

Aufg. 3

Ein INNER-JOIN kombiniert nur Datensätze im Suchergebnis die der JOIN-Bedingung genügen.

Ein OUTER-JOIN nimmt zusätzlich noch Datensätze auf, die NICHT der JOIN-Bedingung entsprechen und zwar:

- beim LEFT JOIN die der ersten (linken) Tabelle
- beim RIGHT JOIN die der zweiten (rechten) Tabelle
- beim FULL JOIN die aus beiden Tabellen

Das Schlüsselwort OUTER kann immer entfallen:

- LEFT OUTER JOIN <-> LEFT JOIN
- RIGHT OUTER JOIN <-> RIGHT JOIN
- FULL OUTER JOIN <-> FULL JOIN

Aufg. 4

- CROSS JOIN
- THETA JOIN
- EQUI JOIN
- NATURAL JOIN

Das Schlüsselwort INNER kann generell entfallen.

a.) CROSS JOIN:

Explizite Variante:

```
SELECT * FROM abteilung CROSS JOIN mitarbeiter;
```

Implizite Variante:

```
SELECT * FROM mitarbeiter, abteilung;
```

b.) THETA JOIN – kein Praxisbeispiel bekannt:

Der Theta-Join ist ein Inner-Join, bei dem die Join-Bedingung kein Gleichheitszeichen enthält:

Explizite Variante:

```
SELECT *  
FROM mitarbeiter AS m  
INNER JOIN abteilung AS a  
ON m.AbtNr > a.AbtNr
```

Implizite Variante:

```
SELECT *  
FROM mitarbeiter AS m, abteilung AS a  
ON m.AbtNr > a.AbtNr
```

- c.) EQUI-JOIN – am häufigsten gebrauchter Inner-Join und auch der am häufigsten verwendeter Join überhaupt:

Die Join-Bedingung beinhaltet die Gleichheit von PS- und FS-Spalte.

Explizite Variante:

```
SELECT *
```

```
FROM mitarbeiter AS m
```

```
INNER JOIN abteilung AS a
```

```
ON m.AbtNr = a.AbtNr
```

Implizite Variante:

```
SELECT *
```

```
FROM mitarbeiter AS m, abteilung AS a
```

```
ON m.AbtNr = a.AbtNr
```

Im Suchergebnis erscheinen nur zusammengehörige Datensätze beider Tabellen, das heißt Datensätze, die die Join-Bedingung erfüllen.

- d.) NATURAL JOIN – in der Praxis eher selten verwendet.

Ein Natural-Join ist ein Equi-Join mit impliziter, aber formell fehlender Join-Bedingung.

Implizit wird die Join-Bedingung über die beiden gleichbenannte Spalten der beteiligten Tabellen gebildet.

Explizite Variante:

```
SELECT *
```

```
FROM mitarbeiter AS m
```

```
NATURAL JOIN abteilung AS a;
```

Es gibt keine implizite Variante.

Outer- Joins teilen sich auf in:

-LEFT OUTER JOIN

-RIGHT OUTER JOIN

-FULL OUTER JOIN

Das Schlüsselwort OUTER kann dabei generell entfallen.

- e.) LEFT OUTER JOIN – oft benötigt:

Ein Left-Join liefert alle Datensätze, des entsprechenden Equi-Joins und zusätzlich alle verbleibenden Datensätze aus der Linken (der zuerst genannten) Tabelle. Fehlende Werte der rechten Tabellen werden mit NULL-Werten aufgefüllt.

```
SELECT *
```

```
FROM abteilung AS a
```

```
LEFT OUTER JOIN mitarbeiter AS m
```

```
ON m.AbtNr = a.AbtNr
```

Jeder Left-Outer-Join kann als Right-Outer-Join formuliert werden:

```
SELECT *
```

```
FROM mitarbeiter AS m
```

```
RIGHT OUTER JOIN abteilung AS a
```

```
ON m.AbtNr = a.AbtNr
```

Es gibt keine implizite Variante.

Die nicht zugeordneten Datensätze der linken oder rechten Tabelle kann man durch Abfrage nach den NULL-Werten ermitteln:

```
SELECT *  
FROM abteilung AS a  
LEFT OUTER JOIN mitarbeiter AS m  
ON m.AbtNr = a.AbtNr  
WHERE m.PNr IS NULL
```

Formell kann man den Outer-Join immer zur Ermittlung der nicht zugeordneten Datensätze der linken oder rechten Tabelle verwenden, erforderlich ist er aber nur, wenn es um die Datensätze der 1-Seite einer 1:N-Beziehung geht:

```
SELECT *  
FROM abteilung AS a  
RIGHT OUTER JOIN mitarbeiter AS m  
ON m.AbtNr = a.AbtNr  
WHERE m.AbtNr IS NULL;
```

Das lässt sich aber immer auch einfacher ohne Outer-Join ermitteln:

```
SELECT *  
FROM mitarbeiter AS m  
WHERE m.AbtNr IS NULL;
```

Zur Ermittlung nicht zugeordneter Datensätze muss der Outer-Join auf die Tabelle zeigen, aus der die nicht zugeordneten Datensätze ermittelt werden sollen. Dabei ist der Outer-Join nur erforderlich, wenn es sich um die 1-Seite der Beziehung handelt. Und auch hier sind Left- und Right-Outer-Join alternativ verwendbar.

f.) RIGHT OUTER JOIN – siehe LEFT OUTER JOIN