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 Tehta-Join ist ein Inner-Join, bei dem die Join-Bedngung 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 mitarbeiterAS 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

g.) FULL OUTER JOIN
Explizite Variante:
Der Full-Outer-Join kombiniert einen Left- und Right-Join.

```
SELECT *
FROM abteilung AS a
FULL OUTER JOIN mitarbeiter AS m
ON m.AbtNr = a.AbtNr;
```

Der Full-Outer-Join funktioniert nicht direkt in MySQL und MariaDB, wohl aber z.B. in PostgreSQL.

Implizite Variante:

```
SELECT *
FROM abteilung AS a
LEFT OUTER JOIN mitarbeiter AS m
ON m.AbtNr = a.AbtNr
UNION
SELECT *
FROM abteilung AS a
RIGHT OUTER JOIN mitarbeiter AS m
ON m.AbtNr = a.AbtNr;
```

Der UNION-Operator arbeitet nur auf Datensatz-Mengen, die dieselben Spalten besitzen.

Der UNION-Operator ist in verschiedenen Datenbanken erforderlich um einen Full-Outer-Join nachzubilden.

Left- und Right-Join werden genutzt, um nicht zugeordnete Datensätze auf der 1-Seite von 1:N-Beziehungen zu ermitteln:

Beispiel: Ermitteln Sie alle Abteilungsnamen aller Abteilungen, denen keine Mitarbeiter zugeordnet sind.

Lösung:
Sql_join Datenbank:

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

Projektdb Datenbank:

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

Merke: Dabei muss LEFT bzw. Right auf die Tabelle auf der 1-Seite zeigen.

Man kann auch mittels Outer-Join nach nicht zugeordneten Datensätzen auf der N-Seite suchen, doch ist dies nicht erforderlich, weil man direkt und einfacher die entsprechenden NULL-Werte der Fremdschlüsselspalte ermitteln kann:

```
SELECT *
FROM mitarbeiter
WHERE F_AbtNr IS NULL;
```

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