Introduction to leveraging data technologies to model bigger datasets

USCOTS 2025 breakout session

Nicholas Horton (nhorton@amherst.edu) and Jo Hardin (jo.hardin@pomona.edu)

July 18, 2025

Data

What happens when our data start getting very big? Where should the data live? How are the data organized and formatted?

Tidy data

- data frame (R) or database / table (SQL)
- columns are variables
- rows are observational units

See Hadley Wickham's paper *Tidy data*, Wickham (2014).

What is a database?

- structured collection of data organized with
  - efficient storage
  - easy retrieval
  - consistent management
  - highly optimized for many tasks ("let the database do the work for you")
- e.g., SQL, Parquet, Arrow
- technology originally developed in the 1970's
- data stored in tables which are linked to one another via keys (called a relational database, think `join()`)

- often much faster to process data in a database than in a csv or tibble or data frame.

Where is a database?

- In the wild, database is typically remote
- Sometimes database is on hard drive
- Sometimes (here today), we use a serverless approach

Differences between data frames and databases

- databases (in **SQL**) can be arbitrarily large
  - live in storage (usually remote),
  - can live in the computer's hard drive
  - are optimized for common operations
- data frames (in **R**)
  - live in memory (RAM) on your personal computer
- multiple tables in a database are linked via keys which make them even more efficient.
- **accessing** a very large data frame is substantially slower than **accessing** a database.
- (**accessing** a small data frame is much faster than **accessing** a database.)
- a remote database has the advantage that many people can use it simultaneously

SQL: Structured Query Language

SQL is used to communicate with databases.

SQL is ubiquitous in the data world

- Learning some **SQL** is important.
- Teaching some **SQL** to your students is important.

. . .

You do not need to be an expert in **SQL** to get students up and running with a basic understanding. There is a lot of very low hanging fruit to teach students.

SQL dialects

SQL is a programming language for managing and querying data.

You may have heard of some of the following **SQL** dialects

- **MySQL**
- **SQLite**
- **DuckDB**
- **PostgreSQL**
- **Google BigQuery**
- **Microsoft Access SQL**

The dialects are very similar, and you will be able to quickly pick up whichever dialect your institution uses. In our work, we will use **DuckDB** (with a little bit of **MySQL**).

Today's example

The `airlines` database

Consider a database of all US flights between 2013 and 2015 The flight information was collected from the Bureau of Transportation Statistics, US Department of Transportation.

The database is a superset of the data in the `nycflights13` **R** package that tracks only flights departing airports serving New York City in 2013.

SQL connection

To set up a **SQL** connection, you need the location of the server (`host`) as well as a `user`name and `password`. For example, you may want to use the subset of of data from 2013 to 2015 which exists in a **MySQL** database hosted by Ben Baumer (see details in the online version of *Modern Data Science in R*).

```{r}
con_mysql <- DBI::dbConnect(
  RMariaDB::MariaDB(),
  dbname = "airlines",
  host = Sys.getenv("MDSR_HOST"),
  user = Sys.getenv("MDSR_USER"),
  password = Sys.getenv("MDSR_PWD")
)
```

Tables in `airlines` database

```
DBI::dbListTables(con_mysql)
```

```
[1] "airports"        "planes"          "carriers"        "flights_summary"
[5] "flights"
```

SQL tables as `tbl`

A `tbl` is different from a `tibble`. A `tbl` is a pointer to a remote object, and a `tibble` is a dataframe in your local environment.

```
carriers_tbl <- dplyr::tbl(con_mysql, "carriers")
dim(carriers_tbl)
```

```
[1] NA  2
```

```
head(carriers_tbl)
```

```
# Source:   SQL [?? x 2]
# Database: mysql  [mdsr_public@mdsr.crcbo51tmesf.us-east-2.rds.amazonaws.com:3306/airlines]
  carrier name
  <chr>   <chr>
1 02Q     Titan Airways
2 04Q     Tradewind Aviation
3 05Q     Comlux Aviation, AG
4 06Q     Master Top Linhas Aereas Ltd.
5 07Q     Flair Airlines Ltd.
6 09Q     Swift Air, LLC
```

SQL tables as `tibble`

The function `collect()` copies a **SQL** table from its server location to your local memory location in **R**.

```
carriers_tibble <- carriers_tbl |>
  dplyr::collect()

dim(carriers_tibble)
```

```
[1] 1610    2
```

```
head(carriers_tibble)
```

```
# A tibble: 6 x 2
  carrier name
  <chr>   <chr>
1 02Q     Titan Airways
2 04Q     Tradewind Aviation
3 05Q     Comlux Aviation, AG
4 06Q     Master Top Linhas Aereas Ltd.
5 07Q     Flair Airlines Ltd.
6 09Q     Swift Air, LLC
```

How much space does `carriers` take up?

The `tibble` (locally in **R**) takes up much more memory than the `tbl`, which just points to the remote object in **SQL**.

```
carriers_tbl |>
  object.size() |>
  print(units = "Kb")
```

```
5.3 Kb
```

```
carriers_tibble |>
  object.size() |>
  print(units = "Kb")
```

```
234.8 Kb
```

Using SQL in Quarto

Using the **DBI** package, we can send **SQL** queries through an `r` chunk.

SQL queries via the DBI package

- Look at the first few rows of the `flights` data. Equivalent to `head()`.

````
```{r}
DBI::dbGetQuery(con_mysql,
                "SELECT * FROM flights LIMIT 8;")
```
````

```
  year month day dep_time sched_dep_time dep_delay arr_time sched_arr_time
1 2013    10   1        2             10        -8      453            505
2 2013    10   1        4           2359         5      730            729
3 2013    10   1       11             15        -4      528            530
4 2013    10   1       14           2355        19      544            540
5 2013    10   1       16             17        -1      515            525
6 2013    10   1       22             20         2      552            554
7 2013    10   1       29             35        -6      808            816
8 2013    10   1       29             35        -6      449            458
  arr_delay carrier tailnum flight origin dest air_time distance cancelled
1       -12      AA  N201AA   2400    LAX  DFW      149     1235         0
2         1      FL  N344AT    710    SFO  ATL      247     2139         0
3        -2      AA  N3KMAA   1052    SFO  DFW      182     1464         0
4         4      AA  N3ENAA   2392    SEA  ORD      191     1721         0
5       -10      UA  N38473   1614    LAX  IAH      157     1379         0
6        -2      UA  N458UA    291    SFO  IAH      188     1635         0
7        -8      US  N551UW    436    LAX  CLT      256     2125         0
8        -9      AS  N402AS    108    ANC  SEA      181     1448         0
  diverted hour minute           time_hour
1        0    0     10 2013-10-01 00:10:00
2        0   23     59 2013-10-01 23:59:00
3        0    0     15 2013-10-01 00:15:00
4        0   23     55 2013-10-01 23:55:00
5        0    0     17 2013-10-01 00:17:00
6        0    0     20 2013-10-01 00:20:00
7        0    0     35 2013-10-01 00:35:00
8        0    0     35 2013-10-01 00:35:00
```

SQL queries via the DBI package

- How many flights per year are in the `flights` table?

```
```{r}
#| cache: true

DBI::dbGetQuery(con_mysql,
  "SELECT year, COUNT(*) AS num_flights
  FROM flights
  GROUP BY year
  ORDER BY num_flights;")
```
```

```
  year num_flights
1 2015     5819079
2 2014     5819811
3 2013     6369482
```

SQL queries via the DBI package

- What is the average Departure Delay (and number of flights) for each destination?

```
```{r}
#| cache: true

DBI::dbGetQuery(con_mysql,
  "SELECT dest,
          AVG(dep_delay) AS mean_delay,
          COUNT(*) AS num_flights
   FROM flights
   GROUP BY dest
   LIMIT 8;")
```
```

```
  dest mean_delay num_flights
1  ABE     9.9056        7253
2  ABI     8.9290        8114
3  ABQ    11.7639       74189
4  ABR     2.4078        2239
5  ABY     9.5748        3001
```

```
6  ACK     5.5521         1295
7  ACT    10.2526         5225
8  ACV    12.5367         7785
```

Good practice

Always a good idea to terminate the **SQL** connection when you are done with it.

```
RMariaDB::dbDisconnect(con_mysql, shutdown = TRUE)
```

Parquet

Setting up a database can be painful (and potentially expensive).

Parquet files can be downloaded into local hard drive to act as a database server that can be queried using **SQL** code!

Parquet files

- Unlike a .csv file, a .parquet file is not human readable
- Contains lots of information about the columns
- Very efficient storage of information
- Can access .parquet files using SQL syntax
- Allows a serverless setup

Why use parquet instead of SQL?

- Setting up a SQL server can
  - be painful
  - be potentially expensive ($ $ $)
  - be potentially expensive (time)
  - have security issues
- Querying a set of parquet files is seamless.
- Easy to do in a breakout format like today

Local (hard drive) parquet files

```
dir.create("data_airlines_24")
dir.create("data_airlines_24/Year=2024", showWarnings = FALSE)

out <- curl::multi_download("https://blobs.duckdb.org/flight-data-partitioned/Year=2024/data_
                            "data_airlines_24/Year=2024/data_0.parquet",
                            resume = TRUE)
```

Connecting to the parquet files via DuckDB

Start an in-memory database using DuckDB. The function `duckdb()` comes from the **duckdb** package.

```
con_parq_24 <- DBI::dbConnect(duckdb())
```

Looking at the database

- Look at the first few rows of the parquet data. Equivalent to `head()`.

```
```{r}
DBI::dbGetQuery(con_parq_24,
                "SELECT *
                FROM read_parquet('data_airlines_24/Year*/*.parquet')
                LIMIT 8;")
```
```

```
  Year Quarter Month DayofMonth DayOfWeek FlightDate Reporting_Airline
1 2024       1     1          8         1 2024-01-08                9E
2 2024       1     1          9         2 2024-01-09                9E
3 2024       1     1         10         3 2024-01-10                9E
4 2024       1     1         11         4 2024-01-11                9E
5 2024       1     1         12         5 2024-01-12                9E
6 2024       1     1         15         1 2024-01-15                9E
7 2024       1     1         16         2 2024-01-16                9E
8 2024       1     1         17         3 2024-01-17                9E
  DOT_ID_Reporting_Airline IATA_CODE_Reporting_Airline Tail_Number
1                    20363                          9E      N485PX
2                    20363                          9E      N912XJ
```

```
3                    20363                          9E      N918XJ
4                    20363                          9E      N490PX
5                    20363                          9E      N915XJ
6                    20363                          9E      N485PX
7                    20363                          9E      N485PX
8                    20363                          9E      N482PX
  Flight_Number_Reporting_Airline OriginAirportID OriginAirportSeqID
1                            4801           12953            1295304
2                            4801           12953            1295304
3                            4801           12953            1295304
4                            4801           12953            1295304
5                            4801           12953            1295304
6                            4801           12953            1295304
7                            4801           12953            1295304
8                            4801           12953            1295304
  OriginCityMarketID Origin OriginCityName OriginState OriginStateFips
1              31703    LGA   New York, NY          NY              36
2              31703    LGA   New York, NY          NY              36
3              31703    LGA   New York, NY          NY              36
4              31703    LGA   New York, NY          NY              36
5              31703    LGA   New York, NY          NY              36
6              31703    LGA   New York, NY          NY              36
7              31703    LGA   New York, NY          NY              36
8              31703    LGA   New York, NY          NY              36
  OriginStateName OriginWac DestAirportID DestAirportSeqID DestCityMarketID
1        New York        22         13871          1387102            33316
2        New York        22         13871          1387102            33316
3        New York        22         13871          1387102            33316
4        New York        22         13871          1387102            33316
5        New York        22         13871          1387102            33316
6        New York        22         13871          1387102            33316
7        New York        22         13871          1387102            33316
8        New York        22         13871          1387102            33316
  Dest DestCityName DestState DestStateFips DestStateName DestWac CRSDepTime
1  OMA    Omaha, NE        NE            31      Nebraska      65       0856
2  OMA    Omaha, NE        NE            31      Nebraska      65       0856
3  OMA    Omaha, NE        NE            31      Nebraska      65       0856
4  OMA    Omaha, NE        NE            31      Nebraska      65       0856
5  OMA    Omaha, NE        NE            31      Nebraska      65       0856
6  OMA    Omaha, NE        NE            31      Nebraska      65       0856
7  OMA    Omaha, NE        NE            31      Nebraska      65       0856
8  OMA    Omaha, NE        NE            31      Nebraska      65       0856
  DepTime DepDelay DepDelayMinutes DepDel15 DepartureDelayGroups DepTimeBlk
```

1	0851	-5	0	0	-1	0800-0859	
2	0851	-5	0	0	-1	0800-0859	
3	0850	-6	0	0	-1	0800-0859	
4	0919	23	23	1	1	0800-0859	
5	0851	-5	0	0	-1	0800-0859	
6	0919	23	23	1	1	0800-0859	
7	<NA>	NA	NA	NA	NA	0800-0859	
8	0923	27	27	1	1	0800-0859	

	TaxiOut	WheelsOff	WheelsOn	TaxiIn	CRSArrTime	ArrTime	ArrDelay	ArrDelayMinutes
1	25.00	0916	1120	4.00	1135	1124	-11	0
2	16.00	0907	1055	12.00	1135	1107	-28	0
3	17.00	0907	1104	6.00	1135	1110	-25	0
4	27.00	0946	1154	8.00	1135	1202	27	27
5	24.00	0915	1120	17.00	1135	1137	2	2
6	12.00	0931	1138	7.00	1135	1145	10	10
7	<NA>	<NA>	<NA>	<NA>	1135	<NA>	NA	NA
8	24.00	0947	1153	13.00	1135	1206	31	31

	ArrDel15	ArrivalDelayGroups	ArrTimeBlk	Cancelled	CancellationCode	Diverted
1	0	-1	1100-1159	0	<NA>	0
2	0	-2	1100-1159	0	<NA>	0
3	0	-2	1100-1159	0	<NA>	0
4	1	1	1100-1159	0	<NA>	0
5	0	0	1100-1159	0	<NA>	0
6	0	0	1100-1159	0	<NA>	0
7	NA	NA	1100-1159	1	B	0
8	1	2	1100-1159	0	<NA>	0

	CRSElapsedTime	ActualElapsedTime	AirTime	Flights	Distance	DistanceGroup
1	219	213	184.00	1	1148	5
2	219	196	168.00	1	1148	5
3	219	200	177.00	1	1148	5
4	219	223	188.00	1	1148	5
5	219	226	185.00	1	1148	5
6	219	206	187.00	1	1148	5
7	219	NA	<NA>	1	1148	5
8	219	223	186.00	1	1148	5

	CarrierDelay	WeatherDelay	NASDelay	SecurityDelay	LateAircraftDelay
1	<NA>	<NA>	<NA>	<NA>	<NA>
2	<NA>	<NA>	<NA>	<NA>	<NA>
3	<NA>	<NA>	<NA>	<NA>	<NA>
4	0.00	0.00	4.00	0.00	23.00
5	<NA>	<NA>	<NA>	<NA>	<NA>
6	<NA>	<NA>	<NA>	<NA>	<NA>
7	<NA>	<NA>	<NA>	<NA>	<NA>

```
8        27.00         0.00     4.00          0.00              0.00
  FirstDepTime TotalAddGTime LongestAddGTime DivAirportLandings DivReachedDest
1         <NA>          <NA>            <NA>                  0           <NA>
2         <NA>          <NA>            <NA>                  0           <NA>
3         <NA>          <NA>            <NA>                  0           <NA>
4         <NA>          <NA>            <NA>                  0           <NA>
5         <NA>          <NA>            <NA>                  0           <NA>
6         <NA>          <NA>            <NA>                  0           <NA>
7         <NA>          <NA>            <NA>                  0           <NA>
8         <NA>          <NA>            <NA>                  0           <NA>
  DivActualElapsedTime DivArrDelay DivDistance Div1Airport Div1AirportID
1                 <NA>        <NA>        <NA>        <NA>          <NA>
2                 <NA>        <NA>        <NA>        <NA>          <NA>
3                 <NA>        <NA>        <NA>        <NA>          <NA>
4                 <NA>        <NA>        <NA>        <NA>          <NA>
5                 <NA>        <NA>        <NA>        <NA>          <NA>
6                 <NA>        <NA>        <NA>        <NA>          <NA>
7                 <NA>        <NA>        <NA>        <NA>          <NA>
8                 <NA>        <NA>        <NA>        <NA>          <NA>
  Div1AirportSeqID Div1WheelsOn Div1TotalGTime Div1LongestGTime Div1WheelsOff
1             <NA>         <NA>           <NA>             <NA>          <NA>
2             <NA>         <NA>           <NA>             <NA>          <NA>
3             <NA>         <NA>           <NA>             <NA>          <NA>
4             <NA>         <NA>           <NA>             <NA>          <NA>
5             <NA>         <NA>           <NA>             <NA>          <NA>
6             <NA>         <NA>           <NA>             <NA>          <NA>
7             <NA>         <NA>           <NA>             <NA>          <NA>
8             <NA>         <NA>           <NA>             <NA>          <NA>
  Div1TailNum Div2Airport Div2AirportID Div2AirportSeqID Div2WheelsOn
1        <NA>        <NA>          <NA>             <NA>         <NA>
2        <NA>        <NA>          <NA>             <NA>         <NA>
3        <NA>        <NA>          <NA>             <NA>         <NA>
4        <NA>        <NA>          <NA>             <NA>         <NA>
5        <NA>        <NA>          <NA>             <NA>         <NA>
6        <NA>        <NA>          <NA>             <NA>         <NA>
7        <NA>        <NA>          <NA>             <NA>         <NA>
8        <NA>        <NA>          <NA>             <NA>         <NA>
  Div2TotalGTime Div2LongestGTime Div2WheelsOff Div2TailNum Div3Airport
1           <NA>             <NA>          <NA>        <NA>        <NA>
2           <NA>             <NA>          <NA>        <NA>        <NA>
3           <NA>             <NA>          <NA>        <NA>        <NA>
4           <NA>             <NA>          <NA>        <NA>        <NA>
5           <NA>             <NA>          <NA>        <NA>        <NA>
```

```
6           <NA>             <NA>          <NA>        <NA>        <NA>
7           <NA>             <NA>          <NA>        <NA>        <NA>
8           <NA>             <NA>          <NA>        <NA>        <NA>
  Div3AirportID Div3AirportSeqID Div3WheelsOn Div3TotalGTime Div3LongestGTime
1          <NA>             <NA>         <NA>           <NA>             <NA>
2          <NA>             <NA>         <NA>           <NA>             <NA>
3          <NA>             <NA>         <NA>           <NA>             <NA>
4          <NA>             <NA>         <NA>           <NA>             <NA>
5          <NA>             <NA>         <NA>           <NA>             <NA>
6          <NA>             <NA>         <NA>           <NA>             <NA>
7          <NA>             <NA>         <NA>           <NA>             <NA>
8          <NA>             <NA>         <NA>           <NA>             <NA>
  Div3WheelsOff Div3TailNum Div4Airport Div4AirportID Div4AirportSeqID
1          <NA>        <NA>        <NA>          <NA>             <NA>
2          <NA>        <NA>        <NA>          <NA>             <NA>
3          <NA>        <NA>        <NA>          <NA>             <NA>
4          <NA>        <NA>        <NA>          <NA>             <NA>
5          <NA>        <NA>        <NA>          <NA>             <NA>
6          <NA>        <NA>        <NA>          <NA>             <NA>
7          <NA>        <NA>        <NA>          <NA>             <NA>
8          <NA>        <NA>        <NA>          <NA>             <NA>
  Div4WheelsOn Div4TotalGTime Div4LongestGTime Div4WheelsOff Div4TailNum
1         <NA>           <NA>             <NA>          <NA>        <NA>
2         <NA>           <NA>             <NA>          <NA>        <NA>
3         <NA>           <NA>             <NA>          <NA>        <NA>
4         <NA>           <NA>             <NA>          <NA>        <NA>
5         <NA>           <NA>             <NA>          <NA>        <NA>
6         <NA>           <NA>             <NA>          <NA>        <NA>
7         <NA>           <NA>             <NA>          <NA>        <NA>
8         <NA>           <NA>             <NA>          <NA>        <NA>
  Div5Airport Div5AirportID Div5AirportSeqID Div5WheelsOn Div5TotalGTime
1        <NA>          <NA>             <NA>         <NA>           <NA>
2        <NA>          <NA>             <NA>         <NA>           <NA>
3        <NA>          <NA>             <NA>         <NA>           <NA>
4        <NA>          <NA>             <NA>         <NA>           <NA>
5        <NA>          <NA>             <NA>         <NA>           <NA>
6        <NA>          <NA>             <NA>         <NA>           <NA>
7        <NA>          <NA>             <NA>         <NA>           <NA>
8        <NA>          <NA>             <NA>         <NA>           <NA>
  Div5LongestGTime Div5WheelsOff Div5TailNum column109
1             <NA>          <NA>        <NA>      <NA>
2             <NA>          <NA>        <NA>      <NA>
3             <NA>          <NA>        <NA>      <NA>
```

```
4             <NA>          <NA>        <NA>      <NA>
5             <NA>          <NA>        <NA>      <NA>
6             <NA>          <NA>        <NA>      <NA>
7             <NA>          <NA>        <NA>      <NA>
8             <NA>          <NA>        <NA>      <NA>
```

SQL queries via the DBI package

- What is the average Departure Delay (and number of flights) for each destination?

````
```{r}
#| cache: true

DBI::dbGetQuery(con_parq_24,
                "SELECT Dest,
                        MEDIAN(DepDelay) AS med_delay,
                        AVG(DepDelay) AS mean_delay,
                        COUNT(*) AS num_flights
                FROM read_parquet('data_airlines_24/Year*/*.parquet')
                GROUP BY Dest
                LIMIT 8;")
```
````

```
  Dest med_delay mean_delay num_flights
1  RSW        -2   14.45851       20337
2  STL        -1   12.13676       31917
3  OMA        -1   14.32007       11775
4  LIT        -1   13.69294        6085
5  ROC        -2   12.83086        5766
6  FLL         0   17.75766       49267
7  CMH        -2   11.82598       20233
8  OKC        -1   13.55237       10992
```

Good practice

Always a good idea to terminate the **SQL** connection when you are done with it.

```
RMariaDB::dbDisconnect(con_parq_24, shutdown = TRUE)
```

References

Wickham, Hadley. 2014. "Tidy Data." *Journal of Statistical Software* 59 (10): 1–23. https://doi.org/10.18637/jss.v059.i10.