Time Series Analysis & Forecasting Using R

1. Introduction to tsibbles

Outline

1. Time series data and tsibbles
2. Example: School students and staff
3. Lab Session 1
4. Example: Internet vacancy index
5. Filtering time series
6. Aggregating time series
7. Lab Session 2

Outline

Tidyverts packages

tidyverts.org

Tidyverts developers

Earo Wang

Mitchell O'Hara-Wild

Time series data

- Four-yearly Olympic winning times
- Annual Google profits
- Quarterly Australian beer production
- Monthly rainfall
- Weekly retail sales
- Daily IBM stock prices
- Hourly electricity demand
- 5-minute freeway traffic counts
- Time-stamped stock transaction data

Class packages

```
# Data manipulation
library(dplyr)
# Plotting functions
library(ggplot2)
# Time and date manipulation
library(lubridate)
# Time series class
library(tsibble)
# Tidy time series data
library(tsibbledata)
# Time series graphics and statistics
library(feasts)
# Forecasting functions
library(fable)
```

Class packages

```
# Data manipulation
library(dplyr)
# Plotting functions
library(ggplot2)
# Time and date manipulation
library(lubridate)
# Time series class
library(tsibble)
# Tidy time series data
library(tsibbledata)
# Time series graphics and statistics
library(feasts)
# Forecasting functions
library(fable)
```

```
# All of the above
library(fpp3)
```

`tsibble` objects

```
global_economy
```

```
# A tsibble: 15,150 x 9 [1Y]
# Key:       Country [263]
   Country     Code   Year      GDP Growth   CPI Imports Exports Population
   <fct>       <fct> <dbl>    <dbl>  <dbl> <dbl>   <dbl>   <dbl>      <dbl>
 1 Afghanistan AFG    1960   5.38e8     NA    NA    7.02    4.13    8996351
 2 Afghanistan AFG    1961   5.49e8     NA    NA    8.10    4.45    9166764
 3 Afghanistan AFG    1962   5.47e8     NA    NA    9.35    4.88    9345868
 4 Afghanistan AFG    1963   7.51e8     NA    NA   16.9     9.17    9533954
 5 Afghanistan AFG    1964   8.00e8     NA    NA   18.1     8.89    9731361
 6 Afghanistan AFG    1965   1.01e9     NA    NA   21.4    11.3     9938414
 7 Afghanistan AFG    1966   1.40e9     NA    NA   18.6     8.57   10152331
 8 Afghanistan AFG    1967   1.67e9     NA    NA   14.2     6.77   10372630
 9 Afghanistan AFG    1968   1.37e9     NA    NA   15.2     8.90   10604346
10 Afghanistan AFG    1969   1.41e9     NA    NA   15.0    10.1    10854428
# i 15,140 more rows
```

`tsibble` objects

```
global_economy
```

```
# A tsibble: 15,150 x 9 [1Y]
# Key:       Country [263]
   Country     Code   Year      GDP Growth   CPI Imports Exports Population
                      <fct> <dbl>    <dbl>  <dbl> <dbl>   <dbl>   <dbl>      <dbl>
 1 Afghanistan AFG    1960   5.38e8     NA     NA    7.02    4.13    8996351
 2 Afghanistan AFG    1961   5.49e8     NA     NA    8.10    4.45    9166764
 3 Afghanistan AFG    1962   5.47e8     NA     NA    9.35    4.88    9345868
 4 Afghanistan AFG    1963   7.51e8     NA     NA   16.9     9.17    9533954
 5 Afghanistan AFG    1964   8.00e8     NA     NA   18.1     8.89    9731361
 6 Afghanistan AFG    1965   1.01e9     NA     NA   21.4    11.3     9938414
 7 Afghanistan AFG    1966   1.40e9     NA     NA   18.6     8.57   10152331
 8 Afghanistan AFG    1967   1.67e9     NA     NA   14.2     6.77   10372630
 9 Afghanistan AFG    1968   1.37e9     NA     NA   15.2     8.90   10604346
10 Afghanistan AFG    1969   1.41e9     NA     NA   15.0    10.1    10854428
# i 15,140 more rows
```

Index

`tsibble` objects

```
global_economy
```

```
# A tsibble: 15,150 x 9 [1Y]
# Key:       Country [263]
   Country      Code   Year      GDP Growth   CPI Imports Exports Population
   <chr>        <chr> <dbl>    <dbl>  <dbl> <dbl>   <dbl>   <dbl>      <dbl>
 1 Afghanistan AFG    1960   5.38e8     NA    NA    7.02    4.13     8996351
 2 Afghanistan AFG    1961   5.49e8     NA    NA    8.10    4.45     9166764
 3 Afghanistan AFG    1962   5.47e8     NA    NA    9.35    4.88     9345868
 4 Afghanistan AFG    1963   7.51e8     NA    NA   16.9     9.17     9533954
 5 Afghanistan AFG    1964   8.00e8     NA    NA   18.1     8.89     9731361
 6 Afghanistan AFG    1965   1.01e9     NA    NA   21.4    11.3      9938414
 7 Afghanistan AFG    1966   1.40e9     NA    NA   18.6     8.57    10152331
 8 Afghanistan AFG    1967   1.67e9     NA    NA   14.2     6.77    10372630
 9 Afghanistan AFG    1968   1.37e9     NA    NA   15.2     8.90    10604346
10 Afghanistan AFG    1969   1.41e9     NA    NA   15.0    10.1     10854428
# i 15,140 more rows
```

Index Key

`tsibble` objects

```
global_economy
```

```
# A tsibble: 15,150 x 9 [1Y]
# Key:       Country [263]
   Country      Code   Year      GDP Growth   CPI Imports Exports Population
   <chr>        <chr> <dbl>      <dbl>  <dbl> <dbl>   <dbl>   <dbl>      <dbl>
 1 Afghanistan AFG    1960   5.38e8     NA    NA     7.02    4.13     8996351
 2 Afghanistan AFG    1961   5.49e8     NA    NA     8.10    4.45     9166764
 3 Afghanistan AFG    1962   5.47e8     NA    NA     9.35    4.88     9345868
 4 Afghanistan AFG    1963   7.51e8     NA    NA    16.9     9.17     9533954
 5 Afghanistan AFG    1964   8.00e8     NA    NA    18.1     8.89     9731361
 6 Afghanistan AFG    1965   1.01e9     NA    NA    21.4    11.3      9938414
 7 Afghanistan AFG    1966   1.40e9     NA    NA    18.6     8.57    10152331
 8 Afghanistan AFG    1967   1.67e9     NA    NA    14.2     6.77    10372630
 9 Afghanistan AFG    1968   1.37e9     NA    NA    15.2     8.90    10604346
10 Afghanistan AFG    1969   1.41e9     NA    NA    15.0    10.1     10854428
# i 15,140 more rows
```

Index

Key

Measured variables

`tsibble` objects

```
tourism
```

```
# A tsibble: 24,320 x 5 [1Q]
# Key:       Region, State, Purpose [304]
   Quarter Region   State           Purpose  Trips
     <qtr> <chr>    <chr>           <chr>    <dbl>
 1 1998 Q1 Adelaide South Australia Business  135.
 2 1998 Q2 Adelaide South Australia Business  110.
 3 1998 Q3 Adelaide South Australia Business  166.
 4 1998 Q4 Adelaide South Australia Business
 5 1999 Q1 Adelaide South Australia Business
 6 1999 Q2 Adelaide South Australia Business
 7 1999 Q3 Adelaide South Australia Business
 8 1999 Q4 Adelaide South Australia Business
 9 2000 Q1 Adelaide South Australia Business  154.
10 2000 Q2 Adelaide South Australia Business  169.
# i 24,310 more rows
```

Domestic visitor nights in thousands by state/region and purpose.

`tsibble` objects

```
tourism
```

```
# A tsibble: 24,320 x 5 [1Q]
# Key:       Region, State, Purpose [304]
   Quarter Region   State           Purpose  Trips
     <qtr> <chr>    <chr>           <chr>    <dbl>
 1 1998 Q1 Adelaide South Australia Business  135.
 2 1998 Q2 Adelaide South Australia Business  110.
 3 1998 Q3 Adelaide South Australia Business  166.
 4 1998 Q4 Adelaide South Australia Business  127.
 5 1999 Q1 Adelaide South Australia Business  137.
 6 1999 Q2 Adelaide South Australia Business  200.
 7 1999 Q3 Adelaide South Australia Business  169.
 8 1999 Q4 Adelaide South Australia Business  134.
 9 2000 Q1 Adelaide South Australia Business  154.
10 2000 Q2 Adelaide South Australia Business  169.
# i 24,310 more rows
```

Index

Domestic visitor nights in thousands by state/region and purpose.

`tsibble` objects

```
tourism
```

```
# A tsibble: 24,320 x 5 [1Q]
# Key:       Region, State, Purpose [304]
   Quarter Region   State           Purpose  Trips
     Index  Keys                    <chr>    <dbl>
 1 1998 Q1 Adelaide South Australia Business  135.
 2 1998 Q2 Adelaide South Australia Business  110.
 3 1998 Q3 Adelaide South Australia Business  166.
 4 1998 Q4 Adelaide South Australia Business
 5 1999 Q1 Adelaide South Australia Business
 6 1999 Q2 Adelaide South Australia Business
 7 1999 Q3 Adelaide South Australia Business
 8 1999 Q4 Adelaide South Australia Business
 9 2000 Q1 Adelaide South Australia Business  154.
10 2000 Q2 Adelaide South Australia Business  169.
# i 24,310 more rows
```

Domestic visitor nights in thousands by state/region and purpose.

`tsibble` objects

```
tourism
```

```
# A tsibble: 24,320 x 5 [1Q]
# Key:       Region, State, Purpose [304]
   Quarter Region   State           Purpose  Trips
                                             <dbl>
 1 1998 Q1 Adelaide South Australia Business  135.
 2 1998 Q2 Adelaide South Australia Business  110.
 3 1998 Q3 Adelaide South Australia Business  166.
 4 1998 Q4 Adelaide South Australia Business  127.
 5 1999 Q1 Adelaide South Australia Business  137.
 6 1999 Q2 Adelaide South Australia Business  200.
 7 1999 Q3 Adelaide South Australia Business  169.
 8 1999 Q4 Adelaide South Australia Business  134.
 9 2000 Q1 Adelaide South Australia Business  154.
10 2000 Q2 Adelaide South Australia Business  169.
# i 24,310 more rows
```

Index | Keys | Measure

Domestic visitor nights in thousands by state/region and purpose.

`tsibble` objects

- A `tsibble` allows storage and manipulation of multiple time series in R.
- It contains:
  - An index: time information about the observation
  - Measured variable(s): numbers of interest
  - Key variable(s): optional unique identifiers for each series
- It works with tidyverse functions.

The `tsibble` index

Example

```
mydata <- tsibble(
  year = 2012:2016,
  y = c(123, 39, 78, 52, 110),
  index = year
)
mydata
```

```
# A tsibble: 5 x 2 [1Y]
   year     y
  <int> <dbl>
1  2012   123
2  2013    39
3  2014    78
4  2015    52
5  2016   110
```

The `tsibble` index

For observations more frequent than once per year, we need to use a time class function on the index.

```
tsbl
```

```
# A tibble: 5 x 2
  Month    Observation
  <chr>          <dbl>
1 2019 Jan          50
2 2019 Feb          23
3 2019 Mar          34
4 2019 Apr          30
5 2019 May          25
```

The `tsibble` index

For observations more frequent than once per year, we need to use a time class function on the index.

```
tsbl |>
  mutate(Month = yearmonth(Month)) |>
  as_tsibble(index = Month)
```

```
# A tsibble: 5 x 2 [1M]
     Month Observation
     <mth>       <dbl>
1 2019 Jan          50
2 2019 Feb          23
3 2019 Mar          34
4 2019 Apr          30
5 2019 May          25
```

The `tsibble` index

Common time index variables can be created with these functions:

Frequency	Function
Annual	`start:end`
Quarterly	`yearquarter()`
Monthly	`yearmonth()`
Weekly	`yearweek()`
Daily	`as_date()`, `ymd()`
Sub-daily	`as_datetime()`

Outline

Australian school students

Download the data and inspect it

https://www.abs.gov.au/statistics/people/education/schools/latest-release

1. Download table 42b from the ABS Schools release.
2. Open in your spreadsheet viewer to inspect the structure.

Download the data and inspect it

https://www.abs.gov.au/statistics/people/education/schools/latest-release

1. Download table 42b from the ABS Schools release.
2. Open in your spreadsheet viewer to inspect the structure.

What's in the data?

- What information is available in this table?
- How ready is this data for analysis? Is it 'tidy'?

Loading ABS data into R

Use the `readxl` package to read the excel spreadsheet into R.

ABS data

Data from the ABS often contains unwanted information in the first few rows. Skip the first 4 rows with `read_excel("data.xlsx", skip = 4)`.

Loading ABS data into R

Use the `readxl` package to read the excel spreadsheet into R.

ABS data

Data from the ABS often contains unwanted information in the first few rows. Skip the first 4 rows with `read_excel("data.xlsx", skip = 4)`.

```
library(readxl)
students <- read_excel(
  "data/schools/Table 42b Number of Full-time and Part-time Students - 2006-2023.xls
  sheet = 3, skip = 4
)
```

Convert the data into a tsibble

Converting into a tsibble

Use `as_tsibble()` to convert a dataset into a tsibble.
Which column(s) are:

- The `index` variable
- Identifying `key` variable(s)
- Measured variable(s)

Hint: Look at the data

Print `students`, look at unique values with `distinct()` or get an overview of the data structure with `str()`.

Convert the data into a tsibble

```
students_tsbl <- as_tsibble(
  students,
  key = c(`State/Territory`, `Affiliation (Gov/Non-gov)`, `Affiliation (Gov/Cath/Ind
          Sex, `Aboriginal and Torres Strait Islander Status`, `School Level`,
          `National Report on Schooling (ANR) School Level`,
          `Year (Grade)`, Age),
  index = Year
)
```

```
Error in `validate_tsibble()`:
! A valid tsibble must have distinct rows identified by key and index.
i Please use `duplicates()` to check the duplicated rows.
```

Convert the data into a tsibble

```
students_tsbl <- as_tsibble(
  students,
  key = c(`State/Territory`, `Affiliation (Gov/Non-gov)`, `Affiliation (Gov/Cath/Ind
          Sex, `Aboriginal and Torres Strait Islander Status`, `School Level`,
          `National Report on Schooling (ANR) School Level`,
          `Year (Grade)`, Age),
  index = Year
)
```

```
Error in `validate_tsibble()`:
! A valid tsibble must have distinct rows identified by key and index.
i Please use `duplicates()` to check the duplicated rows.
```

! Safety included!

The `tsibble` package is strict! This helps make sure your data is correct. Use `duplicates()` to investigate problems in the data.

Detecting duplicates

```
duplicates(
  students,
  key = c(`State/Territory`, `Affiliation (Gov/Non-gov)`, `Affiliation (Gov/Cath/Ind
          Sex, `Aboriginal and Torres Strait Islander Status`, `School Level`,
          `National Report on Schooling (ANR) School Level`,
          `Year (Grade)`, Age),
  index = Year
)
```

Detecting duplicates

```
duplicates(
  students,
  key = c(`State/Territory`, `Affiliation (Gov/Non-gov)`, `Affiliation (Gov/Cath/Ind
          Sex, `Aboriginal and Torres Strait Islander Status`, `School Level`,
          `National Report on Schooling (ANR) School Level`,
          `Year (Grade)`, Age),
  index = Year
)
```

Multiple measurements

You can't make two measurements at the exact same time.
It looks like the ABS has made a mistake here!

De-duplication

When you encounter problems with the data itself, you should follow up with the data provider.

Often duplicates can be resolved by either:

- filtering to remove duplicates
- aggregating to sum over duplicates

How should we fix the data?

Is filtering or aggregation best for fixing this problem?

Tidy tsibble

```
students_tsbl <- students |>
  # Group by Year and all the character variables
  group_by(Year, across(where(is.character), identity)) |>
  # Add up the duplicate rows
  summarise(across(ends_with("count"), sum), .groups = "drop") |>
  # Convert to a tsibble
  as_tsibble(
    key = where(is.character),
    index = Year
  )
students_tsbl
```

```
# A tsibble: 81,307 x 13 [1Y]
# Key:       State/Territory, Affiliation (Gov/Non-gov), Affiliation
#   (Gov/Cath/Ind), Sex, Aboriginal and Torres Strait Islander Status,
#   School Level, National Report on Schooling (ANR) School Level, Year
#   (Grade), Age [6,575]
    Year `State/Territory` Affiliation (Gov/Non-go~1 Affiliation (Gov/Cat~2
   <dbl> <chr>             <chr>                     <chr>
```

Outline

Lab Session 1

Create a tsibble for the annual number of in-school staff.

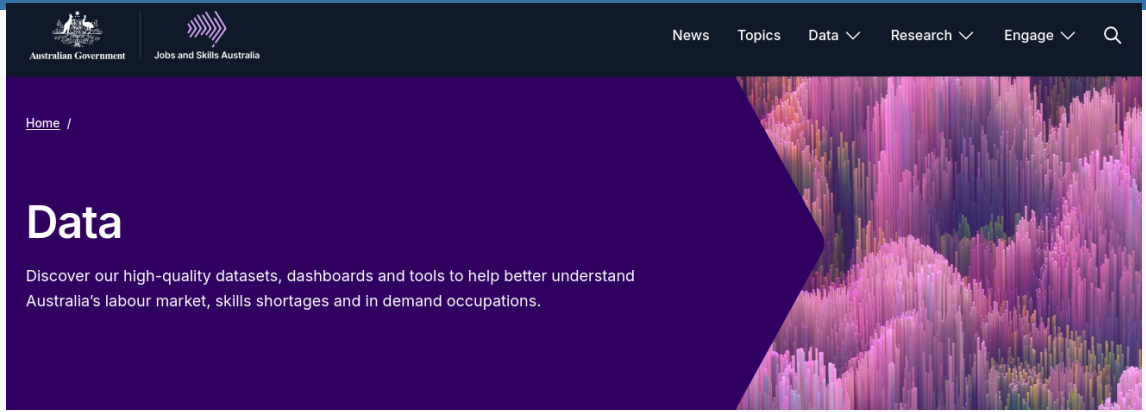
1. Download table 50a from the ABS Schools release.

   https://www.abs.gov.au/statistics/people/education/schools/latest-release

2. Look at the data to see what it contains.
3. Identify the index and key variables to convert it into a tsibble with `as_tsibble()`.
4. How many time series does this dataset contain?

Outline

Internet vacancy index



https://www.jobsandskills.gov.au/data/

Download the data and inspect it

Download the data from:

https://www.jobsandskills.gov.au/data/internet-vacancy-index

Inspect the data's structure in a spreadsheet viewer.

What's in the data?

- What information is available in this table?
- How ready is this data for analysis? Is it 'tidy'?

Read the data into R

Again, using `readxl` we'll load the data into R.

This time we don't need to skip any rows.

```
library(readxl)
read_excel("data/Internet Vacancies, ANZSCO2 Occupations, States and Territories - A
```

```
# A tibble: 513 x 228
   Level ANZSCO_CODE Title    State `38718` `38749` `38777` `38808` `38838`
   <dbl> <chr>       <chr>    <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
 1     1 0           AUSTRAL~ AUST  214082. 216621. 218877. 220183. 220369.
 2     2 1           MANAGERS AUST   20817.  21270.  21684.  22051.  22336.
 3     3 11          Chief E~ AUST     501.    497.    493.    488.    480.
 4     3 12          Farmers~ AUST     123.    123.    121.    119.    116.
 5     3 14          Hospita~ AUST    4392.   4467.   4549.   4625.   4682.
 6     3 1A          Corpora~ AUST    9160.   9349.   9531.   9688.   9769.
 7     3 1B          Constru~ AUST    4907.   5073.   5262.   5456.   5634.
 8     3 1C          Health,~ AUST    1671.   1708.   1749.   1788.   1825.
 9     2 2           PROFESS~ AUST   48643.  49904.  51061.  52114.  52964.
```

Tidy the data into a long form

The time information is in a wide (untidy) format.

We can use the `tidyr` package to convert it into a long (tidy) format with the `pivot_longer()` function.

```
read_excel("data/Internet Vacancies, ANZSCO2 Occupations, States and Territories - A
  # Tidy into a long form
  pivot_longer(matches("\\d{5}"), names_to = "month", values_to = "vacancies")
```

```
# A tibble: 114,912 x 6
   Level ANZSCO_CODE Title            State month vacancies
   <dbl> <chr>       <chr>            <chr> <chr>     <dbl>
 1     1 0           AUSTRALIAN TOTAL AUST  38718   214082.
 2     1 0           AUSTRALIAN TOTAL AUST  38749   216621.
 3     1 0           AUSTRALIAN TOTAL AUST  38777   218877.
 4     1 0           AUSTRALIAN TOTAL AUST  38808   220183.
 5     1 0           AUSTRALIAN TOTAL AUST  38838   220369.
 6     1 0           AUSTRALIAN TOTAL AUST  38869   220428.
```

Tidy the time variable

The time variable doesn't look like dates at all.

We can blame excel for this.

Excel dates

Excel stores dates as numbers, the number of days since January 1st 1900. We can convert these numbers back into dates with `as.Date(month, origin = "1900-01-01")`.

```
read_excel("data/Internet Vacancies, ANZSCO2 Occupations, States and Territories - A
  # Tidy into a long form
  pivot_longer(matches("\\d{5}"), names_to = "month", values_to = "vacancies") |>
  mutate(month = as.Date(as.integer(month), origin = "1900-01-01"))
```

Tidy the time variable

We also need to use the correct temporal granularity. In this case the data is monthly, so we need to use `yearmonth()` for the time index.

```
read_excel("data/Internet Vacancies, ANZSCO2 Occupations, States and Territories - A
  # Tidy into a long form
  pivot_longer(matches("\\d{5}"), names_to = "month", values_to = "vacancies") |>
  mutate(month = yearmonth(as.Date(as.integer(month), origin = "1900-01-01")))
```

```
# A tibble: 114,912 x 6
   Level ANZSCO_CODE Title            State    month vacancies
   <dbl> <chr>       <chr>            <chr>    <mth>     <dbl>
 1     1 0           AUSTRALIAN TOTAL AUST  2006 Jan   214082.
 2     1 0           AUSTRALIAN TOTAL AUST  2006 Feb   216621.
 3     1 0           AUSTRALIAN TOTAL AUST  2006 Mar   218877.
 4     1 0           AUSTRALIAN TOTAL AUST  2006 Apr   220183.
 5     1 0           AUSTRALIAN TOTAL AUST  2006 May   220369.
 6     1 0           AUSTRALIAN TOTAL AUST  2006 Jun   220428.
```

Create the tsibble

The data is now ready to convert into a tsibble.

```
read_excel("data/Internet Vacancies, ANZSCO2 Occupations, States and Territories - A
  # Tidy into a long form
  pivot_longer(matches("\\d{5}"), names_to = "month", values_to = "vacancies") |>
  mutate(month = yearmonth(as.Date(as.integer(month), origin = "1900-01-01"))) |>
  # Convert to a tsibble
  as_tsibble(
    key = c(Level, Title, State),
    index = month
  )
```

```
# A tsibble: 114,912 x 6 [1M]
# Key:       Level, Title, State [513]
   Level ANZSCO_CODE Title     State    month vacancies
   <dbl> <chr>       <chr>     <chr>    <mth>     <dbl>
 1     1 0           ACT TOTAL ACT   2006 Jan     3262.
 2     1 0           ACT TOTAL ACT   2006 Feb     3352.
 3     1 0           ACT TOTAL ACT   2006 Mar     3442.
```

Removing the aggregates

ABS aggregates

Data from the ABS often mixes aggregate values into the data.
This is done inconsistently (“Total”, “AUST”, etc.) and isn’t tidy.
To work with aggregates in tidy data analysis, we prefer to compute them ourselves with `group_by()` and `summarise()`.

```
read_excel("data/Internet Vacancies, ANZSCO2 Occupations, States and Territories - A
  # Tidy into a long form
  pivot_longer(matches("\\d{5}"), names_to = "month", values_to = "vacancies") |>
```

Adding extra information

The ANZSCO2 code contains two different levels of categorisation.

Our data currently only contains the titles for level 3 codes, but it is useful to add back their respective level 2 category.

```
anzsco_categories <- read_excel("data/Internet Vacancies, ANZSCO2 Occupations, State
  filter(Level == 2) |>
  distinct(ANZSCO_CODE, Title)
anzsco_categories
```

```
# A tibble: 8 x 2
  ANZSCO_CODE Title
  <chr>       <chr>
1 1           MANAGERS
2 2           PROFESSIONALS
3 3           TECHNICIANS AND TRADES WORKERS
```

Adding extra information

We can add these categories using a `left_join()`, matching the first character of the `ANZSCO_CODE` with the earlier table.

```
read_excel("data/Internet Vacancies, ANZSCO2 Occupations, States and Territories - A
  # Tidy into a long form
  pivot_longer(matches("\\d{5}"), names_to = "month", values_to = "vacancies") |>
  mutate(month = yearmonth(as.Date(as.integer(month), origin = "1900-01-01"))) |>
  # Remove aggregates
  filter(Level == 3, State != "AUST") |>
  # Add level 2 category information
  mutate(ANZSCO_CODE_CAT = substr(ANZSCO_CODE, 1, 1)) |>
  left_join(anzsco_categories, by = c("ANZSCO_CODE_CAT" = "ANZSCO_CODE"), suffix = c
  select(ANZSCO_CODE, Title_CAT, Title, State, month, vacancies) |>
  # Convert to a tsibble
  as_tsibble(
    key = c(Title_CAT, Title, State),
    index = month
  ) -> internet_vacancies
```

Data cleaning

Data janitors

Data cleaning can be tiresome and unglamourous. It's the janitorial work of the data world.
Without janitors, the world (or your analysis) will crumble.
Starting with quality data is essential for a quality analysis.

To keep this workshop focused on forecasting, I'll be giving you tidy data (and the code I wrote to tidy it) from here on.

Data cleaning

Experts claim data analysis time is 80% data cleaning and preparation, and if you work with public data that is almost certainly true!

Data cleaning

Experts claim data analysis time is 80% data cleaning and preparation, and if you work with public data that is almost certainly true!

More information

Learning and applying the principles of tidy data prevents problems later in the analysis (or worse, after the analysis is complete!).
Learn more about tidy data here:
https://tidyr.tidyverse.org/articles/tidy-data.html

Outline

Manipulating time series data

Since a `tsibble` is rectangular data (like a `data.frame`/`tibble`), we can use existing data tools to manipulate it.

I recommend using the tidyverse (especially `dplyr`) for this.

Manipulating time series data

Since a `tsibble` is rectangular data (like a `data.frame`/`tibble`), we can use existing data tools to manipulate it.

I recommend using the tidyverse (especially `dplyr`) for this.

Other data operations

It is also possible to use other data manipulation functions from other packages. The tidyverse is best for tsibble as there is specific methods added to maintain safe temporal operations.

Filtering time series

A common task is looking at specific time series.

We can use `filter()` on key variables for this.

```
internet_vacancies |>
  filter(Title == "Education Professionals")
```

```
# A tsibble: 2,016 x 6 [1M]
# Key:       Title_CAT, Title, State [9]
   ANZSCO_CODE Title_CAT     Title                  State    month vacancies
   <chr>       <chr>         <chr>                  <chr>    <mth>     <dbl>
 1 24          PROFESSIONALS Education Profession~ ACT    2006 Jan      16.6
 2 24          PROFESSIONALS Education Profession~ ACT    2006 Feb      16.2
 3 24          PROFESSIONALS Education Profession~ ACT    2006 Mar      16.4
 4 24          PROFESSIONALS Education Profession~ ACT    2006 Apr      16.5
 5 24          PROFESSIONALS Education Profession~ ACT    2006 May      16.7
 6 24          PROFESSIONALS Education Profession~ ACT    2006 Jun      16.9
 7 24          PROFESSIONALS Education Profession~ ACT    2006 Jul      17.2
 8 24          PROFESSIONALS Education Profession~ ACT    2006 Aug      17.7
```

Filtering time series

Identifying key variables

To find all the possible values for time series, you can use `key_data()` or `distinct()`.

```
internet_vacancies |>
  distinct(State)
```

```
# A tibble: 8 x 1
  State
  <chr>
1 ACT
2 NSW
3 NT
4 QLD
5 SA
```

Filtering time series

Looking at ACT

How would we further filter our `Education Professionals` series to be specific to the ACT?

Filtering time series

Looking at ACT

How would we further filter our `Education Professionals` series to be specific to the ACT?

```
internet_vacancies |>
  filter(
    Title == "Education Professionals",
    State == "ACT"
  )
```

```
# A tsibble: 224 x 6 [1M]
# Key:       Title_CAT, Title, State [1]
   ANZSCO_CODE Title_CAT     Title                  State    month vacancies
   <chr>       <chr>         <chr>                  <chr>    <mth>     <dbl>
 1 24          PROFESSIONALS Education Profession~ ACT   2006 Jan      16.6
```

Outline

Aggregating time series

The other common task is aggregating time series.

For this we use the `group_by() |> summarise()` combination.

Aggregating time series

The other common task is aggregating time series.

For this we use the `group_by() |> summarise()` combination.

Transforming time with the tidyverse

For *almost* all functions, a tsibble is identical to tibble.
A ***key*** difference is that tsibbles group the time ***index***.

Aggregating time series

Let's compute the total internet vacancies for each ANZSCO2.

```
internet_vacancies |>
  group_by(Title_CAT, Title) |>
  summarise(vacancies = sum(vacancies), .groups = "drop")
```

```
# A tsibble: 10,752 x 4 [1M]
# Key:       Title_CAT, Title [48]
   Title_CAT                          Title                month vacancies
   <chr>                              <chr>                <mth>     <dbl>
 1 CLERICAL AND ADMINISTRATIVE WORKERS Clerical and Off~ 2006 Jan      994.
 2 CLERICAL AND ADMINISTRATIVE WORKERS Clerical and Off~ 2006 Feb     1007.
 3 CLERICAL AND ADMINISTRATIVE WORKERS Clerical and Off~ 2006 Mar     1020.
 4 CLERICAL AND ADMINISTRATIVE WORKERS Clerical and Off~ 2006 Apr     1031.
 5 CLERICAL AND ADMINISTRATIVE WORKERS Clerical and Off~ 2006 May     1039.
 6 CLERICAL AND ADMINISTRATIVE WORKERS Clerical and Off~ 2006 Jun     1042.
 7 CLERICAL AND ADMINISTRATIVE WORKERS Clerical and Off~ 2006 Jul     1041.
 8 CLERICAL AND ADMINISTRATIVE WORKERS Clerical and Off~ 2006 Aug     1039.
 9 CLERICAL AND ADMINISTRATIVE WORKERS Clerical and Off~ 2006 Sep     1041.
```

Aggregating time series

This can be combined with filtering, to give the total internet vacancies for education professionals.

```
internet_vacancies |>
  filter(Title == "Education Professionals") |>
  group_by(Title_CAT, Title) |>
  summarise(vacancies = sum(vacancies), .groups = "drop")
```

```
# A tsibble: 224 x 4 [1M]
# Key:       Title_CAT, Title [1]
   Title_CAT     Title                      month vacancies
   <chr>         <chr>                      <mth>     <dbl>
 1 PROFESSIONALS Education Professionals 2006 Jan     1318.
 2 PROFESSIONALS Education Professionals 2006 Feb     1306.
 3 PROFESSIONALS Education Professionals 2006 Mar     1293.
 4 PROFESSIONALS Education Professionals 2006 Apr     1278.
 5 PROFESSIONALS Education Professionals 2006 May     1264.
 6 PROFESSIONALS Education Professionals 2006 Jun     1259.
```

Aggregating time

It is also common to produce temporal aggregates.

For example: aggregating monthly data to annual.

For this, we use the `index_by()` function (like `group_by()`).

Aggregating time

It is also common to produce temporal aggregates.

For example: aggregating monthly data to annual.

For this, we use the `index_by()` function (like `group_by()`).

```
internet_vacancies |>
  filter(Title == "Education Professionals") |>
  index_by(Year = year(month)) |>
  group_by(Title_CAT, Title) |>
  summarise(vacancies = sum(vacancies), .groups = "drop")
```

```
# A tsibble: 19 x 4 [1Y]
# Key:       Title_CAT, Title [1]
   Title_CAT     Title                    Year vacancies
   <chr>         <chr>                   <dbl>     <dbl>
 1 PROFESSIONALS Education Professionals  2006    15639.
 2 PROFESSIONALS Education Professionals  2007    19218.
```

Aggregating time

Sometimes we want to aggregate over all of time.

In this case we will no longer have a time series.

To do this, we convert back to tibble with `as_tibble()`.

```
internet_vacancies |>
  filter(Title == "Education Professionals") |>
  as_tibble() |>
  group_by(Title_CAT, Title) |>
  summarise(vacancies = sum(vacancies), .groups = "drop")
```

```
# A tibble: 1 x 3
  Title_CAT     Title                    vacancies
  <chr>         <chr>                        <dbl>
1 PROFESSIONALS Education Professionals   433861.
```

Outline

Lab Session 2

With the annual number of in-school staff:

5. Find which state has the most in-school staff in 2022.
6. Create a new tsibble of total in-school staff for each state/territory and school type (affiliation 2).
7. What is the typical difference in total number of male and female in-school staff? (try to visualise this!)