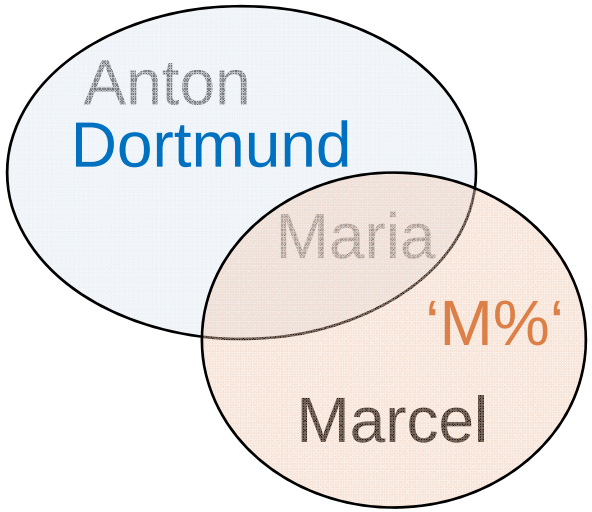
Datenbankanfragen

Mengenoperatoren

Fachhochschule
Dortmund
University of Applied Sciences

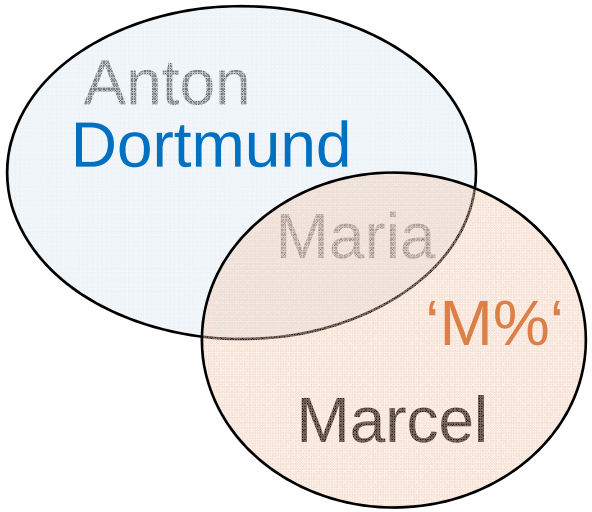
Fragestellung

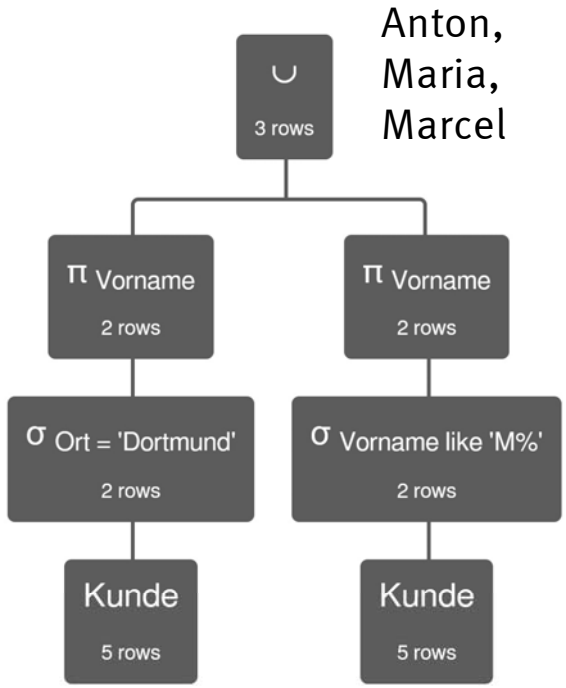


Mengenoperatoren in SQL

- Vereinigungsoperator UNION
- Schnittmengenoperator INTERSECT
- Differenzoperator MINUS

Vereinigungsoperator

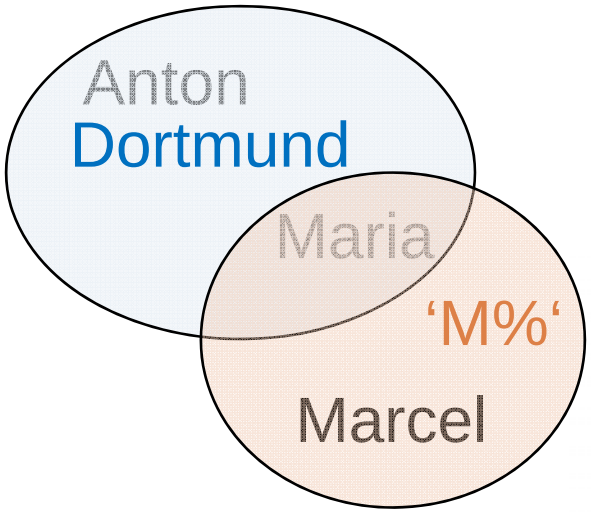




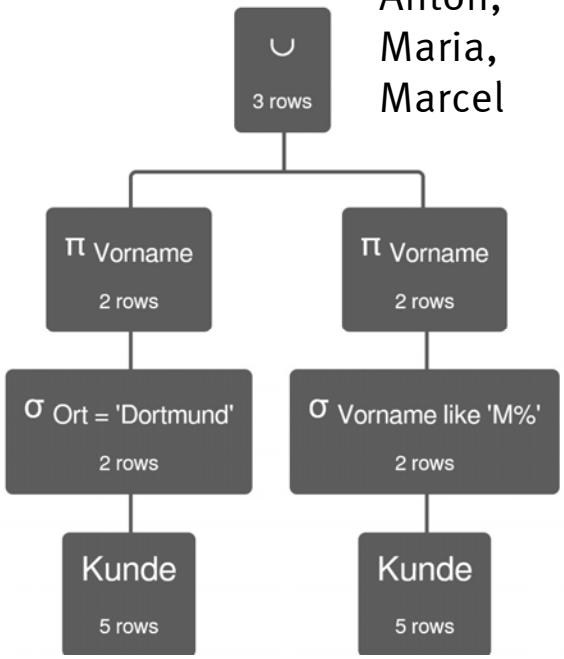
$\pi_{\text{Vorname}}\,\sigma_{\text{Ort = 'Dortmund'}}\,\text{Kunde} \cup \pi_{\text{Vorname}}\,\sigma_{\text{Vorname like 'M%'}}\,\text{Kunde}$

Vereinigungsoperator

Ohne Duplikate

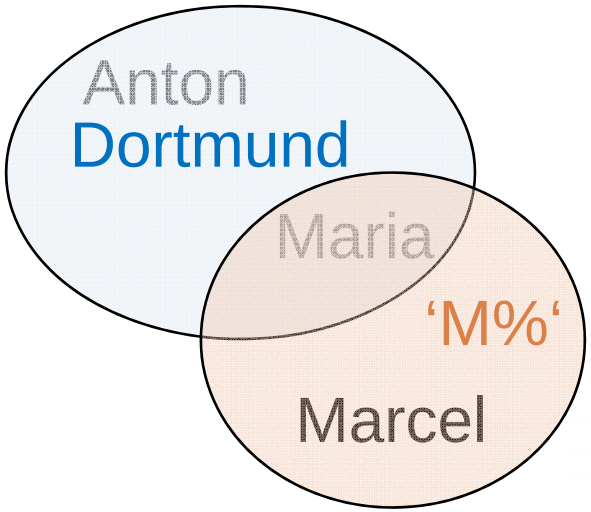


Anton,
Maria,
Marcel



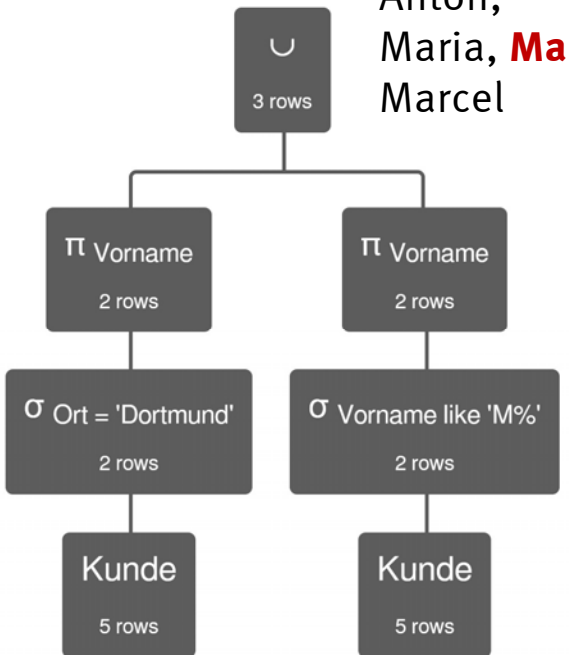

```
SELECT Vorname FROM Kunde WHERE Ort='Dortmund'
UNION
SELECT Vorname FROM Kunde WHERE Vorname LIKE 'M%'
```

Vereinigungsoperator



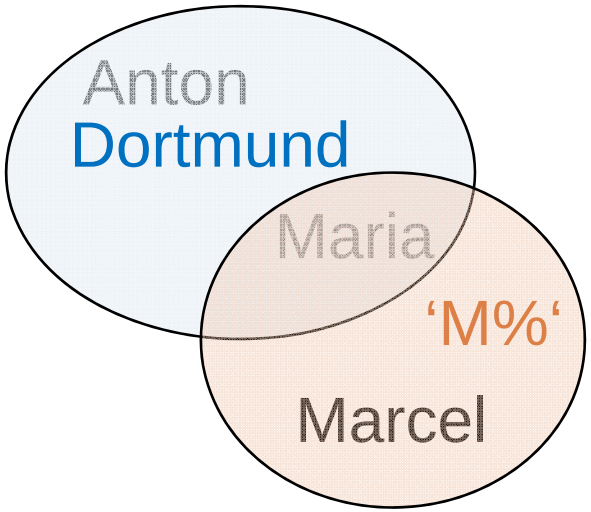
Mit Duplikaten

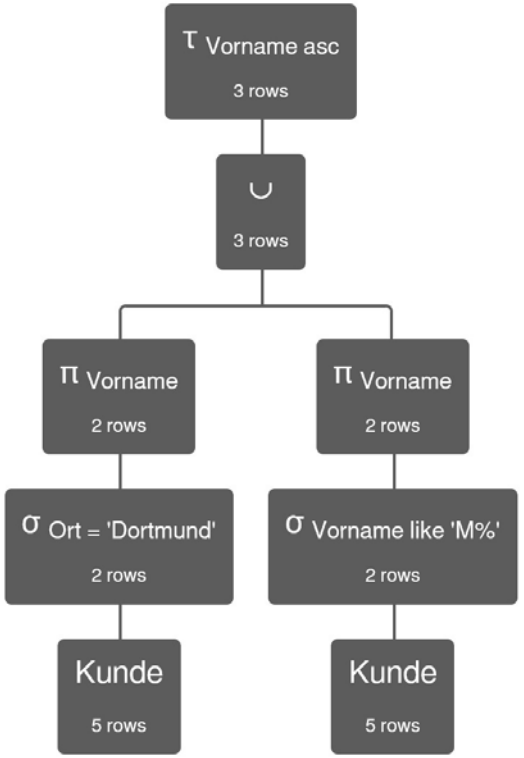
Anton,
Maria, **Maria**
Marcel




```
SELECT Vorname FROM Kunde WHERE Ort='Dortmund'
UNION ALL
SELECT Vorname FROM Kunde WHERE Vorname LIKE 'M%'
```

Sortieroperator

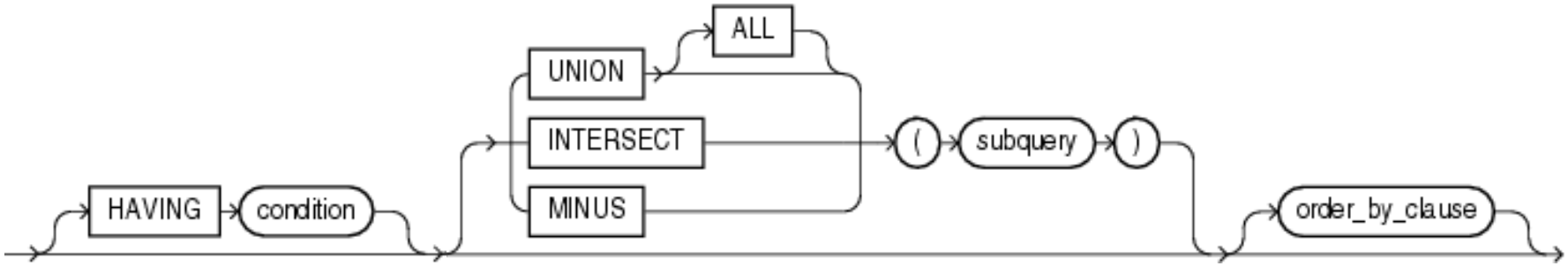





```
SELECT Vorname FROM Kunde WHERE Ort='Dortmund'
UNION
SELECT Vorname FROM Kunde WHERE Vorname LIKE 'M%'
ORDER BY Vorname ASC
```

Sortierung immer erst am Ende

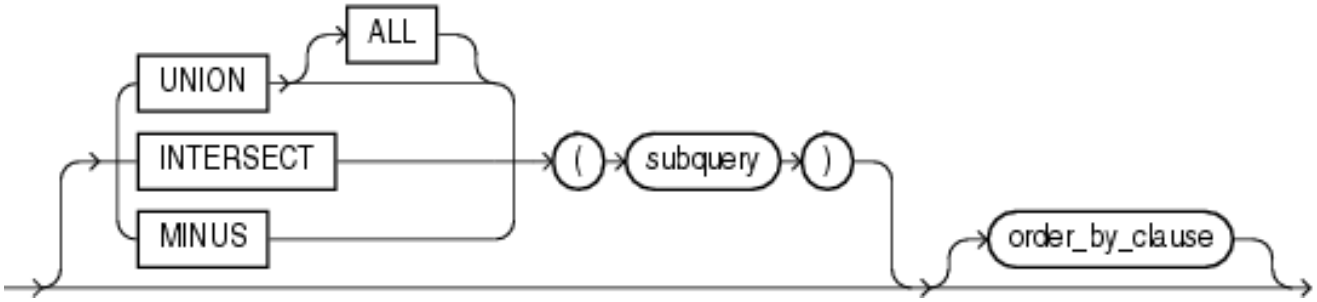
Sortieroperator



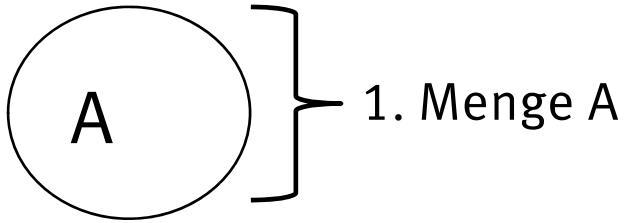

```
SELECT Vorname FROM Kunde WHERE Ort='Dortmund'
UNION ALL
SELECT Vorname FROM Kunde WHERE Vorname LIKE 'M%'
ORDER BY Vorname ASC
```

Sortierung immer erst am Ende

Syntax von Mengenoperationen

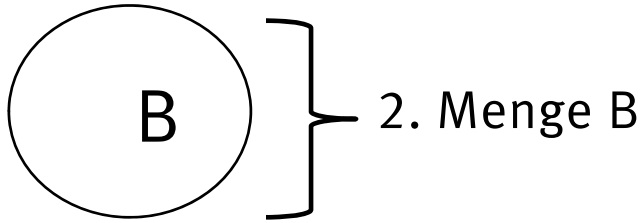


SELECT ‹Spalte$_1$›, ..., ‹Spalte$_n$›
FROM ‹Tabelle$_1$›, ..., ‹Tabelle$_m$›
WHERE ‹Bedingung- A›



{UNION | UNION ALL | INTERSECT | MINUS }

SELECT ‹Spalte$_1$›, ..., ‹Spalte$_n$›
FROM ‹Tabelle$_1$›, ..., ‹Tabelle$_m$›
WHERE ‹Bedingung- B›



ORDER BY ‹Spalte$_1$ › ...

Sortierung

we focus on students

Datenbankanfragen

Komplexe Anfragen aufbauen

Fachhochschule Dortmund
University of Applied Sciences

Fragestellung

Bei welchen Artikeln handelt es sich um
den günstigsten und den teuersten Artikel?

	ARTIKELNUMMER	ARTIKELNAME	AUTOR	PREIS
1	4812	Datenbanksysteme	Elmasri	29,95
2	4811	Datenbanksysteme	Kemper	28,9
3	4813	Märchen von Beedle dem Barden	Rowling	12,9
4	4814	Anatomie-interaktiv	Schattauer	19,95
5	4815	Anatomie	Sobotta	69,95
6	4816	Anatomie-Atlas	Smith	24,95
7	4820	Datenbank-Skript	FH	5,95

	ARTIKELNAME	PREIS	ART
1	Anatomie	69,95	Maximaler Preis
2	Datenbank-Skript	5,95	Minimaler Preis

Vorgehensweise

Komplexe Problemstellung

Bei welchen Artikeln handelt es sich um den günstigsten und den teuersten Artikel?

⇩

Aufspalten der Problemstellung in Teilprobleme, die separat gelöst werden können.

Minimalen (Maximalen) Preis ermitteln

Artikel zu dem minimalen (maximalen) Preis ermitteln

⇩

Integration der Lösungen der Teilprobleme zu einer Lösung des Gesamtproblems.

Artikel zu den minimalen und maximalen Preisen listen

Pseudospalte ROWNUM

Die Pseudospalte ROWNUM gibt die Nummer des Tupels in der Lesereihenfolge aus der Tabelle an. Eine Sortierung erfolgt erst nach der Festlegung der ROWNUM.

```
SELECT Preis, ROWNUM
FROM Artikel
ORDER BY Preis
```

	PREIS	ROWNUM
1	5,95	7
2	12,9	3
3	19,95	4
4	24,95	6
5	28,9	2
6	29,95	1
7	69,95	5
8	(null)	8

Zusammenfassen von Anfrageergebnissen

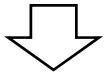

```
SELECT Artikelname, Preis
FROM  (SELECT * FROM  Artikel
        ORDER BY Preis ASC)
WHERE ROWNUM =1
UNION
SELECT Artikelname, Preis
FROM  (SELECT * FROM  Artikel
        WHERE Preis IS NOT NULL
        ORDER BY Preis DESC)
WHERE ROWNUM =1
```

Nachteil: Anzahl der zu selektierenden Tupel muss bekannt sein

Lösungen

Integration der Lösungen der Teilprobleme
zu einer Lösung des Gesamtproblems.

Artikel zu den minimalen und maximalen Preisen listen



```
SELECT Artikelname, Preis, 'Min' Art
FROM Artikel
WHERE Preis = (Select Min(Preis)
               FROM Artikel)
UNION
SELECT Artikelname, Preis, 'Max' Art
FROM Artikel
WHERE Preis = (Select Max(Preis)
               FROM Artikel)
```

```
SELECT Artikelname, Preis,
   CASE Preis
   WHEN  ‹Fall 1› THEN 'Min'
   WHEN  ‹Fall 2› THEN 'Max'
   ELSE 'Nix'
   END AS Art
FROM Artikel
WHERE ...
```

Lösungen


```
SELECT Artikelname, Preis, 'Minimaler Preis' Art
FROM Artikel
WHERE Preis = (Select Min(Preis) FROM Artikel)
UNION
SELECT Artikelname, Preis, 'Maximaler Preis' Art
FROM Artikel
WHERE Preis = (Select Max(Preis) FROM Artikel)
```

```
SELECT Artikelname, Preis,
   CASE Preis
   WHEN  (SELECT MIN(Preis)  FROM Artikel)  THEN 'Minimaler Preis'
   WHEN  (SELECT MAX(Preis) FROM Artikel)  THEN 'Maximaler Preis'
   ELSE 'Nix'
   END AS Art
FROM Artikel
WHERE Preis=(SELECT MIN(Preis) FROM Artikel)
OR       Preis=(SELECT MAX(Preis) FROM Artikel)
```

	ARTIKELNAME	PREIS	ART
1	Anatomie	69,95	Maximaler Preis
2	Datenbank-Skript	5,95	Minimaler Preis

Vorgehensweise

Komplexe Problemstellung

Bei welchen Artikeln handelt es sich um den günstigsten und den teuersten Artikel?

⇩ *DIVIDE*

Aufspalten der Problemstellung in Teilprobleme, die separat gelöst werden können.

Minimalen (Maximalen) Preis ermitteln

Artikel zu dem minimalen (maximalen) Preis ermitteln

⇩ *CONQUER*

Integration der Lösungen der Teilprobleme zu einer Lösung des Gesamtproblems.

Artikel zu den minimalen und maximalen Preisen listen

we focus on students

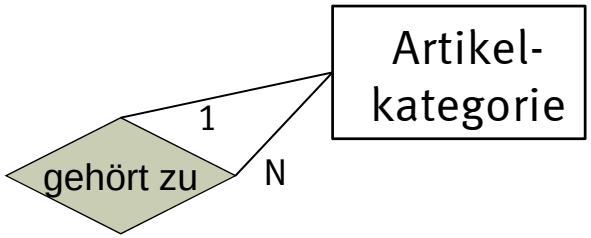
Datenbankanfragen

Rekursive Anfragen

Fachhochschule Dortmund
University of Applied Sciences

Fragestellung

Wie können rekursive Anfragen formuliert werden?



Relationales Modell:

	KATEGORIEKRZ	BEZEICHNUNG	OBERKATEGORIE
1	AK	Alle Kategorien	(null)
2	BUE	Bücher	AK
3	FBU	Fremdsprachige Bücher	AK
4	INF	Computer und Internet	BUE
5	DB	Datenbanken	INF
6	SQL	SQL	DB
7	FAN	Fantasy (Deutsch)	BUE
8	FAE	Fantasy (Fremdsprachig)	FBU
9	FIN	Computer and Internet (Fremdsprachig)	FBU

Unterkategorien abfragen - Unterabfragen

Liefere alle Unterkategorien!

Problem:
Bei der Nutzung von Unterabfragen muss explizit die Tiefe der Rekursion festgelegt werden.

```
SELECT '1. Ebene', Bezeichnung
FROM Artikelkategorie
WHERE KategorieKrz='AK'
UNION
SELECT '2. Ebene', Bezeichnung
FROM Artikelkategorie
WHERE Oberkategorie ='AK'
UNION
SELECT '3. Ebene', Bezeichnung
FROM Artikelkategorie
WHERE Oberkategorie IN
        (SELECT KategorieKrz
        FROM Artikelkategorie
        WHERE Oberkategorie ='AK')
```

	'1.EBENE'	BEZEICHNUNG
1	1. Ebene	Alle Kategorien
2	2. Ebene	Bücher
3	2. Ebene	Fremdsprachige Bücher
4	3. Ebene	Computer and Internet (Fremdsprachig)
5	3. Ebene	Computer und Internet
6	3. Ebene	Fantasy (Deutsch)
7	3. Ebene	Fantasy (Fremdsprachig)

Unterkategorien abfragen - Verbund

Liefere alle Unterkategorien!

Problem:
Bei der Nutzung von Verbundoperationen muss explizit die Tiefe der Rekursion festgelegt werden.

Mit Verbundoperation

```
SELECT ak1.*, ak2.*, ak3.*
FROM Artikelkategorie ak1                    1. Ebene
      JOIN Artikelkategorie ak2              2. Ebene
        ON ak1.KategorieKrz = ak2.Oberkategorie
      JOIN Artikelkategorie ak3              3. Ebene
        ON ak2.KategorieKrz = ak3.Oberkategorie
WHERE ak1.KategorieKrz='AK';
```

1. Ebene *2. Ebene* *3. Ebene*

	BEZEICHNUNG	BEZEICHNUNG_1	BEZEICHNUNG_2
1	Alle Kategorien	Bücher	Computer und Internet
2	Alle Kategorien	Bücher	Fantasy (Deutsch)
3	Alle Kategorien	Fremdsprachige Bücher	Fantasy (Fremdsprachig)
4	Alle Kategorien	Fremdsprachige Bücher	Computer and Internet (Fremdsprachig)

Die WITH-Klausel

Mit der WITH-Klausel (Sub-Query Factoring) kann eine Unterabfrage zu Beginn der Abfrage deklariert werden. Diese kann nur in der FROM-Klausel genutzt werden.

```
WITH ak1 AS (SELECT KategorieKrz, Bezeichnung, Oberkategorie
                    FROM    Artikelkategorie
                    WHERE  KategorieKrz= 'AK')

SELECT '1. Ebene', Bezeichnung FROM ak1

UNION

SELECT '2. Ebene', ak2.Bezeichnung
  FROM ak1 JOIN Artikelkategorie ak2 ON ak2.Oberkategorie=ak1.KategorieKrz
```

Rekursive Abfragen – WITH-Rekursiv

Berechne die Zweierpotenzen!

$$2^n = \begin{cases} 1, & \text{für } n = 0 \quad \text{Induktionsanfang} \\ 2 \cdot 2^{n-1}, & \text{für } n > 0 \quad \text{Induktionsschritt} \end{cases}$$

```
WITH zweierpotenzen (n, potenz)
AS
(SELECT 0,1 FROM dual        Induktionsanfang
 UNION ALL
 SELECT n+1, 2*potenz        Induktionsschritt
 FROM zweierpotenzen
 WHERE n<9                   Abbruchbedingung
)
SELECT * FROM zweierpotenzen
```

ab Oracle 11g

N	POTENZ
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512

Rekursive Abfragen – WITH-Rekursiv

Liefere alle Unterkategorien!

	KATEGORIEKRZ	OBERKATEGORIE
1	AK	(null)
2	BUE	AK
3	FAN	BUE
4	INF	BUE
5	DB	INF
6	SQL	DB
7	FBU	AK
8	FAE	FBU
9	FIN	FBU

```
WITH ak(KategorieKrz, Oberkategorie) AS (
  -- Start-Kategorie.
  SELECT KategorieKrz,
         Oberkategorie
  FROM   Artikelkategorie
  WHERE  KategorieKrz = 'AK'
  UNION ALL
  -- Unterkategorie
  SELECT child.KategorieKrz, child.Oberkategorie
  FROM   Artikelkategorie child, ak
  WHERE  child.Oberkategorie = ak.KategorieKrz
)
SEARCH DEPTH FIRST BY KategorieKrz SET order1
SELECT KategorieKrz, Oberkategorie
FROM   ak
ORDER BY order1;
```

ab Oracle 11g

Breitensuche: BREADTH FIRST, Tiefensuche: DEPTH FIRST

Ober- und Unterkategorien abfragen

Mit der Oracle-spezifischen Erweiterung CONNECT BY PRIOR können rekursive hierarchische Abfragen durchgeführt werden.

Alle Unterkategorien

```
SELECT * FROM Artikelkategorie
START WITH Oberkategorie='BUE'
CONNECT BY PRIOR KategorieKrz=Oberkategorie
```

	KATEGORIEKRZ	BEZEICHNUNG	OBERKATEGORIE
1	FAN	Fantasy (Deutsch)	BUE
2	INF	Computer und Internet	BUE
3	DB	Datenbanken	INF
4	SQL	SQL	DB

Alle Oberkategorien

```
SELECT * FROM Artikelkategorie
START WITH KategorieKrz='SQL'
CONNECT BY PRIOR Oberkategorie=KategorieKrz
```

	KATEGORIEKRZ	BEZEICHNUNG	OBERKATEGORIE
1	SQL	SQL	DB
2	DB	Datenbanken	INF
3	INF	Computer und Internet	BUE
4	BUE	Bücher	AK
5	AK	Alle Kategorien	(null)

Unterkategorien abfragen – Oracle-Spezifisch

Mit der Oracle-spezifischen Erweiterung CONNECT BY PRIOR können rekursive hierarchische Abfragen durchgeführt werden.

```
SELECT Level, Oberkategorie, KategorieKrz,
       SYS_CONNECT_BY_PATH(KategorieKrz, '/') Pfad,
       CONNECT_BY_ISLEAF   istBlatt,
       CONNECT_BY_ISCYCLE  istZyklus,
       CONNECT_BY_ROOT     KategorieKrz Wurzel
FROM Artikelkategorie
START WITH Oberkategorie IS NULL
CONNECT BY NOCYCLE PRIOR KategorieKrz=Oberkategorie;
```

Pseudospalten

	LEVEL	OBERKATEGORIE	KATEGORIEKRZ	PFAD	ISTBLATT	ISTZYKLUS	WURZEL
1	1	(null)	AK	/AK	0	0	AK
2	2	AK	BUE	/AK /BUE	0	0	AK
3	3	BUE	FAN	/AK /BUE/FAN	1	0	AK
4	3	BUE	INF	/AK /BUE/INF	0	0	AK
5	4	INF	DB	/AK /BUE/INF/DB	0	0	AK
6	5	DB	SQL	/AK /BUE/INF/DB /SQL	1	0	AK
7	2	AK	FBU	/AK /FBU	0	0	AK
8	3	FBU	FAE	/AK /FBU/FAE	1	0	AK
9	3	FBU	FIN	/AK /FBU/FIN	1	0	AK

Zusammenfassung

- Rekursive Anfragen
  1. Mehrere Anfragen, die mit UNION verknüpft werden
  2. Eine Anfrage mit mehrfachen Verbundoperationen mit derselben Tabelle
  3. (Rekursive) WITH-Klausel
  4. CONNECT-BY-PRIOR