LU05b - SQL-DDL: Constraints (Einschränkungen)

Lernziele

Sie lernen in dieser Lektion:

- was Constraints (Einschränkungen) sind
- warum sie in Datenbanken wichtig sind
- wie Sie die wichtigsten Constraints direkt beim Erstellen einer Tabelle setzen

Was sind Constraints?

Constraints legen Regeln für Spalten in einer Tabelle fest. Sie sorgen dafür, dass **Daten korrekt, vollständig und widerspruchsfrei** gespeichert werden.

Warum brauchen wir Constraints?

- verhindern fehlerhafte oder unvollständige Einträge
- schützen vor doppelten IDs oder leeren Pflichtfeldern
- stellen sicher, dass Daten logisch und konsistent bleiben

Beispiele aus der Praxis:

- Jeder Schüler/jede Schülerin braucht eine eindeutige Schülernummer (ID)
- Der Name eines Kunden darf nicht leer sein
- Die E-Mail-Adresse eines Benutzers soll nur einmal vorkommen

Constraints helfen also, **Datenqualität** sicherzustellen.

PRIMARY KEY - Eindeutige ID

Der Primärschlüssel (*PRIMARY KEY*) identifiziert jeden Datensatz eindeutig. Jede Tabelle kann **nur einen Primärschlüssel** haben.

```
CREATE TABLE student (
   student_id INT PRIMARY KEY,
   name VARCHAR(50),
);
```

Wirkung: Keine zwei Datensätze dürfen dieselbe student id haben. Die student id darf ausserdem

nicht leer (NULL) sein.

AUTO_INCREMENT - Automatisch hochzählen

Das Attribut AUTO_INCREMENT sorgt dafür, dass bei jedem neuen Datensatz der Wert dieser Spalte automatisch um **1 erhöht** wird. Es wird fast immer zusammen mit dem **Primärschlüssel** verwendet, um automatisch eine **eindeutige ID** zu vergeben – ganz ohne manuelles Eingreifen.

```
Beispiel:

CREATE TABLE kunden (
   kunden_id INT AUTO_INCREMENT PRIMARY KEY,
   name VARCHAR(50) NOT NULL
);

Fügt man nun Datensätze ein, vergibt MySQL automatisch fortlaufende Nummern für kunden_id: 1, 2, 3, ...
```

NOT NULL

Mit NOT NULL legen Sie fest, dass ein Feld **nicht leer** bleiben darf.

```
CREATE TABLE ort (
  plz CHAR(4) NOT NULL,
  name VARCHAR(50) NOT NULL,
);
```

Wirkung: Versucht jemand, ohne PLZ oder Name einen Datensatz einzufügen, meldet die Datenbank einen Fehler.

UNIQUE - Eindeutige Werte erzwingen

Mit dem UNIQUE-Constraint stellen Sie sicher, dass in einer Spalte **kein Wert doppelt vorkommen** darf.

Im Gegensatz zum Primärschlüssel (PRIMARY KEY) kann es in einer Tabelle **mehrere UNIQUE-Spalten** geben. So können Sie zum Beispiel sicherstellen, dass keine zwei Benutzer:innen dieselbe E-Mail-Adresse haben.

https://wiki.bzz.ch/ Printed on 2025/11/21 11:59

Beispiel:

```
CREATE TABLE benutzer (
  benutzer_id INT AUTO_INCREMENT PRIMARY KEY,
  email VARCHAR(100) UNIQUE,
  name VARCHAR(50)
);
```

In diesem Beispiel muss jede *email* **eindeutig** sein. Versucht man, eine E-Mail-Adresse zweimal einzutragen, wird ein Fehler ausgegeben.

□ Übersicht

Constraint	Bedeutung
NOT NULL	Spalte darf nicht leer bleiben
PRIMARY KEY	Eindeutige ID für jeden Datensatz
AUTO_INCREMENT	Automatische laufende Nummer für neue Datensätze
NOT NULL	Spalte darf nicht leer bleiben
UNIQUE	Wert darf in einer Spalte nur einmal vorkommen



From:

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

Permanent link:

https://wiki.bzz.ch/modul/m290_guko/learningunits/lu05/theorie/b_constraints?rev=1757870218

Last update: 2025/09/14 19:16

