

Joins

- Je stärker die Normalisierung desto mehr Tabellen enthält eine Datenbank
- Die Tabellen sind über Schlüsselattribute miteinander verknüpft
- Daten aus mehreren Tabellen müssen zusammengeführt werden
- JOINS sind die Lösung

- Cross Join
- Outer Join
LEFT/RIGHT/FULL
- Inner Join

CROSS JOIN

- Es werden alle möglichen Kombinationen gebildet

Abteilung	
Abt_Nr	Bez
1	FIBU
2	EDV
3	EK

X

Personal		
A_Nummer	Name	Abt
1	Anton	1
2	Fritz	2
3	Karl	1
4	Otto	3
5	Paul	2

Cross - Join	
Abt	Abt_Nr
1	1
1	2
1	3
2	1
2	2
2	3
...	...

SELECT P.Abt, A.Abt_Nr

FROM Personal AS P

CROSS JOIN Abteilung AS A



- EQUI JOIN
 - ◆ Die Verknüpfung erfolgt über Primär- und Fremdschlüssel. Die Tabellen enthalten je eine Spalte mit korrespondierenden Werten.
 - ◆ In der ON-Klausel wird der Operator = verwendet
- NONEQUI JOIN
 - ◆ Es gibt keine korrespondierenden Spalten.
 - ◆ JOIN-Operator ist meistens BETWEEN ... AND ...

99,9% aller Joins sind EQUI JOINS

- ◆ Zwei Tabellen werden über FK und PK verknüpft
- ◆ z. B.: Rechnungs- und Kundentabelle stehen über die Kundennummer, die in beiden Tabellen enthalten ist, miteinander in Beziehung (n : 1)

z. B.: Gehalt der Mitarbeiter soll mit einer Gehaltsstufentabelle verknüpft werden, in der lediglich eine Ober- und eine Untergrenze vorhanden sind.
→ Das Gehalt aus der Mitarbeitertabelle entspricht nicht exakt einem der Grenzwerte, sondern liegt zwischen ihnen: BETWEEN wird als JOIN-Operator verwendet.

Inner Join

-

Abteilung	
Abt_Nr	Bez
1	FIBU
2	EDV
3	EK

Tabelle mit
Primärschlüssel PK

Angestellte		
A_Nummer	Name	Abt
1	Anton	1
2	Fritz	2
3	Karl	1
4	Otto	3
5	Paul	2

Tabelle mit
Fremdschlüssel FK

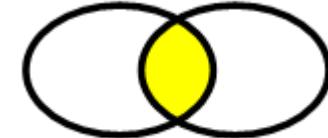
INNER - Join	
Name	Bez
Anton	FIBU
Fritz	EDV
Karl	FIBU
Otto	EK
Paul	EDV

```
SELECT AN.Name, AB.Bez
FROM Angestellte AS AN
    INNER JOIN
        Abteilung AS AB
    ON AN.Abt = AB.Abt_Nr
```



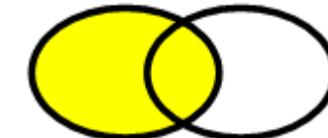
- INNER JOIN

Entsprechungen in beiden Tabellen



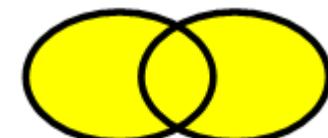
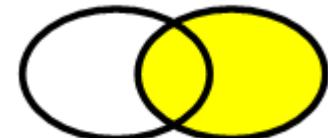
- LEFT/RIGHT OUTER JOIN

Aus einer der Tabellen werden auch jene Daten angezeigt, für die es keine Entsprechungen gibt.



- FULL OUTER JOIN

In beiden Tabellen werden alle Zeilen berücksichtigt



- ```
SELECT tab1.spalte_1, tab2.spalte_2
FROM tab1 INNER JOIN tab2
ON tab1.spalte_x = tab2.spalte_y
WHERE ...
```

  - ◆ ON gibt die Beziehung an (FK zu PK)
  - ◆ WHERE filtert Datensätze

## Alias

AS benennt Tabelle mit Aliasnamen

- Sinnvoll bei langen Tabellennamen

- ```
SELECT P.Name, G.Text
FROM tblPersonal AS P INNER JOIN tblGehalt AS G
ON P.Nr = G.Nr
```

- LEFT: alles aus der zuerst angegebenen Tabelle
 - ◆ Auch Werte aus der linken Tabelle, für die es keine Entsprechung in der rechten Tabelle gibt, werden angezeigt
- RIGHT: alles aus der zweiten Tabelle
 - ◆ Auch Zeilen aus der rechten Tabelle, für die es keine Entsprechung in der linken Tabelle gibt, werden angezeigt
- FULL: alles aus beiden Tabellen
- ```
SELECT a.spalte_1, b.spalte_2
FROM tab1 AS a LEFT OUTER JOIN tab2 AS b
ON a.spalte_x = b.spalte_y
```

# SELF JOIN

- JOIN einer Tabelle mit sich selbst
- Reflexive Beziehung, z. B. Darstellung von Hierarchien
- Dieselbe Tabelle erhält zwei verschiedene Aliasse
- ```
SELECT a.name AS Angestellter,
      b.name AS Vorgesetzter
     FROM Angestellte AS b
INNER JOIN Angestellte AS a
    ON b.vorg = a.a_nr
```

a_nr	name	beruf	vorg	gehalt
112	Mang	Progr	205	4300
117	Seel	Ing	401	4100
198	Feld	Kauf	401	4100
205	Wind	Organ		5100
301	Karl	Progr	401	5000
307	Mill	Progr	205	5400
350	Otto	Kauf	205	4700