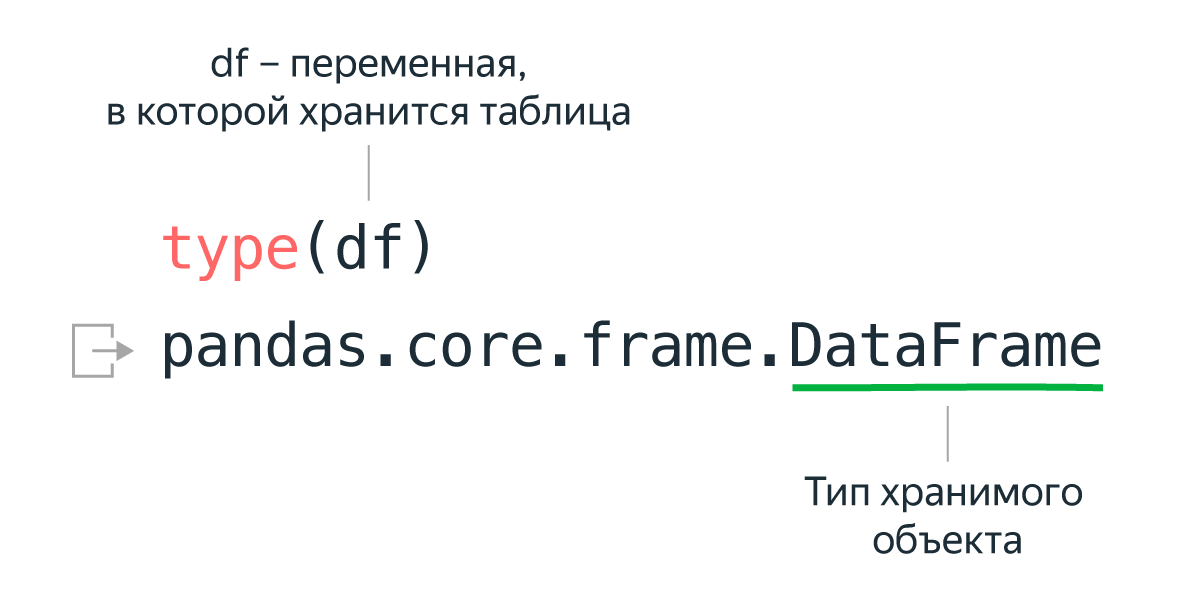
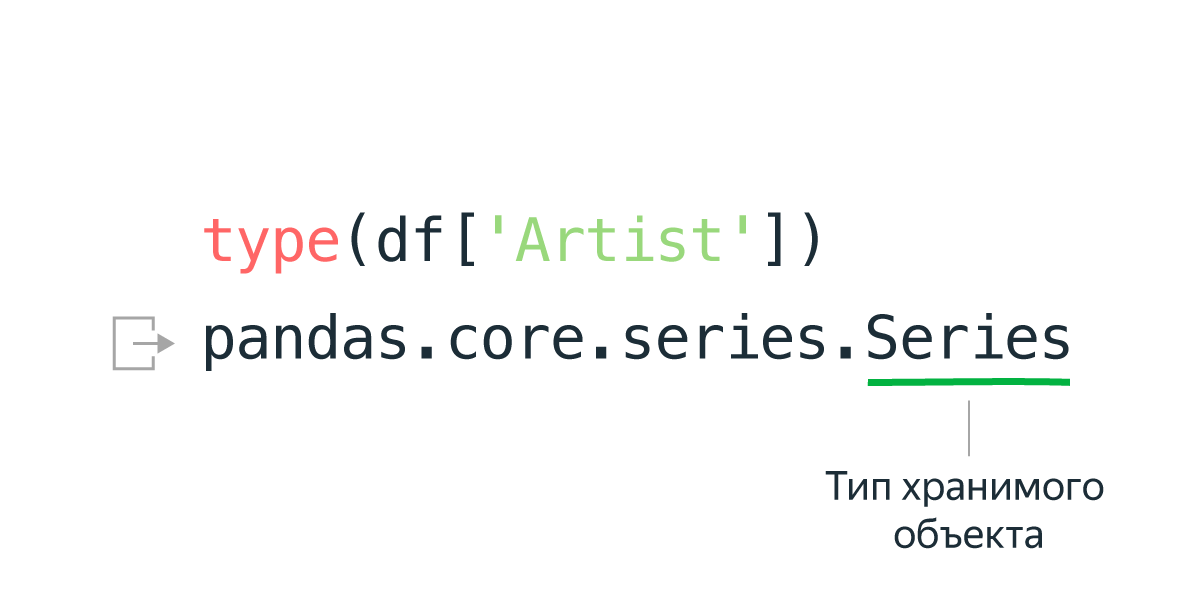
## 7. Объект Series

В таблице, которую мы рассматривали весь прошлый урок, каждый столбец сам по себе — вовсе не структура данных DataFrame. Удивлены? Давайте проверим.

Таблица всё так же имеет тип DataFrame.

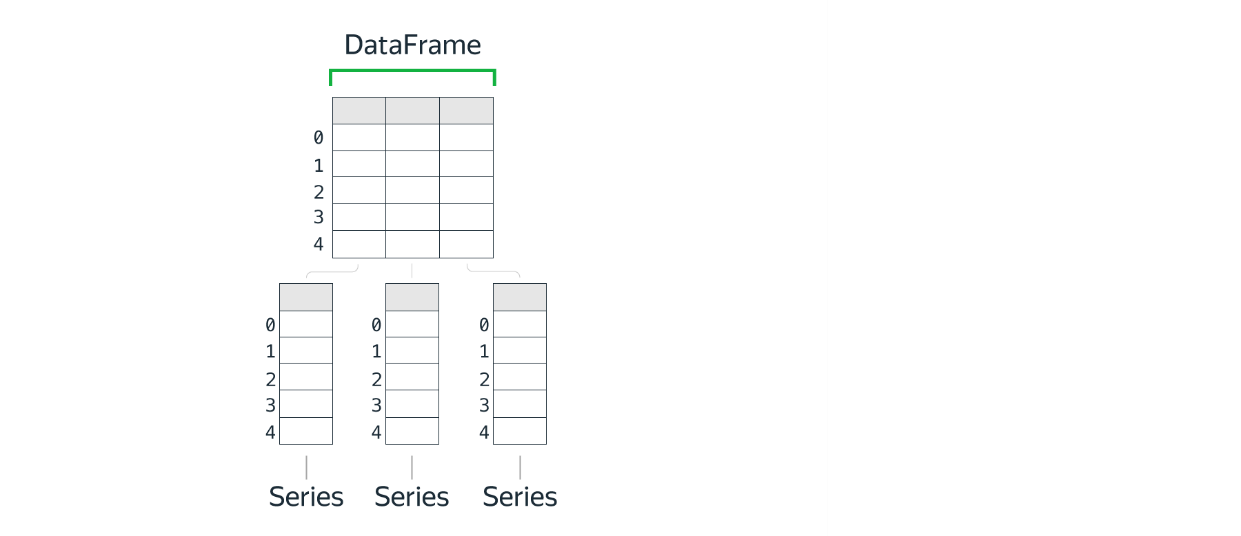


Но если мы возьмём отдельный столбец таблицы, то он представляет собой совсем иную структуру данных — **Series** (англ. series, «серия»).



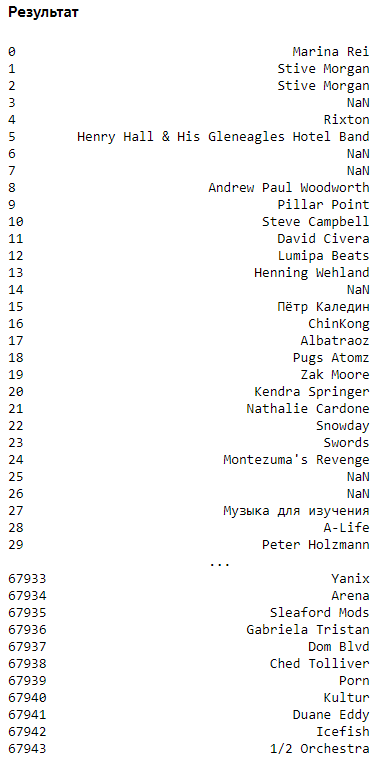
Давайте разберёмся, что интересного в новом объекте Series.

Series — одномерная таблица, и её элементы можно получить по индексу. Каждый индекс — это номер отдельного наблюдения, и поэтому несколько различных Series вместе составляют DataFrame. В Series хранятся данные одного типа.



У каждой Series есть имя (Name), информация о количестве данных в столбце (Length) и тип данных, которые хранятся в ней (dtype).

print(df['Artist'])



Не все данные в этом столбце Artist относятся к типу object — есть и пропущенные значения NaN. О них мы подробно расскажем в теме, посвящённой «информационному мусору». Вместе с ними длина столбца равна общему числу наблюдений 67963.

Индексация в Series аналогична индексации элементов столбца в DataFrame. Давайте рассмотрим на примере. Сохраним столбец total play в переменной total\_play.

total\_play = df.loc[:, 'total play']

Для получения пятого по индексу элемента укажем 5 в квадратных скобках.

print(total\_play[5])

Если надо получить диапазон ячеек, запросите атрибут loc с границами среза в квадратных скобках.

print(total\_play.loc[5:10])

Результат

5 3.007

6 0.100

7 211.880

8 3.161

9 274.390

10 8.836

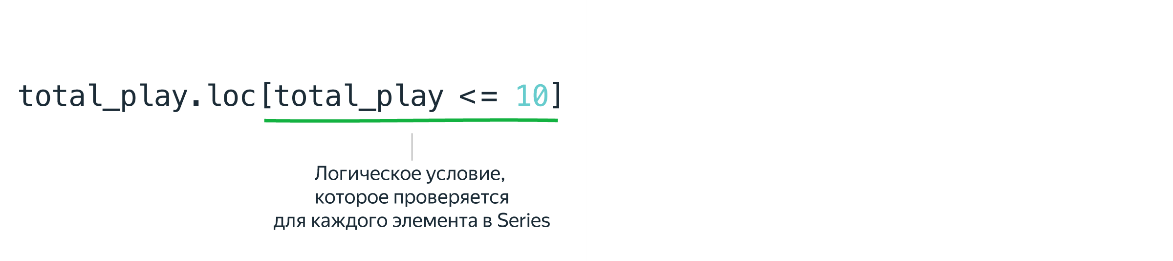
Name: total play, dtype: float64

Вот таблица различных вариантов индексации в Series с сокращённой записью.



Для Series также возможна логическая индексация. Рассмотрим такой пример — пользователю может не понравиться песня, которая начала играть, и он нажмёт кнопку «Следующий трек». Тогда в таблице сохраняется очень малое время прослушивания — от нуля до нескольких секунд. Вы можете проверить, сколько пользователей в течение нескольких секунд — не более 10 — приняли решение пропустить песню, которая только началась.

Для решения задачи воспользуемся логическим условием total\_play <= 10:



print(total\_play.loc[total\_play <= 10]) *# выведем все элементы, которые удовлетворяют условию*

print(total\_play.loc[total\_play <= 10].count()) *# посчитаем общее количество элементов, которые удовлетворяют условиям*

29160

Порог в 10 секунд мы выбрали произвольно. Но было бы интересно установить, существует ли на самом деле какое-нибудь пороговое время — длительность воспроизведения композиции, после которого пользователь чаще всего не пропускает трек, а слушает его до конца. Анализ данных позволяет ставить такие задачи и находить на них ответы.

Давайте вернёмся к войне между роком и попсой! Вам предлагается установить, сколько композиций этих жанров слушали не более 5 секунд и определить: зависит ли быстрота принятия решения о пропуске трека от жанра или на это влияют другие причины?