Урок 3. Работа с данными с библиотекой tidyverse

В этом уроке мы поговорим с библиотекой `tidyverse`. Библиотека `tidyverse` — библиотека для удобной работы с данными, которая активно используется в аналитике. С ее помощью можно более быстро получать описание датафрейма, сохранять полученные результаты, группировать наблюдения по определенному признаку, а также строить красивые графики. Научиться работать с это библиотекой несложно, нужно только понять общую логику, познакомиться с особыми операторами и функциями, а также немного попрактиковаться.

Для начала установим библиотеку и обратимся к ней:

```
install.packages("tidyverse")
```

```
library(tidyverse)
```

Загрузим все тот же файл с фирмами:

```
dat <- read.csv("firms.csv")
```

Теперь приступим к работе.

На самом деле, библиотека `tidyverse` — это набор библиотек, используемых для схожих целей — обработки и визуализации данных. В ней можно выделить две важные составляющие: библиотеку `dplyr` для работы с таблицами и библиотеку `ggplot2` для визуализации (про нее поговорим позже).

В библиотеке `dplyr` есть особый оператор `%>%`, который позволяет выполнять операции пошагово. Смысл этого оператора такой: возьми, то, что слева от `%>%` и передай это на вход функции, стоящей справа от `%>%`. Посмотрим на простом примере:

```
dat %>% View
```

Взять датафрейм `dat` и подать ее на вход функции `View`. Как можно заметить, во `View` уже нет ни скобок, ни названия базы, потому что они и не нужны – R и так знает, с чем ему работать.

Рассмотрим другой пример. Возьмем базу `dat`, сначала выберем столбцы `MarketID`, `Promotion` и `SalesInThousands`, а потом запросим несколько первых строк таблицы:

```
# select - выбор столбцов
dat %>% select(MarketID, Promotion, SalesInThousands) %>%
  head %>% View
```

Теперь отфильтруем строки — выберем только компании среднего уровня с доходом выше 40, а затем выберем только столбцы с числом лет на рынке и числом продаж:

```
# filter - выбор строк
dat %>% filter(MarketSize == "Medium",
               SalesInThousands > 40) %>%
  select(AgeOfStore, SalesInThousands) %>% View
```

Еще одна полезная функция – `mutate()` – используется для создания и добавлению в датафрейм нового столбца. Добавим столбец `LogSales` – натуральный логарифм числа продаж.

```
dat %>% mutate(LogSales = log(SalesInThousands))
```

Если теперь мы посмотрим на базу `dat` привычным образом, нас будет ждать сюрприз:

```
View(dat)
```

Переменной `LogSales` в датафрейме нет! Почему? Дело в том, что когда мы проделываем что-то с базой с помощью `dplyr` и не сохраняем результат, изменения в самом датафрейме не происходят. Как

сохранить изменения? Очень просто: как всегда, сохранить результат в переменную, в которой хранится датафрейм:

```
dat <- dat %>% mutate(LogSales = log(SalesInThousands))
View(dat)
```

Добавлять можно и более одной переменной за раз: достаточно перечислить их через запятую внутри `mutate()`. Предлагаю перейти к практическому заданию и попробовать!