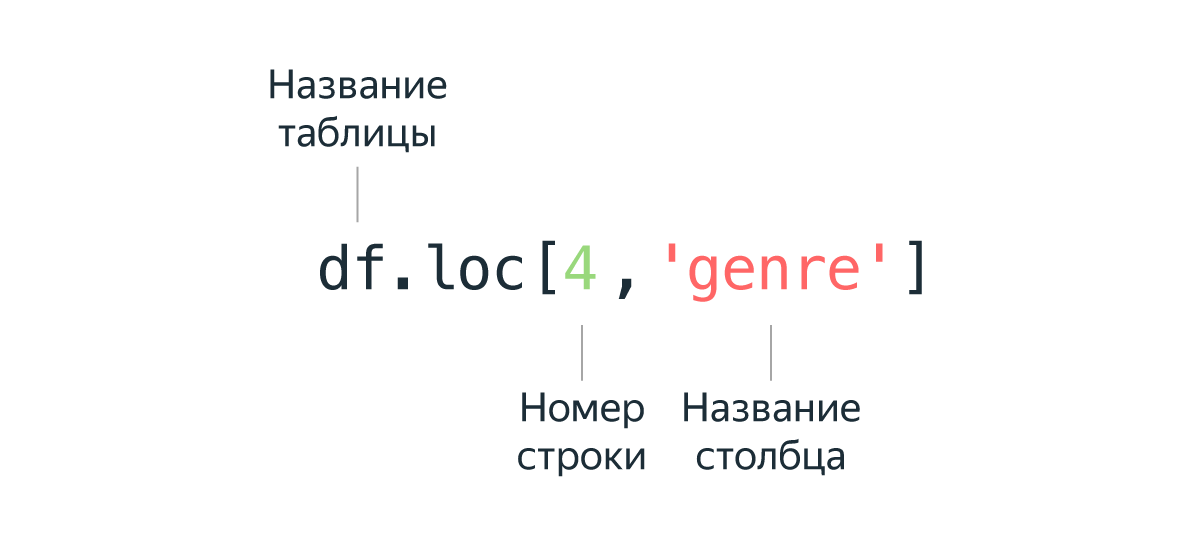
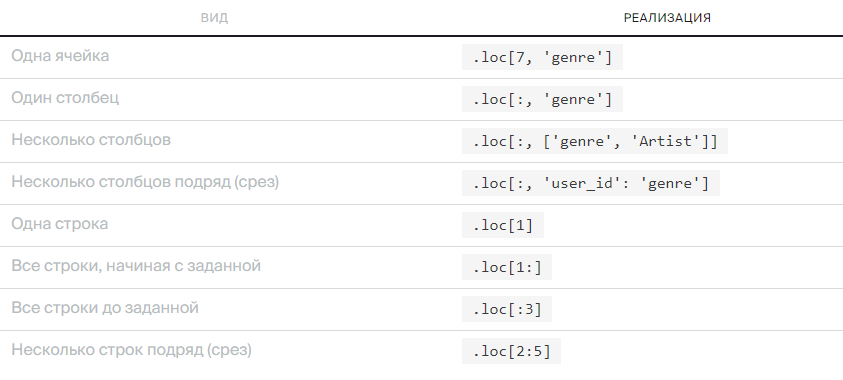
## 6. Индексация в DataFrame

Сейчас мы, наконец, попробуем получить данные, которые понадобятся непосредственно для выполнения задачи, поставленной менеджером Яндекс.Музыки. Для этого надо уметь запрашивать информацию из определённых ячеек таблицы.

К каждой ячейке с данными в DataFrame можно обратиться по её индексу и названию столбца. Мы можем получать различные срезы данных в зависимости от того, какой запрос к DataFrame мы сформулируем. Этот процесс называется **индексация**. Для DataFrame она проводится разными способами.

Атрибут **loc[строка, столбец]** (от англ. location, «положение») даёт доступ к элементу по строке и столбцу.





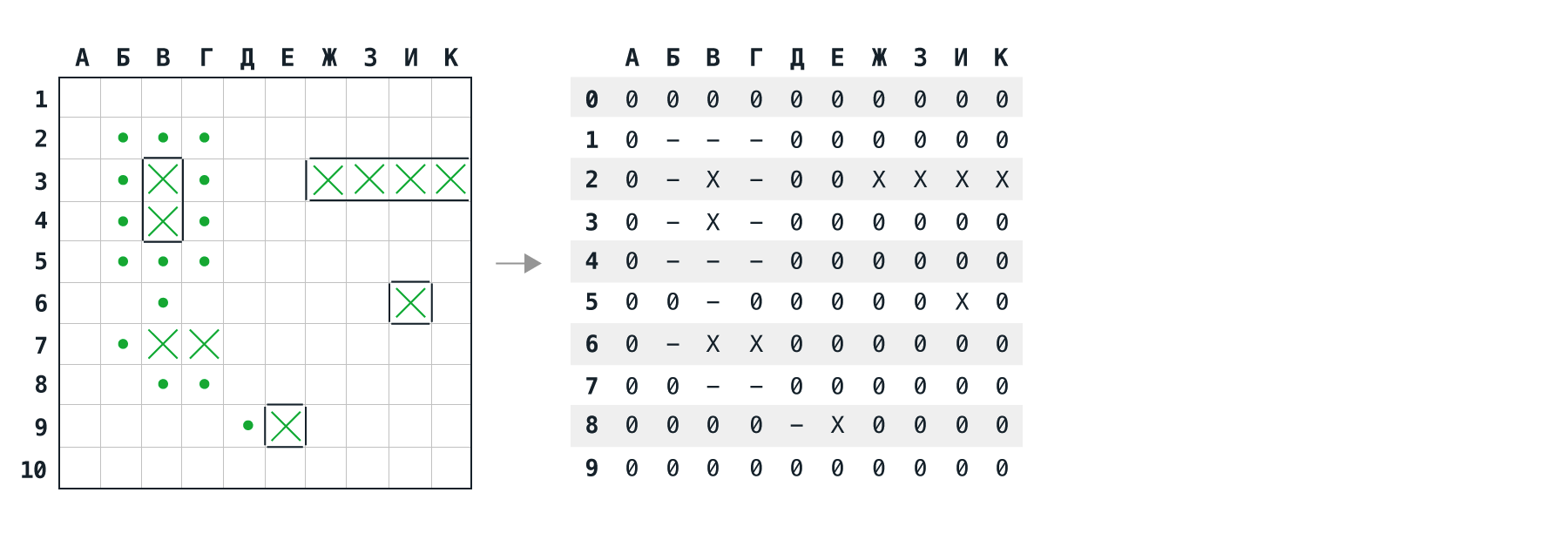
----- смотрим файл 05-06-01-loc\_attribute.gif --------------------

----- смотрим файл 05-06-02-loc\_stroki.gif --------------------

Также вы могли заметить, что запрос к атрибуту loc[] использует квадратные скобки, это напоминает списки в Python. Индексация здесь очень похожа на индексацию списков.

Вспомните, как вы играли в «Морской бой» на уроках. Для победы недостаточно просто стрелять куда попало — нужно обдумывать ситуацию на поле противника, чтобы бить в цель как можно точнее.

Посмотрите на поле для игры, оно подобно DataFrame: есть столбцы с буквенными обозначениями и ось индексов. Обратите внимание на разницу в индексах — в DataFrame они от 0 до 9.



import pandas as pd

data = [[0,0,0,0,0,0,0,0,0,0],

[0,'-','-','-',0,0,0,0,0,0],

[0,'-','X','-',0,0,'X','X','X','X'],

[0,'-','X','-',0,0,0,0,0,0],

[0,'-','-','-',0,0,0,0,0,0],

[0,0,'-',0,0,0,0,0,'X',0],

[0,'-','X','X',0,0,0,0,0,0],

[0,0,'-','-',0,0,0,0,0,0],

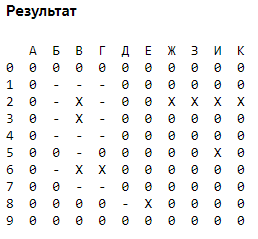
[0,0,0,0,'-','X',0,0,0,0],

[0,0,0,0,0,0,0,0,0,0]]

columns = ['А','Б','В','Г','Д','Е','Ж','З','И','К']

battle = pd.DataFrame(data = data, columns = columns)

print(battle)



Ячейки DataFrame со значением **0** обозначают пустую клетку. Информация в ячейке меняется на **X**, если игрок выстрелил и попал в корабль противника, а если не попал, то меняется на прочерк.

Давайте исследуем текущую ситуацию на поле. Просмотрим столбец 'В' и убедимся, что там уже началась атака на корабль:

print(battle.loc[:,'В'])

Действительно, атака уже началась в 6-ой строке, но корабль еще не убит, а выстрелы в ячейки сверху и снизу кончились промахом. Просмотрим 6-ую строку и узнаем расположение корабля.

print(battle.loc[6])

Корабль ориентирован по горизонтали, по нему произведено два удачных выстрела, которые пришлись на столбцы В и Г. Нужно оценить ситуацию вокруг корабля. Для этого просмотрим строки с 5 по 7.

Очевидно, что следующий выстрел нужно сделать по координате 6Д. Ура, корабль убит!

Теперь мы хотим найти второй трёхпалубный корабль. Оценим, где вероятнее всего он может располагаться. Для этого просмотрим две половины игрового поля. Срез из столбцов от А до Д даст возможность вывести левую часть таблицы:

print(battle.loc[:,'А':'Д'])

Такая же операция для столбцов от Е до К покажет правую часть таблицы:

print(battle.loc[:,'Е':'К'])

Важное замечание: когда мы используем срезы в списках, то конец среза не включается в результат. А вот атрибут .loc[] тем и выделяется, что включает и начало, и конец среза.

Например, есть список исполнителей:

artist = ['Marina Rei', 'Stive Morgan','Rixton','Henry Hall & His Gleneagles Hotel Band', 'Andrew Paul Woodworth', 'Pillar Point','Steve Campbell','David Civera','Lumipa Beats', 'Henning Wehland']

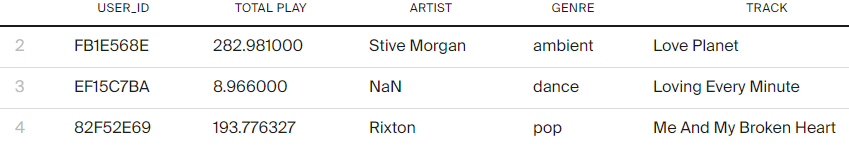
Элементы с 2 по 4 получают запросом:

print(artist[2:5])

['Rixton', 'Henry Hall & His Gleneagles Hotel Band', 'Andrew Paul Woodworth']

Последним в запросе указан индекс 5 — именно для того, чтобы в срез попал элемент с индексом 4. Запрос на получение со 2 по 4 строки в таблице будет выглядеть вот так:

print(df.loc[2:4])



Итак, вы видели, как запрашивать один столбец, одну строку, диапазон столбцов и диапазон строк. Это самые ходовые запросы, которые вам предстоит делать как аналитику данных.

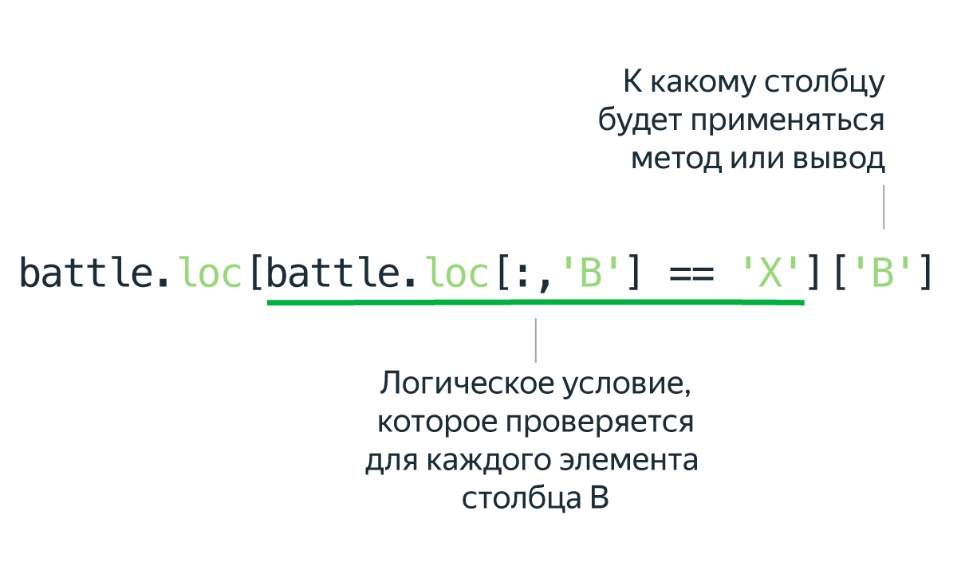
На практике чаще применяют сокращенную форму записи для индексации. Но возможности у неё ограничены. Имейте в виду, что она не всегда возвращает те же результаты, что атрибут .loc[] в его полном варианте.



Неприятельский трёхпалубный корабль, скорее всего, на правой половине поля. Это можно предположить, оглядев по очереди оба среза. Аналитики называют такой метод изучения «посмотреть на данные глазами».

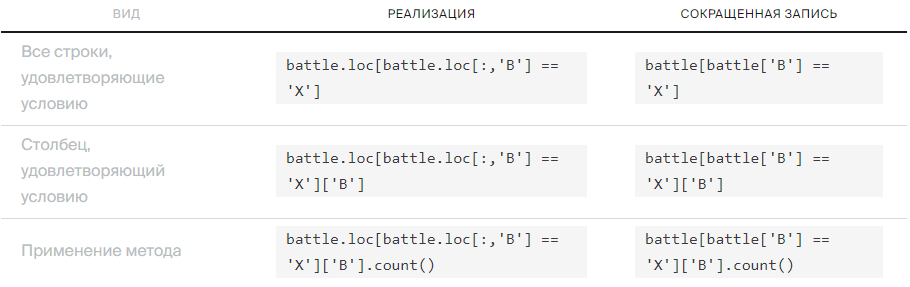
На глаз хорошо выбирать направление дальнейших поисков, но так не получишь точных цифр, которые можно включить в отчёт. Надо уметь подсчитать количество определённых значений, например, точных попаданий. В Pandas для это есть метод **count()** (англ. *count*, «сосчитать»).

Его вызывают и приказывают сосчитать, например, количество ячеек столбца В, где были попадания. Удачный выстрел — это значение "X" в ячейке. Для столбца В таблицы battle такие ячейки отвечают логическому условию battle.loc[:,'В'] == 'X'. Поскольку в указании, какие именно значения считать, нужен логический оператор, такой доступ к значению ячейки называют **логическая индексация**.



print(battle.loc[battle.loc[:,'В'] == 'X']['В'].count()) *# используем метод .count() для подсчета записей, удовлетворяющих условию в столбце В*

3



Конечно, писать вызов метода count() для подсчёта попаданий в «морском бою» то же, что стрелять из пушки по воробьям. Но в анализе таблиц на много тысяч строк счётный метод — мощное орудие. Попробуйте в задаче с данными Яндекс.Музыки определить, какой жанр оказался популярнее у пользователей: поп или рок?