Introduction to Relational Databases

Max Turgeon

SCI 2000–Introduction to Data Science

Lecture Objectives

- Learn the main features of a relational database.
- Query data from such a database using R.'
- Explain the differences between importing data in R and querying a relational database.

Motivation

- Last week, we talked about relational data.
    - Related datasets with a variable in common encoding the relationship.
- Most businesses collect relational data.
    - Employee data and scheduling information
    - Product, client and order data
- **Relational databases** are the main way of efficiently storing this relational data.
    - For easy maintenance and accuracy.
    - For fast retrieval.

Disclaimer

- There is no way we can have a thorough discussion of relational databases in SCI 2000.
    - If you're interested, you should look at COMP 3380
- Topics we will **not** discuss:
    - SQL (Structured Query Language)
    - Database design
    - Relational algebra

Basic vocabulary

- A **relational database** is a collection of tables.
- A **table** is a collection of records.
- A **record** is a collection of attributes.
- An **attribute** is a piece of information that we want to capture.
    - Could be a string, a date-time, an integer, a floating point number, etc
    - Could even be an image or an audio file.
- Typically, each record has a special attribute called a **primary key** to uniquely identify it within the table, and to refer to it in a *different* table.
    - If a record contains the primary key of another table, it is called a **foreign key**.

Example

Product code	Description	Price
A416	Nails, box	0.14$
C923	Drawing pins, box	0.08$

- This is a table, with two records.
- Each record has three attributes, and “Product code” is the primary key.

Invoice code	Invoice line	Product code	Quantity
3804	1	A416	10
3804	2	C923	15

- This is another table, also with two records.
- Each record has four attributes, and “Product code” is a foreign key.
- This table does **not** have a primary key (because “Invoice code” is not unique).

- You could imagine a separate table containing information specific to each invoice.
    - E.g. Date of the order, customer number, invoice total, whether it has been paid, etc.
- Similarly, you could imagine another table containing information on customers.
    - E.g. Name, address, phone number, etc.

How is a database different than a collection of CSV files?

- The acronym RDBMS stands for **Relational Database Management System**.
- It consists of the database itself and the *software* necessary to its functionality.
- In particular, a RDBMS has been optimized to perform certain tasks accurately and efficiently.
    - Data accuracy: make sure attributes satisfy certain conditions (e.g. a quantity should be non-negative).
    - Data retrieval: extract necessary data as quickly as possible.
    - Access: control who can change the data and who can query it.

Databases and R

- R allows us to connect to databases, retrieve information about the tables and attributes, and query the data.
- You can treat the tables as `data.frame`s, use functions like `filter`, `mutate`, `summarise`, and R will transform your code into a query that the database can understand.
- **Important principle**: RDBMS have been optimized to run queries fast, so we want to push as much of these computations to the database as we can, as opposed to extracting *all* the data into R and working directly with the `data.frame`s.

Example i

```
library(DBI)
# duckdb is a type of RDBMS
library(duckdb)
```

Example ii

```
# 1. Create a connection
con <- dbConnect(duckdb())

# This database is actually empty...
# We fill it in with the data from nycflights13
library(nycflights13)
duckdb_register(conn = con, name = "flights",
                df = flights)
```

Example iii

```
library(tidyverse)

tbl(con, "flights") %>%
  group_by(dest) %>%
  summarise(delay = mean(dep_time, na.rm = TRUE))
```

```
## # Source:   lazy query [?? x 2]
## # Database: duckdb_connection
##    dest  delay
##    <chr> <dbl>
##  1 IAH   1266.
```

Example iv

```
##  2 MIA   1245.
##  3 BQN   1375.
##  4 ATL   1293.
##  5 ORD   1310.
##  6 FLL   1327.
##  7 IAD   1306.
##  8 MCO   1337.
##  9 PBI   1335.
## 10 TPA   1346.
## # ... with more rows
```

Example v

```
# To turn the output into a data frame
# use collect()
tbl(con, "flights") %>%
  group_by(dest) %>%
  summarise(delay = mean(dep_time, na.rm = TRUE)) %>%
  collect()
```

```
## # A tibble: 105 x 2
##    dest  delay
##    <chr> <dbl>
##  1 IAH   1266.
```

Example vi

```
##  2 MIA   1245.
##  3 BQN   1375.
##  4 ATL   1293.
##  5 ORD   1310.
##  6 FLL   1327.
##  7 IAD   1306.
##  8 MCO   1337.
##  9 PBI   1335.
## 10 TPA   1346.
## # ... with 95 more rows
```

Exercise i

```
# Let's revisit an example from last week
# where we looked at average delay

# First, add planes table
duckdb_register(conn = con, name = "planes",
                df = planes)
```

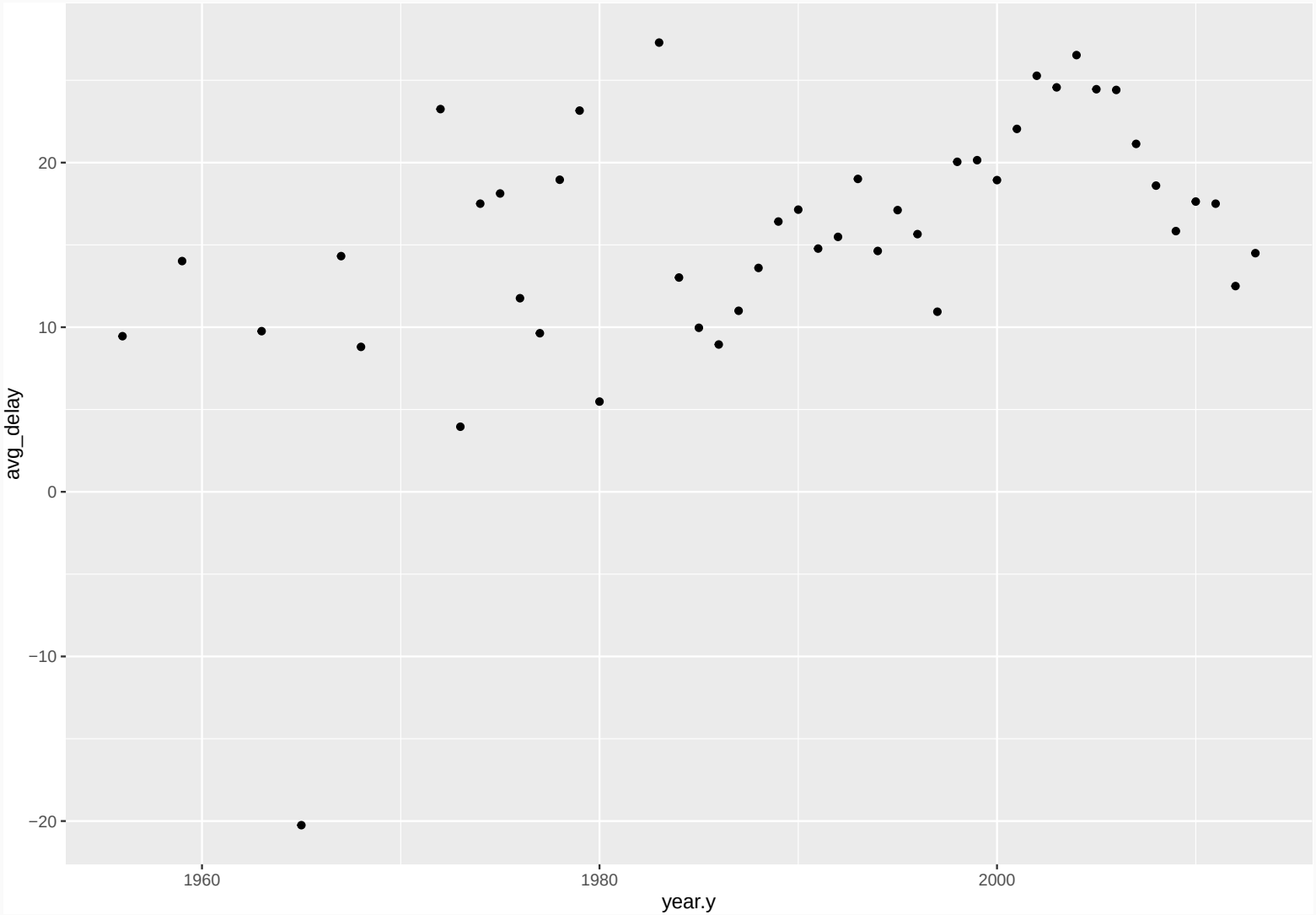
Exercise ii

```
# Replace data.frames by tbl(con, name)
data_avg <- left_join(tbl(con, "flights"),
                      tbl(con, "planes"),
                      by = "tailnum") %>%
  mutate(tot_delay = dep_delay + arr_delay) %>%
  group_by(year.y) %>%
  summarise(avg_delay = mean(tot_delay)) %>%
  collect() # Collect at the end for efficiency
```

Exercise iii

```
data_avg %>%
  ggplot(aes(x = year.y,
             y = avg_delay)) +
  geom_point()
```

Exercise iv



Summary

1. Create a connection to the database using `dbConnect`.
    - Requires we know what type of RDBMS (e.g DuckDB, MySQL, Oracle) it is.
2. Work with tables like `data.frame`s using `tbl(con, name)`.
3. Only once data manipulation is done, `collect` the results.

Live demo

For the live demo, I will use a SQLite database named `database.db` that you can find on UM Learn.