

Datenbanken

Teil 2: Informationen

Kapitel 10: SQL



SQL-Join-Abfragen

Die verschiedenen Join-Abfragen werden vorgestellt:

1. Equi - Join (ohne übereinstimmende Merkmalsnamen)
2. Equi - Join mit Aliasnamen
3. Equi - Join mit Aliasnamen über mehrere Tabellen
4. Left -, Right - Join
5. Left -, Right - Join über mehrere Tabellen
6. Self - Join

Equi - Join (**ohne** übereinstimmende Merkmalsnamen)

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_A
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	

Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

Eva Schulte
ist noch
keiner Abteilung
zugeordnet

Equi - Join (**ohne** übereinstimmende Merkmalsnamen)

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_A
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	

Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

SQL-Abfrage

Zeige den Vor- und Nachnamen der Mitarbeiter
und die zugehörigen Abteilungsbezeichnungen an.

Equi - Join (**ohne** übereinstimmende Merkmalsnamen)

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_A
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	

Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

SQL-Abfrage

Zeige den Vor- und Nachnamen der Mitarbeiter
und die zugehörigen Abteilungsbezeichnungen an.

SELECT Vorname, Nachname, Bezeichnung

Equi - Join (**ohne** übereinstimmende Merkmalsnamen)

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_A
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	

Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

SQL-Abfrage

Zeige den Vor- und Nachnamen der Mitarbeiter
und die zugehörigen Abteilungsbezeichnungen an.

SELECT Vorname, Nachname, Bezeichnung
FROM Mitarbeiter, Abteilung

Equi - Join (**ohne** übereinstimmende Merkmalsnamen)

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_A
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	

Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

SQL-Abfrage

Zeige den Vor- und Nachnamen der Mitarbeiter
und die zugehörigen Abteilungsbezeichnungen an.

```
SELECT Vorname, Nachname, Bezeichnung
FROM Mitarbeiter, Abteilung
WHERE ID_A = ID_Abteilung;
```

Ausgabe:

Equi - Join (**ohne** übereinstimmende Merkmalsnamen)

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_A
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	

Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

SQL-Abfrage

Zeige den Vor- und Nachnamen der Mitarbeiter
und die zugehörigen Abteilungsbezeichnungen an.

SELECT Vorname, Nachname, Bezeichnung
FROM Mitarbeiter, Abteilung
WHERE ID_A = ID_Abteilung;

Ausgabe: Willi, Specht, Entwicklung
 Adam, Ernst, Fertigung
 Silke, Müller, Entwicklung

Equi - Join (*mit* übereinstimmenden Merkmalsnamen)

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	

Abteilung

ID_Abteilung	Name
1	Entwicklung
2	Fertigung



Eva Schulte
ist noch
keiner Abteilung
zugeordnet

Equi - Join (*mit* übereinstimmenden Merkmalsnamen)

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	

Abteilung

ID_Abteilung	Name
1	Entwicklung
2	Fertigung

SQL-Abfrage

Zeige den Vor- und Nachnamen der Mitarbeiter
und die zugehörigen Abteilungsbezeichnungen an.

Equi - Join (*mit* übereinstimmenden Merkmalsnamen)

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	

Abteilung

ID_Abteilung	Name
1	Entwicklung
2	Fertigung

SQL-Abfrage

Zeige den Vor- und Nachnamen der Mitarbeiter
und die zugehörigen Abteilungsbezeichnungen an.

Problem

In den Tabellen sind identische Spaltennamen vorhanden.
Keine Eindeutige Zuordnung möglich.

Equi - Join (*mit* übereinstimmenden Merkmalsnamen)

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	

Abteilung

ID_Abteilung	Name
1	Entwicklung
2	Fertigung

SQL-Abfrage

Zeige den Vor- und Nachnamen der Mitarbeiter
und die zugehörigen Abteilungsbezeichnungen an.

Problem

In den Tabellen sind identische Spaltennamen vorhanden.
Keine Eindeutige Zuordnung möglich.

Lösung

Tabellenname dem Merkmal voranstellen.

Equi - Join (*mit* übereinstimmenden Merkmalsnamen)

Mitarbeiter

<u>ID_Mitarbeiter</u>	Name	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	

Abteilung

<u>ID_Abteilung</u>	Name
1	Entwicklung
2	Fertigung

SQL-Abfrage

Zeige den Vor- und Nachnamen der Mitarbeiter
und die zugehörigen Abteilungsbezeichnungen an.

```
SELECT Mitarbeiter.Vorname, Mitarbeiter.Name,  
        Abteilung.Name  
FROM    Mitarbeiter, Abteilung  
WHERE   Mitarbeiter.ID_Abteilung = Abteilung.ID_Abteilung;
```

Ausgabe: Willi, Specht, Entwicklung
Adam, Ernst, Fertigung
Silke, Müller, Entwicklung

Equi - Join mit Aliasnamen

Mitarbeiter

ID_Mitarbeiter	Name
1	Specht
2	Ernst
3	Müller
4	Schulte

Aliasnamen

Im FROM-Teil der SQL-Anweisung können Abkürzungen für Tabellennamen definiert werden.

FROM Mitarbeiter M, Abteilung A

SQL-Abfrage

Zeige den Vor- und Nachnamen der Mitarbeiter und die zugehörigen Abteilungsbezeichnungen an.

```

SELECT Mitarbeiter.Vorname, Mitarbeiter.Name,
          Abteilung.Name
FROM     Mitarbeiter, Abteilung
WHERE    Mitarbeiter.ID_Abteilung = Abteilung.ID_Abteilung;
  
```

Ausgabe: Willi, Specht, Entwicklung
 Adam, Ernst, Fertigung
 Silke, Müller, Entwicklung

Equi - Join mit Aliasnamen

Mitarbeiter

ID_Mitarbeiter	Name
1	Specht
2	Ernst
3	Müller
4	Schulte

Aliasnamen

Im FROM-Teil der SQL-Anweisung können Abkürzungen für Tabellennamen definiert werden.

FROM Mitarbeiter M, Abteilung A

SQL-Abfrage

Zeige den Vor- und Nachnamen der Mitarbeiter und die zugehörigen Abteilungsbezeichnungen an.

```

SELECT Mitarbeiter.Vorname, Mitarbeiter.Name,
          Abteilung.Name
FROM     Mitarbeiter M, Abteilung A
WHERE    Mitarbeiter.ID_Abteilung = Abteilung.ID_Abteilung;
  
```

Ausgabe: Willi, Specht, Entwicklung
 Adam, Ernst, Fertigung
 Silke, Müller, Entwicklung

Equi - Join mit Aliasnamen

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	ID_Abteilung	Abteilung	Name
1	Specht	Willi			
2	Ernst	Adam			
3	Müller	Silke			
4	Schulte	Eva			

Aliasnamen

Anschließend können diese Abkürzungen in der gesamten Abfrage verwendet werden.

SQL-Abfrage

Zeige den Vor- und Nachnamen der Mitarbeiter und die zugehörigen Abteilungsbezeichnungen an.

```

SELECT Mitarbeiter.Vorname, Mitarbeiter.Name,
          Abteilung.Name
FROM     Mitarbeiter M, Abteilung A
WHERE    Mitarbeiter.ID_Abteilung = Abteilung.ID_Abteilung;
  
```

Ausgabe: Willi, Specht, Entwicklung
 Adam, Ernst, Fertigung
 Silke, Müller, Entwicklung

Equi - Join mit Aliasnamen

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	ID_Abteilung	Abteilung
				Name
1	Specht	Willi		
2	Ernst	Adam		
3	Müller	Silke		
4	Schulte	Eva		

Aliasnamen
Anschließend können diese Abkürzungen in der gesamten Abfrage verwendet werden.

SQL-Abfrage

Zeige den Vor- und Nachnamen der Mitarbeiter und die zugehörigen Abteilungsbezeichnungen an.

```

SELECT M.Vorname, M.Name, A.Name
FROM Mitarbeiter M, Abteilung A
WHERE M.ID_Abteilung = A.ID_Abteilung;
    
```

Ausgabe: Willi, Specht, Entwicklung
 Adam, Ernst, Fertigung
 Silke, Müller, Entwicklung

Equi - Join mit Aliasnamen über mehrere Tabellen

Kunde

<u>ID_Kunde</u>	Name
1	Specht
2	Ernst
3	Müller

vormerken

<u>ID_Kunde</u>	<u>ID_Buch</u>
1	1
1	2
2	2

Buch

<u>ID_Buch</u>	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

, Java

Equi - Join mit Aliasnamen über mehrere Tabellen

Kunde

<u>ID_Kunde</u>	Name
1	Specht
2	Ernst
3	Müller

vormerken

<u>ID_Kunde</u>	<u>ID_Buch</u>
1	1
1	2
2	2

Buch

<u>ID_Buch</u>	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

SQL-Abfrage

Zeige den Namen der Kunden
und den Titel der vorgemerkteten Bücher an.

Equi - Join mit Aliasnamen über mehrere Tabellen

Kunde

<u>ID_Kunde</u>	Name
1	Specht
2	Ernst
3	Müller

vormerken

<u>ID_Kunde</u>	<u>ID_Buch</u>
1	1
1	2
2	2

Buch

<u>ID_Buch</u>	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

SQL-Abfrage

Zeige den Namen der Kunden
und den Titel der vorgemerkteten Bücher an.

SELECT K.Name, B.Titel

Equi - Join mit Aliasnamen über mehrere Tabellen

Kunde

<u>ID_Kunde</u>	Name
1	Specht
2	Ernst
3	Müller

vormerken

<u>ID_Kunde</u>	<u>ID_Buch</u>
1	1
1	2
2	2

Buch

<u>ID_Buch</u>	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

SQL-Abfrage

Zeige den Namen der Kunden
und den Titel der vorgemerkteten Bücher an.

```
SELECT      K.Name, B.Titel
FROM        Kunde K, vormerken v, Buch B
```

Equi - Join mit Aliasnamen über mehrere Tabellen

Kunde

<u>ID_Kunde</u>	Name
1	Specht
2	Ernst
3	Müller

vormerken

<u>ID_Kunde</u>	<u>ID_Buch</u>
1	1
1	2
2	2

Buch

<u>ID_Buch</u>	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

SQL-Abfrage

Zeige den Namen der Kunden
und den Titel der vorgemerkteten Bücher an.

```
SELECT      K.Name, B.Titel
FROM        Kunde K, vormerken v, Buch B
WHERE       K.ID_Kunde = v.ID_Kunde
```

Equi - Join mit Aliasnamen über mehrere Tabellen

Kunde

<u>ID_Kunde</u>	Name
1	Specht
2	Ernst
3	Müller

vormerken

<u>ID_Kunde</u>	<u>ID_Buch</u>
1	1
1	2
2	2

Buch

<u>ID_Buch</u>	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

SQL-Abfrage

Zeige den Namen der Kunden
und den Titel der vorgemerkteten Bücher an.

```

SELECT      K.Name, B.Titel
FROM        Kunde K, vormerken v, Buch B
WHERE       K.ID_Kunde = v.ID_Kunde
AND         v.ID_Buch = B.ID_Buch;
  
```

Ausgabe:

Equi - Join mit Aliasnamen über mehrere Tabellen

Kunde

<u>ID_Kunde</u>	Name
1	Specht
2	Ernst
3	Müller

vormerken

<u>ID_Kunde</u>	<u>ID_Buch</u>
1	1
1	2
2	2

Buch

<u>ID_Buch</u>	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

SQL-Abfrage

Zeige den Namen der Kunden
und den Titel der vorgemerkteten Bücher an.

```
SELECT      K.Name, B.Titel
FROM        Kunde K, vormerken v, Buch B
WHERE       K.ID_Kunde = v.ID_Kunde
AND         v.ID_Buch = B.ID_Buch;
```

Ausgabe: Specht, SQL
 Specht, Java
 Ernst, Java

Left-, Right - Join

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	

Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

Eva Schulte
ist noch
keiner Abteilung
zugeordnet

Left-, Right - Join

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	

Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

SQL-Abfrage

Zeige den Vor- und Nachnamen ALLER Mitarbeiter
und (wenn vorhanden) die zugehörigen Abteilungsbezeichnungen an.

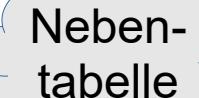
Left-, Right - Join

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	



ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung



SQL-Abfrage

Zeige den Vor- und Nachnamen ALLER Mitarbeiter
und (wenn vorhanden) die zugehörigen Abteilungsbezeichnungen an.

Problem

Haupt- und Nebentabelle müssen festgelegt werden.

Left-, Right - Join

Mitarbeiter

<u>ID_Mitarbeiter</u>	<u>Nachname</u>	<u>Vorname</u>	<u>ID_Abteilung</u>
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	



<u>Abteilung</u>	<u>ID_Abteilung</u>	<u>Bezeichnung</u>
	1	Entwicklung
	2	Fertigung



SQL-Abfrage

Zeige den Vor- und Nachnamen ALLER Mitarbeiter
und (wenn vorhanden) die zugehörigen Abteilungsbezeichnungen an.

Problem

Haupt- und Nebentabelle müssen festgelegt werden.

Lösung

Bestimmen der Haupttabelle in der FROM Anweisung.

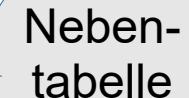
Left-, Right - Join

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	



ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung



SQL-Abfrage

Zeige den Vor- und Nachnamen ALLER Mitarbeiter
und (wenn vorhanden) die zugehörigen Abteilungsbezeichnungen an.

Problem

Haupt- und Nebentabelle müssen festgelegt werden.

Lösung

Bestimmen der Haupttabelle in der FROM Anweisung.

```
FROM Mitarbeiter M LEFT JOIN Abteilung A
```

Left-, Right - Join

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	



ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung



SQL-Abfrage

Zeige den Vor- und Nachnamen ALLER Mitarbeiter
und (wenn vorhanden) die zugehörigen Abteilungsbezeichnungen an.

Problem

Haupt- und Nebentabelle müssen festgelegt werden.

Lösung

Bestimmen der Haupttabelle in der FROM Anweisung.

FROM Mitarbeiter M **LEFT JOIN** Abteilung A

FROM Abteilung A **RIGHT JOIN** Mitarbeiter M



Left-, Right - Join

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	



Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung



SQL-Abfrage

Zeige den Vor- und Nachnamen ALLER Mitarbeiter
und (wenn vorhanden) die zugehörigen Abteilungsbezeichnungen an.

FROM Mitarbeiter M **LEFT JOIN** Abteilung A

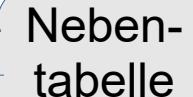
Left-, Right - Join

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	



ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung



SQL-Abfrage

Zeige den Vor- und Nachnamen ALLER Mitarbeiter
und (wenn vorhanden) die zugehörigen Abteilungsbezeichnungen an.

```
SELECT M.Vorname, M.Nachname, A.Bezeichnung
FROM Mitarbeiter M LEFT JOIN Abteilung A
```

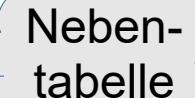
Left-, Right - Join

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	



ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung



SQL-Abfrage

Zeige den Vor- und Nachnamen ALLER Mitarbeiter
und (wenn vorhanden) die zugehörigen Abteilungsbezeichnungen an.

```

SELECT M.Vorname, M.Nachname, A.Bezeichnung
FROM Mitarbeiter M LEFT JOIN Abteilung A
ON A.ID_Abteilung = M.ID_Abteilung
  
```

Ausgabe:

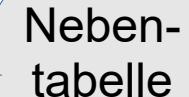
Left-, Right - Join

Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname	ID_Abteilung
1	Specht	Willi	1
2	Ernst	Adam	2
3	Müller	Silke	1
4	Schulte	Eva	



ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung



SQL-Abfrage

Zeige den Vor- und Nachnamen ALLER Mitarbeiter
und (wenn vorhanden) die zugehörigen Abteilungsbezeichnungen an.

SELECT M.Vorname, M.Nachname, A.Bezeichnung
FROM Mitarbeiter M **LEFT JOIN** Abteilung A
ON A.ID_Abteilung = M.ID_Abteilung

Ausgabe: Specht, Willi, Entwicklung
 Ernst, Adam, Fertigung
 Müller, Silke, Entwicklung
 Schulte, Eva

Formulieren Sie dieselbe Abfrage mit RIGHT-JOIN.

Left-, Right - Join über mehrere Tabellen

Kunde

<u>ID_Kunde</u>	Name
1	Specht
2	Ernst
3	Müller

vormerken

<u>ID_Kunde</u>	<u>ID_Buch</u>
1	1
1	2
2	2

Buch

<u>ID_Buch</u>	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

Left-, Right - Join über mehrere Tabellen

Kunde

<u>ID_Kunde</u>	Name
1	Specht
2	Ernst
3	Müller

vormerken

<u>ID_Kunde</u>	<u>ID_Buch</u>
1	1
1	2
2	2

Buch

<u>ID_Buch</u>	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

SQL-Abfrage

Zeige den Name ALLER Kunden an
und (wenn vorhanden) die von ihnen vorgemerkten Buchtitel.

Left-, Right - Join über mehrere Tabellen

Kunde

<u>ID_Kunde</u>	Name
1	Specht
2	Ernst
3	Müller

vormerken

<u>ID_Kunde</u>	<u>ID_Buch</u>
1	1
1	2
2	2

Buch

<u>ID_Buch</u>	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

SQL-Abfrage

Zeige den Name ALLER Kunden an
und (wenn vorhanden) die von ihnen vorgemerkten Buchtitel.

SELECT K.Name, B.Titel

**Left-,
Kunde**

Haupt-
tabelle

<u>ID_Kunde</u>	Name
1	Specht
2	Ernst
3	Müller

Neben-
tabelle

vormerken

<u>ID_Kunde</u>	<u>ID_Buch</u>
1	1
1	2
2	2

SQL-Abfrage

Zeige den Name ALLER Kunden an
und (wenn vorhanden) die von ihnen vorgemerkten Buchtitel.

```
SELECT K.Name, B.Titel
FROM Kunde K LEFT JOIN vormerken v
```

**Left-,
Kunde**

Haupt-
tabelle

ID_Kunde	Name
1	Specht
2	Ernst
3	Müller

Neben-
tabelle

vormerken

ID_Kunde	ID_Buch
1	1
1	2
2	2

mehrere Tabellen

Buch

ID_Buch	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

SQL-Abfrage

Zeige den Name ALLER Kunden an
und (wenn vorhanden) die von ihnen vorgemerkten Buchtitel.

```
SELECT K.Name, B.Titel
FROM Kunde K LEFT JOIN vormerken v
ON K.ID_Kunde = v.ID_Kunde )
```

Left-, Right - Join über mehrere Tabellen

Kunde

ID_Kunde	Name
1	Specht
2	Ernst
3	Müller

vormerken

ID_Kunde	ID_Buch
1	1
1	2
2	2



Kunden

Buch

ID_Buch	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++



SQL-Abfrage

Zeige den Name ALLER Kunden an
und (wenn vorhanden) die von ihnen vorgemerkten Buchtitel.

```
SELECT K.Name, B.Titel
FROM Kunde K LEFT JOIN vormerken v
          ON K.ID_Kunde = v.ID_Kunde
LEFT JOIN Buch B
```

Left-, Right - Join über mehrere Tabellen

Kunde

ID_Kunde	Name
1	Specht
2	Ernst
3	Müller

vormerken

ID_Kunde	ID_Buch
1	1
1	2
2	2



Kunden

Buch

ID_Buch	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++



SQL-Abfrage

Zeige den Name ALLER Kunden an
und (wenn vorhanden) die von ihnen vorgemerkten Buchtitel.

```
SELECT K.Name, B.Titel
FROM Kunde K LEFT JOIN vormerken v
          ON K.ID_Kunde = v.ID_Kunde
LEFT JOIN Buch B
          ON v.ID_Buch = B.ID_BUCH
```

Ausgabe:

Left-, Right - Join über mehrere Tabellen

Kunde

<u>ID_Kunde</u>	Name
1	Specht
2	Ernst
3	Müller

vormerken

<u>ID_Kunde</u>	<u>ID_Buch</u>
1	1
1	2
2	2

Buch

<u>ID_Buch</u>	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

SQL-Abfrage

Zeige den Name ALLER Kunden an
und (wenn vorhanden) die von ihnen vorgemerkten Buchtitel.

```
SELECT K.Name, B.Titel
FROM Kunde K LEFT JOIN vormerken v
      ON K.ID_Kunde = v.ID_Kunde
LEFT JOIN Buch B
      ON v.ID_Buch = B.ID_BUCH
```

Ausgabe: Specht, SQL
 Specht, Java
 Ernst, Java
 Müller

**Left-,
Kunde**

Haupt-
tabelle

ID_Kunde	Name
1	Specht
2	Ernst
3	Müller

Neben-
tabelle

vormerken

ID_Kunde	ID_Buch
1	1
1	2
2	2

Haupt-
tabelle

Kunden
Buch

Neben-
tabelle

ID_Buch	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

SQL-Abfrage

Zeige den Name ALLER Kunden an
und (wenn vorhanden) die von ihnen vorgemerkten Buchtitel.

```
SELECT K.Name, B.Titel
FROM Kunde K LEFT JOIN vormerken v
ON K.ID_Kunde = v.ID_Kunde
LEFT JOIN Buch B
ON v.ID_Buch = B.ID_BUCH
```

Ausgabe:

Specht,	SQL
Specht,	Java
Ernst,	Java
Müller	

Formulieren
Sie dieselbe
Abfrage aus-
schließlich mit
RIGHT-JOIN.

**Left-,
Kunde**

Haupt-
tabelle

ID_Kunde	Name
1	Specht
2	Ernst
3	Müller

Neben-
tabelle

vormerken

ID_Kunde	ID_Buch
1	1
1	2
2	2

Haupt-
tabelle

Kunden
Buch

Neben-
tabelle

ID_Buch	ISBN	Titel
1	3-499-60063-3	SQL
2	4-253-11334-6	Java
3	6-223-53452-9	C++

SQL-Abfrage

Zeige den Name ALLER Kunden an
und (wenn vorhanden) die von ihnen vorgemerkten Buchtitel.

```
SELECT K.Name, B.Titel
FROM Kunde K LEFT JOIN vormerken v
      ON K.ID_Kunde = v.ID_Kunde
LEFT JOIN Buch B
      ON v.ID_Buch = B.ID_BUCH
```

Ausgabe: Specht, SQL
 Specht, Java
 Ernst, Java
 Müller

Formulieren
Sie dieselbe
Abfrage aus-
schließlich mit
RIGHT-JOIN.

Formulieren
Sie dieselbe
Abfrage.
Verwenden
Sie dazu einen
LEFT-JOIN
und einen
RIGHT-JOIN.

Self - Join

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	Gehalt	ID_Mitarbeiter_Vorgesetzter
1	Specht	Willi	3000 €	
2	Ernst	Adam	2000 €	1
3	Müller	Silke	5000 €	1
4	Schulte	Eva	2500 €	2

Self - Join

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	Gehalt	ID_Mitarbeiter_Vorgesetzter
1	Specht	Willi	3000 €	
2	Ernst	Adam	2000 €	1
3	Müller	Silke	5000 €	1
4	Schulte	Eva	2500 €	2

SQL-Abfrage

Welcher Mitarbeiter verdient mehr als sein Vorgesetzter?
Nenne Name und Gehalt des Mitarbeiters und des Vorgesetzten!

Self - Join

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	Gehalt	ID_Mitarbeiter_Vorgesetzter
1	Specht	Willi	3000 €	
2	Ernst	Adam	2000 €	1
3	Müller	Silke	5000 €	1
4	Schulte	Eva	2500 €	2

SQL-Abfrage

Welcher Mitarbeiter verdient mehr als sein Vorgesetzter?
Nenne Name und Gehalt des Mitarbeiters und des Vorgesetzten!

Problem

Fremdschlüssel der Tabelle verweist auf die eigene Tabelle.

Self - Join

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	Gehalt	ID_Mitarbeiter_Vorgesetzter
1	Specht	Willi	3000 €	
2	Ernst	Adam	2000 €	1
3	Müller	Silke	5000 €	1
4	Schulte	Eva	2500 €	2

SQL-Abfrage

Welcher Mitarbeiter verdient mehr als sein Vorgesetzter?
Nenne Name und Gehalt des Mitarbeiters und des Vorgesetzten!

Problem

Fremdschlüssel der Tabelle verweist auf die eigene Tabelle.

Lösungsidee

Zwei Referenzen auf dieselbe Tabelle erzeugen.
Eine Referenz für den Mitarbeiter und eine Referenz für den Vorgesetzten

Self - Join

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	Gehalt	ID_Mitarbeiter_Vorgesetzter
1	Specht	Willi	3000 €	
2	Ernst	Adam	2000 €	1
3	Müller	Silke	5000 €	1
4	Schulte	Eva	2500 €	2

SQL-Abfrage

Welcher Mitarbeiter verdient mehr als sein Vorgesetzter?
Nenne Name und Gehalt des Mitarbeiters und des Vorgesetzten!

Problem

Fremdschlüssel der Tabelle verweist auf die eigene Tabelle.

Lösungsidee

Zwei Referenzen auf dieselbe Tabelle erzeugen.

Eine Referenz für den Mitarbeiter und eine Referenz für den Vorgesetzten

FROM Mitarbeiter M, Mitarbeiter V

Self - Join

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	Gehalt	ID_Mitarbeiter_Vorgesetzter
1	Specht	Willi	3000 €	
2	Ernst	Adam	2000 €	1
3	Müller	Silke	5000 €	1
4	Schulte	Eva	2500 €	2

SQL-Abfrage

Welche Mitarbeiter verdient mehr als sein Vorgesetzter?
Nur die Vorgesetzten sind in der Tabelle Mitarbeiter!

Problem

Fremdschlüssel

Lösungsidee

Zwei Referenzen auf dieselbe Tabelle erzeugen.

Eine Referenz für den Mitarbeiter und eine Referenz für den Vorgesetzten

FROM

Mitarbeiter M, Mitarbeiter V

Self - Join

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	Gehalt	ID_Mitarbeiter_Vorgesetzter
1	Specht	Willi	3000 €	
2	Ernst	Adam	2000 €	1
3	Müller	Silke	5000 €	1
4	Schulte	Eva	2500 €	2

SQL-Abfrage

Welcher Mitarbeiter verdient mehr als sein Vorgesetzter?
Nenne Name und Gehalt des Mitarbeiters und des Vorgesetzten!

FROM Mitarbeiter M, Mitarbeiter V

Self - Join

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	Gehalt	ID_Mitarbeiter_Vorgesetzter
1	Specht	Willi	3000 €	
2	Ernst	Adam	2000 €	1
3	Müller	Silke	5000 €	1
4	Schulte	Eva	2500 €	2

SQL-Abfrage

Welcher Mitarbeiter verdient mehr als sein Vorgesetzter?
Nenne Name und Gehalt des Mitarbeiters und des Vorgesetzten!

```
SELECT M.Name, M.Gehalt, V.Name, V.Gehalt  
FROM Mitarbeiter M, Mitarbeiter V
```

Self - Join

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	Gehalt	ID_Mitarbeiter_Vorgesetzter
1	Specht	Willi	3000 €	
2	Ernst	Adam	2000 €	1
3	Müller	Silke	5000 €	1
4	Schulte	Eva	2500 €	2

SQL-Abfrage

Welcher Mitarbeiter verdient mehr als sein Vorgesetzter?
Nenne Name und Gehalt des Mitarbeiters und des Vorgesetzten!

```
SELECT M.Name, M.Gehalt, V.Name, V.Gehalt  
FROM Mitarbeiter M, Mitarbeiter V  
WHERE M.ID_Mitarbeiter_Vorgesetzter_ = V.ID_Mitarbeiter
```

Self - Join

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	Gehalt	ID_Mitarbeiter_Vorgesetzter
1	Specht	Willi	3000 €	
2	Ernst	Adam	2000 €	1
3	Müller	Silke	5000 €	1
4	Schulte	Eva	2500 €	2

SQL-Abfrage

Welcher Mitarbeiter verdient mehr als sein Vorgesetzter?
Nenne Name und Gehalt des Mitarbeiters und des Vorgesetzten!

```

SELECT      M.Name, M.Gehalt, V.Name, V.Gehalt
FROM        Mitarbeiter M, Mitarbeiter V
WHERE       M.ID_Mitarbeiter_Vorgesetzter = V.ID_Mitarbeiter
AND         M.Gehalt > V.Gehalt
  
```

Ausgabe:

Self - Join

Mitarbeiter

ID_Mitarbeiter	Name	Vorname	Gehalt	ID_Mitarbeiter_Vorgesetzter
1	Specht	Willi	3000 €	
2	Ernst	Adam	2000 €	1
3	Müller	Silke	5000 €	1
4	Schulte	Eva	2500 €	2

SQL-Abfrage

Welcher Mitarbeiter verdient mehr als sein Vorgesetzter?
Nenne Name und Gehalt des Mitarbeiters und des Vorgesetzten!

```

SELECT      M.Name, M.Gehalt, V.Name, V.Gehalt
FROM        Mitarbeiter M, Mitarbeiter V
WHERE       M.ID_Mitarbeiter_Vorgesetzter = V.ID_Mitarbeiter
AND         M.Gehalt > V.Gehalt
  
```

Ausgabe: Müller, 5000 €, Specht, 3000 €
 Schulte, 2500 €, Ernst, 2000 €