SQL: Exercises

You can download this .qmd file from here. Just hit the Download Raw File button.

The code in 15_SQL.qmd walked us through many of the examples in MDSR Chapter 15; now, we present a set of practice exercises in converting from the tidyverse to SQL.

```
library(tidyverse)
library(mdsr)
library(dbplyr)
library(DBI)
```

```
# connect to the database which lives on a remote server maintained by
#   St. Olaf's IT department
library(RMariaDB)
con <- dbConnect(
  MariaDB(), host = "mdb.stolaf.edu",
  user = "ruser", password = "ruserpass",
  dbname = "flight_data"
)
```

On Your Own - Extended Example from MDSR

Refer to Section 15.5 in MDSR, where they attempt to replicate some of FiveThirtyEight's analyses. The MDSR authors provide a mix of SQL and R code to perform their analyses, but the code will not work if you simply cut-and-paste as-is into R. Your task is to convert the book code into something that actually runs, and then *apply it to data from 2024*. Very little of the code needs to be adjusted; it mostly needs to be repackaged.

Hints:

- use `dbGetQuery()`

- note that what they call `carrier` is just called `Reporting_Airline` in the `flightdata` table; you don't have to merge in a `carrier` table, although it's unfortunate that the `Reporting_Airline` codes are a bit cryptic

1. Below Figure 15.1, the MDSR authors first describe how to plot slowest and fastest airports. Instead of using *target time*, which has a complex definition, we will use *arrival time*, which oversimplifies the situation but gets us in the ballpark. Duplicate the equivalent of the table below for 2024 using the code in MDSR:

```
# A tibble: 30 × 3
   dest  avgDepartDelay avgArrivalDelay
   <chr>          <dbl>           <dbl>
 1 ORD            14.3            13.1
 2 MDW            12.8             7.40
 3 DEN            11.3             7.60
 4 IAD            11.3             7.45
 5 HOU            11.3             8.07
 6 DFW            10.7             9.00
 7 BWI            10.2             6.04
 8 BNA             9.47            8.94
 9 EWR             8.70            9.61
10 IAH             8.41            6.75
# 20 more rows
```

2. Following the table above, the MDSR authors mimic one more FiveThirtyEight table which ranks carriers by time added vs. typical and time added vs. target. In this case, we will find average arrival delay after controlling for the routes flown. Again, duplicate the equivalent of the table below for 2024 using the code in MDSR:

```
# A tibble: 14 × 5
   carrier carrier_name                numRoutes numFlights wAvgDelay
   <chr>   <chr>                           <int>      <dbl>     <dbl>
 1 VX      Virgin America                     72      57510   -2.69
 2 FL      AirTran Airways Corporation       170      79495   -1.55
 3 AS      Alaska Airlines Inc.              242     160257   -1.44
 4 US      US Airways Inc.                   378     414665   -1.31
 5 DL      Delta Air Lines Inc.              900     800375   -1.01
 6 UA      United Air Lines Inc.             621     493528   -0.982
 7 MQ      Envoy Air                         442     392701   -0.455
 8 AA      American Airlines Inc.            390     537697   -0.0340
 9 HA      Hawaiian Airlines Inc.             56      74732    0.272
10 OO      SkyWest Airlines Inc.            1250     613030    0.358
11 B6      JetBlue Airways                   316     249693    0.767
```

```
12 EV      ExpressJet Airlines Inc.         1534     686021    0.845
13 WN      Southwest Airlines Co.           1284    1174633    1.13
14 F9      Frontier Airlines Inc.            326      85474    2.29
```

On Your Own - Adapting 164 Code

These problems are based on class exercises from SDS 164, so you've already solved them in R! Now we're going to try to duplicate those solutions in SQL (but with 2023 data instead of 2013).

```
# Read in 2013 NYC flights data
library(nycflights13)
flights_nyc13 <- nycflights13::flights
planes_nyc13 <- nycflights13::planes
```

1. Summarize carriers flying to MSP by number of flights and proportion that are cancelled (assuming that a missing arrival time indicates a cancelled flight). [This was #4 in 17_longer_pipelines.Rmd.]

```
# Original solution from SDS 164
flights_nyc13 |>
  mutate(carrier = fct_collapse(carrier, "Delta +" = c("DL", "9E"),
                                         "American +"= c("AA", "MQ"),
                                        "United +" = c("EV", "OO", "UA"))) |>
  filter(dest == "MSP") |>
  group_by(origin, carrier) |>
  summarize(n_flights = n(),
            num_cancelled = sum(is.na(arr_time)),
            prop_cancelled = mean(is.na(arr_time)))
```

```
# A tibble: 5 x 5
# Groups:   origin [3]
  origin carrier    n_flights num_cancelled prop_cancelled
  <chr>  <fct>          <int>         <int>          <dbl>
1 EWR    Delta +          598            10         0.0167
2 EWR    United +        1779           105         0.0590
3 JFK    Delta +         1095            41         0.0374
4 LGA    Delta +         2420            25         0.0103
5 LGA    American +      1293            62         0.0480
```

First duplicate the output above, then check trends in 2023 across all origins. Here are a few hints:

- use flightdata instead of flights_nyc13
- remember that flights_nyc13 only contained 2013 and 3 NYC origin airports (EWR, JFK, LGA)
- is.na can be replaced with CASE WHEN ArrTime IS NULL THEN 1 ELSE 0 END or with CASE WHEN cancelled = 1 THEN 1 ELSE 0 END
- CASE WHEN can also be used replace fct_collapse

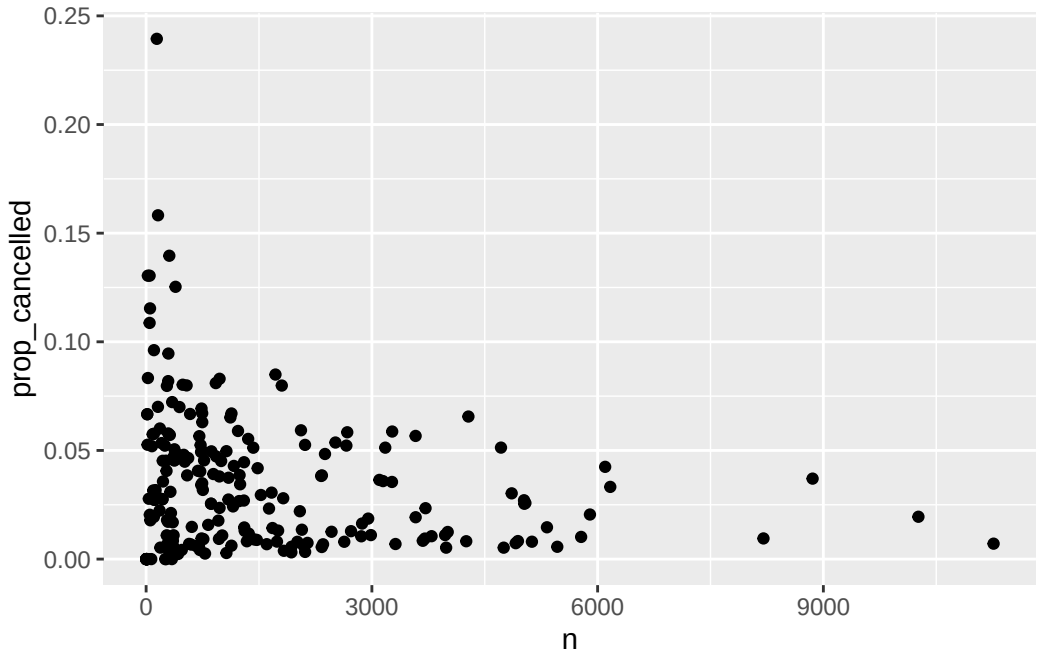
```
SELECT origin,
  CASE WHEN Reporting_Airline IN ("DL", "9E") THEN 'Delta +'
    WHEN Reporting_Airline IN ("AA", "MQ") THEN 'American +'
    WHEN Reporting_Airline IN ("EV", "OO", "UA") THEN 'United +'
    ELSE 'Other' END AS new_carrier,
  SUM(1) AS n_flights,
  SUM(cancelled) AS n_cancelled,
  AVG(cancelled) AS prop_cancelled,
  year
FROM flightdata
WHERE dest = 'MSP' AND origin IN ('JFK', 'EWR', 'LGA') AND year = 2023
GROUP BY origin, new_carrier
LIMIT 1, 10;
```

Table 1: 7 records

origin	new_carrier	n_flights	n_cancelled	prop_cancelled	year
EWR	Other	214	4	0.0187	2023
EWR	United +	859	26	0.0303	2023
JFK	Delta +	1049	20	0.0191	2023
JFK	Other	84	2	0.0238	2023
JFK	United +	63	3	0.0476	2023
LGA	Delta +	1729	35	0.0202	2023
LGA	Other	632	12	0.0190	2023

2. Plot number of flights vs. proportion cancelled for every origin-destination pair (assuming that a missing arrival time indicates a cancelled flight). [This was #7 in 17_longer_pipelines.Rmd.]

```r
# Original solution from SDS 164
flights_nyc13 |>
  group_by(origin, dest) |>
  summarize(n = n(),
            prop_cancelled = mean(is.na(arr_time))) |>
  filter(prop_cancelled < 1) |>
  ggplot(aes(n, prop_cancelled)) +
  geom_point()
```



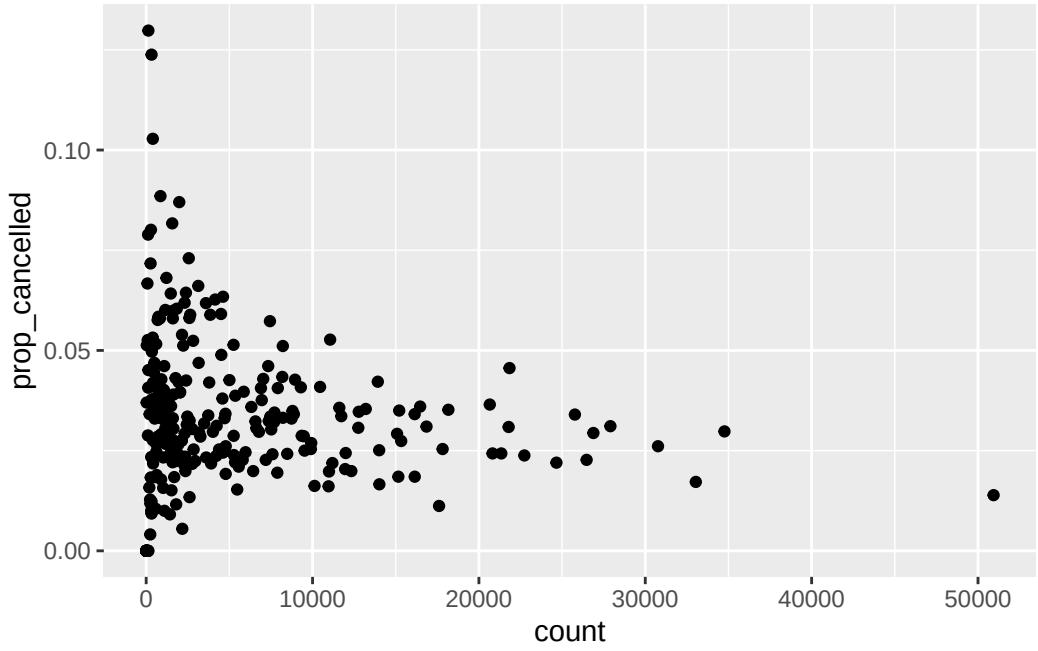
First duplicate the plot above for 2023 data, then check trends across all origins. Do all of the data wrangling in SQL. Here are a few hints:

- use flightdata instead of flights_nyc13
- remember that flights_nyc13 only contained 2013 and 3 NYC origin airports (EWR, JFK, LGA)
- use an `sql` chunk and an `r` chunk
- include `connection =` and `output.var =` in your sql chunk header (this doesn't seem to work with dbGetQuery()…)

```sql
SELECT
  origin, dest,
  SUM(1) AS count,
  AVG(cancelled) AS prop_cancelled
```

```
FROM flightdata
WHERE origin = 'JFK' OR origin = 'EWR' OR origin = 'LGA' AND year = 2023
GROUP BY origin, dest
HAVING prop_cancelled < 1;
```

```
tester |>
  ggplot(aes(count, prop_cancelled)) +
  geom_point()
```



im getting a little sick of just helping

3. [SKIP until the planes dataset becomes available] Produce a table of weighted plane age by carrier, where weights are based on number of flights per plane. [This was #6 in 26_more_joins.Rmd.]

```
# Original solution from SDS 164
flights_nyc13 |>
  left_join(planes_nyc13, join_by(tailnum)) |>
  mutate(plane_age = 2013 - year.y) |>
  group_by(carrier) |>
  summarize(unique_planes = n_distinct(tailnum),
            mean_weighted_age = mean(plane_age, na.rm =TRUE),
```

```
            sd_weighted_age = sd(plane_age, na.rm =TRUE)) |>
  arrange(mean_weighted_age)
```

```
# A tibble: 16 x 4
   carrier unique_planes mean_weighted_age sd_weighted_age
   <chr>           <int>             <dbl>           <dbl>
 1 HA                 14              1.55            1.14
 2 AS                 84              3.34            3.07
 3 VX                 53              4.47            2.14
 4 F9                 26              4.88            3.67
 5 B6                193              6.69            3.29
 6 OO                 28              6.84            2.41
 7 9E                204              7.10            2.67
 8 US                290              9.10            4.88
 9 WN                583              9.15            4.63
10 YV                 58              9.31            1.93
11 EV                316             11.3             2.29
12 FL                129             11.4             2.16
13 UA                621             13.2             5.83
14 DL                629             16.4             5.49
15 AA                601             25.9             5.42
16 MQ                238             35.3             3.13
```

First duplicate the output above for 2023, then check trends across all origins. Do all of the data wrangling in SQL. Here are a few hints:

- use flightdata instead of flights_nyc13
- remember that flights_nyc13 only contained 2013 and 3 NYC origin airports (EWR, JFK, LGA)
- you'll have to merge the flights dataset with the planes dataset *when it becomes available*
- you can use DISTINCT inside a COUNT()

```
SELECT
  Reporting_Airline AS carrier,
  a.name AS carrier_name,
  COUNT(DISTINCT(tailnum)) AS unique_planes,
  AVG(2024 - p.year) AS mean_weighted_age,
  STDDEV_SAMP(2024 - p.year) AS sd_weighted_age
FROM flightdata AS f
JOIN planes AS p ON f.TAIL_NUMBER = p.tailnum
JOIN airlines AS a ON f.Reporting_Airline = a.carrier
```

```
WHERE origin IN ('JFK', 'EWR', 'LGA') AND f.year = 2023
GROUP BY carrier
ORDER BY mean_weighted_age ASC
```

Table 2: Displaying records 1 - 10

carrier	carrier_name	unique_planes	mean_weighted_age	sd_weighted_age
F9	Frontier Airlines Inc.	134	5.135947	2.365734
OO	SkyWest Airlines Inc.	188	6.850972	4.212619
AS	Alaska Airlines Inc.	205	7.404048	4.900640
NK	Spirit Air Lines	196	7.738930	3.613592
HA	Hawaiian Airlines Inc.	24	10.728571	1.938314
MQ	Envoy Air	90	11.690840	5.861740
WN	Southwest Airlines Co.	788	11.848831	6.883556
YX	Republic Airline	196	12.993201	4.826594
9E	Endeavor Air Inc.	127	13.218725	3.968461
AA	American Airlines Inc.	890	13.943699	6.999990