Lernskript: Daten verknüpfen mit SQL `JOIN`s

`JOIN`-Klauseln sind das mächtigste Werkzeug in SQL. Sie ermöglichen es, zusammengehörige Informationen aus mehreren Tabellen in einer einzigen Abfrage zu kombinieren.

Teil 1: Das Setup - Zwei verknüpfte Tabellen

Um `JOIN`s zu verstehen, benötigen wir mindestens zwei Tabellen, die eine Beziehung zueinander haben. In unserem Fall sind das `PRODUKTE` und `AUTOREN`.

- `AUTOREN` **Tabelle:** Enthält `ID` und `NAME` jedes Autors.
- `PRODUKTE` **Tabelle:** Enthält `ID`, `TITEL`, etc. und eine spezielle Spalte `AUTOR_ID`.
- **Die Beziehung:** Die `AUTOR_ID`-Spalte in `PRODUKTE` ist ein **Fremdschlüssel (Foreign Key)**, der auf den **Primärschlüssel (Primary Key)** `ID` in der `AUTOREN`-Tabelle verweist. Dies ist die "Brücke" zwischen den Tabellen.

Teil 2: Die verschiedenen `JOIN`-Typen

1. `INNER JOIN` (Die Schnittmenge)

- **Zweck:** Gibt nur die Datensätze zurück, für die es in **beiden** Tabellen eine Übereinstimmung basierend auf der `ON`-Bedingung gibt.
- **Analogie:** Finde alle Bücher, die einen zugeordneten Autor haben.
- **Code-Beispiel:**

```
-- Wähle Buchtitel und Autorenname
SELECT
    P.TITEL,
    A.NAME
FROM
    PRODUKTE P -- 'P' ist ein Alias für PRODUKTE
INNER JOIN AUTOREN A ON P.AUTOR_ID = A.ID; -- 'A' ist ein Alias für AUTOREN

  Ergebnis: Nur die Datensätze, die auf beiden Seiten der "Brücke" einen
Partner finden. Produkte ohne Autor oder Autoren ohne Buch werden nicht
angezeigt.
```

2. LEFT JOIN (Alles von Links)

   Zweck: Gibt alle Datensätze aus der linken Tabelle (die nach FROM) zurück, plus die passenden Datensätze aus der rechten Tabelle.

   Analogie: Zeige mir alle Produkte, und falls ein Produkt einen Autor hat, zeige dessen Namen an.

SELECT P.TITEL, A.NAME FROM PRODUKTE P LEFT JOIN AUTOREN A ON P.AUTOR_ID = A.ID;

Ergebnis: Die Ergebnisliste enthält alle Produkte. Bei Produkten ohne zugeordneten Autor (z.B. ein Laptop) wird die Spalte A.NAME mit NULL aufgefüllt.

3. RIGHT JOIN (Alles von Rechts)

   Zweck: Gibt alle Datensätze aus der rechten Tabelle (die nach JOIN) zurück, plus die passenden Datensätze aus der linken Tabelle.

   Analogie: Zeige mir alle Autoren, und falls ein Autor Bücher in unserem Katalog hat, zeige deren Titel an.

   SELECT P.TITEL, A.NAME FROM PRODUKTE P RIGHT JOIN AUTOREN A ON P.AUTOR_ID = A.ID;

   Ergebnis: Die Ergebnisliste enthält alle Autoren. Bei Autoren, die (noch) kein Buch im Katalog haben, wird die Spalte P.TITEL mit NULL aufgefüllt.

Teil 3: Vertiefung (JavaMasta's Profi-Tipps)

```
Aliase sind dein Freund: Die Verwendung von kurzen Aliasen für Tabellennamen (z.B. PRODUKTE P) ist absolute Best Practice. Es macht komplexe Abfragen mit vielen JOINs deutlich kürzer und lesbarer.

LEFT JOIN ist der Standard: In der Praxis verwenden Entwickler zu 95% INNER JOIN und LEFT JOIN. Ein RIGHT JOIN kann fast immer durch einen LEFT JOIN ersetzt werden, indem man die Reihenfolge der Tabellen in der FROM- und JOIN-Klausel vertauscht. Viele Teams bevorzugen LEFT JOINs, da sie oft als intuitiver empfunden werden.

FULL OUTER JOIN: Es gibt noch einen weiteren Typ (FULL OUTER JOIN), der alle Zeilen aus beiden Tabellen zurückgibt und mit NULLs auffüllt, wo es keine Übereinstimmung gibt. Dieser wird jedoch seltener benötigt.
```