In [1]:
```
##########################################################
# DataFrames part II
# Author : Rodrique KAFANDO
# Destination : Master FD&IA - UV-BF
# Online  date : 23.07.2021
##########################################################
```

# Filtering data

In [1]:
```
import pandas as pd
```

In [3]:
```
df = pd.read_csv('./pandas/employees.csv')
df.head()
```

Out[3]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 8/6/1993 | 12:42 PM | 97308 | 6.945 | True | Marketing |
| **1** | Thomas | Male | 3/31/1996 | 6:53 AM | 61933 | 4.170 | True | NaN |
| **2** | Maria | Female | 4/23/1993 | 11:17 AM | 130590 | 11.858 | False | Finance |
| **3** | Jerry | Male | 3/4/2005 | 1:00 PM | 138705 | 9.340 | True | Finance |
| **4** | Larry | Male | 1/24/1998 | 4:47 PM | 101004 | 1.389 | True | Client Services |

In [5]:
```
# inspect data type
# what can we remark?
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   First Name       933 non-null    object 
 1   Gender           855 non-null    object 
 2   Start Date       1000 non-null   object 
 3   Last Login Time  1000 non-null   object 
 4   Salary           1000 non-null   int64  
 5   Bonus %          1000 non-null   float64
 6   Senior Management  933 non-null    object 
 7   Team             957 non-null    object 
dtypes: float64(1), int64(1), object(6)
memory usage: 62.6+ KB
```

In [9]:
```
## we can see that Start Date and Login Time are Objects type
# this may causes some date operation or math operation on these variables
# we first need to convert these columns into date type, by using
```

```
datetime() method
df['Start Date'] = pd.to_datetime(df['Start Date'])
```

In [10]:
```
df['Last Login Time'] = pd.to_datetime(df['Last Login Time'])
```

In [11]:
```
df.head(3)
```

Out[11]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| 0 | Douglas | Male | 1993-08-06 | 2021-04-30 12:42:00 | 97308 | 6.945 | True | Marketing |
| 1 | Thomas | Male | 1996-03-31 | 2021-04-30 06:53:00 | 61933 | 4.170 | True | NaN |
| 2 | Maria | Female | 1993-04-23 | 2021-04-30 11:17:00 | 130590 | 11.858 | False | Finance |

In [13]:
```
# let's convert senior management to boolean type
df['Senior Management'] = df['Senior Management'].astype('bool')
```

In [14]:
```
df.head(3)
```

Out[14]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| 0 | Douglas | Male | 1993-08-06 | 2021-04-30 12:42:00 | 97308 | 6.945 | True | Marketing |
| 1 | Thomas | Male | 1996-03-31 | 2021-04-30 06:53:00 | 61933 | 4.170 | True | NaN |
| 2 | Maria | Female | 1993-04-23 | 2021-04-30 11:17:00 | 130590 | 11.858 | False | Finance |

In [17]:
```
# Also, we can notice that Gender can be convert to 'category',
because we just have to value
df['Gender'] = df['Gender'].astype('category')
df['Gender']
```

Out[17]:
```
0        Male
1        Male
2      Female
3        Male
4        Male
       ...
995       NaN
996      Male
997      Male
998      Male
999      Male
Name: Gender, Length: 1000, dtype: category
Categories (2, object): ['Female', 'Male']
```

In [16]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   First Name         933 non-null    object
 1   Gender             855 non-null    category
 2   Start Date         1000 non-null   datetime64[ns]
 3   Last Login Time    1000 non-null   datetime64[ns]
 4   Salary             1000 non-null   int64
 5   Bonus %            1000 non-null   float64
 6   Senior Management  1000 non-null   bool
 7   Team               957 non-null    object
dtypes: bool(1), category(1), datetime64[ns](2), float64(1), int64(1), object
(2)
memory usage: 49.0+ KB
```

# Another way to convert date column while reading the .csv file

In [19]:

```
df = pd.read_csv('./pandas/employees.csv', parse_dates = ['Start
Date','Last Login Time'])
# df['Start Date'] = pd.to_datetime(df['Start Date'])
# df['Last Login Time'] = pd.to_datetime(df['Last Login Time'])
df['Senior Management'] = df['Senior Management'].astype('bool')
df.head()
```

Out[19]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| **1** | Thomas | Male | 1996-03-31 | 2021-05-01 06:53:00 | 61933 | 4.170 | True | NaN |
| **2** | Maria | Female | 1993-04-23 | 2021-05-01 11:17:00 | 130590 | 11.858 | False | Finance |
| **3** | Jerry | Male | 2005-03-04 | 2021-05-01 13:00:00 | 138705 | 9.340 | True | Finance |
| **4** | Larry | Male | 1998-01-24 | 2021-05-01 16:47:00 | 101004 | 1.389 | True | Client Services |

# Filter a DataFrame based on a condition

In [20]:

```
df = pd.read_csv('./pandas/employees.csv', parse_dates = ['Start
Date','Last Login Time'])
# df['Start Date'] = pd.to_datetime(df['Start Date'])
# df['Last Login Time'] = pd.to_datetime(df['Last Login Time'])
df['Senior Management'] = df['Senior Management'].astype('bool')
df['Gender'] = df['Gender'].astype('category')

df.head()
```

Out[20]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| **1** | Thomas | Male | 1996-03-31 | 2021-05-01 06:53:00 | 61933 | 4.170 | True | NaN |
| **2** | Maria | Female | 1993-04-23 | 2021-05-01 11:17:00 | 130590 | 11.858 | False | Finance |
| **3** | Jerry | Male | 2005-03-04 | 2021-05-01 13:00:00 | 138705 | 9.340 | True | Finance |
| **4** | Larry | Male | 1998-01-24 | 2021-05-01 16:47:00 | 101004 | 1.389 | True | Client Services |

In [21]:

```
# broadcasting operation
# let's extract all Gender value that are Male
df['Gender'] == 'Male' # les valeurs TRue sont celles qui satifassent la condition
```

Out[21]:

```
0       True
1       True
2      False
3       True
4       True
       ...
995    False
996     True
997     True
998     True
999     True
Name: Gender, Length: 1000, dtype: bool
```

In [ ]:

```

```

In [30]:

```
# en mettant cette valeur comme parametre de df, nous aurons uniquement les valeurs Male
df[df['Gender'] == 'Male']
```

Out[30]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| **1** | Thomas | Male | 1996-03-31 | 2021-05-01 06:53:00 | 61933 | 4.170 | True | NaN |
| **3** | Jerry | Male | 2005-03-04 | 2021-05-01 13:00:00 | 138705 | 9.340 | True | Finance |
| **4** | Larry | Male | 1998-01-24 | 2021-05-01 16:47:00 | 101004 | 1.389 | True | Client Services |
| **5** | Dennis | Male | 1987-04-18 | 2021-05-01 01:35:00 | 115163 | 10.125 | False | Legal |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **994** | George | Male | 2013-06-21 | 2021-05-01 17:47:00 | 98874 | 4.479 | True | Marketing |

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **996** | Phillip | Male | 1984-01-31 | 2021-05-01 06:30:00 | 42392 | 19.675 | False | Finance |
| **997** | Russell | Male | 2013-05-20 | 2021-05-01 12:39:00 | 96914 | 1.421 | False | Product |
| **998** | Larry | Male | 2013-04-20 | 2021-05-01 16:45:00 | 60500 | 11.985 | False | Business Development |
| **999** | Albert | Male | 2012-05-15 | 2021-05-01 18:24:00 | 129949 | 10.169 | True | Sales |

424 rows × 8 columns

In [31]:
```
# de la même façon, nous pouvons extraire Finance de Team
df[df['Team'] == 'Finance']
```

Out[31]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **2** | Maria | Female | 1993-04-23 | 2021-05-01 11:17:00 | 130590 | 11.858 | False | Finance |
| **3** | Jerry | Male | 2005-03-04 | 2021-05-01 13:00:00 | 138705 | 9.340 | True | Finance |
| **7** | NaN | Female | 2015-07-20 | 2021-05-01 10:43:00 | 45906 | 11.598 | True | Finance |
| **14** | Kimberly | Female | 1999-01-14 | 2021-05-01 07:13:00 | 41426 | 14.543 | True | Finance |
| **46** | Bruce | Male | 2009-11-28 | 2021-05-01 22:47:00 | 114796 | 6.796 | False | Finance |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **907** | Elizabeth | Female | 1998-07-27 | 2021-05-01 11:12:00 | 137144 | 10.081 | False | Finance |
| **954** | Joe | Male | 1980-01-19 | 2021-05-01 16:06:00 | 119667 | 1.148 | True | Finance |
| **987** | Gloria | Female | 2014-12-08 | 2021-05-01 05:08:00 | 136709 | 10.331 | True | Finance |
| **992** | Anthony | Male | 2011-10-16 | 2021-05-01 08:35:00 | 112769 | 11.625 | True | Finance |
| **996** | Phillip | Male | 1984-01-31 | 2021-05-01 06:30:00 | 42392 | 19.675 | False | Finance |

102 rows × 8 columns

In [32]:
```
mask = df['Team'] == 'Finance'
df[mask]
```

Out[32]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **2** | Maria | Female | 1993-04-23 | 2021-05-01 11:17:00 | 130590 | 11.858 | False | Finance |

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **3** | Jerry | Male | 2005-03-04 | 2021-05-01 13:00:00 | 138705 | 9.340 | True | Finance |
| **7** | NaN | Female | 2015-07-20 | 2021-05-01 10:43:00 | 45906 | 11.598 | True | Finance |
| **14** | Kimberly | Female | 1999-01-14 | 2021-05-01 07:13:00 | 41426 | 14.543 | True | Finance |
| **46** | Bruce | Male | 2009-11-28 | 2021-05-01 22:47:00 | 114796 | 6.796 | False | Finance |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **907** | Elizabeth | Female | 1998-07-27 | 2021-05-01 11:12:00 | 137144 | 10.081 | False | Finance |
| **954** | Joe | Male | 1980-01-19 | 2021-05-01 16:06:00 | 119667 | 1.148 | True | Finance |
| **987** | Gloria | Female | 2014-12-08 | 2021-05-01 05:08:00 | 136709 | 10.331 | True | Finance |
| **992** | Anthony | Male | 2011-10-16 | 2021-05-01 08:35:00 | 112769 | 11.625 | True | Finance |
| **996** | Phillip | Male | 1984-01-31 | 2021-05-01 06:30:00 | 42392 | 19.675 | False | Finance |

102 rows × 8 columns

In [36]:
```
# for boolean values, you can do it with the val or not
# 1
df[df['Senior Management']==True]
```

Out[36]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| **1** | Thomas | Male | 1996-03-31 | 2021-05-01 06:53:00 | 61933 | 4.170 | True | NaN |
| **3** | Jerry | Male | 2005-03-04 | 2021-05-01 13:00:00 | 138705 | 9.340 | True | Finance |
| **4** | Larry | Male | 1998-01-24 | 2021-05-01 16:47:00 | 101004 | 1.389 | True | Client Services |
| **6** | Ruby | Female | 1987-08-17 | 2021-05-01 16:20:00 | 65476 | 10.012 | True | Product |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **991** | Rose | Female | 2002-08-25 | 2021-05-01 05:12:00 | 134505 | 11.051 | True | Marketing |
| **992** | Anthony | Male | 2011-10-16 | 2021-05-01 08:35:00 | 112769 | 11.625 | True | Finance |
| **993** | Tina | Female | 1997-05-15 | 2021-05-01 15:53:00 | 56450 | 19.040 | True | Engineering |
| **994** | George | Male | 2013-06-21 | 2021-05-01 17:47:00 | 98874 | 4.479 | True | Marketing |

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **999** | Albert | Male | 2012-05-15 | 2021-05-01 18:24:00 | 129949 | 10.169 | True | Sales |

535 rows × 8 columns

In [37]:
```
# 2
df[df['Senior Management']]
```

Out[37]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| **1** | Thomas | Male | 1996-03-31 | 2021-05-01 06:53:00 | 61933 | 4.170 | True | NaN |
| **3** | Jerry | Male | 2005-03-04 | 2021-05-01 13:00:00 | 138705 | 9.340 | True | Finance |
| **4** | Larry | Male | 1998-01-24 | 2021-05-01 16:47:00 | 101004 | 1.389 | True | Client Services |
| **6** | Ruby | Female | 1987-08-17 | 2021-05-01 16:20:00 | 65476 | 10.012 | True | Product |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **991** | Rose | Female | 2002-08-25 | 2021-05-01 05:12:00 | 134505 | 11.051 | True | Marketing |
| **992** | Anthony | Male | 2011-10-16 | 2021-05-01 08:35:00 | 112769 | 11.625 | True | Finance |
| **993** | Tina | Female | 1997-05-15 | 2021-05-01 15:53:00 | 56450 | 19.040 | True | Engineering |
| **994** | George | Male | 2013-06-21 | 2021-05-01 17:47:00 | 98874 | 4.479 | True | Marketing |
| **999** | Albert | Male | 2012-05-15 | 2021-05-01 18:24:00 | 129949 | 10.169 | True | Sales |

535 rows × 8 columns

In [39]:
```
# let's check now for != operation
mask = df['Team'] != 'Finance'
df[mask]
```

Out[39]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| **1** | Thomas | Male | 1996-03-31 | 2021-05-01 06:53:00 | 61933 | 4.170 | True | NaN |
| **4** | Larry | Male | 1998-01-24 | 2021-05-01 16:47:00 | 101004 | 1.389 | True | Client Services |
| **5** | Dennis | Male | 1987-04-18 | 2021-05-01 01:35:00 | 115163 | 10.125 | False | Legal |

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **6** | Ruby | Female | 1987-08-17 | 2021-05-01 16:20:00 | 65476 | 10.012 | True | Product |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **994** | George | Male | 2013-06-21 | 2021-05-01 17:47:00 | 98874 | 4.479 | True | Marketing |
| **995** | Henry | NaN | 2014-11-23 | 2021-05-01 06:09:00 | 132483 | 16.655 | False | Distribution |
| **997** | Russell | Male | 2013-05-20 | 2021-05-01 12:39:00 | 96914 | 1.421 | False | Product |
| **998** | Larry | Male | 2013-04-20 | 2021-05-01 16:45:00 | 60500 | 11.985 | False | Business Development |
| **999** | Albert | Male | 2012-05-15 | 2021-05-01 18:24:00 | 129949 | 10.169 | True | Sales |

898 rows × 8 columns

In [42]:
```
# >, <, etc.
df['Salary'] > 100000
```

Out[42]:
```
0      False
1      False
2       True
3       True
4       True
       ...
995     True
996    False
997    False
998    False
999     True
Name: Salary, Length: 1000, dtype: bool
```

In [43]:
```
# same for date type
df['Start Date'] > '2002-08-25'
```

Out[43]:
```
0      False
1      False
2      False
3       True
4      False
       ...
995     True
996    False
997     True
998     True
999     True
Name: Start Date, Length: 1000, dtype: bool
```

# Filter with more than one condition (AND)

In [44]:
```
df = pd.read_csv('./pandas/employees.csv', parse_dates = ['Start Date','Last Login Time'])
# df['Start Date'] = pd.to_datetime(df['Start Date'])
```

```
# df['Last Login Time'] = pd.to_datetime(df['Last Login Time'])
df['Senior Management'] = df['Senior Management'].astype('bool')
df['Gender'] = df['Gender'].astype('category')

df.head(2)
```

Out[44]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| **1** | Thomas | Male | 1996-03-31 | 2021-05-01 06:53:00 | 61933 | 4.170 | True | NaN |

In [52]:

```
# let's extract those, that Gender == Male and Team == Marketing
tmp1 = df['Gender'] == 'Male'
tmp2 = df['Team'] == 'Marketing'
tmp3 = df['Senior Management']
df[tmp1 & tmp2 & tmp3].head()
```

Out[52]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| **26** | Craig | Male | 2000-02-27 | 2021-05-01 07:45:00 | 37598 | 7.757 | True | Marketing |
| **77** | Charles | Male | 2004-09-14 | 2021-05-01 20:13:00 | 107391 | 1.260 | True | Marketing |
| **101** | Aaron | Male | 2012-02-17 | 2021-05-01 10:20:00 | 61602 | 11.849 | True | Marketing |
| **204** | Willie | Male | 2006-06-06 | 2021-05-01 09:45:00 | 55281 | 4.935 | True | Marketing |

In [50]:

```
# be careful, df[df['Gender'] == 'Male' & df['Team'] ==
'Marketing'] will not work
```

# Filter with more than one condition (OR -|)

In [53]:

```
df = pd.read_csv('./pandas/employees.csv', parse_dates = ['Start
Date','Last Login Time'])
# df['Start Date'] = pd.to_datetime(df['Start Date'])
# df['Last Login Time'] = pd.to_datetime(df['Last Login Time'])
df['Senior Management'] = df['Senior Management'].astype('bool')
df['Gender'] = df['Gender'].astype('category')

df.head(2)
```

Out[53]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| 0 | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| 1 | Thomas | Male | 1996-03-31 | 2021-05-01 06:53:00 | 61933 | 4.170 | True | NaN |

In [55]:

```
# en utilisant |, il suffit qu'une des conditions soit valable
tmp1 = df['Senior Management']
tmp2 = df['Start Date'] > '1990-01-01'
```

In [58]:

```
df[tmp1 | tmp2].head(10) # il suffit qu'une des conditions soit vraie
```

Out[58]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| 0 | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| 1 | Thomas | Male | 1996-03-31 | 2021-05-01 06:53:00 | 61933 | 4.170 | True | NaN |
| 2 | Maria | Female | 1993-04-23 | 2021-05-01 11:17:00 | 130590 | 11.858 | False | Finance |
| 3 | Jerry | Male | 2005-03-04 | 2021-05-01 13:00:00 | 138705 | 9.340 | True | Finance |
| 4 | Larry | Male | 1998-01-24 | 2021-05-01 16:47:00 | 101004 | 1.389 | True | Client Services |
| 6 | Ruby | Female | 1987-08-17 | 2021-05-01 16:20:00 | 65476 | 10.012 | True | Product |
| 7 | NaN | Female | 2015-07-20 | 2021-05-01 10:43:00 | 45906 | 11.598 | True | Finance |
| 8 | Angela | Female | 2005-11-22 | 2021-05-01 06:29:00 | 95570 | 18.523 | True | Engineering |
| 9 | Frances | Female | 2002-08-08 | 2021-05-01 06:51:00 | 139852 | 7.524 | True | Business Development |
| 10 | Louise | Female | 1980-08-12 | 2021-05-01 09:01:00 | 63241 | 15.132 | True | NaN |

In [67]:

```
# maintenant, essayons de combiner les deux conditions à la fois ----- un peu plus complex----
tmp1 = df['First Name'] == 'Robert'
tmp2 = df['Team'] == 'Client Services'
tmp3 = df['Start Date'] > '1992-01-01'
```

In [72]:

```
# nous voulons recuperer tmp1 mais avec tmp2 ou tmp3
# devons imperativement specifier la premiere operation à evaluer pour eviter toute ambiguité
```

```
df[tmp1 & (tmp2 | tmp3)]
```

Out[72]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **335** | Robert | Male | 2014-11-18 | 2021-05-01 05:00:00 | 85799 | 19.930 | False | Finance |
| **387** | Robert | Male | 1994-10-29 | 2021-05-01 04:26:00 | 123294 | 19.894 | False | Client Services |
| **488** | Robert | Male | 2007-03-11 | 2021-05-01 11:20:00 | 135882 | 19.944 | False | Legal |
| **825** | Robert | NaN | 2000-12-04 | 2021-05-01 01:20:00 | 69267 | 5.890 | True | Sales |
| **880** | Robert | NaN | 2007-05-25 | 2021-05-01 03:17:00 | 90998 | 8.382 | False | Finance |

# The .isin() Method

- très utile lorsque vous souhaitez filtrer plusieurs valeurs à la fois dans une series, sans passer par une multitude de filtre bolean

In [73]:

```
df = pd.read_csv('./pandas/employees.csv', parse_dates = ['Start Date','Last Login Time'])
df['Senior Management'] = df['Senior Management'].astype('bool')
df['Gender'] = df['Gender'].astype('category')

df.head(2)
```

Out[73]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| **1** | Thomas | Male | 1996-03-31 | 2021-05-01 06:53:00 | 61933 | 4.170 | True | NaN |

In [74]:

```
# verifions avec la colonne Team
# disons que nous souhaitons extraire les lignes dont les valeurs de la colonne Team est égale à Legal, Marketing, Product
# jusque là, nous pouvons faire ceci
tmp1 = df['Team'] == 'Legal'
tmp2 = df['Team'] == 'Marketing'
tmp3 = df['Team'] == 'Product'
df[tmp1 | tmp2 | tmp3]
```

Out[74]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| **5** | Dennis | Male | 1987-04-18 | 2021-05-01 01:35:00 | 115163 | 10.125 | False | Legal |
| **6** | Ruby | Female | 1987-08-17 | 2021-05-01 16:20:00 | 65476 | 10.012 | True | Product |
| **11** | Julie | Female | 1997-10-26 | 2021-05-01 15:19:00 | 102508 | 12.637 | True | Legal |
| **15** | Lillian | Female | 2016-06-05 | 2021-05-01 06:09:00 | 59414 | 1.256 | False | Product |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **986** | Donna | Female | 1982-11-26 | 2021-05-01 07:04:00 | 82871 | 17.999 | False | Marketing |
| **989** | Justin | NaN | 1991-02-10 | 2021-05-01 16:58:00 | 38344 | 3.794 | False | Legal |
| **991** | Rose | Female | 2002-08-25 | 2021-05-01 05:12:00 | 134505 | 11.051 | True | Marketing |
| **994** | George | Male | 2013-06-21 | 2021-05-01 17:47:00 | 98874 | 4.479 | True | Marketing |
| **997** | Russell | Male | 2013-05-20 | 2021-05-01 12:39:00 | 96914 | 1.421 | False | Product |

281 rows × 8 columns

In [77]:
```
# Utilisons maintenant la method .isin()
filter_val = df['Team'].isin(['Legal','Marketing','Product'])
```

In [78]:
```
df[filter_val]
```

Out[78]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| **5** | Dennis | Male | 1987-04-18 | 2021-05-01 01:35:00 | 115163 | 10.125 | False | Legal |
| **6** | Ruby | Female | 1987-08-17 | 2021-05-01 16:20:00 | 65476 | 10.012 | True | Product |
| **11** | Julie | Female | 1997-10-26 | 2021-05-01 15:19:00 | 102508 | 12.637 | True | Legal |
| **15** | Lillian | Female | 2016-06-05 | 2021-05-01 06:09:00 | 59414 | 1.256 | False | Product |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **986** | Donna | Female | 1982-11-26 | 2021-05-01 07:04:00 | 82871 | 17.999 | False | Marketing |
| **989** | Justin | NaN | 1991-02-10 | 2021-05-01 16:58:00 | 38344 | 3.794 | False | Legal |
| **991** | Rose | Female | 2002-08-25 | 2021-05-01 05:12:00 | 134505 | 11.051 | True | Marketing |

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **994** | George | Male | 2013-06-21 | 2021-05-01 17:47:00 | 98874 | 4.479 | True | Marketing |
| **997** | Russell | Male | 2013-05-20 | 2021-05-01 12:39:00 | 96914 | 1.421 | False | Product |

281 rows × 8 columns

In [82]:
```
df[df.isin(['Male', 'Legal'])]
```

Out[82]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | NaN | Male | NaT | NaT | NaN | NaN | NaN | NaN |
| **1** | NaN | Male | NaT | NaT | NaN | NaN | NaN | NaN |
| **2** | NaN | NaN | NaT | NaT | NaN | NaN | NaN | NaN |
| **3** | NaN | Male | NaT | NaT | NaN | NaN | NaN | NaN |
| **4** | NaN | Male | NaT | NaT | NaN | NaN | NaN | NaN |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **995** | NaN | NaN | NaT | NaT | NaN | NaN | NaN | NaN |
| **996** | NaN | Male | NaT | NaT | NaN | NaN | NaN | NaN |
| **997** | NaN | Male | NaT | NaT | NaN | NaN | NaN | NaN |
| **998** | NaN | Male | NaT | NaT | NaN | NaN | NaN | NaN |
| **999** | NaN | Male | NaT | NaT | NaN | NaN | NaN | NaN |

1000 rows × 8 columns

# The .isnull() and .notnull() methods

In [85]:
```
# return les valeurs nulles d'une colonne
df[df['Team'].isnull()]
```

Out[85]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **1** | Thomas | Male | 1996-03-31 | 2021-05-01 06:53:00 | 61933 | 4.170 | True | NaN |
| **10** | Louise | Female | 1980-08-12 | 2021-05-01 09:01:00 | 63241 | 15.132 | True | NaN |
| **23** | NaN | Male | 2012-06-14 | 2021-05-01 16:19:00 | 125792 | 5.042 | True | NaN |
| **32** | NaN | Male | 1998-08-21 | 2021-05-01 14:27:00 | 122340 | 6.417 | True | NaN |
| **91** | James | NaN | 2005-01-26 | 2021-05-01 23:00:00 | 128771 | 8.309 | False | NaN |
| **109** | Christopher | Male | 2000-04-22 | 2021-05-01 10:15:00 | 37919 | 11.449 | False | NaN |

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **139** | NaN | Female | 1990-10-03 | 2021-05-01 01:08:00 | 132373 | 10.527 | True | NaN |
| **199** | Jonathan | Male | 2009-07-17 | 2021-05-01 08:15:00 | 130581 | 16.736 | True | NaN |
| **258** | Michael | Male | 2002-01-24 | 2021-05-01 03:04:00 | 43586 | 12.659 | False | NaN |
| **290** | Jeremy | Male | 1988-06-14 | 2021-05-01 18:20:00 | 129460 | 13.657 | True | NaN |
| **314** | Bobby | Male | 1996-03-31 | 2021-05-01 17:40:00 | 112117 | 6.338 | False | NaN |
| **367** | Edward | Male | 1989-08-04 | 2021-05-01 06:06:00 | 66067 | 10.957 | True | NaN |
| **382** | NaN | Female | 1996-04-18 | 2021-05-01 15:57:00 | 107024 | 12.182 | True | NaN |
| **434** | Joyce | Female | 1995-02-07 | 2021-05-01 07:38:00 | 50701 | 14.227 | True | NaN |
| **438** | Jason | Male | 1998-11-20 | 2021-05-01 14:54:00 | 69244 | 6.220 | True | NaN |
| **445** | Chris | Male | 2006-12-12 | 2021-05-01 01:57:00 | 71642 | 1.496 | False | NaN |
| **479** | Richard | Male | 1997-07-04 | 2021-05-01 11:47:00 | 47647 | 18.787 | True | NaN |
| **512** | Wanda | Female | 1993-04-06 | 2021-05-01 03:11:00 | 78883 | 19.695 | False | NaN |
| **513** | Jimmy | Male | 2013-11-19 | 2021-05-01 19:29:00 | 63549 | 19.624 | False | NaN |
| **520** | Peter | Male | 2003-02-22 | 2021-05-01 09:09:00 | 56580 | 8.411 | True | NaN |
| **567** | NaN | Female | 1980-04-01 | 2021-05-01 20:04:00 | 48141 | 12.605 | True | NaN |
| **573** | Kimberly | Female | 1981-12-30 | 2021-05-01 04:51:00 | 81800 | 5.435 | True | NaN |
| **580** | Harry | Male | 1985-01-27 | 2021-05-01 20:18:00 | 65482 | 18.089 | False | NaN |
| **626** | NaN | Female | 1997-04-13 | 2021-05-01 08:03:00 | 131755 | 2.930 | True | NaN |
| **634** | Carl | Male | 1987-03-30 | 2021-05-01 17:59:00 | 75598 | 19.289 | False | NaN |
| **635** | Randy | Male | 2000-09-27 | 2021-05-01 03:04:00 | 89831 | 13.047 | True | NaN |
| **647** | Donald | Male | 1988-04-06 | 2021-05-01 10:00:00 | 122920 | 5.320 | False | NaN |
| **669** | Joseph | NaN | 1982-03-28 | 2021-05-01 13:05:00 | 86564 | 11.879 | True | NaN |
| **684** | Alice | Female | 2016-01-21 | 2021-05-01 17:07:00 | 117787 | 10.485 | False | NaN |
| **706** | Todd | Male | 1993-07-04 | 2021-05-01 18:53:00 | 128175 | 18.473 | True | NaN |
| **726** | Daniel | Male | 2016-02-29 | 2021-05-01 04:04:00 | 77287 | 13.000 | True | NaN |

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **753** | Antonio | Male | 1999-06-06 | 2021-05-01 22:54:00 | 41928 | 5.478 | True | NaN |
| **774** | NaN | Female | 2000-06-18 | 2021-05-01 07:36:00 | 106428 | 10.867 | True | NaN |
| **781** | Lawrence | Male | 1995-07-03 | 2021-05-01 22:55:00 | 46378 | 9.127 | False | NaN |
| **794** | Nicole | Female | 2004-03-01 | 2021-05-01 17:17:00 | 44021 | 10.286 | False | NaN |
| **826** | NaN | NaN | 1988-08-01 | 2021-05-01 01:35:00 | 87103 | 5.665 | True | NaN |
| **850** | Charles | Male | 1997-09-03 | 2021-05-01 10:04:00 | 148291 | 6.002 | False | NaN |
| **851** | Bobby | Male | 1996-08-19 | 2021-05-01 01:16:00 | 147842 | 16.158 | True | NaN |
| **853** | Mildred | Female | 2007-04-06 | 2021-05-01 22:06:00 | 139284 | 11.390 | True | NaN |
| **855** | Phillip | NaN | 2003-10-20 | 2021-05-01 11:09:00 | 89700 | 2.277 | True | NaN |
| **864** | Ryan | Male | 2012-11-16 | 2021-05-01 13:47:00 | 57292 | 6.010 | False | NaN |
| **912** | Joe | Male | 1998-12-08 | 2021-05-01 10:28:00 | 126120 | 1.020 | False | NaN |
| **951** | NaN | Female | 2010-09-14 | 2021-05-01 05:19:00 | 143638 | 9.662 | True | NaN |

In [86]:

```
# return les valeurs non-nulles d'une colonne
df[df['Team'].notnull()]
```

Out[86]:

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| **2** | Maria | Female | 1993-04-23 | 2021-05-01 11:17:00 | 130590 | 11.858 | False | Finance |
| **3** | Jerry | Male | 2005-03-04 | 2021-05-01 13:00:00 | 138705 | 9.340 | True | Finance |
| **4** | Larry | Male | 1998-01-24 | 2021-05-01 16:47:00 | 101004 | 1.389 | True | Client Services |
| **5** | Dennis | Male | 1987-04-18 | 2021-05-01 01:35:00 | 115163 | 10.125 | False | Legal |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **995** | Henry | NaN | 2014-11-23 | 2021-05-01 06:09:00 | 132483 | 16.655 | False | Distribution |
| **996** | Phillip | Male | 1984-01-31 | 2021-05-01 06:30:00 | 42392 | 19.675 | False | Finance |
| **997** | Russell | Male | 2013-05-20 | 2021-05-01 12:39:00 | 96914 | 1.421 | False | Product |
| **998** | Larry | Male | 2013-04-20 | 2021-05-01 16:45:00 | 60500 | 11.985 | False | Business Development |

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **999** | Albert | Male | 2012-05-15 | 2021-05-01 18:24:00 | 129949 | 10.169 | True | Sales |

957 rows × 8 columns

# The .between Method

- permet de retouner des valeurs entre un interval donné

In [116…
```
# utilisons la colonne salaire pour illustrer cela
# include mix and max values
border = df[df['Salary'].between(145146,146908)]
```

In [117…
```
border
```

Out[117…

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **44** | Cynthia | Female | 1988-11-16 | 2021-05-01 18:54:00 | 145146 | 7.482 | True | Product |
| **132** | Carlos | Male | 1995-01-04 | 2021-05-01 07:02:00 | 146670 | 10.763 | False | Human Resources |
| **142** | Elizabeth | Female | 2003-10-09 | 2021-05-01 17:53:00 | 146129 | 5.687 | False | Finance |
| **175** | Willie | Male | 1998-02-17 | 2021-05-01 20:20:00 | 146651 | 1.451 | True | Engineering |
| **269** | NaN | Female | 1992-08-02 | 2021-05-01 20:35:00 | 145316 | 18.517 | True | Human Resources |
| **319** | Jacqueline | Female | 1981-11-25 | 2021-05-01 15:01:00 | 145988 | 18.243 | False | Marketing |
| **452** | Scott | Male | 2012-11-17 | 2021-05-01 14:47:00 | 146812 | 1.965 | True | Marketing |
| **536** | Clarence | Male | 1982-08-26 | 2021-05-01 09:47:00 | 146589 | 4.905 | True | Business Development |
| **665** | Anthony | Male | 2013-02-13 | 2021-05-01 13:35:00 | 146141 | 3.645 | True | Distribution |
| **720** | Marie | Female | 1983-04-08 | 2021-05-01 14:01:00 | 145988 | 18.685 | True | Human Resources |
| **750** | Louis | NaN | 1983-02-05 | 2021-05-01 18:39:00 | 145274 | 16.379 | False | Product |
| **808** | Julie | Female | 1980-03-08 | 2021-05-01 05:13:00 | 145357 | 3.459 | False | Engineering |
| **890** | NaN | Male | 2015-11-24 | 2021-05-01 03:11:00 | 145329 | 7.100 | True | Finance |
| **983** | John | Male | 1982-12-23 | 2021-05-01 22:35:00 | 146907 | 11.738 | False | Engineering |

In [119…
```
# same for date and time
```

```
df['Last Login Time'].between('06:00AM', '06:00PM')
df[df['Last Login Time'].between('06:00AM', '06:00PM')]
```

Out[119…

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **0** | Douglas | Male | 1993-08-06 | 2021-05-01 12:42:00 | 97308 | 6.945 | True | Marketing |
| **1** | Thomas | Male | 1996-03-31 | 2021-05-01 06:53:00 | 61933 | 4.170 | True | NaN |
| **2** | Maria | Female | 1993-04-23 | 2021-05-01 11:17:00 | 130590 | 11.858 | False | Finance |
| **3** | Jerry | Male | 2005-03-04 | 2021-05-01 13:00:00 | 138705 | 9.340 | True | Finance |
| **4** | Larry | Male | 1998-01-24 | 2021-05-01 16:47:00 | 101004 | 1.389 | True | Client Services |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **994** | George | Male | 2013-06-21 | 2021-05-01 17:47:00 | 98874 | 4.479 | True | Marketing |
| **995** | Henry | NaN | 2014-11-23 | 2021-05-01 06:09:00 | 132483 | 16.655 | False | Distribution |
| **996** | Phillip | Male | 1984-01-31 | 2021-05-01 06:30:00 | 42392 | 19.675 | False | Finance |
| **997** | Russell | Male | 2013-05-20 | 2021-05-01 12:39:00 | 96914 | 1.421 | False | Product |
| **998** | Larry | Male | 2013-04-20 | 2021-05-01 16:45:00 | 60500 | 11.985 | False | Business Development |

507 rows × 8 columns

# The .duplicated() method

In [143…

```
df = pd.read_csv('./pandas/employees.csv', parse_dates = ['Start Date','Last Login Time'])
df['Senior Management'] = df['Senior Management'].astype('bool')
df['Gender'] = df['Gender'].astype('category')
df.sort_values('First Name',inplace = True)
df.head(2)
```

Out[143…

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **101** | Aaron | Male | 2012-02-17 | 2021-05-02 10:20:00 | 61602 | 11.849 | True | Marketing |
| **327** | Aaron | Male | 1994-01-29 | 2021-05-02 18:48:00 | 58755 | 5.097 | True | Marketing |

In [126…

```
df['First Name']
```

Out[126…

```
101    Aaron
327    Aaron
```

```
440    Aaron
937    Aaron
137     Adam
       ...
902      NaN
925      NaN
946      NaN
947      NaN
951      NaN
Name: First Name, Length: 1000, dtype: object
```

In [136…

```
# check First Name column duplicated values
# keep considere par defaut la 1ere valeur trouvée comme étant pas dupliquée
# keep = first, last or false (will considere all value as duplicated, use ~ to get non-duplicated)

x = ~df['First Name'].duplicated(keep = False)
```

In [138…

```
df[x].head()
```

Out[138…

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **8** | Angela | Female | 2005-11-22 | 2021-05-01 06:29:00 | 95570 | 18.523 | True | Engineering |
| **688** | Brian | Male | 2007-04-07 | 2021-05-01 22:47:00 | 93901 | 17.821 | True | Legal |
| **190** | Carol | Female | 1996-03-19 | 2021-05-01 03:39:00 | 57783 | 9.129 | False | Finance |
| **887** | David | Male | 2009-12-05 | 2021-05-01 08:48:00 | 92242 | 15.407 | False | Legal |
| **5** | Dennis | Male | 1987-04-18 | 2021-05-01 01:35:00 | 115163 | 10.125 | False | Legal |

# The .drop_duplicates() Method

- delete duplicated df rows

In [141…

```
# let's check first the length of the .df
len(df)
```

Out[141… 1000

In [142…

```
len(df.drop_duplicates())
# pourquoi la même taille ? le problem est que la metho ne prend en compte separement les valeurs dupliquées de caque colonne
# mais plutot de l'ensemble, et dans ce cas, il s'avere qu'il existe au moins 1 valeur dupliqué par colonne et par ligne
```

Out[142… 1000

In [147…
```
# verifions les params de la method
df.drop_duplicates(subset = ['First Name'], keep = 'first')
```

Out[147…

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **101** | Aaron | Male | 2012-02-17 | 2021-05-02 10:20:00 | 61602 | 11.849 | True | Marketing |
| **137** | Adam | Male | 2011-05-21 | 2021-05-02 01:45:00 | 95327 | 15.120 | False | Distribution |
| **300** | Alan | Male | 1988-06-26 | 2021-05-02 03:54:00 | 111786 | 3.592 | True | Engineering |
| **372** | Albert | Male | 1997-02-01 | 2021-05-02 16:20:00 | 67827 | 19.717 | True | Engineering |
| **988** | Alice | Female | 2004-10-05 | 2021-05-02 09:34:00 | 47638 | 11.209 | False | Human Resources |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **433** | Wanda | Female | 2008-07-20 | 2021-05-02 13:44:00 | 65362 | 7.132 | True | Legal |
| **177** | Wayne | Male | 2012-04-07 | 2021-05-02 08:00:00 | 102652 | 14.085 | True | Distribution |
| **820** | William | Male | 1993-11-18 | 2021-05-02 12:27:00 | 54058 | 5.182 | True | Human Resources |
| **450** | Willie | Male | 2009-08-22 | 2021-05-02 13:03:00 | 55038 | 19.691 | False | Legal |
| **7** | NaN | Female | 2015-07-20 | 2021-05-02 10:43:00 | 45906 | 11.598 | True | Finance |

201 rows × 8 columns

# The .unique() and the .nunique() Metods

In [148…
```
df = pd.read_csv('./pandas/employees.csv', parse_dates = ['Start Date','Last Login Time'])
df['Senior Management'] = df['Senior Management'].astype('bool')
df['Gender'] = df['Gender'].astype('category')
df.sort_values('First Name',inplace = True)
df.head(2)
```

Out[148…

| | First Name | Gender | Start Date | Last Login Time | Salary | Bonus % | Senior Management | Team |
|---|---|---|---|---|---|---|---|---|
| **101** | Aaron | Male | 2012-02-17 | 2021-05-02 10:20:00 | 61602 | 11.849 | True | Marketing |
| **327** | Aaron | Male | 1994-01-29 | 2021-05-02 18:48:00 | 58755 | 5.097 | True | Marketing |

In [152…
```
df['Gender']
```

Out[152…
```
101      Male
327      Male
440      Male
937       NaN
137      Male
        ...
902      Male
925    Female
946    Female
947      Male
951    Female
Name: Gender, Length: 1000, dtype: category
Categories (2, object): ['Female', 'Male']
```

In [158…
```
# unique return la liste de valeurs uniques, les NaN ne sont pas comptés
df['Gender'].unique()
```

Out[158…
```
['Male', NaN, 'Female']
Categories (2, object): ['Male', 'Female']
```

In [160…
```
df['Team'].unique()
```

Out[160…
```
array(['Marketing', 'Client Services', 'Distribution', 'Product',
       'Human Resources', 'Engineering', 'Finance',
       'Business Development', 'Sales', nan, 'Legal'], dtype=object)
```

In [161…
```
len(df['Team'].unique())
```

Out[161…
```
11
```

In [167…
```
# nunique() => number of unique values
df['Team'].nunique()
```

Out[167…
```
10
```

In [165…
```
# Why 11 and 10 ? .nunique() method as default True parameter that do not take account NaN value
df['Team'].nunique(dropna=False)
```

Out[165…
```
11
```