

Datas Importantes

Atos acadêmicos no SIGA - Calendário Trimestral	1º Trimestre	2º Trimestre	3º Trimestre	4º Trimestre
Início de atividades	06/07/2020	13/10/2020	01/02/2021	----- X -----
Rematricula de matrícula trancada (destrancamento de matrícula)	Até 27/06/2020	Até 05/10/2020	Até 23/01/2021	----- X -----
Previsão de turmas	Até 19/06/2020	Até 26/09/2020	Até 15/01/2021	----- X -----
Trancamento de matrícula	Até 10/08/2020	Até 17/11/2020	Até 08/03/2021	----- X -----
Pedido de inscrição em disciplinas	De 06/07/2020 a 24/07/2020	De 11/10/2020 a 17/10/2020	De 30/01/2021 a 05/02/2021	----- X -----
Concordância do pedido de inscrição em disciplina	De 27/07/2020 a 30/07/2020	De 18/10/2020 a 24/10/2020	De 06/02/2021 a 12/02/2021	----- X -----
Efetivação do Pedido de Inscrição (Divisão de Ensino – PR2)	31/07/2020	27/10/2020	15/02/2021	----- X -----
Pedido de alteração de inscrição em disciplina – AID	De 02/08/2020 a 08/08/2020	De 28/10/2020 a 31/10/2020	De 16/02/2021 a 19/02/2021	----- X -----
Concordância do pedido de alteração de inscrição em disciplina - AID	De 09/08/2020 a 12/08/2020	De 01/11/2020 a 07/11/2020	De 20/02/2021 a 26/02/2021	----- X -----
Efetivação De Alteração do Pedido de Inscrição (Divisão de Ensino – PR2)	13/08/2020	10/11/2020	01/03/2021	----- X -----
Pedido de trancamento de inscrição em disciplina (desistência de inscrição)	De 14/08/2020 a 19/08/2020	De 11/11/2020 a 14/11/2020	De 02/03/2021 a 05/03/2021	----- X -----
Concordância do pedido de trancamento de inscrição em disciplina	De 20/08/2020 a 31/08/2020	De 15/11/2020 a 28/11/2020	De 06/03/2021 a 19/03/2021	----- X -----
Efetivação do Trancamento do Pedido de Inscrição (Divisão de Ensino – PR2)	24/08/2020	01/12/2020	22/03/2021	----- X -----
Término de atividades	03/10/2020	16/01/2021	24/04/2021	----- X -----
Notas – Pautas de graus e frequência	De 04/10/2020 a 17/10/2020	De 17/01/2021 a 30/01/2021	De 25/04/2021 a 08/05/2021	----- X -----

Avaliação e Atendimento

Critérios de aprovação são os do PPGI/UFRJ.

A avaliação da disciplina consiste em participação em sala de aula (P); exercícios e/ou protótipos desenvolvidos (E); apresentações/ /escritas de artigos (A).

$$MF = 0.2 * P + 0.2 * E + 0.6 * A$$

O aluno que desejar atendimento deverá requisitar o mesmo por e-mail e um horário será agendado pelos responsáveis para o atendimento.



serra@pet-si.ufrj.br



zavaleta@pet-si.ufrj.br

Entregáveis – Março

4/3 - Aula + Definição dos grupos + PIT 5 min (Apresentação geral em PPT - DataSet + Problema + Objetivo)

11/3 - Aula + Descrição do projeto de DS - Tudo é via GIT!

- Entregar o projeto contendo (V1): Detalhamento e descrição textual da definição do problema a ser explorado pela equipe; descrever tecnicamente o dataset e sua fonte, o que pretendem fazer e o que vão e como extrair (plano dos experimentos). Apresentar os objetivos geral e específico do projeto de DS; apresentar métodos de data cleaning/tratamento de dados que serão usados, apresentar a proposta de modelo de extração de conhecimento e visualização de dados que será adotado.

18/3 - Aula + Refinamento do projeto de DS

- Agregar ao projeto (V2) : Plano do experimento a ser executado (scripts no Colab x Jupyter x IDE), projeto de coleta de metadados da proveniência dos experimentos, projeto para tornar seu experimento reproduzível, adicionar qualquer outro melhoria ou diferencial que julguem necessário

25/3 – Aula + entregar da 1a versão do artigo – Recomenda-se usar o template da LNCS e suas regras de formatação.

- Texto final (15-20) páginas com referências (pode ser em português ou inglês, escolha do grupo)

Entregáveis – Abril

1/4 - Aula + Sorteio ordem de apresentação dos grupos

8/4 - Aula + Entregar da 2ª versão do artigo + Apresentação trabalho alunos (à confirmar, vai depender do quantitativo de grupos)

15/4 - Apresentação - trabalho alunos (1ª. Parte dos grupos)

22/4 - Apresentação - trabalho alunos (2ª. Parte dos grupos)

- Entrega da versão final do artigo + projeto completo (V3) + scripts com provenance + datasets anotados+ resultados de DS+ Depósito do notebook reprodutível e executáveis no GIT



Introduction to Data Science

MODULE II – PART I

Data Scraping

Prof Sergio Serra e Jorge Zavaleta

What? The Data Science Process

Ask an interesting question
& learn reproducibility

Get the Data

Explore the Data

Model the Data

Communicate/Visualize the Results

How were the data sampled?

Which data are relevant?

Are there privacy issues?

Module II

Where do data come from?

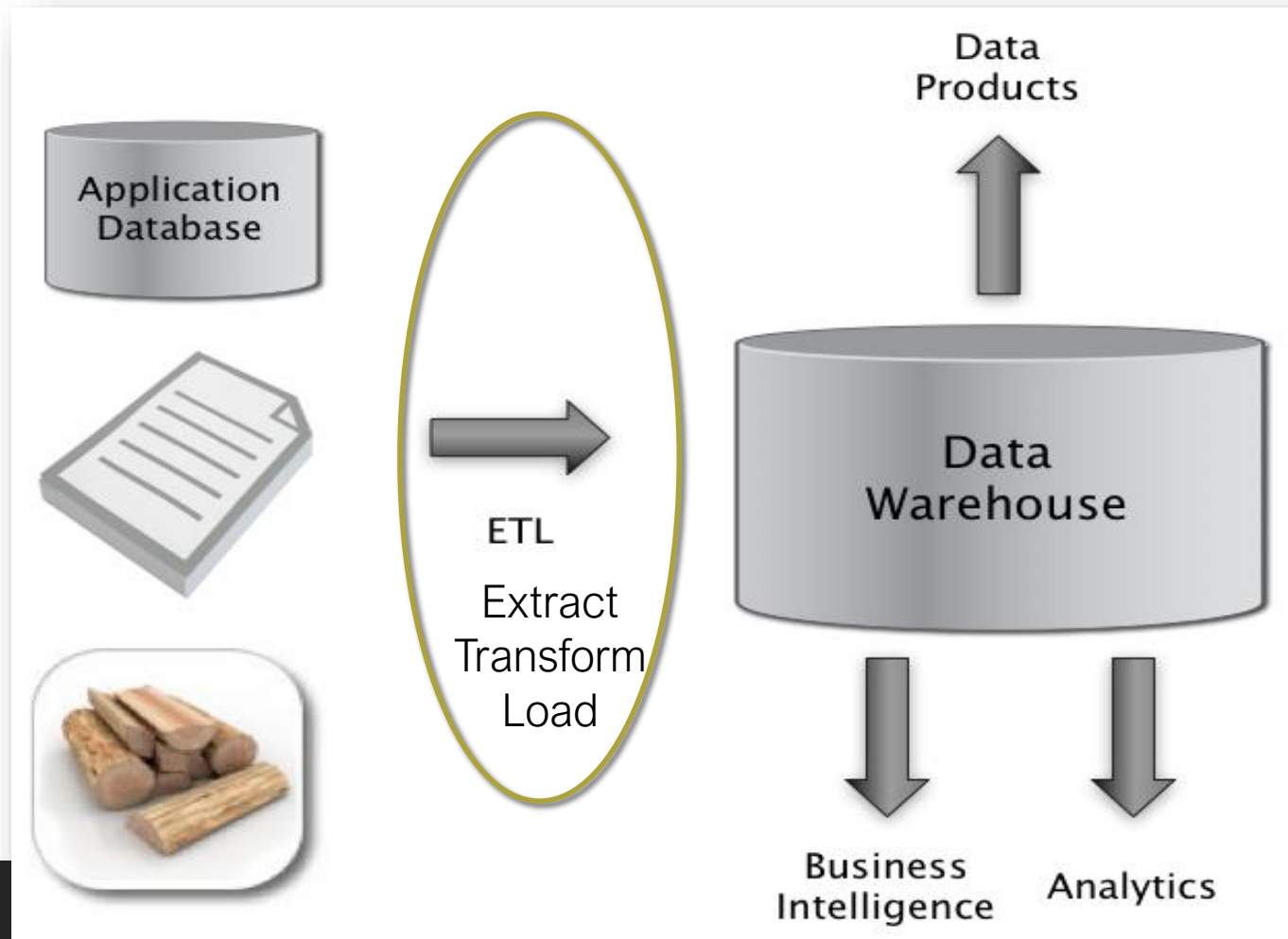
- **Internal sources:** already collected by or is part of the overall data collection of your organization.
 - For example: **business-centric data** available in the organization DB to record day to day operations; **scientific or experimental data** get from an essay.
- **Existing External Sources:** available in ready to read format from an outside source for free or for a fee.
 - For example: public government databases, stock market data, sports, COVID-19.
- **External Sources Requiring Collection Efforts:** available from external source but acquisition requires special processing.
 - For example: data appearing only in print form, or data on websites.

Ways to gather online data

How to get data generated, published or hosted online?

- **API (Application Programming Interface):** Using a pre-builtin set of functions developed by a company to access their services. Often pay to use.
 - For example: Google Map API, Facebook API, Twitter API
- **RSS (Rich Site Summary):** summarizes frequently updated online content in standard format. Free to read if the site has one.
 - For example: news-related sites, blogs
- **Web scraping (crawling):** using software, scripts or by-hand extracting data from what is displayed on a page or what is contained in the HTML file (often in tables).

Good old days...



Web scraping

- Why do it?
 - Older government or smaller news sites might not have APIs for accessing data
 - Publish RSS feeds or have databases for download.
 - You don't have \$\$ to pay to use the API or API provided is rate limited.
 - Monitor a news site for trending new stories on a particular topic of interest
 - Do social network analytics using profile data found on a web forum.

Web scraping

- **Who should you do it?**
 - You just want to explore:
 - Are you violating their terms of service?
 - Privacy concerns for website and their clients?
 - You want to publish your analysis or product:
 - Do they have an API or fee that you are bypassing?
 - Are they willing to share this data?
 - Are you violating their terms of service?
 - Are there privacy concerns?
- **How do you do it?**

Web scraping

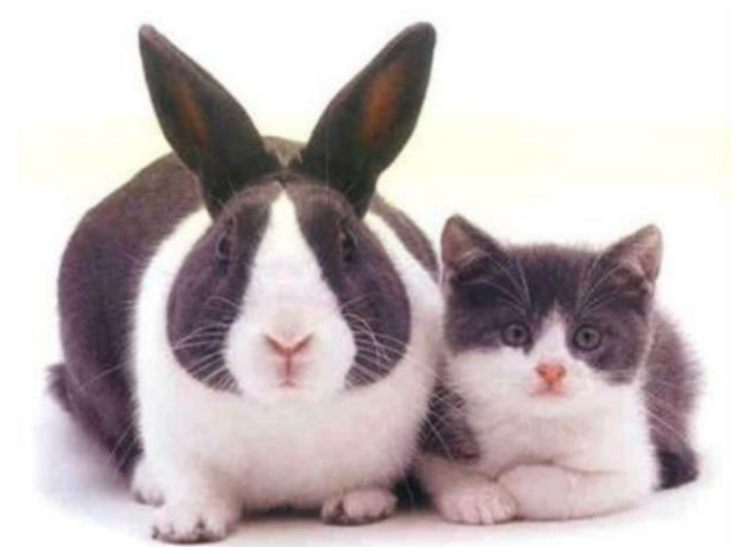
- Using python (or R) programs to get data from online
- Often much faster than manually copying data!
- Transfer the data into a form that is compatible with your code
- Legal and moral issues



Warning: Web scraping

Tips:

- Vast source of information; can combine with multiple datasets
- Automate repetitive tasks
- Keep up with sites / real-time data
- Be careful and polite (don't hit a server too often)
- Be Robust and immune to spider traps and other malicious behavior from web servers
- Give proper credit!
- Care about media law / obey licenses / privacy
- Do not be evil (no spam, overloading sites, etc)
- Do not forget data provenance!



It is Fun!

Obtaining Data: Web scraping

Robots.txt

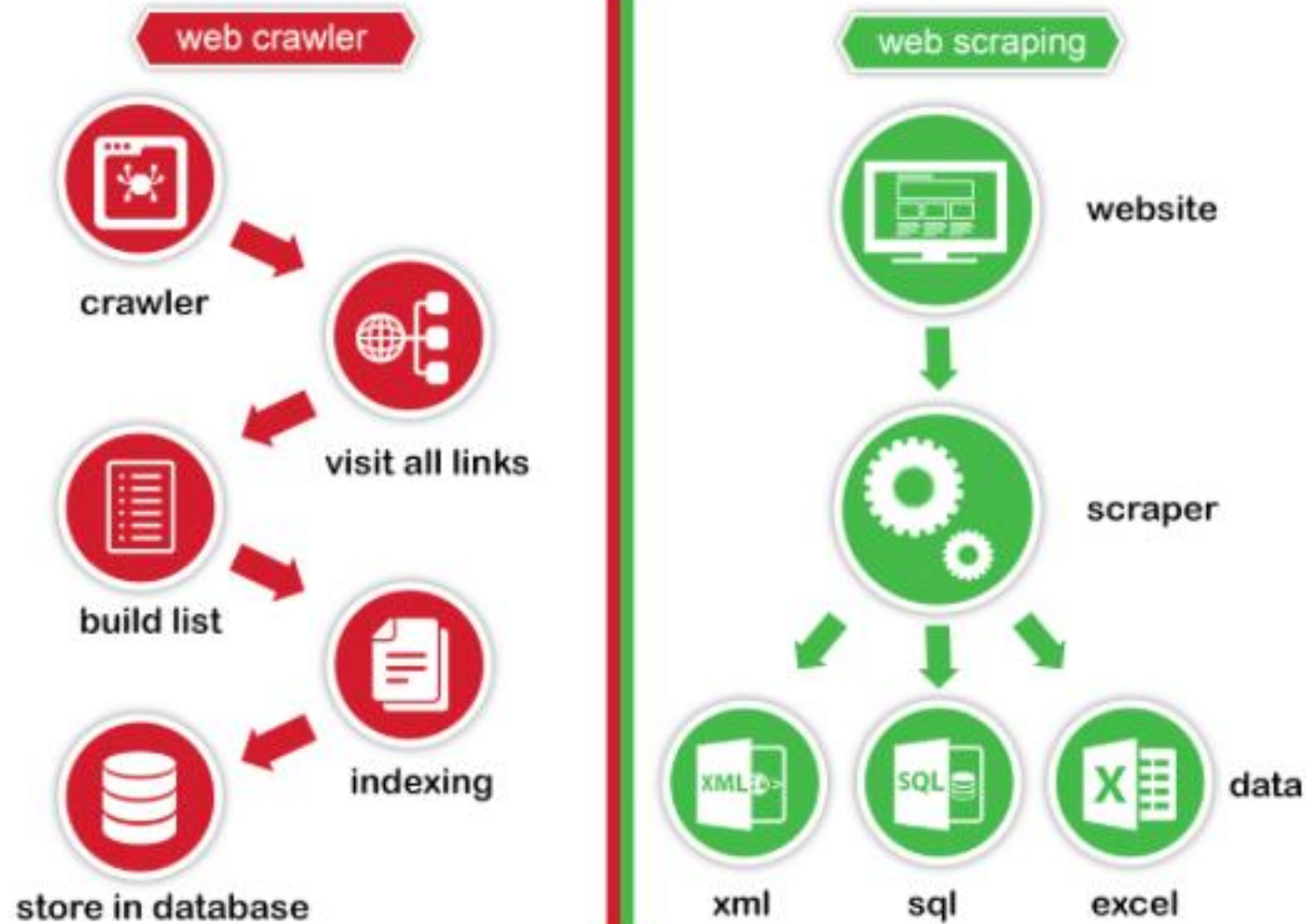
Protocol for giving spiders (“robots”) limited access to a website, originally from 1994

www.robotstxt.org/wc/norobots.html

Website announces its request on what can(not) be crawled

- Specified (access restrictions) by web site owner
- Gives instructions to web robots (e.g., your code)
- Located at the top-level directory of the web server
- E.g., <http://google.com/robots.txt>

Web Crawler x web scraper







Hands on...

NOTEBOOK:

BEAUTIFULSOAP

Remember....Web Servers

- A server maintains a long-running process (also called a daemon), which listens on a pre-specified port
- It responds to requests, which is sent using a protocol called HTTP (HTTPS is secure)
- Our browser sends these requests and downloads the content, then displays it.
- Examples: request was successful, client error, often `page not found`; server error (often that your request was incorrectly formed)

Remember.... HTML

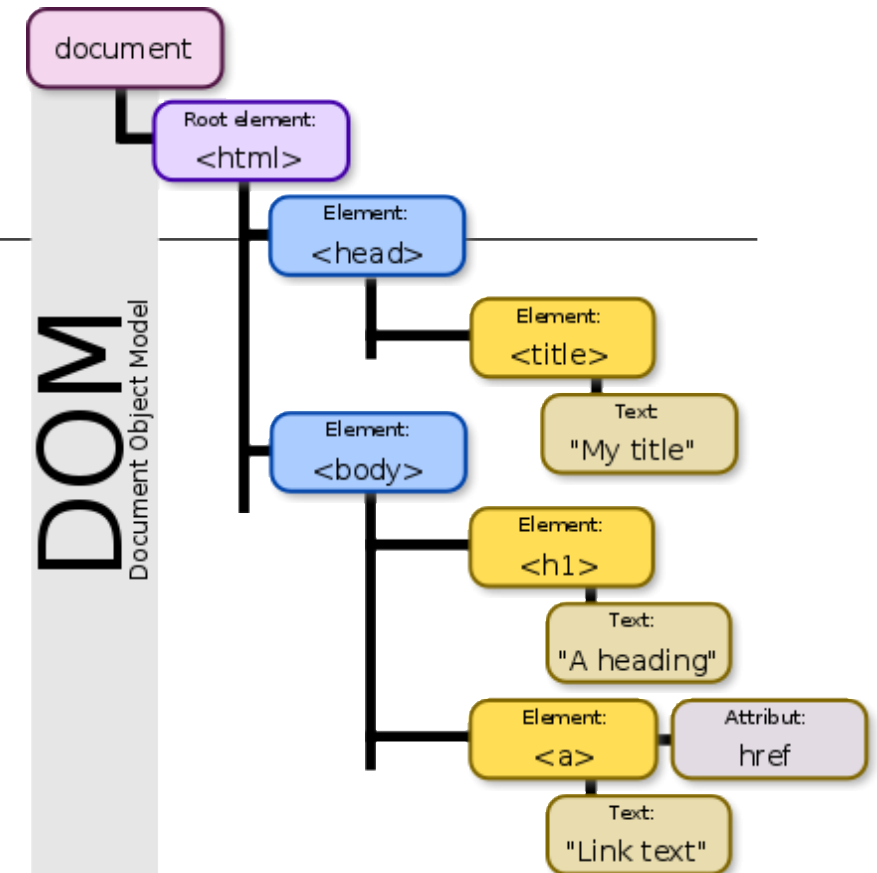
- Tags are denoted by angled brackets
- Almost all tags are in pairs e.g.,
`<p>Hello</p>`
- Some tags do not have a closing tag e.g., `
`

Example

```
<!DOCTYPE html>
<html>
  <head>
    <title>Title</title>
  </head>
  <body>
    <h1>Body Title</h1>
    <p>Body Content</p>
  </body>
</html>
```

Remember.... HTML

- **<html>**, indicates the start of an html page
- **<body>**, contains the items on the actual webpage (text, links, images, etc)
- **<p>**, the paragraph tag. Can contain text and links
- **<a>**, the link tag. Contains a link url, and possibly a description of the link
- **<input>**, a form input tag. Used for text boxes, and other user input
- **<form>**, a form start tag, to indicate the start of a form
- ****, an image tag containing the link to an image



How to Web scrape:

1. **Get** the webpage content

- **Requests** (Python library) **gets** a webpage for you

2. **Parse** the webpage content

- (e.g., find all the text or all the links on a page)
- **BeautifulSoup** (Python library) helps you **parse** the webpage.
- Documentation: <http://crummy.com/software/BeautifulSoup>

The Big Picture Recap



BeautifulSoup only concerns webpage data

1. **Get** the webpage content

Requests (Python library) **gets** a webpage for you

```
page = requests.get(url)
page.status_code
page.content
```

1. Get the webpage content

Requests (Python library) **gets** a webpage for you

```
page = requests.get(url)
page.status_code
page.content
```

Gets the status from the
webpage request.


200 means success.

404 means page not found.

1. **Get** the webpage content

Requests (Python library) **gets** a webpage for you!

```
page = requests.get(url)
page.status_code
page.content
```



Returns the content of the response, in bytes.

2. Parse the webpage content

BeautifulSoup (Python library) helps you **parse a webpage** and, makes messy HTML digestible

```
soup = BeautifulSoup(page.content, "html.parser")  
soup.title  
soup.title.text
```

2. Parse the webpage content

BeautifulSoup (Python library) helps you **parse** a webpage

```
soup = BeautifulSoup(page.content, "html.parser")
```

```
soup.title
```

```
soup.title.text
```

Returns the full context, including the title tag.

e.g.,

```
<title> UFRJ </title>
```

2. Parse the webpage content

BeautifulSoup (Python library) helps you **parse** a webpage

```
soup = BeautifulSoup(page.content, "html.parser")
```

```
soup.title
```

```
soup.title.text
```



Returns the text part of the title tag. e.g.,

UFRJ

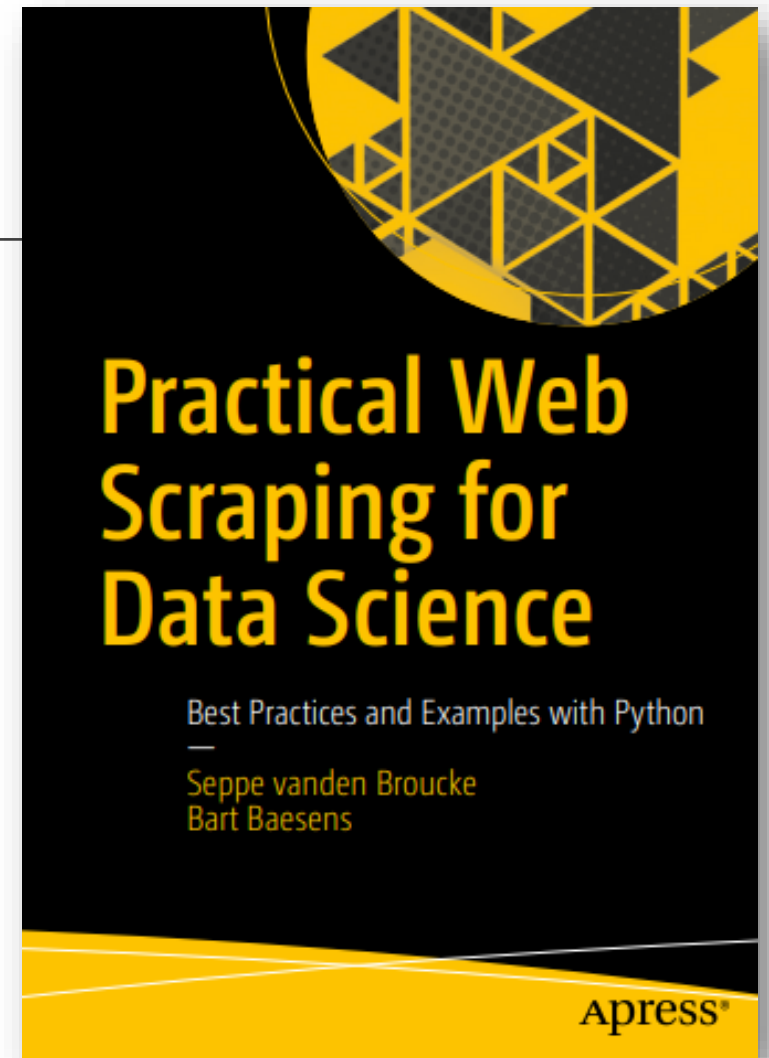
BeautifulSoup

- Provides functions for quickly accessing certain sections of HTML content

Example

```
import bs4
## get bs4 object
soup = bs4.BeautifulSoup(source)
## all a tags
soup.findAll('a')
## first a
soup.find('a')
## get all links in the page
link_list = [l.get('href') for l in soup.findAll('a')]
```


References



[Beautiful Soup Documentation — Beautiful Soup 4.9.0 documentation \(crummy.com\)](#)



Introduction to Data Science

MODULE II – PART II

Data Cleaning & Exploration

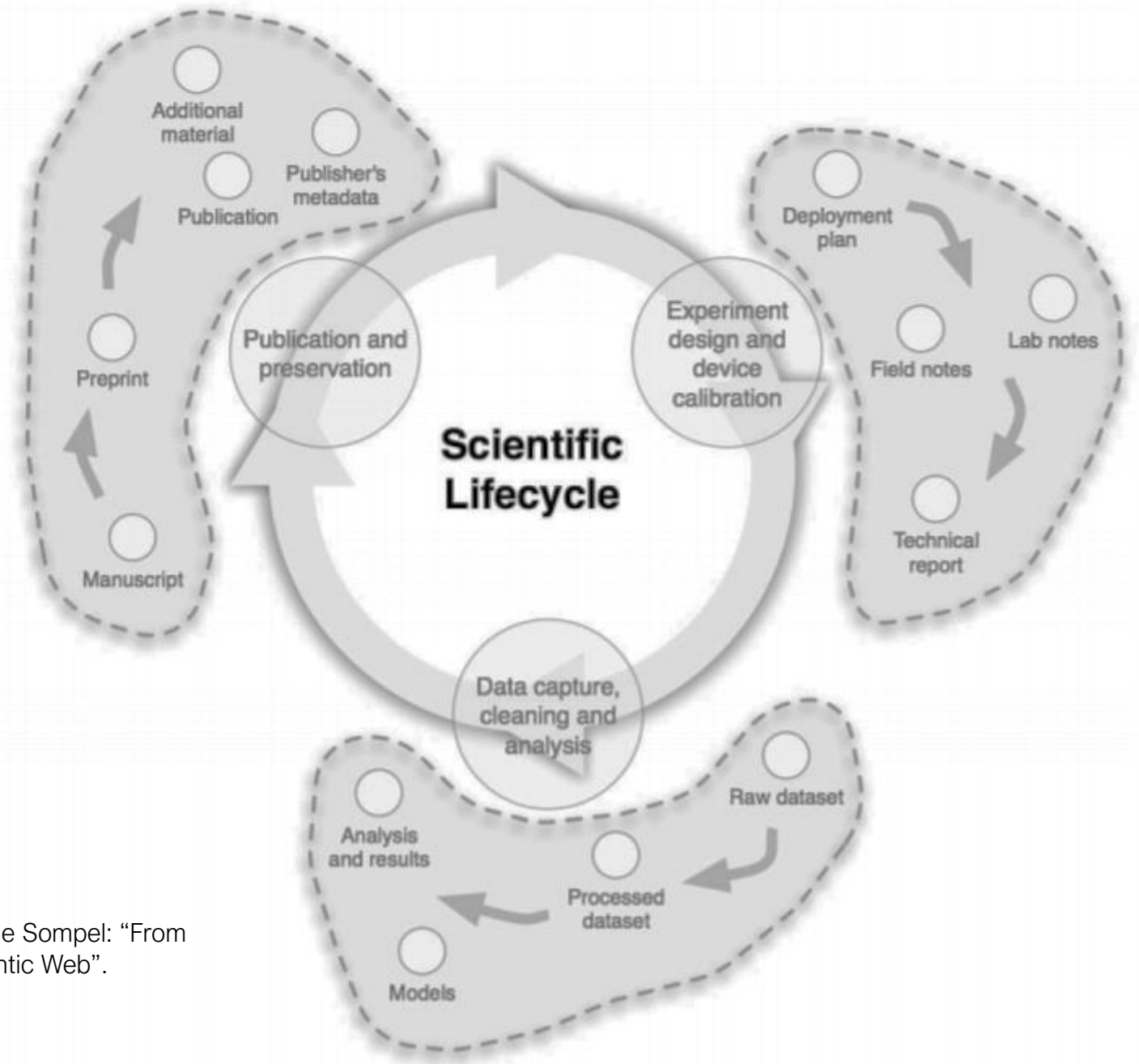
Prof Sergio Serra e Jorge Zavaleta

Scientific lifecycle

1

Very complex !

Alberto Pepe, Matthew Mayernik, Christine L. Borgman, Herbert Van de Sompel: "From Artifacts to Aggregations: Modeling Scientific Life Cycles on the Semantic Web".
<https://arxiv.org/ftp/arxiv/papers/0906/0906.2549.pdf>



Data lifecycle (Data Science)

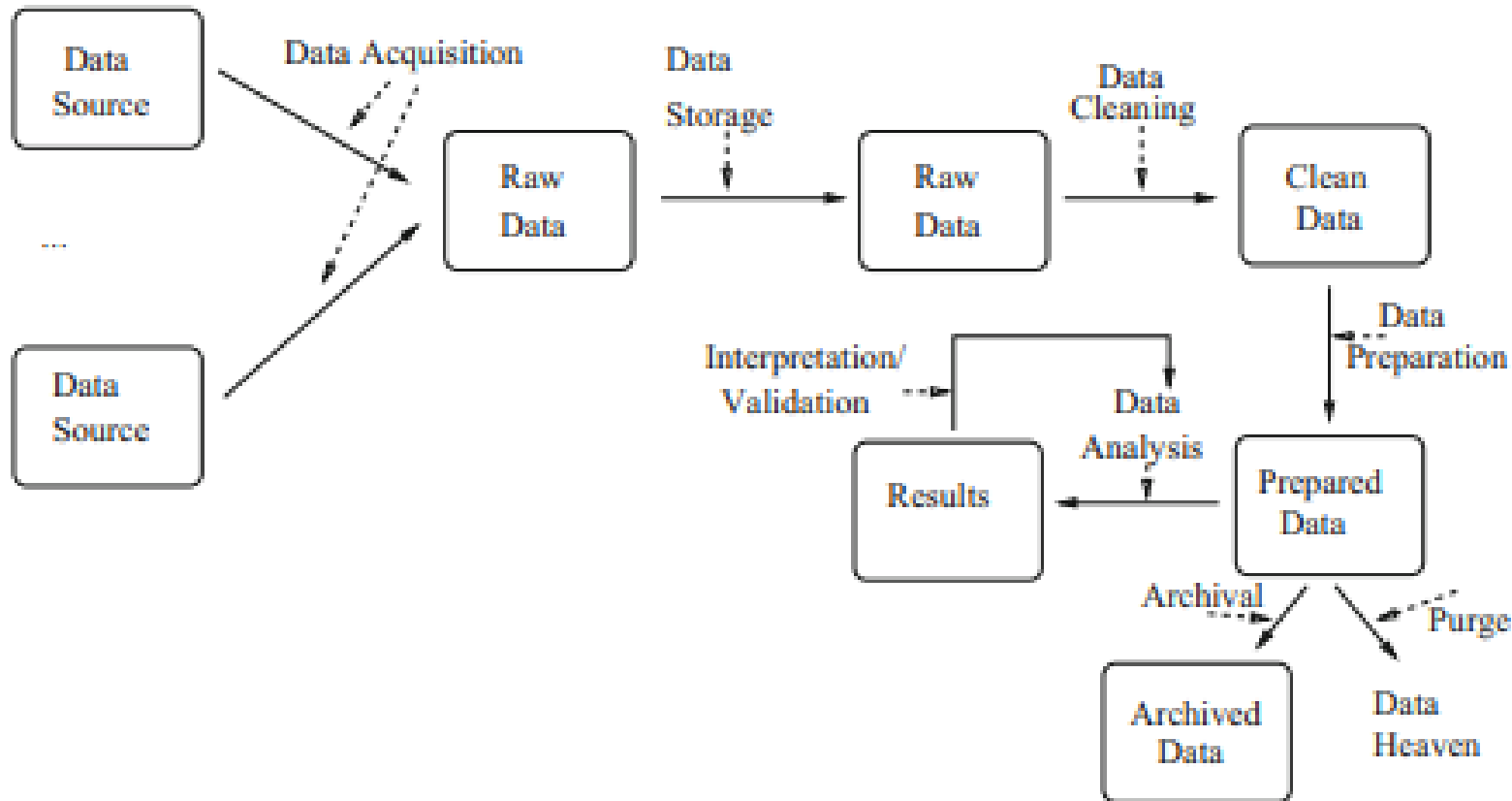
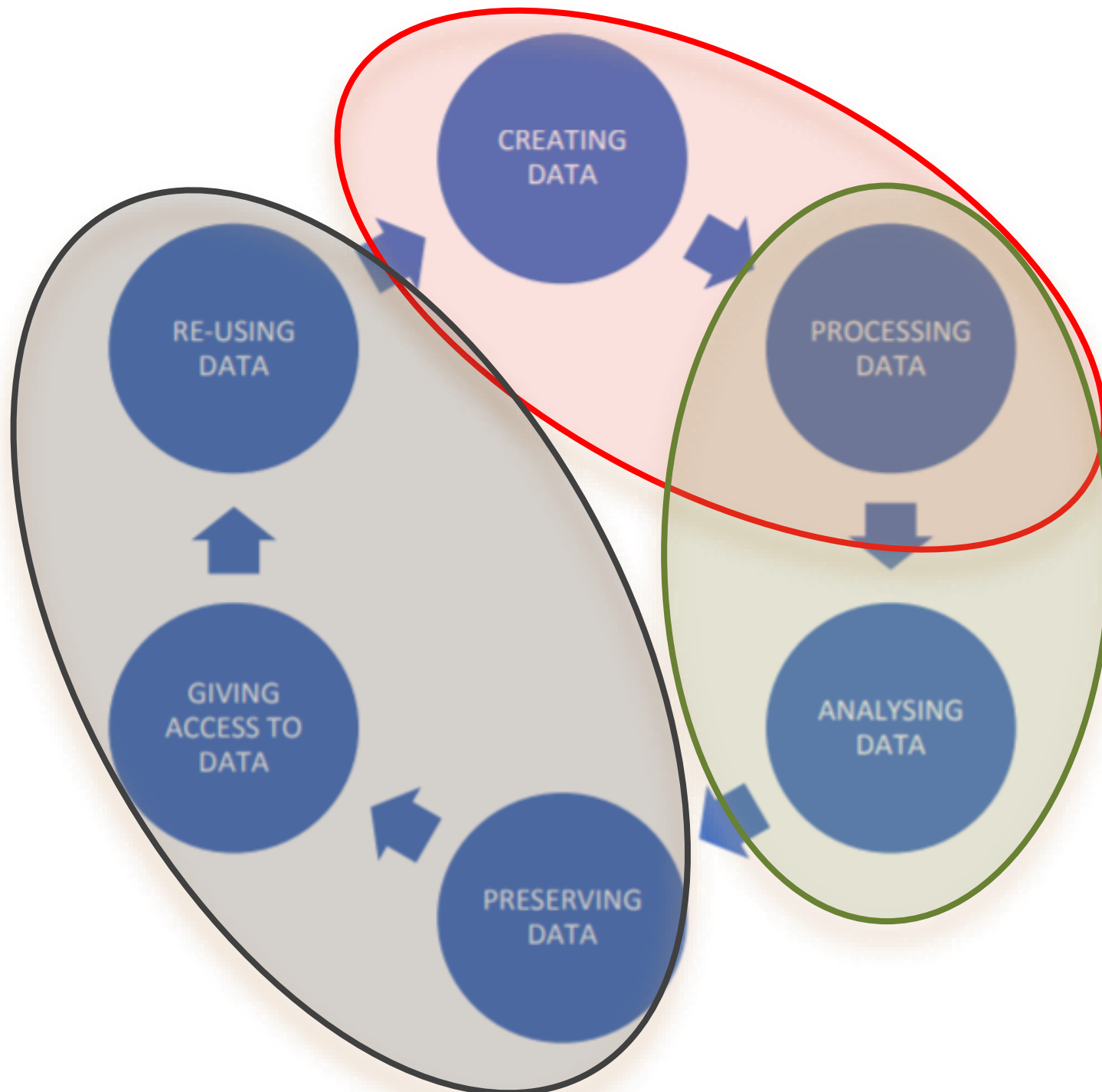


Fig. 1.1 The data life cycle

Badia, A, SQL for Data Science

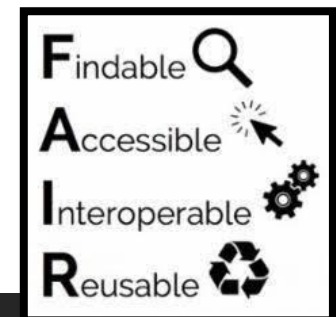
2

Very Poor !



1+2

Very FAIR !



What? The Data Science Process

Ask an interesting question
& learn reproducibility

Get the Data

Explore the Data

Model the Data

Communicate/Visualize the Results

Plot the data.

Are there anomalies or egregious issues?

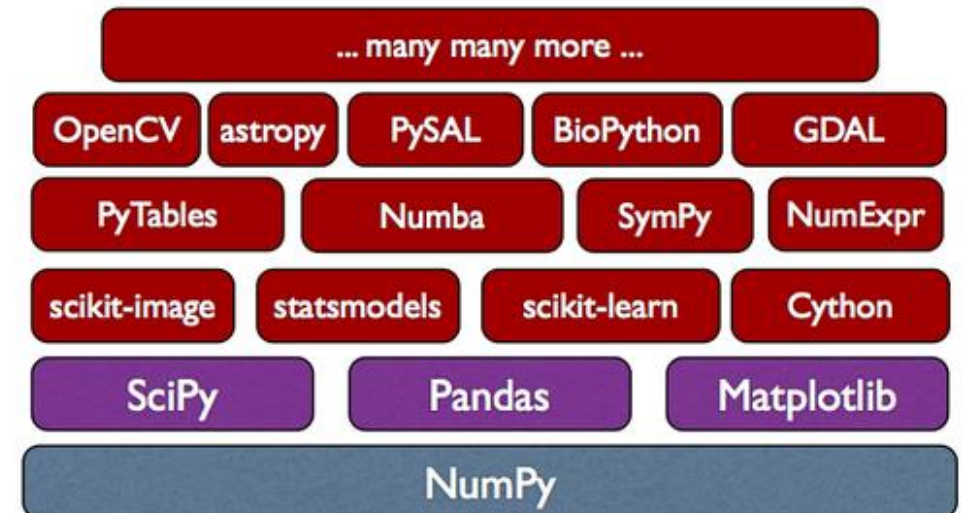
Are there patterns?

Modules II, III and IV

Data Cleaning Techniques

Data cleaning or cleansing is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set, table, or database.

- Refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data.
- Missing Data
- Irregular Data (Outliers)
- Unnecessary Data — Repetitive Data, Duplicates and more
- Inconsistent Data — Capitalization, Addresses and more



Store and Explore Data

Why Pandas?

- Used by a lot of people
- Pandas is a fast, powerful, flexible and easy to use open-source **data analysis** and **manipulation tool**, built on top of the Python programming language.
- Allows for high-performance, easy-to-use data structures and data analysis
- Unlike NumPy library which provides multi-dimensional arrays,
- Pandas provides **1D table object** called **Series**
- Pandas provides **2D table object** called **DataFrame** (akin to a spreadsheet with column names and row labels).

Pandas

Series: a named, ordered dictionary

- The keys of the dictionary are the **indexes**
- Built on NumPy's **ndarray**
- Values can be any Numpy data type object

DataFrame: a table with named columns

- Represented as a Dict (col_name -> series)
- Each Series object represents a column

Series

	apples
0	3
1	2
2	0
3	1

+

Series

	oranges
0	0
1	3
2	7
3	2

=

DataFrame

	apples	oranges
0	3	0
1	2	3
2	0	7
3	1	2

DataFrame

row

column



Applications of Pandas



Python Pandas Features

Multiple file
formats supported

Python
support

Input and
output tools

Optimized
performance

Great handling
of data

Handling
missing data

Grouping

Lot of time
series

Unique data

Cleaning
up data

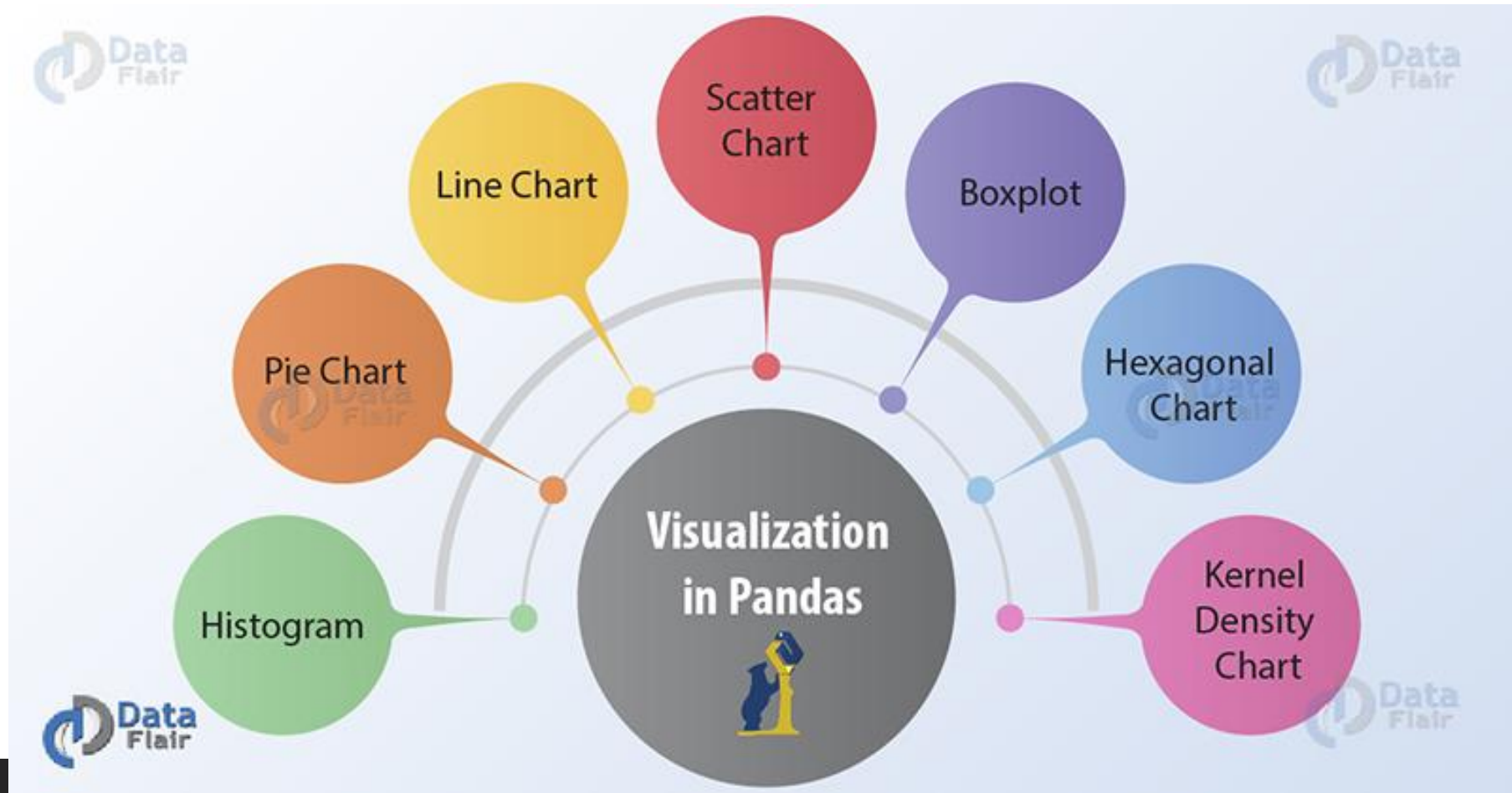
Alignment
and indexing

Merging and
joining of datasets

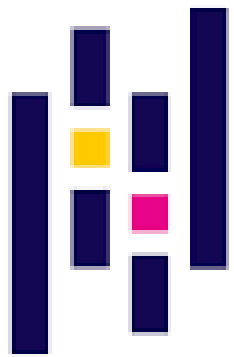
Visualize

Mask data

Visualization







pandas

Hands on...

NOTEBOOK:

PANDAS

A solid dark grey horizontal bar at the bottom of the slide.