# Python Break Statement

Break in Python terminates a loop completely when an external condition is given or not given. Python break is used within the code and is usually placed after an "if" statement.

A break statement can only be used inside a loop. This is because the purpose of a break statement is to stop a loop. You can use a break statement inside an if statement, but only if that if statement is inside a loop. The syntax of the break statement is: The syntax of the break statement is:

```
for val in sequence:

    if condition:
        break
```

```
while condition:

    if condition:
        break
```

# Python Continue Statement

The continue statement is used to skip the current iteration of the loop and the control flow of the program goes to the next iteration. For example-

```
for val in sequence:

    if condition:
        continue
```

```
while condition:

    if condition:
        continue
```

# All about Python List[]

- Ordered
- Changeable
- Allow Duplicates


L1 = [ 'data', 'science' ]
L2 = [ 1, 40, 300, 'shakil', True, False ]

**List Comprehension: Elegant Way to Create Lists**

List comprehension is an elegant and concise way to create a new list from an existing list in Python. A list comprehension consists of an expression followed by **for** statement inside square brackets.

**Syntax:**   newlist = [ Expression for item in iterable if condition == True ]
            newlist = [ x for x in items ]

# All about Python Tuples()

- Ordered
- Unchangeable
- Allow Duplicates

T1 = ('data', 'science')
T2 = (1, 40, 300, 'Shakil', True, False)

# All about Python *Set{}*

- Unordered
- Unchangeable
- Duplicates Not Allowed

S1 = { 'data', 'science' }
S2 = { 1, 40, 300, 'shakil', True, False }

# All about Python Array[]

- Ordered
- Duplicate Values Allowed
- Changeable

A[3,4,5]

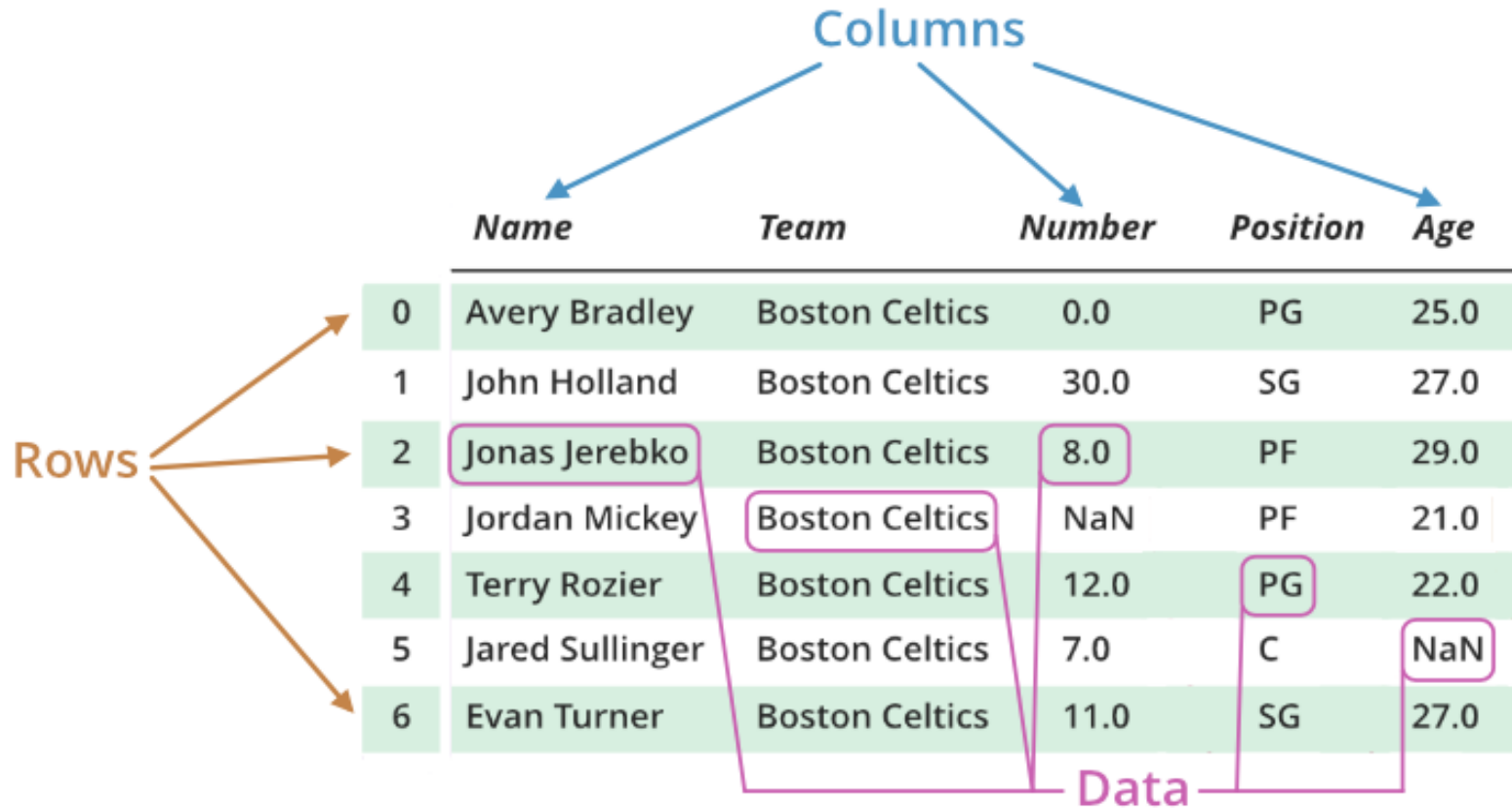| Type Code | C Type | Python Type | Minimum Size In Bytes |
|-----------|--------|-------------|------------------------|
| 'b' | signed cahar | int | 1 |
| 'B' | unsigned char | int | 1 |
| 'u' | Py_UNICODE | unicode character | 2 |
| 'h' | signed short | int | 2 |
| 'H' | unsigned short | int | 2 |
| 'i' | signed int | int | 2 |
| 'I' | unsigned int | int | 2 |
| 'l' | signed long | int | 4 |
| 'L' | unsigned long | int | 4 |
| 'q' | signed long long | int | 8 |
| 'Q' | unsigned long long | int | 8 |
| 'f' | float | float | 4 |
| 'd' | double | float | 8 |

# All about Python Dictionary{}

- Ordered ( Python 3.7+ )
- Changeable
- Does not Allow Duplicates

D1 = { "brand": "Apple", "model": "13 Pro Max", "year": 2022}

Here, ( brand, model, year = id or key ) & ( Apple, 13 pro max, 2022 = Data )

# All about Python Data Frame

# Python Resources:

Python Official Docs: [Link]
60 Days of Python: [Link]
Python with Problem-Solving: [Link]
Book: [Link]

Read our blogs: www.aiquest.org/blog