# Printing (PDF)

## Basic printing

If a tabulation function is called from the top level, it should print out its table(s) on its own.

As usual, first, let's start up the package and pick a survey to analyze:

```r
library(surveytable)
set_survey(namcs2019sv)
```

Table 1: Survey info {NAMCS 2019 PUF}

| Variables | Observations | Design |
|---:|---:|---|
| 33 | 8,250 | Stratified 1 - level Cluster Sampling design (with replacement) With (398) clusters. namcs2019sv = survey::svydesign(ids = ~CPSUM, strata = ~CSTRATM, weights = ~PATWT  , data = namcs2019sv_df) |

Now, when a tabulation function is called from the top level, it prints. You don't need to do anything extra.

```r
tab("AGER")
```

Table 2: Patient age recode {NAMCS 2019 PUF}

| Level | n | Number (000) | SE (000) | LL (000) | UL (000) | Percent | SE | LL | UL |
|---|---:|---:|---:|---:|---:|---:|---:|---:|---:|
| Under 15 years | 887 | 117,917 | 14,097 | 93,229 | 149,142 | 11.4 | 1.3 | 8.9 | 14.2 |
| 15-24 years | 542 | 64,856 | 7,018 | 52,387 | 80,292 | 6.3 | 0.6 | 5.1 | 7.5 |
| 25-44 years | 1,435 | 170,271 | 13,966 | 144,925 | 200,049 | 16.4 | 1.1 | 14.3 | 18.8 |
| 45-64 years | 2,283 | 309,506 | 23,290 | 266,994 | 358,787 | 29.9 | 1.4 | 27.2 | 32.6 |
| 65-74 years | 1,661 | 206,866 | 14,366 | 180,481 | 237,109 | 20.0 | 1.2 | 17.6 | 22.5 |
| 75 years and over | 1,442 | 167,069 | 15,179 | 139,746 | 199,735 | 16.1 | 1.3 | 13.7 | 18.8 |

N = 8250. Checked NCHS presentation standards. Nothing to report.

If a tabulation function is called not from the top level, such as from within a loop or another function, you do need to call `print()` explicitly for it to print. For example:

```r
for (vr in c("AGER", "SEX")) {
  print( tab_subset(vr, "MAJOR", "Preventive care") )
}
```

Table 3: Patient age recode (Major reason for this visit = Preventive care) {NAMCS 2019 PUF}

| Level | n | Number (000) | SE (000) | LL (000) | UL (000) | Percent | SE | LL | UL |
|---|---|---|---|---|---|---|---|---|---|
| Under 15 years | 300 | 50,701 | 8,556 | 36,352 | 70,714 | 22.7 | 3.5 | 16.1 | 30.4 |
| 15-24 years | 121 | 18,196 | 2,889 | 13,246 | 24,996 | 8.1 | 1.2 | 5.9 | 10.9 |
| 25-44 years | 370 | 50,573 | 6,835 | 38,749 | 66,005 | 22.6 | 2.5 | 17.8 | 28.0 |
| 45-64 years | 355 | 53,805 | 9,478 | 37,982 | 76,218 | 24.1 | 3.2 | 17.9 | 31.1 |
| 65-74 years | 225 | 27,985 | 4,669 | 20,073 | 39,017 | 12.5 | 1.8 | 9.2 | 16.5 |
| 75 years and over | 197 | 22,363 | 3,805 | 15,925 | 31,404 | 10.0 | 1.7 | 6.9 | 13.8 |

N = 1568. Checked NCHS presentation standards. Nothing to report.

Table 4: Patient sex (Major reason for this visit = Preventive care) {NAMCS 2019 PUF}

| Level | n | Number (000) | SE (000) | LL (000) | UL (000) | Percent | SE | LL | UL |
|---|---|---|---|---|---|---|---|---|---|
| Female | 1,014 | 139,091 | 11,845 | 117,664 | 164,421 | 62.2 | 2.9 | 56.2 | 68.0 |
| Male | 554 | 84,532 | 10,594 | 66,039 | 108,204 | 37.8 | 2.9 | 32.0 | 43.8 |

N = 1568. Checked NCHS presentation standards. Nothing to report.

## Create HTML or PDF tables

Using a Quarto document, you can create tables in many different formats, such as HTML or PDF. Here is a straightforward example of what a Quarto document might look like:

~~~
---
title: "My tables"
author: "Me"
format: pdf
---

# Welcome

As usual, first, let's start up the package and pick a survey to analyze:

```{r, results='asis'}
library(surveytable)
set_survey(namcs2019sv, output = 'auto')
```

# Tables

Take a look at this table:

```{r, results='asis'}
tab("AGER")
```
~~~

Note the `format` setting, which specifies that this document will create PDF tables. Also note that you do have to add the `results='asis'` argument to the code chunks that print tables.

## Print using various table-making packages

Use the `output` argument of `set_opts()` to select a table-making package. By default (`output = "auto"`), `surveytable` automatically selects a package depending on whether the output is to the screen (`huxtable`), HTML (`gt`), or PDF (`kableExtra`). You can also explicitly select one of these packages.

Changing the table-making package has a couple of uses:

- It allows you to generate tables in the way that you prefer.
- It allows you to print those tables to a variety of destinations, including the screen, HTML files, or PDF files.

### `kableExtra`

```
set_opts(output = "kableExtra")
```

We have not implemented screen printing with `kableExtra` yet. Try one of the other packages.

Here is PDF:

~~~
```{r, results='asis'}
tab("AGER")
```
~~~

Table 5: Patient age recode {NAMCS 2019 PUF}

| Level | n | Number (000) | SE (000) | LL (000) | UL (000) | Percent | SE | LL | UL |
|---|---|---|---|---|---|---|---|---|---|
| Under 15 years | 887 | 117,917 | 14,097 | 93,229 | 149,142 | 11.4 | 1.3 | 8.9 | 14.2 |
| 15-24 years | 542 | 64,856 | 7,018 | 52,387 | 80,292 | 6.3 | 0.6 | 5.1 | 7.5 |
| 25-44 years | 1,435 | 170,271 | 13,966 | 144,925 | 200,049 | 16.4 | 1.1 | 14.3 | 18.8 |
| 45-64 years | 2,283 | 309,506 | 23,290 | 266,994 | 358,787 | 29.9 | 1.4 | 27.2 | 32.6 |
| 65-74 years | 1,661 | 206,866 | 14,366 | 180,481 | 237,109 | 20.0 | 1.2 | 17.6 | 22.5 |
| 75 years and over | 1,442 | 167,069 | 15,179 | 139,746 | 199,735 | 16.1 | 1.3 | 13.7 | 18.8 |

N = 8250. Checked NCHS presentation standards. Nothing to report.

### `auto`

`auto` is the default option. It automatically selects one of the above packages depending on whether the output is to the screen (`huxtable`), HTML (`gt`), or PDF (`kableExtra`).

```
set_opts(output = "auto")
#> * Printing with huxtable for screen, gt for HTML, or kableExtra for PDF.
```

PDF output (this should use `kableExtra`):

~~~
```{r, results='asis'}
tab("AGER")
```
~~~

Table 6: Patient age recode {NAMCS 2019 PUF}

| Level | n | Number (000) | SE (000) | LL (000) | UL (000) | Percent | SE | LL | UL |
|---|---|---|---|---|---|---|---|---|---|
| Under 15 years | 887 | 117,917 | 14,097 | 93,229 | 149,142 | 11.4 | 1.3 | 8.9 | 14.2 |
| 15-24 years | 542 | 64,856 | 7,018 | 52,387 | 80,292 | 6.3 | 0.6 | 5.1 | 7.5 |
| 25-44 years | 1,435 | 170,271 | 13,966 | 144,925 | 200,049 | 16.4 | 1.1 | 14.3 | 18.8 |
| 45-64 years | 2,283 | 309,506 | 23,290 | 266,994 | 358,787 | 29.9 | 1.4 | 27.2 | 32.6 |
| 65-74 years | 1,661 | 206,866 | 14,366 | 180,481 | 237,109 | 20.0 | 1.2 | 17.6 | 22.5 |
| 75 years and over | 1,442 | 167,069 | 15,179 | 139,746 | 199,735 | 16.1 | 1.3 | 13.7 | 18.8 |

N = 8250. Checked NCHS presentation standards. Nothing to report.

# Generate unformatted output

Some analysts might wish to compare the output from `surveytable` to the output from other statistical software, such as SAS / SUDAAN. In this situation, `set_opts(output = "raw")` might be useful. This command tells `surveytable` to print unformatted and unrounded tables.

```
set_opts(output = "raw")
#> * Generating unformatted / raw output.
```

```
tab("AGER")
```

```
#> Patient age recode {NAMCS 2019 PUF}
#>               Level    n    Number       SE        LL        UL   Percent
#> 1    Under 15 years  887 117916772 14097315  93228928 149142177 11.376609
#> 2       15-24 years  542  64855698  7018359  52386950  80292164  6.257277
#> 3       25-44 years 1435 170270604 13965978 144924545 200049472 16.427706
#> 4       45-64 years 2283 309505956 23289827 266994092 358786727 29.861131
#> 5       65-74 years 1661 206865982 14365993 180480708 237108637 19.958428
#> 6 75 years and over 1442 167069344 15179082 139746193 199734713 16.118849
#>          SE        LL        UL
#> 1 1.3108198  8.913995 14.238468
#> 2 0.5933708  5.138530  7.534097
#> 3 1.1296060 14.254174 18.787872
#> 4 1.3662053 27.185465 32.643562
#> 5 1.2288913 17.580833 22.505589
#> 6 1.2673229 13.689540 18.789681
#> N = 8250. Checked NCHS presentation standards. Nothing to report.
```

```
set_opts(output = "auto")
#> * Printing with huxtable for screen, gt for HTML, or kableExtra for PDF.
```

# Save the tables

## Save tables and charts to an Excel workbook

Before using Excel printing, please be sure to install these packages: `openxlsx2` and `mschart`.

To save tables and charts to an Excel file, turn on Excel printing with `set_opts( output = "Excel", file = "my_workbook" )`. Set the `file` argument to the name of an Excel file.

```
set_opts(output = "excel", file = "my_workbook")
```

```
#> * Printing to Excel file my_workbook.xlsx.
#> * NOTE: file already exists!
```

Generate some tables:

```
total()
#> * Printing Total {NAMCS 2019 PUF} to Excel workbook my_workbook.xlsx.
tab("AGER")
#> * Printing Patient age recode {NAMCS 2019 PUF} to Excel workbook my_workbook.xlsx.
```

To turn off Excel printing, set the `output` argument to a value other than `"Excel"`, such as `"auto"`:

```
set_opts(output = "auto")
#> * Printing with huxtable for screen, gt for HTML, or kableExtra for PDF.
```

## Save to a CSV file

To save tables to a CSV file, turn on CSV printing with `set_opts( output = "CSV", file = "my_output" )`. Set the `file` argument to the name of a CSV file.

```
set_opts(output = "csv", file = "my_output")
```

```
#> * Printing to CSV file my_output.csv.
#> * NOTE: file already exists!
```

Generate some tables:

```
total()
#> * Printing Total {NAMCS 2019 PUF} to CSV file my_output.csv.
tab("AGER")
#> * Printing Patient age recode {NAMCS 2019 PUF} to CSV file my_output.csv.
```

To turn off CSV printing, set the `output` argument to a value other than `"CSV"`, such as `"auto"`:

```
set_opts(output = "auto")
#> * Printing with huxtable for screen, gt for HTML, or kableExtra for PDF.
```

## Save to an R data file

Use the built-in `saveRDS()` function to save a table to an R data file:

```
tab("AGER") |> saveRDS("myfile.rds")
```

You can later load this data file back into R. To print the table, just load the file, like so:

```r
readRDS("myfile.rds")
```

Table 7: Patient age recode {NAMCS 2019 PUF}

| Level | n | Number (000) | SE (000) | LL (000) | UL (000) | Percent | SE | LL | UL |
|---|---|---|---|---|---|---|---|---|---|
| Under 15 years | 887 | 117,917 | 14,097 | 93,229 | 149,142 | 11.4 | 1.3 | 8.9 | 14.2 |
| 15-24 years | 542 | 64,856 | 7,018 | 52,387 | 80,292 | 6.3 | 0.6 | 5.1 | 7.5 |
| 25-44 years | 1,435 | 170,271 | 13,966 | 144,925 | 200,049 | 16.4 | 1.1 | 14.3 | 18.8 |
| 45-64 years | 2,283 | 309,506 | 23,290 | 266,994 | 358,787 | 29.9 | 1.4 | 27.2 | 32.6 |
| 65-74 years | 1,661 | 206,866 | 14,366 | 180,481 | 237,109 | 20.0 | 1.2 | 17.6 | 22.5 |
| 75 years and over | 1,442 | 167,069 | 15,179 | 139,746 | 199,735 | 16.1 | 1.3 | 13.7 | 18.8 |

N = 8250. Checked NCHS presentation standards. Nothing to report.

# Advanced printing

## The proper approach

Advanced users can add functionality to use **any** table-making package that they want. For more information, see `help("surveytable-options")`.

## The "quick-and-dirty" approach

The tabulation functions return either:

- for a single table, which is basically a data frame, with certain attributes set; or
- for more than one table, a list of such tables.

You can convert a single table to a `data.frame` with `as.data.frame()`, like so:

```r
tab("AGER") |> as.data.frame()
#>                Level    n Number..000. SE..000. LL..000. UL..000. Percent  SE
#> 1     Under 15 years  887       117917    14097    93229   149142    11.4 1.3
#> 2        15-24 years  542        64856     7018    52387    80292     6.3 0.6
#> 3        25-44 years 1435       170271    13966   144925   200049    16.4 1.1
#> 4        45-64 years 2283       309506    23290   266994   358787    29.9 1.4
#> 5        65-74 years 1661       206866    14366   180481   237109    20.0 1.2
#> 6 75 years and over 1442       167069    15179   139746   199735    16.1 1.3
#>     LL   UL
#> 1  8.9 14.2
#> 2  5.1  7.5
#> 3 14.3 18.8
#> 4 27.2 32.6
#> 5 17.6 22.5
#> 6 13.7 18.8
```

Note that this produces a `data.frame` with unique column names, which improves its usability.

Alternatively, you can pass this `data.frame` to your favorite table-making package. This example passes a table to `gt`. To ensure unique column names, pass the table through `as.data.frame()` first.

| Level | n | Number..000. | SE..000. | LL..000. | UL..000. | Percent | SE | LL | UL |
|---|---:|---:|---:|---:|---:|---:|---:|---:|---:|
| Under 15 years | 887 | 117917 | 14097 | 93229 | 149142 | 11.4 | 1.3 | 8.9 | 14.2 |
| 15-24 years | 542 | 64856 | 7018 | 52387 | 80292 | 6.3 | 0.6 | 5.1 | 7.5 |
| 25-44 years | 1435 | 170271 | 13966 | 144925 | 200049 | 16.4 | 1.1 | 14.3 | 18.8 |
| 45-64 years | 2283 | 309506 | 23290 | 266994 | 358787 | 29.9 | 1.4 | 27.2 | 32.6 |
| 65-74 years | 1661 | 206866 | 14366 | 180481 | 237109 | 20.0 | 1.2 | 17.6 | 22.5 |
| 75 years and over | 1442 | 167069 | 15179 | 139746 | 199735 | 16.1 | 1.3 | 13.7 | 18.8 |

```r
tab("AGER") |> as.data.frame() |> gt::gt()
```

The reason that this is the "quick-and-dirty" approach is that the output it creates is not as nice as conventional tables, described above. The output does not have table title (which has important information about the variable and the survey), table footer (which has important information about sample size and low-precision estimates), and it does not format the estimates. Nevertheless, there could be situations in which this approach is helpful, such as

- extracting an exact value from a table using `as.data.frame()`; or
- quickly using your favorite table-making package.