Wirtschaftsinformatik

V03 — Datenarten, Datenmodelle & Datenqualität

Prof. Dr. Christian Bockermann Prof. Dr. Julian Grote
Prof. Dr. Volker Klingspor

Hochschule Bochum · Fachbereich Wirtschaft

Wintersemester 2026/27

Rückblick: Letzte Woche

- Frage: *Was passiert zwischen Klick auf „Bestellen“ und der Zahl im Monatsbericht?*
- Antwort: Die Bestellung durchläuft **viele Stationen** und wird in jeder umgeformt. Der Bericht zeigt das *Ergebnis*, nicht das Original.
- Genau deshalb heute die Anschlussfrage: Wenn jede Station umformt — können zwei Abteilungen **dieselbe** Sache *unterschiedlich* zählen?

Der Vertrieb meldet 1.616 Bestellungen, das Controlling 1.594. Wer hat recht?

Aktivierung — Wortmeldung: Woran könnte die Differenz von 22 Bestellungen liegen? Sammeln Sie ein bis zwei Ideen — wir prüfen sie gleich an den echten Zeilen.

Bevor wir zählen: Was für Daten liegen da überhaupt?

Strukturiert

passt in Zeilen und Spalten

- die Bestelltabelle, das ERP, jede CSV
- maschinell auswertbar *ohne* Vorarbeit

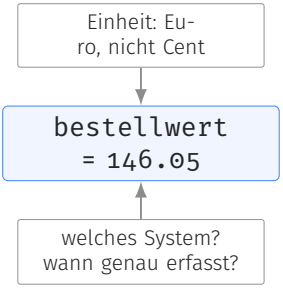
Unstrukturiert

kein festes Raster

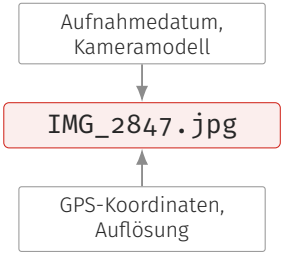
- E-Mail, Retouren-Foto, gescannter Lieferschein
- steckt voller Information, aber schwer zu zählen

Aktivierung — Wortmeldung: Und wie verarbeitet ein Computer das Unstrukturierte eigentlich?

Eine Brick-Flow-Bestellung



Ein Foto auf dem Handy



Metadaten sind Daten über Daten — sie stehen nicht *im* Wert, sondern *um* ihn herum. Ohne sie weiß niemand, was eine Zahl *bedeutet* oder wann/wo ein Foto entstanden ist.

bestellnr	bestelldatum	kundennr	anzahl_pakete	bestellwert	code
8092	2026-01-10	1074	5	146.05	
8093	2026-01-10	1471	5	42.73	
8094	2026-01-10	1491	1	129.48	NEUJAHR26

Zeile

ein *Fall* — hier: eine Bestellung

Spalte

ein *Merkmal* — z. B. der Betrag

Schlüssel

eindeutiger Bezeichner — `bestellnr`

Ein Datenmodell legt fest, *was* als ein Fall zählt und *welche* Merkmale er hat — schon das ist eine Entscheidung.

Zeile 8092 von eben kennt nur: `kundennr = 1074`.
Wer das ist — oder ob es diese Nummer überhaupt *gibt* — steht dort *nicht*.

Bestellungen

bestellnr	kundennr	bestellwert
8092	**1074**	146.05
8093	1471	42.73
8094	1491	129.48

Kunden

kundennr	name	ort
1073	Claudia Wolf	Stuttgart
1074	**Heike Meier**	**Jena**
1075	Niklas Schulze	Dortmund

Beide Tabellen teilen die Spalte `kundennr`: der gemeinsame **Schlüssel**. Um zu prüfen, ob eine Kundennummer *existiert*, braucht man die zweite Tabelle. Genau das machen Sie in der Lernwerkstatt bei den verwaisten Kundennummern.

- **Vollständigkeit** — fehlt etwas?
  leere Kundennummer, kein Bestelldatum
- **Konsistenz** — widerspricht sich etwas?
  dieselbe Bestellung in zwei Systemen verschieden
- **Genauigkeit** — stimmt es mit der Realität?
  Testbestellung über 0 €, Storno als negativer Wert
- **Aktualität** — ist es noch gültig?
  umgezogene Kundin, alte Adresse

Was die vier Dimensionen voraussetzen

Alle vier Fragen setzen voraus, dass klar ist, *was überhaupt als ein Fall zählt*. Genau das ist bei „1.616 Bestellungen“ nicht geklärt — eine Zahl ohne diese **Definition** ist wertlos.

Suchen Sie den Fehler — mit dem Nachbarn

bestellnr	bestelldatum	kundennr	anzahl_pakete	bestellwert	code
8047	2026-01-04	1550	4	83.03	
9596	2026-03-23	9999	1	0.00	
8052	2026-01-06	1119	6	650.25	
8050	2026-01-06	1353	4	173.36	
9604	2026-03-18	1380	2	-129.90	
8053	2026-01-06	1480	2	1039.95	
9605	2026-02-13	9998	2	232.89	
8050	2026-01-07	1353	4	173.36	

Aktivierung — Think–Pair–Share: Acht Zeilen, echte Rohdaten. Mit dem Nachbarn 2 Minuten: Welchen Zeilen trauen Sie nicht — und warum?

Wie kommen solche Fehler hinein?

Genau das haben Sie gerade gefunden — **fast nie durch Schlamperei, meist durch den *Prozess*.** Die vier Fälle von eben, jetzt mit Namen:

- **Doppelerfassung** beim Systemwechsel — dieselbe Bestellung wandert zweimal ins Berichtssystem.
- **Testbestellungen** im Produktivsystem — jemand prüft den Shop, die Zeile bleibt stehen.
- **Stornos** — eine Rücknahme steht als negativer Wert in derselben Tabelle.
- **Verwaiste Verweise** — eine Kundennummer, zu der es keinen Kunden (mehr) gibt.
- **Unterschiedliche Stichtage / Definitionen** — Vertrieb zählt alle Zeilen, Controlling nur bezahlte.

Deshalb ist „Wer hat recht?“ die falsche Frage. Richtig: „Was genau zählt jede Seite?“

12,9 Mio. US-Dollar

durchschnittliche jährliche Kosten schlechter Datenqualität — pro Unternehmen

Genau die vier Fehlerarten, die wir gerade bei Brick-Flow gefunden haben, treiben in echten Unternehmen echte Kosten: falsche Lagerplanung, verpasste Rabattauslösungen, Management-Entscheidungen auf falscher Grundlage.

Die 22 Bestellungen Differenz bei Brick-Flow klingen klein — wie groß es werden kann, zeigt gleich ein echter Fall.

England, Oktober 2020: Positive Corona-Testergebnisse werden in `.xls`-Dateien gesammelt — Format-Grenze: **65.536 Zeilen**. Ist die Datei voll, werden neue Ergebnisse einfach nicht mehr angehängt. *Ohne jede Fehlermeldung.*

15.841 positive Fälle

verschwanden zwischen dem 25. September und dem 2. Oktober 2020 — tagelang nicht in der Statistik, nicht an die Kontaktverfolgung übergeben.

Kein Programmabsturz — ein ungeprüftes Format. Dieselbe Fehlerart wie bei Brick-Flow: Vollständigkeit.

Wie wir die Differenz aufklären: der Plan

Erinnerung an die Frage der Stunde: „Der Vertrieb meldet 1.616 Bestellungen, das Controlling 1.594 — wer hat recht?“

1. **Beide Zählungen reproduzieren.**
   Erst die 1.616 (alle Zeilen), dann sehen, wie man auf 1.594 kommt.
2. **Die Differenz Zeile für Zeile lokalisieren.**
   Doppelte, Testbestellungen, Stornos, verwaiste Nummern — welcher Fehler wie oft?
3. **Entscheiden, welche Definition gilt — und sie aufschreiben.**
   „Wir zählen als Bestellung: …“ — damit die Zahl morgen wieder herauskommt.

Genau diesen Dreischritt machen Sie in der Lernwerkstatt selbst.

Erinnerung an die Frage der Stunde: „Der Vertrieb meldet 1.616 Bestellungen, das Controlling 1.594 — wer hat recht?“

Antwortsatz

Beide zählen richtig, aber nicht dasselbe — der Vertrieb zählt **jede Zeile** der ungeprüften Rohtabelle, das Controlling nur **validierte, eindeutige** Bestellungen. Eine Zahl ohne geprüfte *Definition* ist wertlos.

- Der Vertrieb zählt *jede Zeile* (1.616), das Controlling zieht Duplikate, Testbuchungen, Stornos und verwaiste Verweise ab — bleiben 1.594.
- Nicht alles davon ist *Definitionssache*: Duplikate und verwaiste Verweise sind schlicht Fehler — die zieht man immer ab. Nur bei Testbestellungen und Stornos ist tatsächlich eine Entscheidung nötig, was als „Bestellung“ zählen soll.
- Keiner hat sich verrechnet — der Vertrieb hat nur nie geprüft, was in seiner Zeilenzahl eigentlich alles mitgezählt wird.

Sie zählen zuerst selbst — mit einer *Schleife*, so wie man grundsätzlich zählt: einen Zähler hochzählen, Wert für Wert, an einer kleinen Handvoll Bestellwerte.

- Danach dieselbe Frage auf 1.616 Zeilen — die Schleife funktioniert, aber niemand will das tippen. Genau da übernimmt `pandas` in einer Zeile.
- Mit dem *fehlerbehafteten* Rohdatensatz suchen Sie danach die 22 Bestellungen Differenz selbst: `duplicated`, `isna`, `describe`, `value_counts`.
- Die Lernwerkstatt bearbeiten Sie in der **Digitalkompetenzwoche** im Selbststudium. Das Notebook erklärt jeden Schritt, nennt Kontrollzahlen und enthält aufklappbare Hinweise.

In der Lernwerkstatt suchen Sie die Fehler in den Daten **selbst**.
Dafür brauchen Sie drei kleine Bausteine — die klären wir jetzt kurz.

- **Zahl oder Text?** Eine Zahl, die als Text gespeichert ist, ist ein Datenfehler. Woran erkennt man das?
- **Wahr oder falsch?** Jede Suche nach „doppelt“, „null“ oder „negativ“ ist ein Filter, und der beruht auf `True` oder `False` je Zeile.
- **Fehlermeldungen lesen.** In der Digitalkompetenzwoche arbeiten Sie ohne uns. Wer die Meldung versteht, hilft sich selbst.

Zahl oder Text?

```
print(70 + 98)
print("70" + "98")
```

A 168
B 7098
C eine Fehlermeldung

Aktivierung — Handzeichen: Was gibt die *zweite* Zeile aus? Entscheiden Sie sich für A, B oder C.

Jeder Wert hat einen Datentyp

```
print(type(98.62))          # <class 'float'>   Kommazahl
print(type(3))              # <class 'int'>     Ganzzahl
print(type("B2B"))          # <class 'str'>     Text

text = "120"
print(int(text) + 5)        # 125
```

- Der **Datentyp** sagt, was man mit einem Wert tun darf: Zahlen rechnet man, Texte hängt man aneinander.
- Eine Zahl, die als **Text** gespeichert ist, ist ein Datenfehler: Eine Summe wäre Unsinn, ohne dass eine Fehlermeldung kommt. `int(...)` und `float(...)` wandeln um.

Der Wahrheitswert: True oder False

```python
print(1000 > 500)          # True
print(type(1000 > 500))    # <class 'bool'>
```

bestellnr	bestellwert	bestellwert > 1000
8001	98.62	False
8017	1415.39	True
8023	1101.67	True

- Jeder **Vergleich** liefert einen Wahrheitswert: `True` oder `False`.
- Bei einer ganzen Spalte entsteht ein Wahrheitswert *je Zeile*. Der Filter behält nur die Zeilen mit `True`.

```python
# beide Bedingungen muessen stimmen
df[(df["bestellwert"] > 1000) & (df["anzahl_pakete"] > 3)]

# eine reicht
df[(df["bestellwert"] > 1000) | (df["anzahl_pakete"] > 3)]

# mit and: ValueError
df[(df["bestellwert"] > 1000) and (df["anzahl_pakete"] > 3)]
# ValueError: The truth value of a Series is ambiguous.
```

- Bei einer *ganzen Spalte* verknüpft man mit & (und), | (oder) und ~ (nicht).
- Jede einzelne Bedingung steht in runden Klammern.

Drei Fehler, drei Meldungen

```python
# Fall 1
umsatz_gesamt = 12345.67
print(umsatz)
# NameError: name 'umsatz' is not defined

# Fall 2
preis = 19.90
print("Der Preis betraegt " + preis + " Euro")
# TypeError: can only concatenate str (not "float") to str

# Fall 3
df["bestelwert"].head()
# KeyError: 'bestelwert'
```

Aktivierung — Think–Pair–Share: Zu jedem Fall: Fehlerart, Ursache, Behebung. Zwei Minuten mit dem Nachbarn.

Diese Funktion soll zu einem Bestellwert sagen: **gültig** (über 0 €), **Test** (genau 0 €) oder **Storno** (unter 0 €). Sie enthält mehrere Fehler.

```
1  def bestellstatus(bestellwert):
2      if bestellwert > 0:
3      status = "gueltig"
4      else:
5          if wert = 0:
6              status = "Test"
7          else:
8              status = "Storno"
9      return Status
```

Aktivierung — Think–Pair–Share: Wie viele Fehler finden Sie, und wie muss die Zeile jeweils heißen? Zwei Minuten mit dem Nachbarn.

- Daten sind **strukturiert** oder **unstrukturiert**; **Metadaten** sagen, was eine Zahl bedeutet.
- Datenqualität hat vier Fragen: **Vollständigkeit, Konsistenz, Genauigkeit, Aktualität**.
- Fehler entstehen meist im **Prozess**, nicht durch Schlamperei.
- **Eine Zahl ohne ihre Definition ist wertlos.**
- Python: Eine Zahl kann als **Text** gespeichert sein, Filter beruhen auf **Wahr/Falsch**, und eine **Fehlermeldung** ist ein Hinweis.

Nach der Digitalkompetenzwoche: Eine Kundin heiratet und zieht um — was ändert sich wo? Wir verbinden Tabellen.

- **Gabler Wirtschaftslexikon** — Stichworte „Datenqualität“, „Metadaten“, „strukturierte Daten“, `https://wirtschaftslexikon.gabler.de`.
- **DAMA International** — *DAMA-DMBOK: Data Management Body of Knowledge* (2. Aufl., 2017): Kap. Data Quality (Dimensionen Vollständigkeit, Konsistenz, Genauigkeit, Aktualität) und Kap. Data Modeling and Design (Zeile/Spalte/Schlüssel, mehrere verbundene Tabellen).
- **Gartner** — „Data Quality: Why It Matters and How to Achieve It“, `https://www.gartner.com/en/data-analytics/topics/data-quality` — Kosten schlechter Datenqualität (Erhebung 2020).
- **Digital Health / The Register** — Fallbeispiel Public Health England, Oktober 2020: 15.841 verlorene Covid-Fälle durch Excel-Zeilenlimit (`.xls`, 65.536 Zeilen). `https://www.digitalhealth.net/2020/10/how-using-excel-may-have-caused-thousands-of-unreported-covid-cases/` `https://www.theregister.com/2020/10/05/excel_england_coronavirus_contact_error/`.

- **Brick-Flow AG** — Fallunternehmen des Kurses, `https://brick-flow.ag/`. Der Rohdatensatz `brickflow_bestellungen_roh.csv` und die Zahlen 1.616 / 1.594 sind didaktische Konstruktionen (dokumentiert in `use-case/BRICKFLOW.md` §9.9).