



Пакет dplyr. Манипуляции с таблицами 1



Пакет dplyr

Информация о пакете:

<https://cran.rstudio.com/web/packages/dplyr/>

<https://dplyr.tidyverse.org/>

Установка пакета:

```
install.packages("dplyr")
```

Запуск пакета:

```
library(dplyr)
```

Функция `select()` – выбор столбцов

`select(table, column1, column2, ...)` # выбрать несколько столбцов по названиям или номерам

Различные способы выбора:

`select(tbl, column1:column10)` # выбрать столбцы от первого до десятого

`starts_with("X")`: названия начинающиеся с "X"

`ends_with("X")`: названия заканчивающиеся на "X"

`contains("X")`: все названия содержащие "X"

`num_range("x", 1:5)`: столбцы с порядковыми номерами: x01, x02, x03, x04, x05

`one_of(c("col1", "col2", "col3"))`: все названия из вектора текстовых значений

Функция `filter()` 1 – фильтр по условию

`filter(table, column == 1)` # фильтровать данные на основе условий

Примеры:

`filter(tbl, column == "Tree")`

`Column < 20` – меньше

`Column <= 30` – больше


`Column == 30` или `Column == "Female"` – равно

`Column != 30` или `Column != "Female"` – не равно

`Column >= 40` – больше или равно

`Column > 50` – больше

`Column %in% c(20, 30, 40)` – равно всем значениям в указанном векторе



Функция `filter()` 2 – фильтр по УСЛОВИЯМ

Несколько условий:

«И» - 2 условия одновременно:

```
filter(table, a > 0 & b > 0)
```

«ИЛИ» – либо одно условие, либо другое:

```
filter(table, a > 0 | b > 0)
```




Функция `arrange()` - сортировка

`arrange(table, column1)` # сортировка в восходящем порядке

`arrange(table, desc(column1))` # сортировка в нисходящем порядке

`arrange(table, column1, column2)` # сортировка по двум столбцам



Функция `mutate()` – создание нового столбца

`mutate(tbl, new_column = выражение)` # создать новый столбец на основе данных из других столбцов

`mutate(tbl, new_column = column1 + column2)`

`mutate(tbl, new_column = column1^2)`

`mutate(tbl, new_column = column1 / column2)`

Оператор pipe - %>% (1)

Оператор pipe позволяет использовать несколько функций на одной таблице последовательно

Название таблицы мы выносим за пределы функции в начало строки

```
tbl1 <- tbl %>% filter(column1 == 1) %>% mutate(new_column = column2  
- column3)
```

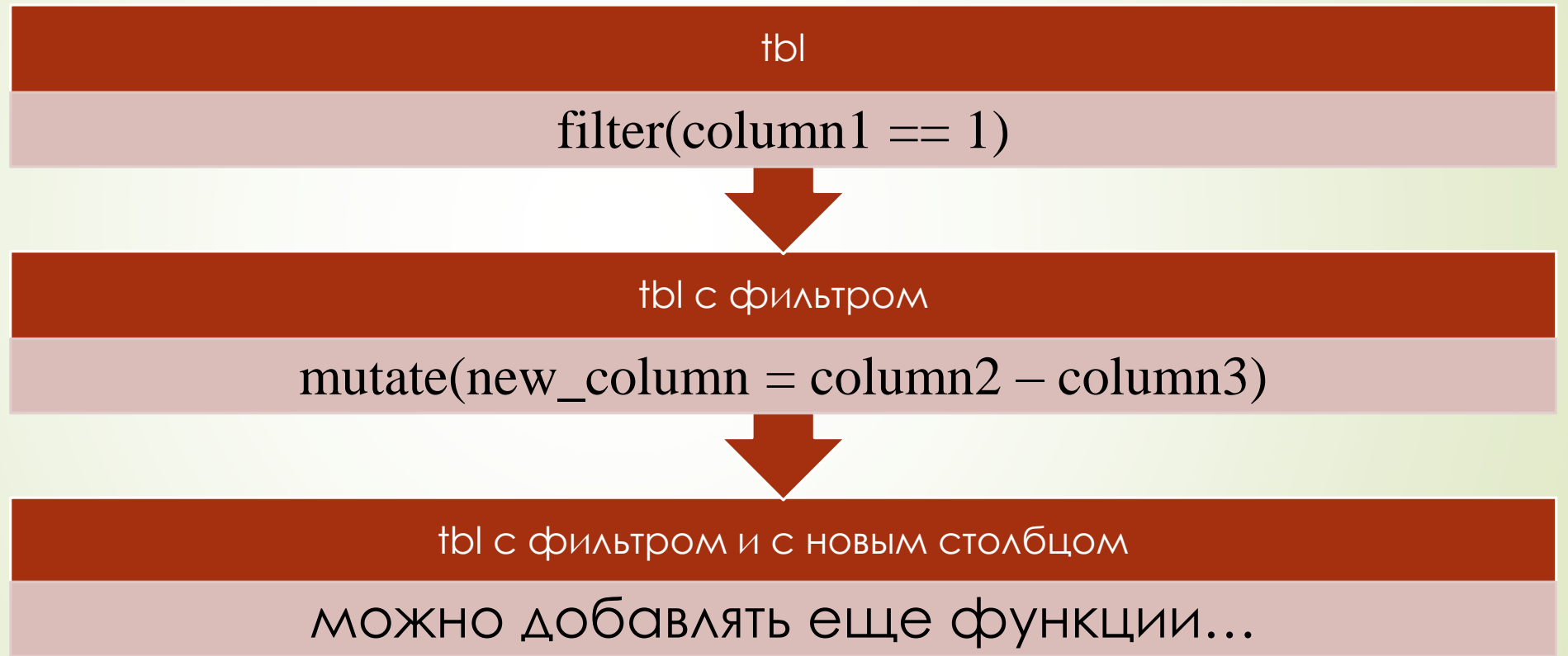
Для удобства чтения длинной строки кода, можно писать так:

```
tbl %>%
```

```
  filter(column1 == 1) %>%
```

```
  mutate(new_column = column2 - column3)
```


Оператор pipe - %>% (2)



Функции `group_by()` и `summarize()` - 1

Используются в связке

```
tbl %>% group_by(column1) %>% summarize(...)
```

Функция `group_by()` делит таблицу на группы по указанному столбцу, и дальнейшие действия будут осуществляться внутри групп

Функция `summarize()` получает новую таблицу по функциям указанным в ней (следующий слайд)

Функции `group_by()` и `summarize()` - 2

`summarise(tbl, new = function(column))` # в результате получается новая таблица

`min(x)` – минимальное значение

`max(x)` – максимальное значение

`mean(x)` – среднее

`median(x)` – медиана

`sd(x)` – стандартное отклонение

`first(x)` – первый элемент

`last(x)` – последний элемент

`nth(x, n)` – элемент под номером n

`n()` – количество строк по группам (ничего не пишем внутри скобок!)

`mean(x == "value")` – с помощью функции `mean()` можно посчитать процент: в данном случае процент значений "value" в столбце x

Функции `group_by()` и `summarize()` - 3

Примеры:

```
tbl %>% group_by(column1) %>% summarize(new_column = n())
```

```
tbl %>% group_by(column1) %>% summarize(min = min(column2),  
max = max(column2), mean = mean(column2), median =  
meadian(column2))
```

```
tbl %>% group_by(column1) %>% summarize(percent_male =  
mean(column2 == "Male"))
```