

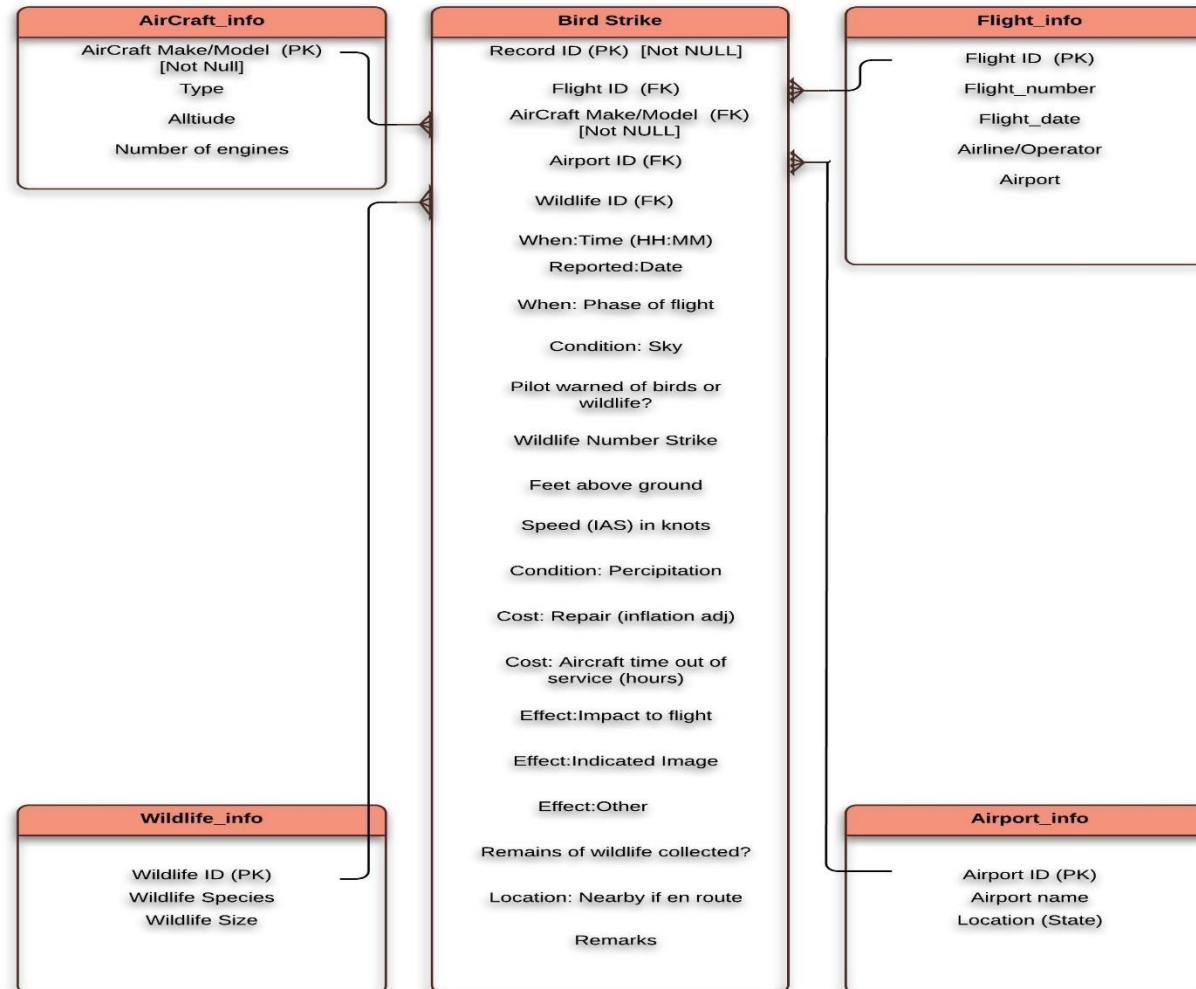
Bird-Strike Database Design

Mohsen Nabian

8/8/2015

Bird-Strike Database Structure Design

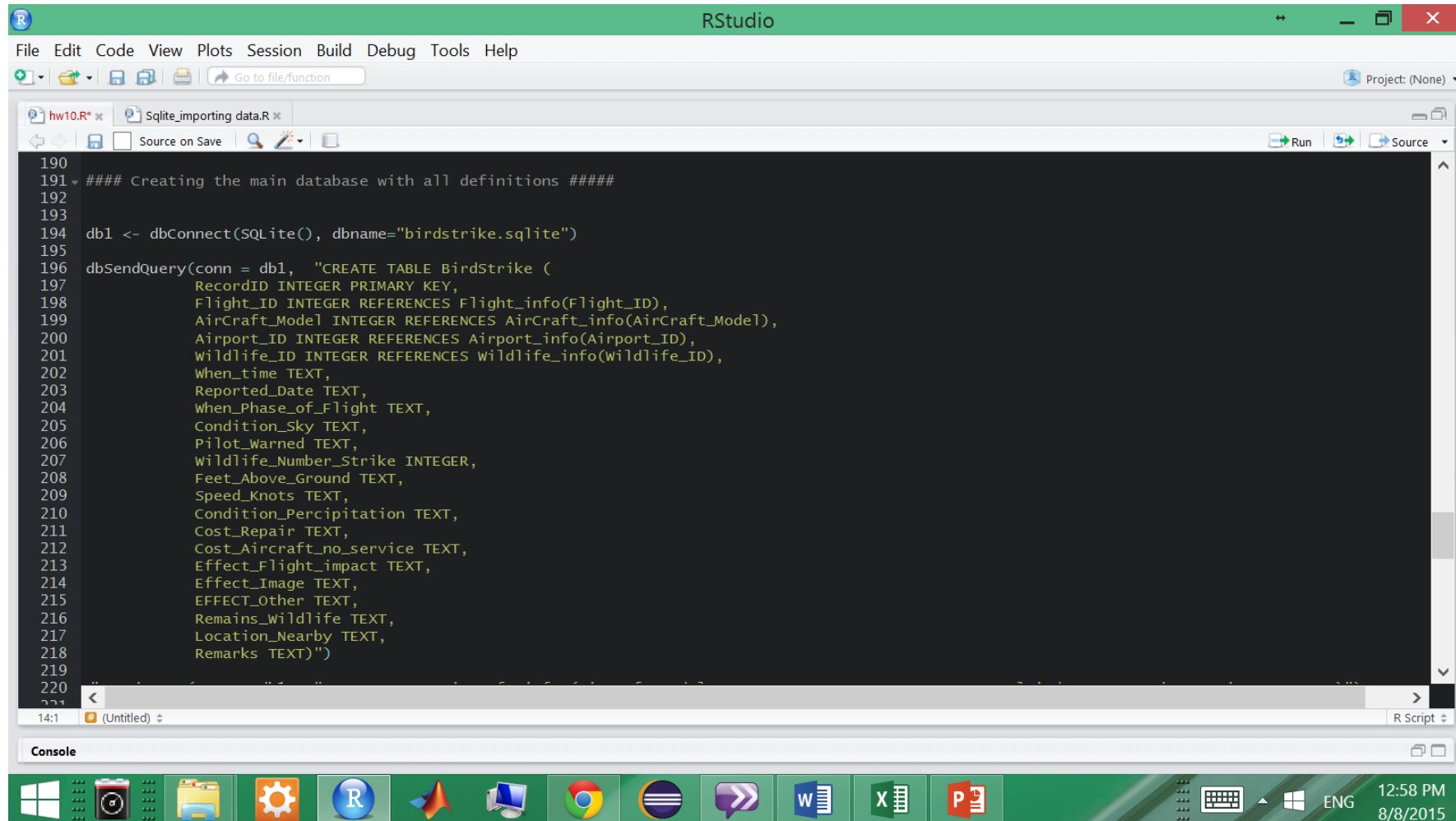
By: Mohsen
Nabian



Data Frames Prepration

- After designing the normalized tables it is time to prepare data :
- 1) we need to create data frames for different tables
- 2) remove repetitive rows from non_based tables
- 3) putting id for those non_based tables as primary key
- 4) finding the corresponding rows in base tables from non_based tables.
- 5) put the corresponding ids in the foreign key columns in the base table

Creating Database and defining the tables in the database

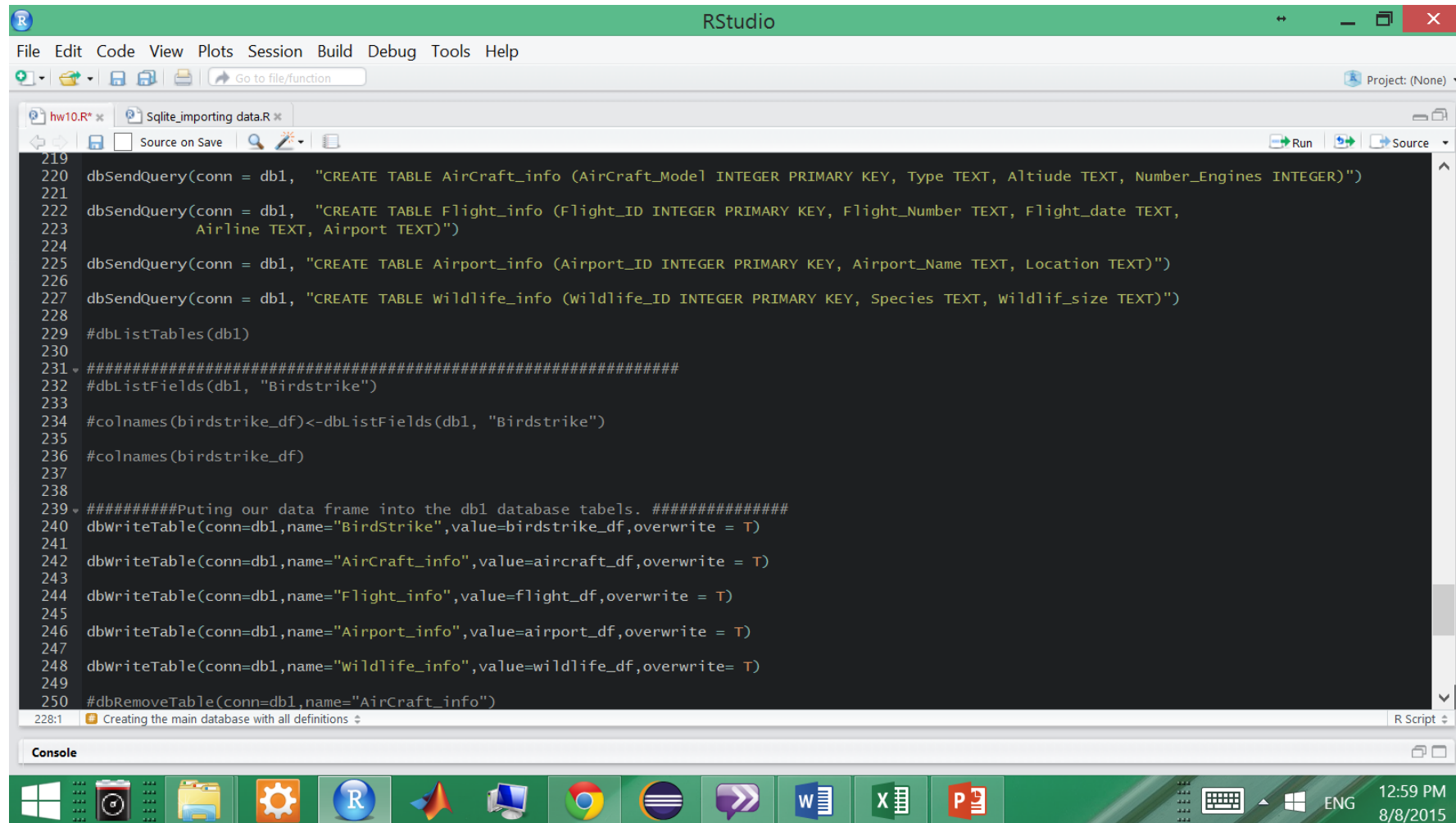


The screenshot shows the RStudio interface with a script editor containing R code to create a SQLite database and a table. The code is as follows:

```
190  
191 ##### Creating the main database with all definitions #####  
192  
193  
194 db1 <- dbConnect(SQLite(), dbname="birdstrike.sqlite")  
195  
196 dbSendQuery(conn = db1, "CREATE TABLE BirdStrike (  
197     RecordID INTEGER PRIMARY KEY,  
198     Flight_ID INTEGER REFERENCES Flight_info(Flight_ID),  
199     Aircraft_Model INTEGER REFERENCES Aircraft_info(Aircraft_Model),  
200     Airport_ID INTEGER REFERENCES Airport_info(Airport_ID),  
201     Wildlife_ID INTEGER REFERENCES Wildlife_info(Wildlife_ID),  
202     When_time TEXT,  
203     Reported_Date TEXT,  
204     When_Phase_of_Flight TEXT,  
205     Condition_Sky TEXT,  
206     Pilot_Warned TEXT,  
207     Wildlife_Number_Strike INTEGER,  
208     Feet_Above_Ground TEXT,  
209     Speed_Knots TEXT,  
210     Condition_Precipitation TEXT,  
211     Cost_Repair TEXT,  
212     Cost_Aircraft_no_service TEXT,  
213     Effect_Flight_impact TEXT,  
214     Effect_Image TEXT,  
215     EFFECT_Other TEXT,  
216     Remains_Wildlife TEXT,  
217     Location_Nearby TEXT,  
218     Remarks TEXT)")  
219  
220  
221
```

The RStudio window title is "RStudio". The menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Tools, and Help. The toolbar shows icons for file operations and a "Go to file/function" search bar. The project is set to "Project: (None)". The script editor has tabs for "hw10.R*" and "Sqlite_importing data.R*". The console at the bottom is empty and labeled "Console". The Windows taskbar at the bottom shows various application icons and the system clock indicating 12:58 PM on 8/8/2015.

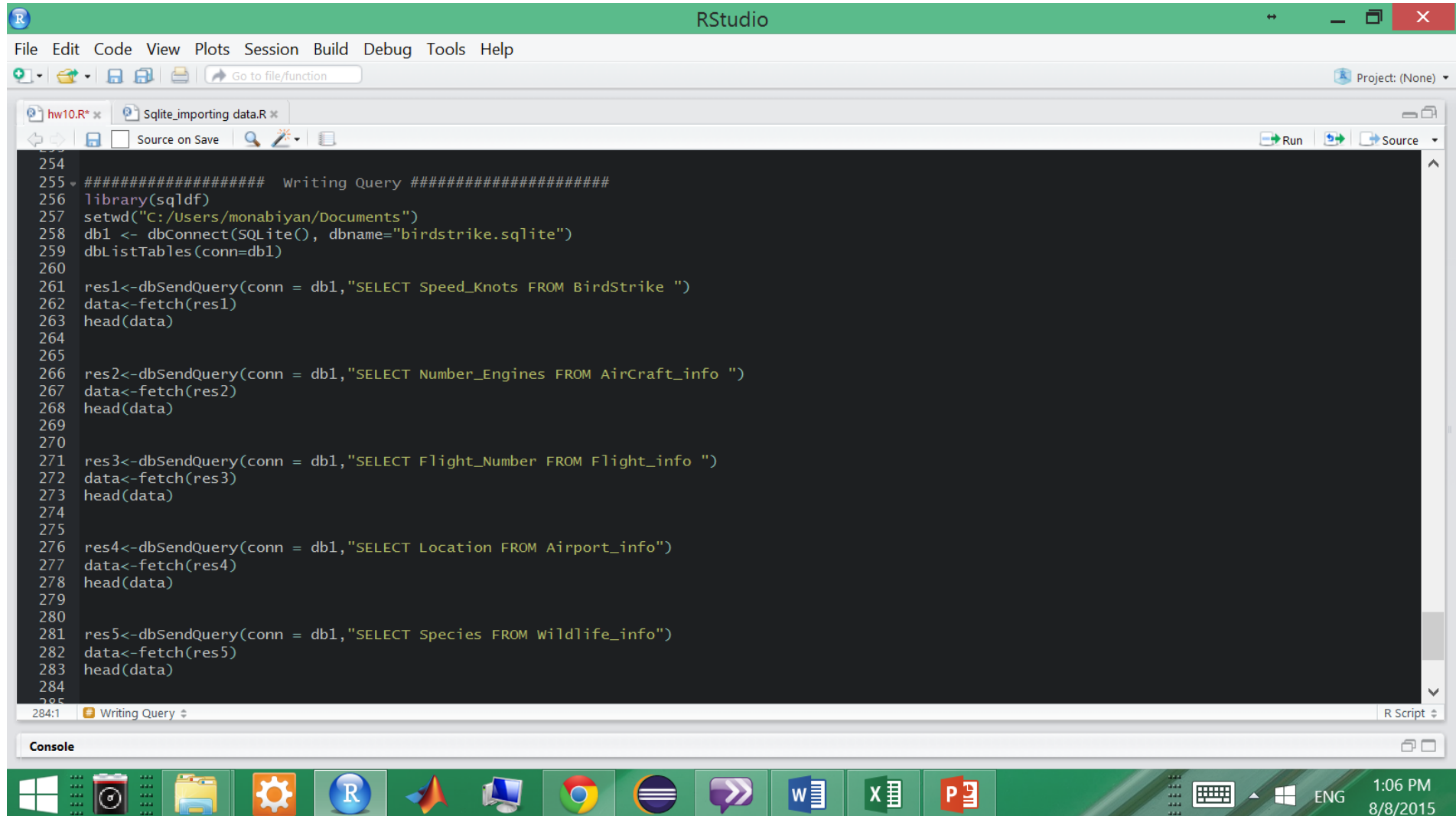
Defining the tables in the database and loading data frames in to the database tables



The screenshot displays the RStudio interface with a script editor containing R code for database operations. The code defines four tables: Aircraft_info, Flight_info, Airport_info, and Wildlife_info. It also lists fields for the Birdstrike table, writes data frames to the database, and removes the Aircraft_info table. The console shows the execution progress.

```
219
220 dbSendQuery(conn = db1, "CREATE TABLE Aircraft_info (Aircraft_Model INTEGER PRIMARY KEY, Type TEXT, Altitude TEXT, Number_Engines INTEGER)")
221
222 dbSendQuery(conn = db1, "CREATE TABLE Flight_info (Flight_ID INTEGER PRIMARY KEY, Flight_Number TEXT, Flight_date TEXT,
223             Airline TEXT, Airport TEXT)")
224
225 dbSendQuery(conn = db1, "CREATE TABLE Airport_info (Airport_ID INTEGER PRIMARY KEY, Airport_Name TEXT, Location TEXT)")
226
227 dbSendQuery(conn = db1, "CREATE TABLE Wildlife_info (Wildlife_ID INTEGER PRIMARY KEY, Species TEXT, Wildlif_size TEXT)")
228
229 #dbListTables(db1)
230
231 #####
232 #dbListFields(db1, "Birdstrike")
233
234 #colnames(birdstrike_df)<-dbListFields(db1, "Birdstrike")
235
236 #colnames(birdstrike_df)
237
238
239 #####Putting our data frame into the db1 database tabels. #####
240 dbWriteTable(conn=db1,name="BirdStrike",value=birdstrike_df,overwrite = T)
241
242 dbWriteTable(conn=db1,name="Aircraft_info",value=aircraft_df,overwrite = T)
243
244 dbWriteTable(conn=db1,name="Flight_info",value=flight_df,overwrite = T)
245
246 dbWriteTable(conn=db1,name="Airport_info",value=airport_df,overwrite = T)
247
248 dbWriteTable(conn=db1,name="Wildlife_info",value=wildlife_df,overwrite= T)
249
250 #dbRemoveTable(conn=db1,name="Aircraft_info")
228:1 Creating the main database with all definitions
```

Writing Queries for each table in the database



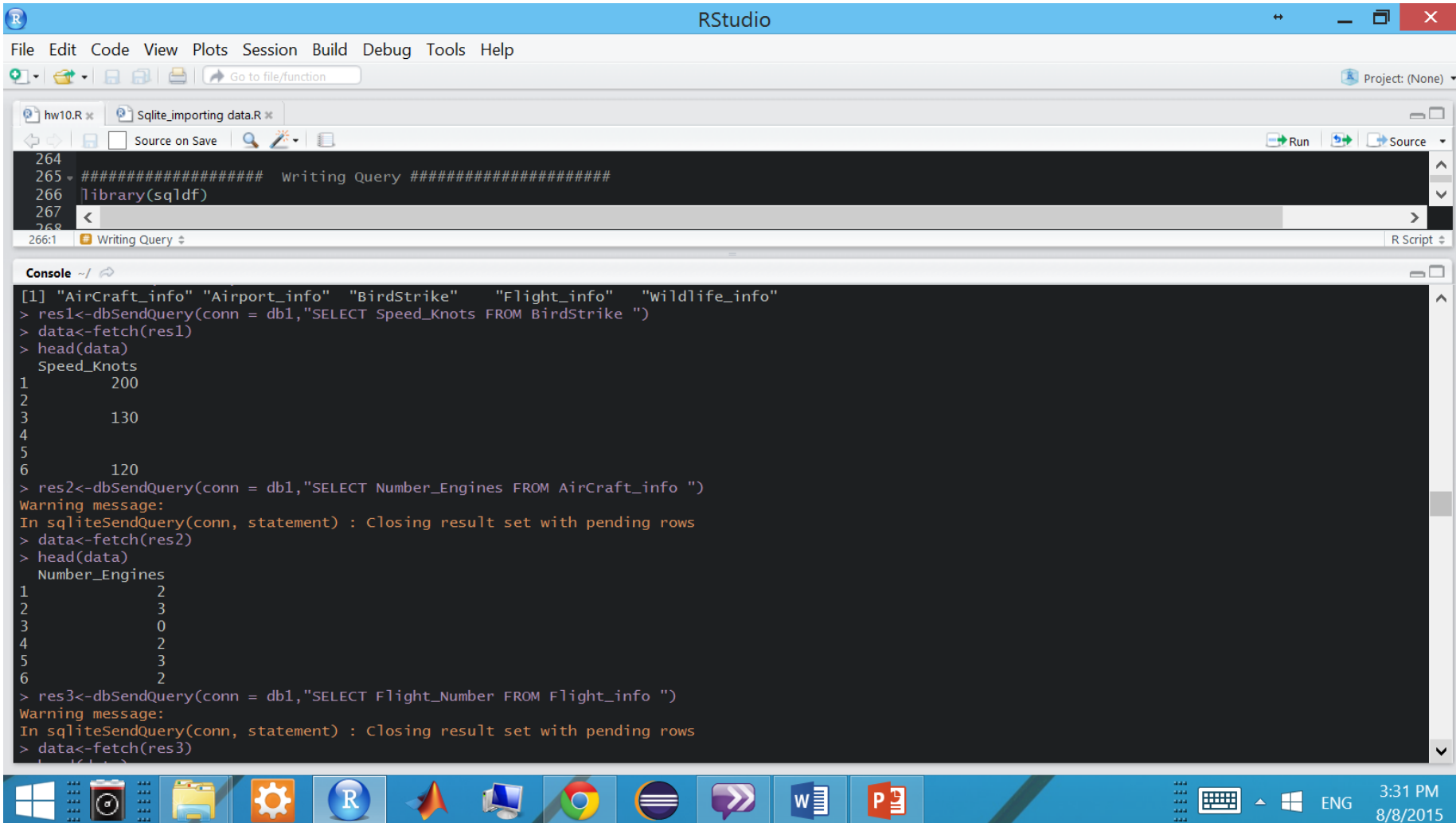
```
254
255 ##### Writing Query #####
256 library(sqldf)
257 setwd("C:/Users/monabiyar/Documents")
258 db1 <- dbConnect(SQLite(), dbname="birdstrike.sqlite")
259 dbListTables(conn=db1)
260
261 res1<-dbSendQuery(conn = db1,"SELECT Speed_Knots FROM BirdStrike ")
262 data<-fetch(res1)
263 head(data)
264
265
266 res2<-dbSendQuery(conn = db1,"SELECT Number_Engines FROM AirCraft_info ")
267 data<-fetch(res2)
268 head(data)
269
270
271 res3<-dbSendQuery(conn = db1,"SELECT Flight_Number FROM Flight_info ")
272 data<-fetch(res3)
273 head(data)
274
275
276 res4<-dbSendQuery(conn = db1,"SELECT Location FROM Airport_info")
277 data<-fetch(res4)
278 head(data)
279
280
281 res5<-dbSendQuery(conn = db1,"SELECT Species FROM wildlife_info")
282 data<-fetch(res5)
283 head(data)
284
285
```

284:1 Writing Query

Console

Windows Taskbar: 1:06 PM 8/8/2015

Writing Queries for each table in the database



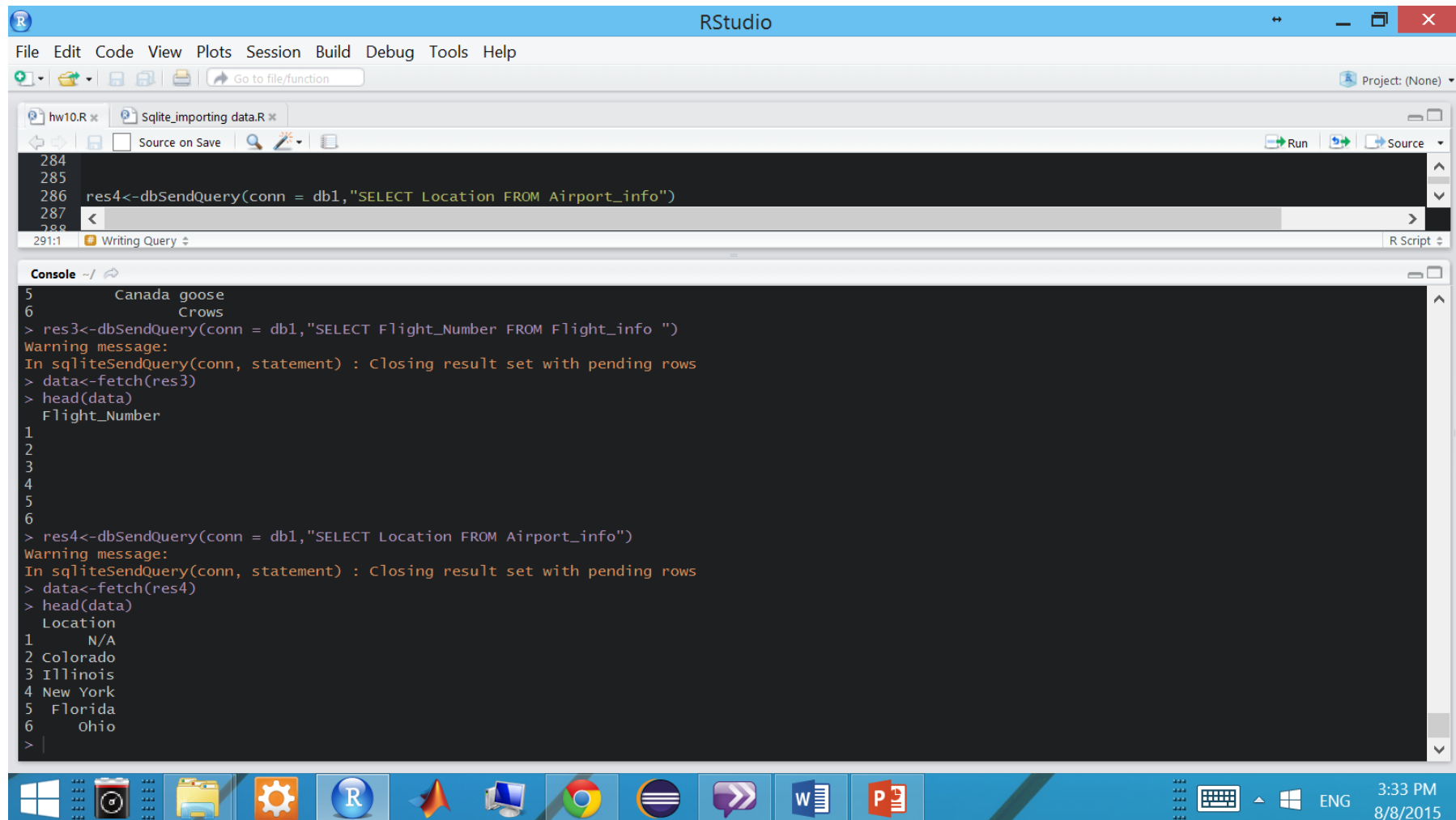
The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains R code for connecting to a database and querying it. The code includes a comment "Writing Query" and the use of the `library(sqldf)` package.
- Console:** Displays the output of the R code, including the list of tables in the database and the results of three SQL queries.
- Taskbar:** Shows the Windows taskbar with various application icons and the system clock.

```
264  
265 ##### Writing Query #####  
266 library(sqldf)  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000
```

```
[1] "AirCraft_info" "Airport_info" "BirdStrike" "Flight_info" "Wildlife_info"  
> res1<-dbSendQuery(conn = db1,"SELECT Speed_Knots FROM BirdStrike ")  
> data<-fetch(res1)  
> head(data)  
Speed_Knots  
1 200  
2  
3 130  
4  
5  
6 120  
> res2<-dbSendQuery(conn = db1,"SELECT Number_Engines FROM AirCraft_info ")  
Warning message:  
In sqliteSendQuery(conn, statement) : Closing result set with pending rows  
> data<-fetch(res2)  
> head(data)  
Number_Engines  
1 2  
2 3  
3 0  
4 2  
5 3  
6 2  
> res3<-dbSendQuery(conn = db1,"SELECT Flight_Number FROM Flight_info ")  
Warning message:  
In sqliteSendQuery(conn, statement) : Closing result set with pending rows  
> data<-fetch(res3)
```

Writing Queries for each table in the database



The screenshot shows the RStudio interface with two open R scripts: 'hw10.R' and 'Sqlite_importing data.R'. The 'Sqlite_importing data.R' script contains the following code:

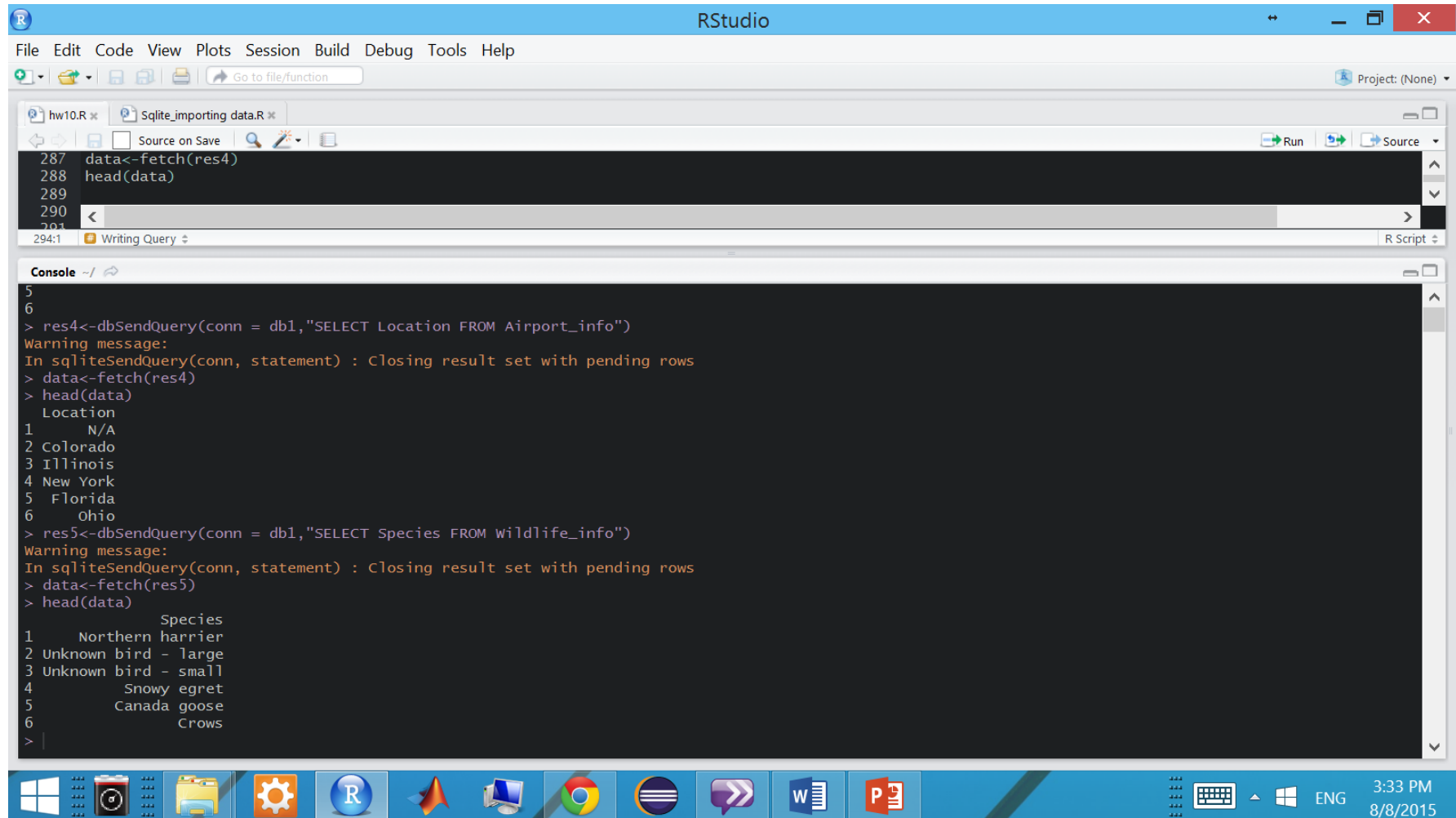
```
284  
285  
286 res4<-dbSendQuery(conn = db1,"SELECT Location FROM Airport_info")  
287  
288
```

The console window displays the output of the previous script, showing the results of two SQL queries. The first query, 'SELECT Flight_Number FROM Flight_info', returns a single column of flight numbers. The second query, 'SELECT Location FROM Airport_info', returns a single column of location names.

```
5      Canada goose  
6      Crows  
> res3<-dbSendQuery(conn = db1,"SELECT Flight_Number FROM Flight_info ")  
Warning message:  
In sqliteSendQuery(conn, statement) : Closing result set with pending rows  
> data<-fetch(res3)  
> head(data)  
  Flight_Number  
1  
2  
3  
4  
5  
6  
> res4<-dbSendQuery(conn = db1,"SELECT Location FROM Airport_info")  
Warning message:  
In sqliteSendQuery(conn, statement) : Closing result set with pending rows  
> data<-fetch(res4)  
> head(data)  
  Location  
1      N/A  
2 Colorado  
3 Illinois  
4 New York  
5  Florida  
6   Ohio  
>
```

The Windows taskbar at the bottom shows the system time as 3:33 PM on 8/8/2015, with the language set to ENG.

Writing Queries for each table in the database



The screenshot shows the RStudio interface with two open scripts: 'hw10.R' and 'Sqlite_importing data.R'. The 'Writing Query' pane shows the current script. The console displays the execution of two SQL queries and their results.

```
287 data<-fetch(res4)
288 head(data)
289
290
291
294:1 Writing Query
```

Console output:

```
5
6
> res4<-dbSendQuery(conn = db1,"SELECT Location FROM Airport_info")
Warning message:
In sqliteSendQuery(conn, statement) : Closing result set with pending rows
> data<-fetch(res4)
> head(data)
  Location
1      N/A
2 Colorado
3 Illinois
4 New York
5  Florida
6    Ohio

> res5<-dbSendQuery(conn = db1,"SELECT Species FROM Wildlife_info")
Warning message:
In sqliteSendQuery(conn, statement) : Closing result set with pending rows
> data<-fetch(res5)
> head(data)
  Species
1 Northern harrier
2 Unknown bird - large
3 Unknown bird - small
4      Snowy egret
5      Canada goose
6           Crows
```

The Windows taskbar at the bottom shows the time as 3:33 PM on 8/8/2015, with the language set to ENG.