER PRIMARY KEY,
    `dep.date` DATE,
    origin INTEGER,
    airline INTEGER,
    aircraft TEXT,
    `flight.phase` ENUM('takeoff', 'landing', 'inflight', 'unknown'),
    altitude INTEGER CHECK (altitude >= 0),
    conditions INTEGER,
    warned BOOLEAN
);
```

drop airports table

```
DROP TABLE IF EXISTS airports
```

Create airports table

```
CREATE TABLE airports (
    aid INTEGER PRIMARY KEY AUTO_INCREMENT,
    airportName TEXT,
    airportCode TEXT,
    state TEXT
);
```

Add origin foreign key to Incidents table

```
ALTER TABLE incidents
ADD FOREIGN KEY (origin) REFERENCES airports(aid);
```

drop conditions table

```
DROP TABLE IF EXISTS conditions
```

Create conditions table

```
CREATE TABLE conditions (
  cid INTEGER PRIMARY KEY AUTO_INCREMENT,
  `condition` TEXT,
  explanation TEXT
);
```

Add conditions foreign key to Incidents table

```
ALTER TABLE incidents
ADD FOREIGN KEY (conditions) REFERENCES conditions(cid);
```

Drop airlines table

```
DROP TABLE IF EXISTS airlines
```

Create airlines table

```
CREATE TABLE airlines (
    eid INTEGER PRIMARY KEY AUTO_INCREMENT,
    airlineName TEXT,
    airlineCode TEXT,
    flag TEXT
);
```

Add airline foreign key to Incidents table

```
ALTER TABLE incidents
ADD FOREIGN KEY (airline) REFERENCES airlines(eid);
```

display incident table structure

```
DESCRIBE incidents
```

Table 1: 9 records

Field	Type	Null	Key	Default	Extra
rid	int	NO	PRI	NA	
dep.date	date	YES		NA	
origin	int	YES	MUL	NA	
airline	int	YES	MUL	NA	
aircraft	text	YES		NA	
flight.phase	enum('takeoff','landing','inflight','unknown')	YES		NA	
altitude	int	YES		NA	
conditions	int	YES	MUL	NA	
warned	tinyint(1)	YES		NA	

display airports table structure

```
DESCRIBE airports
```

Table 2: 4 records

Field	Type	Null	Key	Default	Extra
aid	int	NO	PRI	NA	auto_increment
airportName	text	YES		NA	
airportCode	text	YES		NA	
state	text	YES		NA	

display conditions table structure

```
DESCRIBE conditions
```

Table 3: 3 records

Field	Type	Null	Key	Default	Extra
cid	int	NO	PRI	NA	auto_increment
condition	text	YES		NA	
explanation	text	YES		NA	

display airlines table structure

```
DESCRIBE airlines
```

Table 4: 4 records

Field	Type	Null	Key	Default	Extra
eid	int	NO	PRI	NA	auto_increment
airlineName	text	YES		NA	
airlineCode	text	YES		NA	

Field	Type	Null	Key	Default	Extra
flag	text	YES		NA	

Load BirdStrikesData.csv

```
library(readr)
bds.raw <- read_csv("BirdStrikesData-v2.csv",show_col_types = FALSE)
```

Get conditions from csv file

```
library(RMySQL)
library(sqldf)
```

```
## Loading required package: gsubfn

## Loading required package: proto

## Warning in doTryCatch(return(expr), name, parentenv, handler): unable to load shared object '/Library
##   dlopen(/Library/Frameworks/R.framework/Resources/modules//R_X11.so, 0x0006): Library not loaded: /
##   Referenced from: <05451E21-B5F6-3B2F-9C0F-3EA08D57DC34> /Library/Frameworks/R.framework/Versions/4
##   Reason: tried: '/opt/X11/lib/libSM.6.dylib' (no such file), '/System/Volumes/Preboot/Cryptexes/OS/

## tcltk DLL is linked to '/opt/X11/lib/libX11.6.dylib'

## Could not load tcltk.  Will use slower R code instead.

## Loading required package: RSQLite

##
## Attaching package: 'RSQLite'

## The following object is masked from 'package:RMySQL':
##
##     isIdCurrent

## sqldf will default to using MySQL
```

```
options(sqldf.driver = "SQLite")
conditions<-sqldf('select distinct sky_conditions FROM `bds.raw`')
conditions<-na.omit(conditions)
conditions[nrow(conditions)+1,]=c("unknown")
```

Get airport details from the csv file BirdStrikesData.csv

```
options(sqldf.driver = "SQLite")
airport_details<-sqldf('select distinct airport, origin FROM `bds.raw`')
airport_details<-na.omit(airport_details)
airport_details[nrow(airport_details)+1,]=c("unknown")
```

Get airline details from the csv file BirdStrikesData.csv

```
options(sqldf.driver = "SQLite")
airline_details<-sqldf('select distinct airline FROM `bds.raw`')
```

```
airline_details<-na.omit(airline_details)
airline_details[nrow(airline_details)+1,]=c("unknown")
```

get the flight details from the csv file BirdStrikesData.csv

```
options(sqldf.driver = "SQLite")
flight_details<-sqldf('select distinct flight_phase FROM `bds.raw`')
flight_details<-na.omit(flight_details)
flight_details[nrow(flight_details)+1,]=c("unknown")
```

get the incident details from the csv file BirdStrikesData.csv

```
options(sqldf.driver = "SQLite")
incident_details <- sqldf('select distinct rid, flight_date, origin, airline, aircraft, flight_phase, al
for (i in 1:nrow(incident_details)) {
if(is.na(incident_details[i,3])){
incident_details[i,3] = "unknown"
 }
if(is.na(incident_details[i,4])){
 incident_details[i,4] = "unknown"
 }
}
incident_details<-na.omit(incident_details)
```

insert rows from csv to airlines table

```
insert_airlines <- function() {

  for (i in 1:nrow(airline_details)) {
    airline<-paste0('"',airline_details[i,1],'"')
  cmd<-paste0('insert into airlines(airlineName) values (',airline,')')
      dbSendQuery(mydb,cmd)
  }
}
insert_airlines ()
```

displaying 10 rows of airlines table

```
select * from airlines limit 10;
```

Table 5: Displaying records 1 - 10

eid	airlineName	airlineCode	flag
1	US AIRWAYS*	NA	NA
2	AMERICAN AIRLINES	NA	NA
3	BUSINESS	NA	NA
4	ALASKA AIRLINES	NA	NA
5	COMAIR AIRLINES	NA	NA
6	UNITED AIRLINES	NA	NA
7	AIRTRAN AIRWAYS	NA	NA
8	AIRTOURS INTL	NA	NA

eid	airlineName	airlineCode	flag
9	AMERICA WEST AIRLINES	NA	NA
10	EXECUTIVE JET AVIATION	NA	NA

insert rows into conditions table

```
insert_conditions <- function() {

  for (i in 1:nrow(conditions)) {
    condition<-paste0('"',conditions[i,1],'"')
  cmd<-paste0('insert into conditions(`condition`) values (',condition,')')
      dbSendQuery(mydb,cmd)
  }

}
insert_conditions ()
```

displaying 10 rows of conditions table

```
select * from conditions limit 10;
```

Table 6: 4 records

cid	condition	explanation
1	No Cloud	NA
2	Some Cloud	NA
3	Overcast	NA
4	unknown	NA

insert into airports table

```
insert_airport <- function() {

  for (i in 1:nrow(airport_details)) {
    airport<-paste0('"',airport_details[i,1],'"')
     if(is.na(airport_details[i,2])) {
        state<-'NULL'
      } else {
      state<-paste0('"',airport_details[i,2],'"')
      }
  cmd<-paste0('insert into airports(airportName,state) values (',airport,',',state,')')
      dbSendQuery(mydb,cmd)
  }

}
insert_airport ()
```

displaying 10 rows of airports table

```
select * from airports limit 10;
```

Table 7: Displaying records 1 - 10

aid	airportName	airportCode	state
1	LAGUARDIA NY	NA	New York
2	DALLAS/FORT WORTH INTL ARPT	NA	Texas
3	LAKEFRONT AIRPORT	NA	Louisiana
4	SEATTLE-TACOMA INTL	NA	Washington
5	NORFOLK INTL	NA	Virginia
6	GUAYAQUIL/S BOLIVAR	NA	N/A
7	NEW CASTLE COUNTY	NA	Delaware
8	WASHINGTON DULLES INTL ARPT	NA	DC
9	ATLANTA INTL	NA	Georgia
10	ORLANDO SANFORD INTL AIRPORT	NA	Florida

insert rows into incidents table

```
library(RMySQL)
insert_conditions <- function() {
  for (i in 1:(nrow(incident_details)-1)) {
   id<-paste0('"',incident_details[i,1],'"')
    query <- paste0('select aid from airports where state=','"',incident_details[i,3],'"')
    airport_result <- dbSendQuery(mydb, query)
    airport_data <- dbFetch(airport_result)
    airport <- airport_data[1, 1]
    date<-incident_details[i,2]

    if(is.na(date))
      {
      date<-'NULL'
      } else
        {
        date<-strsplit(date,"/")
        date<-paste(substr(date[[1]][[3]],1,4),date[[1]][[1]],date[[1]][[2]], sep = "-")
        date<-paste0('"',date,'"')
      }
        query<-paste0('select eid from airlines where airlineName=','"',incident_details[i,4],'"')
        airlines_result <- dbSendQuery(mydb, query)
    airlines_result <- dbFetch(airlines_result)
    airline <- airlines_result[1, 1]

query<-paste0('select cid from conditions where `condition`=','"',incident_details[i,8],'"')
        conditions_result <- dbSendQuery(mydb, query)
    conditions_result <- dbFetch(conditions_result)
    condition <- conditions_result[1, 1]

      if(is.na(incident_details[i,5])){
         aircraft<-'NULL'
      }else
        {
```

```
      aircraft<-paste0('"',incident_details[i,5],'"')}

    if(incident_details[i,9] == "Yes" ){
        incident_boolean = 1
      }else
        {
          incident_boolean = 0
        }
    if(incident_details[i,6] == "Take-off run" ||  incident_details[i,6] == "Climb"){
      flight_phase = "\'takeoff\'"
    }
    else if(incident_details[i,6] == "Landing Roll" || incident_details[i,6] == "Descent"){
      flight_phase = "\'landing\'"
    }
    else if(incident_details[i,6] == "Approach" || incident_details[i,6] == "Taxi"|| incident_details[i
      flight_phase = "\'inflight\'"
    }
    else{
      flight_phase = "\'unknown\'"
    }
      cmd<-paste0('insert into incidents(rid, `dep.date`, origin, airline, aircraft, `flight.phase`, al
    dbSendQuery(mydb, cmd)

  }
}
insert_conditions()
```

display 10 rows of incidents table

```
select * from incidents limit 10;
```

Table 8: Displaying records 1 - 10

rid	dep.date	origin	airline	aircraft	flight.phase	altitude	conditions	warned
1195	2002-11-13	3	21	Airplane	inflight	2000	3	0
3019	2002-10-10	11	21	Airplane	takeoff	400	1	0
3500	2001-05-15	3	21	Airplane	inflight	1000	1	0
3504	2001-05-23	3	21	Airplane	inflight	1800	1	0
3597	2001-04-18	2	21	Airplane	inflight	200	2	0
4064	2000-04-06	3	21	Airplane	inflight	1000	1	0
4074	2002-07-15	10	21	Airplane	takeoff	0	1	0
4076	2002-07-15	3	21	Airplane	takeoff	500	2	0
4090	2001-07-02	20	21	Airplane	takeoff	50	2	0
4091	2001-07-07	20	21	Airplane	takeoff	0	2	0

finding the 10 states with greatest number of bird strike incidents

```
SELECT origin, COUNT(rid) as count FROM incidents GROUP BY origin ORDER BY count DESC LIMIT 10;
```

Table 9: Displaying records 1 - 10

origin	count
11	2499
2	2445
10	2045
1	1316
12	1006
52	985
14	956
44	806
34	773
17	716

finding the airlines that have an above average number bird strike incidents.

```
SELECT a.airlineName, COUNT(*) AS num_incidents
FROM incidents i
JOIN airlines a ON i.airline = a.eid
GROUP BY a.airlineName
HAVING COUNT(*) > (
    SELECT AVG(num_incidents)
    FROM (
        SELECT COUNT(*) AS num_incidents
        FROM incidents
        GROUP BY airline
    ) AS avg_incidents
)
```

Table 10: Displaying records 1 - 10

airlineName	num_incidents
US AIRWAYS*	797
AMERICAN AIRLINES	2058
BUSINESS	3074
ALASKA AIRLINES	304
COMAIR AIRLINES	317
UNITED AIRLINES	506
AIRTRAN AIRWAYS	414
AMERICA WEST AIRLINES	157
HAWAIIAN AIR	332
DELTA AIR LINES	1349

finding the number of bird strike incidents by month and by flight phase

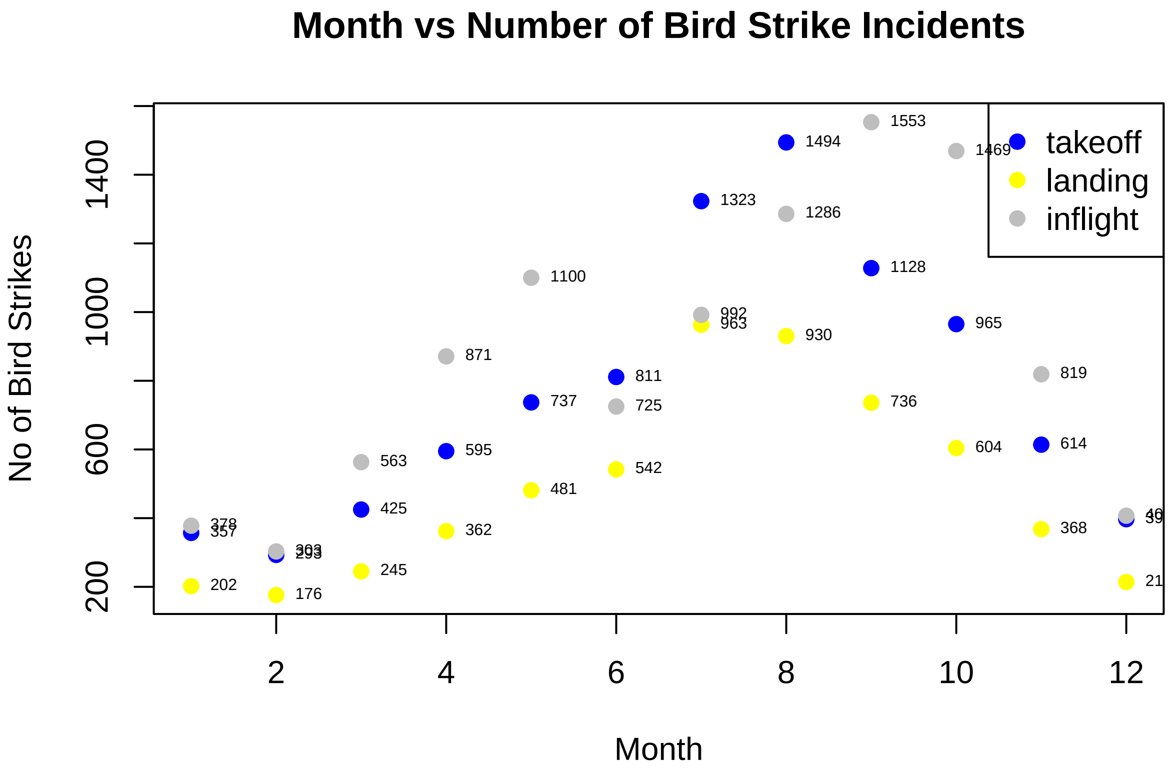
```
cmd<-'SELECT distinct MONTH(`dep.date`) AS month, `flight.phase` as flight_phase, COUNT(*) AS num_incid
numOfIncidentsByMonth <- dbGetQuery(mydb, cmd)
head(numOfIncidentsByMonth, 6)
```

```
##   month flight_phase num_incidents
## 1     1      takeoff           357
## 2      1       landing          202
## 3      1      inflight          378
## 4      2       takeoff          293
## 5      2       landing          176
## 6      2      inflight          303
```

Building a scatter plot that plots month along the x-axis versus number of incidents

```
plot(x = numOfIncidentsByMonth$month,
     y = numOfIncidentsByMonth$num_incidents,
     xlab = "Month", ylab = "No of Bird Strikes",
     main = "Month vs Number of Bird Strike Incidents",
     col = c("blue","yellow","grey"), pch=19)

text(numOfIncidentsByMonth$month,
     numOfIncidentsByMonth$num_incidents,
     labels=numOfIncidentsByMonth$num_incidents,
     cex = 0.5, pos = 4)
legend(x="topright", legend=unique(numOfIncidentsByMonth$flight_phase),
       col = c("blue","yellow","grey"), pch=19)
```



Drop the procedure if it already exists

```
DROP PROCEDURE IF EXISTS addNewIncident;
```

Stored procedure to create new incident

```
CREATE PROCEDURE addNewIncident (
  IN rid INTEGER,
  IN dep_date DATE,
  IN origin INTEGER,
  IN airline_name TEXT,
  IN airline_code TEXT,
  IN aircraft TEXT,
  IN flight_phase ENUM('takeoff', 'landing', 'inflight', 'unknown'),
  IN altitude INTEGER,
  IN `condition` TEXT,
  IN explanation TEXT,
  IN airport_name TEXT,
  IN airport_code TEXT,
  IN state TEXT,
  IN warned BOOLEAN
)
BEGIN
  DECLARE airline_id INTEGER;
  DECLARE airport_id INTEGER;
  DECLARE condition_id INTEGER;

  SELECT eid INTO airline_id FROM airlines WHERE airlineName = airline_name AND airlineCode = airline_c

  IF airline_id IS NULL THEN
    INSERT INTO airlines (airlineName, airlineCode) VALUES (airline_name, airline_code);
    SET airline_id = LAST_INSERT_ID();
  END IF;

  SELECT aid INTO airport_id FROM airports WHERE airportCode = airport_code limit 1;

  IF airport_id IS NULL THEN
    INSERT INTO airports (airportName, airportCode, state) VALUES (airport_name, airport_code, state);
    SET airport_id = LAST_INSERT_ID();
  END IF;

  SELECT cid INTO condition_id FROM conditions WHERE `condition` = `condition` limit 1;

  IF condition_id IS NULL THEN
    INSERT INTO conditions (`condition`, explanation) VALUES (`condition`, explanation);
    SET condition_id = LAST_INSERT_ID();
  END IF;

  INSERT INTO incidents (rid, `dep.date`, origin, airline, aircraft, `flight.phase`, altitude, conditio
  VALUES (rid, dep_date, origin, airline_id, aircraft, flight_phase, altitude, condition_id, warned);

END
```

Call addNewIncident Procedure with test values

```
CALL addNewIncident(319594,'2022-03-03', 123, 'Delta Airlines', 'DL', 'Boeing 737', 'takeoff', 10000, '
```

Testing newly added row by the procedure.

```
result <- dbGetQuery(mydb, "SELECT * FROM incidents WHERE rid=319594")
print(result)
```

```
##      rid   dep.date origin airline   aircraft flight.phase altitude conditions
## 1 319594 2022-03-03    123     294 Boeing 737      takeoff    10000          1
##   warned
## 1      1
```

```
dbDisconnect(mydb)
```

```
## [1] TRUE
```