Pandas: Werken met tabellen in pandas

Dr. Cornelis Stal

April 28, 2022

1 Werken met tabellen in `pandas`

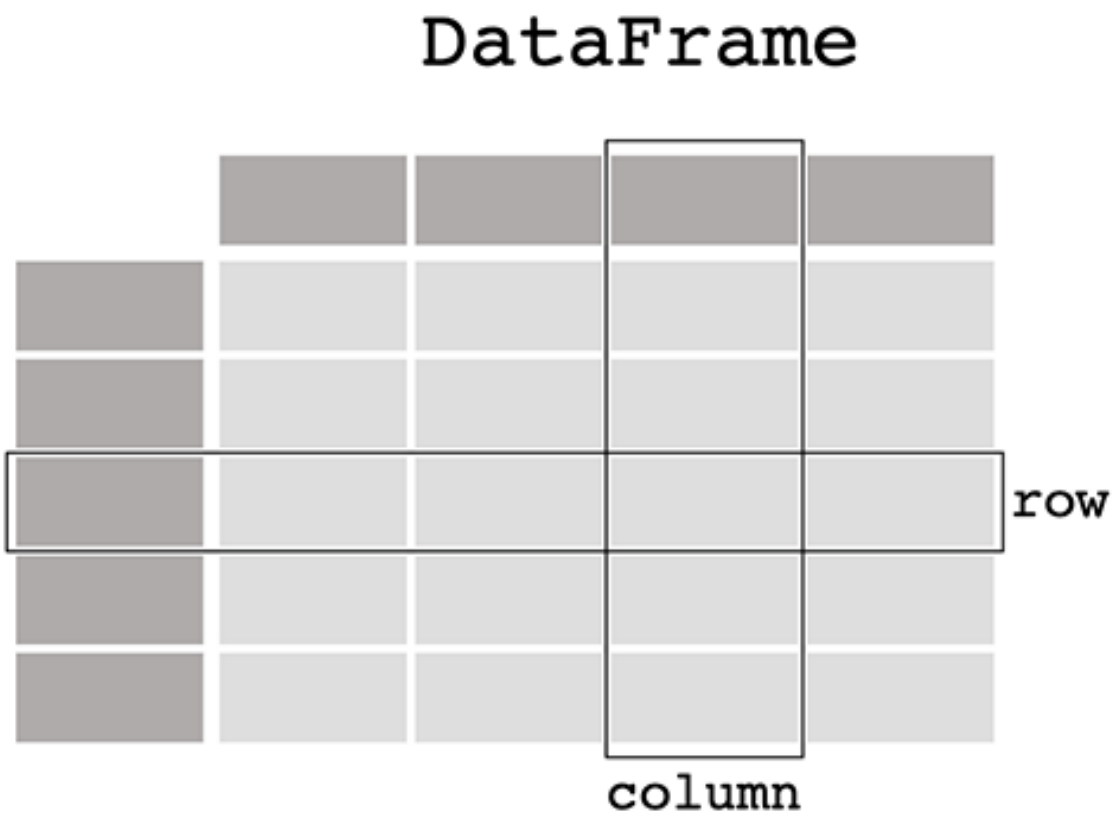
> Deze tutorial is een vertaling van de *Pandas Tutorial* op https://pandas.pydata.org/pandas-docs/stable/getting_started/.

Deze tutorial omvat een eerste algemeen overzicht over het gebruik van `pandas`.

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

Om de `pandas`-bibliotheek te laden en ermee aan de slag te gaan, importeren we eerst deze bibliotheek. Door informele conventie gebruiken we de alias `pd` voor `pandas`. Het laden van `pandas` als `pd` wordt dus standaard aangenomen voor alle `pandas`-documentatie.

1.1 `pandas` data tabel representatie



We nemen een deel van de basisstatistieken van Vlaamse gemeenten. Voor een aantal gemeenten nemen we de naam (`NAAM`, als tekst), het aantal inwonders (`INWONERS`, aantallen in gehele getallen) en de oppervlakte (`OPPERVLKATE`, hectare in decimale getallen).

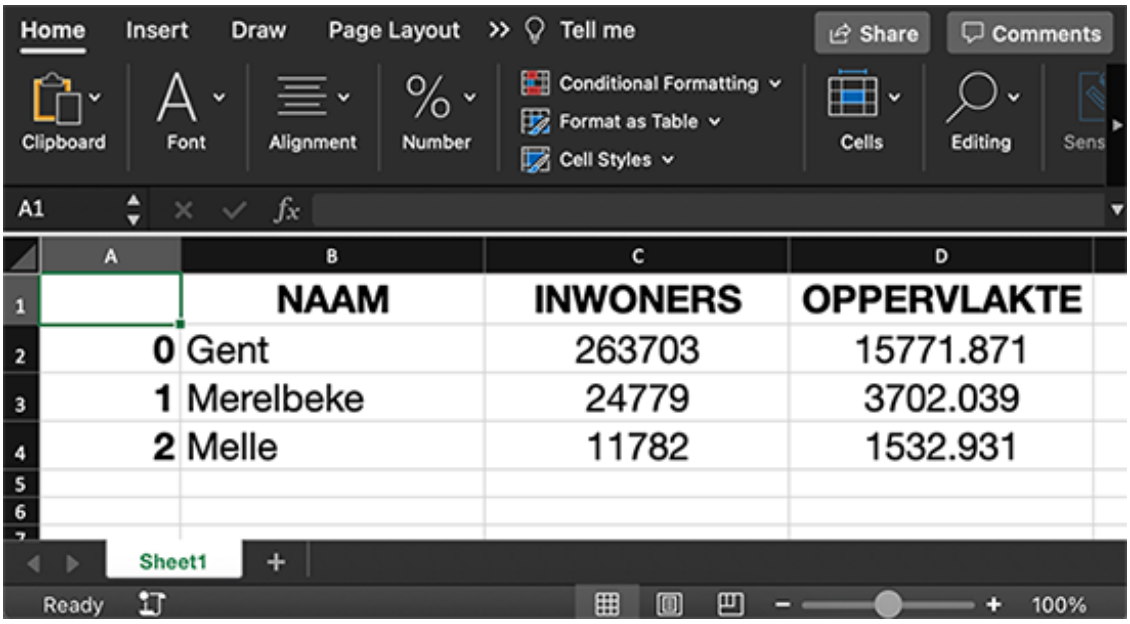
[ ]:
```
df = pd.DataFrame(
    {
        "NAAM": [
            "Gent",
            "Merelbeke",
            "Melle",
        ],
        "INWONERS": [263703, 24779, 11782],
        "OPPERVLAKTE": [15771.871, 3702.039, 1532.931],
    }
)
df
```

We maken een `DataFrame`-object aan om manueel data op te slaan in een tabel. Als deze data bestaan uit een lijst van Python `dict`-objecten, zal de sleutel ('key') van ieder `dict`-object gebruikt worden als kolomhoofding. De waarden ('values') vormen uiteraarde de waarden die in de kolommen van de `DataFrame` verwerkt worden.

Een `DataFrame` is een 2-dimensionele datastructuur waarin verschillende datatypes opgeslagen kunnen worden in verschillende kolommen, zoals karakters, gehele getallen, decimale getallen, categorische data, …). Deze methodiek is dus zeer vergelijkbaar met een spreadsheet, SQL-tabel, een attribuuttabel van een conventionele GIS-databank of een `data.frame` in R, …

- De tabel heeft 3 kolommen, met ieder een kolomhoofding. Deze zijn respectiegelijk `NAAM`, `INWONERS` en `OPPERVLAKTE`.
- De kolom `NAAM` bestaat uit tekstuele data, waarbij iedere waarde bestaan uit een `string`. De kolommen `INWONERS` en `OPPERVLAKTE` bevatten numerieke data, meer bepaald respectievelijk `integers` en `floats`.

In een spreadsheet software, zoals Excel, zou de overeenkomstige tabel van deze data er zeer gelijkaardig uit zien:



1.2 Iedere kolom in een `DataFrame` is een `Series`

Veronderstel dat we enkel geïntereseerd zijn in de kolom met het aantal inwoners (`INWONERS`):

```
[ ]: df["INWONERS"]
```

Wanneer we een enkele kolom uit een pandas `DataFrame` selecteren krijgen we een pandas `Series` terug als resultaat. Om een kolom te selecteren verwijzen we naar de kolomheader als een `string` tussen vierkante haakjes `[]`.

> **Opmerking:** als we vertrouwd zijn met Python `dictionaries`, zullen we op kunnen merken dat het selecteren van een enkele kolom zeer gelijkaardig verloopt als het selecteren van `dict`-waarden met behulp van sleutels ('keys').

We kunnen ook een geheel nieuwe `Series` aanmaken, bijvoorbeeld de 'NIS'-code van een gemeente:

[ ]:
```
nis = pd.Series(['44021', '44043', '44040'], name="NIS")
nis
```

Een pandas `Series` heeft geen kolomhoofdingen, zoals bij een enkele kolom in een `DataFrame`. Een `Series` heeft echter wel rijlabels.

1.3 Handeingen uitvoeren op een `DataFrame` of een `Series`

Laten we de maximale oppervlakte van de beschikbare gemeentes opvragen. We doen dit door de kolom `OPPERLVAKTE` te selecteren uit de `DataFrame`, en vervolgens de `max()`-methode toe te passen:

[ ]:
```
df["OPPERVLAKTE"].max()
```

Deze handeling kunnen we ook uitvoeren op een `Series`-object:

[ ]:
```
oppervlakte = df['OPPERVLAKTE']
oppervlakte.max()
```

Zoals geïllustreerd met de `max()`-methode kunnen we verschillende handelingen uitvoeren op een `DataFrame` en een `Series`. `pandas` beschikt over een groot aantal aanvullende functionaliteiten die beschikbaar zijn als afzonderlijke methoden die we toe kunnen passen op een `DataFrame` of een `Series`. Vermits deze methodes eigenlijk functies zijn, moeten we hierbij niet vergeten om ronde haakjes `()` toe te voegen.

Een interessante methode om een aantal basisstatistieken van numerieke data te verkrijgen is de `describe()`-methode:

[ ]:
```
df.describe()
```

De `describe()`-methode geeft ons dus een snel overzicht van numerieke data in een `DataFrame`. Aangezien de kolom `NAAM` bestaat uit tekstuele data zullen deze waarden standaard niet verwerkt worden door de `describe()`-methode.

Veel `pandas`-operaties resulteren in een nieuw `DataFrame`- of `Series`-object. De `describe()`-methode is hier een mooi voorbeeld van:

[ ]:
```
type(df.describe())
```

> **Gebruikshandleiding:** Voor meer informatie over de `describe()`-methode verwijzen we door naar de gebruikshandleiding over 'aggregations with describe'.

Opmerking: deze tutorial is slechts een beginpunt voor de vele mogelijkheden van `pandas`. Zoals bij spreadsheet software zal data door `pandas` gestructureerd worden in een tabel met kolommen en rijen. Naast deze datarepresentatie kunnen we ook manipulaties en berekeningen uitvoeren op deze tabellen, zoals we deze bijvoorbeeld ook met MS Excel out zouden kunnen voeren. Hier zullen we in de volgende tutorials verder op ingaan.

1.4 Te onthouden:

- Importeer de bibliotheek met behulp van `import pandas as pd`.
- Een tabel wordt opgeslagen als een `pandas DataFrame`.
- Iedere kolom in een `DataFrame` is een `Series`-object.
- Op een `DataFrame`- of `Series`-object kunnen we methodes toepassen.

Gebruikshandleiding: een meer uitgebreide uiteenzetting over `DataFrame`- en `Series`-objecten wordt gegeven in de de sectie 'introduction to data structures'.