
# MACHINE LEARNING

## Preparation Data

# Magister Teknik Informatika

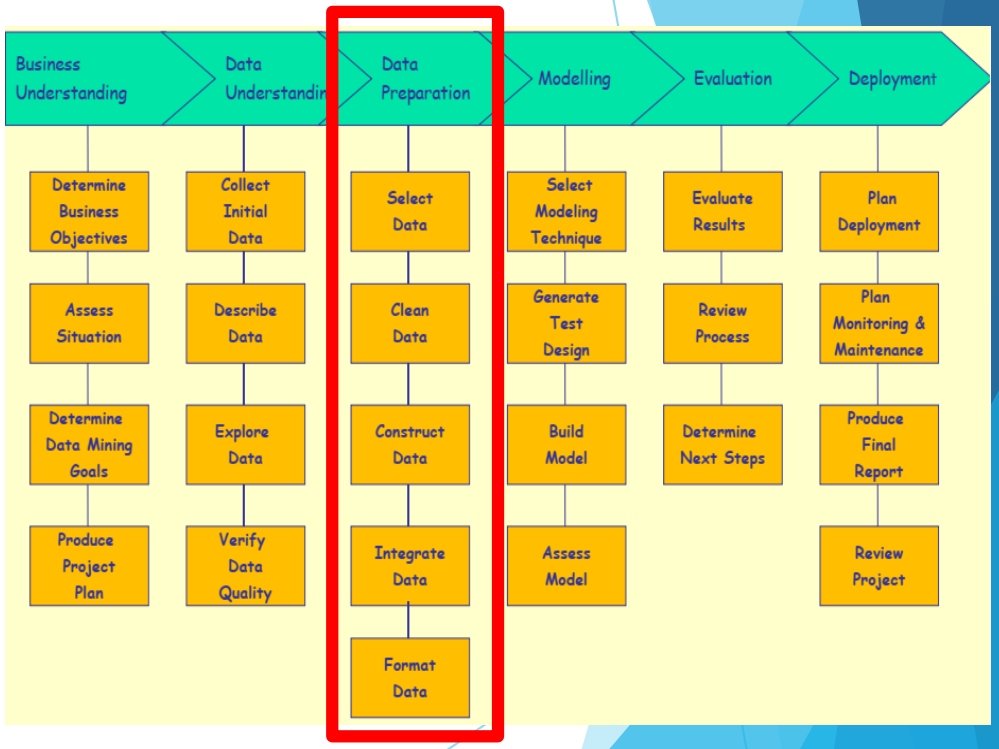
Dr. Chairani, S.Kom., M.Eng
IIB DARMAJAYA, 2023/2024

# Subject

- Bagian Pertama dari Tiga, materi Data Preparation
- Berfokus pada Penentuan Objek Data dan Pembersihan Data
- Konteksnya yaitu:
  - strategi pembersihan data kotor (noise, bias, missing value, outlier, dll)
  - pengecekan kualitas dan tingkat kecukupan data
- Dilanjutkan dengan:
  - transformasi data (modul 9) dan
  - konstruksi data (modul 10)
- Pengetahuan dan pemahaman akan data preparation menjadi syarat mutlak utk mengahasilkan model prediksi yang optimal.

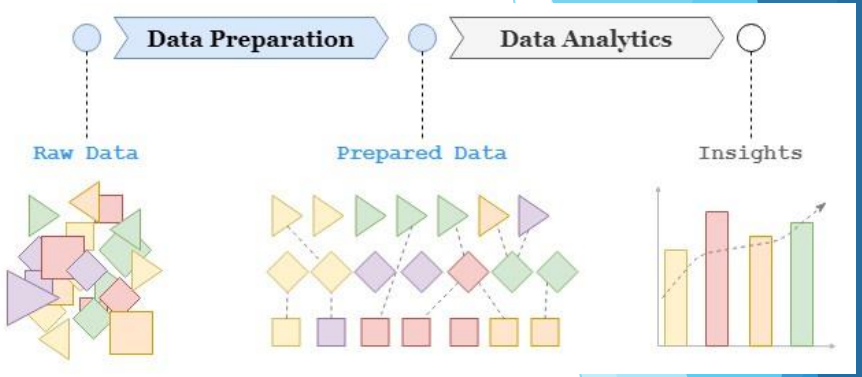
# Data Preparation dalam CRISP-DM

- Akronim dari: **CR**oss **I**ndustry **S**tandard **P**rocess **D**ata **M**ining
- Metodologi umum untuk data mining, analitik, dan proyek data sains, berfungsi menstandarkan proses data mining lintas industri
- Digunakan untuk semua level dari pemula hingga pakar



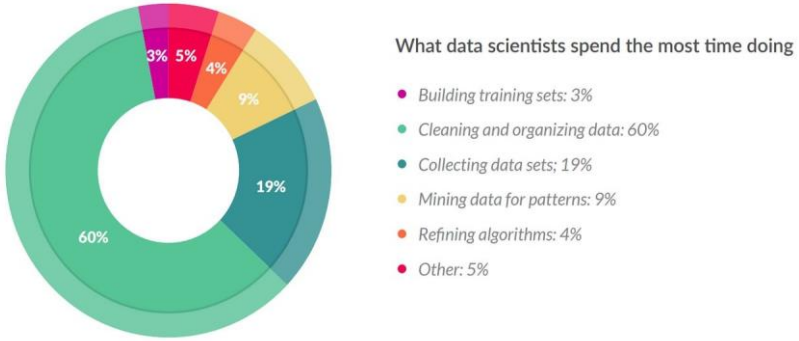
# Terminologi dan Definisi

- Istilah lain: Data Pre-processing, Data Manipulation, Data Cleansing/ Normalization
- De;nisi:
  - *transformasi data mentah menjadi format yang mudah dipahami*
  - menemukan data yang relevan untuk disertakan dalam aplikasi analitik shg memberikan informasi yang dicari oleh analis atau pengguna bisnis
  - *langkah pra-pemrosesan yang melibatkan pembersihan, transformasi, dan konsolidasi data.*

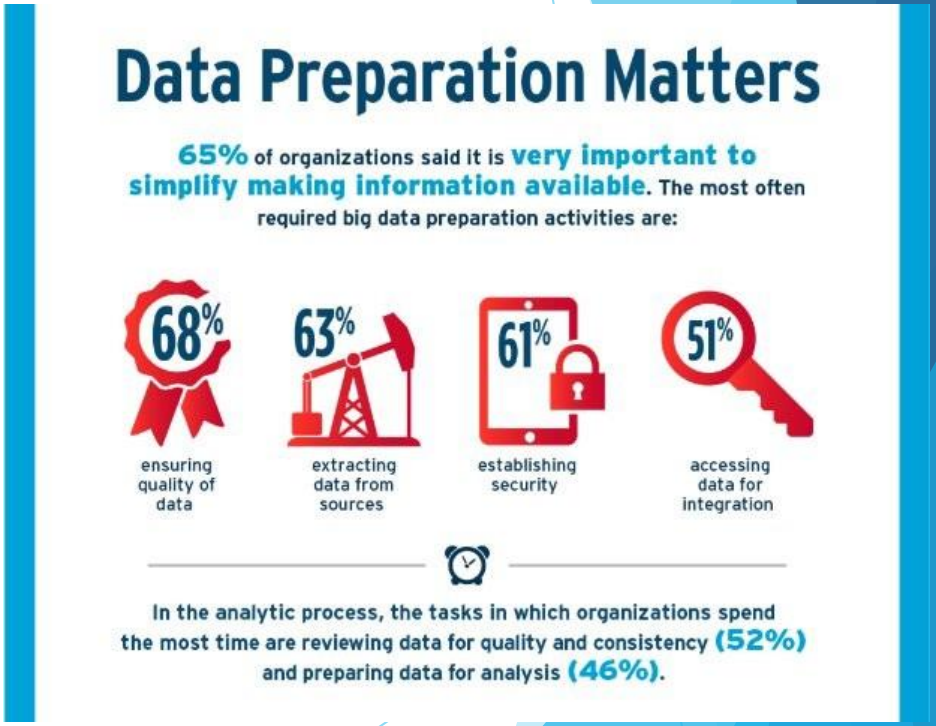



- Definisi:
  - proses yang melibatkan koneksi ke satu atau banyak sumber data yang berbeda, membersihkan data kotor, memformat ulang atau merestrukturisasi data, dan akhirnya menggabungkan data ini untuk digunakan untuk analisis.

# Fakta Terkait Data Preparation

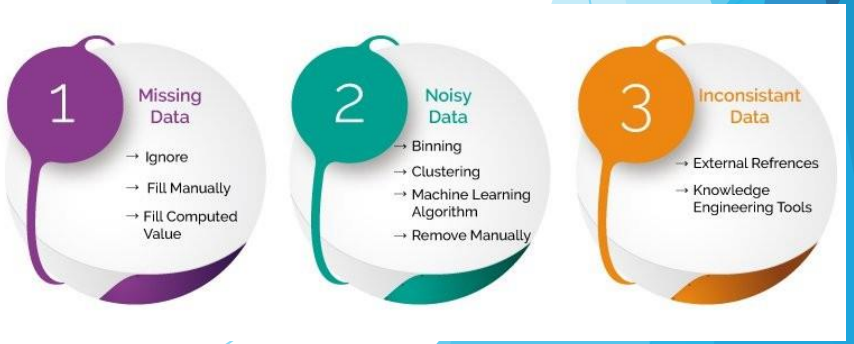



- 60-80% porsi kegiatan data saintis (forbes, crowdflower 2016)
  - data yang ada saat ini dari banyak sumber data dan format yang beragam (terstruktur, semi, dan tidak terstruktur)
  - kualitas model prediktif bergantung pada kualitas data (GIGO)



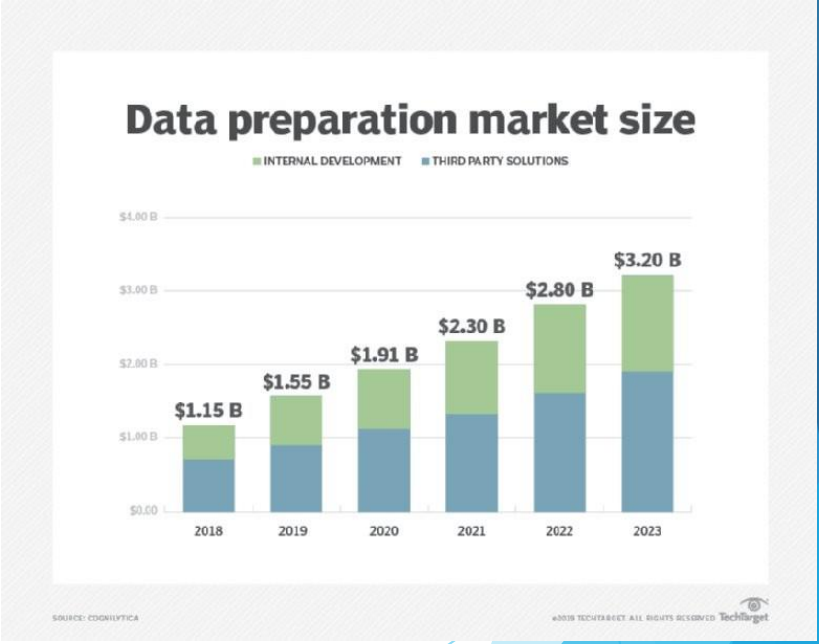
# Pentingnya Data Preparation

- data perlu diformat sesuai dengan software yang digunakan
- data perlu disesuaikan dengan metode data sain yang digunakan
- data real-world cenderung 'kotor':
  - tidak komplit: kurangnya nilai attribute, kurangnya atribut tertentu/penting, hanya berisi data agregate. misal: pekerjaan="" (tidak ada isian)
  - *noisy*: memiliki error atau outlier. misal: Gaji="-10", Usia="222"
- data real-world cenderung 'kotor':
  - tidak konsisten: memliki perbedaan dalam kode dan nama. misal : Usia= "32" TglLahir="03/07/2000"; rating "1,2,3" -- > rating "A, B, C"
- kolom dan baris yang saling bertukar
- banyak variabel dalam satu kolom yang sama

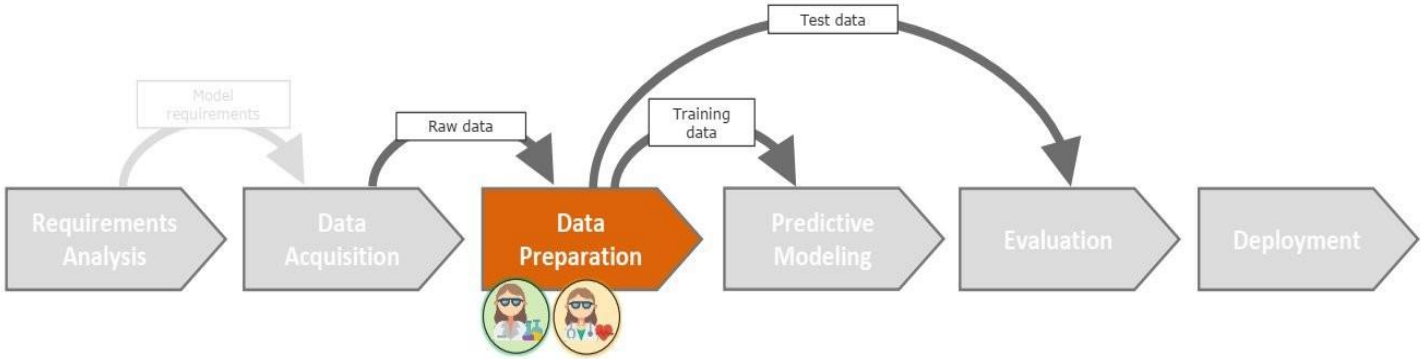


# Manfaat Data Preparation

- Kompilasi Data menjadi E;sien dan Efektif (menghindari duplikasi)
- Identi;kasi dan Memperbaiki Error
- Mudah Perubahan Secara Global
- Menghasilkan Informasi yang Akurat utk Pengambilan Keputusan
- Nilai Bisnis dan ROI (Return on Ivestment) akan Meningkat

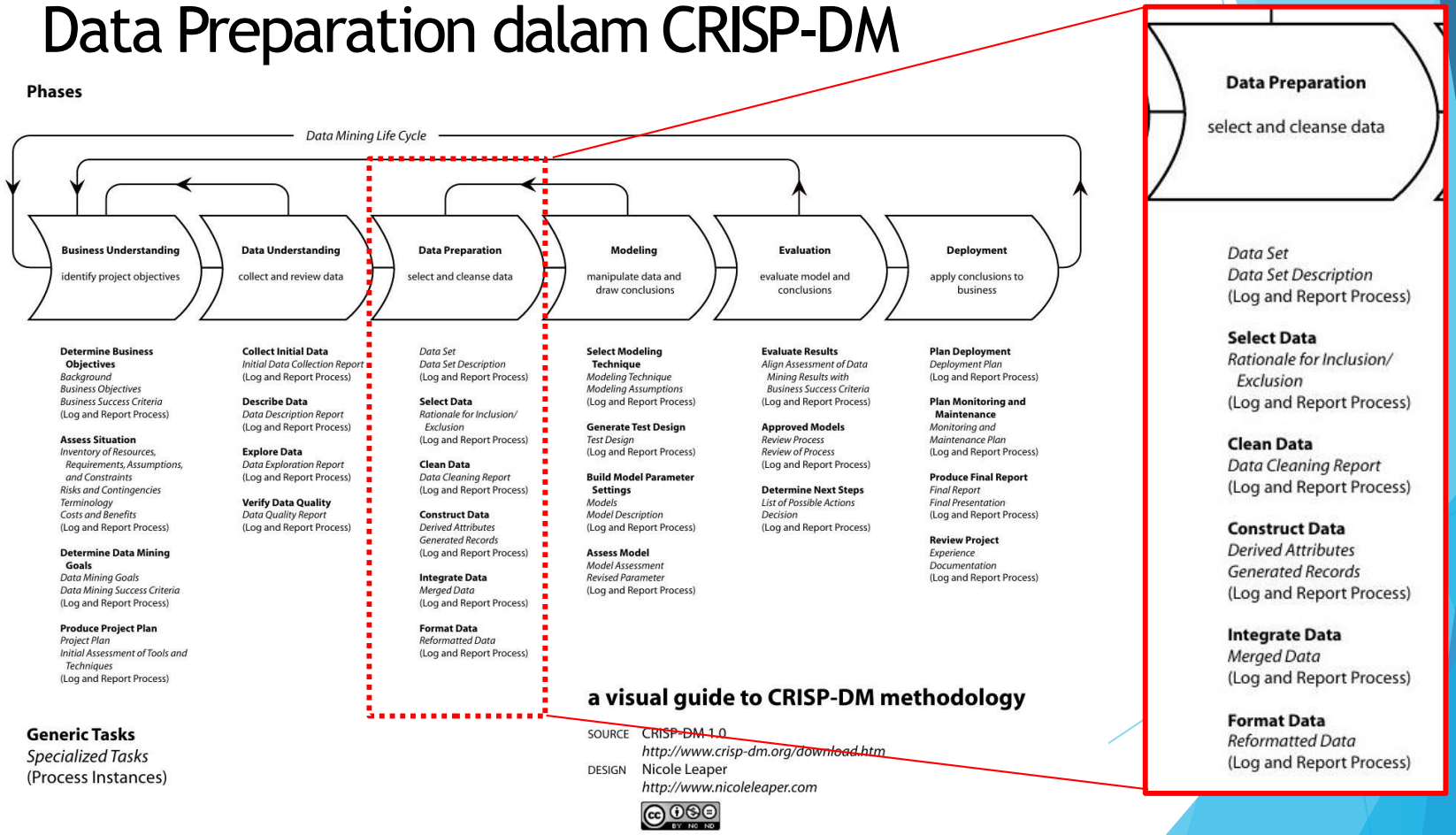


# Tahapan dan Tantangan Data Preparation




- **Memakan Waktu Lama**
- **Porsi Teknis yang Dominan**
- **Data yang Tersedia Tidak Akurat atau Jelas/Tidak Langsung Pakai**
- Data tidak Balance Saat Pengambilan Sampel
- Rentan akan Error

# Data Preparation dalam CRISP-DM



# Tahapan Data Preparation: Pemilihan, Pembersihan & Validasi

Modul 8

1. **Pilih/ Select Data**
   - Pertimbangkan apemilihan data
   - Tentukan dataset yang akan digunakan
   - Kumpulkan data tambahan yang sesuai (internal atau eksternal)
   - Pertimbangkan penggunaan teknik pengambilan sampel
   - Jelaskan mengapa data tertentu dimasukkan atau dikecualikan
2. **Bersihkan/ Clean Data**
   - Perbaiki, hapus atau abaikan noise
   - Putuskan bagaimana menangani nilai-nilai khusus dan maknanya
   - Tingkat agregasi, nilai yang hilang (missing value), dll
   - Bersihkan atau manipulasi outlier
3. **Validasi Data**
   - Periksa/Nilai Kualitas Data
   - Periksa/Nilai Tingkat Kecukupan Data

# Paramater/Daftar Isi Dokumentasi Data Cleaning

Laporan dokumentasi data cleaning, setidaknya memiliki parameter berikut:

- Data Set Description
- Data Set yang digunakan
- Jenis noise yang terjadi pada data (diantaranya: Missing data, Data errors; Coding inconsistencies; Missing/ bad metadata
- Pendekatan yang dilakukan untuk menghilangkan noise tersebut
- Teknik mana yang digunakan sehingga berhasil untuk menghilangkan noise tersebut
- Apakah ada kasus atau atribut yang tak dapat diselamatkan
- Pastikan data yang dikecualikan karena kondisi noisenya

# Paramater/Daftar Isi Dokumentasi Data Validation

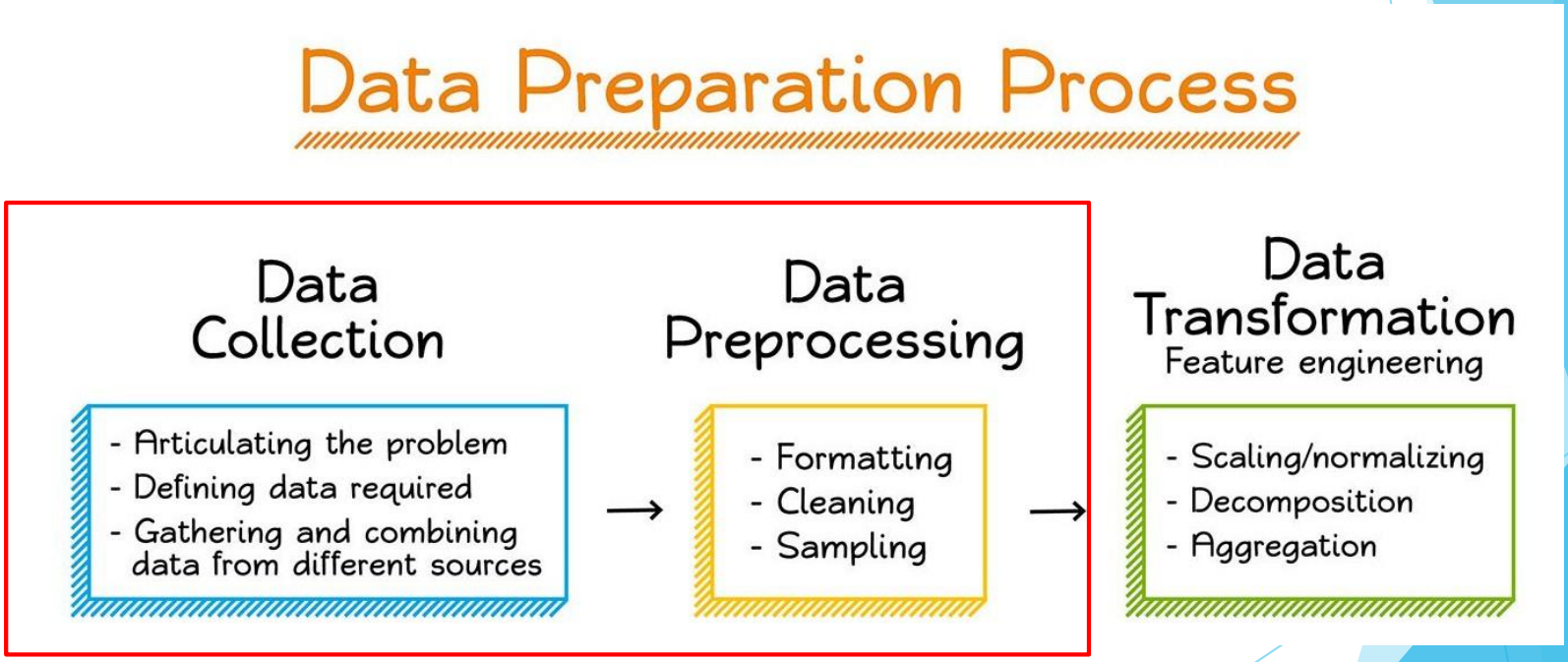
Laporan dokumentasi data cleaning, setidaknya memiliki parameter berikut:

- Validasi data
  - Kebenaran, misal di Indonesia isian Gender yang diakui hanya 2 P/W; Agama hanya 6 (Islam, Protestan, Katholik, Hindu, Budha, Konghucu)
  - Kelengkapan, misal data propinsi seluruh Indonesia (34 prov), namun hanya sebagian yg ada
  - Konsistensi, misal penulisan STM atau SMK;
- Kecukupan data → Perlukan diulang berikan justi;kasi (Resampling)

# Rincian Tahapan Data Preparation

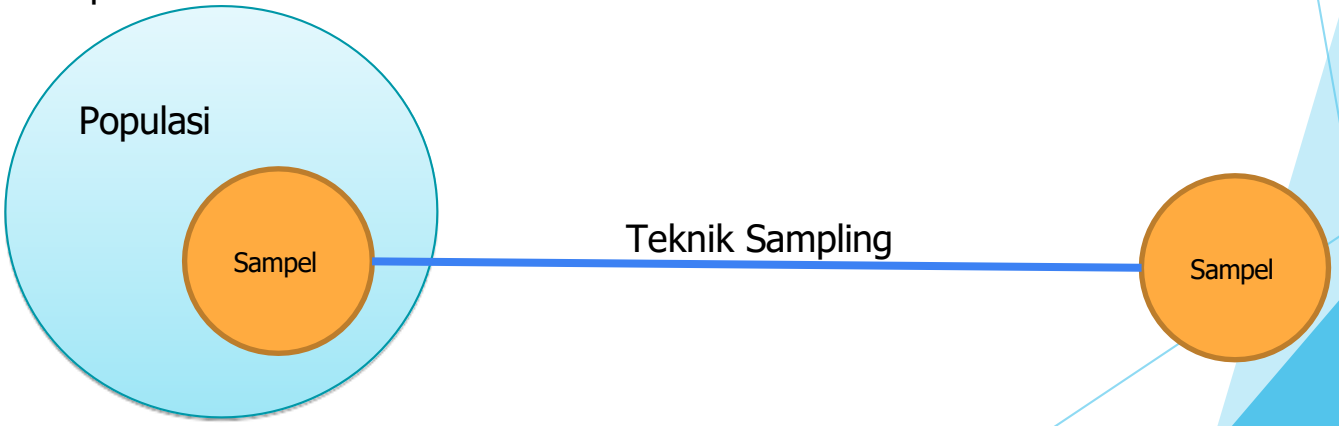
3. **Bangun/ Construct Data**
   - Atribut turunan.
   - Latar belakang pengetahuan.
   - Bagaimana atribut yang hilang dapat dibangun atau diperhitungkan
4. **Integrasi/ Integrate Data**
   - Mengintegrasikan sumber dan menyimpan hasil (tabel dan catatan baru)
5. **Bentuk/ Format Data**

# Tahapan Data Preparation: Versi Simple



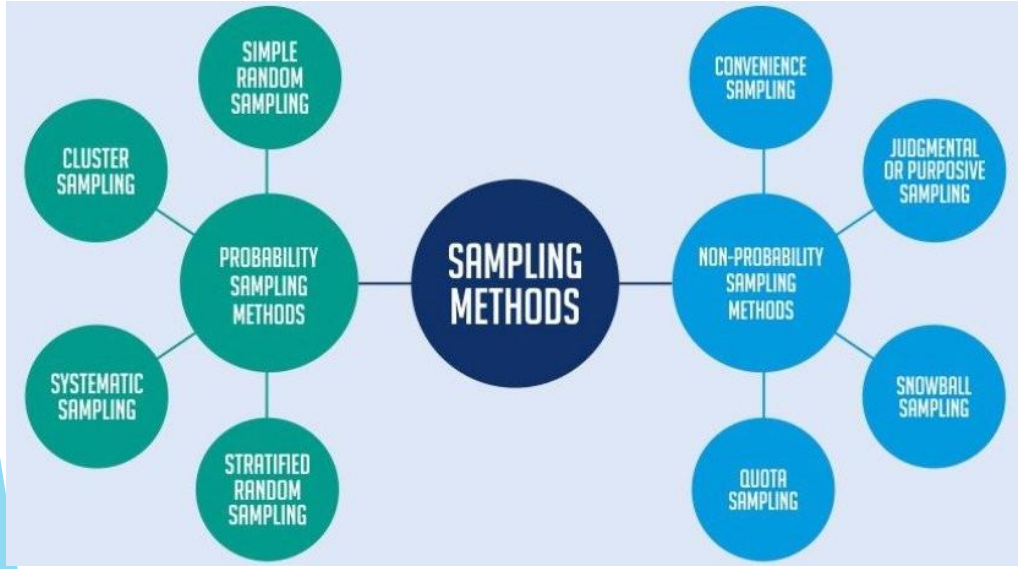
# Sampling Data: Pengertian Sampling

- Sebelum melakukan tahapan dalam data preparation, terlebih dahulu adalah pemilihan/penentuan objek yang dapat dilakukan dengan menggunakan penentuan:
  - Populasi
  - Sampel



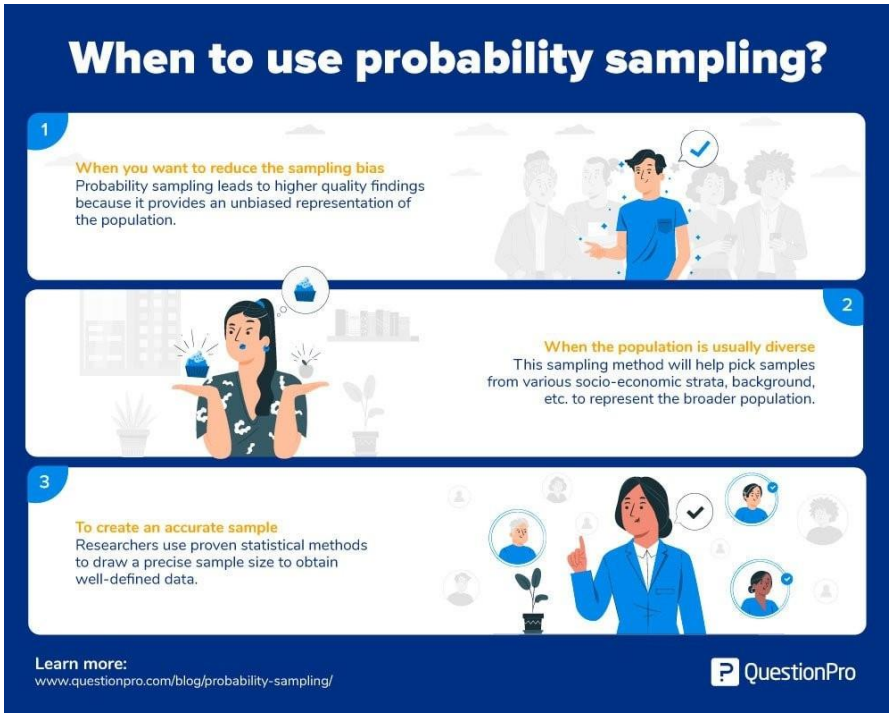
# Sampling Data: Metode Sampling

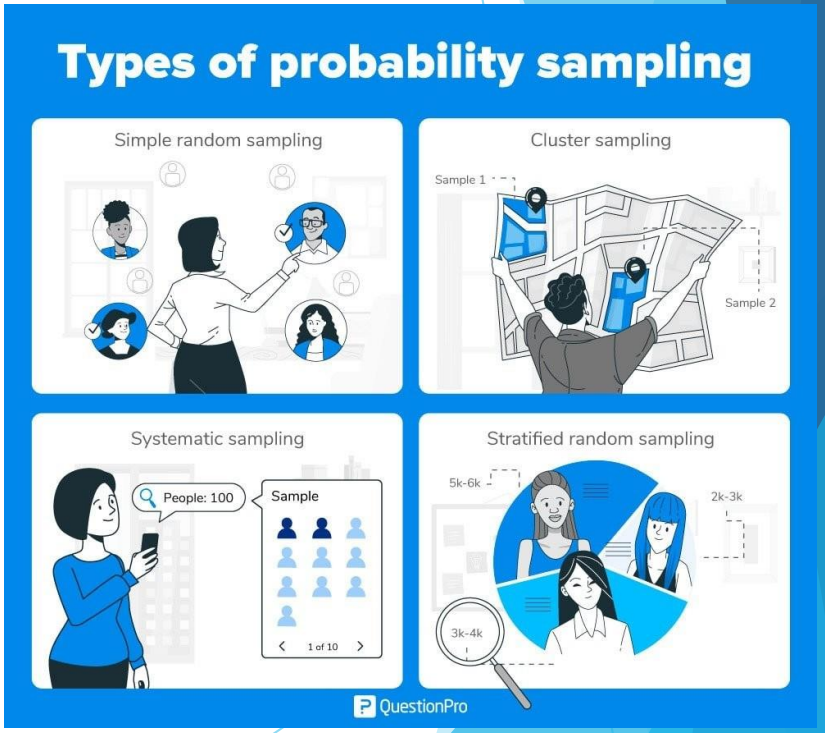
- Kategori Metode Sampling



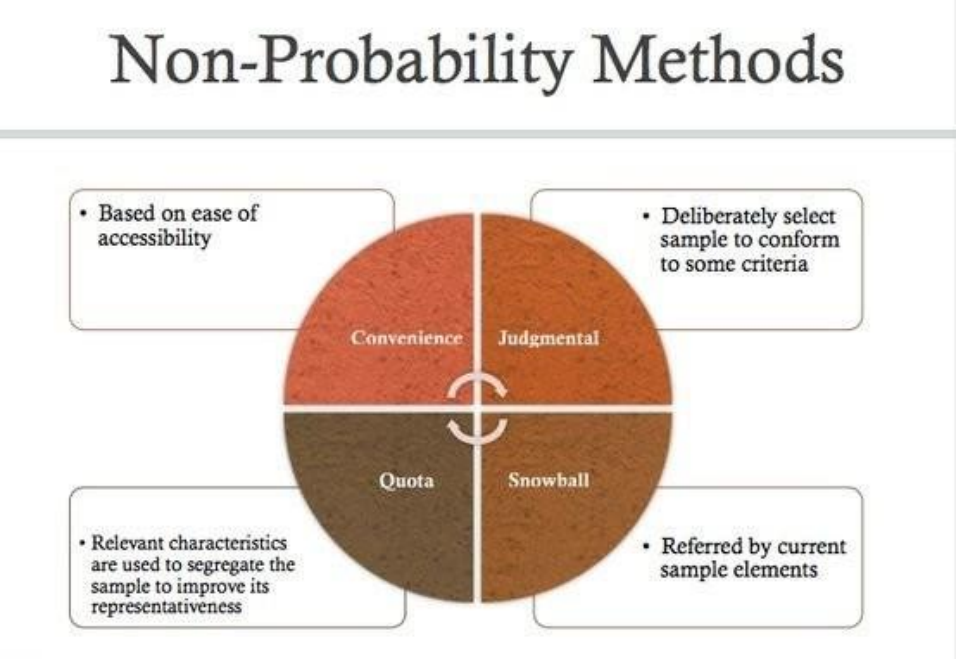

- Probability Sampling:
  - Populasi diketahui
  - Randomisasi/keteracakan: Ya
  - Conclusiver
  - Hasil: Unbiased
  - Kesimpulan: Statistik
- Non-Probability Samplimg
  - Populasi tidak diketahui
  - Keterbatasan penelitian
  - Randomisasi/keteracakan: Tidak
  - Exploratory
  - Hasil: Biased
  - Kesimpulan: Analitik

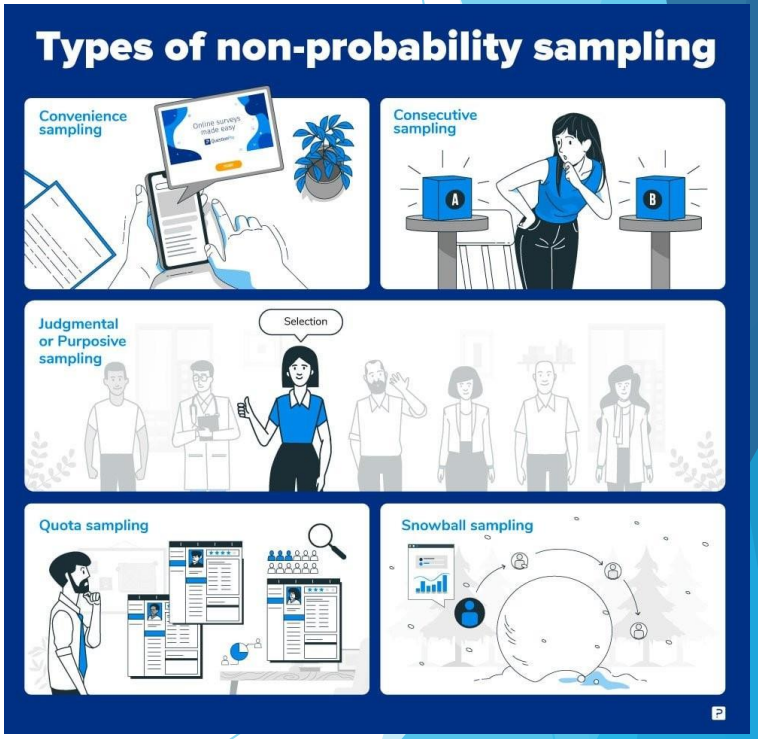
# Sampling Data: Metode Sampling



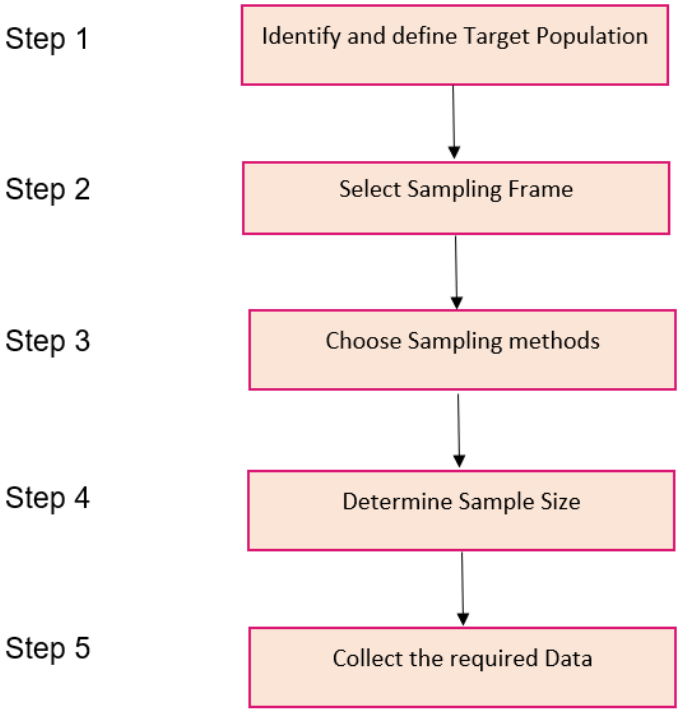



# Sampling Data: Teknik Sampling



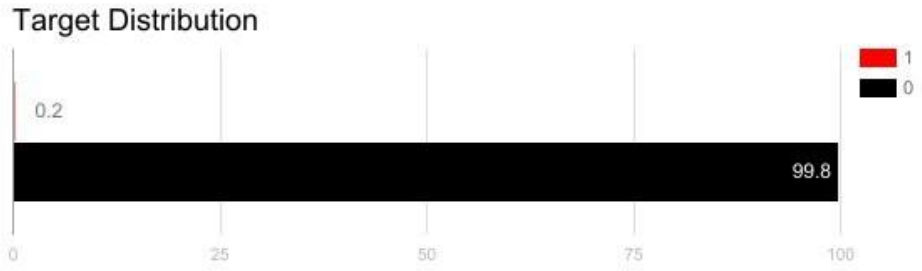



# Sampling Data: Tahapan Sampling



# Imbalance Dataset: Resampling

- Ini dilakukan setelah proses pemilihan, pembersihan dan rekayasa ;tur dilakuan atas pertanyaan:
  - Tanya: apakah kelas target data yang kita inginkan telah secara sama terdistribusi di seluruh dataset?
  - Jawab: Di banyak kasus tidak/belum tentu. Biasanya terjadi imbalance (ketidakseimbangan) antara dua kelas. Misal utk dataset tentang detekis fraud di perbankan, lelang real-time, atau deteksi intrusi di network! Biasanya data dari dataset tersebut berukuran sangat kecil atau kurang dari 1%, namun sangat signi;kan. Kebanyakan algoritma ML tidak bekerja baik utk dataset imbalance tsb.

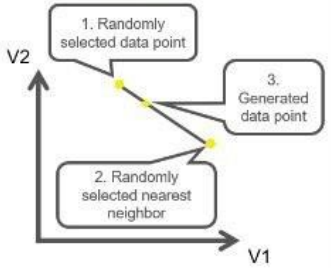
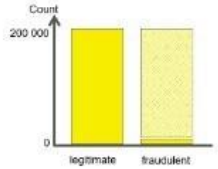
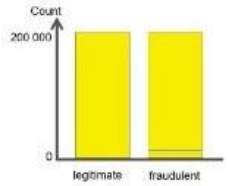
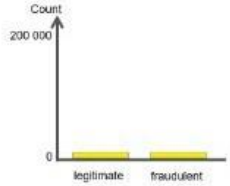


# Imbalance Dataset: Resampling

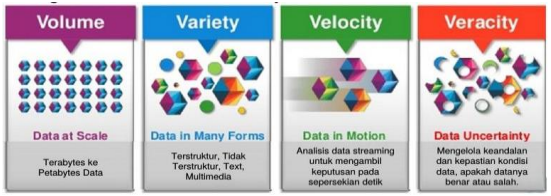
- Berikut adalah bbrp cara utk mengatasi imbalance dataset:
  - Gunakan pengukuran (metrik) yang tepat, misal dengan menggunakan:
    - **Precision/Spesikasi**: berapa banyak instance yang relevan
    - **Recall/Sensitifitas**: berapa banyak instance yang dipilih
    - **F1 score**: harmonisasi mean dari precision dan recall
    - **Matthews correlation coefficient (MCC)**: koe;sien korelasi antara klasi;kasi biner antara observasi vs prediksi
    - **Area under the ROC curve (AUC)**: relasi antara tingkat true-positive vs false-positive
  - Resample data training, dengan dua metode:
    - **Undersampling**: menyeimbangkan dataset dengan mereduksi ukuran kelas yang melimpah. Dilakukan jika kuantitas data mencukupi
    - **Oversampling**: Kebalikan dari undersampling, dilakukan jika kuantitas data tidak mencukupi

# Imbalance Dataset: Resampling

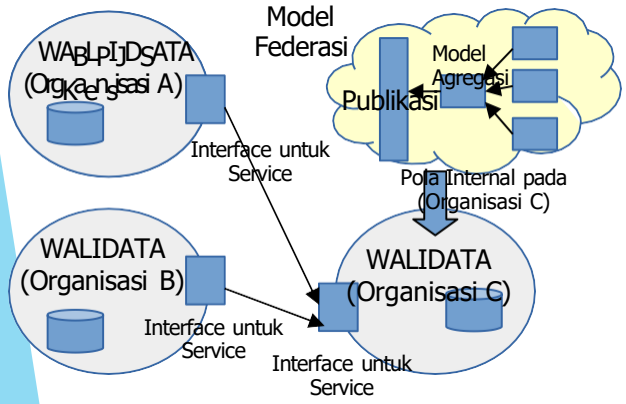
- Teknik Resampling:
  - oversampling (SMOTE)
  - oversampling (Bootstrap)
  - undersampling (Bootstrap)

| Resampling method | Description | Target class distribution after resampling |
|---|---|---|
| Oversampling (SMOTE) | Generate new synthetic fraudulent transactions until the number of fraudulent transactions is ca. equal to the number of legitimate transactions:<br>1. Select one of the fraudulent transactions in the training data randomly<br>2. Select one of its *n* nearest neighbors in the same fraudulent class randomly<br>3. Select a random point between the existing fraudulent transaction and its nearest neighbor<br>  | • Original data in yellow<br>• New synthetic data in light patterned yellow<br>  |
| Oversampling (Bootstrap) | Randomly draw with replacement a sample of fraudulent transactions until the number of fraudulent transactions is ca equal to the number of legitimate transactions |  |
| Undersampling (Bootstrap) | Randomly draw with replacement as many legitimate transactions as there are fraudulent transactions |  |

# Keberadaan data dari berbagai sumber (stakeholder)

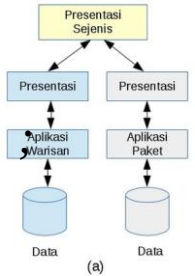



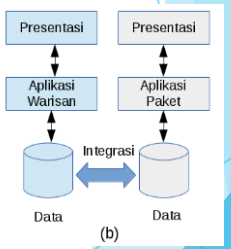

**GIGO**

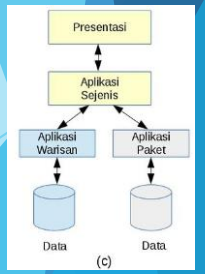


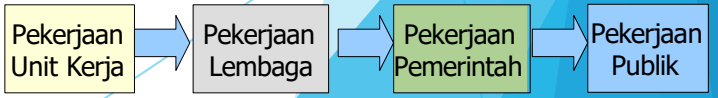

- Sistem informasi di masing-masing organisasi tidak bisa bertukar data/informasi pada lingkungan heterogen
- Interoperabilitas data akan menge;sienkan kerja serta dapat melakukan **prediksi dan analisis berbasiskan AI**
- Mendukung knowledge discovery dan decision making

- **Integrasi Presentasi.** User interface yang menyediakan akses pada suatu aplikasi. kinerja, persepsi, dan tidak adanya interkoneksi antara aplikasi dan data.
- **Integrasi Data**. Dilakukan langsung pada basis data atau struktur data. Jika terjadi perubahan model data, maka integrasinya perlu direvisi atau dilakukan ulang.
- **Integrasi Fungsional** Proses integrasi dilakukan pada level logika bisnis pada beberapa aplikasi.



# Kelengkapan Data sesuai Tujuan

Data Perencanaan

Data Pelaksanaan

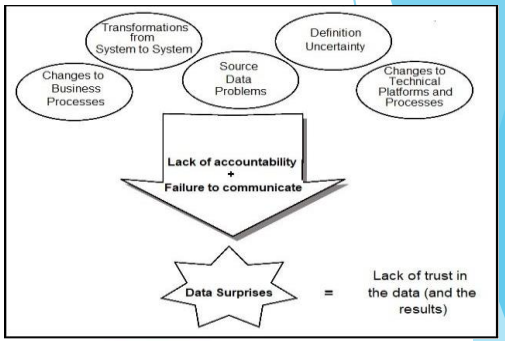
Data Pengawasan

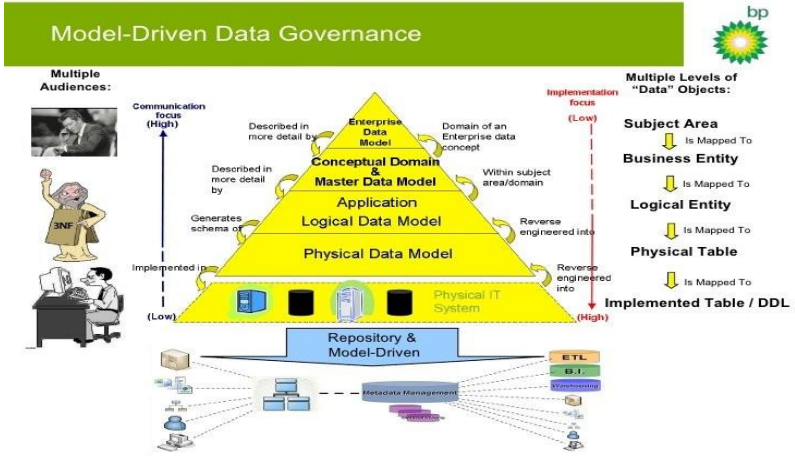
Data Penindakan

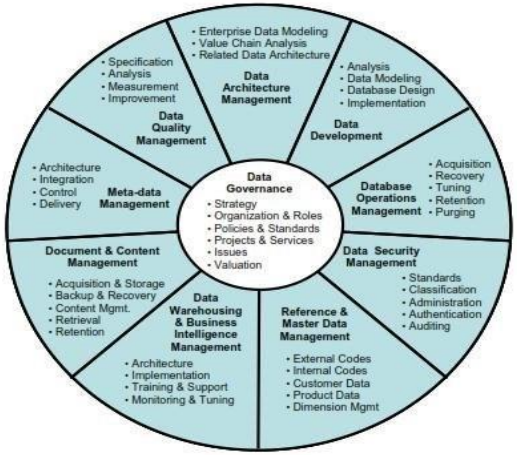
**Penggunaan data/fungsi bersama - interoperabilitas**

- Menaikkan kualitas data di lingkungan organisasi
- Konsistensi data dan update dijaga
- Mengupayakan agar memiliki skema yang sama ataupun pemetaan skema yang terbuka → skema data diketahui umum
- Mengupayakan data referensi sama → data referensi diketahui


# Kualitas Data bergantung Governance

# Federated Database

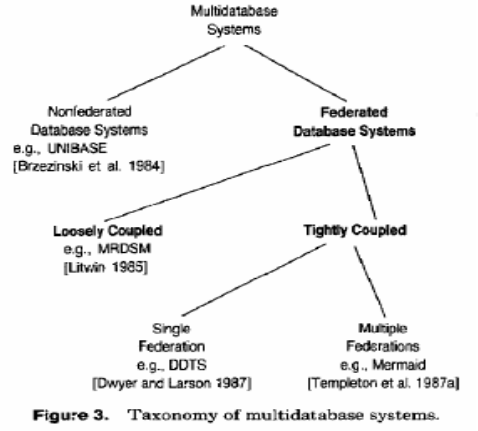



**Figure 3.** Taxonomy of multidatabase systems.

- Tidak mungkin memaksa setiap pihak "menyerahkan" datanya
- Setiap pihak memiliki teknologi dan sistem masing-masing
- Transparency
- Heterogeneity
- Functionality
- Autonomy of underlying federated sources,
- Extensibility & Openness
- Optimized performance


Sistem A

Metadata

XSD1

XML1

Data1

XSD dari Metadata

XML Translator

XSLT

De;nisi
XSD1 → XSD2

Metadata

XML2

Data1

XSD2

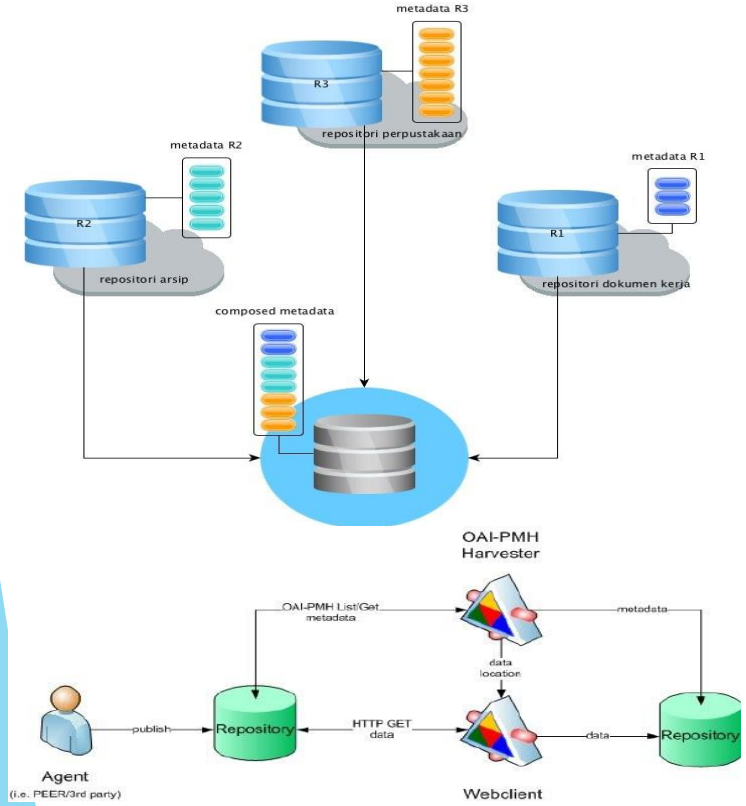
XSD dari Metadata

Sistem B

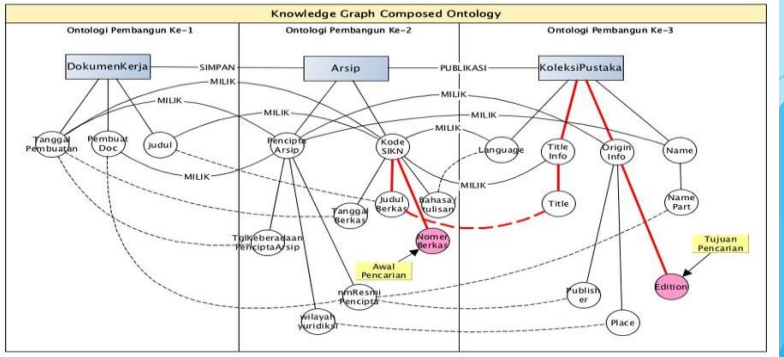
| DocID | FieldA | FieldB | FieldC |
|---|---|---|---|
| | | | |

| ID | FieldX | FieldY |
|---|---|---|
| | | |

# Ontologi dan Database




| DATABASE RELASIONAL | ONTOLOGI |
|---|---|
| Close World Assumption (CWA), Fokus pada data | Open World Assumption (OWA), Fokus pada makna |
| Adanya Constraint untuk mencapai data integritas, namun mungkin menyembunyikan makna | Adanya Ontology axioms untuk menspesi;kasikan makna, dapat digunakan untuk pencapaian integritas |
| Tidak menggunakan hirarki ISA | Hirarki ISA merupakan backbone |
| Skema lebih sederhan, belum tentu dapat digunakan kembali | Skema lebih kompleks, dapat digunakan kembali |



# Pemilihan (Seleksi Fitur) Data

| | K1 | K2 | K3 | K4 | K5 | K6 |
|---|---|---|---|---|---|---|
| R1 | | | | | | |
| R2 | | | | | | |
| R3 | | | | | | |
| R4 | | | | | | |

Fitur:
Kolom yang dipilih
Untuk sesuai tujuan

| Name of the statistical features | Formula/description |
|---|---|
| Standard error | $\sqrt{\frac{1}{n-2}\left[\sum (y-\bar{y})^2 - \frac{\sum[(x-\bar{x})(y-\bar{y})]^2}{\sum (x-\bar{x})^2}\right]}$ |
| Standard deviation | $\sqrt{\frac{n\sum x^2-(\sum x)^2}{n(n-1)}}$ |
| Sample variance | $\sqrt{\frac{n\sum x^2-(\sum x)^2}{n(n-1)}}$ |
| Kurtosis | $\left\{\frac{n(n+1)}{(n-1)(n-2)(n-3)}\sum\left(\frac{x_i-\bar{x}}{S_d}\right)^4\right\} - \frac{3(n-1)^2}{(n-2)(n-3)}$ |
| Skewness | $\frac{n}{(n-1)(n-2)}\sum\left(\frac{x_i-\bar{x}}{S_d}\right)^3$ |
| Maximum value | Maximum signal point value in a given signal. |
| Minimum value | Minimum signal point value in a given signal. |
| Range | Difference in maximum and minimum signal point values for a given signal. |
| Sum | Sum of all feature values for each sample. |
| Mean | The arithmetic average of a set of values or distribution. |
| Median | Middle value separating the greater and lesser halves of a data set. |
| Mode | A statistical term that refers to the most frequently occurring number found in a set of numbers. (i.e.) The |

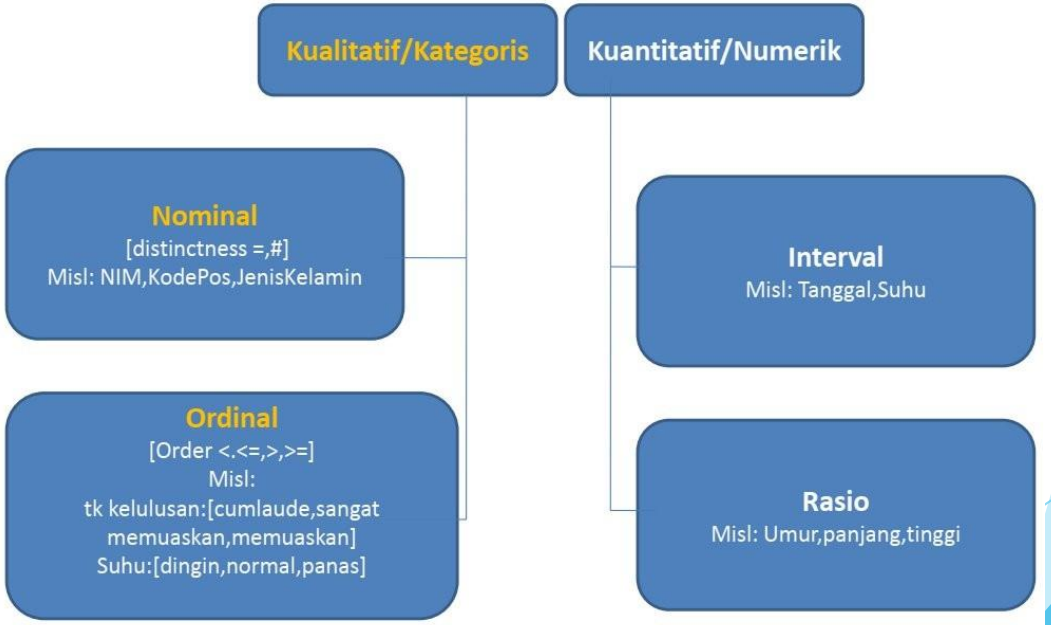
- Setelah menentukan sampling atas data yang akan diambil nanti, selanjutnya adalah melakukan seleksi **fitur** (feature selection) atas data sampling tsb --> Memilih Kolom/Atribut/Variabel yang akan diolah lebih lanjut
- Terminologi **fitur** di Data Science atau Machine Learning adalah Kolom/Atribut/Variabel yang dianggap & dihitung sebagai prioritas (sedikit berbeda dengan terminologi fitur di Statitistika)
- Seleksi fitur merupakan konsep inti dalam ML yang berdampak besar bagi kinerja model prediksi,
- Fitur data yang tidak/sebagian saja relevan dampak berdampak negatif thdp kinerja model
- Definisi Seleksi Fitur: proses otomatis atau manual memilih fitur data yang **paling berkontribusi** thdp variabel prediksi atau output yang diinginkan.

# Seleksi Fitur Data

- Manfaat:
  - Reduksi *Overfirring*: semakin kecil data redundant maka keputusan berdasarkan noise semakin berkurang
  - Meningkatkan Akurasi: semakin kecil data misleading maka akurasi model lebih baik
  - Reduksi Waktu Training: semakin kecil titik data (data point) maka kompleksitas algortima berkurang dan latih algoritma lebih cepat
- Jenis:
  - **Unsupervised**: metode yang **mengabaikan variabel target,** seperti menghapus variabel yang berlebihan menggunakan *korelasi*
  - **Supervised**: metode yang **menggunakan variabel target**, seperti menghapus variabel yang tidak relevan

# Seleksi Fitur

- Membedakan jenis data: Numerik vs Kategorik (lihat modul 6 utk penjabaran)



# Validasi Data

- Veri;kasi vs Validasi
  - Veri;kasi: Benar vs Salah (sesuai prosedur)
  - Validasi: Kuat vs Lemah (sesuai kenyataan)
- Validasi merupakan tahapan kritikal yang sering diabaikan DS-tist pemula, karena memeriksa, diantaranya sbb:
  - Tipe Data (mis. integer, float, string)
  - Range Data
  - Uniqueness (mis. Kode Pos)
  - Consisten expression (mis. Jalan, Jl., Jln.)
  - Format Data (mis. utk tgl "YYYY-MM-DD" VS "DD-MM-YYYY.") → tmt (terhitung mulai tanggal)
  - Nilai Null/Missing Values
  - Misspelling/Type
  - Invalid Data (gender: L/P: L; Laki-laki; P: Pria/Perempuan? )

- Teknik Validasi Data dan Model:
  - Akurasi
  - Kelengkapan
  - Konsistensi
  - Ketepatan Waktu
  - Kepercayaan
  - Nilai Tambah
  - Penafsiran
  - Kemudahan Akses

# Pandas: DataFrame

## Syntax – Creating DataFrames

| | a | b | c |
|---|---|---|---|
| 1 | 4 | 7 | 10 |
| 2 | 5 | 8 | 11 |
| 3 | 6 | 9 | 12 |

```
df = pd.DataFrame(
          {"a" : [4 ,5, 6],
           "b" : [7, 8, 9],
           "c" : [10, 11, 12]},
         index = [1, 2, 3])
```
Specify values for each column.

```
df = pd.DataFrame(
     [[4, 7, 10],
      [5, 8, 11],
      [6, 9, 12]],
     index=[1, 2, 3],
     columns=['a', 'b', 'c'])
```
Specify values for each row.

| | | a | b | c |
|---|---|---|---|---|
| n | v | | | |
| d | 1 | 4 | 7 | 10 |
| | 2 | 5 | 8 | 11 |
| e | 2 | 6 | 9 | 12 |

```
df = pd.DataFrame(
          {"a" : [4 ,5, 6],
           "b" : [7, 8, 9],
           "c" : [10, 11, 12]},
index = pd.MultiIndex.from_tuples(
          [('d',1),('d',2),('e',2)],
             names=['n','v']))
```
Create DataFrame with a MultiIndex

## Reshaping Data – Change the layout of a data set

**pd.melt(df)**
Gather columns into rows.

**df.pivot(columns='var', values='val')**
Spread rows into columns.

**pd.concat([df1,df2])**
Append rows of DataFrames

**pd.concat([df1,df2], axis=1)**
Append columns of DataFrames

**df.sort_values('mpg')**
Order rows by values of a column (low to high).

**df.sort_values('mpg',ascending=False)**
Order rows by values of a column (high to low).

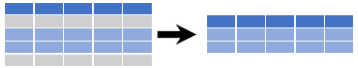
**df.rename(columns = {'y':'year'})**
Rename the columns of a DataFrame

**df.sort_index()**
Sort the index of a DataFrame

**df.reset_index()**
Reset index of DataFrame to row numbers, moving index to columns.

**df.drop(columns=['Length','Height'])**
Drop columns from DataFrame

## Subset Observations (Rows)



**df[df.Length > 7]**
Extract rows that meet logical criteria.

**df.drop_duplicates()**
Remove duplicate rows (only considers columns).

**df.head(n)**
Select first n rows.

**df.tail(n)**
Select last n rows.

**df.sample(frac=0.5)**
Randomly select fraction of rows.

**df.sample(n=10)**
Randomly select n rows.

**df.iloc[10:20]**
Select rows by position.

**df.nlargest(n, 'value')**
Select and order top n entries.

**df.nsmallest(n, 'value')**
Select and order bottom n entries.

## Subset Variables (Columns)

**df[['width','length','species']]**
Select multiple columns with specific names.

**df['width']** *or* **df.width**
Select single column with specific name.

**df.filter(regex='*regex*')**
Select columns whose name matches regular expression *regex*.

# Hands On: Seleksi Fitur

- Dalam praktek kali ini akan digunakan 3 teknik seleksi ;tur yang mudah dan memberikan hasil yang baik:
  - Seleksi Univariat (Univariate Selection)
  - Pentingnya Fitur (Feature Importance)
  - Matriks Korelasi (Correlation Matrix) dengan *Hearmap*
- Sumber dataset: https://www.kaggle.com/iabhishekofficial/mobile-price-classification#train.csv

- Deskripsi variabel dari dataset:
  - *battery_power:* Total energy a battery can store in one time measured in mAh
  - *blue*: Has Bluetooth or not
  - *clock_speed:* the speed at which microprocessor executes instructions
  - *dual_sim*: Has dual sim support or not
  - *fc:* Front Camera megapixels
  - *four_g:* Has 4G or not
  - *int_memory:* Internal Memory in Gigabytes
  - *m_dep:* Mobile Depth in cm
  - *mobile_wt:* Weight of mobile phone
  - *n_cores:* Number of cores of the processor
  - *pc:* Primary Camera megapixels
  - *px_height:* Pixel Resolution Height

# Hands On: Seleksi Fitur

- **Deskripsi variabel dari dataset** (lanjutan):
  - Seleksi Univariate
    - Uji statistik dapat digunakan utk memilih ;tur-
      - ;tur tsb yang memiliki relasi paling kuat dengan variabel output
    - Library scikit-learn menyediakan class
      - SelectKBest yang digunakan utk serangkaian uji statistik berbeda utk memilih angka spesi;k dari ;tur
    - Berikut ini adalah uji statistik chi-square utk
      - ;tur non-negatif utk memilih 10 ;tur terbaik dari dataset *Mobile Price Range Predicrion*.

`px_width:` Pixel Resolution Width
`ram:` Random Access Memory in MegaBytes
`sc_h:` Screen Height of mobile in cm
`sc_w:` Screen Width of mobile in cm
`talk_time:` the longest time that a single battery charge will last when you are

- `three_g:` Has 3G or not
- `touch_screen:` Has touch screen or not
- `wifi:` Has wifi or not
- `price_range:` This is the target variable with a value of 0(low cost), 1(medium cost), 2(high cost) and 3(very high cost).

# Hands On: Seleksi Fitur

- Seleksi Univariat (lanjutan):

```
import pandas as pd
import numpy as np
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2

data = pd.read_csv("C:/Users/Bayu/Documents/DTS 2021/Datasets/train.csv")

X = data.iloc[:,0:20]  #independent columns
y = data.iloc[:,-1]    # target colum i.e price range

# apply SelectKBest class to extract

bestfeatures = SelectKBest(score_func=chi2, k=10)
fit = bestfeatures.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)

#concat two dataframes for better visualization

featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Specs','Score']  #naming the dataframe columns
print(featureScores.nlargest(10,'Score'))  #print 10 best features
```

```
           Specs          Score
13           ram  931267.519053
11     px_height   17363.569536
0   battery_power   14129.866576
12      px_width    9810.586750
8       mobile_wt      95.972863
6       int_memory      89.839124
15          sc_w       16.480319
16     talk_time       13.236400
4              fc       10.135166
14          sc_h        9.614878
```

Import library / modul yang dibutuhkan

Load datasets, sesuaikan dengan path direktori masing-masing

`Iloc[]`, digunakan untuk untuk seleksi/ slicing data dengan parameter index menggunakan bilangan bulat.

Output

# Hands On: Seleksi Fitur

- **Feature Importance (FT)**
  - FT berfungsi memberi skor untuk setiap ;tur data, semakin tinggi skor semakin penting atau relevan ;tur tersebut terhadap variabel output
  - FT merupakan kelas inbuilt yang dilengkapi dengan Pengklasi;kasi Berbasis Pohon (Tree Based Classi;er), kita akan menggunakan Pengklasi;kasi Pohon Ekstra untuk mengekstraksi 10 ;tur teratas untuk kumpulan data

```
import pandas as pd
import numpy as np

data = pd.read_csv("C:/Users/Bayu/Documents/DTS 2021/Datasets/train.csv")
X = data.iloc[:,0:20]  #independent columns
y = data.iloc[:,-1]    #target column i.e price range

from sklearn.ensemble import ExtraTreesClassifier
import matplotlib.pyplot as plt
model = ExtraTreesClassifier()
model.fit(X,y)

print(model.feature_importances_) #use inbuilt class feature_importances of tree based classifiers

#plot graph of feature importances for better visualization
feat_importances = pd.Series(model.feature_importances_, index=X.columns)
feat_importances.nlargest(10).plot(kind='barh')
plt.show()
```

Mende;nisikan model yang akan digunakan yaitu menggunakan algoritma **ExtraTreesClassifier**.

`model. t()` untuk melatih model diikuti oleh parameter variable data

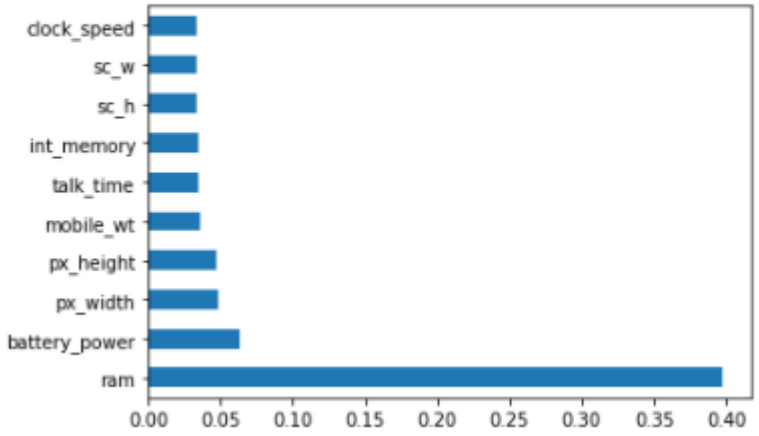
`.largest(10)`: membuat ploting 10 data teratas.

`.plot(kind='barh')`: untuk membuat jenis plot diagram batang horizontal

# Hands On: Seleksi Fitur

- Output:

```
[0.06329642 0.0193987  0.03334552 0.0188696  0.03144026 0.01622896
 0.03468226 0.03269537 0.03574171 0.03269081 0.03317167 0.04704737
 0.04849356 0.39695054 0.03392805 0.03372551 0.03512574 0.01359888
 0.01910327 0.02046578]
```



# Hands On: Seleksi Fitur

- **Matriks Korelasi dengan *Heatmap***
  - Korelasi menyatakan bagaimana ;tur terkait satu sama lain atau variabel target.
  - Korelasi bisa positif (kenaikan satu nilai ;tur meningkatkan nilai variabel target) atau negatif (kenaikan satu nilai ;tur menurunkan nilai variabel target)
  - *Hearmap* memudahkan untuk mengidenti;kasi ;tur mana yang paling terkait dengan variabel target, kami akan memplot peta panas ;tur yang berkorelasi menggunakan `seaborn` library

```python
import pandas as pd
import numpy as np
import seaborn as sns

data = pd.read_csv("C:/Users/Bayu/Documents/DTS 2021/Datasets/train.csv")

X = data.iloc[:,0:20]  #independent columns
y = data.iloc[:,-1]    #target column i.e price range

#get correlations of each features in dataset
corrmat = data.corr()
top_corr_features = corrmat.index
plt.figure(figsize=(20,20))

#plot heat map
g=sns.heatmap(data[top_corr_features].corr(),annot=True,cmap="RdYlGn")
```

*Figure*: adalah window atau page atau halaman dalam objek visual. kalau kita ngegambar di kertas, maka kertas tersebutlah yang di namakan ;gure.

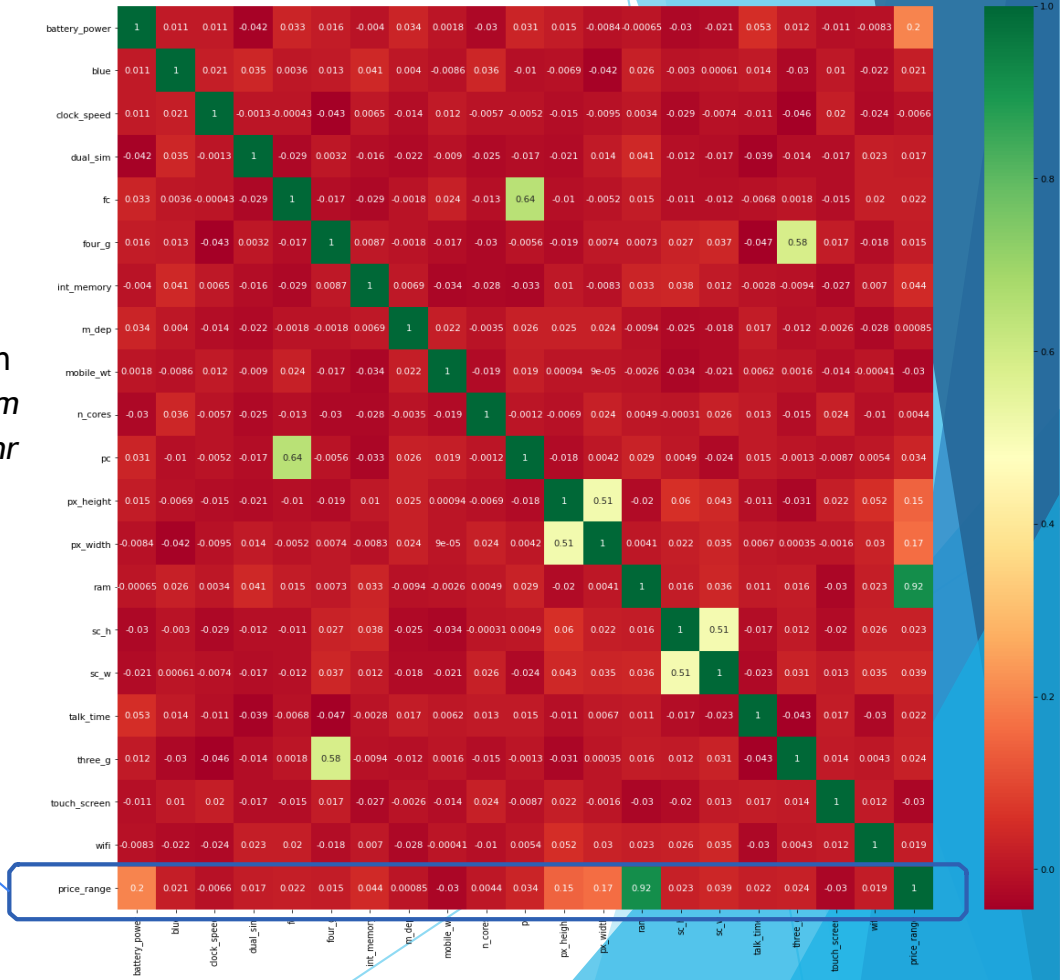
*Ágsize()*: *ukuran dari figure, mengambil dua paramerer lebar dan ringgi (dalam inci)*

*cmap*: Colormap digunakan untuk memetakan nilai data yang dinormalisasi ke warna RGBA.

*annot*=***True*** untuk menampilkan korelasi antar atribut. Jika nilai korelasi mendekati 1 maka hubungan antar atribut semakin tinggi

# Hands On: Seleksi Fitur

- **Matriks Korelasi dengan *Heatmap* (lanjutan)**
  - lihat pada baris terakhir yaitu *price range*, korelasi antara price range dengan ;tur lain dimana ada relasi kuat dengan variabel *ram* dan diikuti oleh var *barrery power* , *px heighr* and *px widrh*.
  - sedangkan utk var *clock_speed* dan *n_cores* berkorelasi lemah dengan *price range*



# Hands-on Data Cleaning

- Data cleaning atas data berantakan (messy data), seperti:
  - missing value,
  - format tidak konsisten
  - record tidak berbentuk baik (malformed record)
  - outlier yang berlebihan
- Lingkup hands-on:
  - Membuang kolom-kolom tidak penting dalam suatu DataFrame
  - Mengubah indeks di DataFrame
  - Membersihkan kolom dengan metode `.str()`
  - Membersihkan semua dataset dengan fungsi `DataFrame.applymap()`
  - Merubah nama kolom sehingga kolom lebih mudah dikenali
  - Melewatkan baris-baris tidak penting dalam ;le CSV

# Hands-on Data Cleaning: Datasets

- File CSV tentang “Daftar Buku dari British Library”, nama ;le “`BL-Flickr- Images-Book.csv`”, link:
  - https://github.com/realpython/python-data-cleaning/blob/master/Datasets/BL-Flickr- Images-Book.csv
- File teks tentang “Kota lokasi Sekolah Tinggi di US”, nama ;le “`university_towns.txt`”, link: https://github.com/realpython/python-data-cleaning/blob/master/Datasets/university_towns.txt
- File CSV tentang “Partisipasi Semua Negara di Olimpiade Musim Dingin dan Musim
  - Panas”, nama ;le “`olympics.csv`”, link: https://github.com/realpython/python-data-cleaning/blob/master/Datasets/olympics.csv

# Hands-on Data Cleaning: Import modul

Diasumsikan peserta sudah memahami library Pandas dan NumPy (lihat di modul sebelumnya) termasuk Pandas workshouse Series dan objek `DataFrame`

1. Import modul yang dibutuhkan

```
import pandas as pd
import numpy as np
```

Jika ingin melihat statistik dasar pada DataFrame di Pandas dengan fungsi .describe():

```
df.describe()
```

# Hands-on Data Cleaning: Membuang (drop) Kolom

- Membuang Kolom pada `DataFrame`
- Sering ditemukan bbrp kategori data tidak terlalu berguna di dataset, misal untuk menganalisis IPK mahasiswa , data nama orangtua, alamat adalah data tidak penting
- Pandas menyediakan fungsi untuk membuang (drop) kolom-kolom yang tidak diinginkan dengan fungsi `drop()`.
  1. Buat DataFrame di luar ;le CSV . Dalam contoh berikut kita lewatkan path relatif ke `pd.read.csv`, yaitu seluruh dataset berada di nama folder Datasets di direktori kerja

# Hands-on Data Cleaning: Membuang (drop) Kolom

```
df = pd.read_csv("C:/Users/Bayu/Documents/DTS 2021/Datasets/BL-Flickr-Images-Book.csv")
df.head()
```

| | Identifier | Edition Statement | Place of Publication | Date of Publication | Publisher | Title | Author | Contributors | Corporate Author | Corporate Contributors | Former owner | Engraver | Issuance type | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 206 | NaN | London | 1879 [1878] | S. Tinsley & Co. | Walter Forbes. [A novel.] By A. A | A. A. | FORBES, Walter. | NaN | NaN | NaN | NaN | monographic | http |
| 1 | 216 | NaN | London; Virtue & Yorston | 1868 | Virtue & Co. | All for Greed. [A novel. The dedication signed... | A., A. A. | BLAZE DE BURY, Marie Pauline Rose - Baroness | NaN | NaN | NaN | NaN | monographic | http |
| 2 | 218 | NaN | London | 1869 | Bradbury, Evans & Co. | Love the Avenger. By the author of "All for Gr... | A., A. A. | BLAZE DE BURY, Marie Pauline Rose - Baroness | NaN | NaN | NaN | NaN | monographic | http |
| 3 | 472 | NaN | London | 1851 | James Darling | Welsh Sketches, chiefly ecclesiastical, to the... | A., E. S. | Appleyard, Ernest Silvanus. | NaN | NaN | NaN | NaN | monographic | http |

`df.head()` : berfungsi untuk menampilkan baris awal dari dataFrame, secara default bila tdk diberi parameter akan menampilkan 5 baris data

# Hands-on Data Cleaning: Membuang (drop) Kolom

- Melihat pada lima entri pertama dengan perintah `head()`.
- Dapat dilihat bahwa beberapa kolom memberikan informasi tambahan yang akan membantu perpustakaan tetapi tidak terlalu deskriptif tentang buku itu sendiri: `Edition Statement`, `Corporate Author`, `Corporate Contributors`, `Former owner`, `Engraver`, `Issuance type` and `Shelfmarks`.
- Kita drop kolom-kolom tsb dengan perintah:

```
to_drop = ['Edition Statement',
           'Corporate Author',
           'Corporate Contributors',
           'Former owner',
           'Engraver',
           'Contributors',
           'Issuance type',
           'Shelfmarks']
```

```
df.drop(to_drop, inplace=True, axis=1)
```

Kita de;nisikan daftar (list) nama dari semua kolom yang ingin kita drop. Kemudian jalankan perintah fungsi `drop()`.

dengan parameter `inplace` bernilai `True` dan parameter `axis` bernilai `1`, di mana 1 adalah angka sumbu (0 untuk baris dan 1 untuk kolom.)

# Hands-on Data Cleaning: Membuang (drop) Kolom

- Inspeksi ulang `DataFrame`, kolom yang tidak diinginkan sudah dibuang

```
df.head()
```

| | Identifier | Place of Publication | Date of Publication | Publisher | Title | Author | Flickr URL |
|---|---|---|---|---|---|---|---|
| 0 | 206 | London | 1879 [1878] | S. Tinsley & Co. | Walter Forbes. [A novel.] By A. A | A. A. | http://www.flickr.com/photos/britishlibrary/ta... |
| 1 | 216 | London; Virtue & Yorston | 1868 | Virtue & Co. | All for Greed. [A novel. The dedication signed... | A., A. A. | http://www.flickr.com/photos/britishlibrary/ta... |
| 2 | 218 | London | 1869 | Bradbury, Evans & Co. | Love the Avenger. By the author of "All for Gr... | A., A. A. | http://www.flickr.com/photos/britishlibrary/ta... |
| 3 | 472 | London | 1851 | James Darling | Welsh Sketches, chiefly ecclesiastical, to the... | A., E. S. | http://www.flickr.com/photos/britishlibrary/ta... |
| 4 | 480 | London | 1857 | Wertheim & Macintosh | [The World in which I live, and my place in it... | A., E. S. | http://www.flickr.com/photos/britishlibrary/ta... |

# Hands-on Data Cleaning: Membuang (drop) Kolom

- Alternatif utk membuang kolom, dengan meneruskannya langsung ke parameter columns daripada memisahkan label-label yang mau dibuang:

```
df.drop(colums=to_drop, inplace=True)
```

inplace=True adalah perintah menimpa kolom dan menyimpan kolom/fitur yang dimanipulasi (dlm hal ini di drop)

- Sintak ini lebih intuitif dan mudah dibaca dibanding sintak sebelumnya?

```
df.drop(to_drop, inplace=True, axis=1)
```

## Handling Missing Data

**df.dropna()**
Drop rows with any column having NA/null data.
**df.fillna(value)**
Replace all NA/null data with value.

# Hands-on Data Cleaning: Mengubah Indeks di DataFrame

- Index dalam Pandas memperluas fungsionalitas array NumPy untuk memungkinkan pemotongan (slicing) dan pelabelan yang lebih fleksibel. Dalam banyak kasus, akan sangat membantu jika menggunakan ;eld pengenal data yang bernilai unik sebagai indeksnya.
- Sebagai contoh, dengan dataset di slide sebelumnya, praktiknya saat pustakawan mencari record, biasanya akan memasukan identi;er unik suatu buku:

```
df['Identifier'].is_unique
```

```
True
```

`df['Identi er']`: slicing/seleksi kolom/;eld yang akan di eksekusi.

`.is_unique`: function untuk mengecek nilai unik

# Hands-on Data Cleaning: Mengubah Indeks di DataFrame

- Gantikan indeks yang ada pada kolom ini menggunakan `set_index` :

`.set_index()`: function untuk merubah index dengan diikuti parameter kolom yang akan dipilih untuk dijadikan index

```
df = df.set_index('Identifier')
df.head()
```

| Identifier | Place of Publication | Date of Publication | Publisher | Title | Author | Flickr URL |
|---|---|---|---|---|---|---|
| 206 | London | 1879 [1878] | S. Tinsley & Co. | Walter Forbes. [A novel.] By A. A | A. A. | http://www.flickr.com/photos/britishlibrary/ta... |
| 216 | London; Virtue & Yorston | 1868 | Virtue & Co. | All for Greed. [A novel. The dedication signed... | A., A. A. | http://www.flickr.com/photos/britishlibrary/ta... |
| 218 | London | 1869 | Bradbury, Evans & Co. | Love the Avenger. By the author of "All for Gr... | A., A. A. | http://www.flickr.com/photos/britishlibrary/ta... |
| 472 | London | 1851 | James Darling | Welsh Sketches, chiefly ecclesiastical, to the... | A., E. S. | http://www.flickr.com/photos/britishlibrary/ta... |
| 480 | London | 1857 | Wertheim & Macintosh | [The World in which I live, and my place in it... | A., E. S. | http://www.flickr.com/photos/britishlibrary/ta... |

**Technical Detail**: Unlike primary keys in SQL, a Pandas `Index` doesn't make any guarantee of being unique, although many indexing and merging operations will notice a speedup in runtime if it is.

# Hands-on Data Cleaning: Mengubah Indeks di DataFrame

- Kami dapat mengakses setiap records dengan cara yang mudah dengan `loc[]`. Cara ini digunakan untuk *label-based indexing*, yaitu memberi label suatu baris atau kolom tanpa memperhatikan posisi/lokasinya.

```
df.loc[206]
```

```
Place of Publication                                        London
Date of Publication                                     1879 [1878]
Publisher                                          S. Tinsley & Co.
Title                                  Walter Forbes. [A novel.] By A. A
Author                                                        A. A.
Flickr URL              http://www.flickr.com/photos/britishlibrary/ta...
Name: 206, dtype: object
```

- Dengan kata lain, 206 adalah label pertama dari indeks. Utk mengakses berdasarkan posisinya, gunakan `df.iloc[]`

loc: untuk seleksi dengan menggunakan label/bilangan bulat
iloc: untuk seleksi dengan menggunakan bilangan bulat
contoh penggunaan lain:
iin.iloc[::3,:].head(10): untuk Memilih baris kelipatan 3, dengan semua kolom dan menampilkan 10 data pertama.

# Hands-on Data Cleaning: Mengubah Indeks di DataFrame

- Pada slide sebelumnya, Indeks yang digunakan adalah `RangeIndex`: integer mulai dari 0, analog dengan range di Python. Dengan meneruskan nama kolom ke `set_index`, maka indeks telah diubah ke nilai dalam Identi;er.

- Diperhatikan pada langkah sebelumnya bahwa telah dilakukan penetapan kembali variabel ke objek yang dikembalikan oleh metode dengan df = `df.set_index(...)`. Ini karena, secara default, metode mengembalikan salinan objek yang dimodi;kasi dan tidak membuat perubahan secara langsung ke objek. Hal ini dapat dihindari dengan mengatur parameter `inplace`:

```
df.set_index('Identifier', inplace=True)
```

# Hands-on Data Cleaning: Merapihkan *Fields* dalam Data

- Slide sebelumnya telah dibuang bbrp kolom tidak penting dan diubah indeks pada `DataFrame` hingga menjadi lebih masuk akal.
- Selanjutnya, akan dibersihkan kolom tertentu dan mengubah menjadi bentuk/format yang seragam hingga dataset lebih mudah dipahami dan memastikan konsistensi. Dalam slide berikutnya akan dibersihkan `Date of Publication` dan `Place of Publication`.
- Dalam inspeksi, semua tipe data saat ini adalah objek `dtype` yang analog dengan `str` di native Python

# Hands-on Data Cleaning: Merapihkan *Fields* dalam Data

- Cara ini dilakukan sebagai rangkuman saat setiap ;eld tidak dapat dirapihkan sebagai data numerik atau data kategorik dan data yang digunakan cukup "kotor" atau "berantakan".

```
df.dtypes.value_counts()
```

```
object    6
dtype: int64
```

# Hands-on Data Cleaning: Merapihkan *Fields* dalam Data

- Satu kolom yang masuk akal untuk menerapkan nilai numerik adalah tanggal publikasi sehingga kita dapat melakukan perhitungan di awal:

- Buku tertentu hanya memiliki satu tanggal publikasi. Oleh karena itu perlu dilakukan hal berikut:
  - Hilangkan tanggal lain dalam kurung siku, `1879[1878]`
  - Konversi rentang tanggal ke "`start date`", `1860-63; 1839, 38-54`
  - Hilangkan tanggal yang tidak jelas dan gantikan dengan `NaN` NumPy, [1879?] -> `NaN`
  - Konversi string nan ke nilai NaN NumPy

```
df.loc[1905:, 'Date of Publication'].head(10)
```

```
Identifier
1905           1888
1929    1839, 38-54
2836           1897
2854           1865
2956        1860-63
2957           1873
3017           1866
3131           1899
4598           1814
4884           1820
Name: Date of Publication, dtype: object
```

`df.loc[1905:, 'Nama Field']`: digunakan untuk mengakses index mulai dari index 1905 dengan output hanya pada ;eld tanggal publikasi

`.head(10)`: function untuk menampilkan baris awal dataFrame dengan parameter hingga index ke 10 atau 10 baris data

# Hands-on Data Cleaning: Merapihkan *Fields* dalam Data

- Mensintesis pola-pola ini, manfaatkan ekspresi reguler (Regex) tunggal untuk mengekstrak tahun publikasi.

```
regex = r'^(\d{4})'
```

- perintah `\d` mewakili sebarang digit dan `{4}` mengulangi aturan (rule) sebanyak empat kali. Kararakter `^` sesuai dengan awal string, dan tanda dalam kurung `()` menunjukkan *capruring group* yang memberikan sinyal ke Pandas bahwa akan dilakukan ekstraksi bagian Regex tersebut.

# Hands-on Data Cleaning: Regex di Pandas

| regex (Regular Expressions) Examples | |
|---|---|
| `'\.'` | Matches strings containing a period '.' |
| `'Length$'` | Matches strings ending with word 'Length' |
| `'^Sepal'` | Matches strings beginning with the word 'Sepal' |
| `'^x[1-5]$'` | Matches strings beginning with 'x' and ending with 1,2,3,4,5 |
| `'^(?!Species$).*'` | Matches strings except the string 'Species' |

**`df.loc[:,'x2':'x4']`**

Select all columns between x2 and x4 (inclusive).

**`df.iloc[:,[1,2,5]]`**

Select columns in positions 1, 2 and 5 (first column is 0).

**`df.loc[df['a'] > 10, ['a','c']]`**

Select rows meeting logical condition, and only the specific columns .

# Hands-on Data Cleaning: Merapihkan *Fields* dalam Data

- Coba jalankan regex di dataset

```
extr = df['Date of Publication'].str.extract(r'^(\d{4})', expand=False)
extr.head()
```

```
Identifier
206    1879
216    1868
218    1869
472    1851
480    1857
Name: Date of Publication, dtype: object
```

**Further Reading:** Not familiar with regex? You can inspect the expression above at regex101.com and learn all about regular expressions with Regular Expressions: Regexes in Python.

Mengekstrak data untuk setiap string subjek hasil tangkapan variabel regex dari kolom `Date of Publication`

`expand=False`: Jika Benar, kembalikan DataFrame dengan satu kolom per grup tangkapan. Jika Salah, kembalikan Seri/Indeks jika ada satu grup tangkapan atau DataFrame jika ada beberapa grup tangkapan.

# Hands-on Data Cleaning: Merapihkan *Fields* dalam Data

- Secara teknis, kolom tsb masih memiliki dtype = object, namun dengan mudah kita dapatkan versi numeriknya dengan perintah pd.to_numeric

```
df['Date of Publication'] = pd.to_numeric(extr)
df['Date of Publication'].dtype
```

```
dtype('float64')
```

- Ini menghasilkan sekitar 1/10 nilai yang hilang, cost yang cukup kecil dampaknya untuk saat ini karena dapat melakukan perhitungan pada nilai valid yang tersisa:

```
df['Date of Publication'].isnull().sum() / len(df)
```

```
0.11717147339205986
```

# Hands-on Data Cleaning: Membersihkan Kolom dengan Kombinasi metode `str` dengan NumPy

- Slide sebelumnya dibahas penggunaan `df['Date of Publication'].str`. Atribut ini adalah cara akses cepat operasi string di Pandas yang menyerupai operasi pada native Python atau mengkompilasi regex seperti `.split()`, `.replace()`, dan `.capitalize()`.
- Utk membersihkan ;eld `Place of Publication`, kombinasikan metode `str` di Panda dengan fungsi `np.where` di NumPy yang mirip dengan bentuk vektor dari makro `IF()` di Excell, dengan sintak berikut:

```
np.where(condition, then, else)
```

# Hands-on Data Cleaning: Membersihkan Kolom dengan Kombinasi metode `str` dengan NumPy

```
np.where(condition, then, else)
```

- condition mirip dengan objek array atau Boolean .then adalah nilai yang digunakan jika condition mengevaluasi menjadi True, dan else untuk mengevaluasi nilai selainnya.
- .where membawa tiap elemen dalam objek digunakan untuk condition dan memeriksa elemen tertentu menjadi True dalam konteks kondisi dan mengembalikan ndarray terdiri dari then atau else, tergantung pada prakteknya.

# Hands-on Data Cleaning: Membersihkan Kolom dengan Kombinasi metode `str` dengan NumPy

- Dapat juga dituliskan dalam bersarang (nested) menjadi pernyataan *If-Then*, memungkinkan menghitung nilai berbasiskan kondisi berganda:

```
Python                                                    >>>

>>> np.where(condition1, x1,
             np.where(condition2, x2,
                 np.where(condition3, x3, ...)))
```

- Kemudian, dapat digunakan dua fungsi tsb untuk membersihkan ;eld Place of Publication karena kolom tsb memiliki objek string. Berikut adalah isi dari kolom:

# Hands-on Data Cleaning: Membersihkan Kolom dengan Kombinasi metode `str` dengan NumPy

```
df['Place of Publication'].head(10)
```

```
Identifier
206                                        London
216                        London; Virtue & Yorston
218                                        London
472                                        London
480                                        London
481                                        London
519                                        London
667     pp. 40. G. Bryan & Co: Oxford, 1898
874                                       London]
1143                                       London
Name: Place of Publication, dtype: object
```

- Dilihat pada hasil di atas, ;eld `place of publication` masih ada informasi yang tidak penting. Jika dilihat lebih teliti, kasus ini untuk beberapa baris yang `place of publication` -nya di "London" dan "Oxford"

# Hands-on Data Cleaning: Membersihkan Kolom dengan Kombinasi metode `str` dengan NumPy

```
df.loc[4157862]
```

```
Place of Publication                          Newcastle-upon-Tyne
Date of Publication                                        1867.0
Publisher                                             T. Fordyce
Title                  Local Records; or, Historical Register of rema...
Author                     FORDYCE, T. - Printer, of Newcastle-upon-Tyne
Flickr URL             http://www.flickr.com/photos/britishlibrary/ta...
Name: 4157862, dtype: object
```

```
df.loc[4159587]
```

```
Place of Publication                          Newcastle upon Tyne
Date of Publication                                        1834.0
Publisher                                          Mackenzie & Dent
Title                  An historical, topographical and descriptive v...
Author                                          Mackenzie, E. (Eneas)
Flickr URL             http://www.flickr.com/photos/britishlibrary/ta...
Name: 4159587, dtype: object
```

- Pada dua entri di samping, dua buku diterbitkan di tempat yang sama (newcastle upon tyne) namun salah satunya memilik tanda hubung (-)
- Untuk membersihkan kolom ini dalam sekali jalan, gunakan `str.contains()` untuk mendapatkan Boolean mask.

```
pub = df['Place of Publication']
london = pub.str.contains('London')
london[:5]
```

```
Identifier
206     True
216     True
218     True
472     True
480     True
Name: Place of Publication, dtype: bool
```

```
oxford = pub.str.contains('Oxford')
```

# Hands-on Data Cleaning: Membersihkan Kolom dengan Kombinasi metode `str` dengan NumPy

- Kombinasikan dengan `np.where`:

```
df['Place of Publication'] = np.where(london, 'London',
                                      np.where(oxford, 'Oxford',
                                               pub.str.replace('-', ' ')))
df['Place of Publication'].head()
```

```
Identifier
206     London
216     London
218     London
472     London
480     London
Name: Place of Publication, dtype: object
```

- Di sini, fungsi `np.where` berbentuk struktur nested, dimana condition berbentuk Series dari Boolean dengan `str.contains()`. Metode `contains()` bekerja mirip dengan keyword in yang digunakan untuk mencari kejadian suatu entitas dalam kondisi pengulangan iterasi (atau substring dalam suatu string)

# Hands-on Data Cleaning: Membersihkan Kolom dengan Kombinasi metode `str` dengan NumPy

- Pergantian tanda hubung (hypen) dengan spasi dengan `str.replace()` dan re-assign ke kolom dalam `DataFrame`.
- Walau pada kenyataan masih banyak dataset ini (kolom dan baris) yang "kotor", namun dalam contoh di sini hanya dibahas pada dua kolom

# Hands-on Data Cleaning: Membersihkan Kolom dengan Kombinasi metode `str` dengan NumPy

- Coba periksa kembali untuk lima entri pertama, hasilnya akan lebih rapih dan "bersih" dibandingkan dataset awal sebelum dilakukan *cleaning dara*.

```
df.head()
```

| Identifier | Place of Publication | Date of Publication | Publisher | Title | Author | Flickr URL |
|---|---|---|---|---|---|---|
| 206 | London | 1879.0 | S. Tinsley & Co. | Walter Forbes. [A novel.] By A. A | A. A. | http://www.flickr.com/photos/britishlibrary/ta... |
| 216 | London | 1868.0 | Virtue & Co. | All for Greed. [A novel. The dedication signed... | A., A. A. | http://www.flickr.com/photos/britishlibrary/ta... |
| 218 | London | 1869.0 | Bradbury, Evans & Co. | Love the Avenger. By the author of "All for Gr... | A., A. A. | http://www.flickr.com/photos/britishlibrary/ta... |
| 472 | London | 1851.0 | James Darling | Welsh Sketches, chiefly ecclesiastical, to the... | A., E. S. | http://www.flickr.com/photos/britishlibrary/ta... |
| 480 | London | 1857.0 | Wertheim & Macintosh | [The World in which I live, and my place in it... | A., E. S. | http://www.flickr.com/photos/britishlibrary/ta... |

**Note:** At this point, `Place of Publication` would be a good candidate for conversion to a `Categorical` dtype, because we can encode the fairly small unique set of cities with integers. (*The memory usage of a Categorical is proportional to the number of categories plus the length of the data; an object dtype is a constant times the length of the data.*)

# Hands-on Data Cleaning: Membersihkan Seluruh Dataset dengan Fungsi `applymap`

- Pada situasi tertentu, data berantakan alias "kotor" tidak hanya berlaku di sebagian kolom atau baris (record) tapi menyebar ke banyak bagian dataset.
- Cara berikut dapat diterapkan untuk semua cell atau elemen di `DataFrame` (dataset).
- Metode `.applymap()` dapat diterapkan, dimana similar dengan fungsi built-in yaitu fungsi `map()`.

# Hands-on Data Cleaning: Membersikan seluruh Dataset

- Terapkan fungsi `applymap()` pada ;le "`university_towns.txt`":

Shell

```
$ head Datasets/univerisity_towns.txt
Alabama[edit]
Auburn (Auburn University)[1]
Florence (University of North Alabama)
Jacksonville (Jacksonville State University)[2]
Livingston (University of West Alabama)[2]
Montevallo (University of Montevallo)[2]
Troy (Troy University)[2]
Tuscaloosa (University of Alabama, Stillman College, Shelton State)[3][4]
Tuskegee (Tuskegee University)[5]
Alaska[edit]
```

Perintah shell

- Dapat dilihat di atas, bahwa nama negara bagian(state) diikuti dengan kota asal universitas:
  StateA TownA1 TownA2 StateB TownB1 TownB2....
  dan memiliki substring "`[edit]`"

# Hands-on Data Cleaning: Membersihkan seluruh Dataset

- Kita dapat memanfaatkan pola ini dengan membuat list of (state, city) tuples dan *wrapping* daftar (list) dalam DataFrame.

```
university_towns = []
with open("C:/Users/Bayu/Documents/DTS 2021/Datasets/university_towns.txt") as file:
    for line in file:
        if '[edit]' in line:
            # Remember this `state` until the next is found
            state = line
        else:
            # Otherwise, we have a city; keep `state` as last-seen
            university_towns.append((state, line))

university_towns[:5]
```

```
[('Alabama[edit]\n', 'Auburn (Auburn University)[1]\n'),
 ('Alabama[edit]\n', 'Florence (University of North Alabama)\n'),
 ('Alabama[edit]\n', 'Jacksonville (Jacksonville State University)[2]\n'),
 ('Alabama[edit]\n', 'Livingston (University of West Alabama)[2]\n'),
 ('Alabama[edit]\n', 'Montevallo (University of Montevallo)[2]\n')]
```

- Kita dapat membungkus (wrap) daftar ini dalam DataFrame dan mengatur kolom sebagai "State" and "RegionName".
- Pandas akan mengambil setiap elemen dalam daftar dan mengatur "State" ke nilai kiri dan "RegionName" ke nilai kanan.
- Hasilnya adalah DataFrame sbb:

```
towns_df = pd.DataFrame(university_towns,
                        columns=['State', 'RegionName'])

towns_df.head()
```

| | State | RegionName |
|---|---|---|
| 0 | Alabama[edit]\n | Auburn (Auburn University)[1]\n |
| 1 | Alabama[edit]\n | Florence (University of North Alabama)\n |
| 2 | Alabama[edit]\n | Jacksonville (Jacksonville State University)[2]\n |
| 3 | Alabama[edit]\n | Livingston (University of West Alabama)[2]\n |
| 4 | Alabama[edit]\n | Montevallo (University of Montevallo)[2]\n |

# Hands-on Data Cleaning: Membersihkan seluruh Dataset

- Pandas, mempermudah dalam pembersihan string dengan hanya membutuhkan nama state dan nama town dan dapat membuang lainnya. Selain dapat kembali menggunakan metode `.str()` di Pandas, dapat juga menggunakan `applymap()` untuk memetakan setiap elemen di DataFrame
- Perhatikan kasus sederhana pada contoh DataFrame berikut:

| | 0 | 1 |
|---|---|---|
| 0 | mock | Dataset |
| 1 | python | pandas |
| 2 | real | python |
| 3 | numpy | clean |

- Pada contoh di atas, setiap sel ("Mock", "Dataset", "Python", "Real", dll) adalah elemen. Oleh karena itu perintah `applymap()` akan menerapkan fungsi ke setiap elemen secara independen. Mari kita de;nisikan fungsi tsb:

# Hands-on Data Cleaning: Membersihkan seluruh Dataset

- Fungsinya dide;nisikan berikut:

```
def get_citystate(item):
    if ' (' in item:
        return item[:item.find(' (')]
    elif '[' in item:
        return item[:item.find('[')]
    else:
        return item
```

- `applymap()` di Pandas hanya butuh satu parameter , yaitu fungsi yang diterapkan ke setiap elemen:

```
towns_df =  towns_df.applymap(get_citystate)
```

- Pertama, de;nisikan fungsi Python yang mengambil setiap elemen dari `DataFrame` sebagai parameternya. Di dalam fungsi, pengecekan dilakukan utk menentukan apakah ada elemen atau tidak!

# Hands-on Data Cleaning: Membersihkan data dengan `applymap()`

- Tergantung pada pengecekan, nilai dikembalikan berdasarkan fungsi.
- Lalu, fungsi applymap() dipanggil pada objek yg ada. Sehingga kita dapatkan DataFrame yang relatif lebih rapih

```
towns_df.head()
```

| | State | RegionName |
|---|---|---|
| 0 | Alabama | Auburn |
| 1 | Alabama | Florence |
| 2 | Alabama | Jacksonville |
| 3 | Alabama | Livingston |
| 4 | Alabama | Montevallo |

**Technical Detail**: While it is a convenient and versatile method, `.applymap` can have significant runtime for larger datasets, because it maps a Python callable to each individual element. In some cases, it can be more efficient to do *vectorized* operations that utilize Cython or NumPY (which, in turn, makes calls in C) under the hood.

# Hands-on Data Cleaning: Mengganti Nama Kolom

- Seringkali dalam dataset yang dimiliki ada nama kolom yang suli utk dipahami atau informasi yang tidak penting dalam beberapa baris awal/akhir, misal de;nisi istilah atau footnotes.
- Oleh karena itu dapat dilakukan penggantian nama dan melewatkan beberapa baris sehingga bisa dilakukan analisis informasi dari baris yang benar atau dapat dipahami.
- Kita akan lakukan utk lima baris awal dataset "`olympic.csv`":

Shell

```
$ head -n 5 Datasets/olympics.csv
0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15
,? Summer,01 !,02 !,03 !,Total,? Winter,01 !,02 !,03 !,Total,? Games,01 !,02 !,03 !,C
Afghanistan (AFG),13,0,0,2,2,0,0,0,0,0,13,0,0,2,2
Algeria (ALG),12,5,2,8,15,3,0,0,0,0,15,5,2,8,15
Argentina (ARG),23,18,24,28,70,18,0,0,0,0,41,18,24,28,70
```

# Hands-on Data Cleaning: Mengganti Nama Kolom

- Kemudian, baca dalam DataFrame di Pandas:

```
olympics_df = pd.read_csv("C:/Users/Bayu/Documents/DTS 2021/Datasets/olympics.csv")
olympics_df.head()
```

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | NaN | ? Summer | 01 ! | 02 ! | 03 ! | Total | ? Winter | 01 ! | 02 ! | 03 ! | Total | ? Games | 01 ! | 02 ! | 03 ! | Combined total |
| 1 | Afghanistan (AFG) | 13 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 13 | 0 | 0 | 2 | 2 |
| 2 | Algeria (ALG) | 12 | 5 | 2 | 8 | 15 | 3 | 0 | 0 | 0 | 0 | 15 | 5 | 2 | 8 | 15 |
| 3 | Argentina (ARG) | 23 | 18 | 24 | 28 | 70 | 18 | 0 | 0 | 0 | 0 | 41 | 18 | 24 | 28 | 70 |
| 4 | Armenia (ARM) | 5 | 1 | 2 | 9 | 12 | 6 | 0 | 0 | 0 | 0 | 11 | 1 | 2 | 9 | 12 |

- Hasilnya berantakan! Kolom adalah bentuk string integer indeks 0. Baris yang harusnya sebegai header pada `olympics_df.iloc[0]`. Hal ini terjadi karena ;le CSV mulai dengan 0, 1, 2, ...., 15.
- Dan, jika kita ke sumber dataset ini, akan terlihat NaN yang ada harusnya berisikan "Country" dan "Summer" maksudnya adala"Summer Games" dan "01!" harusnya adalah "Gold" , dll.

# Hands-on Data Cleaning: Mengganti Nama Kolom

- Oleh karena itu, hal berikut yang perlu dilakukan:
  - Melewatkan (skip) satu baris dan atur header sebagai baris pertama (indeks-0)
  - Mengganti Nama Kolom
- Melewatkan baris dan atur header dapat dilakukan pada saat membaca ;le CSV dengan mempassing beberapa parameter ke fungsi `read_csv()`.
- Fungsi `read_csv()` memerlukan banyak parameter opsional, namun utk kasus ini hanya diperlukan satu (header) yang dihilangkan pada baris ke-0, dengan hasil sbb:

# Hands-on Data Cleaning: Mengganti Nama Kolom

- Hasil fungsi `read_csv()` dan menghilangkan satu baris (header)

```
olympics_df = pd.read_csv("C:/Users/Bayu/Documents/DTS 2021/Datasets/olympics.csv", header=1)
olympics_df.head()
```

| | Unnamed: 0 | ? Summer | 01 ! | 02 ! | 03 ! | Total | ? Winter | 01 !.1 | 02 !.1 | 03 !.1 | Total.1 | ? Games | 01 !.2 | 02 !.2 | 03 !.2 | Combined total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Afghanistan (AFG) | 13 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 13 | 0 | 0 | 2 | 2 |
| 1 | Algeria (ALG) | 12 | 5 | 2 | 8 | 15 | 3 | 0 | 0 | 0 | 0 | 15 | 5 | 2 | 8 | 15 |
| 2 | Argentina (ARG) | 23 | 18 | 24 | 28 | 70 | 18 | 0 | 0 | 0 | 0 | 41 | 18 | 24 | 28 | 70 |
| 3 | Armenia (ARM) | 5 | 1 | 2 | 9 | 12 | 6 | 0 | 0 | 0 | 0 | 11 | 1 | 2 | 9 | 12 |
| 4 | Australasia (ANZ) [ANZ] | 2 | 3 | 4 | 5 | 12 | 0 | 0 | 0 | 0 | 0 | 2 | 3 | 4 | 5 | 12 |

- Sekarang, yang tampak di samping adalah sekumpulan baris yang benar sebagai header dan semua baris yang tidak dibutuhkan telah dihilangkan.
- Pandas telah merubah nama kolom yang mengandung nama "countries" dari NaN menjadi Unnamed:0

# Hands-on Data Cleaning: Mengganti Nama Kolom

- Utk mengganti nama kolom, digunakan metode `rename()` DataFrame yg memungkinkan memberi label pada axis berdasarkan pemetaan (dalam kasus ini yaitu dict)
- Mulai dengan mende;nisikan suatu kamus yang memetakan nama kolom saat ini sebagi kunci ke yang lebih dapat digunakan”

```
new_names =  {'Unnamed: 0': 'Country',
              '? Summer': 'Summer Olympics',
              '01 !': 'Gold',
              '02 !': 'Silver',
              '03 !': 'Bronze',
              '? Winter': 'Winter Olympics',
              '01 !.1': 'Gold.1',
              '02 !.1': 'Silver.1',
              '03 !.1': 'Bronze.1',
              '? Games': '# Games',
              '01 !.2': 'Gold.2',
              '02 !.2': 'Silver.2',
              '03 !.2': 'Bronze.2'}
```

# Hands-on Data Cleaning: Mengganti Nama Kolom

- Kemudian, panggil fungsi `rename()` pada objek dimaksud:

```
olympics_df.rename(columns=new_names, inplace=True)
```

- Atur `inplace` menjadi `True`, dengan hasil sbb:

```
olympics_df.head()
```

| | Country | Summer Olympics | Gold | Silver | Bronze | Total | Winter Olympics | Gold.1 | Silver.1 | Bronze.1 | Total.1 | # Games | Gold.2 | Silver.2 | Bronze.2 | Combined total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Afghanistan (AFG) | 13 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 13 | 0 | 0 | 2 | 2 |
| 1 | Algeria (ALG) | 12 | 5 | 2 | 8 | 15 | 3 | 0 | 0 | 0 | 0 | 15 | 5 | 2 | 8 | 15 |
| 2 | Argentina (ARG) | 23 | 18 | 24 | 28 | 70 | 18 | 0 | 0 | 0 | 0 | 41 | 18 | 24 | 28 | 70 |
| 3 | Armenia (ARM) | 5 | 1 | 2 | 9 | 12 | 6 | 0 | 0 | 0 | 0 | 11 | 1 | 2 | 9 | 12 |
| 4 | Australasia (ANZ) [ANZ] | 2 | 3 | 4 | 5 | 12 | 0 | 0 | 0 | 0 | 0 | 2 | 3 | 4 | 5 | 12 |

# Referensi


- Krensky P. Data Pre Tools: Goals, Bene;ts, and The Advantage of Hadoop. Aberdeen Group Report. July 2015
- SAS. Data Preparation Challenges Facing Every Enterprise. ebook. December 2017
- https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/?sh=6e9aa0e36f63
- https://improvado.io/blog/what-is-data-preparation
- https://searchenterpriseai.techtarget.com/feature/Data-preparation-for-machine-learning-still-requires-humans?
- https://towardsdatascience.com/feature-selection-techniques-in-machine-learning-with-python-f24e7da3f36e
- CRISP-DM

# Tools Lab Online

- jupyter notebook
- scikit-learn
- pandas
- numpy

# Ringkasan

- Data preparation memiliki sebutan lain, diantaranya data pre-processing, data cleaning, data manipulation,
- Data preparation mengambil porsi kerja terbanyak dalam data science 60-80%
- Data preparation membutuhkan ketelitian dan kesabaran/kerajinan dari peneliti DS, terutama pemula
- Data Validation merupakan tahapan kritikal dari DS namun sering diabaikan para peneliti
- Seleksi Fitur harus dilakukan di awal tahapan data preparation setelah melakukan penentuan metode/teknik sampling
- Data cleaning merupakan pekerjaan yang sangat memerlukan keahlian teknik DS terkait penggunakan tools dan coding
- Kebersihan data merupakan sarat mutlak utk Model Prediksi yang Baik.

# Terima Kasih