

# Select columns

Prof. Dr. Nicolas Meseth

This chapter introduces tools to remove unnecessary columns from the data set. Or, if stated in a positive manner, we learn how to specify the columns we need for our analysis. As with most data transformation operations, we mostly introduce functions from the `{dplyr}` package.

## The `select` command

The function `select()` is the designated tool to select columns with `{dplyr}`. By passing different things to the function, we can efficiently define the set of columns in the resulting data frame.

### By column names

The easiest and intuitive way to specify the columns we want is by listing their names. We can pass one or more column names to the `select()` function. In case of two or more, we use commas to separate the names:

```
# Just one column name
orders %>%
  select(order_id)

#> # A tibble: 2,874 x 1
#>       order_id
#>       <dbl>
#> 1 1130007101519
#> 2 1130014965839
#> 3 1130026958927
#> ...

# A list of column names
orders %>%
```

```

select(order_id, total_price)

#> # A tibble: 2,874 x 2
#>       order_id total_price
#>       <dbl>       <dbl>
#> 1 1130007101519         94.7
#> 2 1130014965839         32.2
#> 3 1130026958927         30.2
#> ...

```

When we only want a few columns, this approach works fine and is usually a good choice. I expect you apply this method in more than 90% of all cases. However, there are cases when you'd wish there was something more flexible. Luckily, there is.

## By name patterns

### Names starting with a string

Sometimes we want to select columns based on a pattern of their names. Take the orders data set as an example. Here, all variables that contain information about the shipping address have the prefix `shipping`. We leverage this with the helper function `starts_with()`:

```

orders %>%
  select(starts_with("shipping")) %>%
  colnames()

#> [1] "shipping_address_city"      "shipping_address_zip"      "shipping_address_country"
#> [4] "shipping_address_latitude" "shipping_address_longitude"

```

**Names ending with a string**

**Names with a string anywhere**

**Using regular expressions**

**By data type**

```
orders %>%
  select(where(is.numeric))

orders %>%
  select(where(is.logical))

orders %>%
  select(where(is.character))

orders %>%
  select(where(is.factor))

orders %>%
  select(where(is.list))

# The package lubridate provides a function to check for date (without time)
orders %>%
  select(where(lubridate::is.Date))

# Select all date/time columns
orders %>%
  select(where(lubridate::is.POSIXct))
```