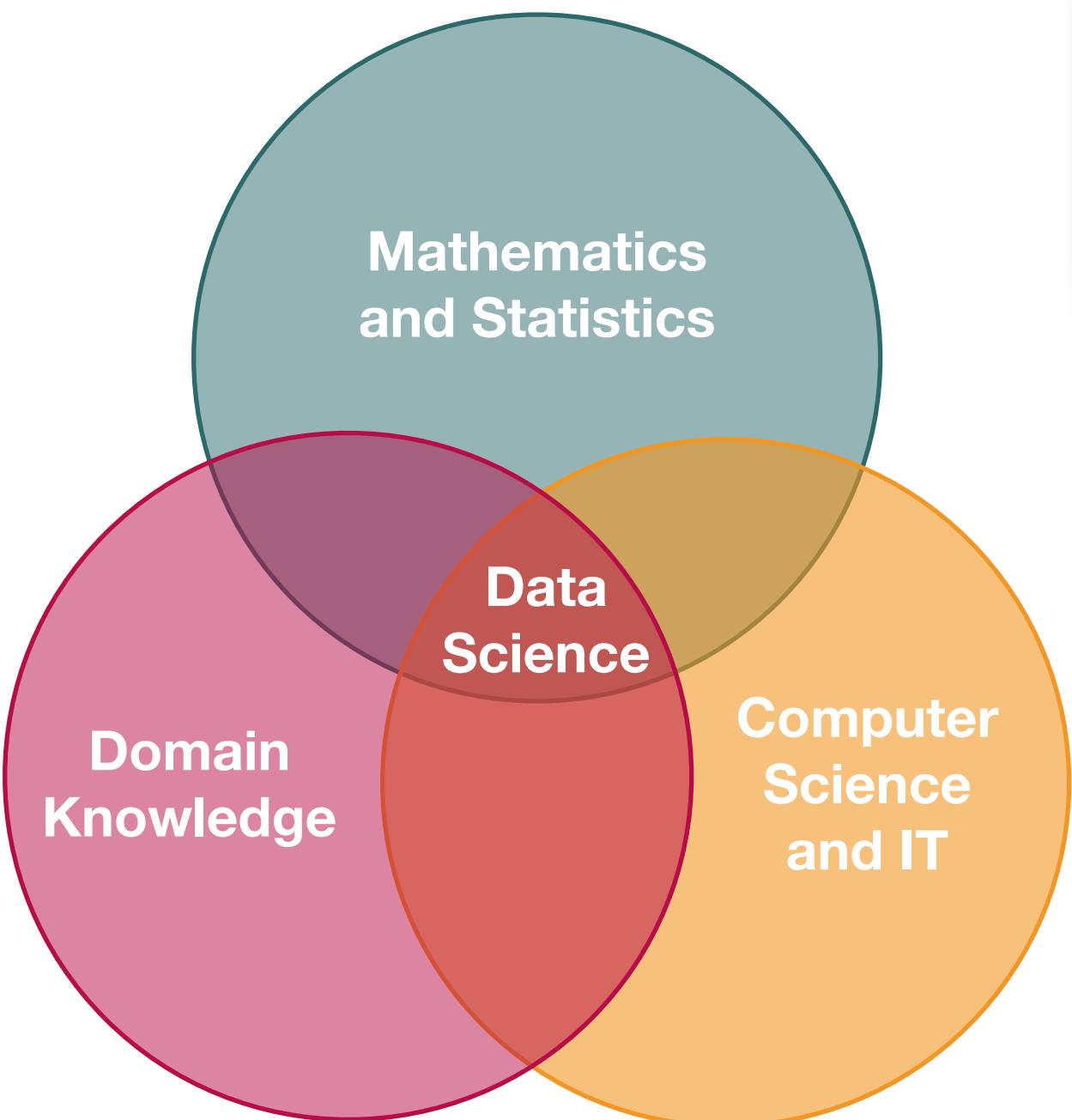


# Static Analysis for Data Scientists

Caterina Urban (Inria & ENS | PSL, France)

# How It Started



## Data Scientist: The Sexiest Job of the 21st Century

Andrew McAfee and Erik Brynjolfsson



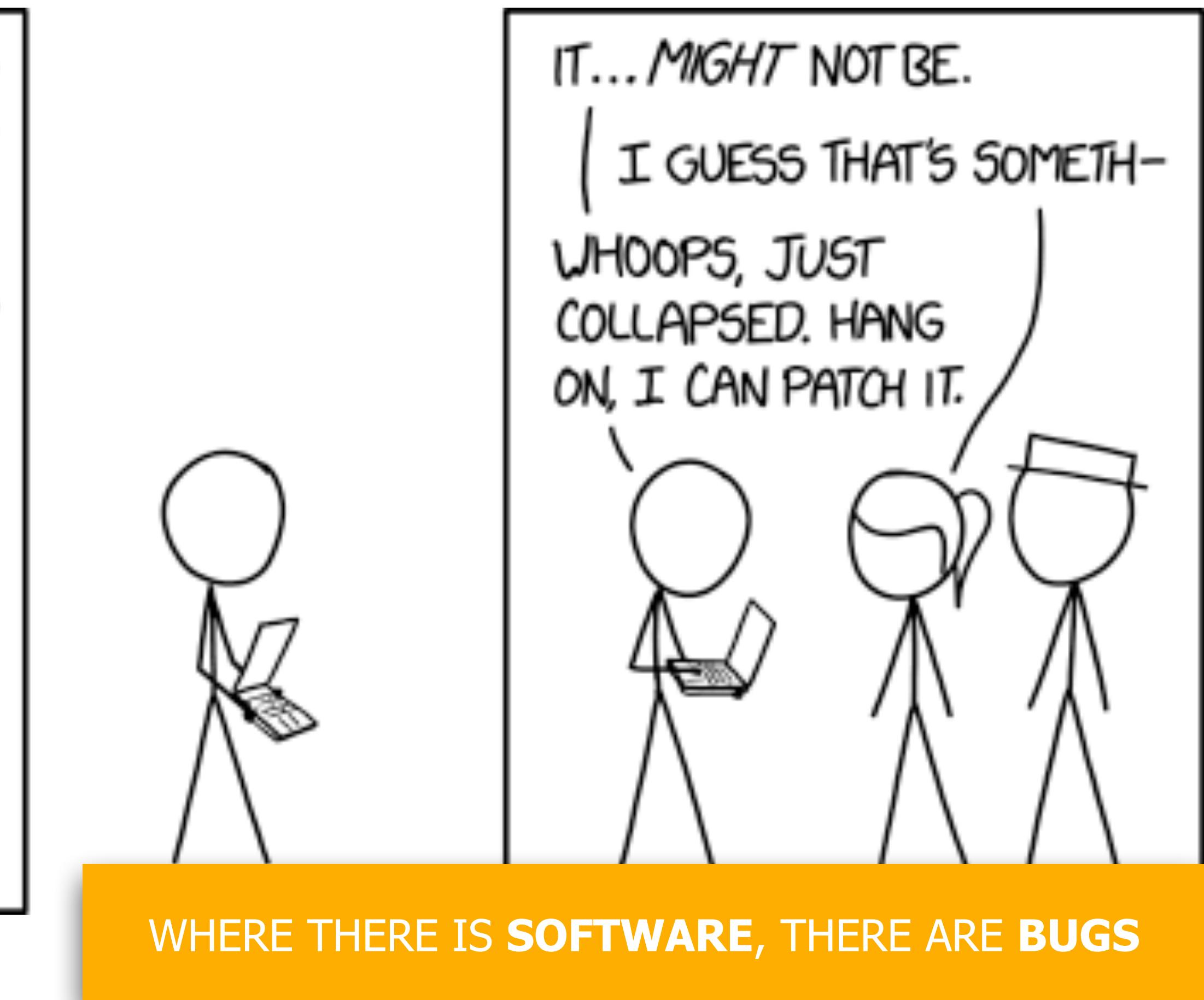
**DATA SCIENTISTS ≠ SOFTWARE DEVELOPERS  
...BUT THEY DEVELOP SOFTWARE NONETHELESS**

Andrew J Buboltz, silk screen on a page from a high school yearbook, 8.5" x 12", 2011 Tamar Cohen

When Jonathan Goldman arrived for work in June 2006 at [LinkedIn](#), the business networking site, the place still felt like a start-up. The company had just under 8 million accounts, and the number was growing quickly as existing members invited their friends and colleagues to join. But users weren't seeking out connections with the people who were already on the site at the rate executives had expected. Something was apparently missing in the social experience. As one LinkedIn manager put it, "It was like arriving at a conference reception and realizing you don't know

who to talk to or what to do or how to fit in the social experience. As one LinkedIn manager put it in a press release, "If we saw it, if it happened in our own world, we'd probably never have heard about it."

# How It Is Going



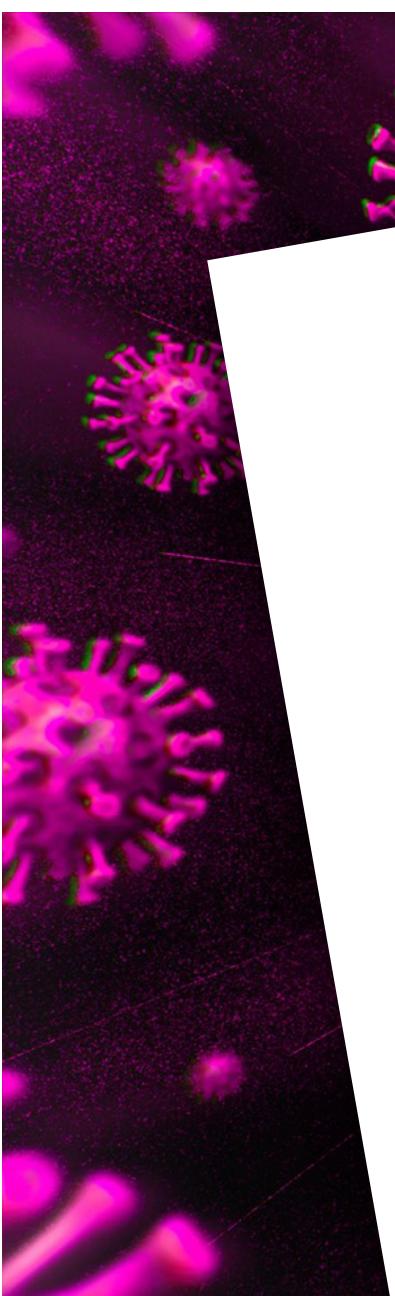
WHERE THERE IS **SOFTWARE**, THERE ARE **BUGS**

<https://xkcd.com/2054/>

# Excel spreadsheet error blamed for UK's 16,000 missing coronavirus cases

*The case went missing after the spreadsheet hit its filesize limit*

By James Vincent | Oct 5, 2020, 9:41am EDT



The BMJ

Cite this as: BMJ 2020;371:m3891  
<http://dx.doi.org/10.1136/bmj.m3891>

Published: 06 October 2020

## Covid-19: Only half of 16 000 patients missed from England's official figures have been contacted

Elisabeth Mahase

Details of nearly 16 000 cases of covid-19 were not transferred to England's NHS Test and Trace service and were missed from official figures because of an error in the process for updating the data.

England's health and social care secretary, Matt Hancock, told the House of Commons on Monday 5 October that after the error was discovered on Friday 2 October "6500 hours of extra contact tracing" had been carried out over the weekend. But as at Monday 1<sup>st</sup> October, 15,511 (51%) of the people had been

WHERE THERE IS DATA SCIENCE SOFTWARE  
THERE ARE INSIDIOUS SILENT BUGS

NEWS

data and furthermore have issued guidance on validation and risk management for these products if they are to be used in such a safety critical manner."

The error came as the Labour Party's leader, Keir Starmer, said that the prime minister had "lost control" of covid-19, with no clear strategy for beating it. Speaking to the Observer, Starmer set out his five point plan for covid-19, which starts with publishing the criteria for local restrictions, as the German government did. Secondly, he said public health messaging should be improved by adding a feature to the NHS covid-19 app so people can search their local restrictions.

BMJ: first published as 10.1136/bmj.m3891 on 6 October 2020. © BMJ Publishing Group Ltd 2020. Reprints and permission: [http://www.bmjjournals.com/info/rights\\_permissions](http://www.bmjjournals.com/info/rights_permissions). DOI: [10.1136/bmj.m3891](http://dx.doi.org/10.1136/bmj.m3891)

## jupyter House Prices (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

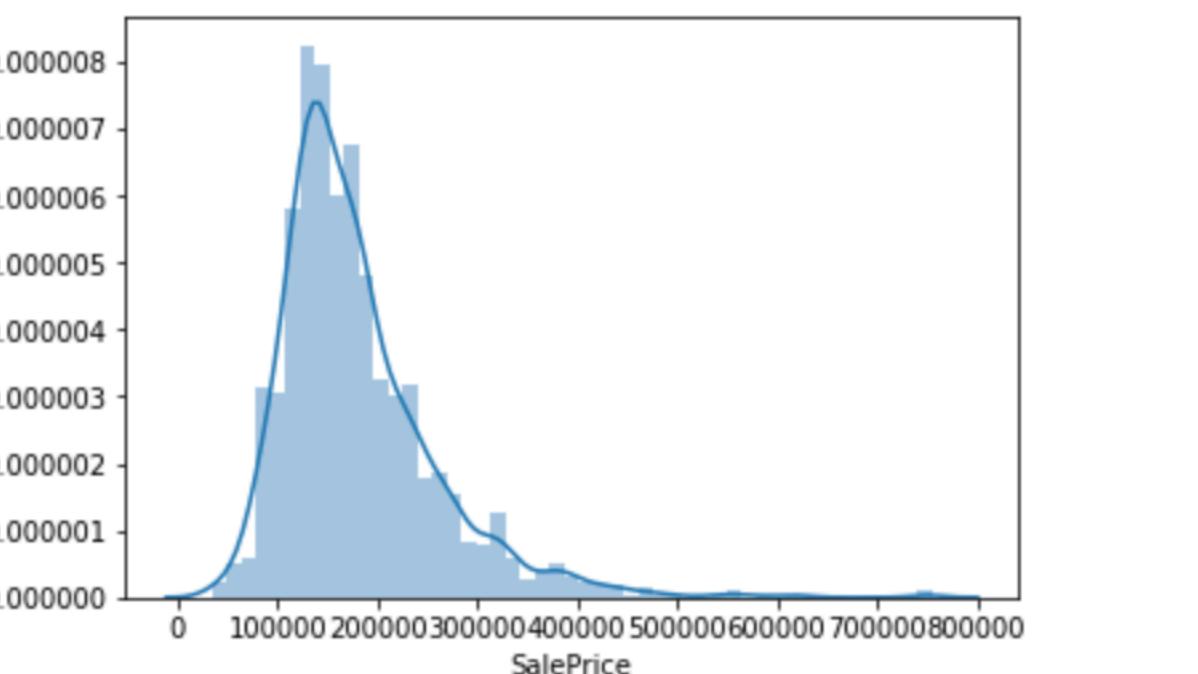
Not Trusted

Logout

In [2]: `df_train = pd.read_csv('../input/HousePrices.csv')`

### Sale Prices

In [5]: `sns.distplot(df_train['SalePrice']);`



### Missing Data

In [14]: `total = df_train.isnull().sum().sort_values(ascending=False)  
percent = (df_train.isnull().sum()/df_train.isnull().count()).sort_values(ascending=False)  
missing_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])  
...`

In [15]: `df_train = df_train.drop((missing_data[missing_data['Total'] > 1]).index,1)  
df_train = df_train.drop(df_train.loc[df_train['Electrical'].isnull()].index)  
df_train.isnull().sum().max()`

Out[15]: 0

# Jupyter Notebooks

# Jupyter Notebooks

NOT JUST A PROTOTYPING TOOL

## Databricks for Data Science

An open and unified platform to collaboratively run all types of analytics workloads, from data preparation to exploratory analysis and predictive analytics, at scale.

The screenshot shows the Databricks interface. On the left, there's a sidebar with icons for Home, Workspace, Recents, Data, and Clusters. The main area is titled 'Loan Analysis' and shows a snippet of SQL code:

```
1. SELECT *
2. FROM silver_loan_stats
3. WHERE int_rate > 6.24
4. ORDER BY int_rate ASC
```

Below the code, there's a preview of a table with columns like loan\_status, int\_rate, revol\_util, issue\_d, earliest\_cr\_line, emp\_length, verification\_status, total\_pymnt, loan\_amnt, grade, annual\_inc, dti, addr\_state, and term. The first two rows are shown:

loan_status	int_rate	revol_util	issue_d	earliest_cr_line	emp_length	verification_status	total_pymnt	loan_amnt	grade	annual_inc	dti	addr_state	term
Fully Paid	6.39	55	Apr-2015	Dec-1999	8	Not Verified	8962.76	8400	A	120000	14.33	NY	36 mon
Fully Paid	6.39	15.4	Mar-2015	Sep-1989	7	Not Verified	12659.08	12000	A	75000	4.66	NY	36 mon

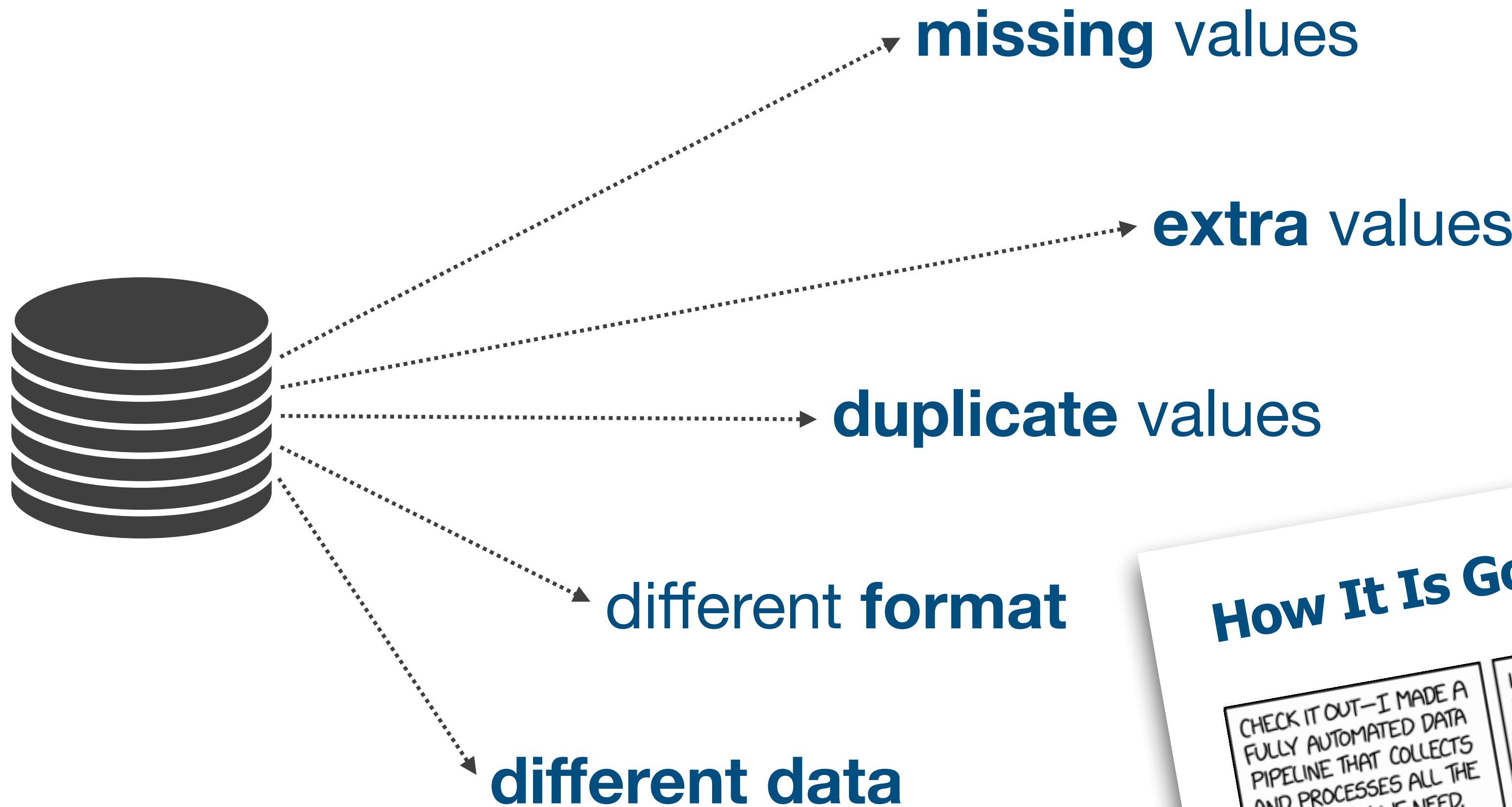
On the right, there's a sidebar with user profiles for Cyrielle Simeone and Jane Data, and a message from Jane: "Let's look at rate > 6.24".

The collage includes the following elements:

- A top-left screenshot of a Jupyter Notebook interface on localhost, showing a cell with Python code: `In [2]: df_train = pd.read_csv('.../input/HousePrices.csv')`.
- A top-middle screenshot of a web browser displaying the Netflix Technology Blog article titled "Beyond Interactive: Notebook Innovation at Netflix" by Michelle Ufford, MPacer, Matthew Seal, and Kyle Kelley.
- A bottom-left screenshot of a Reddit post by u/Desperate-Walk1780, which discusses using Jupyter notebooks in production environments.
- A bottom-middle screenshot of a Jupyter Notebook interface on Databricks, showing a cell with SQL code for loan analysis.
- A bottom-right screenshot of a Jupyter Notebook interface on Databricks, showing a table of house price data.

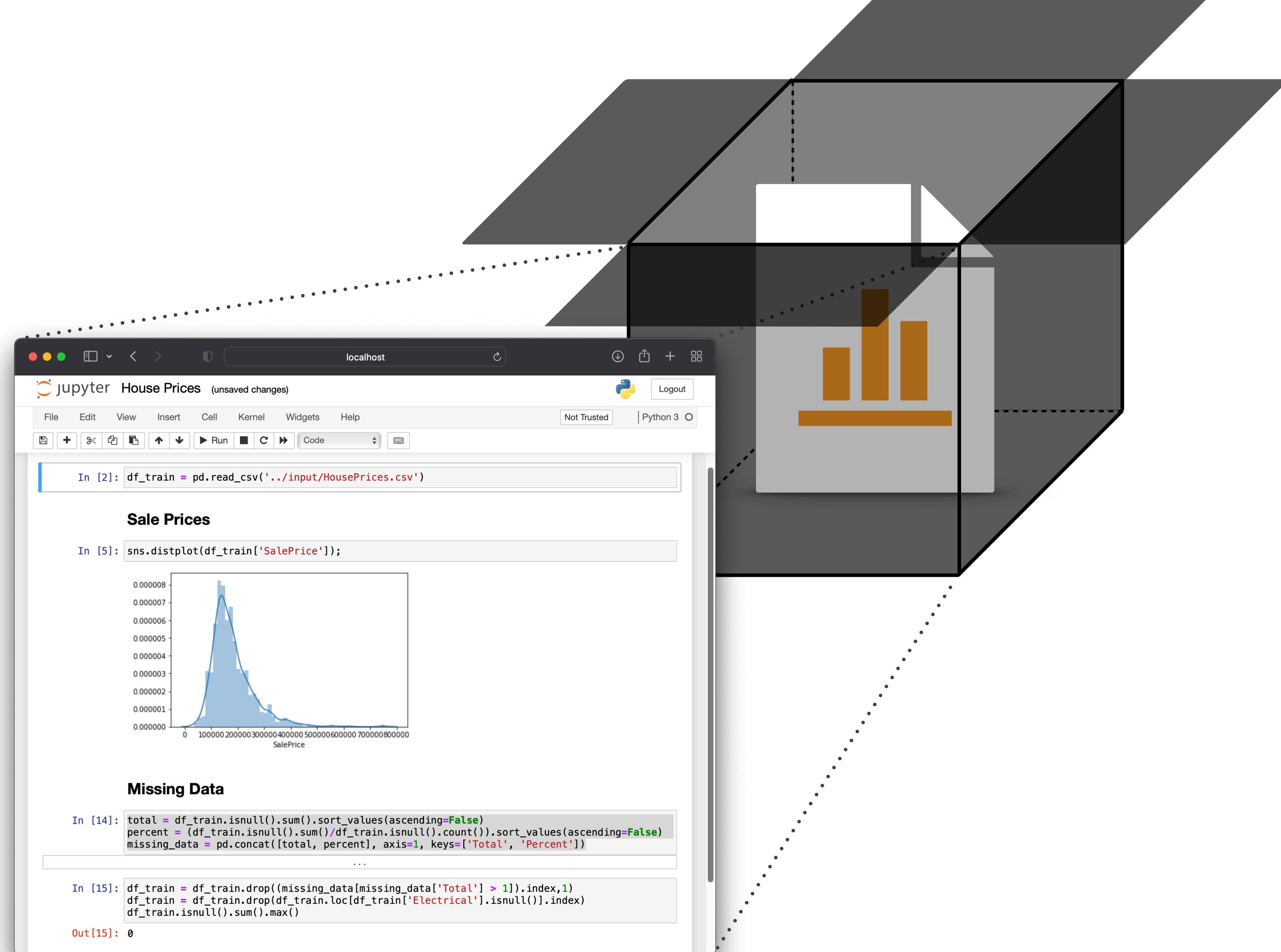
# Unexpected Data

TESTING REQUIRES AWARENESS OF EXPECTATIONS: HARD!



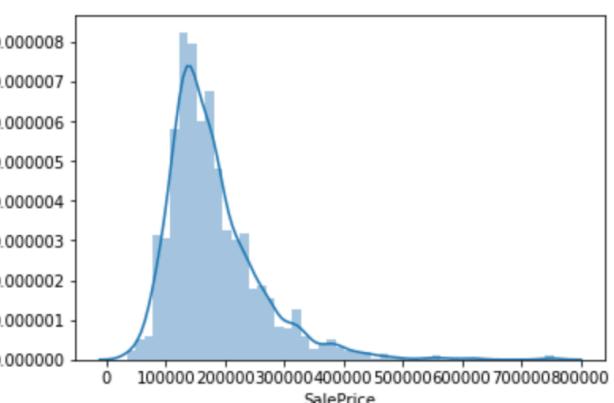
# Data-Expectations Static Analysis

# Data Expectations Analysis



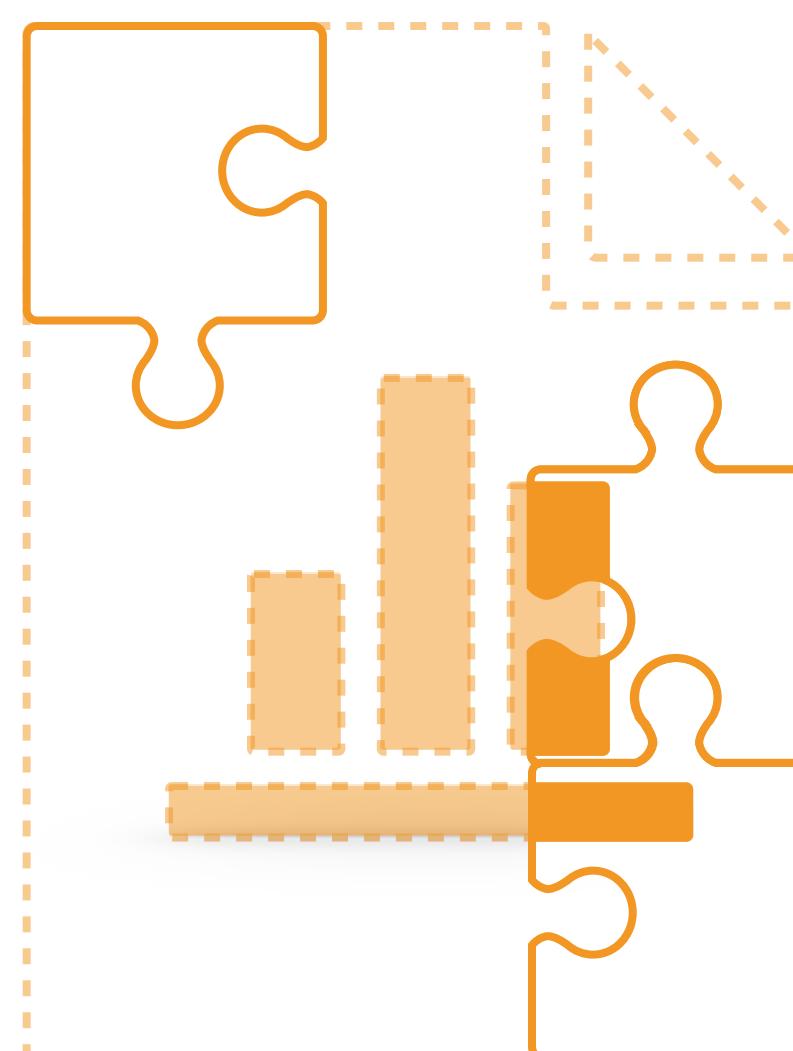
A Jupyter Notebook interface titled "House Prices (unsaved changes)". The notebook shows the following code and plots:

```
In [2]: df_train = pd.read_csv('../input/HousePrices.csv')

Sale Prices
In [5]: sns.distplot(df_train['SalePrice']);

```

```
Missing Data
In [14]: total = df_train.isnull().sum().sort_values(ascending=False)
percent = (df_train.isnull().sum()/df_train.isnull().count()).sort_values(ascending=False)
missing_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])
...
In [15]: df_train = df_train.drop((missing_data[missing_data['Total'] > 1]).index,1)
df_train = df_train.drop(df_train.loc[df_train['Electrical'].isnull()].index)
df_train.isnull().sum().max()

Out[15]: 0
```



# Tabular Data

The diagram illustrates the concept of tabular data analysis. It features a house-shaped icon where the roof is orange with the word "CSV" and the body is grey with a bar chart icon. A dotted line connects the house to a Jupyter Notebook interface on the left. The notebook shows code for reading a CSV file and creating a histogram, along with a histogram plot. To the right is a file icon labeled "names.csv" containing sample data.

**1ST CHALLENGE: MULTI-DIMENSIONAL DATA STRUCTURES**

**names.csv**

fname, Inam  
nancy, davo  
erin , bora  
tony , rapha

df\_train = pd.read\_csv('../input/HousePrices.csv')

Sale Prices

In [5]: sns.distplot(df\_train['SalePrice']);

Missing Data

In [14]: total = df\_train.isnull().sum().sort\_values(ascending=False)  
percent = (df\_train.isnull().sum()/df\_train.isnull().count()).sort\_values(ascending=False)  
missing\_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])  
...

In [15]: df\_train = df\_train.drop((missing\_data[missing\_data['Total'] > 1]).index,1)  
df\_train = df\_train.drop(df\_train.loc[df\_train['Electrical'].isnull()].index)  
df\_train.isnull().sum().max()

Out[15]: 0

# Abstract Interpretation Recipe

**practical tools**

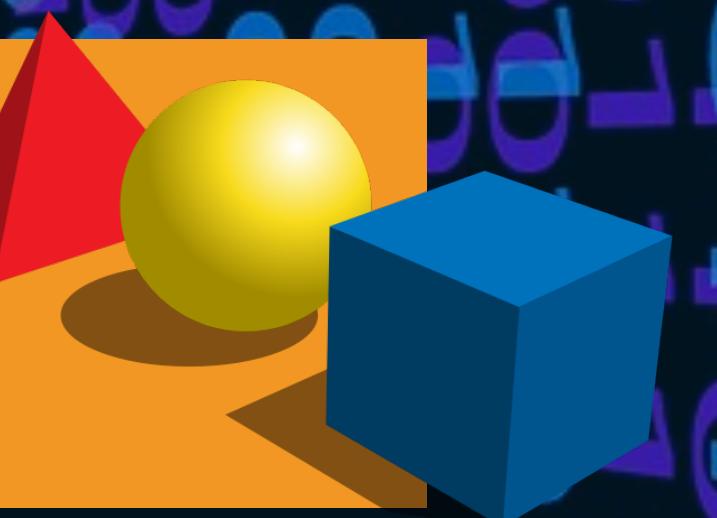
targeting specific programs

**algorithmic approaches**

to decide program properties

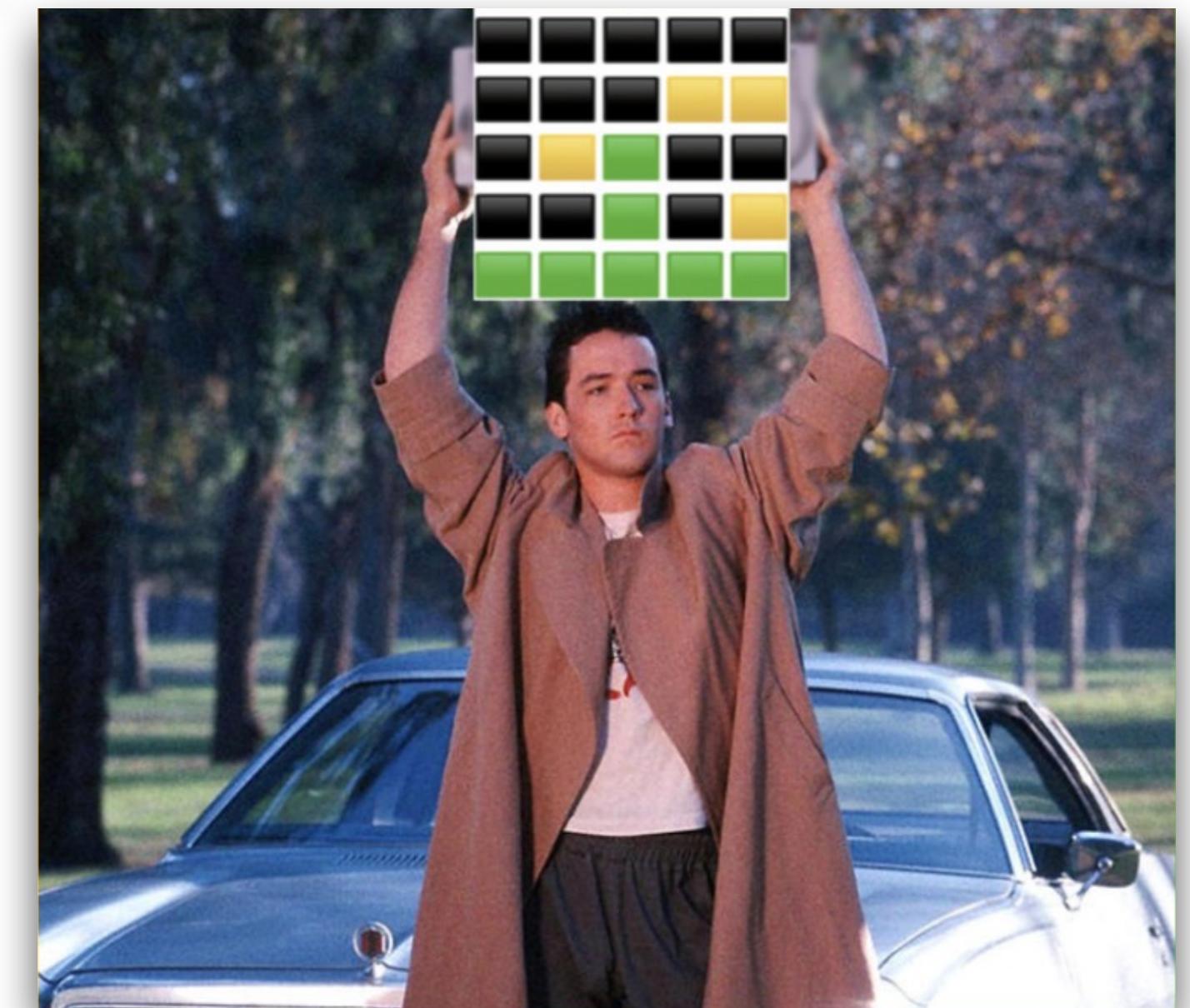
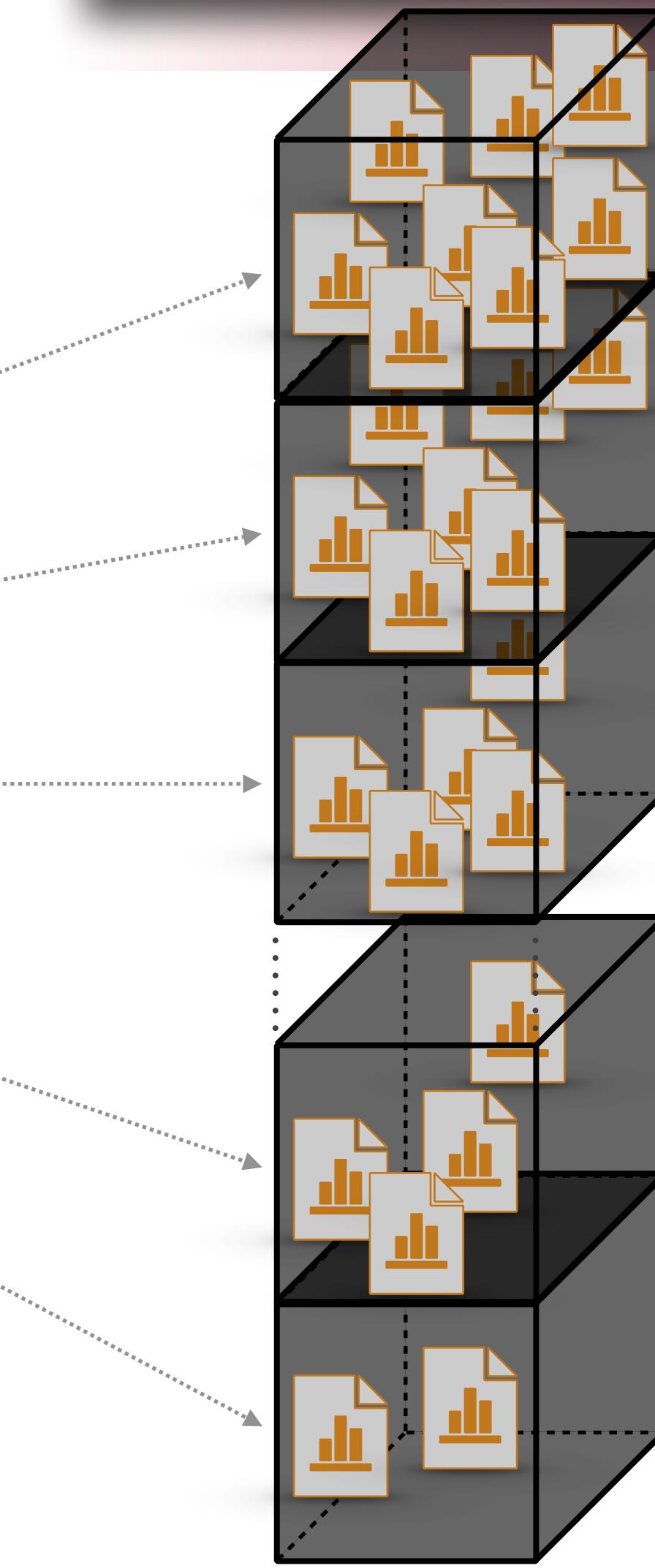
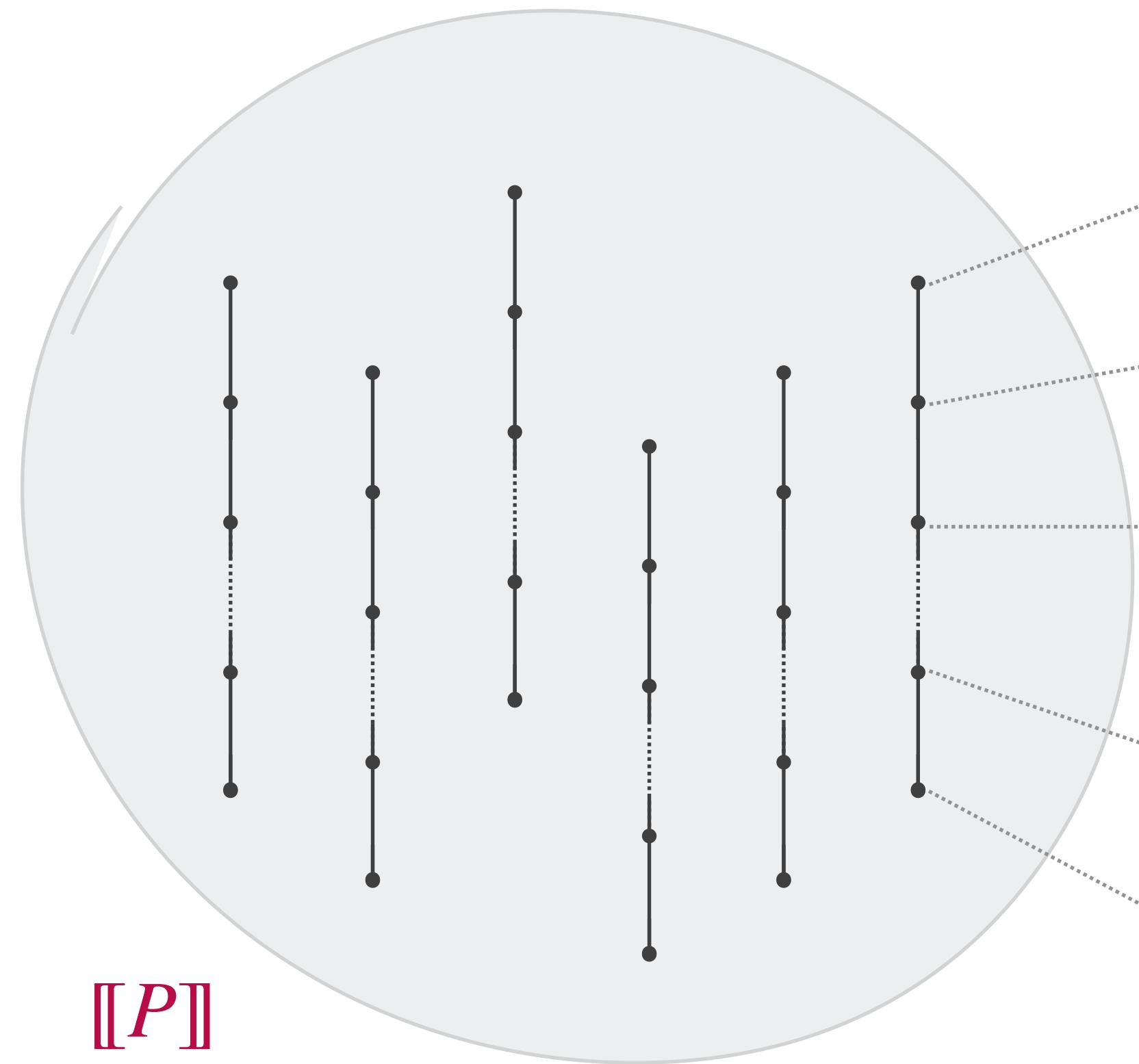
**mathematical models**

of the program behavior



# Concrete Semantics

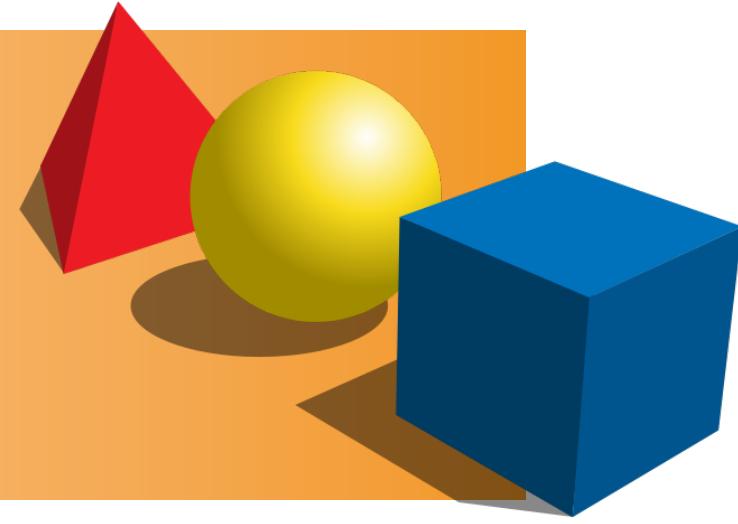
2<sup>ND</sup> CHALLENGE: INDIRECT REASONING



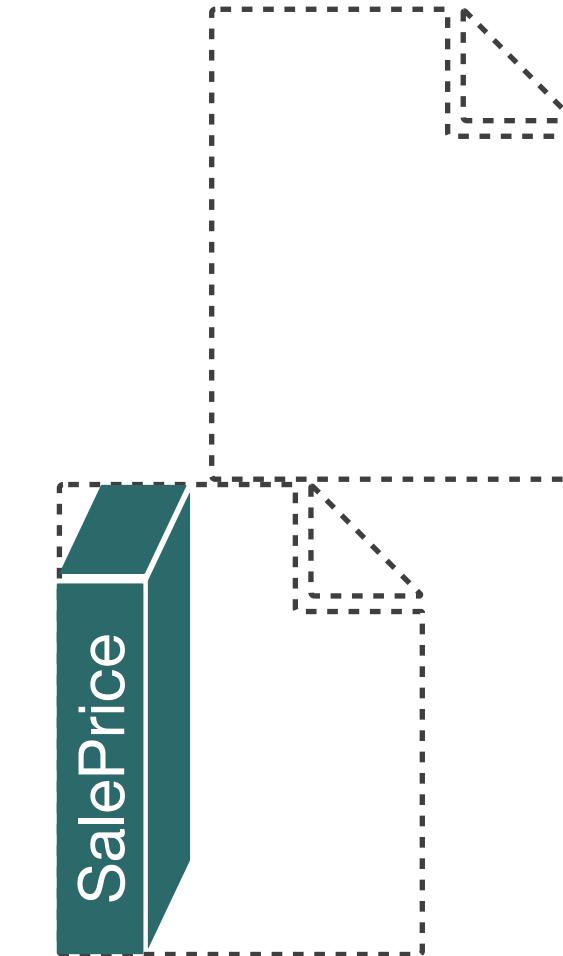
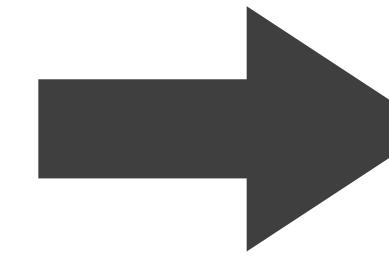
Michael J. Seidliger/@mjsiedliger on Twitter

# Abstract Semantics

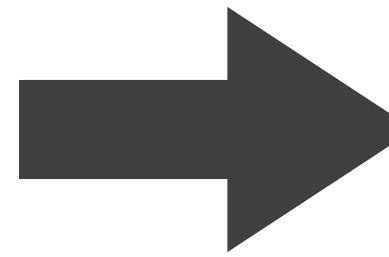
## Column/Field Names Expectations



```
import pandas as pd  
df = pd.read_csv("HousePrices.csv")
```

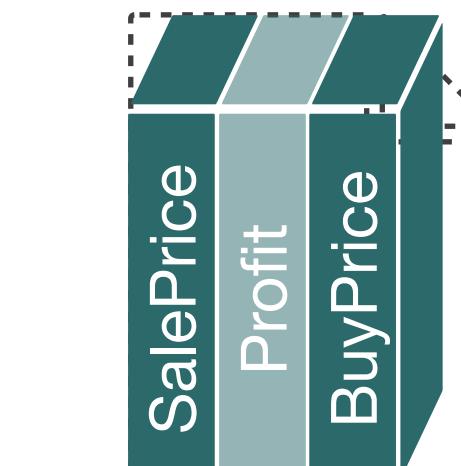
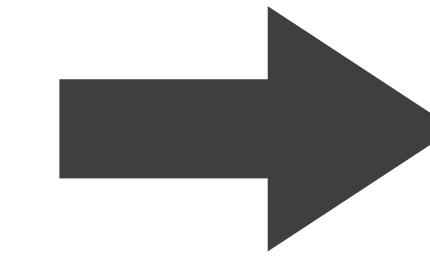


```
ex = df[df.SalePrice >= 1000000]
```

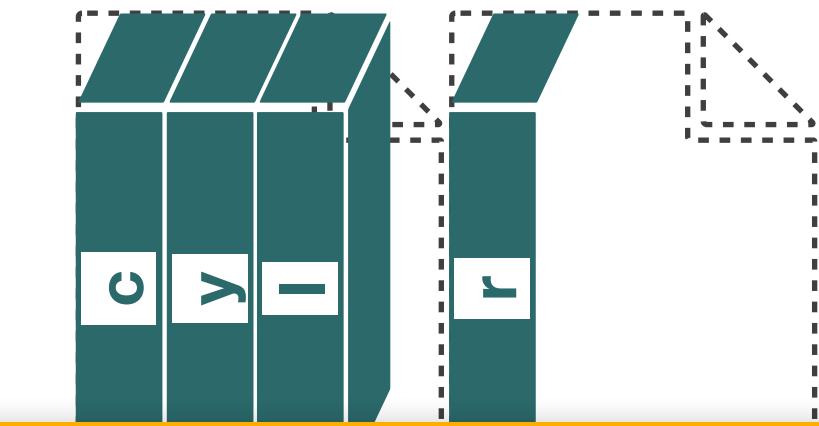
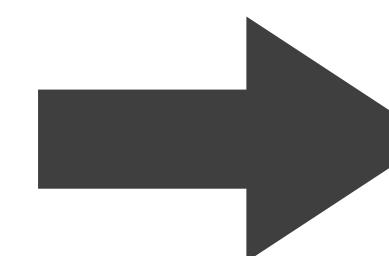


```
ex = df
```

```
ex['Profit'] = ex['SalePrice'] - ex['BuyPrice']
```



```
⋮  
dL = pd.read_csv("L.csv")  
dP = dL.pivot(index=c, columns=y, values=l)  
dR = pd.read_csv("R.csv")  
dG = dP.loc[:, 0:35].groupby(dR[r])
```

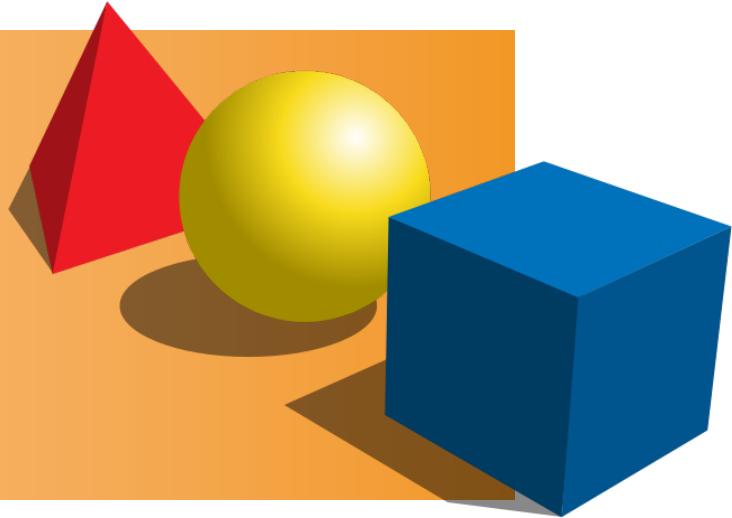


$$\begin{aligned} dR[r] &\in dG \\ dP \cap dR & \end{aligned}$$

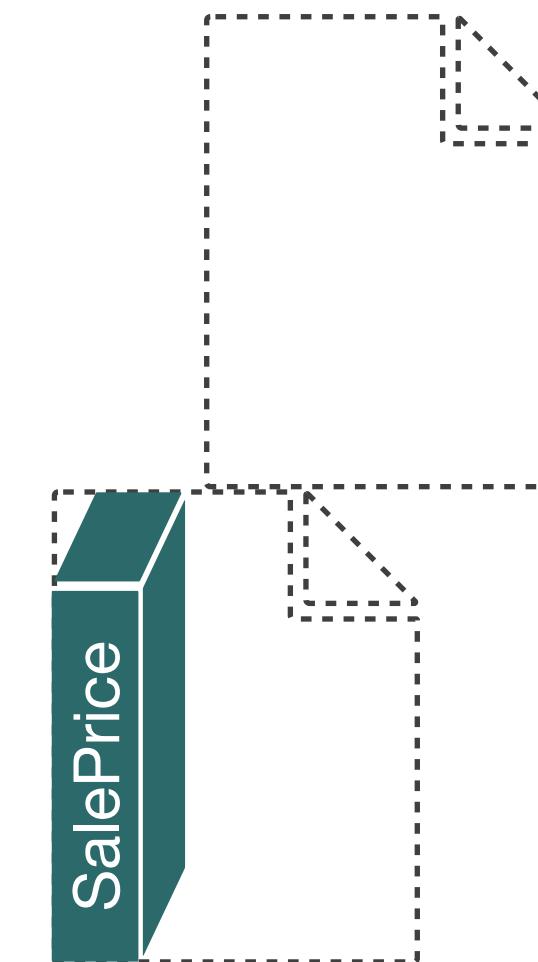
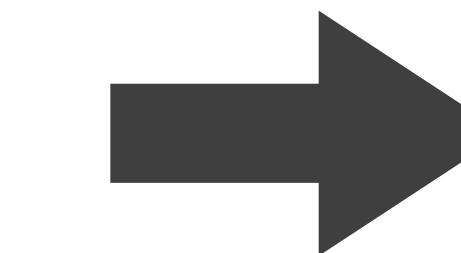
3<sup>RD</sup> CHALLENGE: COMPLEX LIBRARY CALLS

# Practical Static Analysis

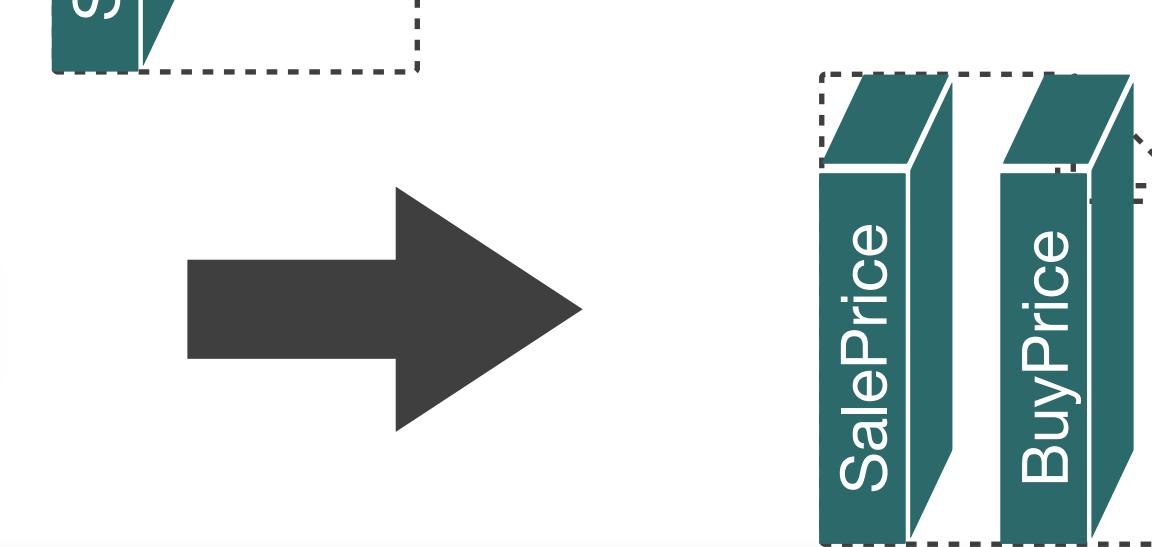
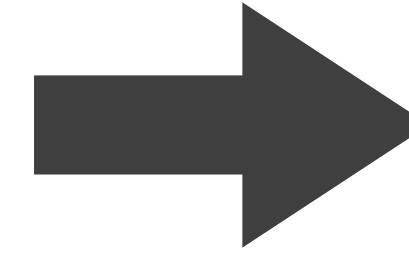
## Necessary vs Sufficient Expectations



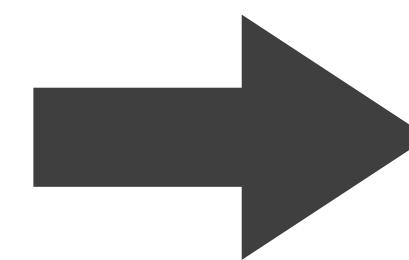
```
import pandas as pd  
df = pd.read_csv("HousePrices.csv")
```



```
ex = df[df.SalePrice >= 1000000]
```



```
ex['Profit'] = ex['SalePrice'] - ex['BuyPrice']
```



**VIOLATING NECESSARY EXPECTATIONS CAUSES TROUBLE FOR SURE**

# Implementation Wish List

The screenshot shows a Jupyter Notebook interface with several annotations:

- INTERACTIVE STATIC ANALYSIS**: A yellow callout points to the code cell `In [5]: sns.distplot(df_train['SalePrice']);` which displays a histogram of house sale prices.
- Missing Data**: A yellow callout points to the code cell `In [14]: total = df_train.isnull().sum().sort_values(ascending=False)` which calculates the total number of missing values across all columns.
- STATIC AND DYNAMIC ANALYSIS COMBINATIONS**: A yellow callout points to the code cells `In [15]: df_train = df_train.drop(missing_data.index[missing_data.sum() > 0])` and `df_train = df_train.drop(df_train.loc[missing_data.sum() > 0].index)`, which drop rows with missing data.

Annotations are highlighted with orange arrows pointing from the text labels to the corresponding code cells.

```
In [5]: sns.distplot(df_train['SalePrice']);

In [14]: total = df_train.isnull().sum().sort_values(ascending=False)
percent = (df_train.isnull().sum()/df_train.isnull().count()).sort_values(ascending=False)
missing_data = pd.concat([total, percent], axis=1)
missing_data.columns = ['Total_NaN', 'Percent_NaN']

In [15]: df_train = df_train.drop(missing_data.index[missing_data.sum() > 0])
df_train = df_train.drop(df_train.loc[missing_data.sum() > 0].index)

Out[15]: 0
```

MULTI-LANGUAGE SUPPORT



# Data Expectations Analysis

```
In [2]: df_train = pd.read_csv('../input/HousePrices.csv')

Sale Prices
In [5]: sns.distplot(df_train['SalePrice'])

Missing Data
In [14]: total = df_train.isnull().sum().sort_values(ascending=False)
percent = (df_train.isnull().sum() / df_train.isnull().count()).sort_values(ascending=False)
missing_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])

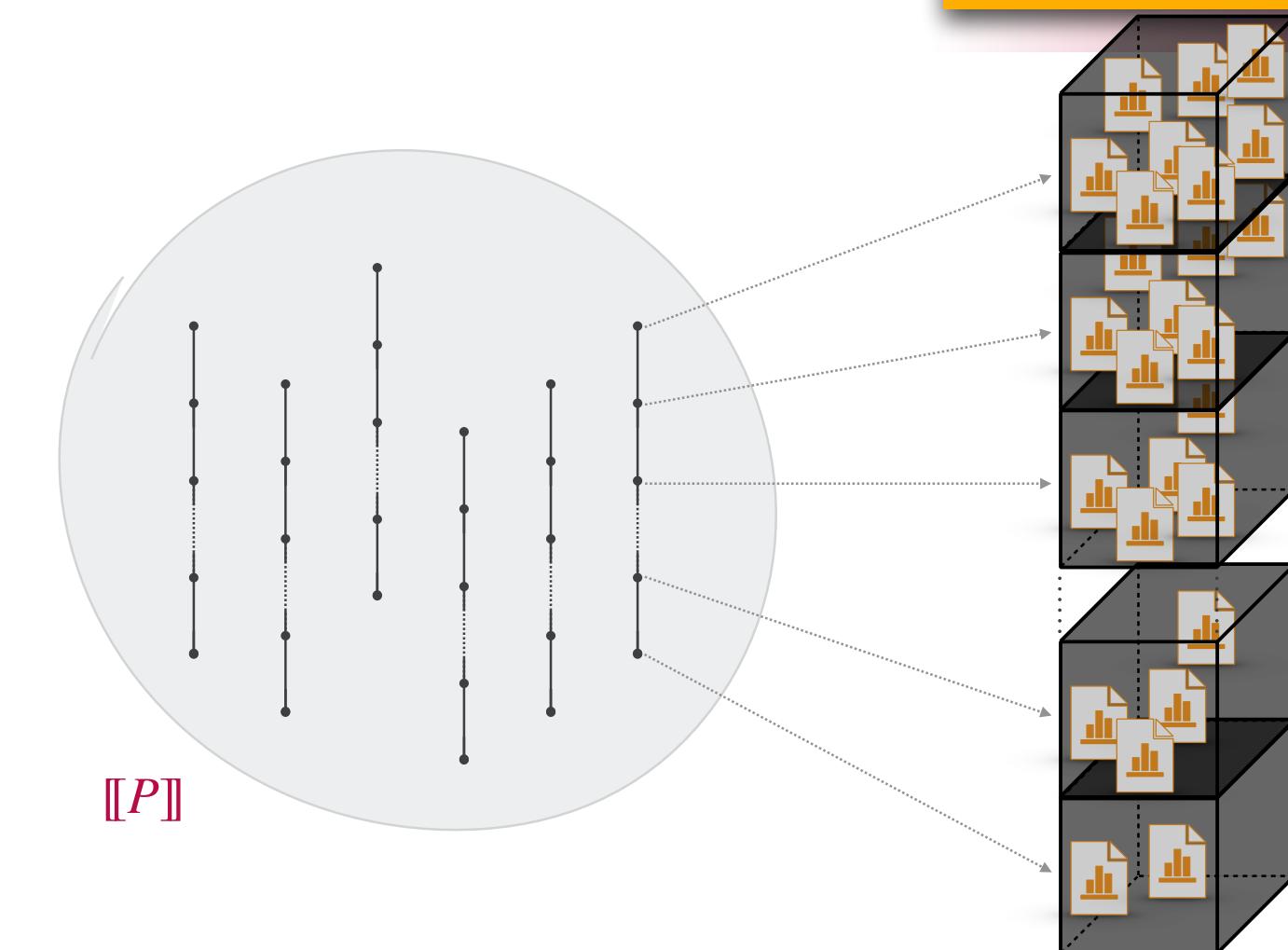
In [15]: df_train = df_train.drop(missing_data['Total'] > 1).index
df_train = df_train.drop(df_train.loc[df_train['Electrical'].isnull()].index)
df_train.isnull().sum().max()

Out[15]: 0
```

9

# Concrete Semantics

2ND CHALLENGE: INDIRECT REASONING



12

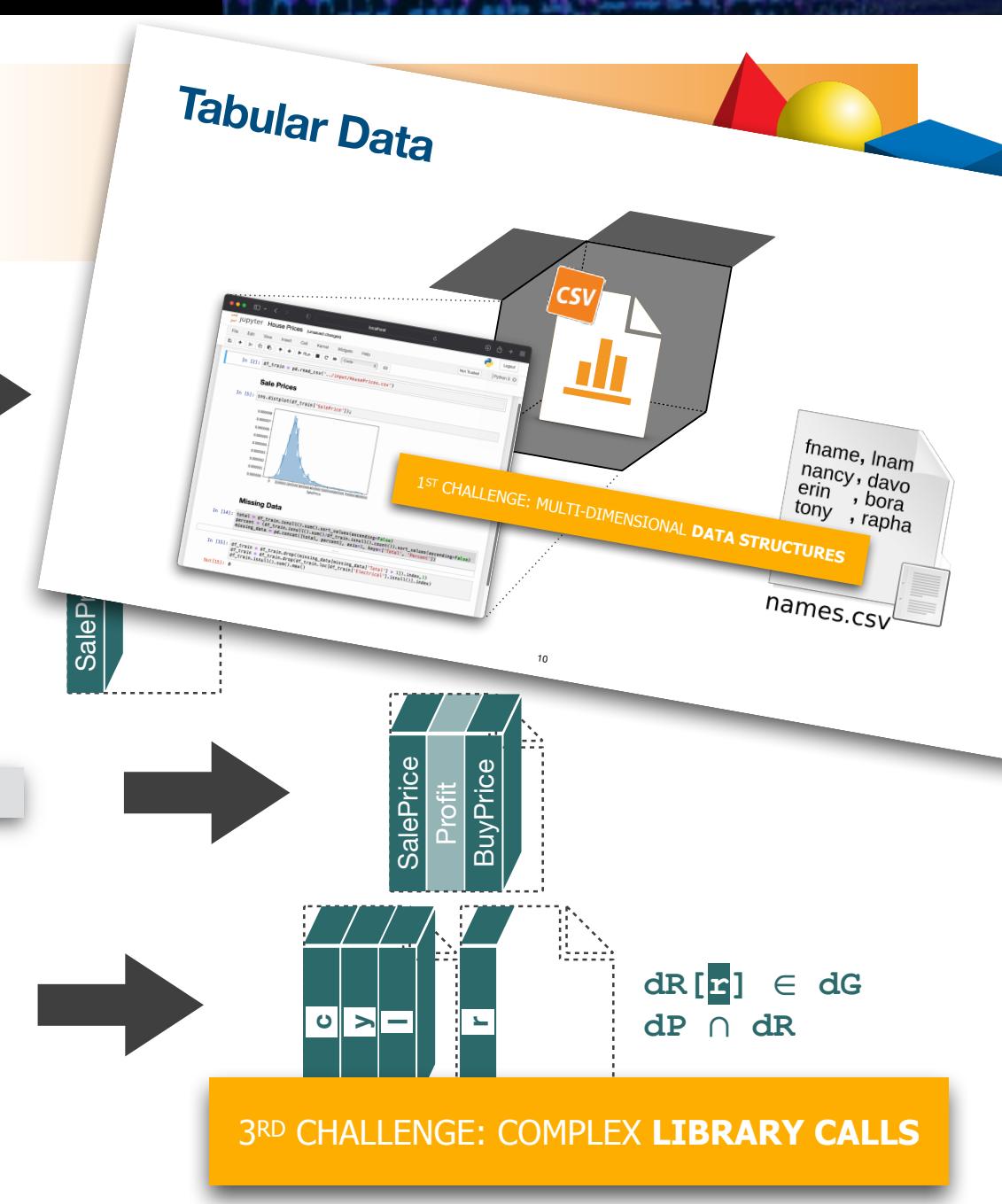
## Abstract Semantics Column/Field Names Expectations

```
import pandas as pd
df = pd.read_csv("HousePrices.csv")

ex = df[df.SalePrice >= 1000000]

ex['Profit'] = ex['SalePrice'] - ex['BuyPrice']

...
dL = pd.read_csv("L.csv")
dP = dL.pivot(index=c, columns=y, values=1)
dR = pd.read_csv("R.csv")
dG = dP.loc[:, 0:35].groupby(dR[r])
```



13

## Implementation Wish List

MULTI-LANGUAGE SUPPORT

INTERACTIVE STATIC ANALYSIS

STATIC AND DYNAMIC ANALYSIS COMBINATIONS

```
In [2]: df_train = pd.read_csv('../input/HousePrices.csv')

Sale Prices
In [5]: sns.distplot(df_train['SalePrice'])

Missing Data
In [14]: total = df_train.isnull().sum().sort_values(ascending=False)
percent = (df_train.isnull().sum() / df_train.isnull().count()).sort_values(ascending=False)
missing_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])

In [15]: df_train = df_train.drop(missing_data['Total'] > 1).index
df_train = df_train.drop(df_train.loc[df_train['Electrical'].isnull()].index)
df_train.isnull().sum().max()

Out[15]: 0
```