

2.3.2 Datenbanksysteme II

Lerninhalte 232-10 SQL-Abfragen über mehrere Tabellen

SQL-Abfragen über mehrere Tabellen

Bei der Datenbankoperation Verbund (engl. JOIN) werden zwei oder mehrere Relationen (Tabellen) zu einer neuen Relation verknüpft. Die Ergebnisrelation liegt dann wieder in Form einer virtuellen Tabelle im Arbeitsspeicher vor. In diesem Beispiel aus der Datenbank RC Wildbach wird ermittelt, wie oft jeder Guide im Einsatz war.

BuchungsNr	KdNr	BootNr	Datum	StartNr	GuideNr
0	42	3	01.05.18	2	4
1	90	3	01.05.18	1	5
2	78	5	02.05.18	0	0
3	29	2	02.05.18	2	2
4	78	2	02.05.18	1	6
5	76	5	03.05.18	1	5
6	10	4	03.05.18	2	2
7	88	5	03.05.18	2	2
8	80	0	03.05.18	2	3
9	43	5	03.05.18	0	5
10	73	5	04.05.18	0	6
11	34	5	04.05.18	1	1
12	70	4	06.05.18	2	4
13	01	2	06.05.18	2	2

Abb. 01: Tabelle *tblBuchungen* aus *RC-Wildbach-final.odt*

GuideNr	Nachname	Vorname	Mobil	E-Mail
0	Hintermair	Sepp	+49 156 9732543	s.hintermair@mail.xyz
1	Santini	Luigi	+39 235 4478992	ls@mail.xyz
2	Gilliusson	Petra	+46 528 4522541	p.gilliusson@mail.xyz
3	Holmbom	Sofia	+46 326 6312417	s.holmbom@mail.xyz
4	Lundholm	Per	+46 114 3666547	per.lundholm@mail.xyz
5	Eriksen	Oyvind	+46 114 5547889	oyvind.eriksen@mail.xyz
6	Jönsson	Eva	+46 811 4468341	eva.jonsson@mail.xyz

Abb. 02: Tabelle *tblGuides* aus *RC-Wildbach-final.odt*

```
SELECT "tblGuides"."Vorname", COUNT( "tblBuchungen"."GuideNr" ) "Anzahl Buchungen"
FROM "tblBuchungen" INNER JOIN "tblGuides"
ON "tblBuchungen"."GuideNr" = "tblGuides"."GuideNr"
GROUP BY "tblGuides"."Vorname"
ORDER BY "Anzahl Buchungen" DESC
```

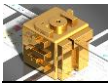
Vorname	Anzahl Buchungen
Sofia	166
Oyvind	149
Luigi	145
Eva	140
Sepp	140
Per	133
Petra	127

Datensatz | von 7

```
SELECT "tblGuides"."Vorname", COUNT( "tblBuchungen"."GuideNr" ) "Anzahl Buchungen"
FROM "tblBuchungen", "tblGuides"
WHERE "tblBuchungen"."GuideNr" = "tblGuides"."GuideNr"
GROUP BY "tblGuides"."Vorname"
ORDER BY "Anzahl Buchungen" DESC
```

Alternative mit WHERE !

Abb. 03: Virtuelle Ergebnistabelle aus der Verbund-Operation (LibreOffice Base: Abfragen SQL-Entwurf)



Erläuterung des Beispiels zum Einsatz der Guides

Abfragen über mehrere Datenbanktabellen werden über sogenannte JOINS realisiert. Die Vornamen der Guides sind in der Tabelle *tblGuides* gespeichert. Die Häufigkeit ihres jeweiligen Einsatzes kann man mit Hilfe der COUNT-Funktion in der Tabelle *tblBuchungen* ermitteln.

Das Datenbankmanagementsystem (DBMS) bildet zunächst mit **JOIN** das **Kreuzprodukt** der zwei Datenbanktabellen. Anschließend filtert es das Ergebnis aufgrund der definierten **Selektionsbedingung** von Primär- und Fremdschlüssel. Die Bedingung gilt als erfüllt, wenn der ausgewählte **Fremdschlüssel** (hier: *tblBuchungen.GuideNr*) der einen Tabelle mit dem **Primärschlüssel** (hier: *tblGuides.GuideNr*) der anderen Tabelle übereinstimmt (**Operator**: =). Es werden somit nur die Datensätze ausgegeben, die gemeinsame Werte enthalten.

Der **INNER JOIN** hebt sich dabei von allen anderen JOIN-Typen durch eine minimale Ergebnismenge ab. Das Resultat ist eine Ergebnistabelle (engl. *View*) ohne Nullwerte¹.

Die Verknüpfung der einzelnen Datensätze aus den zwei Tabellen läuft in der alternativen Schreibweise über die **WHERE**-Klausel. Hier taucht zwar kein JOIN auf, es ist aber einer. Übrigens, das GROUP BY wird wegen der COUNT-Funktion benötigt, weil das so vorgeschrieben ist.

JOIN-Typen als Mengendiagramm

Die verschiedenen Möglichkeiten für JOINS können recht anschaulich über Mengendiagramme dargestellt werden. Hier ein paar grundlegende Beispiele:

INNER JOIN

Bei einem INNER JOIN handelt es sich um eine Schnittmenge der Datensätze aus beiden Tabellen, die die Selektionsbedingung erfüllen.

OUTER JOIN

Bei einem OUTER JOIN handelt es sich um eine Vereinigungsmenge der Datensätze aus beiden Tabellen.

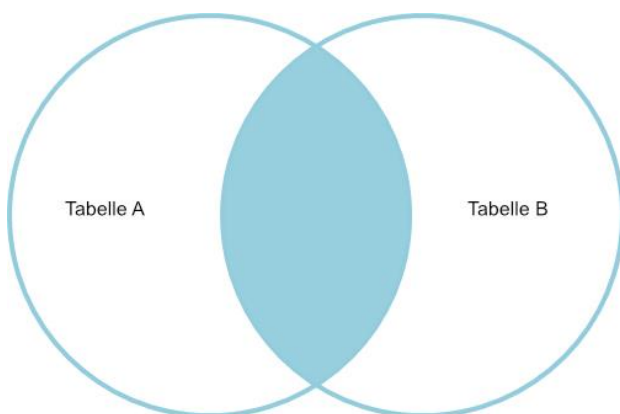


Abb. 04: INNER JOIN

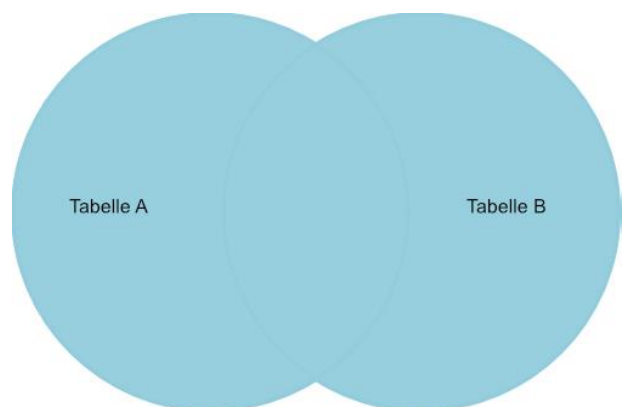
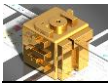


Abb. 05: OUTER JOIN

¹ Da der INNER JOIN der wichtigste SQL-JOIN ist, kannst du das Schlüsselwort „INNER“ bei Bedarf auch weglassen.



2.3.2 Datenbanksysteme II

Lerninhalte 232-10 SQL-Abfragen über mehrere Tabellen

LEFT OUTER JOIN

Bei einem LEFT OUTER JOIN werden alle Datensätze aus der Tabelle A und der Tabelle B zusammengeführt, die die Selektionsbedingung erfüllen, sowie alle Elemente aus der linken Tabelle.

RIGHT OUTER JOIN

Bei einem RIGHT OUTER JOIN werden alle Datensätze aus der Tabelle A und der Tabelle B zusammengeführt, die die Selektionsbedingung erfüllen, sowie alle Elemente aus der rechten Tabelle.

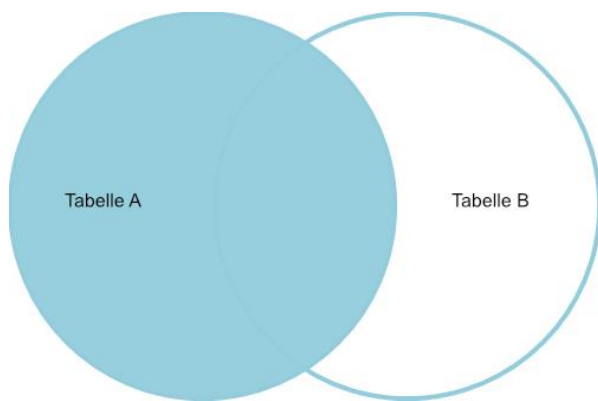


Abb. 06: LEFT OUTER JOIN

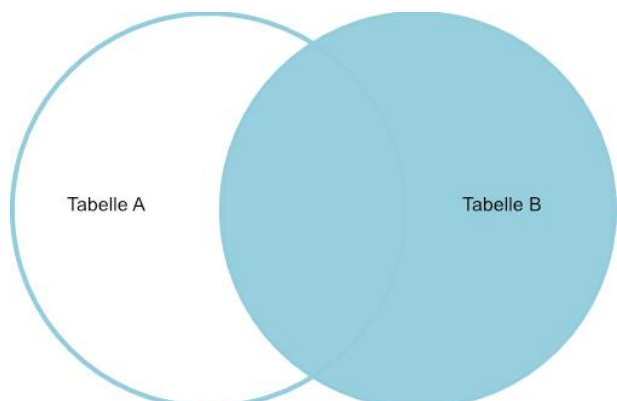


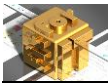
Abb. 07: RIGHT OUTER JOIN

Praxis

Um dich mit SQL weiter vertraut zu machen, studierst du am besten die gespeicherten Abfragen in der RC Wildbach Datenbank (Version: *RC-Wildbach-final.odt*). Du findest sie, wie alle anderen Versionen, im Ordner *232-materialien/datenbanken*. Danach kannst du dich an den Aufgaben auf dem Arbeitsblatt zu diesem Kapitel versuchen.

Vorname	Anzahl Buchungen
Sofia	166
Oyvind	149
Luigi	145
Eva	140
Sepp	140
Per	133
Petra	127

Abb. 08: Abfrage in SQL-Ansicht bearbeiten (Kontextmenü mit Rechtsklick)



2.3.2 Datenbanksysteme II

Lerninhalte 232-10 SQL-Abfragen über mehrere Tabellen

Abfragen von Datensätzen mit der DQL (engl. *data query language*)

Die wichtigste Aufgabe von SQL besteht darin, Informationen aus Datenbanken zu gewinnen. Daher ist der SELECT-Befehl aus der Datenabfragesprache DQL (Teilmenge von SQL) für die meisten Datenbankanwender die bedeutsamste Anweisung.

Hier sind noch einmal² Syntax und Operatoren von SQL für dich zusammengefasst:

Das Grundgerüst der SELECT-Anweisung

Das ist die grundlegende Syntax der SELECT-Anweisung:

```
SELECT [DISTINCT | ALL] Ausdrucksliste  
FROM Tabelle [Tabellenverbund mit JOIN]  
[WHERE Bedingungsliste]  
[GROUP BY Ausdrucksliste]  
[HAVING Bedingungsliste]  
[ORDER BY Ausdrucksliste [ASC3 | DESC4]]
```

Erläuterungen:

- **BLAU** und **FETT**: SQL-Schlüsselwörter
- **Ausdrucksliste**: Liste von Datenfeldern (aus einer oder mehreren Tabellen) oder Berechnungen.
- **Bedingungsliste**: Eine oder mehrere Bedingungen mit logischen Operatoren und Vergleichsoperatoren.

Vergleichs-Operatoren:

>	Größer als (geht auch mit größer-gleich, also >=)
<	Kleiner als (geht auch mit kleiner-gleich, also <=)
=	Gleich
!=	Ungleich (geht auch mit <>)
BETWEEN	Zwischen Minimal- und Maximalwert
LIKE	Zeichenkettenvergleich mit Platzhaltern % steht für eine beliebige Zeichenkette (mit 0 oder mehr Zeichen) _ steht für ein beliebiges einzelnes Zeichen
IN	Der IN-Parameter vergleicht, ob der Inhalt einer Spalte in einer angegebenen Liste enthalten ist.

Logische Operatoren:

OR	Logisches ODER (geht auch mit)
AND	Logisches UND (geht auch mit &&)
NOT	Logisches NICHT

² vgl. Modul Datenbanksysteme I: 231- 11 Abfragen erstellen Teil 1

³ engl. *ascending* -> dt. *aufsteigend*

⁴ engl. *descending* -> dt. *absteigend*