LU08b: Warum Fremdschlüssel bzw. mehrere Tabellen?

Ausgangslage

Naheliegend ist: **Alles in eine Tabelle** (Post + Autor + Kategorie). In echten Blogs (z. B. WordPress) führt das aber zu **Wiederholungen**, **Fehlern** und **hohem Wartungsaufwand**.

Wir bleiben beim Reiseblog-Beispiel von der letzten Seite We Travel The World Blog.

Beispiel: Alles in einer Tabelle (schlechte Idee)

Daten einfügen (nur eine Kategorie pro Zeile möglich):

```
INSERT INTO blog_posts
(post_title, post_content, author_name, author_email,
category_name, created_at)
VALUES
('Hasselt - 10 Highlights',
'Kurzguide: Die schönsten Ecken von Hasselt ...',
'Martin Merten', 'martin@wetraveltheworld.de',
'Städtereise', '2025-05-07 10:15:00'),

('Utrecht - 10 Sehenswürdigkeiten',
'Cafés, Grachten und Restaurant-Tipps ...',
'Martin Merten', 'martin@wetraveltheworld.de',
'Städtereise', '2025-06-05 09:30:00'),

('Lissabon - 8 Tipps zu den wichtigsten Sehenswürdigkeiten',
'Aussichtspunkte, Viertel und Highlights ...',
'Caro Steig', 'caro@wetraveltheworld.de', 'Portugal',
```

'2025-03-21 08:40:00');

Warum nur eine Kategorie pro Zeile? Komma-Listen wirken bequem, sind aber ungünstig:

- **Keine Prüfung / kein FK¹¹.** Die Datenbank (DB) kann bei «Belgien, Städtereise» **nicht** prüfen, ob diese Kategorien wirklich existieren. ²⁾ → Tippfehler bleiben unbemerkt ³⁾.
- Unzuverlässiges Filtern. Suchen mit LIKE liefern leicht Teiltreffer/Varianten. Beispiel: WHERE category_name LIKE '%Guinea%' trifft auch «Equatorial Guinea», «Guinea», «Guinea-Bissau» und «Papua New Guinea» – vier verschiedene Länder.
- **Schwierig auszuwerten & langsam.** Zählen/Gruppieren erfordert Strings⁴⁾ zu zerlegen; darauf kann die DB nicht sinnvoll indexieren⁵⁾.

Besser: Pro Zeile **eine** Kategorie. Für mehrere Kategorien pro Post (N:M-Beziehung) verwenden wir eine **Zwischentabelle** *post_category*.

Auszug:

id	post_title	author_name	author_email	category_name	created_at
1	Hasselt - 10 Highlights	Martin Merten	martin@wetraveltheworld.de	Städtereise	2025-05-07 10:15:00
2	Utrecht - 10 Sehenswürdigkeiten	Martin Merten	martin@wetraveltheworld.de	Städtereise	2025-06-05 09:30:00
3	Lissabon – 8 Tipps zu den wichtigsten Sehenswürdigkeiten	Caro Steig	caro@wetraveltheworld.de	Portugal	2025-03-21 08:40:00

Probleme auf einen Blick:

- **Redundanz**: Autorname/E-Mail wiederholen sich bei mehreren Posts.
- **Fehleranfällig**: Kategorienamen können unterschiedlich geschrieben werden.
- **Aufwendig**: E-Mail-Wechsel eines Autors → alle Zeilen suchen und ändern.
- Nur eine Kategorie möglich: Idealerweise möchten wir aber mehrere Kategorien pro Blog-Post vergeben – z.B. beim Blog-Post «Utrecht – 10 Sehenswürdigkeiten»: Niederlande, Städtereise.

Tippfehler in Kategorie: sichtbare Folgen

Ein fehlender Buchstabe reicht: Städtereise vs. Stätdereise.

https://wiki.bzz.ch/ Printed on 2025/11/25 18:07

```
INSERT INTO blog_posts
(post_title, post_content, author_name, author_email,
category_name, created_at)
VALUES
('Maastricht an einem Tag',
'Spaziergang, Restaurants, Altstadt ...',
'Caro Steig', 'caro@wetraveltheworld.de', 'Stätdereise',
'2025-06-12 11:05:00'); -- Tippfehler!
```

Direkte Folgen in Abfragen:

Abfrage	Zweck	Effekt bei Tippfehler
SELECT DISTINCT category_name FROM blog_posts ORDER BY category_name;	Kategorienliste (Navigation/Filter)	Liste zeigt zwei Einträge: <i>Stätdereise</i> und <i>Städtereise</i> .
SELECT id, post_title FROM blog_posts WHERE category_name = 'Städtereise';	Beiträge in "Städtereise"	Der Datensatz mit Stätdereise (dt vertauscht) fehlt im Resultat.

Teil-Update: uneinheitliche E-Mail

Nur **eine** von mehreren Zeilen eines Autors wird geändert → inkonsistente Daten.

```
UPDATE blog_posts
SET author_email = 'martin.new@wetraveltheworld.de'
WHERE id = 1;

SELECT id, post_title, author_name, author_email
FROM blog_posts
WHERE author_name = 'Martin Merten';
```

Ergebnis:

id	post_title	author_name	author_email
1	Hasselt - 10 Highlights	Martin Merten	martin.new@wetraveltheworld.de
2	Utrecht - 10 Sehenswürdigkeiten	Martin Merten	martin@wetraveltheworld.de

Gleicher Autor, unterschiedliche E-Mail → Daten sind inkonsistent.

Warum eine N:M-Beziehung (Posts ↔ Kategorien) auflösen?

In einer relationalen Tabelle beschreibt jede Zeile genau ein Ding (entweder einen Post oder eine Kategorie). Ein Fremdschlüssel in einer Zeile kann deshalb nur auf ein anderes Ding zeigen.

Bei N:M gilt jedoch: Ein Post gehört zu 0...n Kategorien und eine Kategorie enthält 0...n Posts. \rightarrow Ein einzelnes FK-Feld reicht dafür nicht aus:

- FK in posts nur als category id: Pro Post wäre nur eine Kategorie möglich. Mehrere Kategorien erzwingen Duplikate desselben Posts - Änderungen und Löschungen werden fehleranfällig.
- FK in categories nur als post id: Pro Kategorie wäre nur ein Post möglich. Dadurch müsste man Kategorien duplizieren – gleiches Problem wie bei obigen Punkt.
- Kommaliste in einer Spalte (z. B. Belgien, Städtereise): In einer Zelle stehen mehrere Werte. Die Datenbank kann nicht prüfen, ob diese Kategorien existieren (kein FK), Abfragen werden unzuverlässig und langsam.

Lösung: die Zwischentabelle post category. Wir trennen die Objekte (posts, categories) und speichern jede einzelne Zuordnung als eigene Zeile in post category - genau ein Paar (post id, category id) pro Zeile. So bleibt jede Beziehung eindeutig modelliert, Primärschlüssel bleiben eindeutig, Fremdschlüssel sind prüfbar, und Abfragen bleiben klar und performant.

Ausblick

Auf der nächsten Seite bauen wir genau dieses Mehrtabellen-Schema mit Fremdschlüsseln auf und füllen es mit den obigen Reiseblog-Beispieldaten.

Foreign Key = Fremdschlüssel

Wenn Kategorien und Posts in separate Tabellen liegen und per Fremdschlüssel verbunden sind, kontrolliert die DB beim Speichern, ob «Belgien» in der Kategorien-Tabelle vorhanden ist.

z. B. «Belgien, Städtereisen» statt «Belgien, Städtereise»

Zeichenketten

Index: Ein Index ist wie ein Inhaltsverzeichnis der Datenbank. Er beschleunigt Suchen/Sortieren auf einzelnen Spaltenwerten. Bei Komma-Listen stecken mehrere Werte in einem Feld – darauf lässt sich kein brauchbarer Index aufbauen: zudem können Suchmuster wie LIKE '%Wort%' einen vorhandenen Index oft nicht nutzen.

https://wiki.bzz.ch/ - BZZ - Modulwiki

Permanent link:

https://wiki.bzz.ch/modul/m290 guko/learningunits/lu08/theorie/b fk-grundlagen?rev=1760820126

Last update: 2025/10/18 22:42



https://wiki.bzz.ch/ Printed on 2025/11/25 18:07