



# Information Retrieval

## 02: Boolean Retrieval

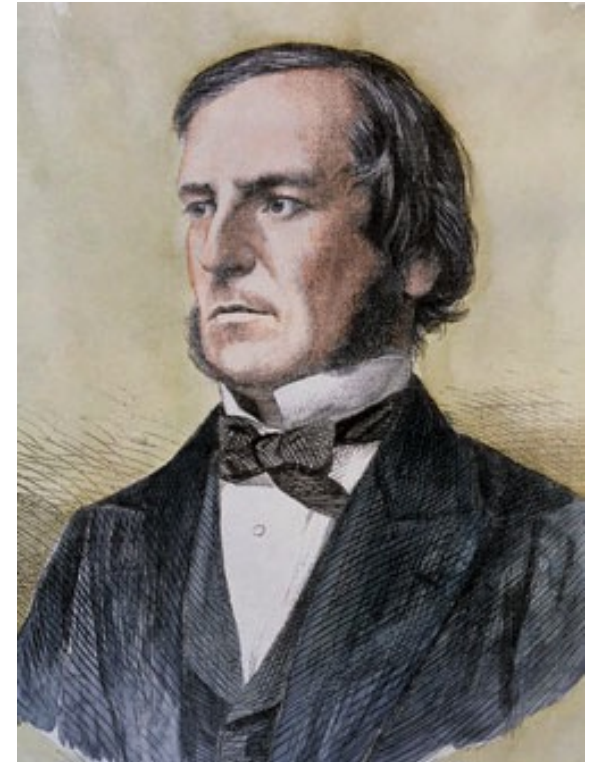
Philipp Schaer, Technische Hochschule Köln, Cologne, Germany

Version: 2022-04-07

# Überblick

- Boolesche Anfragen und das Boolesche Retrievalmodell
- Beispielsysteme
- Beispielanfragen
- Term-Dokument-Matrizen
- Vor- und Nachteile des Booleschen Modells

Was die Wikipedia sagt: George Boole (\* 2. November 1815 in Lincoln, England; † 8. Dezember 1864 in Ballintemple, in der Grafschaft Cork, Irland) war ein englischer Mathematiker (Autodidakt), Logiker und Philosoph.





# Beispiel 1: Bibliothekskataloge



Deutscher Bundestag

Menü

## Elektronischer Katalog

### Erweiterte Suche

 Suche starten
  Leeren
  Wiederholen

 Hilfe
  Katalog verlassen

In den mit \* gekennzeichneten Kategorien suchen Sie über den Gesamtbestand der Bibliothek, ansonsten über den Bestand ab Erscheinungsjahr 1987.

Bitte füllen Sie mindestens ein Suchfeld aus


	<input type="text" value="Titel / Stichwort *"/>	<input type="text"/>	Register	
<div> <input checked="" type="checkbox"/> UND  <input type="checkbox"/> NICHT  <input type="checkbox"/> ODER  <input type="checkbox"/> UND         </div>	<input type="text" value="Person *"/>	<input type="text"/>	Register	
	<input type="text" value="Körperschaft"/>	<input type="text"/>	Register	
<input type="text" value="UND"/>	<input type="text" value="Schlagwort"/>	<input type="text"/>	Register	

Berichtsjahr präzise

Jahr von

Jahr bis

# Beispiel 2: Chefkoch.de


**CHEFKOCH.DE**

[Kostenlos registrieren](#)
[Newsletter](#)
[Hilfe/FAC](#)

[Startseite](#)
[Magazin](#)
[Rezepte](#)
[Specials](#)
[Community](#)
[Bilder](#)
[Videos](#)
[Blog](#)
[Login](#)
[myChefkoch](#)

[Startseite](#) » [Rezeptsuche](#)

## Rezeptsuche

[Erweiterte Suche](#)

### Erweiterte Suche

☒ Rezeptnamen durchsuchen
  max. Arbeitszeit in Min.

☒ Zutaten durchsuchen
  max. Kalorien in kcal


☐ Beschreibungen durchsuchen
  max. Alter der Rezepte in Tagen

☐ nur mit Bild

### Nützliche Rezeptlinks

- » [Neue Rezepte](#)
- » [Neue Rezeptbilder](#)
- » [Rezeptsammlungen](#)
- » [Zufallsrezept](#)
- » [Zutatenverwertung](#)
- » [Rezept des Tages](#)
- » [... als Newsletter](#)
- » [... für Ihre Homepage](#)
- » [Rezepte einsenden](#)

### Rezept des Tages


[Mediterranes Ofengemüse](#)

# Beispiel 3: Erweiterte Suche bei Google



## Erweiterte Suche

Seiten suchen, die...

alle diese Wörter enthalten:

genau dieses Wort oder diese  
Wortgruppe enthalten:

eines dieser Wörter enthalten:

keines der folgenden Wörter  
enthalten:

Zahlen enthalten im Bereich  
von:

bis

# Boolesche Retrievalmodell und Anfragen

Das **Boolesche Retrievalmodell** kann alle Anfragen auflösen, die sich als ein **Boolescher Ausdruck** formulieren lassen.

- Es erlaubt den Einsatz der Operatoren **UND**, **ODER** sowie **NICHT** um einzelne Anfrageterme zu verknüpfen.
- Jedes Dokument ist in diesem Modell eine **Menge von Termen (bag of words)**, die keiner besonderen Ordnung folgen.
- Es ist sehr präzise: Ein Dokument passt zur Anfrage oder nicht!

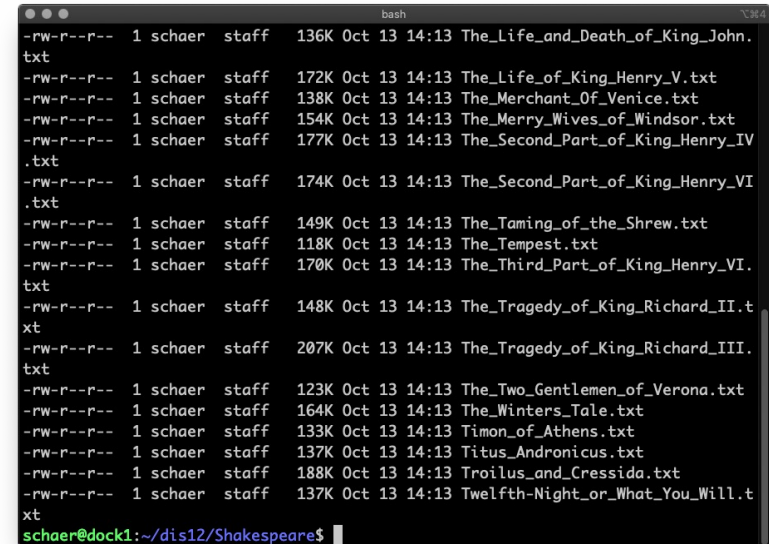
Im **professionellen Einsatz** seit mehr als 40 Jahren und immer noch sehr beliebt.

- Man weiß, was man bekommt – Nachvollziehbarkeit des Ergebnisses.
- Viele Suchsysteme basieren auf dem Booleschen Modell...

# Unstrukturierte Daten im Jahr 1680

Welches Stück von Shakespeare enthält die Wörter ***Brutus*** UND ***Caesar*** aber ***NICHT Calphurnia***?

- Ein Versuch: Wir lesen alle Texte, die Shakespeare geschrieben hat, Zeile für Zeile durch und merken uns alle Stücke, die die Wörter ***Brutus*** und ***Caesar enthalten***, danach werden alle Stücke mit dem Wort ***Calphurnia*** gelöscht.



```
bash
-rw-r--r-- 1 schaer staff 136K Oct 13 14:13 The_Life_and_Death_of_King_John.
txt
-rw-r--r-- 1 schaer staff 172K Oct 13 14:13 The_Life_of_King_Henry_V.txt
-rw-r--r-- 1 schaer staff 138K Oct 13 14:13 The_Merchant_Of_Venice.txt
-rw-r--r-- 1 schaer staff 154K Oct 13 14:13 The_Merry_Wives_of_Windsor.txt
-rw-r--r-- 1 schaer staff 177K Oct 13 14:13 The_Second_Part_of_King_Henry_IV
.txt
-rw-r--r-- 1 schaer staff 174K Oct 13 14:13 The_Second_Part_of_King_Henry_VI
.txt
-rw-r--r-- 1 schaer staff 149K Oct 13 14:13 The_Taming_of_the_Shrew.txt
-rw-r--r-- 1 schaer staff 118K Oct 13 14:13 The_Tempest.txt
-rw-r--r-- 1 schaer staff 170K Oct 13 14:13 The_Third_Part_of_King_Henry_VI.
txt
-rw-r--r-- 1 schaer staff 148K Oct 13 14:13 The_Tragedy_of_King_Richard_II.t
xt
-rw-r--r-- 1 schaer staff 207K Oct 13 14:13 The_Tragedy_of_King_Richard_III.
txt
-rw-r--r-- 1 schaer staff 123K Oct 13 14:13 The_Two_Gentlemen_of_Verona.txt
-rw-r--r-- 1 schaer staff 164K Oct 13 14:13 The_Winters_Tale.txt
-rw-r--r-- 1 schaer staff 133K Oct 13 14:13 Timon_of_Athens.txt
-rw-r--r-- 1 schaer staff 137K Oct 13 14:13 Titus_Andronicus.txt
-rw-r--r-- 1 schaer staff 188K Oct 13 14:13 Troilus_and_Cressida.txt
-rw-r--r-- 1 schaer staff 137K Oct 13 14:13 Twelfth-Night_or_What_You_Will.t
xt
schaer@dock1:~/dis12/Shakespeare$
```

Warum ist das eine schlechte Idee...?

# Term-Dokument-Matrix

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calphurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

***Brutus UND Caesar  
ABER NICHT Calphurnia***

1 wenn **Stück** das **Wort**  
enthält, ansonsten 0



# Einschub: Boolesche Algebra

- Die boolesche Algebra hat nur die zwei Elemente 0 und 1.
- Es sind die folgenden Verknüpfungen definiert:
  - **Konjunktion** ( $\wedge$ ) bzw. „und“,
  - **Disjunktion** ( $\vee$ ) bzw. „oder“ und
  - **Negation** ( $\neg$ ) bzw. „nicht“.
- **Klammerungen** für Gruppierungen sind erlaubt.

Konjunktion

$\wedge$	0	1
0	0	0
1	0	1

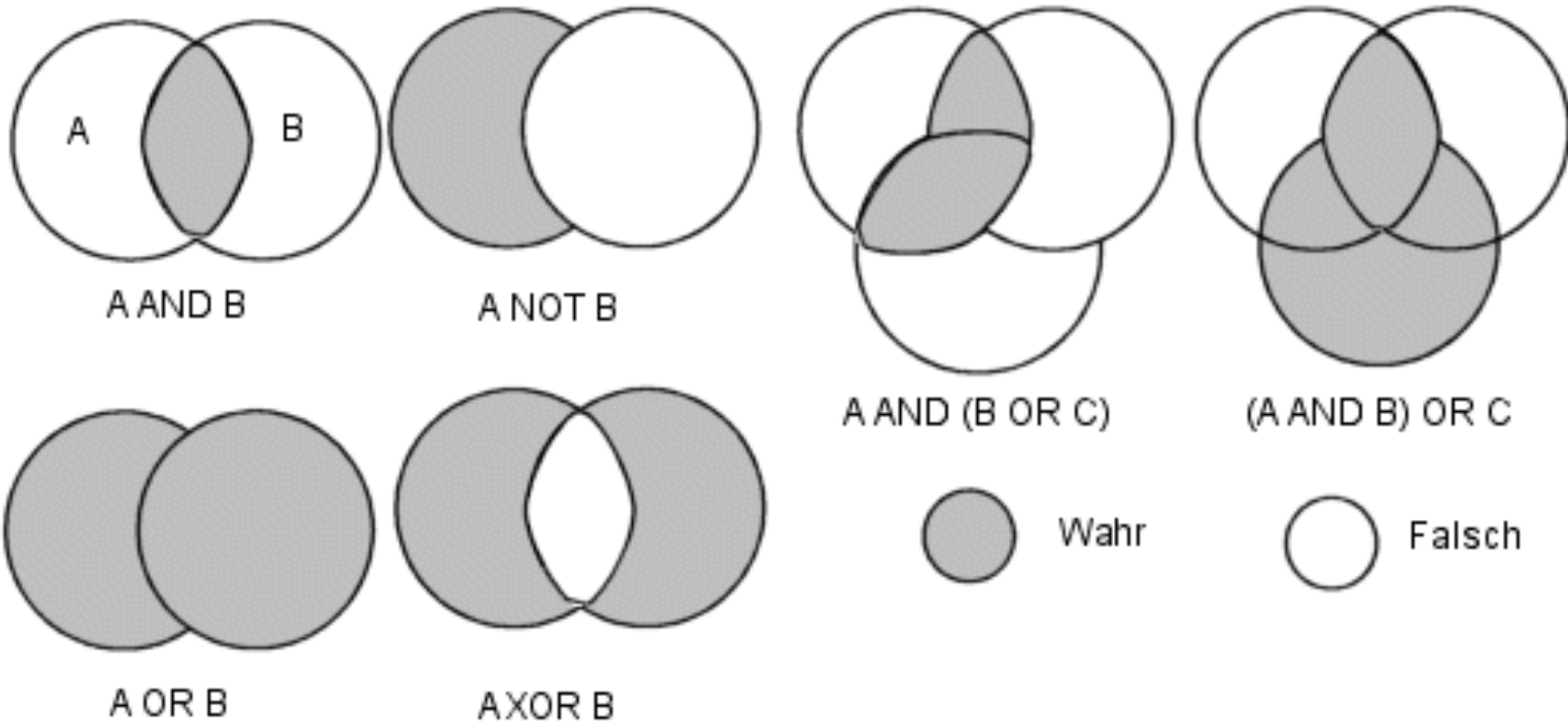
Disjunktion

$\vee$	0	1
0	0	1
1	1	1

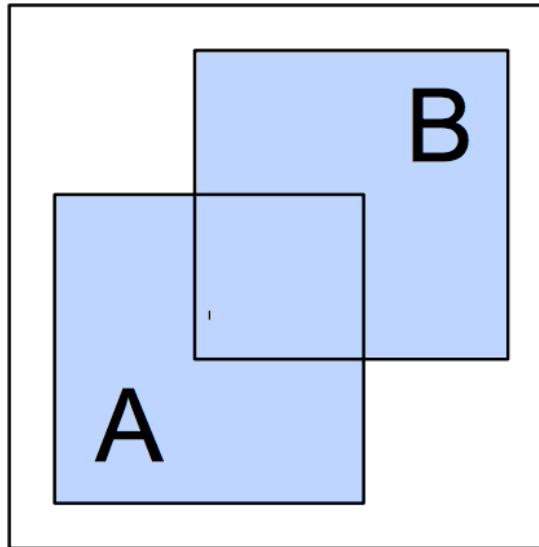
Negation

	$\neg$
0	1
1	0

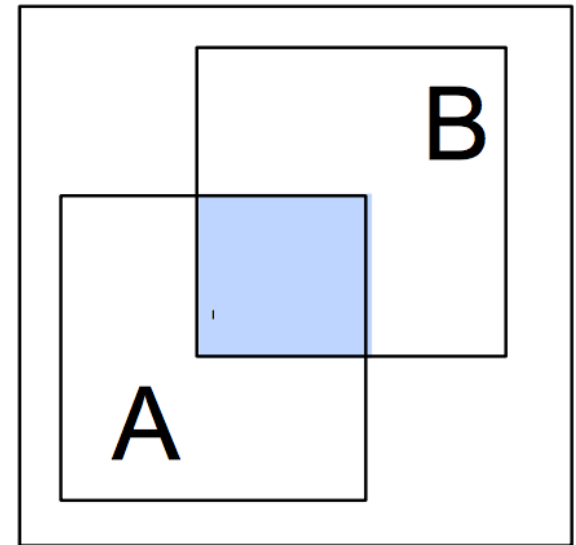
# Beispiele für boolesche Algebra



# Boolesche Anfrage: AND / OR visualisiert als Venn-Diagramme



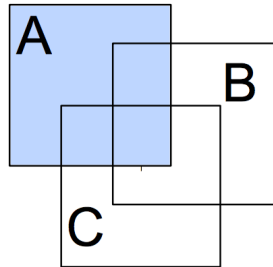
A OR B



A AND B

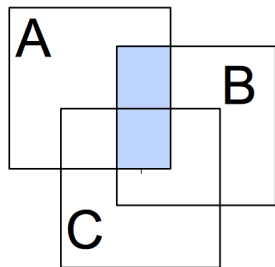
# Kleine Fingerübung I

**A AND (B OR C)**



...

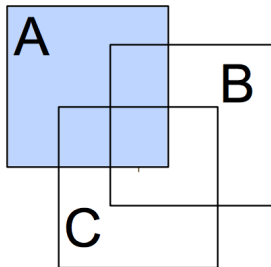
**(A AND B) OR (A AND C)**



...

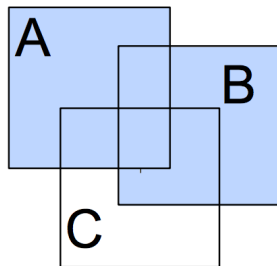
# Kleine Fingerübung II

$A \text{ OR } (B \text{ AND } C)$



...

$(A \text{ OR } B) \text{ AND } (B \text{ OR } C)$



...



# Lösungsweg: Rechnen mit Term-Vektoren

Für jeden Term (Brutus, etc.) gibt es einen **0/1-Vektor** (Zeile):

- 1: der Term kommt in dem Theaterstück vor
- 0: der Term kommt nicht in dem Theaterstück vor.

Um die Frage zu beantworten: Nehme die drei Vektoren für Brutus, Caesar und Calphurnia:

<b>Brutus</b>	1	1	0	1	0	0
<b>Caesar</b>	1	1	0	1	1	1
<b>Calphurnia</b>	0	1	0	0	0	0

# Lösungsweg: Rechnen mit Term-Vektoren

Den Umstand, dass wir Calphurnia NICHT im Ergebnis haben wollen, müssen wir mit einem Trick umsetzen:

- **Calphurnia** (010000  $\rightarrow$  101111, **invertiert**)

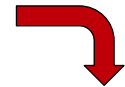
<b><math>\neg</math> Calphurnia</b>	1	0	1	1	1	1
-------------------------------------	---	---	---	---	---	---

Als letzten Schritt verknüpfen wir alle Vektoren über ein *bitweisem UND* ( $\wedge$ ).

<b>Brutus</b>	1	1	0	1	0	0
<b>Caesar</b>	1	1	0	1	1	1
<b><math>\neg</math> Calphurnia</b>	1	0	1	1	1	1
<b><math>\wedge</math></b>	<b>1</b>	<b>0</b>	<b>0</b>	<b>1</b>	<b>0</b>	<b>0</b>

# Was sagt uns dieser Vektor?

1	0	0	1	0	0
---	---	---	---	---	---



	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calphurnia (invertiert)	1	0	1	1	1	1
Caesar	1	0	0	1	0	0

# Die Antwort auf die Anfrage

## Antony and Cleopatra, Akt III, Szene ii

- Textstelle:

Agrippa [Aside to DOMITIUS ENOBARBUS]: Why, Enobarbus,  
When Antony found Julius **Caesar** dead,  
He cried almost to roaring; and he wept  
When at Philippi he found **Brutus** slain.

## Hamlet, Akt III, Szene ii

- Textstelle

Lord Polonius: I did enact Julius **Caesar**  
I was killed i' the Capitol; **Brutus** killed me.



# Die Matrix der Term-Vorkommen

Die Matrix der Term-Dokument-Vorkommen ist nur eine vorläufige Lösung, da

- die Matrix **sehr schnell sehr groß** wird (Anzahl Dokumente \* Anzahl der Terme → zu groß bei großen Dokumentensammlungen)
- die Matrix **dünnbesetzt** ist (sparse, fast nur Nullen, wenige Einsen)

Eine Lösung für das Problem ist der sogenannte **Index**, bei dem nur die Einsen gespeichert werden (nächste Vorlesung).



# Vorteile

Kerneigenschaft: **Präzise Anfragen** sind möglich – Dokumente passen zur Anfrage oder nicht!

- Daher **gut für Experten** geeignet, die
  - das zugrundeliegende Modell verstehen und anwenden können,
  - die verwendete Dokumentenbasis (den Korpus) kennen und
  - die wissen, was sie wollen!
- **Gut für (Computer-)Systeme**, die einfach Tausende von Ergebnissen verarbeiten können.

# Nachteile

## Nicht für die Mehrheit der Nutzer geeignet!

- Viele Nutzer sind nicht in der Lage mit Booleschen Anfragen zu arbeiten
  - viele syntaktische Fehler,
  - verstehen das Modell nicht,
  - können ihr Informationsbedürfnis nicht in Anfragesprache übersetzen...

**Das fehlende Ranking** der Ergebnisse ist für normale Anwender nicht praxistauglich, da sie nicht Hunderte von Ergebnissen auswerten möchten.

- Dies gilt besonders im Bereich der **Web-Suche**.

# AND, OR, NOT ... Was noch?

Üblicherweise werden noch mehr Suchoperatoren unterstützt als nur AND, OR bzw. NOT

- Diese Operatoren alleine wären zu restriktiv
- Precision und Recall sind schwer auszubalancieren

Weitere (gebräuchliche) Operatoren

- **Proximity-Operatoren:** Setzt voraus, dass Suchterme nah beieinander auftreten (z.B. nur 5 Worte voneinander entfernt).
- **Feld-Restriktionen:** Setzt voraus, dass Suchterme nur in bestimmten Teilen des Dokumentes vorkommen dürfen, z.B. im Titel
- **Wildcard-Operatoren:** Suchterme dürfen z.T. von dem Dokumenttermen abweichen (z.B. h?nne → hunne, henne, etc.)

# Feast or Famine

Boolesche Anfragen liefern oft **zu wenige** (=0) oder **zu viele** Ergebnisse (1.000+).

- Anfrage 1: „fräuleinwunder“
  - 200.000 Treffer → Feast
- Anfrage 2: „fräuleinwunder mischungsverhältnis sozialkritik streusalz zyxel“
  - 0 Treffer → Famine

Beim Booleschen Retrievalmodell benötigt es eine Menge Kenntnis und Übung eine Anfrage zu formulieren, die eine **überschaubare Anzahl** an Ergebnissen hervorbringt!

# Feast or Famine → Ranked Retrieval

- Große Ergebnismengen sind mit Ranked Retrieval kein Problem mehr.
- Ranked Retrieval erlaubt es **z.B. nur die Top 10-Ergebnisse** zu betrachten und so den Nutzer zu entlasten.
- Voraussetzung ist ein **Ranking-Algorithmus**, der relevantere Ergebnisse vor weniger relevantere Ergebnisse sortiert.
- Wir benötigen also eine **Funktion**, die eine ungeordnete Ergebnis-**Menge D** in eine geordnete Ergebnis-**Liste L** überführt:

$$f: D \rightarrow L$$



# Erweitertes Boolesches Modell

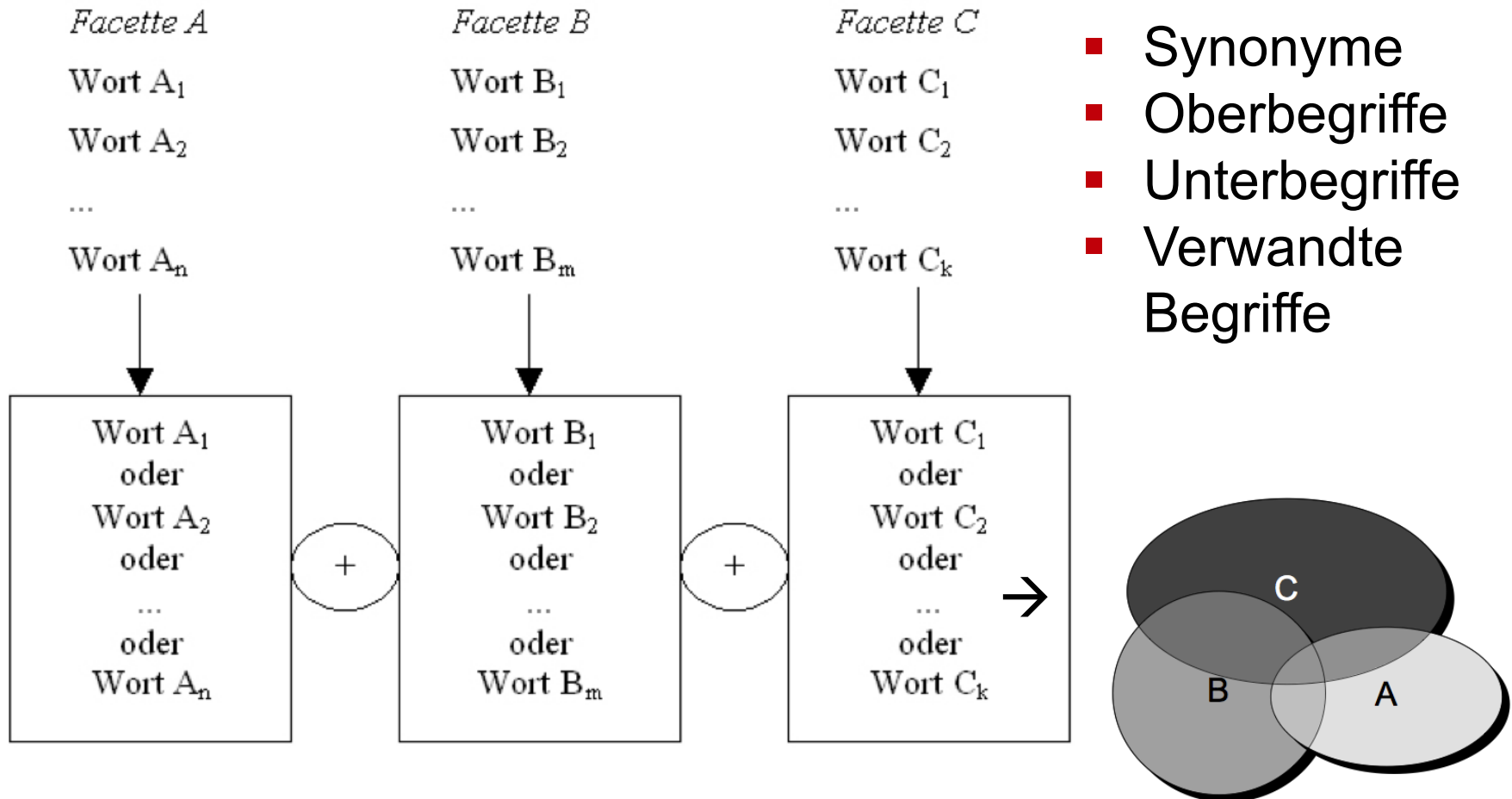
In der Praxis wird die Funktion  $f: D \rightarrow L$  z.B. durch einfache Sortierungen umgesetzt:

- **Chronologische** Sortierung (neuste Ergebnisse zuerst),
- **Alphabetische** Sortierung (z.B. der Autorennamen).

Es sind aber auch **Gewichtungen** möglich z.B. Häufigkeit der Anfrageterme im Dokument: Häufiges Auftreten von Anfragetermen im Dokument ist ein Zeichen für Relevanz.

Allerdings handelt es sich hierbei um **Sortierungen**, kein **Ranking** (dass eine Bewertung der Relevanz voraussetzt).

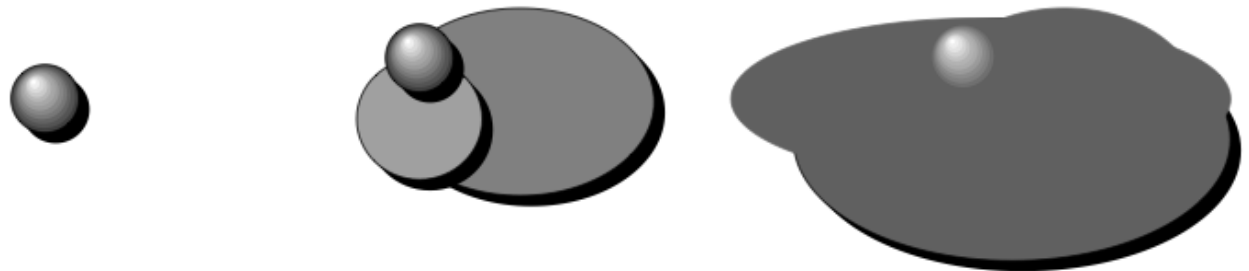
# Anwendung I: Blockstrategie



# Anwendung II: Perlenstrategie

Ziel: Bestes Dokument für das Informationsbedürfnis finden

- Zunächst kleine Treffermenge
  - Mit maximal diskriminierenden Termen suchen,
  - ggf. ein als relevant bekanntes Dokument als Grundlange verwenden.
  - Anschließend: Dokument(e) terminologisch oder zitatenanalytisch ausschachten
- Ggf. gefundene neue Terme in die Blockstrategie übernehmen



# Zusammenfassung Boolesches Retrieval

Suche mit einfachen booleschen/binären Entscheidungen (vorhanden / nicht vorhanden).

## Vorteile:

- Simple Anfragen sind leicht zu verstehen
- Relativ leicht zu implementieren (**Term-Dokument-Matrix**)

## Nachteile:

- Schwierig, genaue Anfragen zu spezifizieren
- Zu viel / zu wenig (**Feast or Famine**)
- Sortierung, aber nicht Ranking

Meistgenutzte IR-Modell bis zum Durchbruch des Web.