Tag 8
Inhaltsverzeichnis

- Bio- und konventionellen BENEJU Landwirtschaftsbetriebe vergleichen
- Umgang mit fehlenden Daten
- Umgang mit Ausreissern
- Daten mit dem AutorBuch Beispiel "joinen"
- Übungen
- BYOQ

Vergleich der Kantone BE NE JU
Daten auslesen

```
import pandas as pd
biobetriebe_df = pd.read_csv('../Data/px-x-0702000000_107-BIOBETRIEB.csv',
                             sep=';', usecols=['Kanton', '1990', '2000', '2010', '2019'], encoding='ISO-8859-1')
print('Shape: ' + str(biobetriebe_df.shape))
biobetriebe_df.head()
```

```
Shape: (26, 5)
```

	Kanton	2019	2010	2000	1990
0	Zürich	1835	1073	1144	338
1	Bern / Berne	4366	3504	3524	677
2	Luzern	1339	832	649	174
3	Uri	153	143	83	3
4	Schwyz	447	400	285	36

```
jubenel_bio_df = biobetriebe_df[biobetriebe_df['Kanton'].isin(['Bern / Berne', 'Jura', 'Neuchâtel'])] \
    [['Kanton', '1990', '2000', '2010', '2019']]
jubenel_bio_df
```

	Kanton	1990	2000	2010	2019
1	Bern / Berne	677	3524	3504	4366
23	Neuchâtel	23	104	109	318
25	Jura	54	138	254	545

Vergleich der Kantone BE NE JU
Daten transponieren

Leider sind die Achsen vertauscht...

Man möchte die Jahre auf der X-Achse und die Kantone auf der Y-Achse.

Lösung: Die *melt*-Funktion verwenden. Diese Funktion transponiert für uns die Matrix.

In [11]:
```
jubenel_bio_melted_df = jubenel_bio_df.melt(id_vars=['Kanton'], \
                                            var_name='Jahr', \
                                            value_name='Bio-Value')

jubenel_bio_melted_df
```

Out[11]:

	Kanton	Jahr	Bio-Value
0	Bern / Berne	1990	677
1	Neuchâtel	1990	23
2	Jura	1990	54
3	Bern / Berne	2000	3524
4	Neuchâtel	2000	104
5	Jura	2000	138

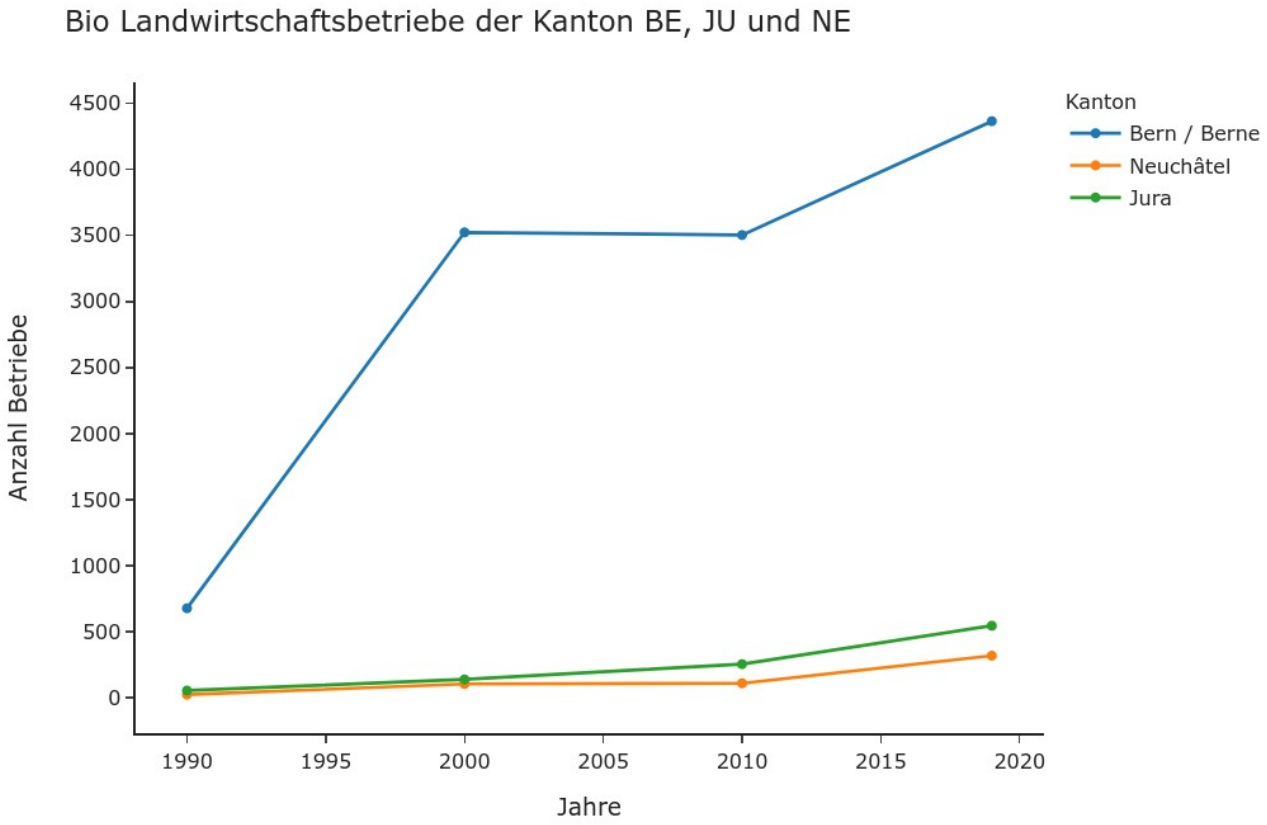
Vergleich der Kantone BE NE JU
Grafische Darstellung

```
:  fig = px.scatter(jubenel_bio_melted_df,
                    x = 'Jahr', y = 'Bio-Value',
                    color = 'Kanton',
                    template = 'simple_white')

   fig.update_traces(mode = 'lines+markers')

   fig.update_layout (title = 'Bio Landwirtschaftsbetriebe der Kanton BE, JU und NE',
                      xaxis = dict(title = 'Jahre'),
                      yaxis = dict(title = 'Anzahl Betriebe'),
                      hovermode = 'x unified')

   fig.show()
```



Umgang mit fehlenden Daten
Unterschiedliche Lösungen

missingdata.csv

```
"Landwirtschaftliche Betriebe und Beschäftigte nach Kanton"

"Variable";"Kanton";"Betriebssystem";"1990";"2000";"2010";"2017"
"Beschäftigte Total";"Zürich";"Konventionelle Betriebe";18832;13832;10907;9227
"Beschäftigte Total";"Bern / Berne";"Konventionelle Betriebe";52640;38082;;27658
"Beschäftigte Total";"Luzern";"Konventionelle Betriebe";19821;15921;13638;12004
```

In [19]:
```
missingdata_df = pd.read_csv('../Data/missingdata.csv',
                             sep=';', header=1, usecols=['Kanton', '1990', '2000', '2010', '2017'], encoding='ISO-8859-1')
print('Shape: ' + str(missingdata_df.shape))
missingdata_df.head()
```

```
Shape: (3, 5)
```

Out[19]:

	Kanton	1990	2000	2010	2017
0	Zürich	18832	13832	10907.0	9227
1	Bern / Berne	52640	38082	NaN	27658
2	Luzern	19821	15921	13638.0	12004

NaN == Not a Number

- Problem: Was machen wir, wenn Daten fehlen?
- Pandas bietet unterschiedliche Lösungen
  1) Sie zuerst mit isnull() entdecken
  2) Datensatz mit dropna() löschen (Zeile oder Kolonne)
  3) NaN-Wert mit einem bestimmten Wert ersetzen (Zeile oder Kolonne)

Umgang mit Ausreissern
Problem und Lösung (1)

- Problem: Was machen wir, wenn Ausreisser vorkommen?
  Definition: Siehe Wikipedia

- Pandas Lösungsansätze
  1) Sie zuerst mit describe() entdecken
  2) Daten visualisieren
  3) Ausreisser eventuell löschen...
     1) Von Hand
     2) Per Programm, wie hier

outlierdata.csv

```
"Landwirtschaftliche Betriebe und Beschäftigte nach Kanton"

"Variable";"Kanton";"Betriebssystem";"1990";"2000";"2010";"2017"
"Beschäftigte Total";"Zürich";"Konventionelle Betriebe";18832;13832;10907;9227
"Beschäftigte Total";"Bern / Berne";"Konventionelle Betriebe";52640;38082;123456789;27658
"Beschäftigte Total";"Luzern";"Konventionelle Betriebe";19821;15921;13638;12004
```

	Kanton	1990	2000	2010	2017
0	Zürich	18832	13832	10907	9227
1	Bern / Berne	52640	38082	123456789	27658
2	Luzern	19821	15921	13638	12004

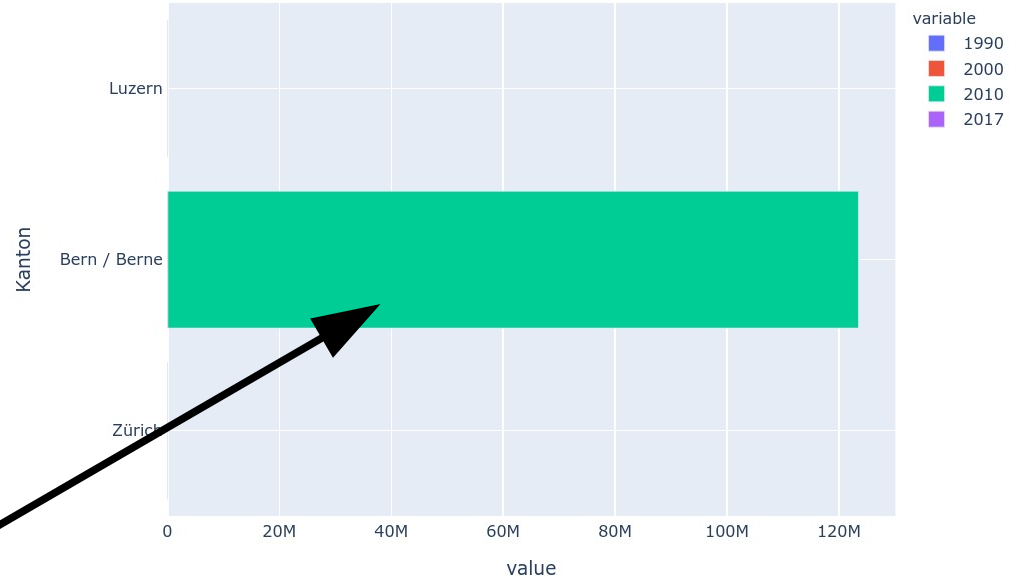
Umgang mit Ausreissern
Problem und Lösung (2)

- Jetzt sieht man deutlich, dass 2010 im Kanton Bern etwas mit den Daten schief gelaufen ist.
- Aber… was machen wir jetzt?
- Spannende Diskussion hier

```
outlierdata_df.describe()
```

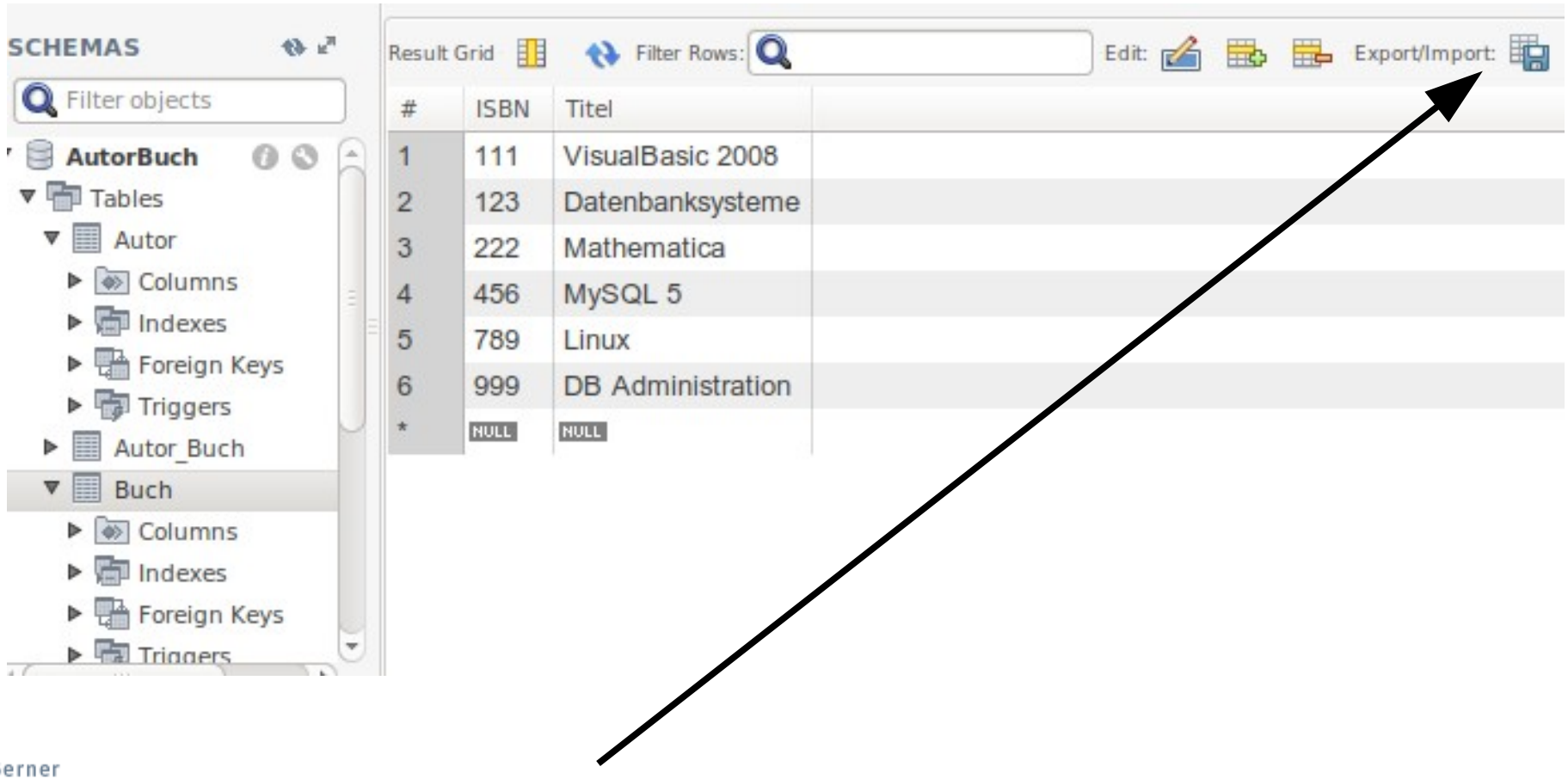
	1990	2000	2010	2017
count	3.000000	3.000000	3.000000e+00	3.000000
mean	30431.000000	22611.666667	4.116044e+07	16296.333333
std	19239.914007	13438.355194	7.127072e+07	9936.978129
min	18832.000000	13832.000000	1.090700e+04	9227.000000
25%	19326.500000	14876.500000	1.227250e+04	10615.500000
50%	19821.000000	15921.000000	1.363800e+04	12004.000000
75%	36230.500000	27001.500000	6.173521e+07	19831.000000
max	52640.000000	38082.000000	1.234568e+08	27658.000000

```
px.bar(outlierdata_df, x = ['1990', '2000','2010', '2017'], y = 'Kanton')
```



Daten "joinen", mit dem AutorBuch Beispiel
Daten aus MySQL als CSV exportieren

- Wissen alle noch was ein Join ist?
- Am einfachsten Datenexport direkt mit MySQL Workbench



Daten "joinen", mit dem AutorBuch Beispiel
Daten im Notebook direkt "joinen"

```
joined_autorBuch_df = pd.merge(autor_df, autorBuch_df, left_on='PersNr', right_on='PersonNr') \
    .drop('PersonNr', axis=1) \
    .merge(buch_df, on='ISBN')
joined_autorBuch_df
```

	PersNr	Vorname	Name	ISBN	Titel
0	12	Alfons	Kemper	123	Datenbanksysteme
1	34	Michael	Kofler	111	VisualBasic 2008
2	34	Michael	Kofler	222	Mathematica
3	34	Michael	Kofler	456	MySQL 5
4	34	Michael	Kofler	789	Linux

Übungen
Weitere Analysen und Vergleiche

1) Starten Sie das Notebook Notebook_ITP_DMOD8_Bio-und-Konv-LWBetriebe-Datenvergleich. Führen Sie alle Zelle von Hand aus und probieren Sie zu verstehen, was da alles passiert.

2) Erstellen Sie ein neues Notebook und bringen Sie das Beispiel der Seiten 10-11 "*Daten "joinen", mit dem AutorBuch Beispiel*" zum Laufen.