What is pandas?

INTRODUCTION TO DATA SCIENCE IN PYTHON

Hillary Green-Lerman
Lead Data Scientist, Looker

What can pandas do for you?

- Loading tabular data from different sources
- Search for particular rows or columns
- Calculate aggregate statistics
- Combining data from multiple sources

Tabular data with pandas

Tabular Data

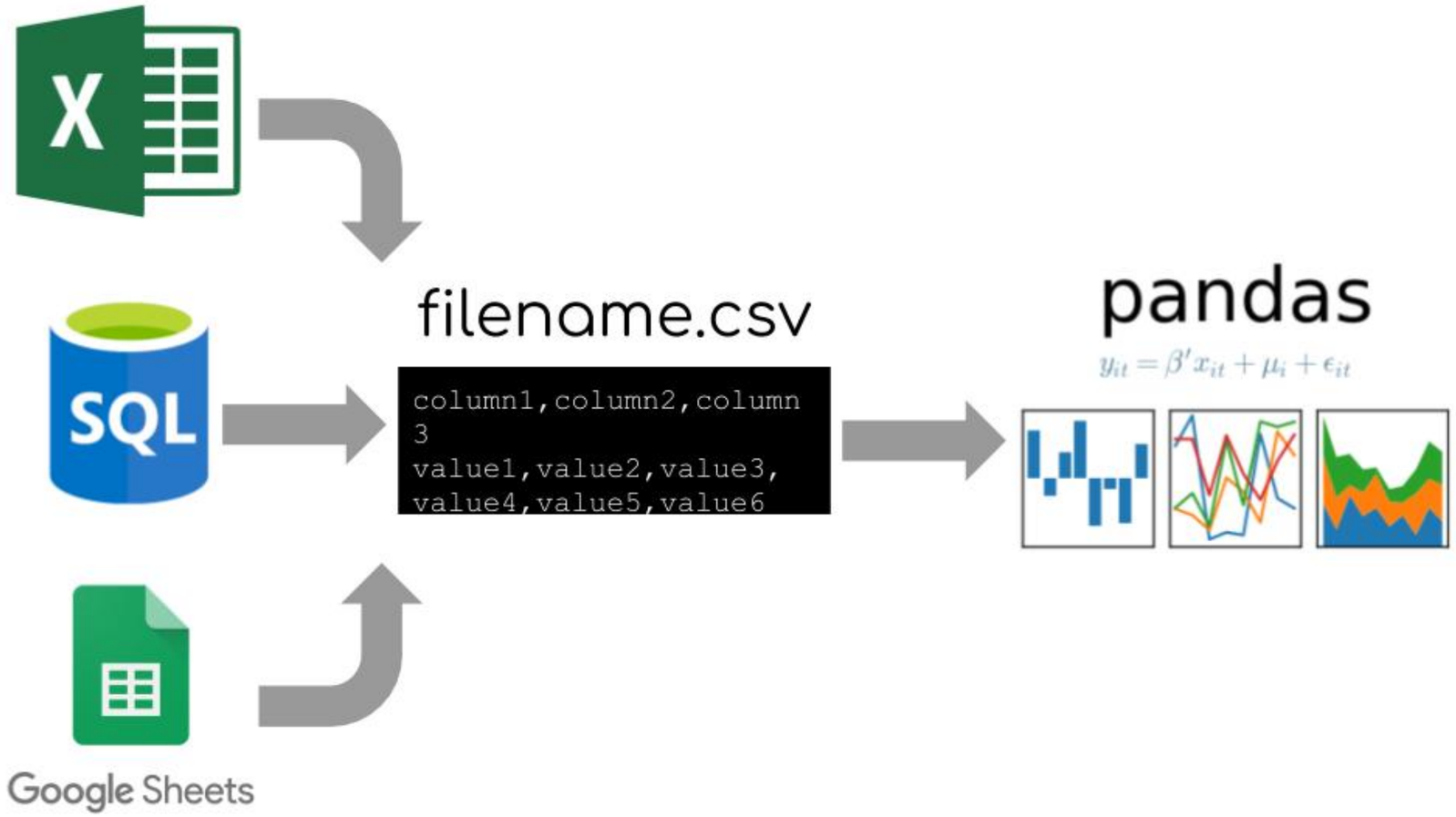
```
+-----------------------------------------------+
|        suspect        |     location    | price |
+-----------------------+-----------------+-------+
| Fred Frequentist      | Petroleum Plaza | 24.95 |
| Ronald Aylmer Fisher  | Clothing Club   | 20.15 |
+-----------------------------------------------+
```

3. Tabular data with pandas
You already know two data types: floats and strings. Pandas introduces a new, more powerful data type: the DataFrame, which represents tabular data. Loading data into a DataFrame is the first step in using Pandas.

DataFrame

```
                suspect         location  price
0      Fred Frequentist  Perolium Plaza  24.95
1  Ronald Aylmer Fisher   Clothing Club  20.15
```

CSV files


filename.csv
column1,column2,column
3
value1,value2,value3,
value4,value5,value6
pandas
$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$
SQL
Google Sheets

Loading a CSV

```
import pandas as pd
```

```
df = pd.read_csv('ransom.csv')
```

5. Loading a CSV
Before we can start using Pandas, we have to import the pandas module. Recall that we always import Pandas under the alias "pd". Next, we create our first DataFrame from a CSV. Turning a CSV into a DataFrame is easy. We use a function: pd-dot-read_csv. This function takes one argument, the name of a CSV file as a string. In this example, the name of the file is ransom-dot-csv.

Displaying a DataFrame

```
df = pd.read_csv('filename.csv')
print(df)
```

```
                suspect         location          item  price
0         Kirstine Smith  Petroleum Plaza           gas  24.95
1       Fred Frequentist      Burger Mart         fries   1.95
2           Gertrude Cox      Burger Mart         fries   1.95
3   Ronald Aylmer Fisher    Clothing Club         shirt  14.25
4         Kirstine Smith    Clothing Club         dress  20.15
5       Fred Frequentist   Groceries R Us     cucumbers   2.05
6         Kirstine Smith    Clothing Club         dress  20.15
7           Gertrude Cox  Petroleum Plaza   fizzy drink   1.90
8           Gertrude Cox      Burger Mart         fries   1.95
9   Ronald Aylmer Fisher    Clothing Club         shirt  14.25
10  Ronald Aylmer Fisher  Petroleum Plaza       carwash  13.25
11  Ronald Aylmer Fisher    Clothing Club         shirt  14.25
12        Kirstine Smith  Petroleum Plaza           gas  24.95
13      Fred Frequentist   Groceries R Us          eggs   6.50
14          Gertrude Cox  Petroleum Plaza           gas  24.95
15      Fred Frequentist   Groceries R Us          eggs   6.50
16  Ronald Aylmer Fisher   Groceries R Us          eggs   6.50
17      Fred Frequentist   Groceries R Us        cheese   5.00
```

6. Displaying a DataFrame
Notice that we saved this DataFrame to a variable called "df". We can display this variable by using the "print" function, which we learned about in a previous lesson. When we print a DataFrame, we get to see every row in the DataFrame. In this example, the DataFrame is so large that it doesn't entirely fit on the slide!

Inspecting a DataFrame

```
df.head()
```

```
print(df.head())
```

```
                suspect         location          item  price
0         Kirstine Smith  Petroleum Plaza           gas  24.95
1      Fred Frequentist      Burger Mart         fries   1.95
2          Gertrude Cox      Burger Mart         fries   1.95
3  Ronald Aylmer Fisher    Clothing Club         shirt  14.25
4         Kirstine Smith    Clothing Club         dress  20.15
```

Inspecting a DataFrame

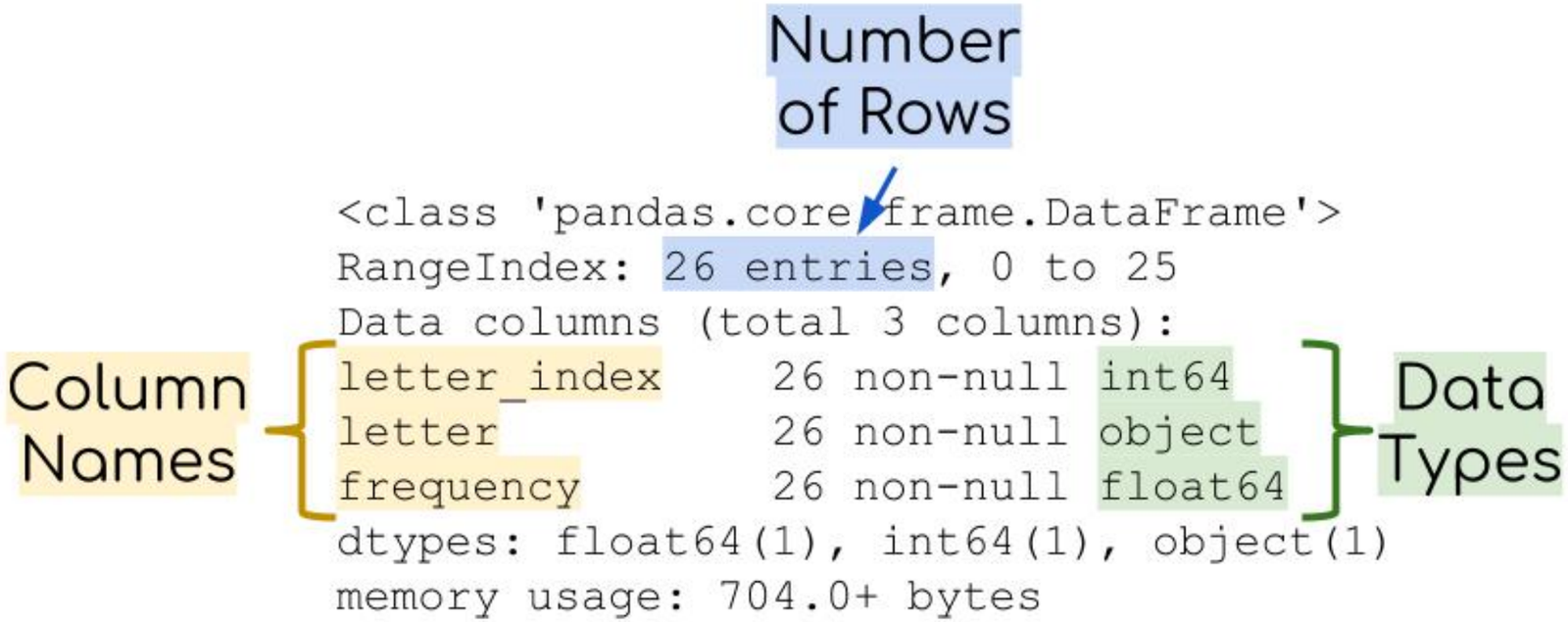
```
df.info()
```

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26 entries, 0 to 25
Data columns (total 3 columns):
letter_index    26 non-null int64
letter          26 non-null object
frequency       26 non-null float64
dtypes: float64(1), int64(1), object(1)
memory usage: 704.0+ bytes
```

9. Inspecting a DataFrame
Notice that we can see the names of the columns, the number of rows, and data types for each column. This method is particularly useful for DataFrames with many columns that are difficult to display using head.

Inspecting a DataFrame



Let's practice!

INTRODUCTION TO DATA SCIENCE IN PYTHON

Selecting columns

INTRODUCTION TO DATA SCIENCE IN PYTHON

Hillary Green-Lerman
Lead Data Scientist, Looker

Why select columns?

- Use in a calculation

```
credit_records.price.sum()
```

- Plot data

```
plt.plot(ransom['letter'], ransom['frequency'])
```

For example, this code selects the column "price" and then calls the method "sum" on that column to get the total amount of money spent by our suspects. We might also want to use the data as the input to a function. You might recognize this code from when we learned about functions. It will create a plot of the frequencies of each letter in the ransom note. The data comes from a DataFrame called "ransom" with columns "letter" and "frequency".

Columns names are strings

```
print(credit_records.head())
```

```
            suspect         location                 date         item  price
0    Kirstine Smith   Groceries R Us   January 6, 2018     broccoli   1.25
1      Gertrude Cox  Petroleum Plaza   January 6, 2018  fizzy drink   1.90
2  Fred Frequentist   Groceries R Us   January 6, 2018     broccoli   1.25
3      Gertrude Cox   Groceries R Us  January 12, 2018     broccoli   1.25
4    Kirstine Smith    Clothing Club   January 9, 2018        shirt  14.25
```

```
'suspect'
'location'
'date'
'item'
'price'
```

Selecting with brackets and string

```
suspect = credit_records['suspect']
```

```
print(suspect)
```

```
0             Kirstine Smith
1               Gertrude Cox
2           Fred Frequentist
3               Gertrude Cox
4             Kirstine Smith
5               Gertrude Cox
...
99              Gertrude Cox
100         Fred Frequentist
101             Gertrude Cox
102           Kirstine Smith
103    Ronald Aylmer Fisher
```

Selecting with a dot

```
price = credit_records.price
```

```
print(price)
```

```
0       1.25
1       1.90
2       1.25
3       1.25
4      14.25
5       3.95
...
99     14.25
100    12.05
101    20.15
102     3.95
103     2.05
```

5. Selecting with a dot
There's a second way we can select columns from a DataFrame. If the columns string only contains letters, numbers, and underscores, we can use dot notation. For dot notation, we simply type the name of the variable, followed by a dot, followed by the name of the column. In this case, we don't use quotation marks around the column name.

Common mistakes in column selection

Use brackets and string for column names with spaces or special characters (`-` , `?` , etc.)

```
police_report['Is Golden Retriever?']
```

NOT

```
police_report.Is Golden Retriever?
```

```
Object `Retriever` not found.
```

Common mistakes in column selection

When using brackets and string, don't forget the quotes around the column name!

```
credit_report['location']
```

NOT

```
credit_report[location]
```

```
Object `location` not found.
```

Common mistakes in column selection

Brackets, not parentheses

```
credit_report['location']
```

NOT

```
credit_report('location')
```

```
---------------------------------------------------------------------------
TypeError  Traceback (most recent call last)
<ipython-input-5-aabdb8981438> in <module>()
----> 1 credit_report('location')

TypeError: 'DataFrame' object is not callable
```

Let's practice!

INTRODUCTION TO DATA SCIENCE IN PYTHON

Select rows with logic

INTRODUCTION TO DATA SCIENCE IN PYTHON

Hillary Green-Lerman
Lead Data Scientist, Looker

Continuing the investigation

```
print(credit_records.head())
```

```
            suspect         location              date         item  price
0    Kirstine Smith   Groceries R Us   January 6, 2018     broccoli   1.25
1      Gertrude Cox  Petroleum Plaza   January 6, 2018  fizzy drink   1.90
2  Fred Frequentist   Groceries R Us   January 6, 2018     broccoli   1.25
3      Gertrude Cox   Groceries R Us  January 12, 2018     broccoli   1.25
4    Kirstine Smith    Clothing Club   January 9, 2018        shirt  14.25
```

Logical statements in Python

```
question = 12 * 8
solution = 96
```

```
question == solution
```

```
True
```

Booleans: `True` and `False`

Other types of logic

>, >=, <, <=

```
price = 2.25
price > 5.00
```

```
False
```

Not equal to

```
name = 'bayes'
name != 'Bayes'
```

```
True
```

Using logic with DataFrames

```
credit_records.price > 20.00
```

```
0       False
1       False
2       False
3       False
4        True
5       False
...
99       True
100      True
101      True
102     False
103     False
```

5. Using logic with DataFrames
In the previous examples, we were just comparing two values. In a DataFrame we can compare one value to all values in a DataFrame. For example, we can compare if each purchase in our credit cared records had a price that was greater than $20. This returns an entire column of True or False.

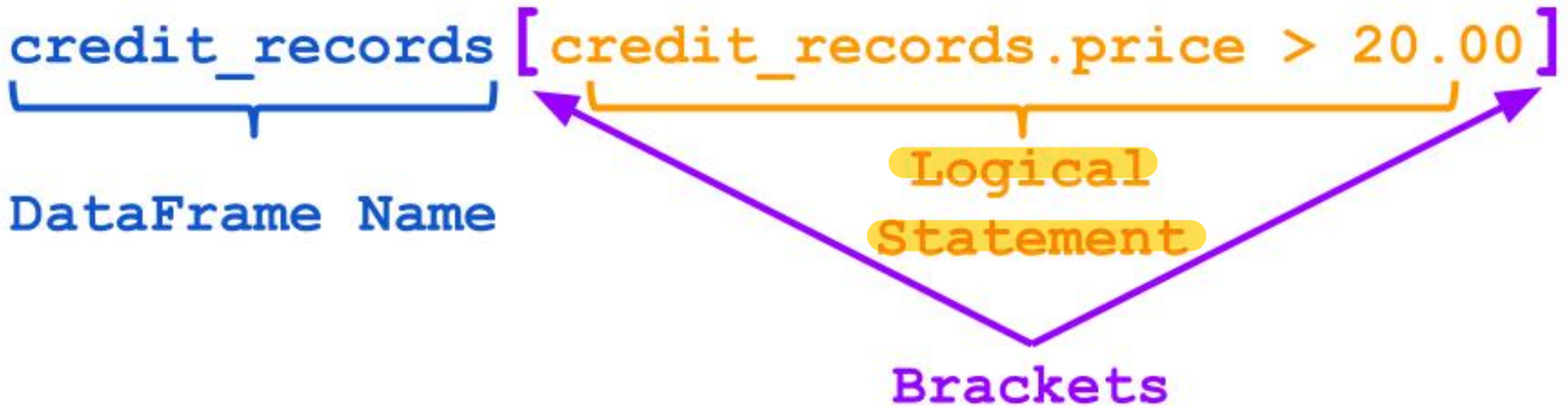
Using logic with DataFrames

```
credit_records[credit_records.price > 20.00]
```

```
                 suspect         location                date   item  price
28       Fred Frequentist    Clothing Club   January 3, 2018  dress  20.15
29         Kirstine Smith    Clothing Club   January 5, 2018  dress  20.15
33   Ronald Aylmer Fisher  Petroleum Plaza   January 7, 2018    gas  24.95
37       Fred Frequentist    Clothing Club   January 8, 2018  dress  20.15
40           Gertrude Cox    Clothing Club   January 1, 2018  dress  20.15
41         Kirstine Smith  Petroleum Plaza   January 5, 2018    gas  24.95
...
```

7. Using logic with DataFrames
Let's examine this line of code more closely. We start with the name of the DataFrame we want to select rows from. In this case, "credit_records". Next we have a set of square brackets. Inside of the square brackets we put our logical test. In this case, the logical test is whether the "price" column of "credit records" is greater than $20. This statement will select all rows of credit_records where the column price is greater than $20.

Using logic with DataFrames



Using logic with DataFrames

```
credit_records[credit_records.suspect == 'Ronald Aylmer Fisher']
```

```
                  suspect         location                date         item  price
7    Ronald Aylmer Fisher    Clothing Club   January 8, 2018        pants  12.05
8    Ronald Aylmer Fisher    Clothing Club  January 13, 2018        shirt  14.25
12   Ronald Aylmer Fisher  Petroleum Plaza  January 10, 2018      carwash  13.25
22   Ronald Aylmer Fisher   Groceries R Us  January 13, 2018         eggs   6.50
26   Ronald Aylmer Fisher      Burger Mart   January 8, 2018        fries   1.95
...
```

8. Using logic with DataFrames
Let's try another example. Suppose, we want to select all rows of credit_records where the "suspect" is equal to "Ronald Aylmer Fisher". Again, we have the name of the DataFrame, followed by square brackets. Inside of the brackets, we check if credit_records-dot-suspect is equal to "Ronald Aylmer Fisher". Note that we use the double equals sign to test equality.

Let's practice!

INTRODUCTION TO DATA SCIENCE IN PYTHON