Week 02 - Data cleaning and management

you

2024/1/23

Contents

Data Frames

We read in the data, but it doesn't look good. It's looking for a comma (,) to separate the columns.

Reading in Delimited Data

- This file says it's a ".csv" or **C**omma **S**eperated **V**alues.
- We use the `head` command to see that it really isn't. It seems the columns are separated by a hash sign (#) instead.

```
jobseekers <- read_csv('./w01_jobseekers.csv')
```

```
## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)
```

```
## Rows: 1866 Columns: 1
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): id#FirstName#LastName#PostCode#PhoneNumber#OnMarket#NumContacts#Job...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(jobseekers)
```

```
## # A tibble: 6 x 1
##   `id#FirstName#LastName#PostCode#PhoneNumber#OnMarket#NumContacts#JobSought`
##   <chr>
## 1 644#^Jasmine#al-(ID= 644)Mona#EX2 9BY#04464 17408#4#2#Emergency planning/mana~
## 2 1138#^Mazeed#Bi(ID=1138)ano#EX4 2BD#01219 91595#6#5#Nurse, learning disability
## 3 298#^Sandra#Beltra(ID= 298)n#EX2 4AY#+44(0)5700289298#2#2#Therapist, horticul~
## 4 1352#^Brooke#Harne(ID=1352)y#EX2 6HD#(07963) 168037#2#1#Media planner
## 5 343#^Tuhfa#Cho(ID= 343)ng#EX2 6BW#00065 12918#2#2#Ranger/warden
## 6 323#^Gabriela#Cowa(ID= 323)n#EX1 1NX#+44(0)4031790535#3#3#Accountant, charter~
```

```
tail(jobseekers)
```

```
## # A tibble: 6 x 1
##   `id#FirstName#LastName#PostCode#PhoneNumber#OnMarket#NumContacts#JobSought`
##   <chr>
## 1 1159#^Alexander#al(ID=1159)-Yusuf#EX4 4XD#+44(0)4310 69090#7#5#Information of~
## 2 1143#^Daniel#el-(ID=1143)Jamil#EX2 9JX#0583729259#4#5#Scientist, marine
## 3 996#^Abdur Razzaaq#al-Mu(ID= 996)hammed#EX1 9NX#06541 578059#3#2#Trade mark a~
## 4 160#^Paris#Pu(ID= 160)revsuren#EX4 4XD#+44(0)110819093#1#2#Sports coach
## 5 753#^Salwa#River(ID= 753)a-Garfio#EX4 8PD#+44(0)778986111#3#2#Ranger/warden
## 6 331#^Nicholas#War(ID= 331)d#EX2 6HD#+44(0)3018 701288#3#2#Software engineer
```

Using the correct separator, `#`, means it will automatically split the data into columns.

```
jobseekers <- read_delim('./w01_jobseekers.csv', delim="#")
```

```
## Rows: 1866 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: "#"
## chr (6): FirstName, LastName, PostCode, PhoneNumber, NumContacts, JobSought
## dbl (2): id, OnMarket
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(jobseekers)
```

```
## # A tibble: 6 x 8
##      id FirstName LastName   PostCode PhoneNumber OnMarket NumContacts JobSought
##   <dbl> <chr>     <chr>      <chr>    <chr>          <dbl> <chr>       <chr>
## 1   644 ^Jasmine  al-(ID= 6~ EX2 9BY  04464 17408        4 2           Emergenc~
## 2  1138 ^Mazeed   Bi(ID=113~ EX4 2BD  01219 91595        6 5           Nurse, l~
## 3   298 ^Sandra   Beltra(ID~ EX2 4AY  +44(0)5700~        2 2           Therapis~
## 4  1352 ^Brooke   Harne(ID=~ EX2 6HD  (07963) 16~        2 1           Media pl~
## 5   343 ^Tuhfa    Cho(ID= 3~ EX2 6BW  00065 12918        2 2           Ranger/w~
## 6   323 ^Gabriela Cowa(ID= ~ EX1 1NX  +44(0)4031~        3 3           Accounta~
```

Summaries of data tables

The number of rows and columns.

```
dim(jobseekers)
```

```
## [1] 1866    8
```

```
ncol(jobseekers)
```

```
## [1] 8
```

```
nrow(jobseekers)
```

```
## [1] 1866
```

The variable names.

```
colnames(jobseekers)
```

```
## [1] "id"          "FirstName"   "LastName"    "PostCode"    "PhoneNumber"
## [6] "OnMarket"    "NumContacts" "JobSought"
```

```
names(jobseekers)
```

```
## [1] "id"          "FirstName"   "LastName"    "PostCode"    "PhoneNumber"
## [6] "OnMarket"    "NumContacts" "JobSought"
```

The case names

```
head(rownames(jobseekers))
```

```
## [1] "1" "2" "3" "4" "5" "6"
```

Structure of the object.

```
str(jobseekers)
```

```
## spc_tbl_ [1,866 x 8] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ id         : num [1:1866] 644 1138 298 1352 343 ...
##  $ FirstName  : chr [1:1866] "^Jasmine" "^Mazeed" "^Sandra" "^Brooke" ...
##  $ LastName   : chr [1:1866] "al-(ID= 644)Mona" "Bi(ID=1138)ano" "Beltra(ID= 298)n" "Harne(ID=1352)y
##  $ PostCode   : chr [1:1866] "EX2 9BY" "EX4 2BD" "EX2 4AY" "EX2 6HD" ...
##  $ PhoneNumber: chr [1:1866] "04464 17408" "01219 91595" "+44(0)5700289298" "(07963) 168037" ...
##  $ OnMarket   : num [1:1866] 4 6 2 2 2 3 4 1 1 7 ...
##  $ NumContacts: chr [1:1866] "2" "5" "2" "1" ...
##  $ JobSought  : chr [1:1866] "Emergency planning/management officer" "Nurse, learning disability" "T
##  - attr(*, "spec")=
##   .. cols(
##   ..   id = col_double(),
##   ..   FirstName = col_character(),
##   ..   LastName = col_character(),
##   ..   PostCode = col_character(),
##   ..   PhoneNumber = col_character(),
##   ..   OnMarket = col_double(),
##   ..   NumContacts = col_character(),
##   ..   JobSought = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Using a data frame

There are a lot of different packages for managing data these days. We are using the `tibble` package in the `tidyverse`. But all of this will work with the basic data frames.

```
?data.frame
```

```
## starting httpd help server ... done
```

There are different ways of accessing columns / variables.

```
x <- head(jobseekers$FirstName)
y <- head(jobseekers[,'FirstName'])
head(jobseekers[['FirstName']])
```

```
## [1] "^Jasmine"  "^Mazeed"   "^Sandra"   "^Brooke"   "^Tuhfa"    "^Gabriela"
```

```
head(jobseekers[,2])
```

```
## # A tibble: 6 x 1
##   FirstName
##   <chr>
## 1 ^Jasmine
## 2 ^Mazeed
## 3 ^Sandra
## 4 ^Brooke
## 5 ^Tuhfa
## 6 ^Gabriela
```

There somewhat fewer ways of accessing rows / cases.

```
jobseekers[1,]
```

```
## # A tibble: 1 x 8
##      id FirstName LastName   PostCode PhoneNumber OnMarket NumContacts JobSought
##   <dbl> <chr>     <chr>      <chr>    <chr>          <dbl> <chr>       <chr>
## 1   644 ^Jasmine  al-(ID= 6~ EX2 9BY  04464 17408        4 2           Emergenc~
```

Note that accessing row *named* "644" is not the `id` of "644".

```
jobseekers['644',]
```

```
## # A tibble: 1 x 8
##      id FirstName LastName   PostCode PhoneNumber OnMarket NumContacts JobSought
##   <dbl> <chr>     <chr>      <chr>    <chr>          <dbl> <chr>       <chr>
## 1  1119 ^Hilmiyya Go(ID=111~ EX4 4XD  +44(0)7055~        2 2           Horticul~
```

Get `id == 644`

```
jobseekers[jobseekers$id == "644",]
```

```
## # A tibble: 6 x 8
##      id FirstName LastName   PostCode PhoneNumber OnMarket NumContacts JobSought
##   <dbl> <chr>     <chr>      <chr>    <chr>          <dbl> <chr>       <chr>
## 1   644 ^Jasmine  al-(ID= 6~ EX2 9BY  04464 17408        4 2           Emergenc~
## 2    NA <NA>      <NA>       <NA>     <NA>            NA <NA>        <NA>
## 3   644 ^Jasmine  al-(ID= 6~ EX2 9BY  04464 17408        4 2           Emergenc~
## 4    NA <NA>      <NA>       <NA>     <NA>            NA <NA>        <NA>
## 5    NA <NA>      <NA>       <NA>     <NA>            NA <NA>        <NA>
## 6   644 ^Jasmine  al-(ID= 6~ EX2 9BY  04464 17408        4 2           Emergenc~
```

The best way to do this is with tidyverse / dplyr.

```
jobseekers %>%
  filter(id == '644')
```

```
## # A tibble: 3 x 8
##      id FirstName LastName   PostCode PhoneNumber OnMarket NumContacts JobSought
##   <dbl> <chr>     <chr>      <chr>    <chr>          <dbl> <chr>       <chr>
## 1   644 ^Jasmine  al-(ID= 6~ EX2 9BY  04464 17408        4 2           Emergenc~
## 2   644 ^Jasmine  al-(ID= 6~ EX2 9BY  04464 17408        4 2           Emergenc~
## 3   644 ^Jasmine  al-(ID= 6~ EX2 9BY  04464 17408        4 2           Emergenc~
```

Your turn: Using the same code , can you filter and find people from Post code 'EX4 2PN'

```
#your code
```

Check Variable Types

```
str(jobseekers)
```

```
## spc_tbl_ [1,866 x 8] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ id         : num [1:1866] 644 1138 298 1352 343 ...
##  $ FirstName  : chr [1:1866] "^Jasmine" "^Mazeed" "^Sandra" "^Brooke" ...
##  $ LastName   : chr [1:1866] "al-(ID= 644)Mona" "Bi(ID=1138)ano" "Beltra(ID= 298)n" "Harne(ID=1352)y
##  $ PostCode   : chr [1:1866] "EX2 9BY" "EX4 2BD" "EX2 4AY" "EX2 6HD" ...
##  $ PhoneNumber: chr [1:1866] "04464 17408" "01219 91595" "+44(0)5700289298" "(07963) 168037" ...
##  $ OnMarket   : num [1:1866] 4 6 2 2 2 3 4 1 1 7 ...
##  $ NumContacts: chr [1:1866] "2" "5" "2" "1" ...
##  $ JobSought  : chr [1:1866] "Emergency planning/management officer" "Nurse, learning disability" "T
##  - attr(*, "spec")=
##   .. cols(
##   ..   id = col_double(),
##   ..   FirstName = col_character(),
##   ..   LastName = col_character(),
##   ..   PostCode = col_character(),
##   ..   PhoneNumber = col_character(),
##   ..   OnMarket = col_double(),
##   ..   NumContacts = col_character(),
##   ..   JobSought = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Most of these seem fine, except `NumContacts`, which should probably be a numeric variable as it indicates the number of days on the market. The 'table' function is used to create a frequency table - a count of occurrences for each unique value.

```
table(jobseekers$NumContacts)
```

```
##
##    1    2    3    5    6    7 five four  one  two
##  366  679  472  296   28   20    1    1    1    2
```

Converting to numeric creates some NA values ... missing values. That means that some of the text values couldn't be directly converted to numbers.

```
as.numeric(jobseekers$NumContacts)
```

```
## Warning: NAs introduced by coercion
```

```
##    [1]  2  5  2  1  2  3  3  1  2  5  3  2  1  3  1  2  2  1  2  7  1  1  3  2
##   [25]  5  3  2  1  1  2  5  2  5  1  3  1  6  1  3  1  2  5  3  1  5  3  5  1
##   [49]  2  1  3  5  2  2  2  3  7  1  5  1  3  3  2  2  3  1  2  5  2  2  1  2
##   [73]  1  3  5  3  3  5  1  2  1  3  3  2  5  2  1  3  2  2  5  3  5  3  5  1
##   [97]  3  1  5  2  2  2  3  2  2  3  2  5  2  2  5  2  3  1  2  3  1  3  3  1
##  [121]  2  3  2  5  2  3  1  1  2  2  2  3  2  2  1  1  3  2  1  2  2  2  3  5
##  [145]  1  1  3  3  1  6  3  3  2  2  3  1  3  1  5  1  6  2  1  1  3  2  3  5
##  [169]  2  1  3  5  3  2  5  2  2  1  5  1  1  1  2  3  3  1  3  5  3  5  1  1
##  [193]  2  3  6  5  1  2  1  5  2  1  5  5  2  5  1  3  5  1  3  2  3  3  2  3
##  [217]  2  5  2  2  1  3  2  5  3  3  1  3  5  6  5  1  2  5  2  5  2  2  1  3
##  [241]  3  3  7  5  1  5  1  2  1  5  2  2  2  2  2  5  1  1  1  5  1  5  5  5
##  [265]  2  5  1  5  1  1  1  2  3  2  3  3  3  2  3  3  5  5  5  1  3  2  3  3
##  [289]  3  2  3  2  1  2  3  5  5  2  3  3  2  2  7  3  1  1  1  5  3  3  5  1
##  [313]  2  2  5  1  2  2  2  3  2  1  2  2  2  3  3  2  3  2  2  1  2  2  5  3
##  [337]  2  5  2  2  3  2  5  5  5  2  1  2  3  5  2  2  5  1  2  3  2  2  2  2
##  [361]  1  5  2  3  2  3  2  2  6  2  2  2  3  3  3  3  1  2  1  1  2  3  5  5
##  [385]  1  2  1  5  2  1  2  2  3  2  3  5  5  5  2  2  5  2  2  1  5  2  3  3
##  [409]  2  2  5  2  3  1  2  1  1  1  1  3  2  2  2  2  3  1  2  3  2  2  3  5
##  [433]  2  2  2  5  2  1  3  1  3  1  3  3  3  5  1  3  2  2  2  5  3  3  3  2
##  [457]  1  3  2  2  5  2  2  1  3  2  1  5  3  5  3  3  2  3  3  3  3  3  2  5
##  [481]  5  2  2  3  2  2  3  1  5  5  3  5  3  2  1  5  7  2  2  3  2  2  1  2
##  [505]  2  5  2  2  5  2  2  5  5  1  2  3  7  1  3  2  3  1  2  3  3  5  2  2
##  [529]  2  2  5  2  3  2  5  2  2  1  2  1  5  2  2  5  3  3  2  3  2  2  3  2
##  [553]  2  3  5  2  2  5  1  2  2  3  2  1  3  1  1  3  2  3  2  2  2  3  2  2
##  [577]  3  1  1  2  3  3  3  2  1  5  3  5  3  1  2  3  2  3  3  2  1  2  1  3
##  [601]  1  2  3  2  3  3  2  5  3  1  1  1  3  5  2  1  2  3  2  3  5  3  3  3
##  [625]  5  2  2  2  1  6  3  2  5  2  2  2  1  1  2  2  5  2  5  2  3  2  6  3
##  [649]  2  2  5  5  5  2  5  5  3  2  7  2  1  5  2  1  3  5  2  5  5  3  3  5
##  [673]  2 NA  2  3  2  2  3  3  2  2  2  2  3  3  3  1  2  2  2  3  2  2  1  2
##  [697]  3  6  2  1  2  3  2  2  1  3  3  1  3  3  1  3  5  1  7  5  2  6  2  1
##  [721]  3  6  2  3  3  2  3  3  3  7  1  3  5  5  3  5  5  5  2  3  2  3  2 NA
##  [745]  3  2  3  1  1  3  7  5  2  3  2  1  2  5  1  2  2  2  2  1  2  3  5  1
##  [769]  1  3  6  2  2  3  2  2  1  2  3  3  1  2  1  3  5  3  2  1  2  3  2  3
##  [793]  2  3  5  2  2  5  1  3  6  2  2  2  2  3  2  2  1  1  1  2  2  3  2  3
##  [817]  1  2  6  2  5  7  3  5  2  3  1  1  1 NA  2  3 NA  1  3  1  2  3  2  2
##  [841]  2  3  1  1  5  1  2  1  5  2  5  1  3  2  2  3  1  2  3  1  5  1  2  1
##  [865]  1  2  2  2  2  5  1  1  5  2  1  2  3  2  2  2  3  1  1  3  1  5  2  1
##  [889]  1  2  3  3  3  2  5  2  2  1  5  1  2  2  2  5  2  2  3  2  1  1  1  3
##  [913]  2  2  2  2  2  2  3  1  3  2  2  3  3  3  1  3  1  2  3  2  3  2  1  1
##  [937]  2  2  5  3  5  3  1  3  5  3  2  2  5  2  1  2  2  2  6  2  2  3  1  3
##  [961]  3  2  5  3  6  6  1  1  1  2  3  1  1  2  1  3  3  5  3  3  2  1  2  2
##  [985]  2  2  1  1  2  2  1  2  2  1  5  2  1  2  2  2  5  5  2  1  2  2  5  3
## [1009]  2  3  1  2  3  1  2  2  2  1  3  1  5  2  1  2  2  3  1  1  5  5  3  2
## [1033]  1  1  3  2  2  2  2  3  2  2  2  3  2  3  1  2  3  5  5  3  3  5  3  2
## [1057]  2  3  5  2  3  1  1  5  3  2  3  5  3  3  1  3  6  2  2  3  2  2  3  3
## [1081]  5  1  7  1  3  2  2  3  2  2  2  1  5  2  1  5  2  1  3  5  2  1  2  3
## [1105]  3  2  6  3  3  3  5  5  1  1  1  5  5  1  3  1  2  2  3  2  2  5  1  3
## [1129]  3  3  2  2  2  2  5  5  2  2  3  2  3  2  5  2  2  1  2  1  2  7  3  3
## [1153]  5  2  2  6  1  3  1  2  1  3  5  3  5  2  2  2  3  3  1  5  3  5  2  5
## [1177]  3  2  2  1  2  3  1  2  3  2  7  1  1  3  2  1  2  3  3  2  2  3  5  3
## [1201]  2  2  3  1  3  5  2  2  2  3  5  3  3  1  3  5  3  3  1  2  2  2  3  3
## [1225]  5  3  2  2  7  2  5  1  3  5  2  3  1  3  3  2  5  3  5  3  3  5  5  3
## [1249]  3  3  3  5  3  3  2  2  5  3  5  2  3  2  1  3  2  5  2  3  3  2  2  1
## [1273]  2  1  2  2  2  2  1  5  2  3  1  1  1  5  2  3  3  2  3  3  5  5  1  1
```

```
## [1297]  5  3  1  2  2  3  2  1  1  2  1  3  2  1  3  2  2  1  5  1  2  2  5  2
## [1321]  1  1  2  3  3  1  5  6  3  5  2  5  3  2  2  5  3  2  3  2  3  1  1  2
## [1345]  2  2  3  2  6  3  1  2  2  5  1  1  5  5  1  5  3  5  2  2  2  1  2  1
## [1369]  3  1  1  6  5  2  2  1  2  2  2  3  2  5  1  3  3  2  2  2  2  3  3  2
## [1393]  2  1  2  2  3  2  5  1  2  2  2  2  2  1  2  3  2  2  5  2  3  3  5  3
## [1417]  3  3  3  1  5  2  1  1  2  2  1  2  5  5  3  2  2  3  5  3  5  5  3  2
## [1441]  2  2  2  3  3  3  1  2  3  1  3  3  1  2  3  2  1  1  1  3  2  3  1  5
## [1465]  2  5  1  1  2  2  3  2  1  2  1  5  2  1  3  5  5  5  2  2  3  3  3  2
## [1489]  5  2  1  2  2  5  5  3  2  3  2  3  5  2  5  1  5  3  1  5  3  5  1  3
## [1513]  5  2  2  2  1  1  3  2  3  5  2  1  2  2  3  2  2  2  5  3  2  2  3  3
## [1537]  5  3  1  5  3  2  1  3  5  2  5  2  2  3  2  2  2  1  1  2  3  5  3  5
## [1561]  2  3  3  3  3  2  3  2  3  1  2  2  1  3  3  3  5  2  1  6  5  2  1  1
## [1585]  3  2  1  5  2  5  1  2  2  3  7  3  6  2  2  2  2  3  1  1  1  2  3  1
## [1609]  3  5  1  2  3  2  2  1  3  1  2  1  2  2  1  2  2  5  3  5  2  5  2  2
## [1633]  1  2  5  1  5  5  5  3  3  3  2  2  2  2  3  2  2  2  2  3  1  1  3  1
## [1657]  2  1  1  1  2  2  2  5  2  5  2  1  6  2  3  2  2  2  2  7  3  5  2  2
## [1681]  3  1  7  1  2  2  3  1  3  2  2  1  5  3  2  5  2  5  5  5  1  2  2  1
## [1705]  3  5  6  2  3  2  5  2  5  2  1  2  1  1  2  2  3  2  5  2  2  1  2  2
## [1729]  3  1  2  5  1  3  1  3  1  3  5  1  1  5  2  3  2  3  3  2  2  2  5  5
## [1753]  3  6  3  2  5  3  5  2  5  3  5  2  5  3  2  2  1  3  3  5  3  3  3  5
## [1777]  1  2  1  2  2  1  2  7  3  3  1  2  2  1  1  7  5  1  3  1  2  5  2  2
## [1801]  2  2  5  2  2  1  5  5  3  2  3  2  2  3  2  5  5  2  3  3  1  2  1  1
## [1825]  2  1  3  5  2  5  2  2  2  3  1  2  2  2  2  2  5  1  2  1  1  3  1  3
## [1849]  2  3 NA  5  3  5  3  3  3  3  3  2  5  5  2  2  2  2
```

```
table(jobseekers$NumContacts,
      as.numeric(jobseekers$NumContacts),
      useNA = 'always')
```

```
## Warning in table(jobseekers$NumContacts, as.numeric(jobseekers$NumContacts), :
## NAs introduced by coercion
```

```
##
##          1   2   3   5   6   7 <NA>
##   1    366   0   0   0   0   0    0
##   2      0 679   0   0   0   0    0
##   3      0   0 472   0   0   0    0
##   5      0   0   0 296   0   0    0
##   6      0   0   0   0  28   0    0
##   7      0   0   0   0   0  20    0
##   five   0   0   0   0   0   0    1
##   four   0   0   0   0   0   0    1
##   one    0   0   0   0   0   0    1
##   two    0   0   0   0   0   0    2
##   <NA>   0   0   0   0   0   0    0
```

Convert strings into numberic values using mutate.

```
jobseekers <- jobseekers %>%
  mutate(NumContacts = case_when(
    NumContacts == 'five' ~ '5',
    NumContacts == 'four' ~ '4',
    NumContacts == 'one' ~ '1',
```

```
    NumContacts == 'two' ~ '2',
    T ~ NumContacts
  )) %>%
  mutate(NumContacts = as.numeric(NumContacts))
```

Extreme Values

Nothing extreme here. It makes sense to have up to 7 contacts with a client.

```
table(jobseekers$NumContacts)
```

```
##
##   1   2   3   4   5   6   7
## 367 681 472   1 297  28  20
```

But here something is different.

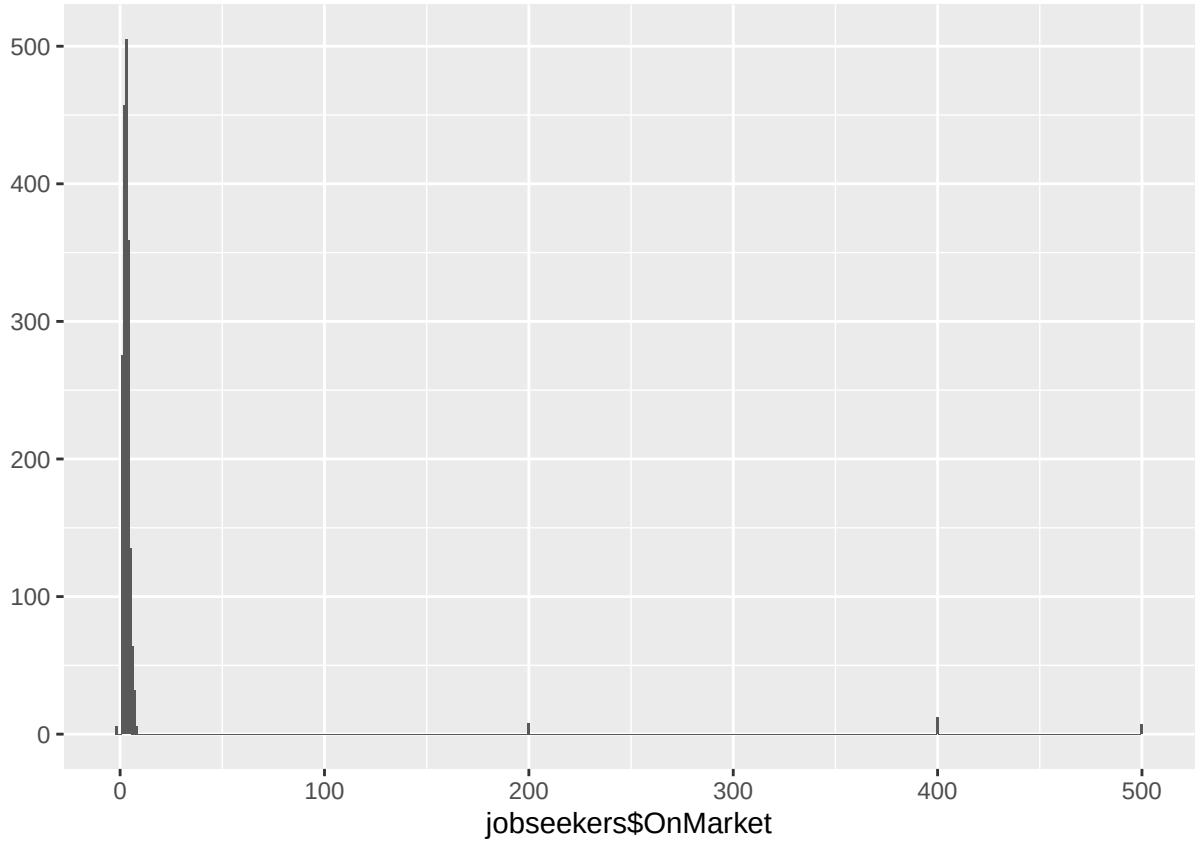
```
table(jobseekers$OnMarket)
```

```
##
##  -2   1   2   3   4   5   6   7   8 200 400 500
##   6 275 457 505 359 135  64  32   6   8  12   7
```

Really obvious if you plot it.

```
qplot(jobseekers$OnMarket, binwidth = 1)
```

```
## Warning: `qplot()` was deprecated in ggplot2 3.4.0.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



We have a decison to make. Are these errors or real values?

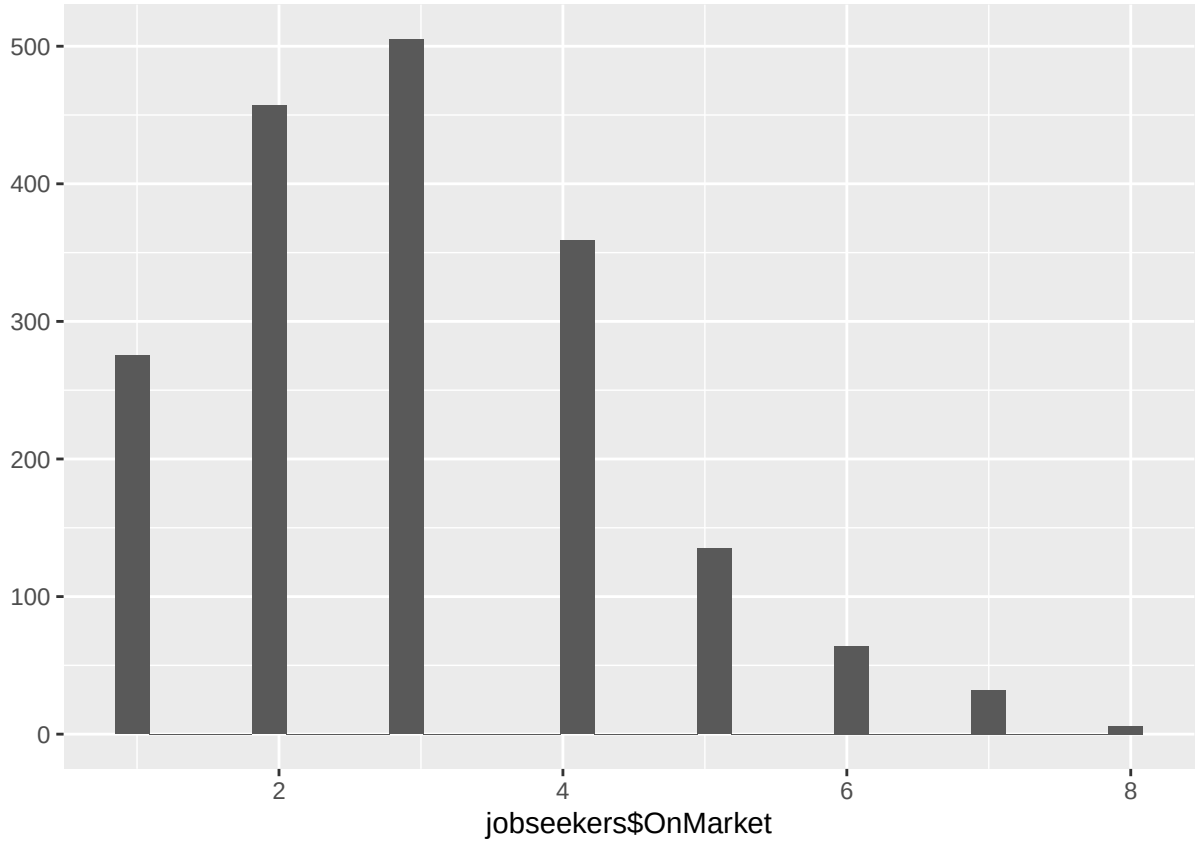
```
jobseekers <- jobseekers %>%
  mutate(OnMarket = case_when(OnMarket > 20 ~ NA_real_,
                              OnMarket < 0 ~ NA_real_,
                              T ~ OnMarket))
```

This looks better.

```
qplot(jobseekers$OnMarket)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 33 rows containing non-finite values (`stat_bin()`).
```

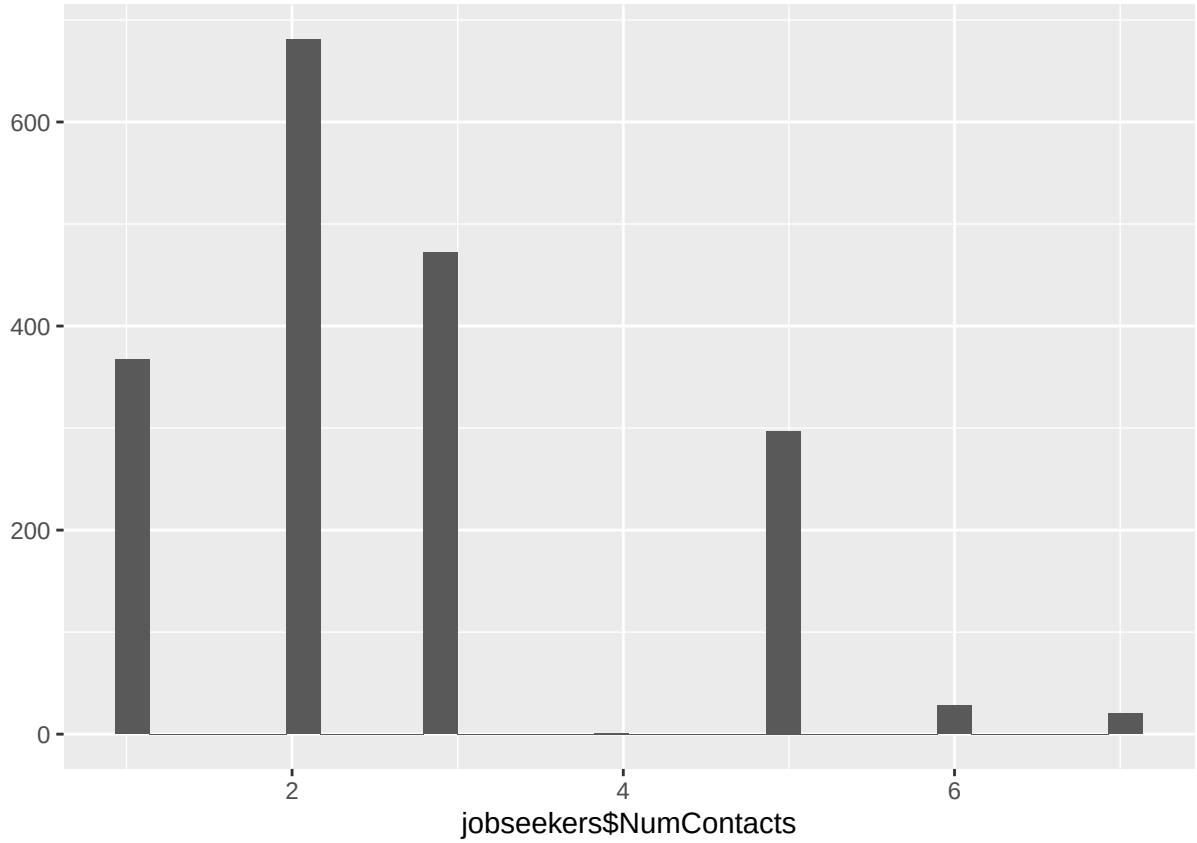


Missing data?

Here is an example. Why are there no people with 4 contacts (aside from the one we recoded from "four")?

```
qplot(jobseekers$NumContacts)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



There's often nothing to do to the data here to fix this. But maybe there's some issue with the data that needs to be examined. Are there problems with the data collection? It's possible that this is the real data, but it seems unlikely.

Missing values

The `na.omit` function will eliminate any row that has any values of NA for any variable.

```
dim(jobseekers)
```

```
## [1] 1866    8
```

```
na.omit(jobseekers) %>%
  dim
```

```
## [1] 1824    8
```

These are missing ID's

```
jobseekers[is.na(jobseekers$id),]
```

```
## # A tibble: 3 x 8
```

```
##      id FirstName LastName   PostCode PhoneNumber OnMarket NumContacts JobSought
##   <dbl> <chr>     <chr>      <chr>    <chr>           <dbl>       <dbl> <chr>
## 1    NA ^Gabriela Cowa(ID= ~ EX1 1NX  +44(0)4031~        3           3 Accounta~
## 2    NA ^Cameron  el-Ama(ID~ EX4 6QL  +44(0)4035~        3           3 Curator
## 3    NA ^Ali      Bartlin(I~ EX4 2PN  +44(0)0294~        3           2 Physiolo~
```

```
jobseekers <- na.omit(jobseekers)
```

It looks like it kicks out 1866 - 1824 = 42 cases for missing data.

Duplicate cases

There are serious duplicates here.

```
distinct(jobseekers) %>% dim
```

```
## [1] 307   8
```

```
dim(jobseekers)
```

```
## [1] 1824    8
```

```
jobseekers %>%
  group_by(id) %>%
  summarize(n = n()) %>%
  arrange(desc(n))
```

```
## # A tibble: 306 x 2
##       id     n
##    <dbl> <int>
##  1  1186    12
##  2  1348    12
##  3   504    11
##  4   761    11
##  5  1053    11
##  6  1134    11
##  7  1147    11
##  8  1242    11
##  9  1326    11
## 10   253    10
## # i 296 more rows
```

```
jobseekers %>%
  group_by(id) %>%
  summarize(n = n()) %>%
  group_by(n) %>%
  summarize(nn = n())
```

```
## # A tibble: 12 x 2
##        n    nn
##    <int> <int>
##  1     1     2
##  2     2     8
##  3     3    32
##  4     4    44
##  5     5    48
##  6     6    56
##  7     7    43
##  8     8    34
##  9     9    16
## 10    10    14
## 11    11     7
## 12    12     2
```

```
distinct(jobseekers) %>% arrange(id)
```

```
## # A tibble: 307 x 8
##       id FirstName  LastName PostCode PhoneNumber OnMarket NumContacts JobSought
##    <dbl> <chr>      <chr>    <chr>    <chr>          <dbl>       <dbl> <chr>
##  1     5 ^Talaal    Tra(ID=~ EX4 4XD  +44(0)5028~        2           2 Meteorol~
##  2     9 ^Johnathon Rona(ID~ EX4 4XD  04877 10922        4           5 Libraria~
##  3    16 ^Mardiyya  Hi(ID= ~ EX1 3LF  (06867) 86~        6           5 Fast foo~
##  4    21 ^Bailey    Ir(ID= ~ EX1 1NX  05438 7855~        2           2 Solicito~
##  5    26 ^Aasima    Cord(ID~ EX4 4XD  02274 69189        1           1 Clinical~
##  6    28 ^Colin     al-(ID=~ EX4 8PD  00387 76935        1           2 Emergenc~
##  7    32 ^Jessica   el-A(ID~ EX4 2PN  +44(0)8134~        4           5 Medical ~
##  8    40 ^April     So(ID= ~ EX2 6HD  +44(0)0070~        2           2 Soil sci~
##  9    41 ^Craig     Sala(ID~ EX4 8AY  +44(0)7226~        2           2 Therapis~
## 10    47 ^Meskerem  Bake(ID~ EX2 7DP  +44(0)4221~        4           5 Trade ma~
## # i 297 more rows
```

```
jobseekers <- distinct(jobseekers)
```

Fixing text

Text replacement

```
gsub("o", "#", "^The quick brown fox jumped over the lazy dogs.")
```

```
## [1] "^The quick br#wn f#x jumped #ver the lazy d#gs."
```

```
gsub("^", "#", "^The quick brown fox jumped over the lazy dogs.")
```

```
## [1] "#^The quick brown fox jumped over the lazy dogs."
```

```
gsub("\\^", "#", "^The quick brown fox jumped over the lazy dogs.")
```

```
## [1] "#The quick brown fox jumped over the lazy dogs."
```

Remove the caret character (^).

```
jobseekers$FirstName <- gsub('\\^', '', jobseekers$FirstName)
head(jobseekers)
```

```
## # A tibble: 6 x 8
##      id FirstName LastName   PostCode PhoneNumber OnMarket NumContacts JobSought
##   <dbl> <chr>     <chr>      <chr>    <chr>          <dbl>       <dbl> <chr>
## 1   644 Jasmine   al-(ID= 6~ EX2 9BY  04464 17408        4           2 Emergenc~
## 2  1138 Mazeed    Bi(ID=113~ EX4 2BD  01219 91595        6           5 Nurse, l~
## 3   298 Sandra    Beltra(ID~ EX2 4AY  +44(0)5700~        2           2 Therapis~
## 4  1352 Brooke    Harne(ID=~ EX2 6HD  (07963) 16~        2           1 Media pl~
## 5   343 Tuhfa     Cho(ID= 3~ EX2 6BW  00065 12918        2           2 Ranger/w~
## 6   323 Gabriela  Cowa(ID= ~ EX1 1NX  +44(0)4031~        3           3 Accounta~
```

Remove the stuff between the parentheses in `LastName`.

The regular expression pattern \(.+\) matches any substring that is enclosed in parentheses. The double backslashes are used to escape the parentheses, which have a special meaning in regular expressions. The .+ matches one or more characters of any type. The resulting regular expression matches any substring enclosed in parentheses.

```
jobseekers$LastName <- gsub('\\(.+\\)', '', jobseekers$LastName)
head(jobseekers)
```

```
## # A tibble: 6 x 8
##      id FirstName LastName PostCode PhoneNumber   OnMarket NumContacts JobSought
##   <dbl> <chr>     <chr>    <chr>    <chr>            <dbl>       <dbl> <chr>
## 1   644 Jasmine   al-Mona  EX2 9BY  04464 17408          4           2 Emergenc~
## 2  1138 Mazeed    Biano    EX4 2BD  01219 91595          6           5 Nurse, l~
## 3   298 Sandra    Beltran  EX2 4AY  +44(0)570028~        2           2 Therapis~
## 4  1352 Brooke    Harney   EX2 6HD  (07963) 1680~        2           1 Media pl~
## 5   343 Tuhfa     Chong    EX2 6BW  00065 12918          2           2 Ranger/w~
## 6   323 Gabriela  Cowan    EX1 1NX  +44(0)403179~        3           3 Accounta~
```

Saving

And that's it, you have a clean database to send to the client. Saving to an actual CSV.

```
write_csv(jobseekers, path = 'w02_jobseekers_rcleaned.csv')
```

```
## Warning: The ‘path‘ argument of ‘write_csv()‘ is deprecated as of readr 1.4.0.
## i Please use the ‘file‘ argument instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

Save the data to an R-data file. This loads the data with all the attributes and features of the original data and can save any R objects.

```
save(jobseekers, file = 'jobseekers_cleaned.rda')
```

To test it, let's remove the data from the environment.

```
rm(jobseekers)
```

Reload the data.

```
load('./jobseekers_cleaned.rda')
```

A Single Piped Block

And here it all is in a single chunk.

```
jobseekers <- read_delim('./w01_jobseekers.csv', delim="#") %>%
  mutate(OnMarket = case_when(
    OnMarket > 20 ~ NA_real_,
    OnMarket < 0 ~ NA_real_,
    T ~ OnMarket)) %>%
  mutate(NumContacts = case_when(
    NumContacts == 'five' ~ '5',
    NumContacts == 'four' ~ '4',
    NumContacts == 'one' ~ '1',
    NumContacts == 'two' ~ '2',
    T ~ NumContacts)) %>%
  mutate(NumContacts = as.numeric(NumContacts)) %>%
  mutate(FirstName = gsub('\\^', '', FirstName)) %>%
  mutate(LastName = gsub('\\(.+\\)', '', LastName)) %>%
  na.omit %>%
  distinct()
```

```
## Rows: 1866 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: "#"
## chr (6): FirstName, LastName, PostCode, PhoneNumber, NumContacts, JobSought
## dbl (2): id, OnMarket
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Sort and filter

Sort by PostCode and OnMarket

```
jobseekers %>%
  arrange(PostCode, OnMarket)
```

```
## # A tibble: 307 x 8
##       id FirstName  LastName PostCode PhoneNumber OnMarket NumContacts JobSought
##    <dbl> <chr>      <chr>    <chr>    <chr>          <dbl>       <dbl> <chr>
##  1  1397 Elizabeth  Adams    EX1 1NX  0037764280         1           2 Communit~
##  2   552 David      Pepperl  EX1 1NX  +44(0)3202~        1           1 Veterina~
##  3   300 Sabrina    Gordon   EX1 1NX  +44(0)6969~        1           1 Medical ~
##  4    21 Bailey     Ironshi~ EX1 1NX  05438 7855~        2           2 Solicito~
##  5    89 Brody      Campbell EX1 1NX  +44(0)8766~        2           2 Sports c~
##  6  1069 Ashley     Maldona~ EX1 1NX  +44(0)1967~        2           1 Meteorol~
##  7  1136 Abdus Sam~ el-Dib   EX1 1NX  05871 3851~        2           3 Fast foo~
##  8   321 Karen      Maqbool  EX1 1NX  04302807616        2           2 Accounta~
##  9   323 Gabriela   Cowan    EX1 1NX  +44(0)4031~        3           3 Accounta~
## 10   585 Matthew    Reynolds EX1 1NX  +44(0)9865~        3           3 Sports c~
## # i 297 more rows
```

Filter

```
jobseekers %>%
  filter(grepl("*Engineer*", JobSought))
```

```
## # A tibble: 6 x 8
##      id FirstName LastName   PostCode PhoneNumber OnMarket NumContacts JobSought
##   <dbl> <chr>     <chr>      <chr>    <chr>          <dbl>       <dbl> <chr>
## 1  1415 Mario     Guadarrama EX2 9JX  +44(0)5537~        2           2 Engineer~
## 2  1456 Mark      Posey      EX4 2PN  +44(0)2703~        1           2 Engineer~
## 3   651 Nyandi    Park       EX2 6HD  (09537) 32~        4           3 Engineer~
## 4   804 Jasmine   Cordova    EX2 6HD  (05620) 49~        3           3 Engineer~
## 5   852 Miguel    el-Mattar  EX4 4XD  (00975) 55~        5           3 Engineer~
## 6  1220 Blas      Sterling   EX1 1NX  (09346) 01~        4           3 Engineer~
```

Merging

We know that some of the job seekers are already been employed. But the employment data is part of a different data set.

```
employed <- read_csv('./w01_employed.csv')
```

```
## Rows: 62 Columns: 2
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): name, comp
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Create a variable for full name so we can match to the employed data.

```
jobseekers <- jobseekers %>%
  mutate(fullname = paste(FirstName, LastName))
```

Are any of the job seekers in the employed list?

```
(jobseekers$fullname %in% employed$name) %>%
  any
```

```
## [1] TRUE
```

How many?

```
(jobseekers$fullname %in% employed$name) %>%
  sum
```

```
## [1] 32
```

But as it turns out some of the names are spelled slightly differently in each place. For instance the difference between "April alHamidi" and "April al-Hamidi".

The given code uses the amatch function from the stringdist library in R to perform fuzzy matching between two sets of names. The amatch function returns the indices of the closest matches in the second set of names for each name in the first set, based on a specified distance metric and maximum distance threshold. The resulting indices are then used to create a new column in the first set of names, which contains the closest matching name from the second set.

ix <- amatch(employed$name, jobseekers$fullname, method = 'lv', maxDist = 3) Perform fuzzy matching between employed names and jobseeker full names and store the resulting indices of the closest matches in a variable called ix

employed$matched_name <- jobseekers$fullname[ix] Create a new column in the employed data frame called matched_name Populate the column with the closest matching full name from the jobseekers data frame, based on the indices in ix

```
library(stringdist)
```

```
##
## Attaching package: 'stringdist'
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
ix <- amatch(employed$name, jobseekers$fullname, method = 'lv', maxDist = 3)
employed$matched_name <- jobseekers$fullname[ix]
```

The code is used to join two data frames based on a common column and create a new data frame with all the columns from both data frames.

```
jobseekers <- left_join(jobseekers, employed,
                        c("fullname" = "matched_name"))
```

```
jobseekers %>%
  filter(!is.na(comp))
```

```
## # A tibble: 60 x 11
##       id FirstName  LastName PostCode PhoneNumber OnMarket NumContacts JobSought
##    <dbl> <chr>      <chr>    <chr>    <chr>          <dbl>       <dbl> <chr>
##  1   343 Tuhfa      Chong    EX2 6BW  00065 12918        2           2 Ranger/w~
##  2   707 Jaclyn     Gomez    EX4 8AY  (02532) 65~        1           1 Insuranc~
##  3   965 Christoph~ Le       EX4 8AY  04810 65359        4           3 Medical ~
##  4   350 Laron      Fairban~ EX2 9BY  +44(0)8062~        1           1 Maintena~
##  5  1531 Natalie    Rocken   EX2 6BH  +44(0)3264~        2           3 Solicito~
##  6  1496 Unshante   el-Badie EX4 5DW  +44(0)0979~        4           2 Emergenc~
##  7    21 Bailey     Ironshi~ EX1 1NX  05438 7855~        2           2 Solicito~
##  8   436 Jacqueline Galligan EX4 4NY  0121394244         2           2 Clinical~
##  9  1147 Sidney     Botts    EX4 6QL  0469387233         3           3 Administ~
## 10  1186 Mariah     Rivera   EX2 6HD  +44(0)6595~        3           3 Acupunct~
## # i 50 more rows
## # i 3 more variables: fullname <chr>, name <chr>, comp <chr>
```

Extra

- Can you clean the phone number?
- w01_jobs.csv has current jobs. Can you match job seekers with jobs? How would you do that?