

Querying Using SQL

Brady Lamson

2022-04-20

NOTE

Any random extra query parameters were added due to either curiosity or a desire to shorten the query time.

Exercise 1

```
db_con <-  
  DBI::dbConnect(  
    drv = MySQL(),  
    dbname = "airlines",  
    host = "mdsr.cdc7tgkkqd0n.us-east-1.rds.amazonaws.com",  
    user = "mdsr_public",  
    password = "ImhsmflMDSwR"  
  )
```

A

```
db_con %>% class()
```

```
## [1] "MySQLConnection"  
## attr(,"package")  
## [1] "RMySQL"
```

The class is a MySQLConnection.

B

```
DBI::dbListTables(db_con)
```

```
## [1] "airports" "carriers" "flights" "planes"
```

There are four separate tables in the database.

```
# C

# Described the contents of the airports table:
dbGetQuery(
  conn = db_con,
  statement = "DESCRIBE airports;"
) %>%
  nrow()
```

```
## [1] 9
```

There are 9 fields in airports.

```
# D

# Described the contents of the flights table:
dbGetQuery(
  conn = db_con,
  statement = "DESCRIBE flights;"
) %>%
  nrow()
```

```
## [1] 21
```

There are 21 fields in flights.

Exercise 2

```
DBI::dbGetQuery(
  conn = db_con,
  statement = "SELECT carrier, tailnum
              FROM flights
              LIMIT 0,5;"
)
```

```
##   carrier tailnum
## 1      XE  N11137
## 2      B6  N659JB
## 3      B6  N563JB
## 4      XE  N16559
## 5      00  N908SW
```

Exercise 3

```
# A

my_carriers <-
  DBI::dbGetQuery(
    conn = db_con,
    statement = "
      SELECT *
      FROM carriers;
    "
  )

my_carriers %>% is.data.frame()
```

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

Cool thing I learned passively googling is that you can just do SQL queries directly in a separate code chunk if you specify sql as the language and provide it the connection we created earlier. The output is a table that is very easy on the eyes.

```
# Using SQL directly

SELECT * FROM carriers
```

Table 1: Displaying records 1 - 10

carrier	name
02Q	Titan Airways
04Q	Tradewind Aviation
05Q	Comlux Aviation, AG
06Q	Master Top Linhas Aereas Ltd.
07Q	Flair Airlines Ltd.
09Q	Swift Air, LLC
0BQ	DCA
0CQ	ACM AIR CHARTER GmbH
0GQ	Inter Island Airways, d/b/a Inter Island Air
0HQ	Polar Airlines de Mexico d/b/a Nova Air

```
# B

my_carriers %>%
  object.size() %>%
  print(units = "Kb")
```

```
## 234.7 Kb
```

my_carries is approximately 235 kilobytes large.

Exercise 4

```
my_airports <-  
  DBI::dbGetQuery(  
    conn = db_con,  
    statement = "  
      SELECT *  
      FROM airports;  
    "  
  )
```

```
## Warning in .local(conn, statement, ...): Decimal MySQL column 2 imported as  
## numeric
```

```
## Warning in .local(conn, statement, ...): Decimal MySQL column 3 imported as  
## numeric
```

```
# A and B
```

```
glue::glue("The airports data set has  
           {my_airports %>% nrow()} rows and {my_airports %>% ncol()} columns.")
```

```
## The airports data set has  
## 1458 rows and 9 columns.
```

Exercise 5

```
DBI::dbGetQuery(  
  conn = db_con,  
  statement = "  
    SELECT distance / air_time * 60 AS trvl_speed  
    FROM flights  
    LIMIT 0,5;  
  "  
)
```

```
## Warning in .local(conn, statement, ...): Decimal MySQL column 0 imported as  
## numeric
```

```
##   trvl_speed  
## 1  419.6296  
## 2  512.5191  
## 3  488.8776  
## 4  396.5854  
## 5  338.9189
```

```
# Using SQL directly
```

```
SELECT distance / air_time * 60 AS trvl_speed  
FROM flights  
LIMIT 5
```

Table 2: 5 records

trvl_speed
419.6296
512.5191
488.8776
396.5854
338.9189

Exercise 6

A

```
DBI::dbGetQuery(  
  conn = db_con,  
  statement = "  
    SELECT *  
    FROM flights  
    WHERE arr_delay > 120  
    LIMIT 0,5;  
  "  
)
```

```
##   year month day dep_time sched_dep_time dep_delay arr_time sched_arr_time  
## 1 2010    10   1         1           2100         181        159           2320  
## 2 2010    10   1         1           1920         281        230           2214  
## 3 2010    10   1         7           2150         137        139           2337  
## 4 2010    10   1        12           2045         207        136           2209  
## 5 2010    10   1        20           2145         155        305            27  
##   arr_delay carrier tailnum flight origin dest air_time distance cancelled  
## 1         159      XE  N11137   2558   EWR  OMA        162       1133          0  
## 2         256      B6  N659JB    562   FLL  SWF        131       1119          0  
## 3         122      DL  N347NW   1752   ATL  IAD         70        533          0  
## 4         207      B6  N267JB   1329   BOS  BWI         61        370          0  
## 5         158      B6  N715JB    383   LGA  FLL        150       1076          0  
##   diverted hour minute           time_hour  
## 1          0   21      0 2010-10-01 21:00:00  
## 2          0   19     20 2010-10-01 19:20:00  
## 3          0   21     50 2010-10-01 21:50:00  
## 4          0   20     45 2010-10-01 20:45:00  
## 5          0   21     45 2010-10-01 21:45:00
```

B

```
DBI::dbGetQuery(  
  conn = db_con,  
  statement = "  
    SELECT year, month, day, dest  
    FROM flights  
    WHERE dest IN ('IAH', 'HOU')  
    LIMIT 0,5;  
  "  
)
```

```
##   year month day dest  
## 1 2010    10   1  HOU  
## 2 2010    10   1  HOU  
## 3 2010    10   1  HOU  
## 4 2010    10   1  HOU  
## 5 2010    10   1  HOU
```

```
# C

DBI::dbGetQuery(
  conn = db_con,
  statement = "
    SELECT dep_time, dep_delay, arr_delay, carrier
    FROM flights
    WHERE carrier IN ('UA', 'AA', 'DL')
    LIMIT 0,5;
  "
)
```

```
##   dep_time dep_delay arr_delay carrier
## 1         7         -3         -9      AA
## 2        21         -4          2      AA
## 3        43         -2         -7      AA
## 4       119         44         45      AA
## 5       538          3         15      AA
```

For fun I'll just use SQL directly on parts d, e and f.

```
# D

SELECT year, month, day
FROM flights
WHERE month BETWEEN 7 and 9 AND year = 2013
LIMIT 0,5;
```

Table 3: 5 records

year	month	day
2013	7	1
2013	7	1
2013	7	1
2013	7	1
2013	7	1

```
# E

SELECT dep_time, arr_time
FROM flights
WHERE dep_time = 2400 OR dep_time BETWEEN 0 and 600 AND year = 2013
LIMIT 0,5;
```

Table 4: 5 records

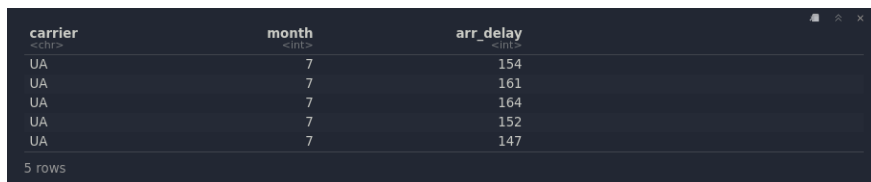
dep_time	arr_time
2400	341
2400	742
2400	51

dep_time	arr_time
2400	105
2400	521

```
# F

DBI::dbGetQuery(
  conn = db_con,
  statement = "
    SELECT carrier, month, arr_delay
    FROM flights
    WHERE carrier = 'UA' AND month = 7 AND arr_delay > 120 AND year = 2013
    LIMIT 0,5;
  "
)
```

```
knitr::include_graphics("images/prob_f_query.png")
```



carrier <chr>	month <int>	arr_delay <int>
UA	7	154
UA	7	161
UA	7	164
UA	7	152
UA	7	147

5 rows

Exercise 7

A

```
DBI::dbGetQuery(  
  conn = db_con,  
  statement = "  
    SELECT carrier, MIN(dep_delay) AS minimum_delay  
    FROM flights  
    WHERE year = 2013 AND month = 6 AND day = 26  
    GROUP BY carrier  
    LIMIT 0,5;  
  "  
)
```

```
##   carrier minimum_delay  
## 1      9E           -19  
## 2      AA           -28  
## 3      AS           -22  
## 4      B6           -18  
## 5      DL           -19
```

B

```
DBI::dbGetQuery(  
  conn = db_con,  
  statement = "  
    SELECT carrier, MIN(dep_delay) AS minimum_delay, MAX(dep_delay) AS maximum_delay  
    FROM flights  
    WHERE year = 2013 AND month = 6 AND day = 26  
    GROUP BY carrier  
    LIMIT 0,5;  
  "  
)
```

```
##   carrier minimum_delay maximum_delay  
## 1      9E           -19           506  
## 2      AA           -28           877  
## 3      AS           -22           199  
## 4      B6           -18           406  
## 5      DL           -19           721
```

Exercise 8

```
dbGetQuery(  
  conn = db_con,  
  statement = "SELECT carrier, dest, AVG(arr_delay) AS meanArrDelay  
               FROM flights  
               WHERE year = 2013 AND month = 6 AND day = 26 AND origin = 'BDL'  
               GROUP BY dest;"  
)
```

This selects the carrier and destination columns unchanged. It also creates a new column for the AVERAGE arrival delay for each destination. This selection only occurs on June 26th, 2013 where the origin of the flight was BDL.

```
dbGetQuery(  
  conn = db_con,  
  statement = "SELECT carrier, dest, AVG(arr_delay) AS meanArrDelay,  
               AVG(distance) AS meanDist  
               FROM flights  
               WHERE year = 2013 AND month = 6 AND day = 26 AND origin = 'BDL'  
               GROUP BY dest;"  
)
```

This is the same as the previous example, but instead we also calculate the average distance for each destination.

Exercise 9

A

```
dbGetQuery(  
  conn = db_con,  
  statement = "SELECT dest, AVG(air_time) AS avg_travel_time  
               FROM flights  
               WHERE year = 2013 AND origin = 'BDL'  
               GROUP BY dest  
               ORDER BY avg_travel_time ASC  
               LIMIT 0,5;"  
)
```

Description: df [5 × 2]

dest <chr>	avg_travel_time <dbl>
EWR	32.8673
BWI	52.6590
PHL	53.3492
IAD	57.5096
DCA	65.5490

5 rows

EWR was the shortest.

B

```
dbGetQuery(  
  conn = db_con,  
  statement = "SELECT dest, AVG(air_time) AS avg_travel_time  
               FROM flights  
               WHERE year = 2013 AND origin = 'BDL'  
               GROUP BY dest  
               ORDER BY avg_travel_time DESC  
               LIMIT 0,5;"  
)
```

Description: df [5 × 2]

dest <chr>	avg_travel_time <dbl>
LAX	341.7165
LAS	310.8626
DEN	236.4110
DFW	205.8531
SJU	203.8560

5 rows

LAX was the longest average travel time.

Exercise 10

```
# A
dbGetQuery(
  conn = db_con,
  statement = "SELECT dest, COUNT(*) AS num_flights
              FROM flights
              WHERE year = 2013 AND origin = 'BDL'
              GROUP BY dest
              ORDER BY num_flights DESC
              LIMIT 0,5;"
)
```

dest <chr>	num_flights <dbl>
ORD	2657
BWI	2613
ATL	2277
CLT	1842
MCO	1789

5 rows

ORD had the most flights.

```
# B
dbGetQuery(
```

```
conn = db_con,  
statement = "SELECT tailnum, COUNT(*) AS num_flights  
            FROM flights  
            WHERE year = 2013 AND origin = 'BDL'  
            GROUP BY tailnum  
            ORDER BY num_flights DESC  
            LIMIT 0,5;"  
)
```

tailnum <chr>	num_flights <dbl>
NA	97
N128UW	36
N504MJ	35
N505MJ	35
N503MJ	34

Tailnumber N128UW had the most flights.

Exercise 11

A

```
dbGetQuery(  
  conn = db_con,  
  statement = "SELECT dest, COUNT(*) AS numFlights,  
                AVG(arr_delay) AS avg_arr_delay  
                FROM flights  
                WHERE year = 2013 AND origin = 'BDL'  
                GROUP BY dest  
                HAVING avg_arr_delay > 0;"  
)
```

dest <chr>	numFlights <dbl>	avg_arr_delay <dbl>
ATL	2277	4.4704
BWI	2613	5.0325
DFW	1062	0.7495
EWR	437	13.1968
FLL	1011	0.2770
IAD	622	4.9084
MCO	1789	8.3784
MDW	974	8.5144
ORD	2657	7.3643
PBI	365	8.4466

1-10 of 15 rows

B

```
dbGetQuery(  
  conn = db_con,  
  statement = "SELECT dest, COUNT(*) AS numFlights,  
                AVG(arr_delay) AS avg_arr_delay  
                FROM flights  
                WHERE year = 2013 AND origin = 'BDL'  
                GROUP BY dest  
                HAVING avg_arr_delay > 0 AND numFlights > 1000;"  
)
```

dest <chr>	numFlights <dbl>	avg_arr_delay <dbl>
ATL	2277	4.4704
BWI	2613	5.0325
DFW	1062	0.7495
FLL	1011	0.2770
MCO	1789	8.3784
ORD	2657	7.3643

6 rows

Exercise 12

```

dbGetQuery(
  conn = db_con,
  statement = "SELECT carrier, AVG(dep_delay) AS avg_dep_delay
              FROM flights
              WHERE year = 2013 AND origin = 'BDL'
              GROUP BY carrier
              HAVING avg_dep_delay > 10;"
)

```

carrier <chr>	avg_dep_delay <dbl>
EV	10.7332
MQ	12.8813
WN	10.9046
3 rows	

Exercise 13

```

dbGetQuery(conn = db_con,
  statement = "SELECT dest, COUNT(*) AS numFlights, AVG(arr_delay) AS avg_arr_delay
              FROM flights
              WHERE year = 2013 AND origin = 'BDL'
              GROUP BY dest
              HAVING numFlights > 365 * 2
              ORDER BY avg_arr_delay ASC
              LIMIT 5, 4;"
)

```

dest <chr>	numFlights <dbl>	avg_arr_delay <dbl>
ATL	2277	4.4704
BWI	2613	5.0325
ORD	2657	7.3643
MCO	1789	8.3784
4 rows		