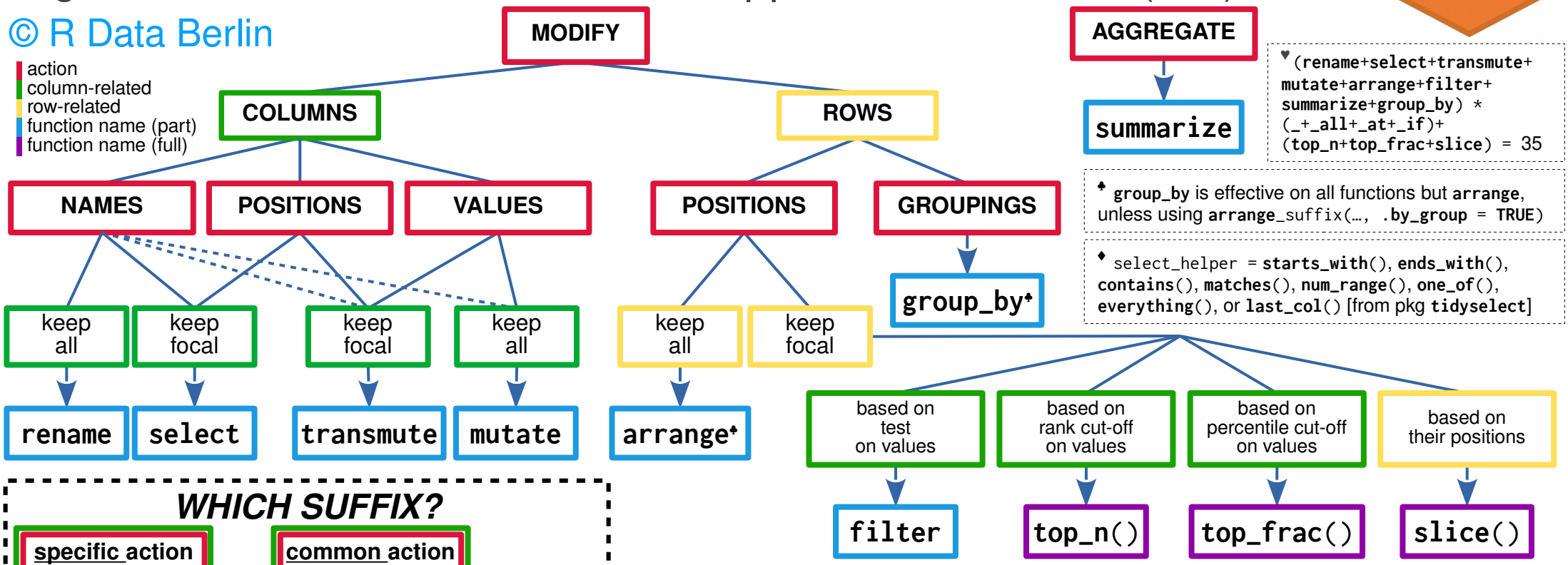
Data Transformation with `dplyr::`

A guide to 35♥ different behaviours applied to one tibble (`tbl`)

© R Data Berlin

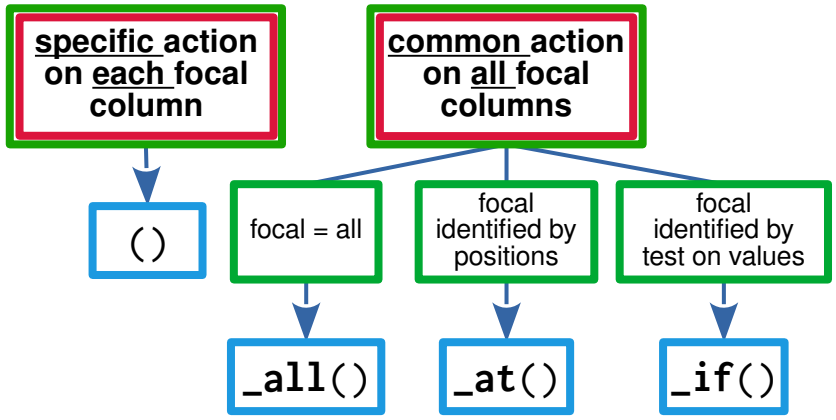


♥ (`rename`+`select`+`transmute`+`mutate`+`arrange`+`filter`+`summarize`+`group_by`) * (`_`+`_all`+`_at`+`_if`)+ (`top_n`+`top_frac`+`slice`) = 35

♠ `group_by` is effective on all functions but `arrange`, unless using `arrange_suffix(…, .by_group = TRUE)`

♦ `select_helper` = `starts_with()`, `ends_with()`, `contains()`, `matches()`, `num_range()`, `one_of()`, `everything()`, or `last_col()` [from pkg `tidyselect`]

WHICH SUFFIX?



♠ if the function `fn` does not return a scalar or a `nrow`-long vector, use `list(fn())` to nest the output ♠♠ for multiple columns dispatch use `list(suffix_for_column_names = fn)` or in case of ♠+♠♠ `list(suffix_for_column_names = ~ list(fn(.x, fn_args))`

SYNTAX: `tbl %>% verb_suffix(., …, .args)`

[font style varies to tease apart `placeholders` from true `R_commands`]

```
tbl %>% rename(new_name_columnX = old_name_columnX)
tbl %>% select(name_columnX, new_name_columnY = old_name_columnY, select_helper♦("text"))
tbl %>% transmute(name_columnZ = fn_scalar_or_vectorized♠(name_columnX))
tbl %>% mutate(name_columnZ = fn_scalar_or_vectorized♠(name_columnX))
tbl %>% arrange(name_columnX, desc(name_columnY))
tbl %>% filter(fn_test_vectorized(name_columnX), fn_test_vectorized(name_columnY))
tbl %>% top_n(n = number, name_columnX); tbl %>% top_frac(n = fraction, name_columnX)
tbl %>% slice(row_indices)
tbl %>% summarize(name_columnZ = fn_scalar♠(name_columnX))
tbl %>% group_by(name_columnX, name_columnY) %>% verb_suffix()
tbl %>% verb_all(.funs = fn♠♠, fn_arg_name1 = fn_arg1,fn_arg_name2 = fn_arg2) # included below
tbl %>% verb_at(.vars = indices_for_focal_columns OR .vars = vars(select_helper♦("text")), )
tbl %>% verb_if(.predicate = logicals_for_each_column, )
```