Practicum I CS5200

Gowreesh Gunupati, Vasumathi Narayanan

Spring 2023

Question 3 - Connect to Database

We created a new database in MySQL using command "CREATE DATABASE birdstrikes" and we connected the MySQL server to R using workbench. We are using library lubridate for parsing and manipulating the date.

```
library(RMySQL)
```

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

```
library(sqldf)
```

```
## Loading required package: gsubfn

## Loading required package: proto

## Loading required package: RSQLite

##
## Attaching package: 'RSQLite'

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

## sqldf will default to using MySQL
```

```
library(lubridate)
```

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

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

```
birdStrikesDBCon = dbConnect(RMySQL::MySQL(),
                             dbname='birdstrikes',
                             host='localhost',
                             port=3306,
                             user='ecommerceapp',
                             password='ecommerceapp')
```

setting local_infile to True

```
#show global variables like 'local_infile';
SET global local_infile = TRUE;
```

This is loaded to use sqldf for loading the data frame with respective columns using SQLite.

```
options(sqldf.driver = "SQLite")
```

Drop If Tables already Exist

```
DROP TABLE IF EXISTS incidents;
```

```
DROP TABLE IF EXISTS airports;
```

```
DROP TABLE IF EXISTS conditions;
```

```
DROP TABLE IF EXISTS airlines;
```

We have created the airports table with the appropriate attributes and the constraints are mentioned as below. aid is a synthetic primary key, so we have added AUTO_INCREMENT for that.

```
CREATE TABLE airports (
  aid INT AUTO_INCREMENT,
  airportName VARCHAR(100) ,
  airportCode VARCHAR(10),
  state VARCHAR(100),
  PRIMARY KEY(aid)
  );
```

We have created the conditions table with the appropriate attributes and the constraints are mentioned as below. cid is the primary key here.

```
CREATE TABLE conditions (
cid INT(10) AUTO_INCREMENT,
skyCondition VARCHAR(50),
explaination VARCHAR(50),
PRIMARY KEY(cid)
);
```

We have created the airlines table with the appropriate attributes and the constraints are mentioned as below. “eid” is the synthetic primary key so we have added AUTO_INCREMENT.

```
CREATE TABLE airlines (
eid INT AUTO_INCREMENT,
airlineName VARCHAR(50),
airlineCode VARCHAR(50),
flag VARCHAR(10),
PRIMARY KEY(eid)
);
```

We have created the incidents table with the appropriate attributes and the constraints are mentioned as below. origin is the foreign key that is referenced to aid conditions is the foreign key that is referenced to conditions airline is the foreign key that is reference to airlines

```
CREATE TABLE incidents(
  rid INT,
  depDate DATE,
  origin INT NOT NULL ,
  airline INT NOT NULL,
  aircraft VARCHAR(50),
  flightPhase VARCHAR(20),
  altitude INT,
  conditions INT NOT NULL,
  warned TINYINT,
  PRIMARY KEY (rid),
  CONSTRAINT CHK_altitude CHECK (altitude>=0),
  FOREIGN KEY (origin) REFERENCES airports(aid),
  FOREIGN KEY (conditions) REFERENCES conditions(cid),
  FOREIGN KEY (airline) REFERENCES airlines(eid),
  CONSTRAINT CHK_flightPhase CHECK
  (flightPhase = 'takeoff' OR
  flightPhase='landing' OR
  flightPhase='inflight' OR
  flightPhase ='unknown')
  );
```

Verifying Schemas for birdstrikes

```
DESC  incidents;
```

Table 1: 9 records

Field	Type	Null	Key	Default	Extra
rid	int	NO	PRI	NA	
depDate	date	YES		NA	
origin	int	NO	MUL	NA	
airline	int	NO	MUL	NA	
aircraft	varchar(50)	YES		NA	
flightPhase	varchar(20)	YES		NA	
altitude	int	YES		NA	

Field	Type	Null	Key	Default	Extra
conditions	int	NO	MUL	NA	
warned	tinyint	YES		NA	

```
DESC airports;
```

Table 2: 4 records

Field	Type	Null	Key	Default	Extra
aid	int	NO	PRI	NA	auto_increment
airportName	varchar(100)	YES		NA	
airportCode	varchar(10)	YES		NA	
state	varchar(100)	YES		NA	

```
DESC conditions;
```

Table 3: 3 records

Field	Type	Null	Key	Default	Extra
cid	int	NO	PRI	NA	auto_increment
skyCondition	varchar(50)	YES		NA	
explaination	varchar(50)	YES		NA	

```
DESC airlines;
```

Table 4: 4 records

Field	Type	Null	Key	Default	Extra
eid	int	NO	PRI	NA	auto_increment
airlineName	varchar(50)	YES		NA	
airlineCode	varchar(50)	YES		NA	
flag	varchar(10)	YES		NA	

Question 5 Loading CSV to R Dataframe

```
bds.raw<- read.csv("BirdStrikesData-V2.csv", header = TRUE, sep = ",")
head(bds.raw)
```

```
##      rid aircraft                    airport        model wildlife_struck
## 1 202152 Airplane                LAGUARDIA NY    B-737-400             859
## 2 208159 Airplane DALLAS/FORT WORTH INTL ARPT        MD-80             424
## 3 207601 Airplane           LAKEFRONT AIRPORT        C-500             261
## 4 215953 Airplane         SEATTLE-TACOMA INTL    B-737-400             806
## 5 219878 Airplane                NORFOLK INTL CL-RJ100/200             942
## 6 218432 Airplane         GUAYAQUIL/S BOLIVAR        A-300             537
```

```
##                  impact     flight_date        damage           airline
## 1      Engine Shut Down 11/23/2000 0:00 Caused damage       US AIRWAYS*
## 2                  None  7/25/2001 0:00 Caused damage AMERICAN AIRLINES
## 3                  None  9/14/2001 0:00     No damage          BUSINESS
## 4 Precautionary Landing   9/5/2002 0:00     No damage   ALASKA AIRLINES
## 5                  None  6/23/2003 0:00     No damage   COMAIR AIRLINES
## 6                  None  7/24/2003 0:00     No damage AMERICAN AIRLINES
##       origin flight_phase remains_collected_flag
## 1   New York        Climb                 FALSE
## 2      Texas Landing Roll                 FALSE
## 3  Louisiana     Approach                 FALSE
## 4 Washington        Climb                  TRUE
## 5   Virginia     Approach                 FALSE
## 6        N/A Take-off run                 FALSE
##
## 1  FLT 753. PILOT REPTD A HUNDRED BIRDS ON UNKN TYPE. #1 ENG WAS SHUT DOWN AND DIVERTED TO EWR. SLIG
## 2
## 3
## 4 NOTAM WARNING. 26 BIRDS HIT THE A/C, FORCING AN EMERGENCY LDG. 77 BIRDS WERE FOUND DEAD ON RWY/TWY
## 5
## 6
##   wildlife_size sky_conditions               species pilot_warned_flag
## 1        Medium       No Cloud Unknown bird - medium                 N
## 2         Small     Some Cloud           Rock pigeon                 Y
## 3         Small       No Cloud     European starling                 N
## 4         Small     Some Cloud     European starling                 Y
## 5         Small       No Cloud     European starling                 N
## 6         Small       No Cloud  Unknown bird - small                 N
##   altitude_ft heavy_flag
## 1       1,500        Yes
## 2           0         No
## 3          50         No
## 4          50        Yes
## 5          50         No
## 6           0         No
```

Data Cleaning and pre-processing to format the date

1. We are normalizing the flight phases to have only values takeoff, landing, inflight,unknown
2. The flight_date field was transformed into %Y-%m-%d
3. Changed all the flights, airline, aircrafts that holds the value as empty string to unknown

Parsing Date

```
bds.raw$flight_date<- parse_date_time(bds.raw$flight_date, orders=c("m-d-y H:M","m/d/y H:M"))
bds.raw$flight_date <- format(bds.raw$flight_date,"%Y-%m-%d")
```

Data cleaning to rename empty values to unknown

```
case5 <- which(bds.raw$airline==''| bds.raw$airline==' '|
                bds.raw$airline=='NA' | bds.raw$airline=='N/A')
bds.raw[case5,"airline"] <- 'unknown'

case6 <- which(bds.raw$aircraft==''|bds.raw$aircraft==' '|
                bds.raw$aircraft=='NA'| bds.raw$aircraft=='N/A' )
bds.raw[case5,"aircraft"] <- 'unknown'
```

Harmonizing the flight phases to be one of takeoff, landing, inflight, unknown.

```
case1 <- which(bds.raw$flight_phase=='Take-off run'
               | bds.raw$flight_phase=='Climb' )
bds.raw[case1,"flight_phase"] <- 'takeoff'

case2 <- which(bds.raw$flight_phase=='Landing Roll'
               |bds.raw$flight_phase=='Taxi'|bds.raw$flight_phase=='Parked')
bds.raw[case2,"flight_phase"] <- 'landing'

case3 <-which(bds.raw$flight_phase=='Approach' | bds.raw$flight_phase=='Descent')
bds.raw[case3,"flight_phase"] = 'inflight'

case4 <- which(bds.raw$flight_phase=='' | bds.raw$flight_phase=='N/A'
               | bds.raw$flight_phase=='NA' | bds.raw$flight_phase==' ')
bds.raw[case4,"flight_phase"] <- 'unknown'
```

Change the warned flag as tinyint datatype based on pilot_warned_flag

```
warned <- which(bds.raw$pilot_warned_flag=="Y")
notwarned <- which(bds.raw$pilot_warned_flag=='N')

bds.raw[warned,"pilot_warned_flag"] <- 1
bds.raw[notwarned,"pilot_warned_flag"] <- 0
```

setting default sentinel values for model, origin and airport

```
case8 <- which(bds.raw$airport==''| bds.raw$airport==' '
               | bds.raw$airport=='NA' |bds.raw$airport=='N/A')
bds.raw[case8,"airport"]<- 'unknown'
case9 <- which(bds.raw$model==''  | bds.raw$model==' '
               |bds.raw$model=='NA' |bds.raw$model=='N/A')
bds.raw[case9,"model"]<- 'unknown'
case10<- which(bds.raw$origin=='NA'| bds.raw$origin==''
               |bds.raw$origin==' '| bds.raw$origin=='N/A')
bds.raw[case10,"origin"]<-'unknown'
```

omit flights without flight information

```
bds.raw <- bds.raw[which(bds.raw$airport!='' & bds.raw$model!='' & bds.raw$origin!='' ),]
```

Question 6 Loading R Dataframe to bird-strikes-DB

Inserting into conditions table

```
conditionsdata <- sqldf::sqldf("SELECT distinct(sky_conditions)
                               AS skyCondition FROM [bds.raw]");
dbWriteTable(birdStrikesDBCon, "conditions", conditionsdata,
             append = TRUE, row.names = FALSE)
```

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

```
n.conditionsdata <- nrow(conditionsdata)
conditionsdata$cid <- seq(1, n.conditionsdata)
```

```
head(conditionsdata)
```

Inserting into airlines table

```
airlinesData <- sqldf::sqldf("SELECT DISTINCT(TRIM(airline)) AS airlineName,
                             pilot_warned_flag AS flag  from [bds.raw]");
dbWriteTable(birdStrikesDBCon, "airlines", airlinesData, append = TRUE, row.names = FALSE)
```

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

```
n.airlinesData <- nrow(airlinesData)
airlinesData$eid <- seq(1, n.airlinesData)
```

```
head(airlinesData)
```

Inserting into airport table

```
airportsdata <- sqldf::sqldf("SELECT DISTINCT(trim(airport)) AS airportName,
                             origin AS state from [bds.raw]")
dbWriteTable(birdStrikesDBCon, "airports", airportsdata, append = TRUE, row.names = FALSE)
```

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

```
n.airportsdata <- nrow(airportsdata)
airportsdata$aid <- seq(1, n.airportsdata)
```

```
head(airportsdata)
```

Inserting into incidents table

```
incidentsdata <- sqldf::sqldf("SELECT rid, flight_date as depDate, origin,
                              airport, airline, aircraft,flight_phase as flightPhase,
                              altitude_ft as altitude, sky_conditions as conditions,
                              pilot_warned_flag as warned FROM [bds.raw]");
n.incidentsdata <- nrow(incidentsdata)
for(r in 1:n.incidentsdata)
{

  a <- airportsdata$aid[which(airportsdata$airportName == incidentsdata$airport[r])]
  incidentsdata$origin[r] <- a[[1]][1]

  b <- conditionsdata$cid[which(conditionsdata$skyCondition == incidentsdata$conditions[r])]
  incidentsdata$conditions[r] <- b[[1]][1]

  c<-airlinesData$eid[which(airlinesData$airlineName == incidentsdata$airline[r])]
  incidentsdata$airline[r] <- c[[1]][1]

}

incidentsdata <- sqldf::sqldf("SELECT rid, depDate , origin, airline,
                              aircraft,flightPhase, altitude, conditions,
                              warned FROM incidentsdata");

dbWriteTable(birdStrikesDBCon, "incidents", incidentsdata, append = TRUE, row.names = FALSE)
```

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

```
head(incidentsdata)
```

1.Since birdstrikes dataframe is too large, we are splitting it into airportsdata, airlinesData, incidentsdata and conditionsdata in order to make the data easier to handle. 2.We have then proceeded to populate the dataframes into the respective sql tables.

Question 7 Verifying the table populations

```
SELECT * from conditions limit 5;
```

Table 5: 3 records

cid	skyCondition	explaination
1	No Cloud	NA
2	Some Cloud	NA
3	Overcast	NA

```
SELECT * from airports limit 5;
```

Table 6: 5 records

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

```
SELECT * from airlines limit 5;
```

Table 7: 5 records

eid	airlineName	airlineCode	flag
1	US AIRWAYS*	NA	0
2	AMERICAN AIRLINES	NA	1
3	BUSINESS	NA	0
4	ALASKA AIRLINES	NA	1
5	COMAIR AIRLINES	NA	0

```
SELECT * from incidents limit 5;
```

Table 8: 5 records

rid	depDate	origin	airline	aircraft	flightPhase	altitude	conditions	warned
1195	2002-11-13	37	26	Airplane	inflight	2	3	1
3019	2002-10-10	717	26	Airplane	takeoff	400	1	1
3500	2001-05-15	37	26	Airplane	inflight	1	1	1
3504	2001-05-23	37	26	Airplane	inflight	1	1	1
3597	2001-04-18	123	26	Airplane	inflight	200	2	1

Test query to show all the foreign key constraints work properly after data loaded into their respective Tables
Test query

```
SELECT incidents.rid,incidents.depDate, airports.airportName AS origin,
airlines.airlineName AS airline, incidents.aircraft,
incidents.flightPhase, incidents.altitude,
conditions.skyCondition AS conditions, incidents.warned
FROM incidents,airports, airlines,conditions
WHERE rid = 1195
AND airports.aid = incidents.origin
AND airlines.eid = incidents.airline
AND conditions.cid= incidents.conditions
```

Table 9: 1 records

rid	depDate	origin	airline	aircraft	flightPhase	altitude	conditions	warned
1195	2002-11-13	BARKSDALE AIR FORCE BASE ARPT	MILITARY	Airplane	inflight	2	Overcast	1

This test query is to show that airline has UNKNOWN value in its column whose incident rid is 263834

```
SELECT incidents.rid,incidents.depDate, airports.airportName AS origin,
airlines.airlineName AS airline, incidents.aircraft,
incidents.flightPhase, incidents.altitude,
conditions.skyCondition AS conditions, incidents.warned
FROM incidents,airports, airlines,conditions
WHERE rid=263834
AND airports.aid = incidents.origin
AND airlines.eid = incidents.airline
AND conditions.cid= incidents.conditions
```

Table 10: 1 records

rid	depDate	origin	airline	aircraft	flightPhase	altitude	conditions	warned
263834	2009-08-23	ST. PAUL DOWNTOWN ARPT	UNKNOWN	Airplane	landing	0	No Cloud	0

Question 8

```
SELECT airports.state, COUNT(incidents.rid) AS "Number of Incidents"
FROM incidents,airports
WHERE airports.aid= incidents.origin
GROUP BY (airports.state)
ORDER BY COUNT(incidents.rid)
DESC LIMIT 10;
```

Table 11: Displaying records 1 - 10

state	Number of Incidents
California	2499
Texas	2445
Florida	2045
New York	1316
Illinois	1007
Pennsylvania	985
Missouri	956
Kentucky	806
Ohio	773
Hawaii	716

Question 9

```
SELECT DISTINCT  airlines.airlineName, COUNT(incidents.rid) AS "num_incidents"
FROM incidents,airlines
WHERE airlines.eid= incidents.airline
GROUP BY (airlines.airlineName) HAVING num_incidents>
(SELECT AVG(num_incidents) FROM (SELECT DISTINCT  airlines.airlineName,
COUNT(incidents.rid) AS "num_incidents"
FROM incidents,airlines
WHERE airlines.eid= incidents.airline
GROUP BY airlines.airlineName) AS average);
```

Table 12: 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

```
no_of_incidents_group_by_month<- dbGetQuery(birdStrikesDBCon,"SELECT DISTINCT MONTH(depDate) AS months
                                          COUNT(rid) as num_of_incidents_by_month FROM incidents
                                          GROUP BY MONTH(depDate) ORDER BY  MONTH(depDate)")
no_of_incidents_group_by_flightphase <-dbGetQuery(birdStrikesDBCon,"SELECT DISTINCT incidents.flightPhas
```

```
head(no_of_incidents_group_by_month,6)
```

```
##   months num_of_incidents_by_month
## 1     NA                      129
## 2      1                      937
## 3      2                      772
## 4      3                     1233
## 5      4                     1828
## 6      5                     2318
```

```
head(no_of_incidents_group_by_flightphase,6)
```

```
##   flightPhase num_of_incidents_by_flightphase
## 1    inflight                           11158
## 2     takeoff                            9140
## 3     landing                            5131
## 4     unknown                             129
```

Question 10

```
no_of_incidents_group_by_month_flightPhase<-
  dbGetQuery(birdStrikesDBCon,"select distinct flightPhase,MONTH(depDate) as months ,
             COUNT(rid) as num_of_incidents_by_month
             from incidents group by MONTH(depDate),
             flightPhase ORDER BY  MONTH(depDate)")
no_of_incidents_group_by_month_flightPhase=no_of_incidents_group_by_month_flightPhase[which(no_of_incid
head(no_of_incidents_group_by_month_flightPhase,6)
```

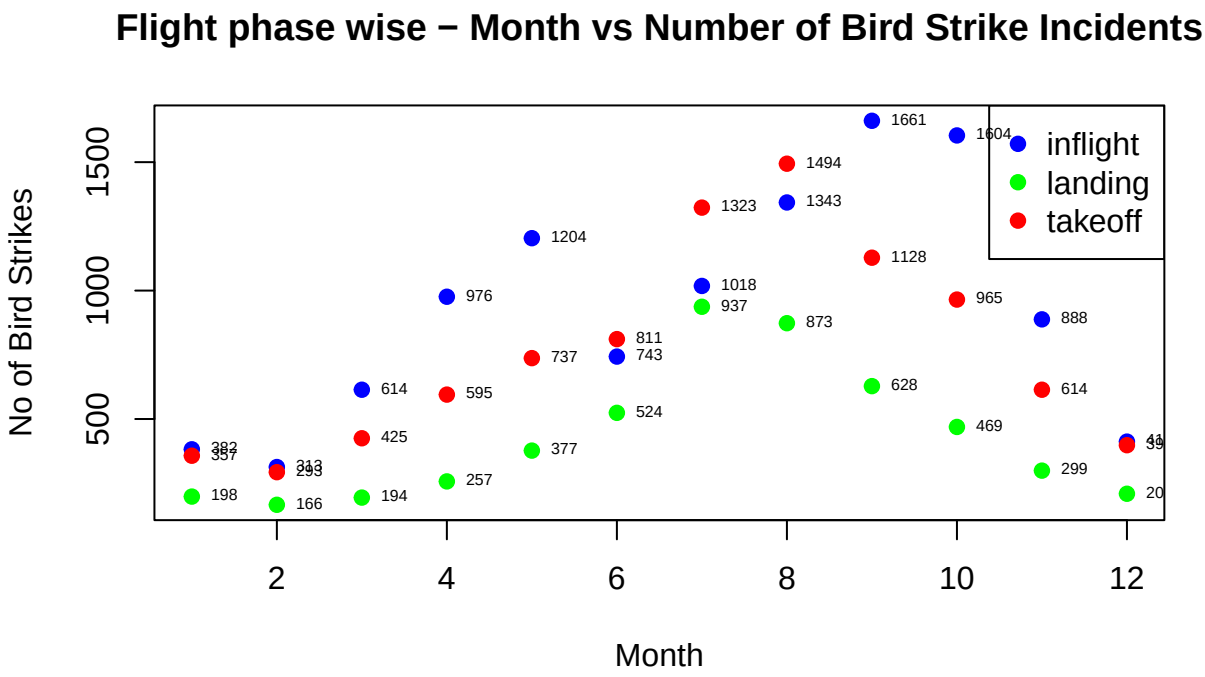
```
##   flightPhase months num_of_incidents_by_month
## 2    inflight      1                       382
## 3     landing      1                       198
## 4     takeoff      1                       357
## 5    inflight      2                       313
## 6     landing      2                       166
## 7     takeoff      2                       293
```

Question 11

```
plot(
  x = no_of_incidents_group_by_month_flightPhase$months,y =
       no_of_incidents_group_by_month_flightPhase$num_of_incidents_by_month
,
   xlab = "Month",
   ylab = "No of Bird Strikes",
   main = "Flight phase wise - Month vs Number of Bird Strike Incidents",
   col =  c("blue","green","red"),
   pch=19)

text(no_of_incidents_group_by_month_flightPhase$months,
     no_of_incidents_group_by_month_flightPhase$num_of_incidents_by_month, labels=no_of_incidents_group
     cex = 0.5, pos = 4)

legend(x = "topright", legend=unique(no_of_incidents_group_by_month_flightPhase$flightPhase),
       col = c("blue","green","red"), pch=19 )
```



Question 12

```
drop procedure if exists add_incident;
```

```
create procedure add_incident(
in airport1 text,
in state1 text,
in date1 date,
in airline1 text,
in aircraft1 text,
in flightPhase1 text,
in altitude1 int,
in condition1 text,
in warning1 tinyint )

begin

declare cid1 int;
declare aid1 int default 0;
declare rid1 int;
declare eid1 int;

SELECT cid INTO cid1 FROM conditions WHERE conditions.skyCondition = condition1 ;

SELECT aid INTO aid1 FROM airports WHERE airportName = airport1 AND state = state1 ;

SELECT eid INTO eid1 FROM airlines WHERE airlineName= airline1 and flag=warning1;
```

```
SELECT (MAX(rid) + 1) INTO rid1 FROM incidents ;

IF aid1 = 0 THEN
SELECT (MAX(aid) + 1) INTO aid1 FROM airports ;
  INSERT INTO airports(aid, airportName, state) VALUES ( aid1, airport1, state1 ) ;
  END IF;

INSERT INTO incidents(rid, depDate,origin, airline, aircraft,
flightPhase, altitude, conditions, warned)
VALUES
( rid1, date1, aid1, eid1, aircraft1, flightPhase1, altitude1, cid1, warning1 ) ;
end
```

```
CALL add_incident(
'Newyork Airport',
'New York',
'2022-01-01',
'BUSINESS',
'Airplane',
'takeoff',
363,
'Overcast',
0);
```

```
SELECT *FROM incidents WHERE depDate = '2022-01-01'
```

Table 13: 1 records

rid	depDate	origin	airline	aircraft	flightPhase	altitude	conditions	warned
321910	2022-01-01	1142	3	Airplane	takeoff	363	3	0

From the below query we can see that the foreign key is used to retrieve the actual origin name, airline name and sky conditions by mapping the id. This is same as what we gave as the input for the stored procedure.

```
SELECT incidents.rid,incidents.depDate, airports.airportName AS origin,
airlines.airlineName AS airline, incidents.aircraft,
incidents.flightPhase, incidents.altitude,
conditions.skyCondition AS conditions, incidents.warned
FROM incidents,airports, airlines,conditions
WHERE depDate = '2022-01-01'
AND airports.aid = incidents.origin
AND airlines.eid = incidents.airline
AND conditions.cid= incidents.conditions
```

Table 14: 1 records

rid	depDate	origin	airline	aircraft	flightPhase	altitude	conditions	warned
321910	2022-01-01	Newyork Airport	BUSINESS	Airplane	takeoff	363	Overcast	0

```
SELECT * FROM airports WHERE airportName = 'Newyork Airport'
```

Table 15: 1 records

aid	airportName	airportCode	state
1142	Newyork Airport	NA	New York

Disconnecting database

```
dbDisconnect(birdStrikesDBCon)
```

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