

# Das Relationale Modell

- 1.1 Datenmodell
- 1.2 Anfragen
- 1.3 Datenkomplexität und kombinierte Komplexität

# Das Relationale Modell

## 1.1 Datenmodell

## 1.2 Anfragen

## 1.3 Datenkomplexität und kombinierte Komplexität

# Datenmodell

## Begriff "Datenmodell":

- ▶ Rahmen zur Repräsentation / Speicherung von Daten
- ▶ Operationen zum Zugriff auf Daten
- ▶ Mechanismen zur Beschreibung von erwünschten Eigenschaften (Integritätsbedingungen)

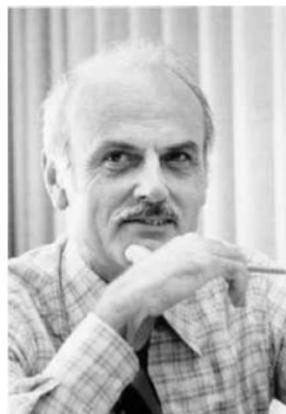
Der Begriff "Datenmodell" ist nicht präzise definiert (im mathematischen Sinn). Im Folgenden wird eine präzise Definition des "Relationalen Modells" gegeben.

# Das Relationale Modell

- ▶ Daten werden in Relationen (“Tabellen”) organisiert
- ▶ Mengen-orientierte Operationen
- ▶ Deklarative Anfragespezifikation
- ▶ Effiziente Anfragebearbeitung
- ▶ 1970 eingeführt von Edgar F. Codd
- ▶ seit Ende der 80er Jahre “Industriestandard”

## Beispiel-Relation:

A	B	C	D
a	1	z	9
b	2	z	8
c	1	y	8
d	4	x	7



Edgar F. Codd (1923-2003)

# Grundbegriff: Relationsschema

## Beispiel-Relation:

A	B	C	D
a	1	z	9
b	2	z	8
c	1	y	8
d	4	x	7

ist vom Schema  $R$  mit  
 $sort(R) = \{A, B, C, D\}$ ,  
 $arity(R) = 4$

Wir legen ein für alle Mal fest:

- ▶ Eine abzählbar unendliche Menge **att** von **Attribut-Namen**. Diese Menge sei **geordnet** via  $\leq_{att}$ .
- ▶ Eine abzählbar unendliche Menge **dom** von "potentiellen Datenbankeinträgen" ("**Konstanten**") (**dom** steht für "Domain")
- ▶ Eine abzählbar unendliche Menge **relname** von **Relations-Namen**. Die Mengen **att**, **dom**, **relname** seien disjunkt.

- ▶ Eine Funktion  $sort: relname \rightarrow \mathcal{P}^{fin}(att)$ , die jedem Relations-Namen eine endliche Menge von Attribut-Namen zuordnet ... und zwar so, dass f.a.  $U \in \mathcal{P}^{fin}(att)$  gilt:  $sort^{-1}(U)$  ist unendlich.

**Notation:**  $\mathcal{P}^{fin}(att)$  ist die Menge aller *endlichen* Teilmengen von **att**.

- ▶ Die **Stelligkeit** eines Relations-Namens  $R$  ist  $arity(R) := |sort(R)|$ .
- ▶ Ein **Relationsschema** ist einfach ein Relations-Name  $R$ .
- ▶ Manchmal schreiben kurz  $R[U]$  für  $sort(R) = U$  und  $R[k]$  für  $arity(R) = k$ .

# Relationenschema vs. Relation

## Beispiel-Relation:

A	B	C	D
a	1	z	9
b	2	z	8
c	1	y	8
d	4	x	7

ist vom Schema  $R$  mit  
 $sort(R) = \{A, B, C, D\}$ ,  
 $arity(R) = 4$

Relation  $\hat{=}$  Tabelle

Tupel  $\hat{=}$  Zeile in der Tabelle

Schreibweise:  $t.A$  an Stelle von  $t(A)$   
 für "Eintrag in Zeile  $t$  und Spalte  $A$ "

Beachte: **Mengensemantik**, d.h.:  
 Relation  $\hat{=}$  die Menge aller Tabellenzeilen

## Definition 1.1

Sei  $R$  ein Relationenschema.

- ▶ Ein  **$R$ -Tupel** ist eine Abbildung  $t : sort(R) \rightarrow dom$ .
- ▶ Eine  **$R$ -Relation** ist eine endliche Menge von  $R$ -Tupeln.
- ▶  **$inst(R)$**  bezeichnet die Menge aller Relationen über  $R$ .  
 ( $inst$  steht für "instances")

# Grundbegriffe: Datenbankschema und Datenbank

## Definition 1.2

- ▶ Ein **Datenbankschema  $\mathbf{R}$**  ist eine endliche, nicht-leere Menge von Relationsschemata.

Manchmal schreiben wir  $\mathbf{R} = \{R_1[U_1], \dots, R_n[U_n]\}$  um die Relationsschemata anzugeben, die zu  $\mathbf{R}$  gehören.

- ▶ Eine **Datenbank  $\mathbf{I}$**  vom Schema  $\mathbf{R}$  ist eine Funktion, die jedem Relationsschema  $R_j \in \mathbf{R}$  eine  $R_j$ -Relation zuordnet.
- ▶  $inst(\mathbf{R})$  bezeichnet die Menge aller Datenbanken vom Schema  $\mathbf{R}$ .
- ▶  $schema(\mathbf{I}) := \mathbf{R}$  bezeichnet das Schema der Datenbank  $\mathbf{I}$ .

## Beispieldatenbank mit Kinodaten (1/5)

Zur Illustration von Anfragen verwenden wir eine kleine Datenbank mit Kinodaten, bestehend aus

- ▶ einer Relation *Orte*, die Informationen über Kinos (Kino, Adresse, Telefonnummer) enthält.
- ▶ einer Relation *Filme*, die Informationen über Filme enthält (Titel, Regie, Schauspieler)
- ▶ einer Relation *Programm*, die Informationen zum aktuellen Kinoprogramm enthält (Kino, Titel, Zeit)

# Beispieldatenbank mit Kinodaten (2/5)

Orte-Relation:

Kino	Adresse	Telefon
Babylon	Dresdner Str. 2	61609693
Casablanca	Friedenstr. 12	6775752
Cinestar Cubix Alexanderplatz	Rathausstr. 1	2576110
Die Kurbel	Giesebrechtstr. 4	88915998
Filmpalast Berlin	Kurfürstendamm 225	8838551
International	Karl-Marx-Allee 33	24756011
Kino in der Kulturbrauerei	Schönhauser Allee 36	44354422
Moviemento	Kottbusser Damm 22	6924785
⋮	⋮	⋮

# Beispieldatenbank mit Kinodaten (3/5)

## Filme-Relation:

Titel	Regie	Schauspieler
Capote	Bennet Miller	Philip Seymour Hoffman
Capote	Bennet Miller	Catherine Keener
Das Leben der Anderen	F. Henkel von Donnersmarck	Martina Gedeck
Das Leben der Anderen	F. Henkel von Donnersmarck	Ulrich Tukur
Der ewige Gärtner	Fernando Meirelles	Ralph Fiennes
Der ewige Gärtner	Fernando Meirelles	Rachel Weisz
Good Night and Good Luck	George Clooney	David Strathairn
Good Night and Good Luck	George Clooney	Patricia Clarkson
Knallhart	Detlev Buck	Jenny Elvers
Knallhart	Detlev Buck	Jan Henrik Stahlberg
Raupatrouille Orion – Rücksturz ins Kino	Michael Braun	Dietmar Schönherr
Raupatrouille Orion – Rücksturz ins Kino	Theo Mezger	Dietmar Schönherr
Raupatrouille Orion – Rücksturz ins Kino	Michael Braun	Eva Pflug
Raupatrouille Orion – Rücksturz ins Kino	Theo Mezger	Eva Pflug
Raupatrouille Orion – Rücksturz ins Kino	Michael Braun	Wolfgang Völz
Raupatrouille Orion – Rücksturz ins Kino	Theo Mezger	Wolfgang Völz
Requiem	Hans-Christian Schmid	Sandra Hüller
Sommer vorm Balkon	Andreas Dresen	Nadja Uhl
Sommer vorm Balkon	Andreas Dresen	Inka Friedrich
Sommer vorm Balkon	Andreas Dresen	Andreas Schmidt
Syriana	Stephen Gaghan	George Clooney
Syriana	Stephen Gaghan	Matt Damon
V wie Vendetta	James McTeigue	Natalie Portman
Walk the Line	James Mangold	Joaquin Phoenix
Walk the Line	James Mangold	Reese Witherspoon

# Beispieldatenbank mit Kinodaten (4/5)

## Programm-Relation:

Kino	Titel	Zeit
Babylon	Capote	17:00
Babylon	Capote	19:30
Kino in der Kulturbrauerei	Capote	17:30
Kino in der Kulturbrauerei	Capote	20:15
International	Das Leben der Anderen	14:30
International	Das Leben der Anderen	17:30
International	Das Leben der Anderen	20:30
Filmpalast Berlin	Good Night and Good Luck	15:30
Filmpalast Berlin	Good Night and Good Luck	17:45
Filmpalast Berlin	Good Night and Good Luck	20:00
Kino in der Kulturbrauerei	Good Night and Good Luck	18:00
Kino in der Kulturbrauerei	Good Night and Good Luck	20:30
Kino in der Kulturbrauerei	Good Night and Good Luck	22:45
Babylon	Sommer vorm Balkon	21:45
Kino in der Kulturbrauerei	Sommer vorm Balkon	21:45
Filmmuseum Potsdam	Raumpatrouille Orion – Rücksturz ins Kino	22:00
⋮	⋮	⋮

# Beispieldatenbank mit Kinodaten (5/5)

## Datenbankschema der Kinodatenbank:

- ▶ Datenbankschema **KINO** = { *Filme*, *Orte*, *Programm* }
- ▶  $sort(Filme) = \{ \text{Titel, Regie, Schauspieler} \}$
- ▶  $sort(Orte) = \{ \text{Kino, Adresse, Telefon} \}$
- ▶  $sort(Programm) = \{ \text{Kino, Titel, Zeit} \}$

Wir schreiben **I<sub>KINO</sub>**, um unsere konkrete Datenbank vom Schema **KINO** zu bezeichnen. Analog schreiben wir *I<sub>Filme</sub>*, *I<sub>Orte</sub>* und *I<sub>Programm</sub>* für die konkreten Relationen, die zur Datenbank **I<sub>KINO</sub>** gehören.

# Attribute: Benannte vs. Unbenannte Perspektive

*Sind die Attribut-Namen Teil des expliziten Datenbankschemas?*

In SQL: ja! Beispiel:

```
SELECT Titel FROM Filme WHERE Schauspieler="Natalie Portman"
```

*Aber werden die Namen vom System nicht "weg-compiliert"?*

## ► Benannte Perspektive:

Ein Tupel über Relationsschema  $R[U]$  ist eine Abbildung von  $U$  nach **dom**.

Schreibweise:  $t = \langle A : a, B : 1, C : z, D : 9 \rangle$

## ► Unbenannte Perspektive:

Ein Tupel über Relationsschema  $R[k]$  ist ein Element aus **dom** <sup>$k$</sup>  (Cartesisches Produkt aus  $k$  Kopien von **dom**).

Schreibweise:  $t = \langle a, 1, z, 9 \rangle$

## Beispiel-Relation:

A	B	C	D
a	1	z	9
b	2	z	8
c	1	y	8
d	4	x	7

ist vom Schema  $R$  mit  
 $sort(R) = \{A, B, C, D\}$ ,  
 $arity(R) = 4$

# Notation

Konstanten (Elemente aus <b>dom</b> ) .....	$a, b, c$ , "George Clooney", ...
Attribute .....	$A, B, C$ , ...
Mengen von Attributen .....	$U, V$ , ...
Relations-Namen (-schemata) .....	$R, S, R[U], S[V]$ , ...
Datenbankschemata .....	<b>R, S</b>
Tupel .....	$t, s$
Relationen (Relations-Instanzen) .....	$I, J$
Datenbanken (Datenbank-Instanzen) ...	<b>I, J</b>

# Das Relationale Modell

1.1 Datenmodell

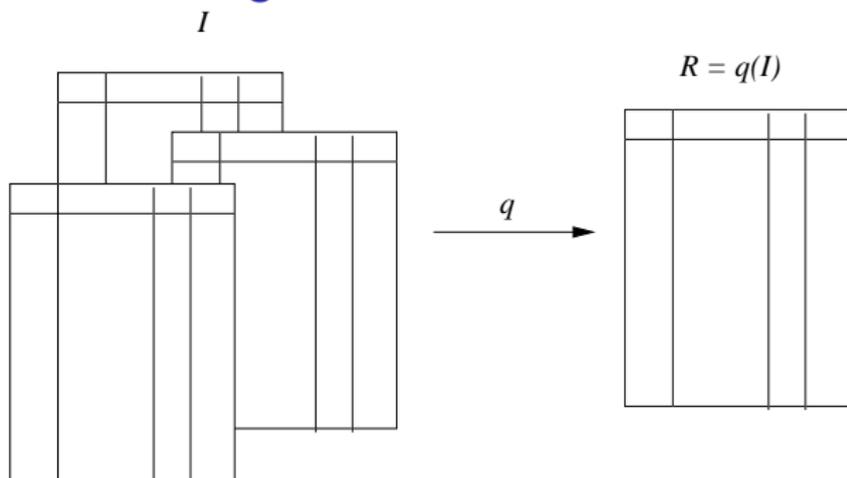
**1.2 Anfragen**

1.3 Datenkomplexität und kombinierte Komplexität

## Beispiel-Anfragen

- (1) Wer führt Regie in "Knallhart"?
- (2) Wo läuft "Capote"?
- (3) Welches sind Adresse und Telefonnummer des "Kino in der Kulturbrauerei"?
- (4) Welche Kinos spielen einen Film mit "George Clooney"?
- (5) Läuft zur Zeit ein "Detlev Buck" Film?
- (6) Welche (je 2) Schauspieler haben schon in Filmen gespielt, in denen der jeweils andere Regie geführt hat?
- (7) Welche Regisseure haben in einem ihrer eigenen Filme mitgespielt?
- (8) Gib die 2-Tupel von Schauspielern an, die gemeinsam in einem Film gespielt haben!
- (9) Egal für welche Datenbank, gib  $\langle$ "Knallhart", "Detlev Buck"  $\rangle$  als Antwort aus!
- (10) Wo wird "Syriana" oder "Walk the Line" gespielt?
- (11) Welche Filme laufen in mindestens 2 Kinos?
- (12) In welchen Filmen spielt "George Clooney" mit oder führt Regie?
- (13) Gib alle Filme aus, die im Kino "International" laufen oder in denen "Andreas Dresen" Regie führt!
- (14) Liste alle Schauspieler und den Regisseur von "Capote" auf!
- (15) Welche Filme laufen nur zu 1 Uhrzeit?
- (16) In welchen Filmen spielt "George Clooney" mit, ohne Regie zu führen?
- (17) Welche Filme haben nur Schauspieler, die schon mal in einem Film von "Stephen Spielberg" mitgespielt haben?

# Anfragen und Anfragefunktionen



## Definition 1.3

- ▶ Eine **Anfragefunktion** ist eine Abbildung  $q$ , die, für ein Datenbankschema  $\mathbf{R}$  und ein Relationsschema  $R$ , jeder Datenbank  $I$  vom Schema  $\mathbf{R}$ , eine Relation,  $q(I)$  vom Schema  $R$  zuordnet. ( $q$  steht für "query")
- ▶ Eine **Anfrage** ist eine Zeichenkette, die eine Anfragefunktion in einer bestimmten Syntax beschreibt.

# Wünschenswerte Eigenschaften von Anfragefunktionen

## Forderungen an eine Anfragefunktion $q$ :

- (a) Das Ergebnis sollte nicht von Details der Speicherung abhängen, sondern nur von der logischen Sicht auf die Daten

$q$  ist eine Funktion  $inst(\mathbf{R}) \rightarrow inst(R)$ ,  
für ein DB-Schema  $\mathbf{R}$  und ein Rel.schema  $R$

- (b) Sie sollte berechenbar sein.

$q$  berechenbar

- (c) Das Ergebnis sollte möglichst wenig von den einzelnen Datenwerten und möglichst viel von den Beziehungen der Daten untereinander abhängen

... siehe nächste Folie; Stichwort: "generisch"

# Erläuterungen zu Eigenschaft (c)

Beispiel-Anfrage:

(3) Welches sind Adresse und Telefonnummer des “Kino in der Kulturbrauerei”?

## Eigenschaft (c):

- ▶ Wenn sich die Telefonnummer vom “Kino in der Kulturbrauerei” in der DB ändert, soll sich das Ergebnis der Anfrage entsprechend ändern.
- ▶ Aber wenn der Name “Kino in der Kulturbrauerei” sich in der DB ändert, soll das Ergebnis leer sein.
- ▶ Allgemein:
  - ▶ Werden Elemente der Datenmenge **dom**, die in der Anfrage nicht explizit (als “Konstanten”) vorkommen, umbenannt, so sollen sie im Ergebnis auch umbenannt werden.
- ▶ mathematische Präzisierung: Begriff der **generischen Anfragefunktion**

# Generische Anfragefunktionen

## Definition 1.4

Sei  $C$  eine endliche Menge von Datenwerten  
(also  $C \subseteq^{\text{fin}} \mathbf{dom}$ ).

Eine Anfragefunktion  $q$  heißt **C-generisch**, falls für jede Datenbank  $I$  (vom zu  $q$  passenden DB-Schema) und jede Permutation  $\pi$  von  $\mathbf{dom}$  mit  $\pi|_C = \text{id}$  (d.h.  $\pi(x) = x$  für alle  $x \in C$ ) gilt:

$$q(\pi(I)) = \pi(q(I)).$$

$q$  heißt **generisch**, falls  $q$   $\emptyset$ -generisch ist.

## Beispiel:

(3) Welches sind Adresse und Telefonnummer des “Kino in der Kulturbrauerei”?  
ist {"Kino in der Kulturbrauerei"}-generisch.

(7) Welche Regisseure haben in einem ihrer eigenen Filme mitgespielt?  
ist generisch.

## Illustration:

$$\begin{array}{ccc}
 I & \xrightarrow{q} & q(I) \\
 \pi \downarrow & & \pi \downarrow \\
 \pi(I) & \xrightarrow{q} & q(\pi(I))
 \end{array}$$

# Boolesche Anfragen

Manche Anfragen lassen sich nur mit “ja” oder “nein” beantworten.

Beispiel: (5) Lauft zur Zeit ein “Detlev Buck” Film?

## Konvention:

- ▶ Ergebnis ist eine 0-stellige Relation.
- ▶ Davon gibt es genau zwei Stuck:  $\emptyset$  und  $\{\langle \rangle\}$ .  
 $\langle \rangle$  steht fur das “Tupel der Stelligkeit 0”.
- ▶ Vereinbarung:
  - ▶  $\emptyset$  steht fur “nein”
  - ▶  $\{\langle \rangle\}$  steht fur “ja”

# Anfragesprachen

Dieselbe Anfragefunktion kann in verschiedenen Anfragesprachen beschrieben werden.

**Beispiel:** (4) Welche Kinos spielen einen Film mit "George Clooney"?

► **SQL:**

```
SELECT Orte.Kino, Orte.Adresse
FROM Filme, Programm, Orte
WHERE Filme.Schauspieler = "George Clooney" AND
       Filme.Titel = Programm.Titel AND
       Programm.Kino = Orte.Kino
```

► **Regelbasierte Anfrage:**

$$\text{Ans}(x_{Kino}, x_{Adr}) \leftarrow \text{Filme}(x_{Titel}, x_{Regie}, \text{"George Clooney"}), \\ \text{Programm}(x_{Kino}, x_{Titel}, x_{Zeit}), \\ \text{Orte}(x_{Kino}, x_{Adr}, x_{Tel})$$

► **Relationenkalkül:**

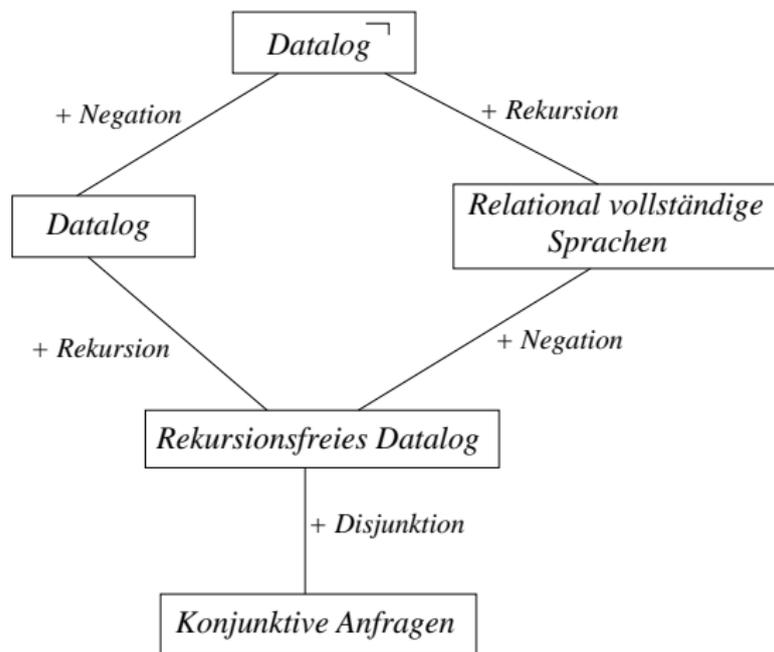
$$\left\{ \langle x_{Kino}, x_{Adr} \rangle : \exists x_{Titel} \exists x_{Regie} \exists x_{Zeit} \exists x_{Tel} \left( \text{Filme}(x_{Titel}, x_{Regie}, \text{"George Clooney"}) \wedge \right. \right. \\ \left. \left. \text{Programm}(x_{Kino}, x_{Titel}, x_{Zeit}) \wedge \text{Orte}(x_{Kino}, x_{Adr}, x_{Tel}) \right) \right\}$$

► **Relationale Algebra:**  $\pi_{Kino, Adresse} \left( \sigma_{\text{Schauspieler} = \text{"George Clooney"}} (\text{Filme} \bowtie \text{Programm} \bowtie \text{Orte}) \right)$

# Hierarchie der Anfragesprachen

**Bemerkung:** Anfragesprachen unterscheiden sich in ihrer Ausdrucksstärke.

Übersicht:



# Das Relationale Modell

1.1 Datenmodell

1.2 Anfragen

1.3 Datenkomplexität und kombinierte Komplexität

# Typische Problemstellungen bzgl. Anfrageauswertung

Sei  $\mathcal{A}$  eine Anfragesprache.

AUSWERTUNGSPROBLEM FÜR  $\mathcal{A}$ :

*Eingabe:* Anfrage  $Q \in \mathcal{A}$ , Datenbank  $I$

*Aufgabe:* Berechne  $Q(I)$

Variante:

NICHT-LEERHEITS-PROBLEM FÜR  $\mathcal{A}$ :

*Eingabe:* Anfrage  $Q \in \mathcal{A}$ , Datenbank  $I$

*Aufgabe:* Ist  $Q(I) \neq \emptyset$  ?

**Wichtige Fragestellung:**

Welche Ressourcen (etwa Zeit, Platz) sind nötig, um diese Probleme zu lösen?

# Datenkomplexität und Kombinierte Komplexität

Die Komplexität der Anfrageauswertung kann unter zwei Blickwinkeln betrachtet werden:

1. Anfrage und Datenbank sind Eingabe ↪ **Kombinierte Komplexität**  
gemessen in  $n$  und  $k$ , wobei  $n = ||I||$  und  $k = ||Q||$
2. Anfrage fest, Datenbank ist Eingabe: ↪ **Datenkomplexität**  
gemessen nur in  $n = ||I||$

Rechtfertigung für "Datenkomplexität":

i.d.R. ist die Anfrage kurz, die Datenbank aber sehr groß.

Typische Form von Ergebnissen, die im Laufe der Vorlesung bewiesen werden:

- ▶ Die Datenkomplexität der Relationalen Algebra ist in **LOGSPACE**.
- ▶ Die kombinierte Komplexität der Relationalen Algebra ist **PSPACE-vollständig**.