

OUTER JOIN

Bei den Abfragen im vorigen Kapitel nach „alle Mitarbeiter und ihre Dienstwagen“ werden nicht alle Mitarbeiter aufgeführt, weil in der Datenbank nicht für alle Mitarbeiter ein Dienstwagen registriert ist. Ebenso gibt es einen Dienstwagen, der keinem bestimmten Mitarbeiter zugeordnet ist.

Mit einem **OUTER JOIN** werden auch Mitarbeiter ohne Dienstwagen oder Dienstwagen ohne Mitarbeiter aufgeführt.

Die Syntax von OUTER JOIN

Die Syntax entspricht derjenigen von JOIN allgemein. Wegen der speziellen Bedeutung sind die Tabellen nicht gleichberechtigt, sondern werden begrifflich unterschieden:

```
SELECT <spaltenliste>
FROM <linke tabelle>
[<join-typ>] JOIN <rechte tabelle> ON <bedingung>
```

Als Spezialfälle des OUTER JOIN gibt es die JOIN-Typen **LEFT JOIN**, **RIGHT JOIN**, **FULL JOIN**.

Anstelle von <haupttabelle> und <zusatztabelle> wird bei OUTER JOIN von <linke tabelle> und <rechte tabelle> gesprochen, weil diese Tabellen unterschiedlich behandelt werden.

Allgemeine Hinweise zu OUTER JOIN

Das Wort OUTER kann entfallen und wird üblicherweise nicht benutzt, weil durch die Begriffe LEFT, RIGHT, FULL bereits ein OUTER JOIN gekennzeichnet wird.

Die Begriffe <linke tabelle> und <rechte tabelle> beziehen sich auf die beiden Tabellen bezüglich der normalen Lesefolge: Wir lesen von links nach rechts, also ist die unter FROM genannte Tabelle die <linke Tabelle> (bisher <Haupttabelle> genannt) und die unter JOIN genannte Tabelle die <rechte Tabelle> (bisher <Zusatztabelle> genannt). Bei Verknüpfungen mit mehreren Tabellen ist ebenfalls die unter JOIN genannte Tabelle die <rechte Tabelle>; die unmittelbar vorhergehende Tabelle ist die <linke Tabelle>.

Auch wenn die folgenden Beispiele so aussehen, als wenn die Datensätze sinnvoll sortiert wären, ist das Zufall; bitte denken Sie daran, dass SQL unsortierte Datenmengen liefert. Eine bestimmte Reihenfolge erhalten Sie erst durch ORDER BY.

Die Anzeige der Ergebnismengen bei den Beispielen ist in der Regel nur ein Auszug des vollständigen Ergebnisses.

LEFT OUTER JOIN

Inhaltsverzeichnis

- 1 Die Syntax von OUTER JOIN
- 2 Allgemeine Hinweise zu OUTER JOIN
- 3 LEFT OUTER JOIN
- 4 RIGHT OUTER JOIN
- 5 FULL OUTER JOIN
- 6 Verknüpfung mehrerer Tabellen
 - 6.1 Mehrere Tabellen parallel
 - 6.2 Gliederung durch Klammern
- 7 Zusammenfassung
- 8 Übungen

Dieser JOIN liefert alle Datensätze der linken Tabelle, ggf. unter Berücksichtigung der WHERE-Klausel. Aus der rechten Tabelle werden nur diejenigen Datensätze übernommen, die nach der Verknüpfungsbedingung passen.

```
SELECT <spaltenliste>
FROM <linke Tabelle>
LEFT [OUTER] JOIN <rechte Tabelle> ON <bedingung>;
```

Für unser Beispiel sieht das dann so aus:

▶ Hole alle Mitarbeiter und (sofern vorhanden) die Angaben zum Dienstwagen.

```
SELECT mi.Personalnummer AS MitNr,
       mi.Name, mi.Vorname,
       dw.ID AS DIW, dw.Kennzeichen, dw.Fahrzeugtyp_ID AS Typ
FROM Mitarbeiter mi
LEFT JOIN Dienstwagen dw ON dw.Mitarbeiter_ID = mi.ID;
```

Ausgabe

MITNR	NAME	VORNAME	DIW	KENNZEICHEN	TYP
30001	Wagner	Gaby	3	DO-WB 423	14
30002	Feyerabend	Werner			
40001	Langmann	Matthias	4	DO-WB 424	14
40002	Peters	Michael			
50001	Pohl	Helmut	5	DO-WB 425	14
50002	Braun	Christian	14	DO-WB 352	2
50003	Polovic	Frantisek	15	DO-WB 353	3
50004	Kalman	Aydin	16	DO-WB 354	4
60001	Aagenau	Karolin	6	DO-WB 426	14
60002	Pinkart	Petra			

Und wenn wir jetzt die beiden Tabellen vertauschen?

▶ Dann erhalten wir alle Dienstwagen und dazu die passenden Mitarbeiter.

```
SELECT mi.Personalnummer AS MitNr,
       mi.Name, mi.Vorname,
       dw.ID AS DIW, dw.Kennzeichen, dw.Fahrzeugtyp_ID AS Typ
FROM Dienstwagen dw
LEFT JOIN Mitarbeiter mi ON dw.Mitarbeiter_ID = mi.ID;
```

Ausgabe

MITNR	NAME	VORNAME	DIW	KENNZEICHEN	TYP
80001	Schindler	Christina	8	DO-WB 428	14
90001	Janssen	Bernhard	9	DO-WB 429	14
100001	Grosser	Horst	10	DO-WB 4210	14
110001	Eggert	Louis	11	DO-WB 4211	14
120001	Carlsen	Zacharias	12	DO-WB 4212	14

50002	Braun	Christian	14	DO-WB 352	2
50003	Polovic	Frantisek	15	DO-WB 353	3
50004	Kalman	Aydin	16	DO-WB 354	4

Bitte überlegen Sie selbst, wie sich WHERE-Klauseln auf das Ergebnis einer Abfrage auswirken.

RIGHT OUTER JOIN

Dieser JOIN liefert alle Datensätze der rechten Tabelle, ggf. unter Berücksichtigung der WHERE-Klausel. Aus der linken Tabelle werden nur diejenigen Datensätze übernommen, die nach der Verknüpfungsbedingung passen.

```
SELECT <spaltenliste>
FROM <linke Tabelle>
    RIGHT [OUTER] JOIN <rechte Tabelle> ON <bedingung>;
```

Für unser Beispiel „Mitarbeiter und Dienstwagen“ sieht das dann so aus:

```
SELECT mi.Personalnummer AS MitNr,
       mi.Name, mi.Vorname,
       dw.ID AS DIW, dw.Kennzeichen, dw.Fahrzeugtyp_ID AS Typ
FROM Mitarbeiter mi
    RIGHT JOIN Dienstwagen dw ON dw.Mitarbeiter_ID = mi.ID;
```

Ausgabe

MITNR	NAME	VORNAME	DIW	KENNZEICHEN	TYP
80001	Schindler	Christina	8	DO-WB 428	14
90001	Janssen	Bernhard	9	DO-WB 429	14
100001	Grosser	Horst	10	DO-WB 4210	14
110001	Eggert	Louis	11	DO-WB 4211	14
120001	Carlsen	Zacharias	12	DO-WB 4212	14
			13	DO-WB 111	16
50002	Braun	Christian	14	DO-WB 352	2
50003	Polovic	Frantisek	15	DO-WB 353	3
50004	Kalman	Aydin	16	DO-WB 354	4

Nanu, dieses Ergebnis hatten wir doch gerade? Bei genauerem Überlegen wird klar: Beim LEFT JOIN gibt es alle Datensätze der linken Tabelle mit Informationen der rechten Tabelle; nun haben wir die beiden Tabellen vertauscht. Beim RIGHT JOIN werden alle Datensätze der rechten Tabelle mit Daten der linken Tabelle verknüpft; das entspricht diesem Beispiel.

Ob wir also die beiden Tabellen vertauschen oder LEFT gegen RIGHT, bleibt sich zwangsläufig gleich. Kurz und „knackig“ formuliert kann man sich also merken:

"A LEFT JOIN B" liefert dasselbe Ergebnis wie "B RIGHT JOIN A".

Bitte überlegen Sie, welches Ergebnis die Vertauschung der beiden Tabellen beim RIGHT JOIN liefert und welche Auswirkung WHERE-Klauseln haben.

FULL OUTER JOIN

Dieser JOIN liefert alle Datensätze beider Tabellen, ggf. unter Berücksichtigung der WHERE-Klausel. Wenn Datensätze nach der Verknüpfungsbedingung zusammenpassen, werden sie in einer Zeile angegeben; wo es keinen „Partner“ gibt, wird ein NULL-Wert angezeigt.

```
SELECT <spaltenliste>
FROM <linke Tabelle>
FULL [OUTER] JOIN <rechte Tabelle> ON <bedingung>;
```

Für unser Beispiel sieht das dann so aus:

```
SELECT mi.Personalnummer AS MitNr,
       mi.Name, mi.Vorname,
       dw.ID AS DIW, dw.Kennzeichen, dw.Fahrzeugtyp_ID AS Typ
FROM Mitarbeiter mi
FULL JOIN Dienstwagen dw ON dw.Mitarbeiter_ID = mi.ID;
```

Ausgabe

MITNR	NAME	VORNAME	DIW	KENNZEICHEN	TYP
100001	Grosser	Horst	10	DO-WB 4210	14
110001	Eggert	Louis	11	DO-WB 4211	14
120001	Carlsen	Zacharias	12	DO-WB 4212	14
			13	DO-WB 111	16
50002	Braun	Christian	14	DO-WB 352	2
50003	Polovic	Frantisek	15	DO-WB 353	3
50004	Kalman	Aydin	16	DO-WB 354	4
80002	Aliman	Zafer	17	DO-WB 382	2
80003	Langer	Norbert	18	DO-WB 383	3
80004	Kolic	Ivana	19	DO-WB 384	4
10002	Schneider	Daniela			
20002	Schmitz	Michael			
30002	Feyerabend	Werner			
40002	Peters	Michael			

Auch hier wollen wir wieder die beiden Tabellen vertauschen:

```
SELECT mi.Personalnummer AS MitNr,
       mi.Name, mi.Vorname,
       dw.ID AS DIW, dw.Kennzeichen, dw.Fahrzeugtyp_ID AS Typ
FROM Dienstwagen dw
FULL JOIN Mitarbeiter mi ON dw.Mitarbeiter_ID = mi.ID;
```

Ausgabe

MITNR	NAME	VORNAME	DIW	KENNZEICHEN	TYP
-------	------	---------	-----	-------------	-----

```

-----
80001   Schindler   Christina   8   DO-WB 428   14
80002   Aliman     Zafer      17  DO-WB 382   2
80003   Langer      Norbert    18  DO-WB 383   3
80004   Kolic       Ivana      19  DO-WB 384   4
90001   Janssen     Bernhard   9   DO-WB 429   14
90002   Hinkel      Martina    10  DO-WB 4210  14
100001  Grosser     Horst      10  DO-WB 4210  14
100002  Friedrichsen Angelina   11  DO-WB 4211  14
110001  Eggert      Louis      11  DO-WB 4211  14
110002  Deiters     Gisela     12  DO-WB 4212  14
120001  Carlsen     Zacharias  12  DO-WB 4212  14
120002  Baber       Yvonne     13  DO-WB 111   16
-----

```

Bei detailliertem Vergleich des vollständigen Ergebnisses ergibt sich: Es ist gleich, nur in anderer Reihenfolge. Das sollte nicht mehr verwundern.

Verknüpfung mehrerer Tabellen

Alle bisherigen Beispiele kranken daran, dass als Typ des Dienstwagens nur die ID angegeben ist. Selbstverständlich möchte man die Typbezeichnung und den Hersteller lesen. Dazu müssen die beiden Tabellen *Fahrzeugtyp* und *Fahrzeughersteller* eingebunden werden. Beim INNER JOIN war das kein Problem; probieren wir aus, wie es beim OUTER JOIN aussehen könnte.

Mehrere Tabellen parallel

► Erweitern wir dazu die Aufstellung „alle Dienstwagen zusammen mit den zugeordneten Mitarbeitern“ um die Angabe zu den Fahrzeugen.

```

SELECT mi.Personalnummer AS MitNr,
       mi.Name, mi.Vorname,
       dw.ID AS DIW, dw.Kennzeichen, dw.Fahrzeugtyp_ID AS TypID,
       ft.Bezeichnung as Typ, ft.Hersteller_ID as FheID
FROM Dienstwagen dw
left JOIN Mitarbeiter mi ON dw.Mitarbeiter_ID = mi.ID
join      Fahrzeugtyp ft on dw.Fahrzeugtyp_ID = ft.ID;

```

Ausgabe

```

MITNR  NAME      VORNAME  DIW  KENNZEICHEN  TYPID  TYP              FHEID
-----
100001 Grosser   Horst    10   DO-WB 4210   14     A160              6
110001 Eggert    Louis    11   DO-WB 4211   14     A160              6
120001 Carlsen   Zacharias 12   DO-WB 4212   14     A160              6
        13   DO-WB 111     16     W211 (E-Klasse)   6
50002  Braun    Christian 14   DO-WB 352     2     Golf              1
50003  Polovic  Frantisek 15   DO-WB 353     3     Passat            1
50004  Kalman   Aydin     16   DO-WB 354     4     Kadett            2

```

Der zweite JOIN wurde nicht genauer bezeichnet, ist also ein INNER JOIN. Das gleiche Ergebnis erhalten wir, wenn wir die Tabelle *Fahrzeugtyp* ausdrücklich als LEFT JOIN verknüpfen (bitte selbst ausprobieren!). Anders sieht es beim Versuch mit RIGHT JOIN oder FULL JOIN aus:

```

SELECT mi.Personalnummer AS MitNr,
       mi.Name, mi.Vorname,
       dw.ID AS DIW, dw.Kennzeichen, dw.Fahrzeugtyp_ID AS TypID,
       ft.Bezeichnung as Typ, ft.Hersteller_ID as FheID
FROM Dienstwagen dw
  left JOIN      Mitarbeiter mi ON dw.Mitarbeiter_ID = mi.ID
  right | full join Fahrzeugtyp ft on dw.Fahrzeugtyp_ID = ft.ID;

```

Ausgabe

MITNR	NAME	VORNAME	DIW	KENNZEICHEN	TYPID	TYP	FHEID
80001	Schindler	Christina	8	DO-WB 428	14	A160	6
90001	Janssen	Bernhard	9	DO-WB 429	14	A160	6
100001	Grosser	Horst	10	DO-WB 4210	14	A160	6
110001	Eggert	Louis	11	DO-WB 4211	14	A160	6
120001	Carlsen	Zacharias	12	DO-WB 4212	14	A160	6
						W204 (C-Klasse)	6
			13	DO-WB 111	16	W211 (E-Klasse)	6
						Saab 9-3	8
						S40	9
						C30	9

Versuchen wir eine Erklärung: Die beiden JOINS stehen sozusagen auf der gleichen Ebene; jede JOIN-Klausel wird für sich mit der Tabelle *Dienstwagen* verknüpft. An der Verknüpfung zwischen *Dienstwagen* und *Mitarbeiter* ändert sich nichts. Aber für die Fahrzeugtypen gilt:

- Das erste Beispiel benutzt einen INNER JOIN, nimmt also für jeden vorhandenen Dienstwagen genau „seinen“ Typ.
- Wenn man stattdessen einen LEFT JOIN verwendet, erhält man alle vorhandenen Dienstwagen, zusammen mit den passenden Typen. Das ist faktisch identisch mit dem Ergebnis des INNER JOIN.
- Das zweite Beispiel benutzt einen RIGHT JOIN, das liefert alle registrierten Fahrzeugtypen und (soweit vorhanden) die passenden Dienstwagen.
- Wenn man stattdessen einen FULL JOIN verwendet, erhält man alle Kombinationen von Dienstwagen und Mitarbeitern, zusammen mit allen registrierten Fahrzeugtypen. Das ist faktisch identisch mit dem Ergebnis des RIGHT JOIN.

Sie sehen: Es kommt genau auf die gewünschten und die tatsächlich vorhandenen Verknüpfungen an.

Gliederung durch Klammern

Für Verknüpfungen, die durch Klammern gegliedert werden, nehmen wir ein anderes Beispiel, nämlich „Mitarbeiter RIGHT JOIN Dienstwagen“, denn die Fahrzeugtypen sind eine Ergänzung zu den Dienstwagen, nicht zu den Mitarbeitern (auch wenn den Abteilungsleitern ein Mercedes zugestanden wird, aber das ist ein anderes Thema und hat nichts mit SQL zu tun).

```

SELECT mi.Personalnummer AS MitNr,
       mi.Name, mi.Vorname,
       dw.ID AS DIW, dw.Kennzeichen, dw.Fahrzeugtyp_ID AS TypID,
       ft.Bezeichnung as Typ, ft.Hersteller_ID as FheID
FROM Mitarbeiter mi
   right JOIN ( Dienstwagen dw
              join Fahrzeugtyp ft on ft.ID = dw.Fahrzeugtyp_id )
   ON dw.Mitarbeiter_ID = mi.ID;

```

Ausgabe

MITNR	NAME	VORNAME	DIW	KENNZEICHEN	TYPID	TYP	FHEID
80001	Schindler	Christina	8	DO-WB 428	14	A160	6
90001	Janssen	Bernhard	9	DO-WB 429	14	A160	6
100001	Grosser	Horst	10	DO-WB 4210	14	A160	6
110001	Eggert	Louis	11	DO-WB 4211	14	A160	6
120001	Carlsen	Zacharias	12	DO-WB 4212	14	A160	6
			13	DO-WB 111	16	W211 (E-Klasse)	6
50002	Braun	Christian	14	DO-WB 352	2	Golf	1
50003	Polovic	Frantisek	15	DO-WB 353	3	Passat	1
50004	Kalman	Aydin	16	DO-WB 354	4	Kadett	2

Auch hier erhalten wir ein vergleichbares Ergebnis. Prüfen wir zunächst die Abfrage in der Klammer:

- LEFT JOIN und INNER JOIN haben als Grundlage „alle Dienstwagen“, es wird also eine Datenmenge „alle Dienstwagen“ (mit Zusatzinformationen über die Fahrzeugtypen) erstellt.
- RIGHT JOIN und FULL JOIN gehen aus von „alle Fahrzeugtypen“, es wird also eine Datenmenge „alle Fahrzeugtypen“ (mit Zusatzinformationen über die Dienstwagen) erstellt.

Da der Ausdruck innerhalb der Klammern zuerst ausgewertet wird, wird diese Datenmenge anschließend mit den Mitarbeitern verknüpft, soweit es der Verknüpfungsbedingung auf der Basis von dw.Mitarbeiter_ID entspricht.

Mit diesen Erkenntnissen können wir nun auch den Hersteller mit seinem Namen anzeigen; dazu benutzen wir wegen der bisherigen Erkenntnisse das erste Beispiel:

```

SELECT mi.Personalnummer AS MitNr,
       mi.Name, mi.Vorname,
       dw.ID AS DIW, dw.Kennzeichen, dw.Fahrzeugtyp_ID AS TypID,
       ft.Bezeichnung as Typ, fh.Name as Hersteller
FROM Dienstwagen dw
   left JOIN Mitarbeiter mi ON mi.ID = dw.Mitarbeiter_ID
   inner join Fahrzeugtyp ft on ft.ID = dw.Fahrzeugtyp_ID
   inner join Fahrzeughersteller fh on fh.ID = ft.Hersteller_ID;

```

Ausgabe

MITNR	NAME	VORNAME	DIW	KENNZEICHEN	TYPID	TYP	HERSTELLER
80001	Schindler	Christina	8	DO-WB 428	14	A160	Mercedes-Benz
90001	Janssen	Bernhard	9	DO-WB 429	14	A160	Mercedes-Benz
100001	Grosser	Horst	10	DO-WB 4210	14	A160	Mercedes-Benz
110001	Eggert	Louis	11	DO-WB 4211	14	A160	Mercedes-Benz

120001	Carlsen	Zacharias	12	DO-WB	4212	14	A160	Mercedes-Benz
150002	Braun	Christian	13	DO-WB	111	16	W211 (E-Klasse)	Mercedes-Benz
150003	Polovic	Frantisek	14	DO-WB	352	2	Golf	Volkswagen
150004	Kalman	Aydin	15	DO-WB	353	3	Passat	Volkswagen
180002	Aliman	Zafer	16	DO-WB	354	4	Kadett	Opel
180003	Langer	Norbert	17	DO-WB	382	2	Golf	Volkswagen
180004	Kolic	Ivana	18	DO-WB	383	3	Passat	Volkswagen
			19	DO-WB	384	4	Kadett	Opel

Zusammenfassung

In diesem Kapitel lernten Sie die Verwendung von OUTER JOIN kennen:

- Mit dieser Verknüpfung werden auch Datensätze abgefragt und angezeigt, bei denen es in einer der Tabellen keinen zugeordneten Datensatz gibt.
- Mit einem LEFT JOIN erhält man alle Datensätze der linken Tabelle, ergänzt durch passende Angaben aus der rechten Tabelle.
- Mit einem RIGHT JOIN erhält man alle Datensätze der rechten Tabelle, ergänzt durch passende Angaben aus der linken Tabelle.
- Mit einem FULL JOIN erhält man alle Datensätze beider Tabellen, wenn möglich ergänzt durch passende Angaben aus der jeweils anderen Tabelle.

Bei der Verknüpfung mehrerer Tabellen ist genau auf den JOIN-Typ und ggf. auf Klammerung zu achten.

Übungen

Übung 1

Allgemeines

Zur Lösung

Welche der folgenden Aussagen sind wahr, welche sind falsch?

1. Um alle Mitarbeiter mit Dienstwagen aufzulisten, benötigt man einen LEFT OUTER JOIN.
2. LEFT JOIN ist nur eine Kurzschreibweise für LEFT OUTER JOIN und hat keine zusätzliche inhaltliche Bedeutung.
3. Ein LEFT JOIN von zwei Tabellen enthält alle Zeilen, die nach Auswahlbedingung in der linken Tabelle enthalten sind.
4. Ein RIGHT JOIN von zwei Tabellen enthält nur noch diejenigen Zeilen, die nach der Verknüpfungsbedingung in der linken Tabelle enthalten sind.
5. Wenn wir bei einer LEFT JOIN-Abfrage mit zwei Tabellen die beiden Tabellen vertauschen und stattdessen einen RIGHT JOIN verwenden, erhalten wir dieselben Zeilen in der Ergebnismenge.
6. Wir erhalten dabei nicht nur dieselben Zeilen, sondern auch dieselbe Reihenfolge.

Übung 2

Allgemeines

Zur Lösung

Was ist am folgenden SELECT-Befehl falsch und warum? Die Aufgabe dazu lautet:

- Gesucht werden Kombinationen von Fahrzeug-Kennzeichen und Fahrzeugtypen, wobei alle Typen aufgeführt werden sollen; es werden nur die ersten 20 Fahrzeuge nach ID benötigt.

```

1 select Kennzeichen, Bezeichnung
2   from Fahrzeug fz
3   left join Fahrzeugtyp ft on fz.Fahrzeugtyp_ID = ft.ID
4  where fz.ID <= 20 ;

```