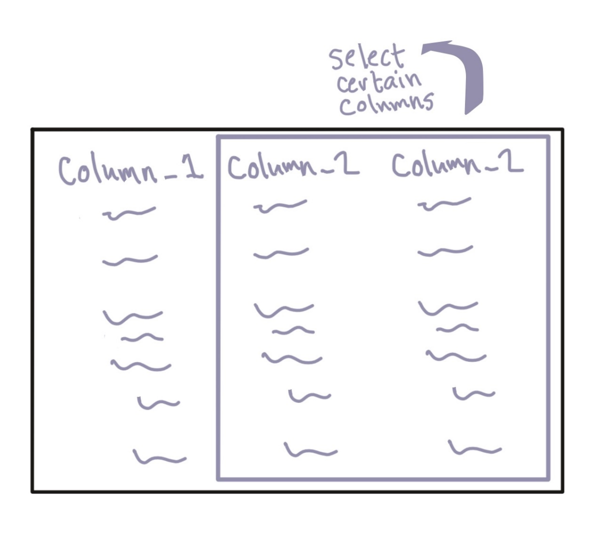
Entering / cleaning data 1

Data cleaning: Extracting columns

Extracting columns

Next, you may want to extract some columns of the dataframe. You can use the `select` function for this.



Extracting columns

You have already used `select` to extract columns by position. You can also use it to extract columns by name.

In this case, the basic structure of this command is:

```r
## Generic code
select(.data = dataframe,
       column_name_1, column_name_2, ...)
```

Where `column_name_1`, `column_name_2`, etc., are the names of the columns you want to keep.

Extracting columns

For example, to extract all columns except `year` (since that information is already included in `date`), run:

```r
select(.data = daily_show,
       job, date, category, guest_name)
```

```
## # A tibble: 2,693 x 4
##     job              date    category guest_name
##     <chr>            <chr>   <chr>    <chr>
##   1 actor            1/11/99 Acting   Michael J. Fox
##   2 Comedian         1/12/99 Comedy   Sandra Bernha~
##   3 television act~ 1/13/99 Acting   Tracey Ullman
##   4 film actress     1/14/99 Acting   Gillian Ander~
##   5 actor            1/18/99 Acting   David Alan Gr~
##   6 actor            1/19/99 Acting   William Baldw~
##   7 Singer-lyricist 1/20/99 Musician Michael Stipe
##   8 model            1/21/99 Media    Carmen Electra
##   9 actor            1/25/99 Acting   Matthew Lilla~
```

Extracting columns

As a reminder, we could have selected these columns using position, too:

```
select(.data = daily_show, 2:5)
```

```
## # A tibble: 2,693 x 4
##    job              date    category guest_name
##    <chr>            <chr>   <chr>    <chr>
##  1 actor            1/11/99 Acting   Michael J. Fox
##  2 Comedian         1/12/99 Comedy   Sandra Bernha~
##  3 television act~ 1/13/99 Acting   Tracey Ullman
##  4 film actress     1/14/99 Acting   Gillian Ander~
##  5 actor            1/18/99 Acting   David Alan Gr~
##  6 actor            1/19/99 Acting   William Baldw~
##  7 Singer-lyricist 1/20/99 Musician Michael Stipe
##  8 model            1/21/99 Media    Carmen Electra
##  9 actor            1/25/99 Acting   Matthew Lilla~
## 10 stand-up comed~ 1/26/99 Comedy   David Cross
## # ... with 2,683 more rows
```

Extracting columns

The `select` function also provides some time-saving tools. For example, in the last example, we wanted all the columns except one. Instead of writing out all the columns we want, we can use `-` with the columns we don't want to save time:

```
daily_show <- select(daily_show, -year)
head(daily_show, 3)
```

```
## # A tibble: 3 x 4
##   job               date   category guest_name
##   <chr>             <chr>  <chr>    <chr>
## 1 actor             1/11/~ Acting   Michael J. Fox
## 2 Comedian          1/12/~ Comedy   Sandra Bernha~
## 3 television actre~ 1/13/~ Acting   Tracey Ullman
```

Extracting columns

Another cool trick with `select` is that, if you want to keep several columns in a row, you can use a colon (`:`) with column names (rather than column position numbers) to select those columns:

```
daily_show <- select(daily_show, job:guest_name)
```

This call says that we want to select all columns from the one named “job” to the one named “guest_name”.

Extracting columns

The `select` function has some cool extra options that we'll explore later in the course, including:

- Selecting all columns that start with a certain pattern
- Selecting all columns that end with a certain pattern
- Selecting all columns that contain a certain pattern