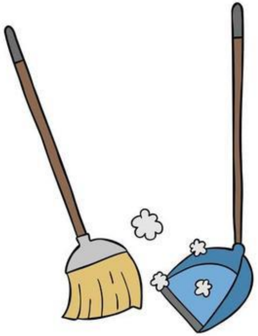
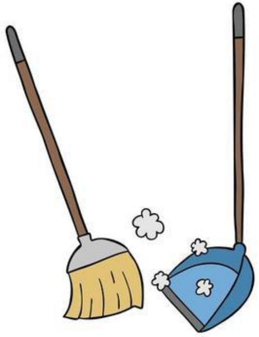
Library Carpentry



# Tidy Data



Corey DiPietro
Smithsonian Digitization Program Office

2025-01-29

# Summary and Setup

Library Carpentry aims to teach librarians, information professionals, and researchers basic concepts, skills, and tools for working with data so that they can get more done in less time, and with less pain. This lesson was designed to teach the principles of working with data in spreadsheets.

### Lesson purpose

The purpose of this lesson is not to teach how to do data analysis in spreadsheets, but to teach good data organization and how to do some data cleaning and quality control in a spreadsheet program.

https://librarycarpentry.github.io/lc-spreadsheets/

# Summary and Setup

## Prerequisites

This lesson requires working copies of the data, linked below, as well as a working spreadsheet program, such as Microsoft Excel, LibreOffice, or another program described below.

To interact with spreadsheets, we can use LibreOffice, Microsoft Excel, Gnumeric, OpenOffice.org, or other programs. Google Sheets will also work.

Commands may differ a bit between programs, but general ideas for thinking about spreadsheets is the same.

## Dataset

**Download** this data file to your computer: https://librarycarpentry.github.io/lc-spreadsheets/data/training_attendance.xlsx

The file is an `xlsx` file that should automatically download. You may need to right click or control click in order to save the file (NOTE: In Safari, right click and select **download linked file**; in Chrome and Firefox, right click and select **save link as**).

Make a note of the location (i.e the folder, your desktop) to which you save the file.

# Using spreadsheet programs for data organization

Good **data organization** is the foundation of much of our day-to-day work in libraries, museums, and archives. Most of us have data or do data entry in spreadsheets. Spreadsheet programs are very useful for designing data tables and handling very basic data quality control functions.

# Spreadsheet outline

## In this lesson, we will look at:

- Good data entry practices
- Avoiding common formatting mistakes
- Dates as data
- Basic quality control and data manipulation in spreadsheets
- Exporting data from spreadsheets

## Utilizing tidy data principles facilitates the following common tasks in our field:

- Data analysis
- Data cleanup
- Statistics
- Plotting

# Spreadsheet outline

**What this lesson will not teach you**

- How to do *statistics* in a spreadsheet
- How to do *plotting* or visualization in a spreadsheet
- How to do data analysis in a spreadsheet
- How to *write code* in spreadsheet programs

# Questions (1 of 2)

- How many people have used spreadsheets in their work?
- What kind of operations do you do in spreadsheets?
- Which ones do you think spreadsheets are good for?

# Questions (2 of 2)

- Spreadsheets can be very useful, but they can also be frustrating and even sometimes give us incorrect results. What are some things that you’ve accidentally done in a spreadsheet, or have been frustrated that you can’t do easily?

# Problems with Spreadsheets

- Spreadsheets are good for data entry, but we sometimes use them for more than just data entry
- Things like:
  - Formatted and labelled tables for publications and reports
  - Summary statistics and figures

# Problems with Spreadsheets

- Why can this cause issues?
- Information is formatted in a way that’s not really meant to be read as data
- For example:
  - Merged cells
  - Colored cells
  - Notes and notations

# Problems with Spreadsheets

- There’s nothing wrong with using a spreadsheet in this way, but one should understand the inherent limitations of doing so, especially if you may need to use the data for subsetting and sorting, statistics, plotting, or interoperability with other systems
- **Understand and anticipate the current and future uses of your data**

# Lesson Topics for Today

- Formatting data tables in spreadsheets
- Formatting problems
- Dates as data
- Basic quality control and data manipulation in spreadsheets
- Exporting data from spreadsheets
- Data export formats caveats

# Formatting data tables in Spreadsheets

- **Data organization is the foundation of any data-related work.**
- But we first have to set up our data for the computer to be able to understand it.
- It’s important to set up well-formatted data tables from the outset **before** you even start collecting data to analyse.
- Unorganized data can make it harder to work with your data
- Aim to organize your data in a way that allows other programs and people to easily understand and use the data.

# Structuring data in spreadsheets

## The cardinal rules of using spreadsheet programs for data:

1) **Leave the raw data raw** - don’t mess with it!
2) Put all your **variables in columns** –
    i. These are the things you’re measuring, like ‘date’ or ‘length’ or ‘attendance’.
3) Put each **observation or instance in its own row**.
4) **Don’t combine multiple pieces of information in one cell**.
5) Export the cleaned data to a **text based format** like CSV. This ensures that anyone can use the data, and is the format required by most data repositories.

# Structuring data in spreadsheets

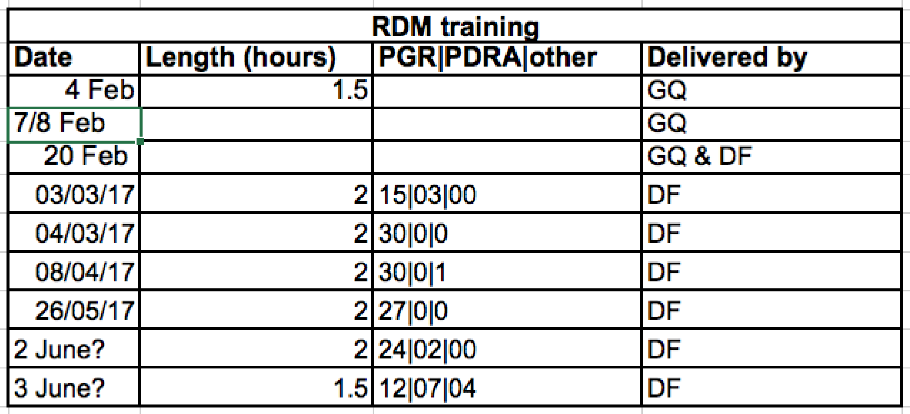
## Example:

Data from attendance and instruction for previous research data management workshops.

Different people have entered data into a single spreadsheet.

They keep track of things like date, number of attendees, and who delivered the workshop.

If they were to keep track of the data like this, then what are the potential issues with the spreadsheet?



| RDM training | | | |
|---|---|---|---|
| **Date** | **Length (hours)** | **PGR\|PDRA\|other** | **Delivered by** |
| 4 Feb | 1.5 | | GQ |
| 7/8 Feb | | | GQ |
| 20 Feb | | | GQ & DF |
| 03/03/17 | 2 | 15\|03\|00 | DF |
| 04/03/17 | 2 | 30\|0\|0 | DF |
| 08/04/17 | 2 | 30\|0\|1 | DF |
| 26/05/17 | 2 | 27\|0\|0 | DF |
| 2 June? | 2 | 24\|02\|00 | DF |
| 3 June? | 1.5 | 12\|07\|04 | DF |

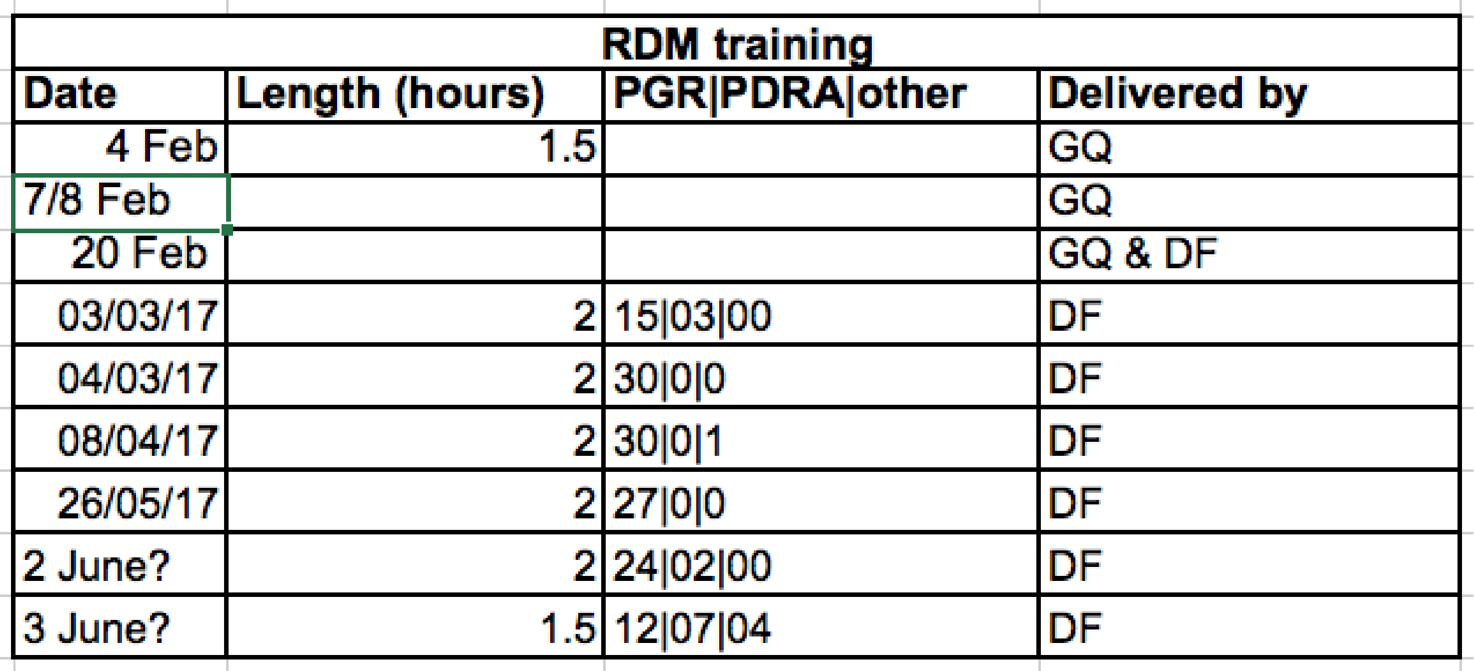
# Structuring data in spreadsheets (again)

## The cardinal rules of using spreadsheet programs for data:

1) **Leave the raw data raw** - don’t mess with it!
2) Put all your **variables in columns** –
   i. There are the things you’re measuring, like ‘date’ or ‘length’ or ‘attendance’.
3) Put each **observation or instance in its own row**.
4) **Don’t combine multiple pieces of information in one cell**.
5) Export the cleaned data to a **text based format** like CSV. This ensures that anyone can use the data, and is the format required by most data repositories.

# Structuring data in spreadsheets

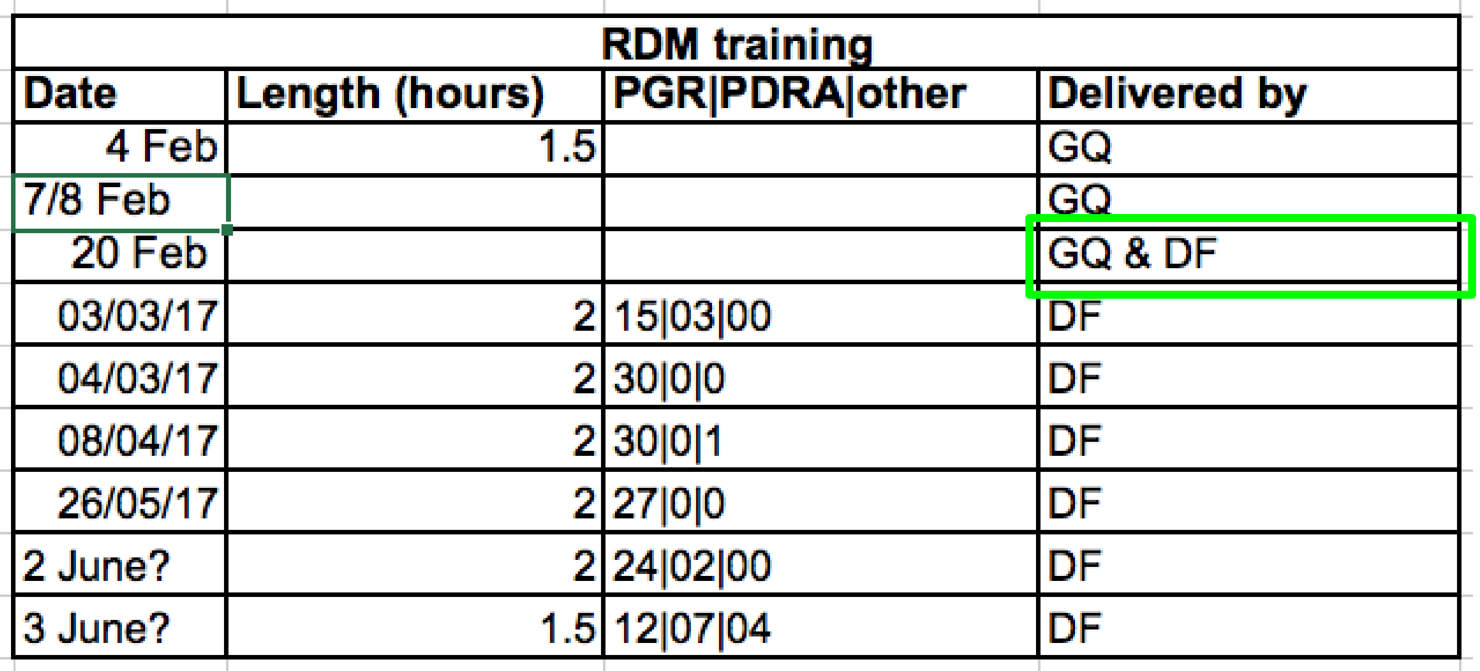
What are the potential issues with this spreadsheet?



<table>
<tr><th colspan="4">RDM training</th></tr>
<tr><th>Date</th><th>Length (hours)</th><th>PGR|PDRA|other</th><th>Delivered by</th></tr>
<tr><td>4 Feb</td><td>1.5</td><td></td><td>GQ</td></tr>
<tr><td>7/8 Feb</td><td></td><td></td><td>GQ</td></tr>
<tr><td>20 Feb</td><td></td><td></td><td>GQ & DF</td></tr>
<tr><td>03/03/17</td><td>2</td><td>15|03|00</td><td>DF</td></tr>
<tr><td>04/03/17</td><td>2</td><td>30|0|0</td><td>DF</td></tr>
<tr><td>08/04/17</td><td>2</td><td>30|0|1</td><td>DF</td></tr>
<tr><td>26/05/17</td><td>2</td><td>27|0|0</td><td>DF</td></tr>
<tr><td>2 June?</td><td>2</td><td>24|02|00</td><td>DF</td></tr>
<tr><td>3 June?</td><td>1.5</td><td>12|07|04</td><td>DF</td></tr>
</table>

# Structuring data in spreadsheets

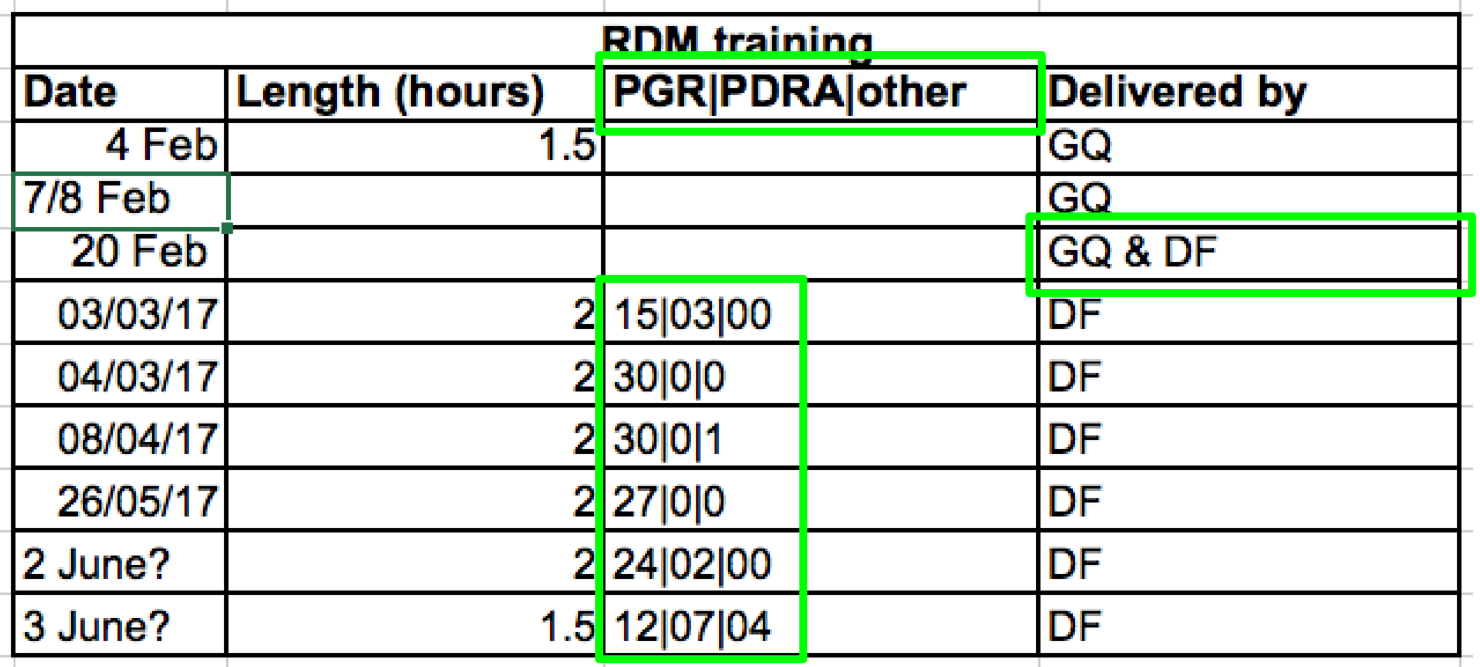
What are the potential issues with this spreadsheet?



| RDM training | | | |
|---|---|---|---|
| Date | Length (hours) | PGR\|PDRA\|other | Delivered by |
| 4 Feb | 1.5 | | GQ |
| 7/8 Feb | | | GQ |
| 20 Feb | | | GQ & DF |
| 03/03/17 | 2 | 15\|03\|00 | DF |
| 04/03/17 | 2 | 30\|0\|0 | DF |
| 08/04/17 | 2 | 30\|0\|1 | DF |
| 26/05/17 | 2 | 27\|0\|0 | DF |
| 2 June? | 2 | 24\|02\|00 | DF |
| 3 June? | 1.5 | 12\|07\|04 | DF |

# Structuring data in spreadsheets

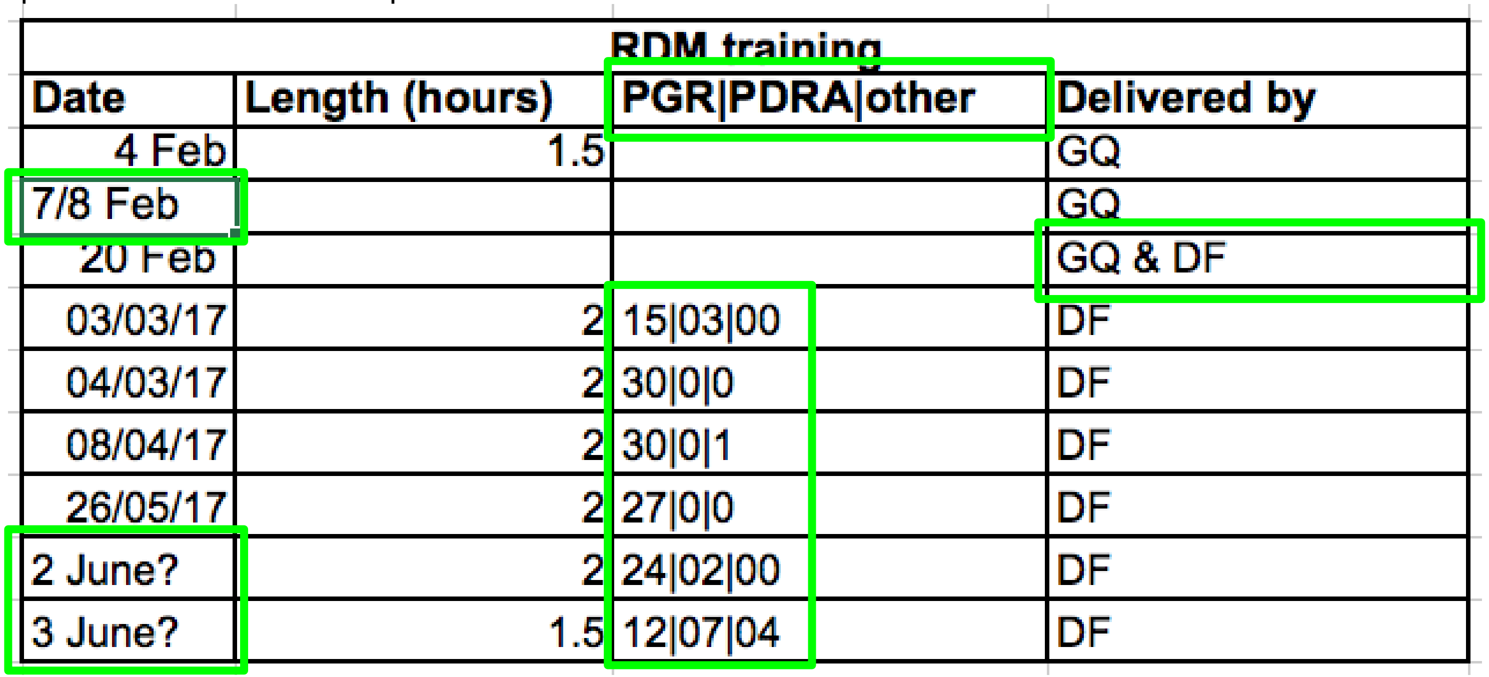
What are the potential issues with this spreadsheet?



| RDM training | | | |
|---|---|---|---|
| Date | Length (hours) | PGR\|PDRA\|other | Delivered by |
| 4 Feb | 1.5 | | GQ |
| 7/8 Feb | | | GQ |
| 20 Feb | | | GQ & DF |
| 03/03/17 | 2 | 15\|03\|00 | DF |
| 04/03/17 | 2 | 30\|0\|0 | DF |
| 08/04/17 | 2 | 30\|0\|1 | DF |
| 26/05/17 | 2 | 27\|0\|0 | DF |
| 2 June? | 2 | 24\|02\|00 | DF |
| 3 June? | 1.5 | 12\|07\|04 | DF |

# Structuring data in spreadsheets

What are the potential issues with this spreadsheet?

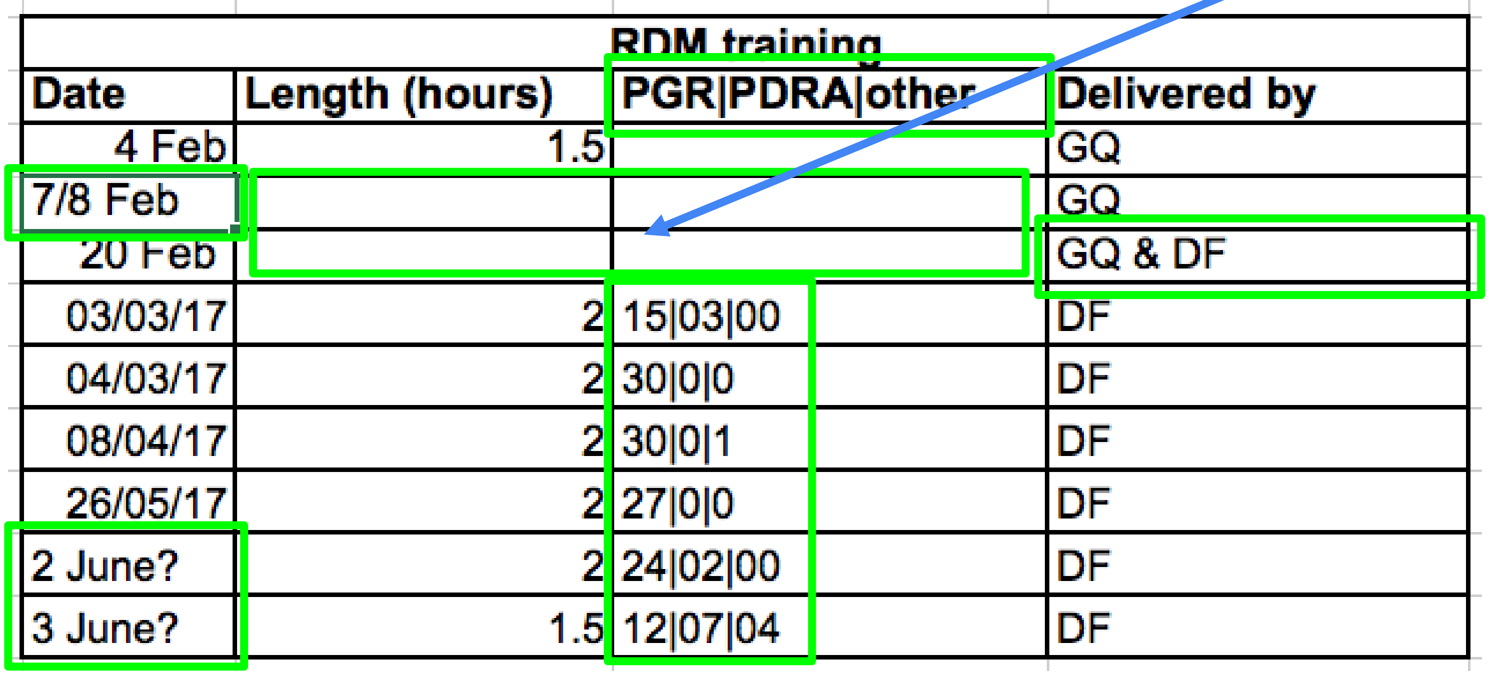


<table>
<tr><th colspan="4">RDM training</th></tr>
<tr><th>Date</th><th>Length (hours)</th><th>PGR|PDRA|other</th><th>Delivered by</th></tr>
<tr><td>4 Feb</td><td>1.5</td><td></td><td>GQ</td></tr>
<tr><td>7/8 Feb</td><td></td><td></td><td>GQ</td></tr>
<tr><td>20 Feb</td><td></td><td></td><td>GQ & DF</td></tr>
<tr><td>03/03/17</td><td>2</td><td>15|03|00</td><td>DF</td></tr>
<tr><td>04/03/17</td><td>2</td><td>30|0|0</td><td>DF</td></tr>
<tr><td>08/04/17</td><td>2</td><td>30|0|1</td><td>DF</td></tr>
<tr><td>26/05/17</td><td>2</td><td>27|0|0</td><td>DF</td></tr>
<tr><td>2 June?</td><td>2</td><td>24|02|00</td><td>DF</td></tr>
<tr><td>3 June?</td><td>1.5</td><td>12|07|04</td><td>DF</td></tr>
</table>

# Structuring data in spreadsheets

The uncertainty of blanks!

What are the potential issues with this spreadsheet?



<table>
<tr><th colspan="4">RDM training</th></tr>
<tr><th>Date</th><th>Length (hours)</th><th>PGR|PDRA|other</th><th>Delivered by</th></tr>
<tr><td>4 Feb</td><td>1.5</td><td></td><td>GQ</td></tr>
<tr><td>7/8 Feb</td><td></td><td></td><td>GQ</td></tr>
<tr><td>20 Feb</td><td></td><td></td><td>GQ & DF</td></tr>
<tr><td>03/03/17</td><td>2</td><td>15|03|00</td><td>DF</td></tr>
<tr><td>04/03/17</td><td>2</td><td>30|0|0</td><td>DF</td></tr>
<tr><td>08/04/17</td><td>2</td><td>30|0|1</td><td>DF</td></tr>
<tr><td>26/05/17</td><td>2</td><td>27|0|0</td><td>DF</td></tr>
<tr><td>2 June?</td><td>2</td><td>24|02|00</td><td>DF</td></tr>
<tr><td>3 June?</td><td>1.5</td><td>12|07|04</td><td>DF</td></tr>
</table>

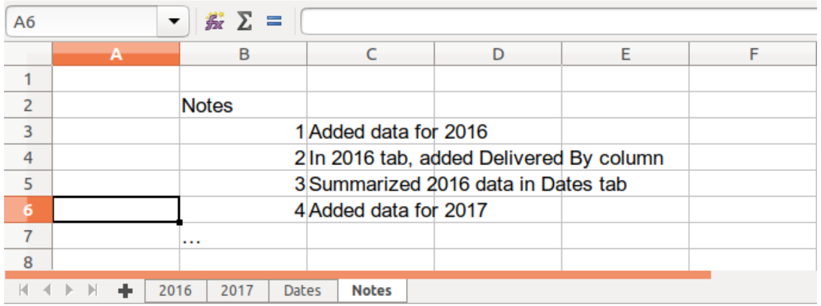
# Keeping track of data changes

When working with spreadsheets during data clean up or assessment it’s very easy to end up with a spreadsheet that looks very different from the one you started with.

You should be able reproduce your analyses or figure out what you did.

**Create a new file with your cleaned or analyzed data. Don’t modify the original dataset, or you will never know where you started!**

- Keep track of the steps you took in your clean up or analysis.
- Record the steps you followed in your data cleanup or analysis.



|   | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 |  |  |  |  |  |  |
| 2 |  | Notes |  |  |  |  |
| 3 |  |  | 1 | Added data for 2016 |  |  |
| 4 |  |  | 2 | In 2016 tab, added Delivered By column |  |  |
| 5 |  |  | 3 | Summarized 2016 data in Dates tab |  |  |
| 6 |  |  | 4 | Added data for 2017 |  |  |
| 7 |  | ... |  |  |  |  |
| 8 |  |  |  |  |  |  |

# Exercise

**We’re going to take a messy version of some library training data and clean it up.**

- Download the data: https://librarycarpentry.github.io/lc-spreadsheets/data/training_attendance.xlsx
- Open the data in a spreadsheet program.
- Various people have recorded training attendance statistics over 2016 and 2017, and they have kept track of the data in their own way.
- Now you’re being asked to evaluate the training programme and you want to be able to start doing statistics with the data.
- Work on the messy data so that a computer will be able to understand it.
- **Clean up the 2016 and 2017 tabs, and merge them into a single spreadsheet.**
- After you go through this exercise, we’ll discuss as a group what you think was wrong with this data and how you fixed it.

# Exercise

**We’re going to take a messy version of some library training data and clean it up.**

- Download the data: https://librarycarpentry.github.io/lc-spreadsheets/data/training_attendance.xlsx
- Open the data in a spreadsheet program.
- Various people have recorded training attendance statistics over 2016 and 2017, and they have kept track of the data in their own way.
- Now you’re being asked to evaluate the training programme and you want to be able to start doing statistics with the data.
- Work on the messy data so that a computer will be able to understand it.
- **Clean up the 2016 and 2017 tabs, and merge them into a single spreadsheet.**
- After you go through this exercise, we’ll discuss as a group what you think was wrong with this data and how you fixed it.

**Let’s take a look in Excel…**

# Common Spreadsheet Errors

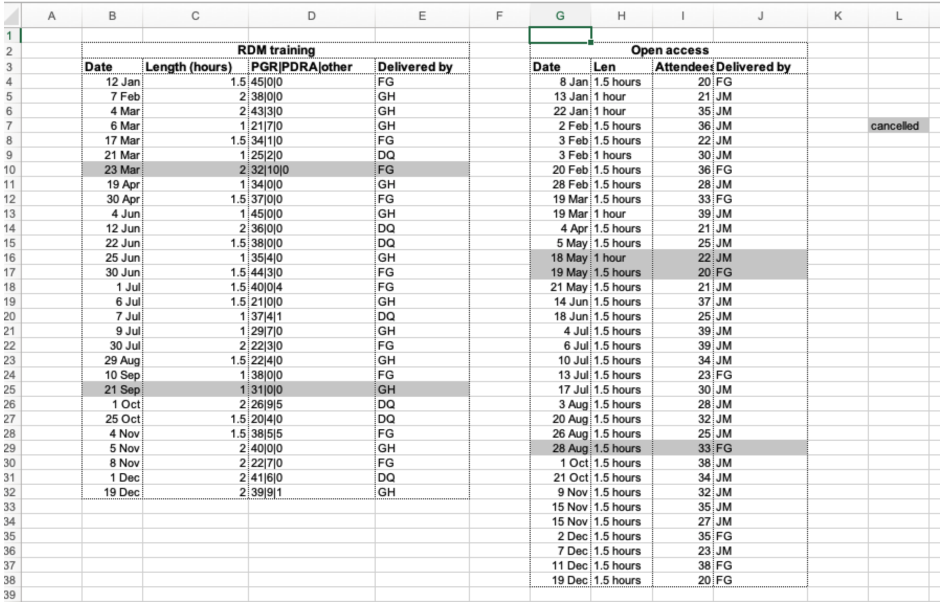
**Issues to cover:**

- Multiple tables
- Multiple tabs
- Zeroes and not filling in zeroes
- Recording null values
- Using formatting to convey information
- Using formatting to make the data sheet look pretty
- Placing comments or units in cells
- More than one piece of information in a cell
- Field name problems
- Special characters in data
- Date formatting

**You got a preview of some of these issues when we looked at the Excel sheet, but we'll go through examples of each…**

# Multiple Tables

- A common strategy is creating multiple data tables within one spreadsheet.
- **This confuses the computer, so don’t do this!**
- When you create multiple tables within one spreadsheet, you’re drawing false associations between things for the computer, which sees each row as an observation.
- You’re also potentially using the same field name in multiple places, which will make it harder to clean your data up into a usable form.
- The example from our worksheet illustrates this problem



| | A | B | C | D | E | F | G | H | I | J | K | L |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | | | | | | | | | | |
| 2 | | RDM training | | | | | Open access | | | | | |
| 3 | | **Date** | **Length (hours)** | **PGR\|PDRA\|other** | **Delivered by** | | **Date** | **Len** | **Attendees** | **Delivered by** | | |
| 4 | | 12 Jan | 1.5 | 45\|0\|0 | FG | | 8 Jan | 1.5 hours | 20 | FG | | |
| 5 | | 7 Feb | 2 | 38\|0\|0 | GH | | 13 Jan | 1 hour | 21 | JM | | |
| 6 | | 4 Mar | 2 | 43\|3\|0 | GH | | 22 Jan | 1 hour | 35 | JM | | |
| 7 | | 6 Mar | 1 | 21\|7\|0 | GH | | 2 Feb | 1.5 hours | 36 | JM | | cancelled |
| 8 | | 17 Mar | 1.5 | 34\|1\|0 | FG | | 3 Feb | 1.5 hours | 22 | JM | | |
| 9 | | 21 Mar | 1 | 25\|2\|0 | DQ | | 3 Feb | 1 hours | 30 | JM | | |
| 10 | | 23 Mar | 2 | 32\|10\|0 | FG | | 20 Feb | 1.5 hours | 36 | FG | | |
| 11 | | 19 Apr | 1 | 34\|0\|0 | GH | | 28 Feb | 1.5 hours | 28 | JM | | |
| 12 | | 30 Apr | 1.5 | 37\|0\|0 | FG | | 19 Mar | 1.5 hours | 33 | FG | | |
| 13 | | 4 Jun | 1 | 45\|0\|0 | GH | | 19 Mar | 1 hour | 39 | JM | | |
| 14 | | 12 Jun | 2 | 36\|0\|0 | DQ | | 4 Apr | 1.5 hours | 21 | JM | | |
| 15 | | 22 Jun | 1.5 | 38\|0\|0 | DQ | | 5 May | 1.5 hours | 25 | JM | | |
| 16 | | 25 Jun | 1 | 35\|4\|0 | GH | | 18 May | 1 hour | 22 | JM | | |
| 17 | | 30 Jun | 1.5 | 44\|3\|0 | FG | | 19 May | 1.5 hours | 20 | FG | | |
| 18 | | 1 Jul | 1.5 | 40\|0\|4 | FG | | 21 May | 1.5 hours | 21 | JM | | |
| 19 | | 6 Jul | 1.5 | 21\|0\|0 | GH | | 14 Jun | 1.5 hours | 37 | JM | | |
| 20 | | 7 Jul | 1 | 37\|4\|1 | DQ | | 18 Jun | 1.5 hours | 25 | JM | | |
| 21 | | 9 Jul | 1 | 29\|7\|0 | GH | | 4 Jul | 1.5 hours | 39 | JM | | |
| 22 | | 30 Jul | 2 | 22\|3\|0 | FG | | 6 Jul | 1.5 hours | 39 | JM | | |
| 23 | | 29 Aug | 1.5 | 22\|4\|0 | GH | | 10 Jul | 1.5 hours | 34 | JM | | |
| 24 | | 10 Sep | 1 | 38\|0\|0 | FG | | 13 Jul | 1.5 hours | 23 | FG | | |
| 25 | | 21 Sep | 1 | 31\|0\|0 | GH | | 17 Jul | 1.5 hours | 30 | JM | | |
| 26 | | 1 Oct | 2 | 26\|9\|5 | DQ | | 3 Aug | 1.5 hours | 28 | JM | | |
| 27 | | 25 Oct | 1.5 | 20\|4\|0 | DQ | | 20 Aug | 1.5 hours | 32 | JM | | |
| 28 | | 4 Nov | 1.5 | 38\|5\|5 | FG | | 26 Aug | 1.5 hours | 25 | JM | | |
| 29 | | 5 Nov | 2 | 40\|0\|0 | GH | | 28 Aug | 1.5 hours | 33 | FG | | |
| 30 | | 8 Nov | 2 | 22\|7\|0 | FG | | 1 Oct | 1.5 hours | 38 | JM | | |
| 31 | | 1 Dec | 2 | 41\|6\|0 | DQ | | 21 Oct | 1.5 hours | 34 | JM | | |
| 32 | | 19 Dec | 2 | 39\|9\|1 | GH | | 9 Nov | 1.5 hours | 32 | JM | | |
| 33 | | | | | | | 15 Nov | 1.5 hours | 35 | JM | | |
| 34 | | | | | | | 15 Nov | 1.5 hours | 27 | JM | | |
| 35 | | | | | | | 2 Dec | 1.5 hours | 35 | FG | | |
| 36 | | | | | | | 7 Dec | 1.5 hours | 23 | JM | | |
| 37 | | | | | | | 11 Dec | 1.5 hours | 38 | FG | | |
| 38 | | | | | | | 19 Dec | 1.5 hours | 20 | FG | | |
| 39 | | | | | | | | | | | | |

# Tabs and Headers

**Tabs:**

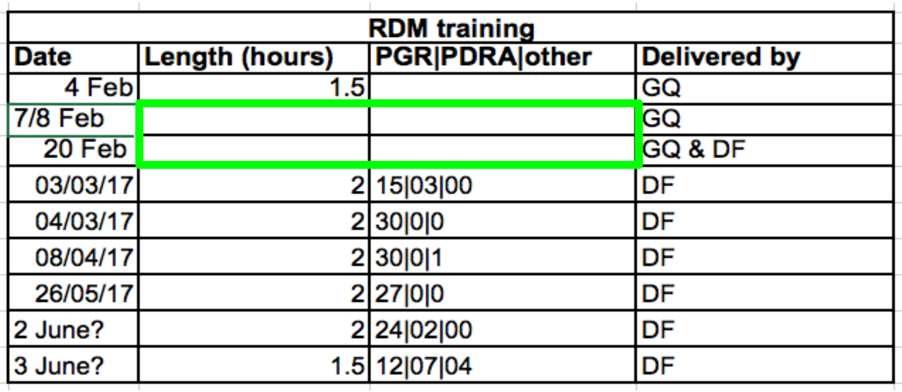
- When you create extra tabs, the computer can’t necessarily see connections in the data
- This can be bad practice for a  few reasons:
    - Have to explicitly tell the computer how to combine tabs
    - If the tabs are inconsistently formatted, you might have to do it by hand
    - Adds an extra step for yourself before you analyze the data because you will have to combine the data into a single datatable.
- Your data sheet might get very long over the course of recording data.

**Headers:**

- A long data sheet does make it harder to enter data if you can’t see your headers at the top of the spreadsheet.
- Do NOT repeat headers. These can easily get mixed into the data, leading to problems down the road.
- Instead you can Freeze the column headers.

# Zeroes and not filling in zeroes

- Why bother writing in the number zero in a column?
- **There’s a difference between a zero and a blank cell in a spreadsheet.**
- **To the computer, a zero is actually data.**
  - **You measured or counted it.**
  - **A blank cell means that it wasn’t measured and the computer will interpret it as a null value.**
- The spreadsheets or statistical programs will likely mis-interpret blank or null cells that are actually meant to be zero.
- By not recording anything, you are leaving out data.
- Callback to the uncertainty of blanks!



| RDM training | | | |
|---|---|---|---|
| **Date** | **Length (hours)** | **PGR\|PDRA\|other** | **Delivered by** |
| 4 Feb | 1.5 | | GQ |
| 7/8 Feb | | | GQ |
| 20 Feb | | | GQ & DF |
| 03/03/17 | 2 | 15\|03\|00 | DF |
| 04/03/17 | 2 | 30\|0\|0 | DF |
| 08/04/17 | 2 | 30\|0\|1 | DF |
| 26/05/17 | 2 | 27\|0\|0 | DF |
| 2 June? | 2 | 24\|02\|00 | DF |
| 3 June? | 1.5 | 12\|07\|04 | DF |

**0 = I know for a fact that no one attended this class**

**[blank/null] = I did not record any attendance for this class**

# Recording null values

**Example**:

- Sometimes using other numerical values or text to represent missing values comes up because a field cannot be left blank or empty
- Whatever the reason, it can be problem if unknown or missing data is recorded as (example): -999, 999, or 0.
- How these values are interpreted will depend on the software you use to analyze your data.

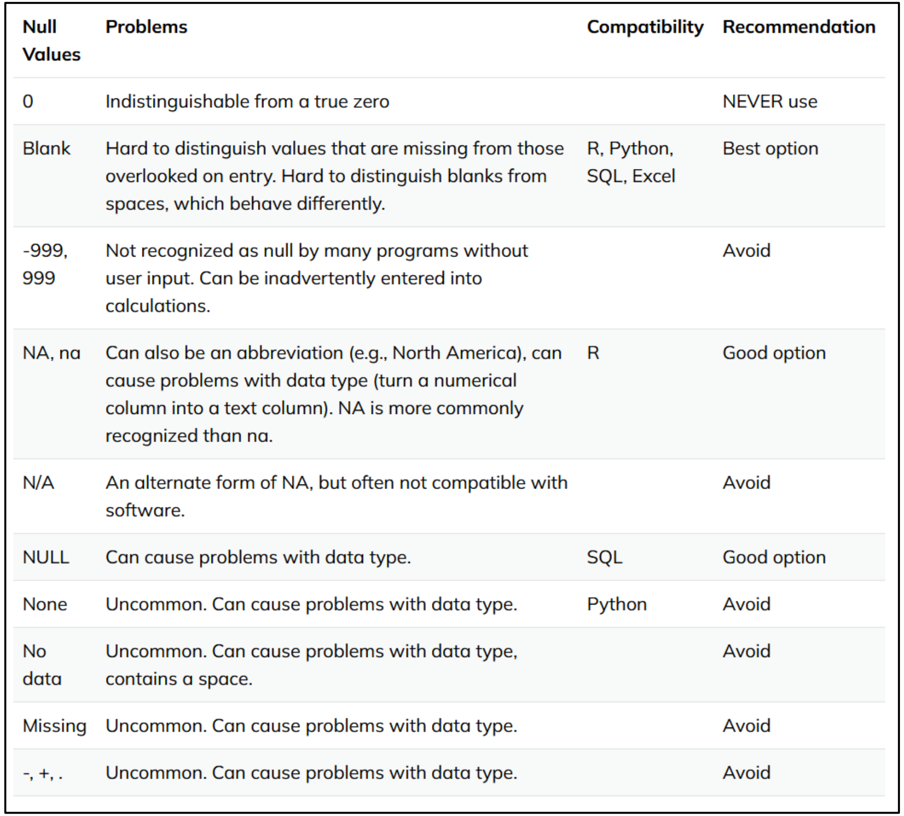
# Recording null values

**Solution**:

- Blank cells are the best choices for most applications.
- There are many reasons that null values may be represented differently within a dataset.
- **Bottom Line: Use a <u>clearly defined and CONSISTENT</u> null indicator.**

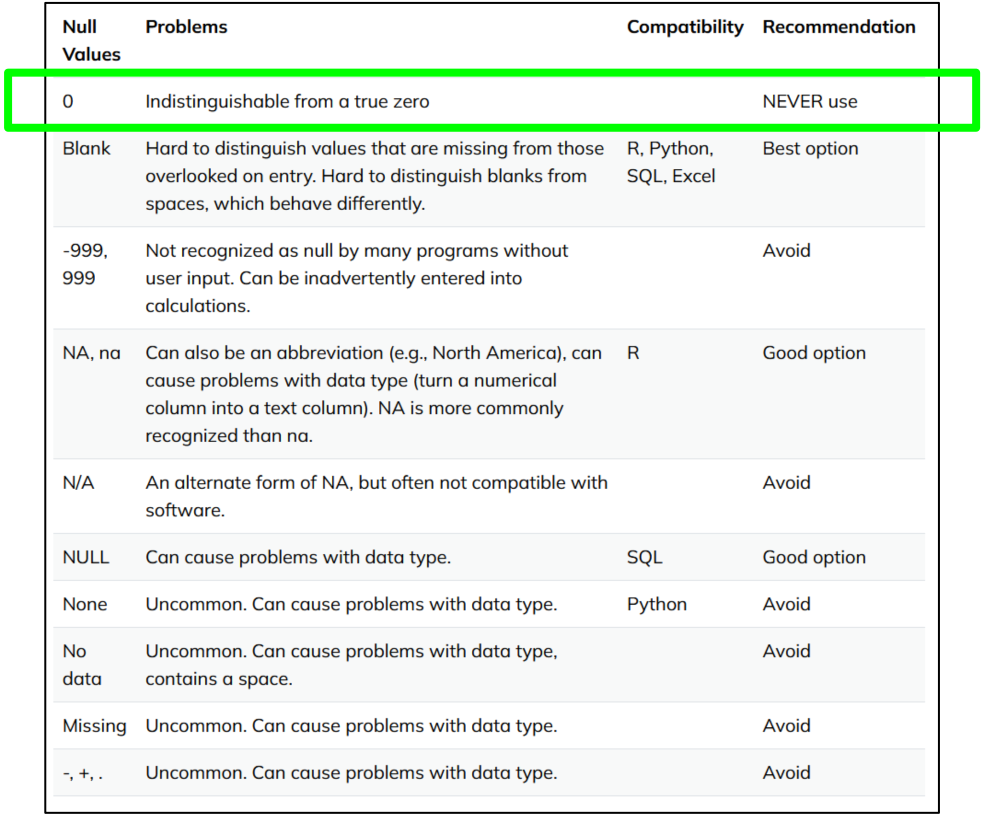
- Some commonly used null values are as follows…

# Recording null values



| Null Values | Problems | Compatibility | Recommendation |
|---|---|---|---|
| 0 | Indistinguishable from a true zero | | NEVER use |
| Blank | Hard to distinguish values that are missing from those overlooked on entry. Hard to distinguish blanks from spaces, which behave differently. | R, Python, SQL, Excel | Best option |
| -999, 999 | Not recognized as null by many programs without user input. Can be inadvertently entered into calculations. | | Avoid |
| NA, na | Can also be an abbreviation (e.g., North America), can cause problems with data type (turn a numerical column into a text column). NA is more commonly recognized than na. | R | Good option |
| N/A | An alternate form of NA, but often not compatible with software. | | Avoid |
| NULL | Can cause problems with data type. | SQL | Good option |
| None | Uncommon. Can cause problems with data type. | Python | Avoid |
| No data | Uncommon. Can cause problems with data type, contains a space. | | Avoid |
| Missing | Uncommon. Can cause problems with data type. | | Avoid |
| -, +, . | Uncommon. Can cause problems with data type. | | Avoid |

# Recording null values

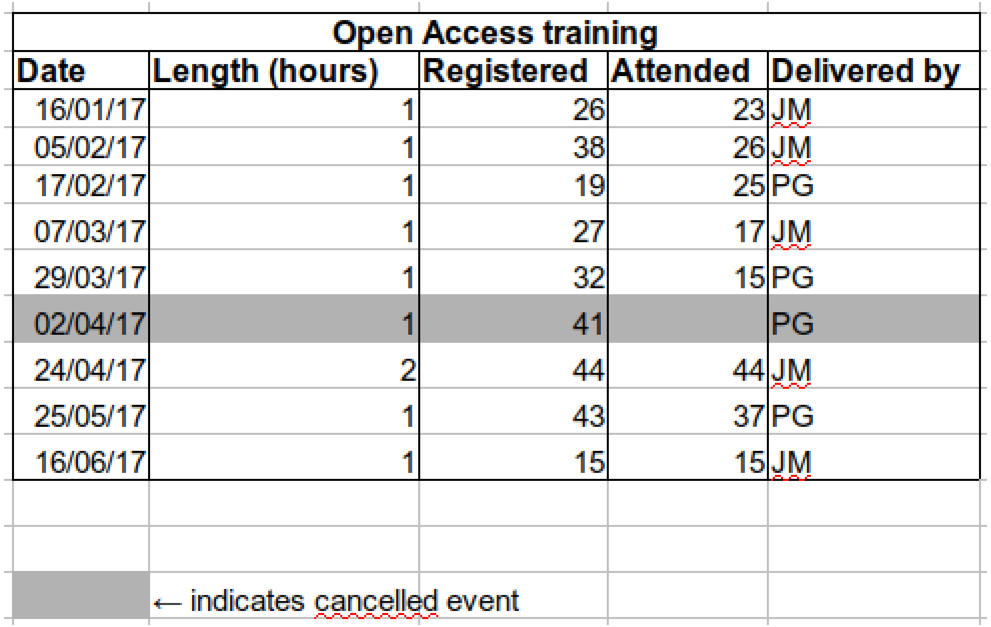


| Null Values | Problems | Compatibility | Recommendation |
|---|---|---|---|
| 0 | Indistinguishable from a true zero | | NEVER use |
| Blank | Hard to distinguish values that are missing from those overlooked on entry. Hard to distinguish blanks from spaces, which behave differently. | R, Python, SQL, Excel | Best option |
| -999, 999 | Not recognized as null by many programs without user input. Can be inadvertently entered into calculations. | | Avoid |
| NA, na | Can also be an abbreviation (e.g., North America), can cause problems with data type (turn a numerical column into a text column). NA is more commonly recognized than na. | R | Good option |
| N/A | An alternate form of NA, but often not compatible with software. | | Avoid |
| NULL | Can cause problems with data type. | SQL | Good option |
| None | Uncommon. Can cause problems with data type. | Python | Avoid |
| No data | Uncommon. Can cause problems with data type, contains a space. | | Avoid |
| Missing | Uncommon. Can cause problems with data type. | | Avoid |
| -, +, . | Uncommon. Can cause problems with data type. | | Avoid |

# Using formatting to convey information

**Example**:

- Highlighting cells, rows or columns that should be excluded from an analysis, leaving blank rows to indicate separations in data.



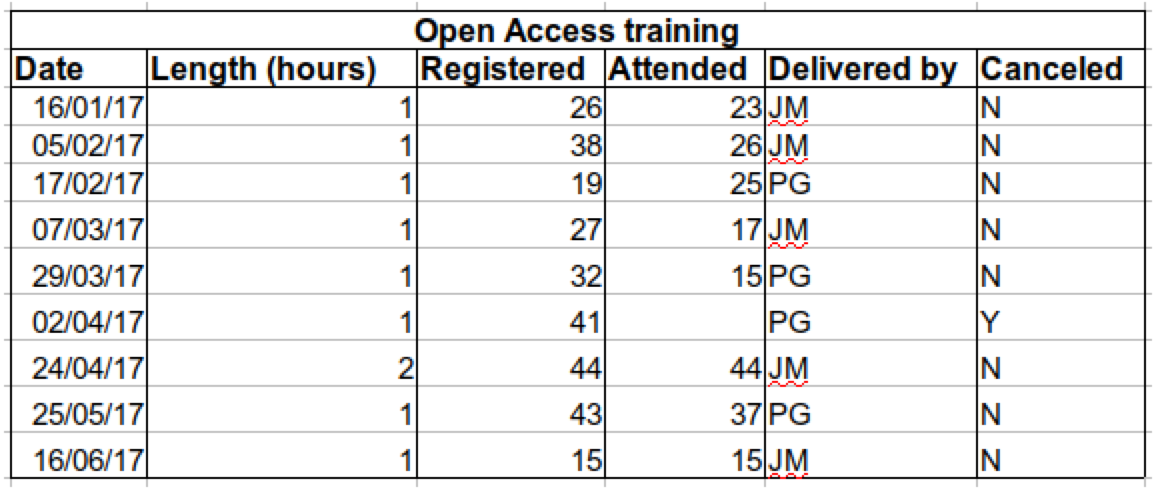
| Open Access training | | | | |
|---|---|---|---|---|
| Date | Length (hours) | Registered | Attended | Delivered by |
| 16/01/17 | 1 | 26 | 23 | JM |
| 05/02/17 | 1 | 38 | 26 | JM |
| 17/02/17 | 1 | 19 | 25 | PG |
| 07/03/17 | 1 | 27 | 17 | JM |
| 29/03/17 | 1 | 32 | 15 | PG |
| 02/04/17 | 1 | 41 | | PG |
| 24/04/17 | 2 | 44 | 44 | JM |
| 25/05/17 | 1 | 43 | 37 | PG |
| 16/06/17 | 1 | 15 | 15 | JM |

▨ ← indicates cancelled event

# Using formatting to convey information

**Solution**:

- Create a new field to encode which data should be excluded.



| Open Access training | | | | | |
|---|---|---|---|---|---|
| Date | Length (hours) | Registered | Attended | Delivered by | Canceled |
| 16/01/17 | 1 | 26 | 23 | JM | N |
| 05/02/17 | 1 | 38 | 26 | JM | N |
| 17/02/17 | 1 | 19 | 25 | PG | N |
| 07/03/17 | 1 | 27 | 17 | JM | N |
| 29/03/17 | 1 | 32 | 15 | PG | N |
| 02/04/17 | 1 | 41 | | PG | Y |
| 24/04/17 | 2 | 44 | 44 | JM | N |
| 25/05/17 | 1 | 43 | 37 | PG | N |
| 16/06/17 | 1 | 15 | 15 | JM | N |

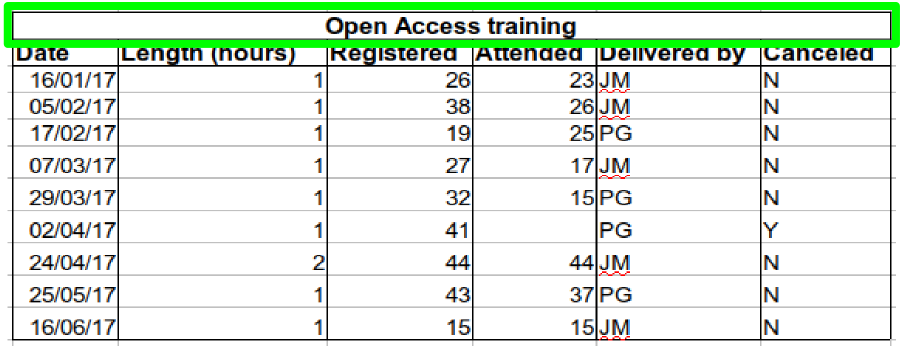
# Using formatting to make the data sheet look pretty

**Example**:

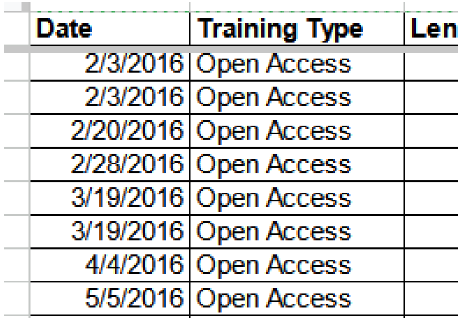
- Merging cells.

**Solution**:

- Formatting a worksheet to be more aesthetically pleasing can compromise your computer’s ability to see associations in the data.
- Merged cells are an absolute formatting NO-NO if you want to make your data readable by statistics software.
- Consider restructuring your data in such a way that you will not need to merge cells to organize your data.



| Open Access training | | | | | |
|---|---|---|---|---|---|
| Date | Length (hours) | Registered | Attended | Delivered by | Canceled |
| 16/01/17 | 1 | 26 | 23 | JM | N |
| 05/02/17 | 1 | 38 | 26 | JM | N |
| 17/02/17 | 1 | 19 | 25 | PG | N |
| 07/03/17 | 1 | 27 | 17 | JM | N |
| 29/03/17 | 1 | 32 | 15 | PG | N |
| 02/04/17 | 1 | 41 | | PG | Y |
| 24/04/17 | 2 | 44 | 44 | JM | N |
| 25/05/17 | 1 | 43 | 37 | PG | N |
| 16/06/17 | 1 | 15 | 15 | JM | N |



| Date | Training Type | Len |
|---|---|---|
| 2/3/2016 | Open Access | |
| 2/3/2016 | Open Access | |
| 2/20/2016 | Open Access | |
| 2/28/2016 | Open Access | |
| 3/19/2016 | Open Access | |
| 3/19/2016 | Open Access | |
| 4/4/2016 | Open Access | |
| 5/5/2016 | Open Access | |

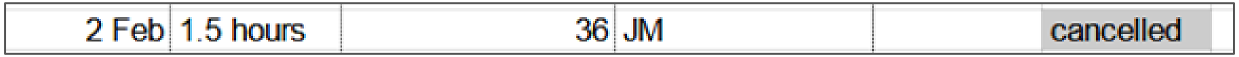
# Placing comments or units in cells

**Example**:

- Your data was collected, in part, by a summer student who you later found out was mis-recording the duration of training sessions, some of the time. You want a way to note these data are suspect.

**Solution**:

- Most statistical programs can’t see Excel’s comments, and would be confused by comments placed within your data cells.
- As described above for formatting, create another field if you need to add notes to cells.
- Similarly, don’t include units in cells (such as “hours”,“min”): ideally, all the units or measurements you place in one column should be of the same standard.
- If for some reason they aren’t, insert another column and specify the units.



| | | | | |
|---|---|---|---|---|
| 2 Feb | 1.5 hours | 36 | JM | cancelled |

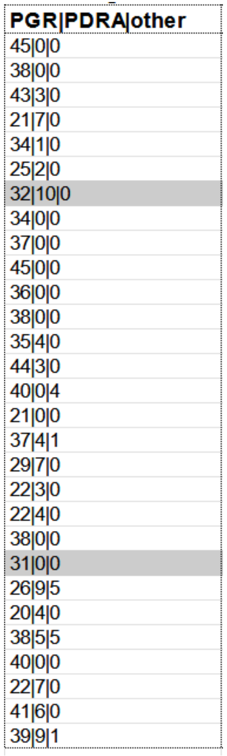
# More than one piece of information in a cell

**Example**:

- One table recorded attendance by the different types of attendees.
- This table recorded number of attendees of different types:
  - Post-graduate researcher (PGR)
  - Post-doctoral research associate (PDRA)
  - Other

**Solution**:

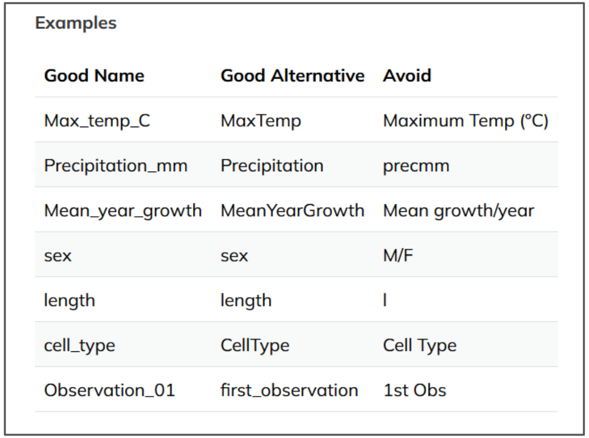
- Never include more than one piece of information in a cell.
- Design your data sheet to include a column for each type of attendee, if this information is important to collect, rather than just a total number.



<table>
<tr><th>PGR|PDRA|other</th></tr>
<tr><td>45|0|0</td></tr>
<tr><td>38|0|0</td></tr>
<tr><td>43|3|0</td></tr>
<tr><td>21|7|0</td></tr>
<tr><td>34|1|0</td></tr>
<tr><td>25|2|0</td></tr>
<tr><td>32|10|0</td></tr>
<tr><td>34|0|0</td></tr>
<tr><td>37|0|0</td></tr>
<tr><td>45|0|0</td></tr>
<tr><td>36|0|0</td></tr>
<tr><td>38|0|0</td></tr>
<tr><td>35|4|0</td></tr>
<tr><td>44|3|0</td></tr>
<tr><td>40|0|4</td></tr>
<tr><td>21|0|0</td></tr>
<tr><td>37|4|1</td></tr>
<tr><td>29|7|0</td></tr>
<tr><td>22|3|0</td></tr>
<tr><td>22|4|0</td></tr>
<tr><td>38|0|0</td></tr>
<tr><td>31|0|0</td></tr>
<tr><td>26|9|5</td></tr>
<tr><td>20|4|0</td></tr>
<tr><td>38|5|5</td></tr>
<tr><td>40|0|0</td></tr>
<tr><td>22|7|0</td></tr>
<tr><td>41|6|0</td></tr>
<tr><td>39|9|1</td></tr>
</table>

# Field name problems

- Choose descriptive field names, but **don’t include spaces, numbers, or special characters of any kind.**
- Spaces can be misinterpreted by parsers that use whitespace as delimiters
- Some programs don’t like field names that are text strings that start with numbers.
- Underscores (_) are a good alternative to spaces and consider writing names in camel-case to improve readability.
- Including the units in the field names avoids confusion and enables others to readily interpret your fields.



Examples

| Good Name | Good Alternative | Avoid |
|---|---|---|
| Max_temp_C | MaxTemp | Maximum Temp (°C) |
| Precipitation_mm | Precipitation | precmm |
| Mean_year_growth | MeanYearGrowth | Mean growth/year |
| sex | sex | M/F |
| length | length | l |
| cell_type | CellType | Cell Type |
| Observation_01 | first_observation | 1st Obs |

# Special characters in data

**Example**:

- Treating Excel as a word processor when writing notes, even copying data directly from Word or other applications.

**Solution**:

- When writing longer text in a cell, people often include **line breaks, carriage returns, etc.** in their spreadsheet.
- When copying data in from applications such as Word, formatting and fancy non-standard characters (such as left- and right-aligned quotation marks) are included.
- **When exporting this data into a coding/statistical environment or into a relational database, you may get lines being cut in half and encoding errors being thrown.**

**General best practice is to avoid adding characters such as newlines, tabs, and vertical tabs. In other words, treat a text cell as if it were a simple web form that can only contain text and spaces.**

# Inclusion of metadata in data table

**Example**:

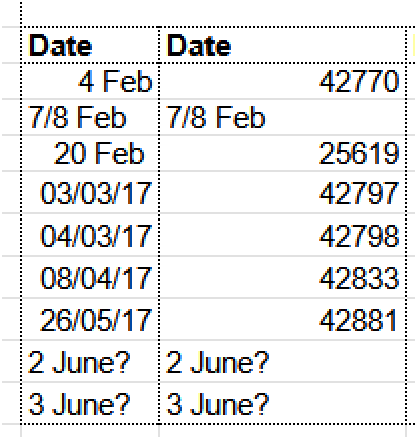
- Adding a legend in your data table explaining column meaning, units, exceptions, etc.

**Solution**:

- While recording data about your data (“metadata”) is essential, this information should not be contained in the data file itself.
- It can disrupt how computer programs interpret your data file.
  - Example: Meaning of “PGR/PGRA/Other”
- Metadata should be stored as a separate file in the same directory as your data file, preferably in plain text format with a name that clearly associates it with your data file.
- Because metadata files are free text format, they also allow you to encode comments, units, information about how null values are encoded, etc. that are important to document but can disrupt the formatting of your data file.

# Dates as data

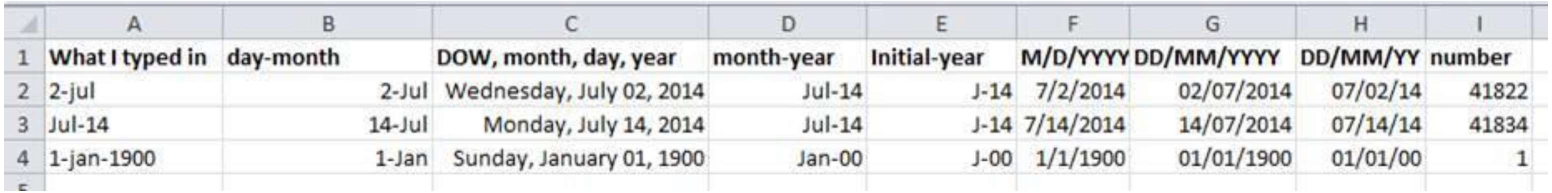
- Dates in spreadsheets are often stored in one column.
- A spreadsheet application may display the dates correctly (for readability) but how it actually handles and stores the dates may be problematic.
- This can cause problems if the date displayed does not fully represent the information that the spreadsheet application is using, such as when the year is not visually displayed
- Date information may be changed when data is converted to different spreadsheet formats, such as between `.xlsx` and `.csv`, or opened in different applications.



| Date | Date |
|---|---|
| 4 Feb | 42770 |
| 7/8 Feb | 7/8 Feb |
| 20 Feb | 25619 |
| 03/03/17 | 42797 |
| 04/03/17 | 42798 |
| 08/04/17 | 42833 |
| 26/05/17 | 42881 |
| 2 June? | 2 June? |
| 3 June? | 3 June? |

# Date formats in spreadsheets

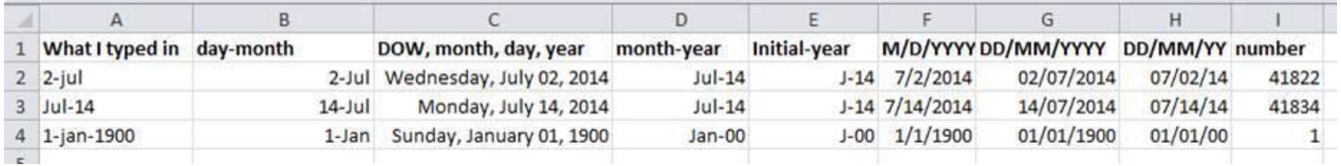
- Spreadsheet applications employ numerous features that facilitate the processing and display of date information.
- These features often make date information more easily readable, but the underlying data handling techniques can create data ambiguity in a variety of ways.
- The figure below illustrates some of the ways that the display of information representing the same date can vary.
- Column A is the information as entered by a user,
- The following columns show different ways that the information may be displayed:



|  | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | What I typed in | day-month | DOW, month, day, year | month-year | Initial-year | M/D/YYYY | DD/MM/YYYY | DD/MM/YY | number |
| 2 | 2-jul | 2-Jul | Wednesday, July 02, 2014 | Jul-14 | J-14 | 7/2/2014 | 02/07/2014 | 07/02/14 | 41822 |
| 3 | Jul-14 | 14-Jul | Monday, July 14, 2014 | Jul-14 | J-14 | 7/14/2014 | 14/07/2014 | 07/14/14 | 41834 |
| 4 | 1-jan-1900 | 1-Jan | Sunday, January 01, 1900 | Jan-00 | J-00 | 1/1/1900 | 01/01/1900 | 01/01/00 | 1 |
| 5 |  |  |  |  |  |  |  |  |  |

# Date formats in spreadsheets

## How can these features create data ambiguity?



|  | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | What I typed in | day-month | DOW, month, day, year | month-year | Initial-year | M/D/YYYY | DD/MM/YYYY | DD/MM/YY | number |
| 2 | 2-jul | 2-Jul | Wednesday, July 02, 2014 | Jul-14 | J-14 | 7/2/2014 | 02/07/2014 | 07/02/14 | 41822 |
| 3 | Jul-14 | 14-Jul | Monday, July 14, 2014 | Jul-14 | J-14 | 7/14/2014 | 14/07/2014 | 07/14/14 | 41834 |
| 4 | 1-jan-1900 | 1-Jan | Sunday, January 01, 1900 | Jan-00 | J-00 | 1/1/1900 | 01/01/1900 | 01/01/00 | 1 |

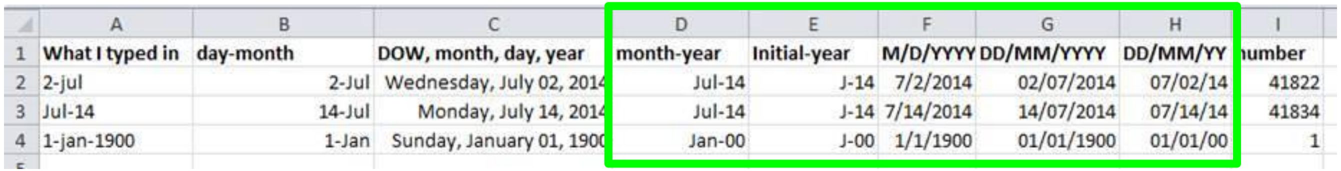
Ideally, data should be as unambiguous as possible.

## Question:

- What do you notice about the display of the date information above? What information changes between the columns?
- What aspects of the display lack specificity and may introduce ambiguity?

# Date formats in spreadsheets

## How can these features create data ambiguity?



|  | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | What I typed in | day-month | DOW, month, day, year | month-year | Initial-year | M/D/YYYY | DD/MM/YYYY | DD/MM/YY | number |
| 2 | 2-jul | 2-Jul | Wednesday, July 02, 2014 | Jul-14 | J-14 | 7/2/2014 | 02/07/2014 | 07/02/14 | 41822 |
| 3 | Jul-14 | 14-Jul | Monday, July 14, 2014 | Jul-14 | J-14 | 7/14/2014 | 14/07/2014 | 07/14/14 | 41834 |
| 4 | 1-jan-1900 | 1-Jan | Sunday, January 01, 1900 | Jan-00 | J-00 | 1/1/1900 | 01/01/1900 | 01/01/00 | 1 |

Ideally, data should be as unambiguous as possible.

## Question:

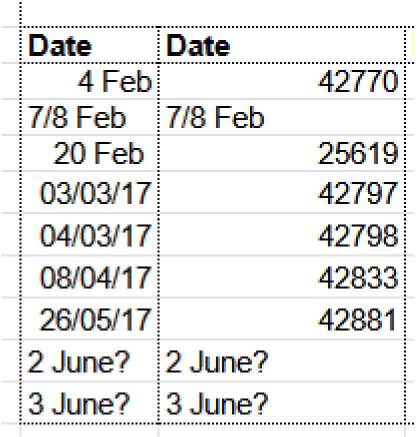
- What do you notice about the display of the date information above? What information changes between the columns?
- What aspects of the display lack specificity and may introduce ambiguity?

# Displaying dates

- The display format of each cell can be modified.
- To change the display in Excel, navigate to the Format menu and choose “Cells…”.
- In the “Format Cells” dialog box, you can select a Date format and choose various display outputs (some are shown in the above figure).
- In the dialog box, you can also choose to format the cell as a number or text.
- It may be useful to format the cell as one of these other data types, since as we will discuss next, the spreadsheet program understands the date information as a number.
- Let’s take a look in Excel…

# Storing dates

- Spreadsheet applications, including Excel, **store dates as a number**.
- The application developers chose a single day to designate as day zero, and each subsequent day is incremented by a value of one.
- **Excel counts the days from a default of December 31, 1899.**
    - July 2, 2014 is stored as the number 41822 because it is 41,822 days after day zero.
- Not all applications or operating systems use the same date for day zero.
- Understanding the spreadsheet program uses serial numbers to process dates can be useful in some circumstances.
- Using the above functions, you can easily add days, months or years to a given date.



| Date | Date |
|---|---|
| 4 Feb | 42770 |
| 7/8 Feb | 7/8 Feb |
| 20 Feb | 25619 |
| 03/03/17 | 42797 |
| 04/03/17 | 42798 |
| 08/04/17 | 42833 |
| 26/05/17 | 42881 |
| 2 June? | 2 June? |
| 3 June? | 3 June? |

# Storing dates

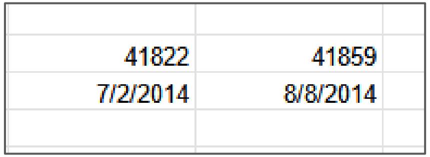
**Example:**

- Creating a series of dates where each date is advanced by thirty seven days.
- Let’s take a look in Excel…

# Storing dates

**Example:**

- Creating a series of dates where each date is advanced by thirty seven days.

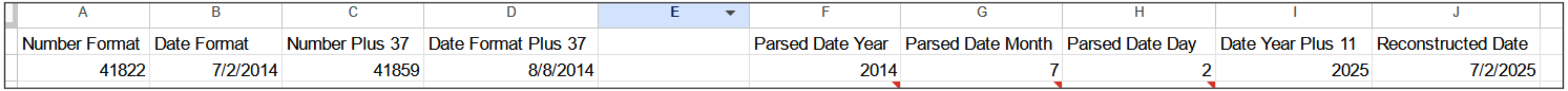


| | |
|---|---|
| 41822 | 41859 |
| 7/2/2014 | 8/8/2014 |

- This happens because Excel processes the date July 2, 2014 as the number 41822.
- Adding 41822 + 37 results in 41859 which Excel interprets as August 8, 2014.
- The program retains the format of the cell that is being operated upon (unless you did some sort of formatting to the cell before)
- Month and year rollovers are internally tracked and applied.

# Useful spreadsheet functions for working with date information

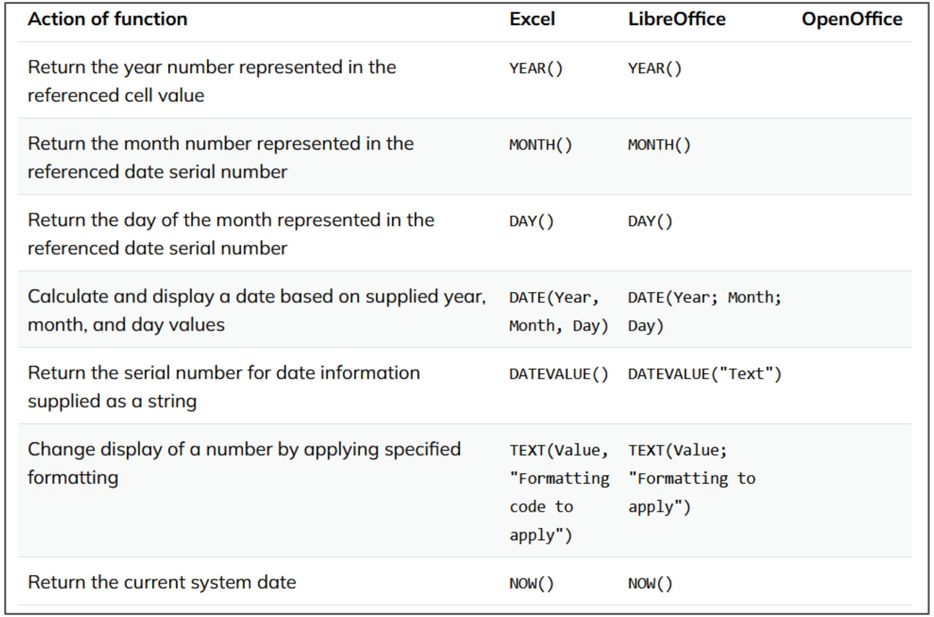
- If you later need to export the data and need to preserve the dates or times, consider recording date information using one of the solutions discussed below:
  - If a date is entered in one column, use functions to extract information from that column into other columns.
  - It can be useful to display the specific information about the year, month, and day.
- Date-related functions allow us to:
  - Convert date values from the stored numerical value to a readable display value.
  - Make calculations between date values.
  - Extract the date values so that they do not change as data is transformed, exchanged, or exported between new users and systems.



| A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|
| Number Format | Date Format | Number Plus 37 | Date Format Plus 37 | | Parsed Date Year | Parsed Date Month | Parsed Date Day | Date Year Plus 11 | Reconstructed Date |
| 41822 | 7/2/2014 | 41859 | 8/8/2014 | | 2014 | 7 | 2 | 2025 | 7/2/2025 |

# Useful spreadsheet functions for working with date information

The table below outlines a few useful date-related functions and how they differ between some of the widely used spreadsheet applications.



| Action of function | Excel | LibreOffice | OpenOffice |
|---|---|---|---|
| Return the year number represented in the referenced cell value | `YEAR()` | `YEAR()` | |
| Return the month number represented in the referenced date serial number | `MONTH()` | `MONTH()` | |
| Return the day of the month represented in the referenced date serial number | `DAY()` | `DAY()` | |
| Calculate and display a date based on supplied year, month, and day values | `DATE(Year, Month, Day)` | `DATE(Year; Month; Day)` | |
| Return the serial number for date information supplied as a string | `DATEVALUE()` | `DATEVALUE("Text")` | |
| Change display of a number by applying specified formatting | `TEXT(Value, "Formatting code to apply")` | `TEXT(Value; "Formatting to apply")` | |
| Return the current system date | `NOW()` | `NOW()` | |

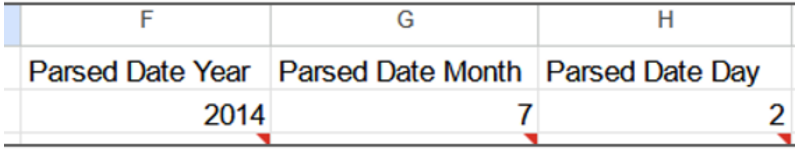
# Using Date-Related Functions (Excel)

## Exercise:

Pulling month, day, and year out of dates:

- In the `Dates` tab of your Excel file we summarized training data from 2015. There's a `date` column.
- Extract month, day and year from the date to three new columns.

Tip: Make sure the new column is formatted as a number and not as a date. Change the function to correspond to each row: i.e., =MONTH(A3), =DAY(A3), =YEAR(A3) for the next row.



| F | G | H |
|---|---|---|
| Parsed Date Year | Parsed Date Month | Parsed Date Day |
| 2014 | 7 | 2 |

# Adding Dates

Adding years and months and days is slightly trickier because we need to make sure that we are adding the amount to the correct entity.

- First we extract the single entities (day, month, or year)
- We can then add values to do that
- Finally the complete date string is reconstructed using the `DATE()` function.

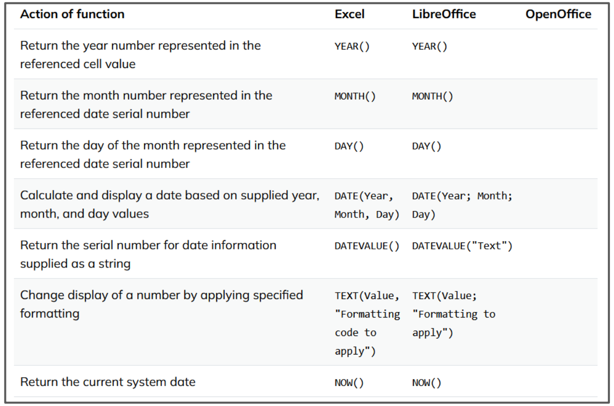
# Working with historical dates

Especially important given our line of work!

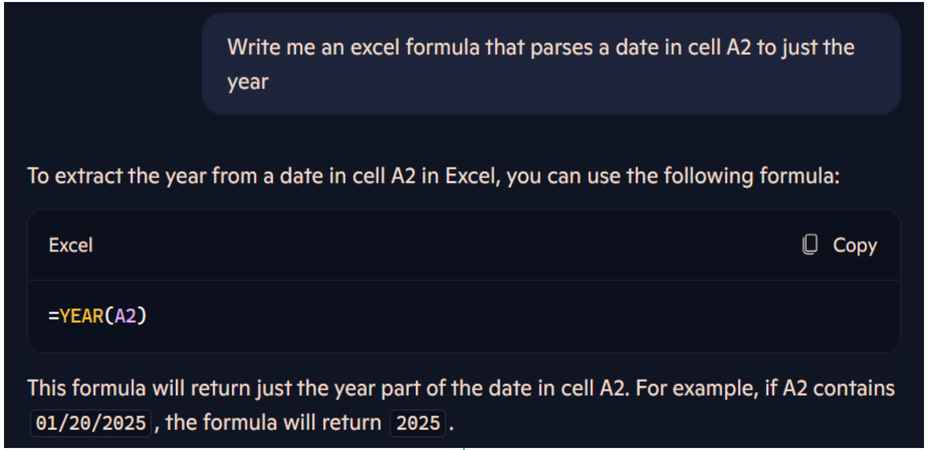
- **Excel is unable to parse dates from before 1899-12-31.**
- **Let’s take a look in Excel again…**

# Useful spreadsheet functions for working with date information

And if you can’t remember what terms to use in a formula…



| Action of function | Excel | LibreOffice | OpenOffice |
|---|---|---|---|
| Return the year number represented in the referenced cell value | `YEAR()` | `YEAR()` | |
| Return the month number represented in the referenced date serial number | `MONTH()` | `MONTH()` | |
| Return the day of the month represented in the referenced date serial number | `DAY()` | `DAY()` | |
| Calculate and display a date based on supplied year, month, and day values | `DATE(Year, Month, Day)` | `DATE(Year; Month; Day)` | |
| Return the serial number for date information supplied as a string | `DATEVALUE()` | `DATEVALUE("Text")` | |
| Change display of a number by applying specified formatting | `TEXT(Value, "Formatting code to apply")` | `TEXT(Value; "Formatting to apply")` | |
| Return the current system date | `NOW()` | `NOW()` | |



Write me an excel formula that parses a date in cell A2 to just the year

To extract the year from a date in cell A2 in Excel, you can use the following formula:

```
=YEAR(A2)
```

This formula will return just the year part of the date in cell A2. For example, if A2 contains `01/20/2025`, the formula will return `2025`.

Large Language AI models can help you out

Microsoft CoPilot, ChatGPT…

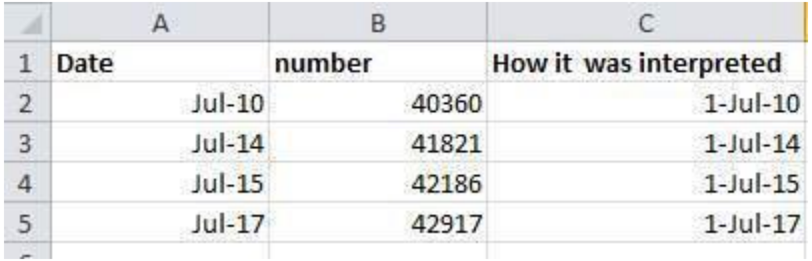
# Exchanging date information between formats

## Exercise:

- What happens to the dates in the `dates` tab of our workbook if we save this sheet in Excel (in `csv` format) and then open the file in a plain text editor (like TextEdit or Notepad)? What happens to the dates if we then open the `csv` file in Excel?

# Storing dates as YEAR, MONTH, DAY

- Storing dates in YEAR, MONTH, DAY format helps remove this ambiguity.
- For instance, in a spreadsheet recording insect counts every few days in July 2001, the data displayed as shown below in Column A. Note that the data was recorded in only one cell in each row, and the data only included reference to the month and day (`<MONTH>-<DAY>`).

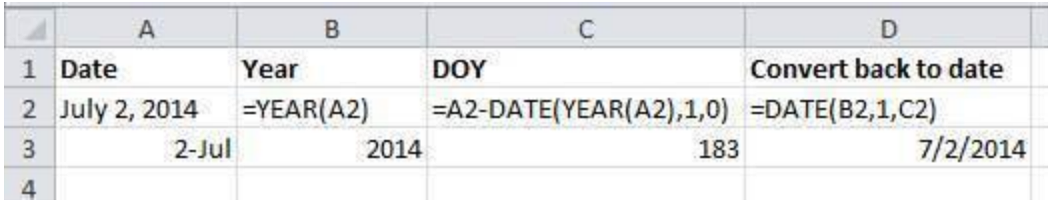


|  | A | B | C |
|---|---|---|---|
| 1 | Date | number | How it was interpreted |
| 2 | Jul-10 | 40360 | 1-Jul-10 |
| 3 | Jul-14 | 41821 | 1-Jul-14 |
| 4 | Jul-15 | 42186 | 1-Jul-15 |
| 5 | Jul-17 | 42917 | 1-Jul-17 |

- When interpreted in Excel, it appears that the observations had been recorded in 2010, 2014, 2015 and 2017 even though the data was gathered in 2001.
- Separating **date data into separate fields** (day, month, year), which eliminates any chance of ambiguity.

# Storing dates as YEAR, DAY-OF-YEAR

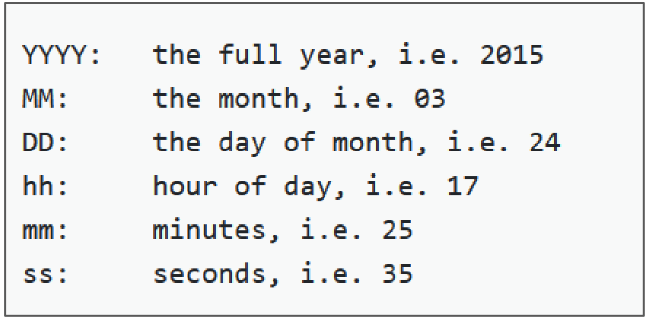
- You can also store dates as year, and day of year (DOY).
- Statistical models often incorporate year as a factor, to account for year-to-year variation, and DOY can be used to measure the passage of time within a year.



|   | A | B | C | D |
|---|---|---|---|---|
| 1 | **Date** | **Year** | **DOY** | **Convert back to date** |
| 2 | July 2, 2014 | =YEAR(A2) | =A2-DATE(YEAR(A2),1,0) | =DATE(B2,1,C2) |
| 3 | 2-Jul | 2014 | 183 | 7/2/2014 |
| 4 |  |  |  |  |

# Storing dates and times as a single string

- When dealing with dates and times, the best alternative is to convert the date string into a single string using the `YYYYMMDDhhmmss` format, following the international date standard **ISO 8601**.
- For example the date `March 24, 2015 17:25:35` would become `20150324172535`, where:



```
YYYY:   the full year, i.e. 2015
MM:     the month, i.e. 03
DD:     the day of month, i.e. 24
hh:     hour of day, i.e. 17
mm:     minutes, i.e. 25
ss:     seconds, i.e. 35
```

- Such strings will be correctly sorted in ascending or descending order, and by knowing the format they can then be correctly processed by the receiving software.
- In some systems, date is recorded as YYYY-MM-DD, e.g. 2025-01-29

# Quality Assurance

- Quality assurance stops bad data from ever being entered by checking to see if values are valid during data entry.
- For example a drop-down list of the available items or a restriction on the type of data that can be entered
- To control the kind of data entered into a spreadsheet we use Data Validation (Excel) or Validity (LibreOffice Calc), to set the values that can be entered in each data column.

**Let’s try it in the training_attendance Excel sheet…**

# Sorting

- **Bad values often sort to bottom or top of the column**.
    - For example, if your data should be numeric, then alphabetical and null data will group at the ends of the sorted data.
- Sort your data by each field, one at a time.
- Scan through each column, but pay the most attention to the top and the bottom of a column.
- If your dataset is well-structured and does not contain formulas, sorting should never affect the integrity of your dataset.

# Sorting

## Exercise:

- Let’s try this with the *Dates* tab in our messy spreadsheet. Go to that tab. Select **Data** then select **Sort**
- Sort by `len_hours` in the order *Largest to Smallest*
    - When you do this sort, do you notice anything strange?
    - Try sorting by other columns. Anything strange there?

## Conditional formatting

- Conditional formatting can color code your values by some criteria or from lowest to highest.
- This makes it easy to scan your data for outliers.
- Can be a great way to flag inconsistent values when entering data, but use with caution…
- Interoperability issues – the highlighting won’t necessarily translate to other systems.
- It is nice to be able to do these scans in spreadsheets, but we also can do these checks in a programming language like Python or R, or in OpenRefine or SQL.

# Conditional formatting

## Exercise:

- In the *Dates* tab, make sure the `num_attended` column is highlighted.
- Go to **Format** then **Conditional Formatting**.
- Apply any 2-Color Scale formatting rule.
- Now we can scan through and different colors will stand out. Do you notice any strange values?

# Exporting data from spreadsheets

- It is not a good idea to store the data you’re going to work with for your analyses in Excel file formats (`*.xls` or `*.xlsx` - depending on the Excel version) if you are hoping to preserve the data for the long term.
- Excel is a **proprietary format**,
- It is possible that in the future, technology won’t exist (or will become sufficiently rare) to make it inconvenient, if not impossible, to open the file.

- **Other spreadsheet software** may not be able to open files saved in a proprietary Excel format.
- **Different versions of Excel** may handle data differently, leading to inconsistencies.

# Exporting data from spreadsheets

- We discussed how Excel stores **dates**
- There are **multiple defaults for different versions of the software** that you can switch between
- If you’re compiling Excel-stored data from multiple sources.
    - If there’s dates in each file- Excel interprets them as their own internally consistent serial numbers.
    - When you combine the data, Excel will take the serial number from the place you’re importing it from, and interpret it using the rule set for the version of Excel you’re using.
    - You could be adding a huge error to your data, and it wouldn’t necessarily be flagged by any data cleaning methods if your ranges overlap.
- Storing data in a **universal**, **open**, **static format** will help deal with this problem.

# Exporting data from spreadsheets

- **Try comma separated values (CSV) or tab-delimited values (TSV) formats**.
- CSV files are plain text files where the columns are separated by commas, hence ‘**comma separated variables**’ or **CSV**.
- The advantage of a CSV is that we can open and read a CSV file using just about any software, including a simple **text editor**.
- Data in a CSV can also be **easily imported** into other formats and environments, such as SQLite and R. We’re not tied to a certain version of a certain program when we work with CSV, so it’s a good format to work with for maximum portability and endurance.
- Most spreadsheet programs can save to delimited text formats like CSV easily, although they complain and make you feel like you’re doing something wrong along the way.
- **Let’s export the dates tab in CSV format and see what it looks like in a text editor…**

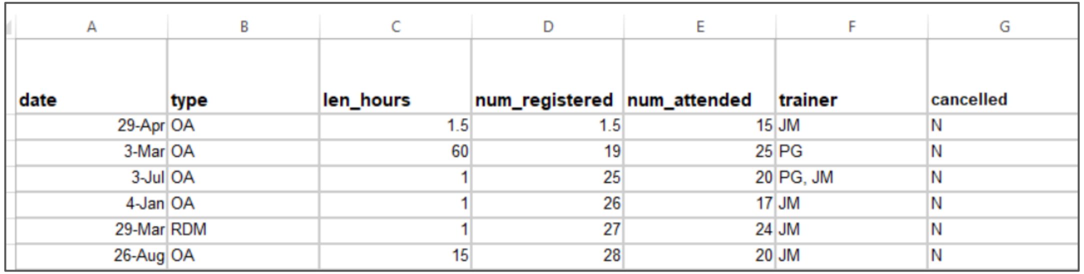
# Dealing with commas as part of data values in `*.csv` files

- We discussed how to export Excel file formats into `*.csv`. **Comma Separated Value** files, which are useful allowing for easily exchanging and sharing data.
- What happens when the data values themselves may include commas (,)?

# Dealing with commas as part of data values in *.csv files

- What happens when the data values themselves may include commas (,)?
- In that case, the software which you use (including Excel) will most likely incorrectly display the data in columns.
- It is because the commas which are a part of the data values will be interpreted as a delimiter.

- Example from training_attendance:

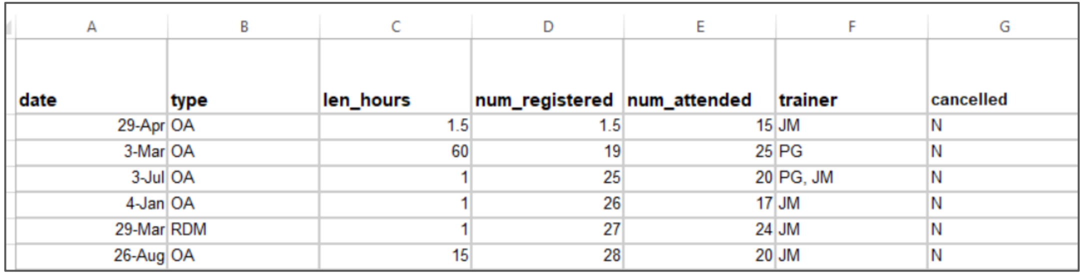


| A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|
| **date** | **type** | **len_hours** | **num_registered** | **num_attended** | **trainer** | **cancelled** |
| 29-Apr | OA | 1.5 | 1.5 | 15 | JM | N |
| 3-Mar | OA | 60 | 19 | 25 | PG | N |
| 3-Jul | OA | 1 | 25 | 20 | PG, JM | N |
| 4-Jan | OA | 1 | 26 | 17 | JM | N |
| 29-Mar | RDM | 1 | 27 | 24 | JM | N |
| 26-Aug | OA | 15 | 28 | 20 | JM | N |

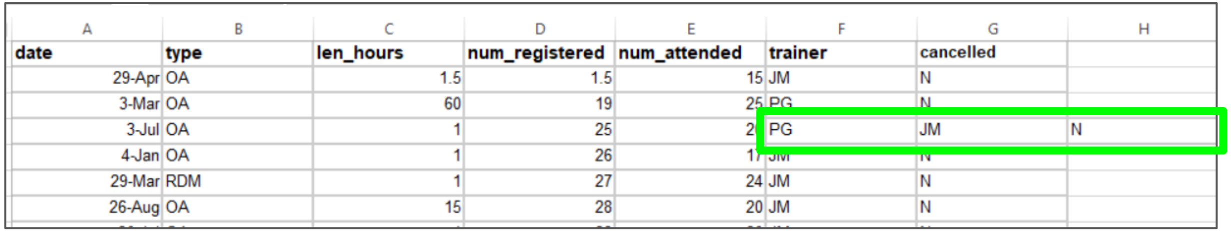
# Dealing with commas as part of data values in `*.csv` files

- What happens when the data values themselves may include commas (,)?
- In that case, the software which you use (including Excel) will most likely incorrectly display the data in columns.
- It is because the commas which are a part of the data values will be interpreted as a delimiter.

- Example from training_attendance:



| A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|
| date | type | len_hours | num_registered | num_attended | trainer | cancelled |
| 29-Apr | OA | 1.5 | 1.5 | 15 | JM | N |
| 3-Mar | OA | 60 | 19 | 25 | PG | N |
| 3-Jul | OA | 1 | 25 | 20 | PG, JM | N |
| 4-Jan | OA | 1 | 26 | 17 | JM | N |
| 29-Mar | RDM | 1 | 27 | 24 | JM | N |
| 26-Aug | OA | 15 | 28 | 20 | JM | N |



| A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|
| date | type | len_hours | num_registered | num_attended | trainer | cancelled | |
| 29-Apr | OA | 1.5 | 1.5 | 15 | JM | N | |
| 3-Mar | OA | 60 | 19 | 25 | PG | N | |
| 3-Jul | OA | 1 | 25 | 20 | PG | JM | N |
| 4-Jan | OA | 1 | 26 | 17 | JM | N | |
| 29-Mar | RDM | 1 | 27 | 24 | JM | N | |
| 26-Aug | OA | 15 | 28 | 20 | JM | N | |

# Dealing with commas as part of data values in `*.csv` files

- If you want to store your data in `csv` format and expect that your data values may contain commas, you can avoid the problem discussed above by putting the values to be included in the same column in quotes (““).
- Applying this rule, the data might look like this:

```
date,type,len_hours,num_registered,num_attended,trainer,cancelled
29 Apr,OA,1.5,1.5,15,JM,N
3 Mar,OA,60,19,25,PG,N
3 Jul,OA,1,25,20,"PG, JM",N
4 Jan,OA,1,26,17,JM,N
29 Mar,RDM,1,27,24,JM,N
```

- Opening this file as a `csv` in Excel will not lead to an extra column, because Excel will only use commas that fall outside of quotation marks as delimiting characters.
- If you are working with an already existing dataset in which the data values are not included in “” but which have commas as both delimiters and parts of data values, **you are potentially facing a major problem with data cleaning**.
- **Let’s go back to our `csv` text file though…**

# Dealing with commas as part of data values in `*.csv` files

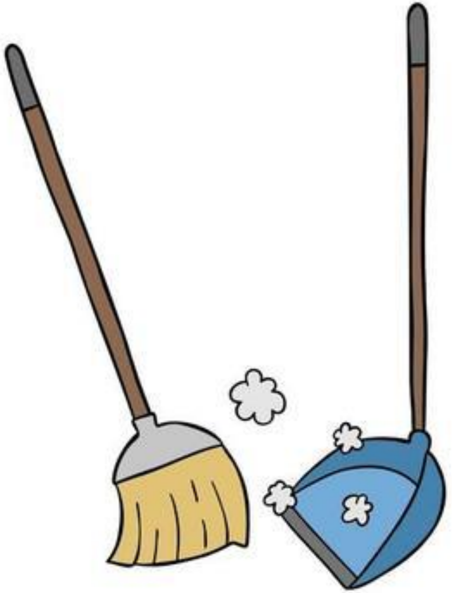
- If the dataset you’re dealing with contains hundreds or thousands of records…
  - Cleaning them up manually is not only going to take hours and hours.
  - May also potentially end up with you accidentally introducing many errors.
- Cleaning up datasets is one of the major problems in many information-based disciplines.
- The approach almost always depends on the particular context.
- However, it is a good practice to clean the data in an automated fashion, for example by writing and running a script.
  - The Python and R lessons will give you the basis for developing skills to build relevant scripts.
  - OpenRefine lessons will also give you an overview on a specific tool that’s very useful for cleaning up records

# Additional Excel Resources

## …and that’s it!

## What we covered today:

- Formatting data tables in spreadsheets
- Formatting problems
- Dates as data
- Basic quality control and data manipulation in spreadshe
- Exporting data from spreadsheets
- Data export formats caveats

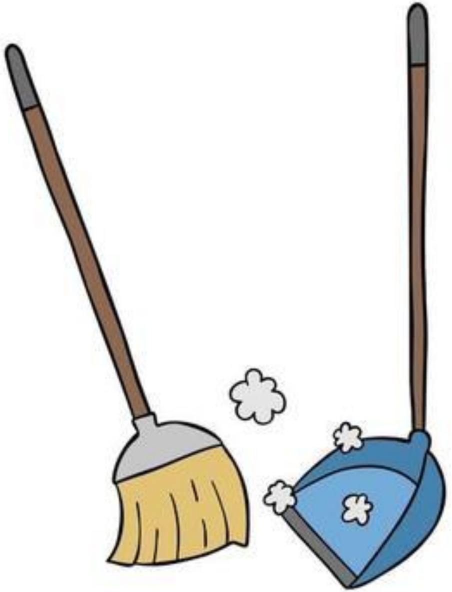


# Additional Excel Resources

**…and that's it!**

**Additional Excel Resources:**

- **Lynn Cherny** Excel Tips & Tricks for Data
- Beginning Excel (2019)
- GCF Global Excel 2016
- Excel video training (MicroSoft)
- Analyzing Library Collection Use with Excel (ALA)



# QUESTIONS?