Wirtschaftsinformatik

V05 — Datenbanken: Grundideen

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

Hochschule Bochum · Fachbereich Wirtschaft

Wintersemester 2026/27

Rückblick: Letzte Woche

- Frage: *Eine Kundin heiratet und zieht um — was muss geändert werden, und an wie vielen Stellen?*
- Antwort: **An genau einer Stelle** — wenn die Daten richtig aufgeteilt sind. Dafür gibt es Tabellen mit Schlüsseln.
- Zum Schluss ging es um **inner** und **left**: Ein `inner`-Join behält nur, was auf *beiden* Seiten vorkommt — alles andere fällt *stillschweigend* raus. Heute lesen wir am Datenbankmodell, wann das passiert.

Welche Marketing-Kampagne hat sich gelohnt?

Aktivierung — Diskussion: Woran würden Sie erkennen, dass sich eine Kampagne gelohnt hat? Was müssten wir dafür wissen?

Warum eine Datenbank?

Viele arbeiten gleichzeitig

Online-Shop, ERP, Controlling und Logistik schreiben und lesen im selben Moment.

Die Daten haben Regeln

Jede Kundennummer gibt es nur einmal, jede Bestellung hat ein Datum, ein Verweis zeigt auf etwas, das existiert.

Große Mengen bleiben schnell

Bis gut eine Million Zeilen passen in Excel, mehrere Millionen nicht mehr.

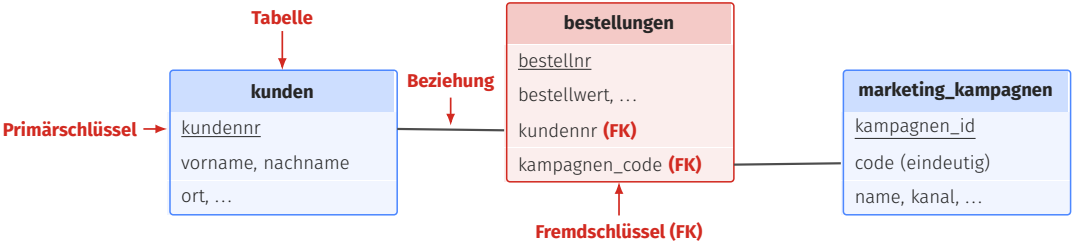
Eine Abfragesprache

Mit **SQL** holt man Daten aus einer Datenbank. Wir erwähnen sie hier nur, in der Lernwerkstatt nutzen wir sie kurz zum Laden.

Heute geht es um den *Aufbau* einer Datenbank und darum, ihn zu lesen.

	bisher (CSV)	ab heute (`brickflow.db`)
Bestellungen	1.616	35.350
Kunden	550	2.656
Kampagnen	4	48
Zeitraum	Jan.–Juni 2026	2015–2026

Drei Tabellen brauchen wir heute: `kunden`, `bestellungen` und `marketing_kampagnen`. Dasselbe Unternehmen, nur elf Jahre statt fünf Monate. Und wie in jedem echten System heißen manche Spalten anders als in den CSV-Dateien: `kampagnen_code` statt `code`, dazu `vorname` und `nachname`.



Ein **Fremdschlüssel** verweist auf den **Primärschlüssel** einer anderen Tabelle. Die **Linie** verbindet beide.

Die Zahlen an den Linien



1 = genau einer
Jede Bestellung gehört zu **genau einer** Kundin.

0..n = keiner, einer oder viele
Eine Kundin hat **keine, eine oder viele** Bestellungen.

0..n = keiner, einer oder viele
Eine Kampagne steht hinter **keiner, einer oder vielen** Bestellungen.

0..1 = keiner oder einer
Eine Bestellung hat **keine oder eine** Kampagne.

Eine Zahl an einer Linie sagt, wie viele Zeilen der Tabelle, neben der sie steht, zu *einer* Zeile der Tabelle am anderen Ende der Linie gehören können.



Drei Fragen, beantwortet allein mit dem Modell und den Zahlen:

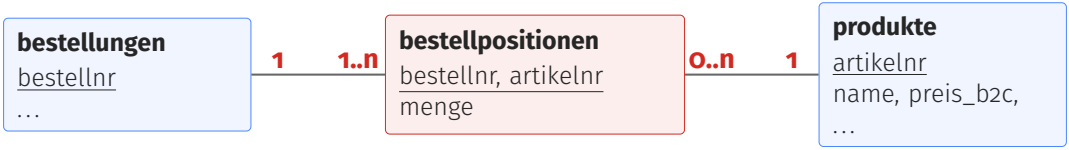
- **a)** Kann es eine Bestellung geben, deren Kundin gar nicht in der Kundentabelle steht?
- **b)** Kann es eine Kundin ohne eine einzige Bestellung geben?
- **c)** Kann eine Bestellung zu zwei Kampagnen gleichzeitig gehören?

> **Aktivierung — Wortmeldung:** Sagen Sie jeweils ja oder nein und zeigen Sie auf die Zahl im Diagramm, die es verrät.

Beziehung	**Das Modell sagt**	**Für die Daten heißt das**
Bestellung → Kundin	genau eine (`1`)	Jede Bestellung hat eine Kundin.
Kundin → Bestellung	keine bis viele (`0..n`)	Es kann Kundinnen **ohne** Bestellung geben.
Bestellung → Kampagne	keine oder eine (`0..1`)	Viele Bestellungen haben **keine** Kampagne.

Die Kampagne ist bei einer Bestellung *optional*, die Kundin ist es nicht. In unserer Datenbank trifft das zu: Nur ein kleiner Teil der Bestellungen trägt einen Kampagnencode.

Eine Bestellung kann mehrere Produkte enthalten, und ein Produkt steckt in vielen Bestellungen. Das ist eine **n:m-Beziehung**. Eine Datenbank löst sie mit einer dritten Tabelle:



Unsere Datenbank hat diese dritte Tabelle nicht: Eine Bestellung kennt nur Bestellwert und Paketzahl, keine Artikelnummer. Der Artikelkatalog `produkte` hängt deshalb an keiner Bestellung, und wir können mit ihr keine Frage nach Produkten beantworten.

Regel im Modell	Sie verhindert	Bei Brick-Flow
Primärschlüssel eindeutig	zwei Kundinnen mit derselben Nummer	`kundennr` nie doppelt
Pflichtfeld	eine Bestellung ohne Datum	jede Bestellung hat Datum und Wert
Fremdschlüssel muss existieren	eine Bestellung für eine unbekannte Kundin	0 Bestellungen ohne Kundin

Das Modell ist nicht nur eine Zeichnung. Es ist ein Regelwerk, das die Datenbank durchsetzt.

Wie wir die Frage beantworten: der Plan

Erinnerung an die Frage der Stunde: „Welche Marketing-Kampagne hat sich gelohnt?“

1. **Welche Tabellen brauchen wir, und wie hängen sie zusammen?** `bestellungen` und `marketing_kampagnen`, verbunden über den Kampagnencode.
2. **???**
3. **Womit vergleichen wir?** Zeitraum davor? Vorjahr? Bestellungen ohne Kampagnencode?

Schritt 2 fehlt — ab dieser Woche liefert die Vorlesung den Plan nicht mehr vollständig.

Bestellungen (Ausschnitt)

bestellnr	bestellwert	kampagnen_code
43913	205.51	
43914	909.96	
43915	112.72	
43916	56.72	
43917	430.75	NEUJAHR26
43918	131.25	NEUJAHR26

Kampagnen (Ausschnitt)

code	name
NEUJAHR26	Neujahrsaktion 2026

Verbinden mit how="inner"

bestellnr	name
43917	Neujahrsaktion 2026
43918	Neujahrsaktion 2026

Verbinden mit how="left"

bestellnr	name
43913	
43914	
43915	
43916	
43917	Neujahrsaktion 2026
43918	Neujahrsaktion 2026

Der **innere** Join (`inner`) behält nur Zeilen mit Treffer: Vier von sechs Bestellungen verschwinden, **ohne Fehlermeldung**. Das Modell hätte es verraten: `0..1` heißt, nicht jede Bestellung hat eine Kampagne. In der ganzen Datenbank sind es 93 %.

Aber was heisst eigentlich „hat sich gelohnt“?

Naive Antwort: die Kampagne mit dem höchsten Umsatz (bei uns: **NEUJAHR26**).

Die Frage ist	**Dafür braucht man einen Vergleichsmaßstab**
Mehr Bestellungen?	Bestellungen im Zeitraum *davor* oder im Vorjahr
Größere Bestellungen?	Durchschnittlicher Bestellwert *ohne* Kampagnencode
Mehr Kundinnen?	Neue Kundinnen gegenüber ohnehin bestellenden

Ein hoher Umsatz allein sagt noch nicht, dass die Kampagne etwas bewirkt hat: Entscheidend ist, *womit* man vergleicht.

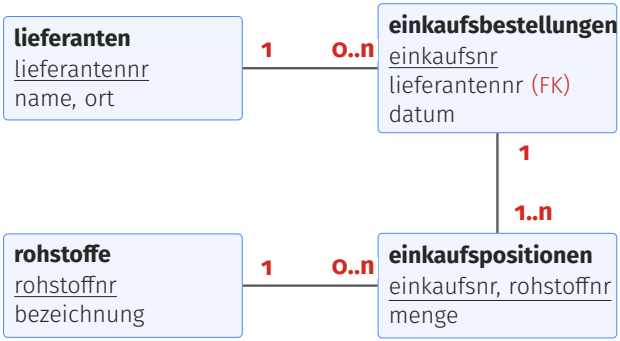
Erinnerung an die Frage der Stunde: „Welche Marketing-Kampagne hat sich gelohnt?“

Antwortsatz

NEUJAHR26 brachte den höchsten Umsatz. Ob sich die Kampagne gelohnt hat, sagt das noch nicht: Dafür braucht man einen Vergleichsmaßstab, und das Verbinden darf die Bestellungen *ohne* Kampagne nicht stillschweigend wegwerfen. Das Modell hat es vorher verraten: Die Kampagne ist bei einer Bestellung optional (`0..1`).

- Ein `inner`-Join auf die Kampagnentabelle hätte 93 % aller Bestellungen verworfen, und die Zahl hätte trotzdem plausibel ausgesehen.
- „Hat sich gelohnt“ bleibt eine Definitionsfrage: ohne Vergleichsmaßstab (Schritt 3 aus dem Plan) keine Antwort.

Aufgabe: ein neues Modell lesen



a) Kann ein Lieferant in der Datenbank stehen, ohne dass wir je bei ihm bestellt haben?

b) Kann es eine Einkaufsbestellung ohne eine einzige Position geben?

c) Kann derselbe Rohstoff in mehreren Bestellungen vorkommen?

d) Über welche Tabellen müssen Sie gehen, um zu einer Position den Lieferanten zu nennen?

Sie laden die Tabellen mit einer kurzen SQL-Abfrage aus `brickflow.db` in `pandas` und arbeiten dort weiter: mit `merge`, wie in der letzten Woche.

Dabei verbinden Sie `bestellungen` mit `marketing_kampagnen`, und zwar so, dass keine Bestellung verloren geht. Das Modell von heute sagt Ihnen, wie.

- Ein **Datenbankmodell** lesen heißt: Kästen (Tabellen), Schlüssel, Linien und die Zahlen an den Linien verstehen.
- `0..1` und `0..n` sagen, dass eine Beziehung *optional* ist. Dann gibt es Zeilen ohne Partner.
- Eine **n:m-Beziehung** braucht eine Zwischentabelle. Fehlt sie, lässt sich eine Frage mit dieser Datenbank nicht beantworten.
- Ein **inner-Join** verwirft alles ohne Partner, ohne Fehlermeldung. Das Modell und ein Blick auf die Zeilenzahl decken es auf.

Nächste Woche: Wie viel gibt ein typischer Kunde aus? Warum der Mittelwert manchmal in die Irre führt.

- **Gabler Wirtschaftslexikon** — Stichwort „SQL“, `https://wirtschaftslexikon.gabler.de`.
- **Microsoft** — „Excel specifications and limits“, `https://support.microsoft.com/en-us/office/excel-specifications-and-limits-1672b34d-7043-467e-8e27-269d656771c3` — 1.048.576 Zeilen je Tabellenblatt (.xlsx).
- **Brick-Flow AG** — Fallunternehmen des Kurses, `https://brick-flow.ag/`. Alle Zahlen dieser Folien (35.350 Bestellungen, 2.347/33.003 mit/ohne Kampagnencode, Umsätze) sind reale Auszüge aus `use-case/data/brickflow.db`, nachgerechnet.