Querying Using SQL

Brady Lamson

2022-04-27

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.

```r
# C

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

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

There are 9 fields in airports.

```r
# 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

```r
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      OO  N908SW
```

Exercise 3

```r
# 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.

```sql
# 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

```r
# B

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

```
## 234.7 Kb
```

`my_carries` is approximately 235 kilobytes large.

Exercise 4

```r
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
```

```r
# 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

```r
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
```

```sql
# 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

```r
# 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
```

```r
# 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
```

```r
# 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

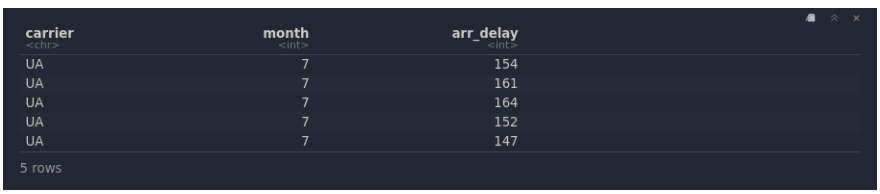
dep_time	arr_time
2400	341
2400	742
2400	51

dep_time	arr_time
2400	105
2400	521

```r
# 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;
    "
)
```

```r
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

```r
# 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
```

```r
# 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

```r
# 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.

```r
# 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

```r
# 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 Previous 1 2 Next

```r
# 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

```r
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

```r
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

Exercise 14

```
dbGetQuery(conn = db_con,
           statement = "SELECT dest, flight, carrier, airports.name
                        FROM flights
                        JOIN airports ON flights.dest = airports.faa
                        WHERE year = 2013 AND month = 6 AND day = 26 AND origin = 'BDL'
                        LIMIT 10;"
)
```

```
##    dest flight carrier                              name
## 1   EWR   4714      EV                Newark Liberty Intl
## 2   MIA   2015      AA                        Miami Intl
## 3   DTW   1644      DL            Detroit Metro Wayne Co
## 4   BWI   2584      WN         Baltimore Washington Intl
## 5   ATL   1065      DL   Hartsfield Jackson Atlanta Intl
## 6   DCA   1077      US     Ronald Reagan Washington Natl
## 7   TPA    627      WN                        Tampa Intl
## 8   MSP    797      DL          Minneapolis St Paul Intl
## 9   CLT   1705      US            Charlotte Douglas Intl
## 10  CVG   3787      9E Cincinnati Northern Kentucky Intl
```

The destination airport for flight 4714 is Neward Liberty Intl.

Exercise 15

```r
# A

dbGetQuery(conn = db_con,
           statement = "SELECT dest, flight, carriers.carrier
                        FROM flights
                        JOIN carriers ON flights.carrier = carriers.carrier
                        WHERE year = 2013 AND month = 6 AND day = 26 AND origin = 'BDL' AND dest = 'MSP
                        LIMIT 10;"
)
```

```
##   dest flight carrier
## 1  MSP    797      DL
## 2  MSP   3338      9E
## 3  MSP   1226      DL
```

```r
# B

dbGetQuery(conn = db_con,
           statement = "SELECT dest, flight, carriers.carrier, carriers.name
                        FROM flights
                        JOIN carriers ON flights.carrier = carriers.carrier
                        WHERE year = 2013 AND month = 6 AND day = 26 AND origin = 'BDL' AND name = 'Mesa
                        LIMIT 50;"
)
```

```
##   dest flight carrier               name
## 1  ORD   3755      YV Mesa Airlines Inc.
## 2  ORD   3737      YV Mesa Airlines Inc.
## 3  IAD   3745      YV Mesa Airlines Inc.
```

Exercise 16

```
dbGetQuery(conn = db_con,
           statement = "SELECT origin, dest, name AS dest_name, flight, carrier
                        FROM flights
                        LEFT JOIN airports ON flights.dest = airports.faa
                        WHERE year = 2013 AND month = 6 AND day = 26 AND origin = 'PBI';")
```

```
dbGetQuery(conn = db_con,
           statement = "SELECT origin, dest, name AS dest_name, flight, carrier
                        FROM flights
                        JOIN airports ON flights.dest = airports.faa
                        WHERE year = 2013 AND month = 6 AND day = 26 AND origin = 'PBI';")
```

`LEFT JOIN` returns everything that matches the query for the flights dataset. Rows that don't have information in the airports dataset are kept, just get NAs tacked onto stuff that doesn't apply. `JOIN` doesn't work that way, and will exclude rows that aren't in both. Based on `JOIN` returning one less, it can be assumed that there's one row that isn't in both.

Exercise 17

```
dbGetQuery(conn = db_con,
           statement = "(SELECT year, month, day, origin, dest, flight, carrier
                        FROM flights
                        WHERE year = 2013 AND month = 6 AND day = 26
                        AND origin = 'BDL' AND dest = 'ORD')
                        UNION
                        (SELECT year, month, day, origin, dest, flight, carrier
                        FROM flights
                        WHERE year = 2013 AND month = 6 AND day = 26
                        AND origin = 'MSP' AND dest = 'JFK');"
)
```

This query is a combination of the flights on June 6th, 2016 where **either** (the origin is BDL **and** the destination is ORD) **or** (the origin is MSP **and** the destination is JFK).

Exercise 18

```
dbGetQuery(conn = db_con,
           statement = "SELECT dest, COUNT(*)
                        FROM flights
                        WHERE year = 2013
                        AND origin = 'BDL'
                        AND dest IN
                            (SELECT faa
                            FROM airports
                            WHERE tz < -7)
                        GROUP BY dest;"
)
```

dest <chr>	COUNT(*) <dbl>
LAS	262
LAX	127

2 rows

LAS and LAX were the two destinations traveled to.

Exercise 19

```
dbGetQuery(conn = db_con,
           statement = "SELECT dest, COUNT(*)
                        FROM flights
                        WHERE year = 2013
                        AND origin = 'JFK'
                        AND dest IN
                            (SELECT faa
                            FROM airports
                            WHERE tz < -8)
                        GROUP BY dest;"
)
```

dest <chr>	COUNT(*) <dbl>
HNL	342

1 row

HNL was the only airport traveled to.

Exercise 20

```r
flights <- tbl(db_con, "flights")
carriers <- tbl(db_con, "carriers")
```

```r
flights %>%
    select(carrier, tailnum) %>%
    head()
```

```
## # Source:   lazy query [?? x 2]
## # Database: mysql 5.7.33-log
## #   [@mdsr.cdc7tgkkqd0n.us-east-1.rds.amazonaws.com:/airlines]
##   carrier tailnum
##   <chr>   <chr>
## 1 XE      N11137
## 2 B6      N659JB
## 3 B6      N563JB
## 4 XE      N16559
## 5 OO      N908SW
## 6 AA      N3FRAA
```

Exercise 21

```r
# A

flights %>%
    filter(arr_delay > 120) %>%
    head()
```

```
## # Source:   lazy query [?? x 21]
## # Database: mysql 5.7.33-log
## #   [@mdsr.cdc7tgkkqd0n.us-east-1.rds.amazonaws.com:/airlines]
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <int>    <int>          <int>
## 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
## 6  2010    10     1       30           2135       175      130           2233
## # ... with 13 more variables: arr_delay <int>, carrier <chr>, tailnum <chr>,
## #   flight <int>, origin <chr>, dest <chr>, air_time <int>, distance <int>,
## #   cancelled <int>, diverted <int>, hour <int>, minute <int>, time_hour <chr>
```

```r
# B

# Note: select() added just to showcase the query working w/ less columns.
flights %>%
    select(dest, tailnum, carrier) %>%
    filter(dest %in% c('IAH', 'HOU')) %>%
    head()
```

```
## # Source:   lazy query [?? x 3]
## # Database: mysql 5.7.33-log
## #   [@mdsr.cdc7tgkkqd0n.us-east-1.rds.amazonaws.com:/airlines]
##   dest  tailnum carrier
##   <chr> <chr>   <chr>
## 1 HOU   N527SW  WN
## 2 HOU   N347SW  WN
## 3 HOU   N262WN  WN
## 4 HOU   N660SW  WN
## 5 HOU   N741SA  WN
## 6 HOU   N791SW  WN
```

```r
# C

# summarise used to showcase the query working and not only returning AA flights.
flights %>%
    select(dest, tailnum, carrier) %>%
    filter(carrier %in% c('UA', 'AA', 'DL')) %>%
    group_by(carrier) %>%
    summarise(count = n())
```

```
## # Source:   lazy query [?? x 2]
## # Database: mysql 5.7.33-log
## #   [@mdsr.cdc7tgkkqd0n.us-east-1.rds.amazonaws.com:/airlines]
##   carrier   count
##   <chr>     <dbl>
## 1 AA      5216777
## 2 DL      6469415
## 3 UA      3830135
```

```
# D

flights %>%
    select(dest, tailnum, month) %>%
    filter(between(month, 7, 9)) %>%
    group_by(month) %>%
    summarise(count = n())
```

Source: lazy query [?? x 2] Database: mysql 5.7.33-log [@mdsr.cdc7tgkkqd0n.us-east-1.rds.amazonaws.com:/airlines]

month <int>	count <dbl>
7	4287886
8	4241198
9	3862620

3 rows

```
# E

flights %>%
    filter(dep_time == 2400 | between(dep_time, 0, 600)) %>%
    head()
```

```
## # Source:   lazy query [?? x 21]
## # Database: mysql 5.7.33-log
## #   [@mdsr.cdc7tgkkqd0n.us-east-1.rds.amazonaws.com:/airlines]
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <int>    <int>          <int>
## 1  2010    10     1        1           2100       181      159           2320
## 2  2010    10     1        1           1920       281      230           2214
## 3  2010    10     1        3           2355         8      339            334
## 4  2010    10     1        5           2200       125       41           2249
## 5  2010    10     1        7           2245        82      104           2347
## 6  2010    10     1        7             10        -3      451            500
## # ... with 13 more variables: arr_delay <int>, carrier <chr>, tailnum <chr>,
## #   flight <int>, origin <chr>, dest <chr>, air_time <int>, distance <int>,
## #   cancelled <int>, diverted <int>, hour <int>, minute <int>, time_hour <chr>
```

```
# F

flights %>%
    filter(carrier == 'UA' & month == 7 & arr_delay > 120) %>%
    head()
```

```
## # Source:   lazy query [?? x 21]
```

```
## # Database: mysql 5.7.33-log
## #   [@mdsr.cdc7tgkkqd0n.us-east-1.rds.amazonaws.com:/airlines]
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <int>    <int>          <int>
## 1  2010     7     1       23           2130       173      148           2305
## 2  2010     7     1     1600           1216       224     1843           1502
## 3  2010     7     1     2049           1631       258     2210           1756
## 4  2010     7     1     2107           1415       412       18           1744
## 5  2010     7     1     2148           1735       253     2320           1936
## 6  2010     7     1     2205           1805       240       15           2024
## # ... with 13 more variables: arr_delay <int>, carrier <chr>, tailnum <chr>,
## #   flight <int>, origin <chr>, dest <chr>, air_time <int>, distance <int>,
## #   cancelled <int>, diverted <int>, hour <int>, minute <int>, time_hour <chr>
```

Exercise 22

```r
db_con <- dbConnect(drv = RSQLite::SQLite(),
                    dbname = ":memory:")
```

```r
dbWriteTable(conn = db_con,
             name = "whoTable",
             value = tidyr::who)

#Load the population table:
dbWriteTable(conn = db_con,
             name = "popTable",
             value = tidyr::population)
```

```r
# A

dbGetQuery(conn = db_con,
           statement = "SELECT country, year, new_sp_m014
                        FROM whoTable
                        LIMIT 10;")
```

```
##        country year new_sp_m014
## 1  Afghanistan 1980          NA
## 2  Afghanistan 1981          NA
## 3  Afghanistan 1982          NA
## 4  Afghanistan 1983          NA
## 5  Afghanistan 1984          NA
## 6  Afghanistan 1985          NA
## 7  Afghanistan 1986          NA
## 8  Afghanistan 1987          NA
## 9  Afghanistan 1988          NA
## 10 Afghanistan 1989          NA
```

```r
dbGetQuery(conn = db_con,
           statement = "SELECT country, year, new_sp_m014
                        FROM whoTable
                        WHERE country = 'United States of America' AND year >= 2010
                        LIMIT 10;")
```

```
##                    country year new_sp_m014
## 1 United States of America 2010           5
## 2 United States of America 2011          12
## 3 United States of America 2012          10
## 4 United States of America 2013          NA
```

Exercise 23

```
dbGetQuery(conn = db_con,
           statement = "SELECT country, year, AVG(new_sp_m014) AS avg_m014
                        FROM whoTable
                        WHERE year = 2013
                        GROUP BY country
                        ORDER BY avg_m014 ASC
                        LIMIT 10;")
```

```
##                country year avg_m014
## 1          Afghanistan 2013       NA
## 2              Albania 2013       NA
## 3              Algeria 2013       NA
## 4       American Samoa 2013       NA
## 5              Andorra 2013       NA
## 6               Angola 2013       NA
## 7             Anguilla 2013       NA
## 8  Antigua and Barbuda 2013       NA
## 9            Argentina 2013       NA
## 10             Armenia 2013       NA
```

Only got NAs here, not sure how to fix this in SQL.