DataCraft

3
DataFrames verbinden

DataFrames direkt verbinden

DataCraft

- Zwei DataFrames können untereinander mit der pandas-Funktion concat() geklebt werden

```
a = pd.DataFrame({"Produkt":["Poster","T-
Shirt","Buch"],"Preis":[14.99,19.99,9.99]})
b = pd.DataFrame({"Produkt":["Pullover"],"Preis":[34.99]})
pd.concat([a,b])
```

- Auch das Nebeneinander-Kleben geht mit concat(), dafür setzen wir die Achse auch 1

```
a = pd.DataFrame({"Produkt":["Poster","T-Shirt","Buch"],
                  "Preis":[14.99,19.99,9.99]})
c = pd.DataFrame({"Lager":[20,15,100]})
pd.concat([a,c], axis=1)
```

- Die gleiche Ordnung ist entscheidend!

DataFrames mit ID-Spalte verbinden

DataCraft

Anstatt die Ordnung zu benutzen, verwenden wir eine ID-Spalte (Schlüssel)

Verkäufe

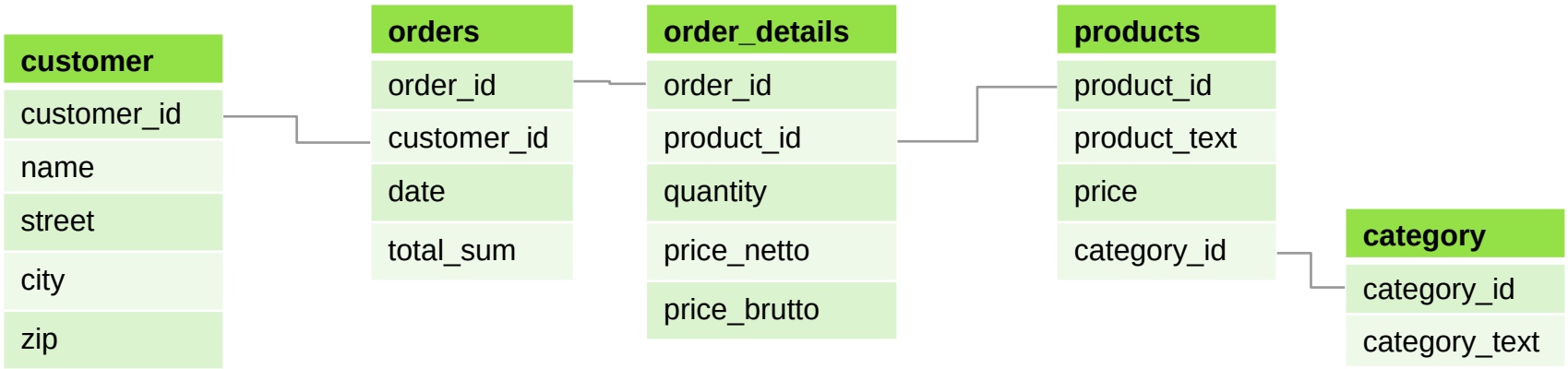
Order-ID	Datum	Produkt-ID
1	24.03.2022	234567
1	24.03.2022	123456
1	24.03.2022	345678
2	25.03.2022	234567

Produktinformationen

Produkt-ID	Beschreibung	Preis
123456	Buch	9,99 €
234567	T-Shirt	19,99 €
345678	Poster	14,99 €
456789	Pullover	34,99 €

DataCraft

In relationalen Datenbanken sind die Tabellen über Schlüssel miteinander verbunden



Die vier JOIN-Arten

DataCraft

Es gibt vier Arten, Tabellen miteinander zu verbinden

INNER JOIN

behalte nur Zeilen, die in beiden Tabellen vorkommen

ID	X
1	Jahr
2	Tag
3	Monat

+

ID	Y
2	365
3	12
4	52

=

ID	X	Y
2	Tag	365
3	Monat	12

FULL OUTER JOIN

behalte alle Zeilen aus beiden Tabellen

ID	X
1	Jahr
2	Tag
3	Monat

+

ID	Y
2	365
3	12
4	52

=

ID	X	Y
1	Jahr	NA
2	Tag	365
3	Monat	12
4	NA	52

LEFT JOIN

behalte alle Zeilen aus der linken Tabelle

ID	X
1	Jahr
2	Tag
3	Monat

+

ID	Y
2	365
3	12
4	52

=

ID	X	Y
1	Jahr	NA
2	Tag	365
3	Monat	12

RIGHT JOIN

behalte alle Zeilen aus der rechten Tabelle

ID	X
1	Jahr
2	Tag
3	Monat

+

ID	Y
2	365
3	12
4	52

=

ID	X	Y
2	Tag	365
3	Monat	12
4	NA	52

Die join-Funktion in pandas

DataCraft

- In pandas heißt die Funktion **join()**

- Die beste Art ist es, den Index als ID-Spalte zu verwenden

```
a = pd.DataFrame({"Produkt-ID":[123,234,345],
                  "Produkt":["Poster","T-Shirt","Buch"],
                  "Preis":[14.99,19.99,9.99]})
c = pd.DataFrame({"Produkt-ID":[345,456,234],
                 "Lager":[20,15,100]})

a = a.set_index("Produkt-ID")
c = c.set_index("Produkt-ID")
a.join(c)
```

- Der Index kann aber auch temporär gebildet werden, wenn man die Tabelle nicht ändern möchte

```
a.join(c.set_index('Produkt-ID'), on='Produkt-ID')
```

Die join-Funktion in pandas

DataCraft

- Mit dem Parameter *how* wird die Art des JOINs festgelegt (left, right, outer, inner)
- Mit den Parametern lsuffix und rsuffix wird bei gleichnamigen Spalten das angegebene Suffix angehangen
- Es können auch mehrere Spalten als Index verwendet werden (Multiindex)

```
a.join(b.set_index(["Produkt-ID","Land"]),
       on=["Produkt-ID","Land"])
```

Die merge-Funktion in pandas

DataCraft

- Einfacher, aber weniger effizient ist die Funktion **merge()**, da hier nicht über den Index verbunden werden muss

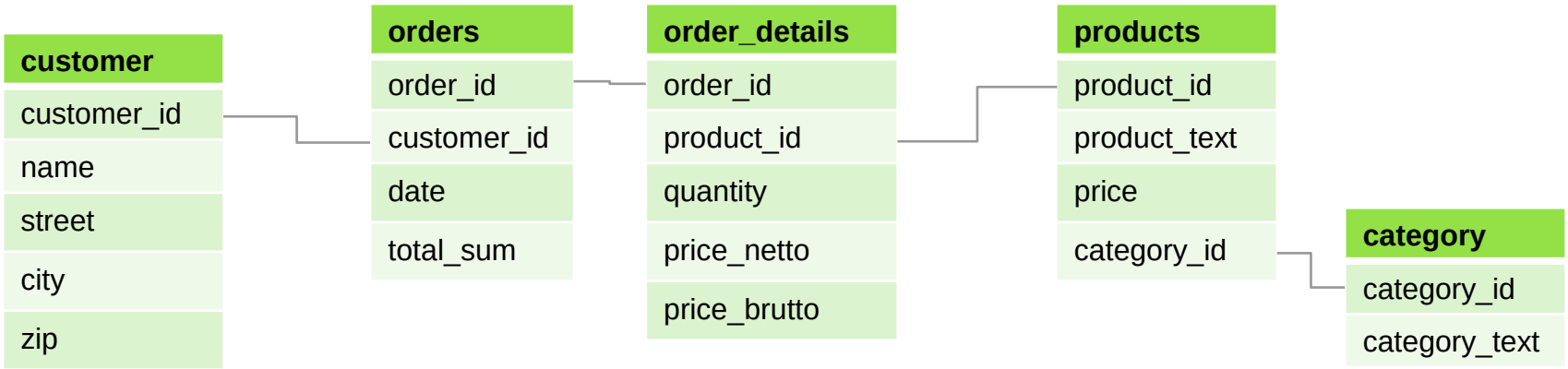
```
a.merge(b, on=["Produkt-ID","Land"])
```

- merge hat auch die Parameter *how* und *on* (oder *left_on* und *right_on*) und *suffixes* (statt lsuffix und rsuffix)

- Interessant sind auch die Parameter *indicator* und *validate*. indicator erstellt eine Spalte _merge, in der die Schlüssel-Quelle steht. validate prüft, um welches Verhältnis es sich handelt
  - "1:1": Schlüssel muss eindeutig sein
  - "1:m": Schlüssel ist eindeutig im linken Datensatz
  - "m:1": Schlüssel ist eindeutig im rechten Datensatz

DataCraft

Wie sind die Beziehungen (1:1, 1:m, m:1, m:n)?



- merge kann mit how="cross" auch alle möglichen Kombinationen erzeugen (kartesisches Produkt, cross join)

```
x = pd.DataFrame({"x":[1, 2, 3]})
y = pd.DataFrame({"y":[True, False]})
x.merge(y, how="cross")
```

Dummy Variablen

- Umwandlung einer Spalte mit mehreren Ausprägungen zu mehreren Indikatorspalten, welche nur die Werte 0 und 1 annehmen mit **get_dummies()**

```
coffee2 = coffee["Species"].str.get_dummies().join(coffee)
del coffee2["Species"]
```