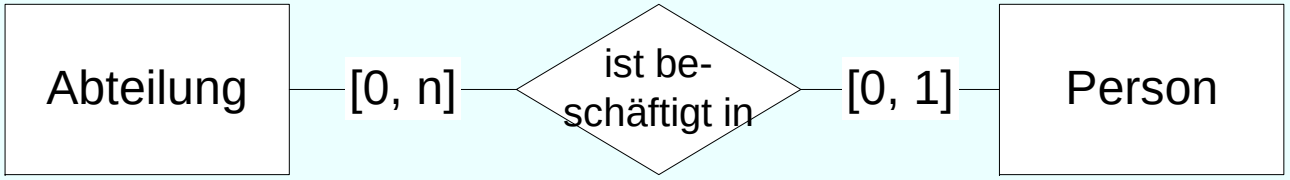
Diskussion: Personal (1)

ER-Diagramm:



Umsetzung ins Relationenmodell?

Diskussion: Personal (2)

Zusätzliche Regel:

In jeder Abteilung (Person) muss mindestens eine beschäftigt sein ([1, N])

Umsetzung ins Relationenmodell?

Zusätzliche Regel:

Jeder Angestellte (Person) muss in einer Abteilung beschäftigt sein ([1, 1])

Umsetzung ins Relationenmodell?

Die relationale Algebra

- σ Selektion
- π Pojektion
- x Kreuzprodukt
- $\bowtie$ Join (Verbund)
- ρ Umbenennung
- $\ltimes$ Semi-Join (linker)
- $\rtimes$ Semi-Join (rechter)
- $⟕$ linker äußerer Join
- $⟖$ rechter äußerer Join

Allg. Mengenoperationen:

- $-$ Differenz
- $\div$ Division
- $\cup$ Vereinigung
- $\cap$ Durchschnitt

Beispiel Mengendurchschnitt

Finde die *PersNr* aller C4-Professoren, die mindestens eine Vorlesung halten.

$$\Pi_{\text{PersNr}}(\rho_{\text{PersNr}\leftarrow\text{gelesenVon}}(\text{Vorlesungen})) \cap \Pi_{\text{PersNr}}(\sigma_{\text{Rang=C4}}(\text{Professoren}))$$

→ prozedural !

Relationaler Tupelkalkül

Eine Anfrage im Relationenkalkül hat die Form

$$\{t \mid P(t)\}$$

mit t Tupelvariable und P Prädikat

einfaches **Beispiel:**

C4-Professoren

$$\{p \mid p \in \text{Professoren} \wedge p.\text{Rang} = \text{'C4'}\}$$

Relationaler Tupelkalkül: weiteres Beispiel

Studenten mit mindestens einer Vorlesung von Curie

$$\{s \mid s \in \text{Studenten} \wedge \exists h \in \text{hören}(s.\text{MatrNr}=h.\text{MatrNr} \wedge \exists v \in \text{Vorlesungen}(h.\text{VorlNr}=v.\text{VorlNr} \wedge \exists p \in \text{Professoren}(p.\ \text{PersNr}=v.\text{gelesenVon} \wedge p.\text{Name} = \text{'Curie'}))))\}$$

Dieselbe Anfrage in SQL …
… belegt die Verwandtschaft

```
select s.*
from Studenten s
where exists (
        select h.*
        from hören h
        where h.MatrNr = s.MatrNr and exists (
              select *
              from Vorlesungen v
              where v.VorlNr = h.VorlNr and exists (
                    select *
                     from Professoren p
                     where p.Name =‚Curie' and
                                  p.PersNr= v.gelesenVon )))
```

Relationaler Domänenkalkül

Anfrage im Domänenkalkül hat die Form:

$\{[v1, v2, \ldots, vn] \mid P(v1, \ldots, vn)\}$

mit v1, . . . , v2 Domänenvariablen und P Prädikat

Beispiel:

MatrNr und Namen der Prüflinge von Sokrates

$$\{[m, n] \mid \exists\, ([m, n, s] \in Studenten$$
$$\wedge \exists\, p, v, g\, ([m, p, v, g] \in prüfen$$
$$\wedge \exists\, a, r, b\, ([p, a, r, b] \in Professoren$$
$$\wedge\, a = \text{‘Sokrates’}))\}$$

Ausdruckskraft

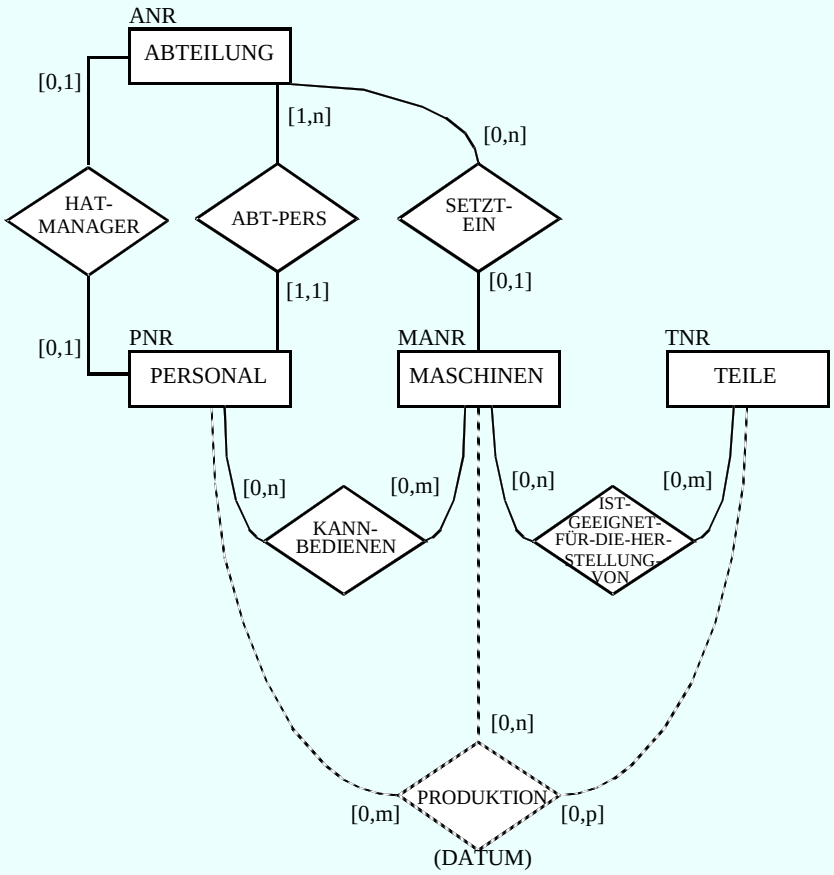
Die drei Sprachen

- relationale Algebra
- relationaler Tupelkalkül, eingeschränkt auf sichere Ausdrücke
- relationaler Domänenkalkül, eingeschränkt auf sichere Ausdrücke

sind gleich mächtig

$\{n \mid \neg(n \in Professoren)\}$ *z.B. ist nicht sicher, da das Ergebnis* unendlich ist

Fertigungsdatenbank



ABTEILUNG (<u>ANR</u>, ANAME, AMNR)

PERSONAL (<u>PNR</u>, PNAME, PBERUF, ANR)

MASCHINEN (<u>MANR</u>, MFABRIKAT, MTYP, ANR)

TEILE (<u>TNR</u>, TBEZ, TGEWICHT)

KANN-BEDIENEN (<u>PNR, MANR</u>)

GEEIGNET-FÜR-DIE-HERSTELLUNG-VON (<u>MANR, TNR</u>)

PRODUKTION (<u>PNR, MANR, TNR, DATUM</u>, MENGE)

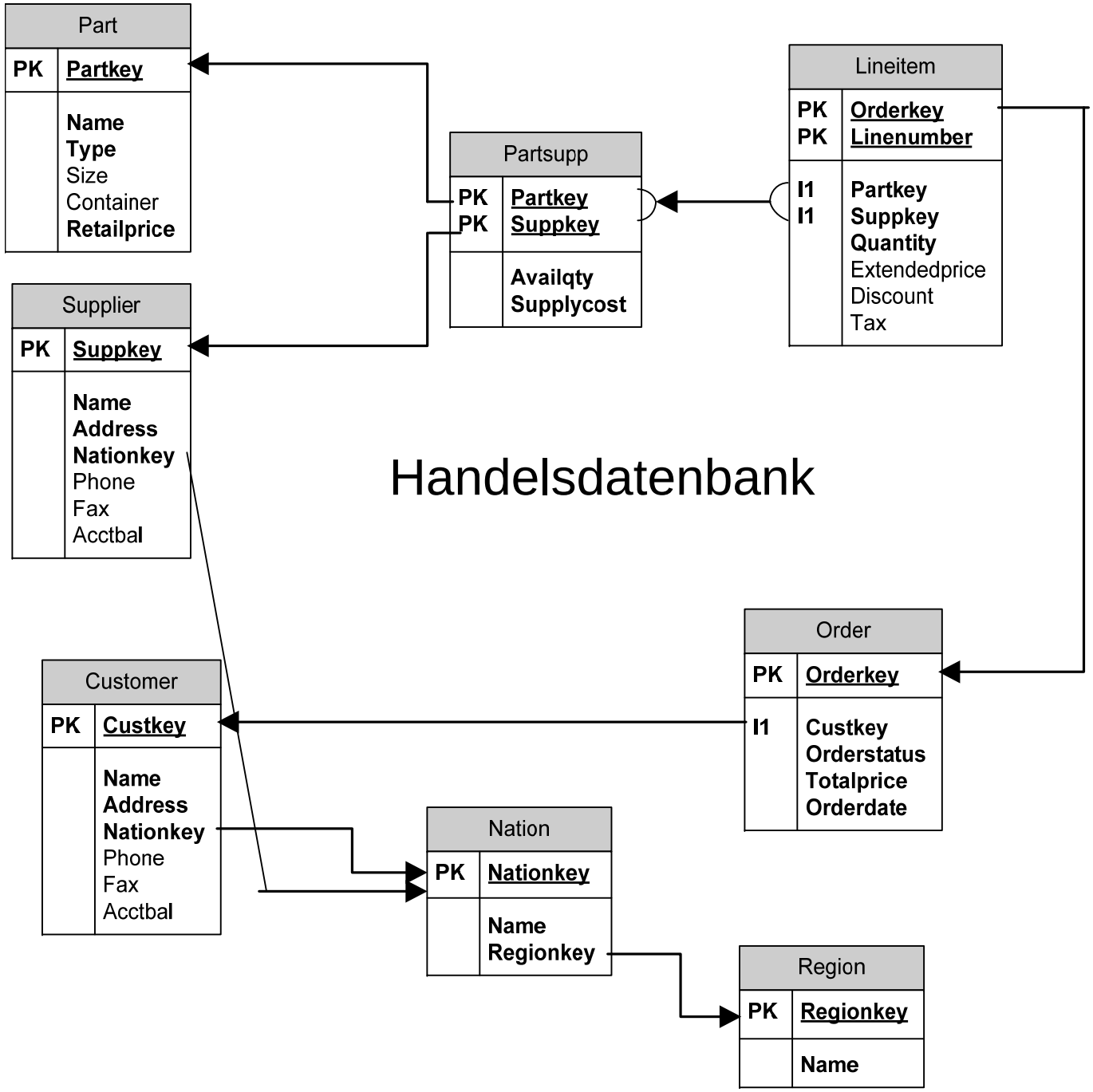
SQL - DRL

Tutorials für erste Einblicke in SQL:

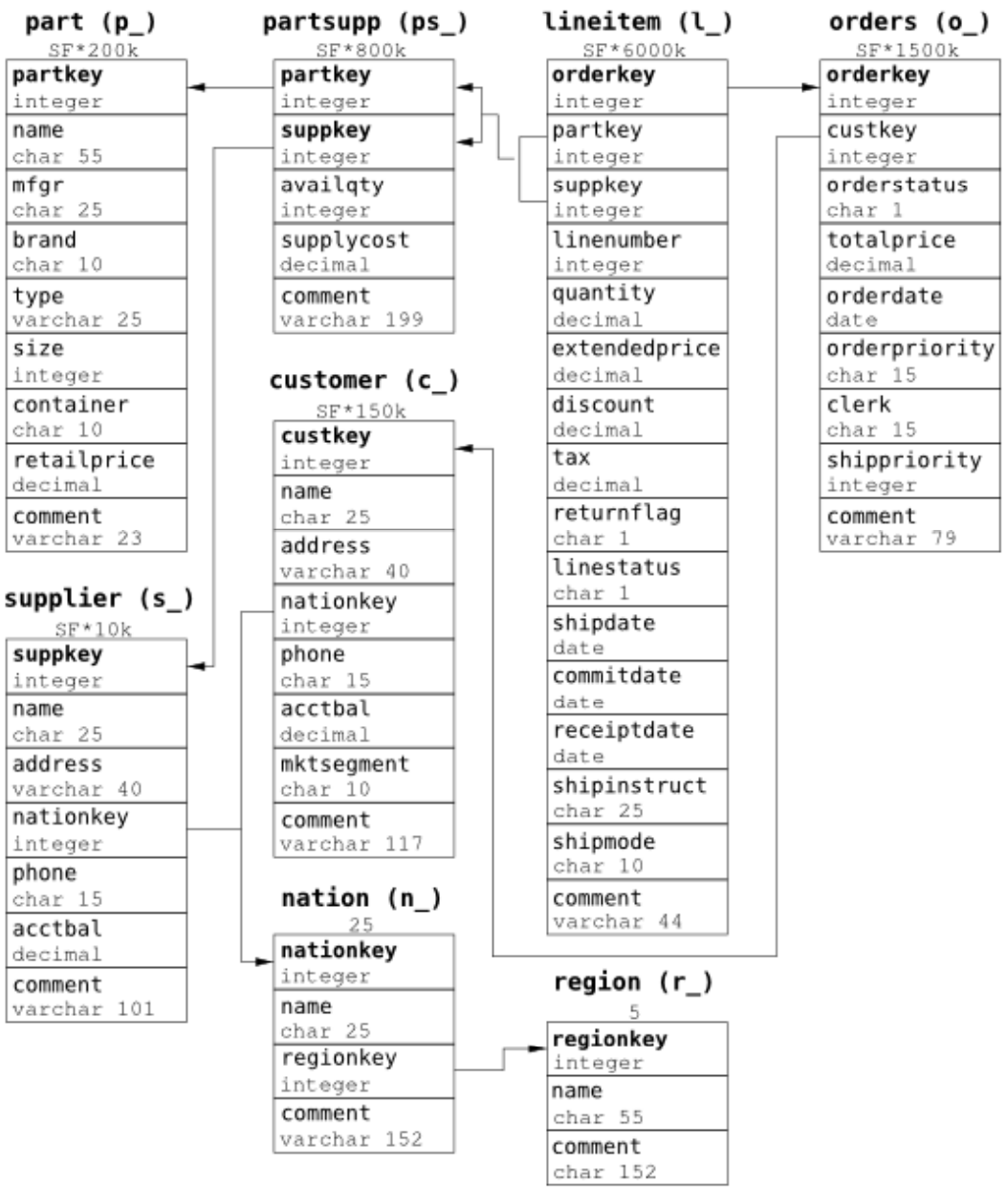
- sql.lernenhoch2.de/lernen/
- www.w3schools.com/sql

Webschnittstellen für SQL:

- sqlfiddle.com (MySQL, Oracle, PostgreSQL, SQLite, MS SQL):
  auch Tabellen anlegen möglich
- hyper-db.com/interface.html (HyPer):
  Universitätsdatenbank, TPC-H Schema
  Query-Ausführungspläne


Part
PK Partkey
Name
Type
Size
Container
Retailprice
Partsupp
PK Partkey
PK Suppkey
Availqty
Supplycost
Lineitem
PK Orderkey
PK Linenumber
I1 Partkey
I1 Suppkey
Quantity
Extendedprice
Discount
Tax
Supplier
PK Suppkey
Name
Address
Nationkey
Phone
Fax
Acctbal
Handelsdatenbank
Order
PK Orderkey
I1 Custkey
Orderstatus
Totalprice
Orderdate
Customer
PK Custkey
Name
Address
Nationkey
Phone
Fax
Acctbal
Nation
PK Nationkey
Name
Regionkey
Region
PK Regionkey
Name

TPC-H Schema


part (p_)
SF*200k
partkey integer
name char 55
mfgr char 25
brand char 10
type varchar 25
size integer
container char 10
retailprice decimal
comment varchar 23
partsupp (ps_)
SF*800k
partkey integer
suppkey integer
availqty integer
supplycost decimal
comment varchar 199
lineitem (l_)
SF*6000k
orderkey integer
partkey integer
suppkey integer
linenumber integer
quantity decimal
extendedprice decimal
discount decimal
tax decimal
returnflag char 1
linestatus char 1
shipdate date
commitdate date
receiptdate date
shipinstruct char 25
shipmode char 10
comment varchar 44
orders (o_)
SF*1500k
orderkey integer
custkey integer
orderstatus char 1
totalprice decimal
orderdate date
orderpriority char 15
clerk char 15
shippriority integer
comment varchar 79
customer (c_)
SF*150k
custkey integer
name char 25
address varchar 40
nationkey integer
phone char 15
acctbal decimal
mktsegment char 10
comment varchar 117
supplier (s_)
SF*10k
suppkey integer
name char 25
address varchar 40
nationkey integer
phone char 15
acctbal decimal
comment varchar 101
nation (n_)
25
nationkey integer
name char 25
regionkey integer
comment varchar 152
region (r_)
5
regionkey integer
name char 55
comment char 152

Gerüst SQL-Anfrage

select	<Attributliste>		5
from	<Relationenliste>		1
[where	<Prädikatsliste>		2
group by	<Attributliste>		3
having	<Prädikatsliste>		4
order by	<Attributliste>		6
fetch first	<Anzahl Ergebnistupel>	**]**	7

Einfaches Beispiel

Anfrage:
"Gib mir die gesamte Information über alle Professoren„

select *

from Professoren

Professoren

PersNr	Name	Rang
2136	Curie	C4
2137	Kant	C4
2126	Russel	C4
2125	Sokrates	C4
2134	Augustinus	C3
2127	Kopernikus	C3
2133	Popper	C3

Ergebnis

PersNr	Name	Rang
2136	Curie	C4
2137	Kant	C4
2126	Russel	C4
2125	Sokrates	C4
2134	Augustinus	C3
2127	Kopernikus	C3
2133	Popper	C3

Attribute selektieren

Anfrage:
"Gib mir die PersNr und den Namen aller Professoren„

select PersNr, Name

from Professoren

Professoren

PersNr	Name	Rang
2136	Curie	C4
2137	Kant	C4
2126	Russel	C4
2125	Sokrates	C4
2134	Augustinus	C3
2127	Kopernikus	C3
2133	Popper	C3

Ergebnis

PersNr	Name
2136	Curie
2137	Kant
2126	Russel
2125	Sokrates
2134	Augustinus
2127	Kopernikus
2133	Popper

Duplikateliminierung

• Im Gegensatz zur relationalen Algebra (Mengen!) eliminert SQL keine Duplikate

• Falls Duplikateliminierung erwünscht, muss das Schlüsselwort **distinct** benutzt werden

• Beispiel:
Anfrage: „Welche Ränge haben Professoren?„

select distinct Rang

from Professoren

Ergebnis:

Rang
C3
C4

Where Klausel: Tupel selektieren

Anfrage:
"Gib mir die PersNr und den Namen aller Professoren, die den Rang C4 haben„

```
select   PersNr, Name
from     Professoren
where    Rang= ´C4´;
```

Ergebnis:

PersNr	Name
2125	Sokrates
2126	Russel
2136	Curie
2137	Kant

Where Klausel: Prädikate

• Prädikate in der where-Klausel können logisch kombiniert werden mit:

AND, OR, NOT

• Als Vergleichsoperatoren können verwendet werden:

=, <,<=, >,>=, between, like

Beispiel für between

Anfrage:
"Gib mir die Namen aller Studenten, die zwischen 1987-01-01 und 1989-01-01 geboren wurden„

```
select Name
from   Student
where Geburtstag between 1987-01-01 and 1989-01-01;
```

Anfrage äquivalent zu:

```
select Name
from Student
where Geburtstag >= 1987-01-01
        and Geburtstag <= 1989-01-01;
```

String-Vergleiche

• Stringkonstanten müssen in einfachen Anführungszeichen eingeschlossen sein

Anfrage:
"Gib mir alle Informationen über den Professor
mit dem Namen Kant„

```
select *
from    Professoren
where Name = 'Kant';
```

Suche mit Jokern (Wildcards)

Anfrage:
"Gib mir alle Informationen über Professoren,
deren Namen mit einem K anfängt“

select *
from Professoren
where Name **like** ’K%’;

Mögliche Joker:
- _ steht für ein beliebiges Zeichen
- % steht für eine beliebige Zeichenkette (auch der Länge 0)

Nullwerte

- In SQL gibt es einen speziellen Wert **NULL**
- Dieser Wert existiert für alle verschiedenen Datentypen und repräsentiert Werte, die
  - *unbekannt* oder
  - *nicht verfügbar* oder
  - *nicht anwendbar* sind.
- Nullwerte können auch im Zuge der Anfrageauswertung entstehen
- Auf NULL wird mit **is NULL** geprüft:

Beispiel:

```
select   *
from     Professoren
where    Raum is NULL;
```

Nullwerte cont.

- Nullwerte werden in arithmetischen Ausdrücken durchgereicht: mindestens ein Operand NULL → Ergebnis ebenfalls NULL
- manchmal sehr überraschende Anfrageergebnisse, wenn Nullwerte vorkommen, z.B.:

```
select count (*)
from Studenten
where Semester < 13 or Semester > = 13
```

- Wenn es Studenten gibt, deren Semester-Attribut den Wert NULL hat, werden diese nicht mitgezählt
- Der Grund liegt in dreiwertiger Logik unter Einbeziehung von NULL-Werten:

Auswertung bei Null-Werten

- SQL: dreiwertige Logik, mit den Werten **true**, **false** und **unknown**
- **unknown** liefern Vergleichsoperationen zurück, wenn mindestens eines ihrer Argumente NULL ist.
- In einer **where**-Bedingung werden nur Tupel weitergereicht, für die die Bedingung **true** ist. Insbesondere werden Tupel, für die die Bedingung zu **unknown** auswertet, nicht ins Ergebnis aufgenommen.
- Bei einer Gruppierung wird NULL als ein eigenständiger Wert aufgefasst und in eine eigene Gruppe eingeordnet.
- Logische Ausdrücke werden nach den folgenden Tabellen berechnet:

Dreiwertige Logik-Tabellen

not	
true	false
unknown	unknown
false	true

and	true	unknown	false
true	true	unknown	false
unknown	unknown	unknown	false
false	false	false	false

or	true	unknown	false
true	true	true	true
unknown	true	unknown	unknown
false	true	unknown	false