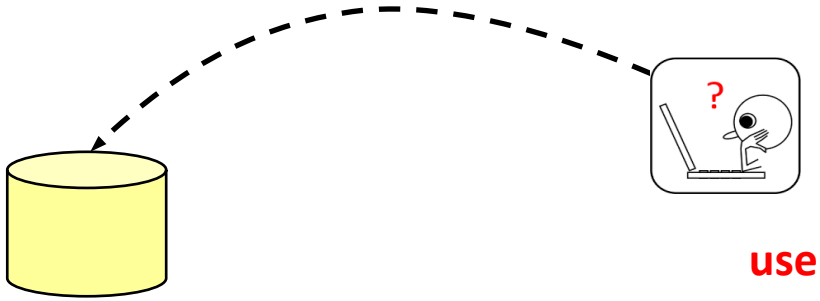
understand how a DBMS efficiently evaluates a query over a database

(the main focus of this course)



user query

database

understand the mathematical foundations of query evaluation

(next two weeks)

A Quick Recap

Flight	***origin***	***destination***	***airline***

Airport	***code***	***city***

A Quick Recap

List all the airlines

Flight	*origin*	*destination*	*airline*
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

Airport	*code*	*city*
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

A Quick Recap

List all the airlines

Flight	*origin*	*destination*	*airline*
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

{BA, U2, OS}

Airport	*code*	*city*
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

$\pi_{airline}$ Flight

A Quick Recap

List the codes of the airports in London

Flight	***origin***	***destination***	***airline***
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

Airport	***code***	***city***
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

A Quick Recap

List the codes of the airports in London

Flight	*origin*	*destination*	*airline*
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

Airport	*code*	*city*
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

{LHR, LGW}

$\pi_{code}\ (\sigma_{city='London'}\ Airport)$

A Quick Recap

List the airlines that fly directly from London to Glasgow

Flight	*origin*	*destination*	*airline*
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

Airport	*code*	*city*
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

$$\pi_{\text{airline}}\ ((\text{Flight}\ \bowtie_{\text{origin=code}}\ (\sigma_{\text{city='London'}}\ \text{Airport}))\ \bowtie_{\text{destination=code}}\ (\sigma_{\text{city='Glasgow'}}\ \text{Airport}))$$

A Quick Recap

$$\pi_{airline}\ ((Flight\ \bowtie_{origin=code}\ \underbrace{(\sigma_{city='London'}\ Airport)}) \bowtie_{destination=code}\ \underbrace{(\sigma_{city='Glasgow'}\ Airport)})$$

code	***city***
LHR	London
LGW	London

code	***city***
GLA	Glasgow

A Quick Recap

$\pi_{airline}$ ((Flight $\bowtie_{origin=code}$ ($\sigma_{city='London'}$ Airport)) $\bowtie_{destination=code}$ ($\sigma_{city='Glasgow'}$ Airport))

code	*city*
LHR	London
LGW	London

code	*city*
GLA	Glasgow

origin	*destination*	*airline*	*code*	*city*
LHR	EDI	BA	LHR	London
LGW	GLA	U2	LGW	London

A Quick Recap

$\pi_{airline}$ ((Flight $\bowtie_{origin=code}$ ($\sigma_{city='London'}$ Airport)) $\bowtie_{destination=code}$ ($\sigma_{city='Glasgow'}$ Airport))

code	*city*
LHR	London
LGW	London

code	*city*
GLA	Glasgow

origin	*destination*	*airline*	*code*	*city*
LHR	EDI	BA	LHR	London
LGW	GLA	U2	LGW	London

origin	*destination*	*airline*	*code*	*city*	*code*	*city*
LGW	GLA	U2	LGW	London	GLA	Glasgow

A Quick Recap

$\pi_{airline}$ ((Flight $\bowtie_{origin=code}$ ($\sigma_{city='London'}$ Airport)) $\bowtie_{destination=code}$ ($\sigma_{city='Glasgow'}$ Airport))

code	*city*
LHR	London
LGW	London

code	*city*
GLA	Glasgow

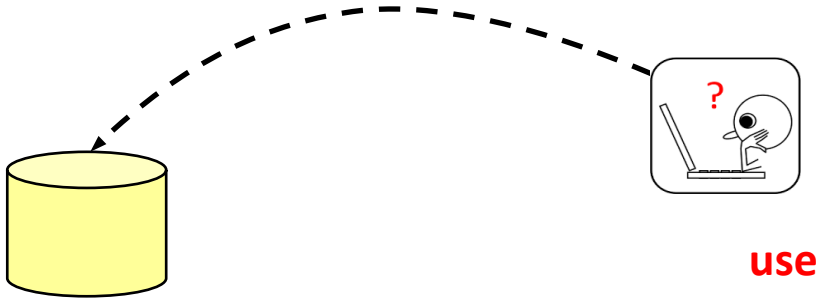
origin	*destination*	*airline*	*code*	*city*
LHR	EDI	BA	LHR	London
LGW	GLA	U2	LGW	London

origin	*destination*	*airline*	*code*	*city*	*code*	*city*
LGW	GLA	U2	LGW	London	GLA	Glasgow

airline
U2

understand how a DBMS efficiently evaluates a query over a database

(the main focus of this course)



user query

database

understand the mathematical foundations of query evaluation

(next two weeks)

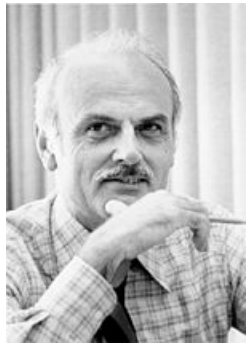
Conjunctive Queries: Syntax and Semantics

(Chapters 12 and 13 of DBT)

[DBT] Database Theory, https://github.com/pdm-book/community

Codd's Theorem

Relational Algebra = Relational Calculus



Edgar F. Codd
(1923 - 2003)
Turing Award 1981

- Queries are written using a declarative language
- DBMS converts declarative queries into procedural queries that are optimized and executed

Relational Calculus

List all the airlines

Flight	*origin*	*destination*	*airline*
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

{BA, U2, OS}

Airport	*code*	*city*
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

$\{z \mid \exists x \exists y\ (Flight(x,y,z))\}$

Relational Calculus

List the codes of the airports in London

Flight	***origin***	***destination***	***airline***
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

Airport	***code***	***city***
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

{LHR, LGW}

{x | Airport(x,London)}

Relational Calculus

List the airlines that fly directly from London to Glasgow

Flight	*origin*	*destination*	*airline*
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

{U2}

Airport	*code*	*city*
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

$$\{z \mid \exists x \exists y\ (\text{Airport}(x,\text{London}) \wedge \text{Airport}(y,\text{Glasgow}) \wedge \text{Flight}(x,y,z))\}$$

A Core Relational Query Language

= **Conjunctive Queries (CQ)**

= {σ,π,⋈}-fragment of relational algebra

= relational calculus without ¬, ∀, ∨, =

= simple SELECT-FROM-WHERE SQL queries

= (only AND and equality in the WHERE clause)

Syntax of Conjunctive Queries

$$Q(\mathbf{x}) := \exists \mathbf{y}\, (R_1(\mathbf{v_1}) \wedge \cdots \wedge R_m(\mathbf{v_m}))$$

- $R_1,\ldots,R_m$ are relation names
- $\mathbf{x}, \mathbf{y}, \mathbf{v_1},\ldots,\mathbf{v_m}$ are tuples of variables
- each variable mentioned in $\mathbf{v_i}$ appears either in $\mathbf{x}$ or $\mathbf{y}$
- the variables in $\mathbf{x}$ are free called distinguished or output variables

It is very convenient to see conjunctive queries as rule-based queries of the form

$$Q(\mathbf{x}) :- \underbrace{R_1(\mathbf{v_1}),\ldots,R_m(\mathbf{v_m})}$$

this is called the body of Q that can be seen as a set of atoms

Conjunctive Queries: Example 1

List all the airlines

Flight	*origin*	*destination*	*airline*
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

{BA, U2, OS}

Airport	*code*	*city*
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

$\pi_{airline}$ Flight

Q(z) :- Flight(x,y,z)

{z | ∃x∃y Flight(x,y,z)}

Conjunctive Queries: Example 2

List the codes of the airports in London

Flight	*origin*	*destination*	*airline*
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

Airport	*code*	*city*
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

{LHR, LGW}

$\pi_{code}\ (\sigma_{city='London'}\ Airport)$

{x | Airport(x,London)}

Q(x) :- Airport(x,London)

Conjunctive Queries: Example 3

List the airlines that fly directly from London to Glasgow

Flight	*origin*	*destination*	*airline*
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

Airport	*code*	*city*
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

{U2}

$\pi_{airline}$ ((Flight $\bowtie_{origin=code}$ ($\sigma_{city='London'}$ Airport)) $\bowtie_{destination=code}$ ($\sigma_{city='Glasgow'}$ Airport))

$\{z \mid \exists x \exists y\ (Airport(x,London) \wedge Airport(y,Glasgow) \wedge Flight(x,y,z))\}$

Conjunctive Queries: Example 3

List the airlines that fly directly from London to Glasgow

Flight	***origin***	***destination***	***airline***
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

{U2}

Airport	***code***	***city***
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

Q(z) :- Airport(x,London), Airport(y,Glasgow), Flight(x,y,z)

Pattern Matching Problem

List the airlines that fly directly from London to Glasgow

Flight(VIE,LHR,BA),
Flight(LHR,EDI,BA),
Flight(LGW,GLA,U2),
Flight(LCA,VIE,OS),

Airport(VIE,Vienna),
Airport(LHR,London),
Airport(LGW,London),
Airport(LCA,Larnaca),
Airport(GLA,Glasgow),
Airport(EDI,Edinburgh)

Q(z) :- Airport(x,London), Airport(y,Glasgow), Flight(x,y,z)

Pattern Matching Problem

List the airlines that fly directly from London to Glasgow

	Airport(VIE,Vienna),
Flight(VIE,LHR,BA),	Airport(LHR,London),
Flight(LHR,EDI,BA),	**Airport(LGW,London)**,
Flight(LGW,GLA,U2),	Airport(LCA,Larnaca),
Flight(LCA,VIE,OS),	**Airport(GLA,Glasgow)**,
	Airport(EDI,Edinburgh)

Q(z) :- Airport(x,London), Airport(y,Glasgow), Flight(x,y,z)

Homomorphism

- Pattern matching - properly formalized via the key notion of homomorphism

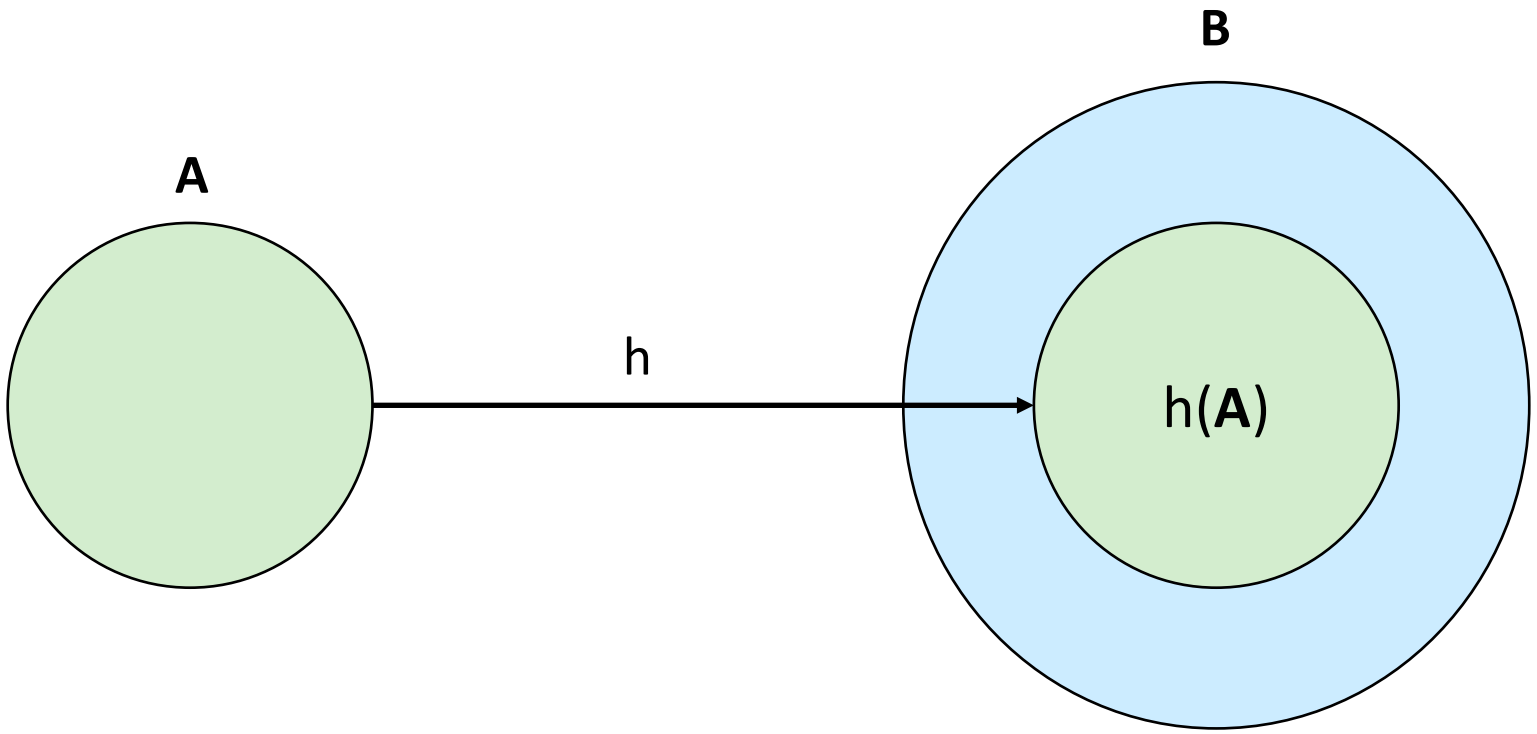
- A substitution from a set of terms **S** to a set of terms **T** is a function h : **S** → **T**, i.e., h is a set of mappings of the form s ↦ t, where s ∈ **S** and t ∈ **T**

- A homomorphism from a set of atoms **A** to a set of atoms **B** is a substitution h : terms(**A**) → terms(**B**) such that:

    1. t is a constant value ⇒ h(t) = t
    2. $R(t_1,\ldots,t_k) \in \mathbf{A} \Rightarrow h(R(t_1,\ldots,t_k)) = R(h(t_1),\ldots,h(t_k)) \in \mathbf{B}$

(terms(**A**) = {t | t is a variable or a constant value that occurs in **A**})

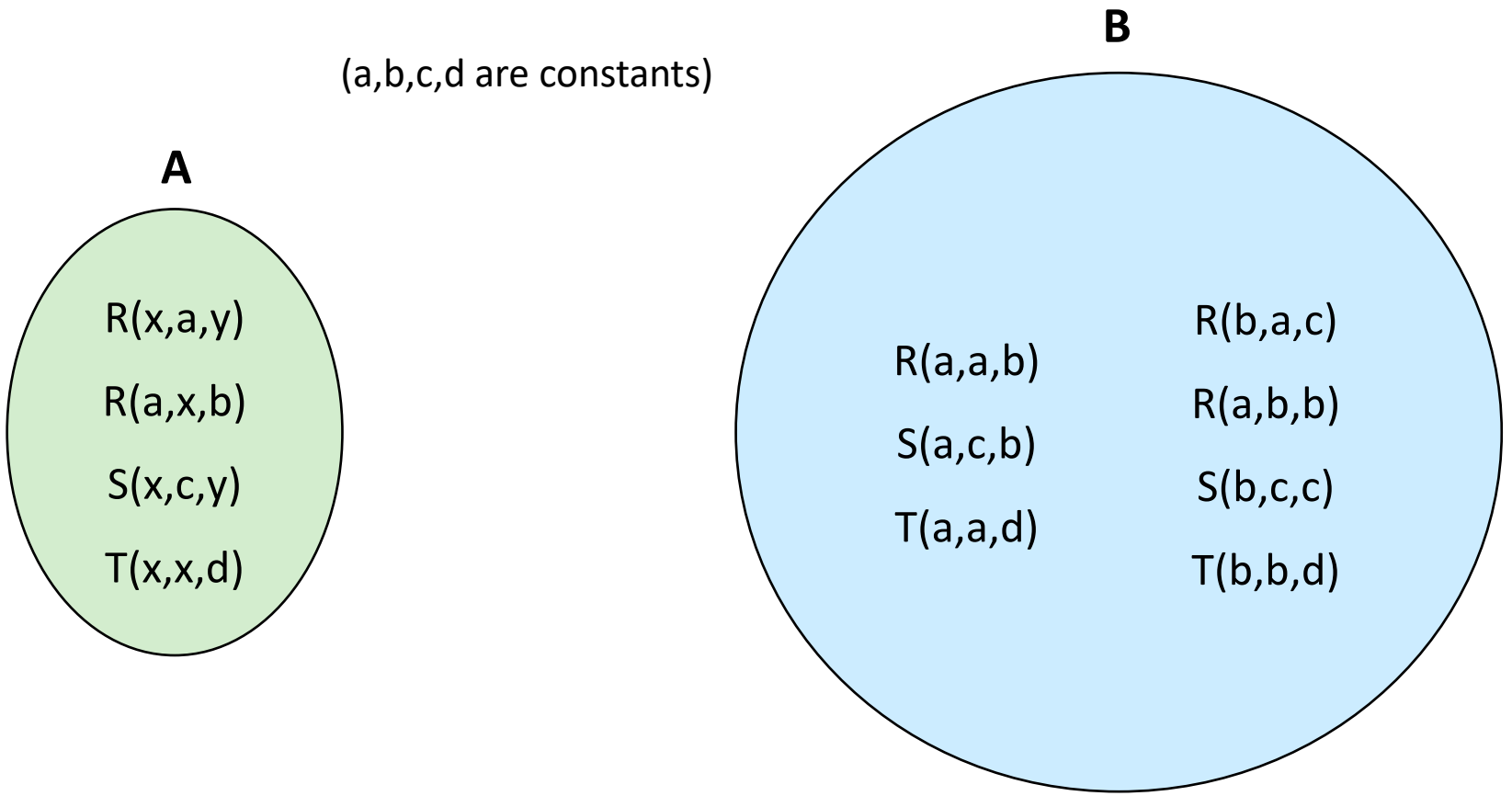
Homomorphism



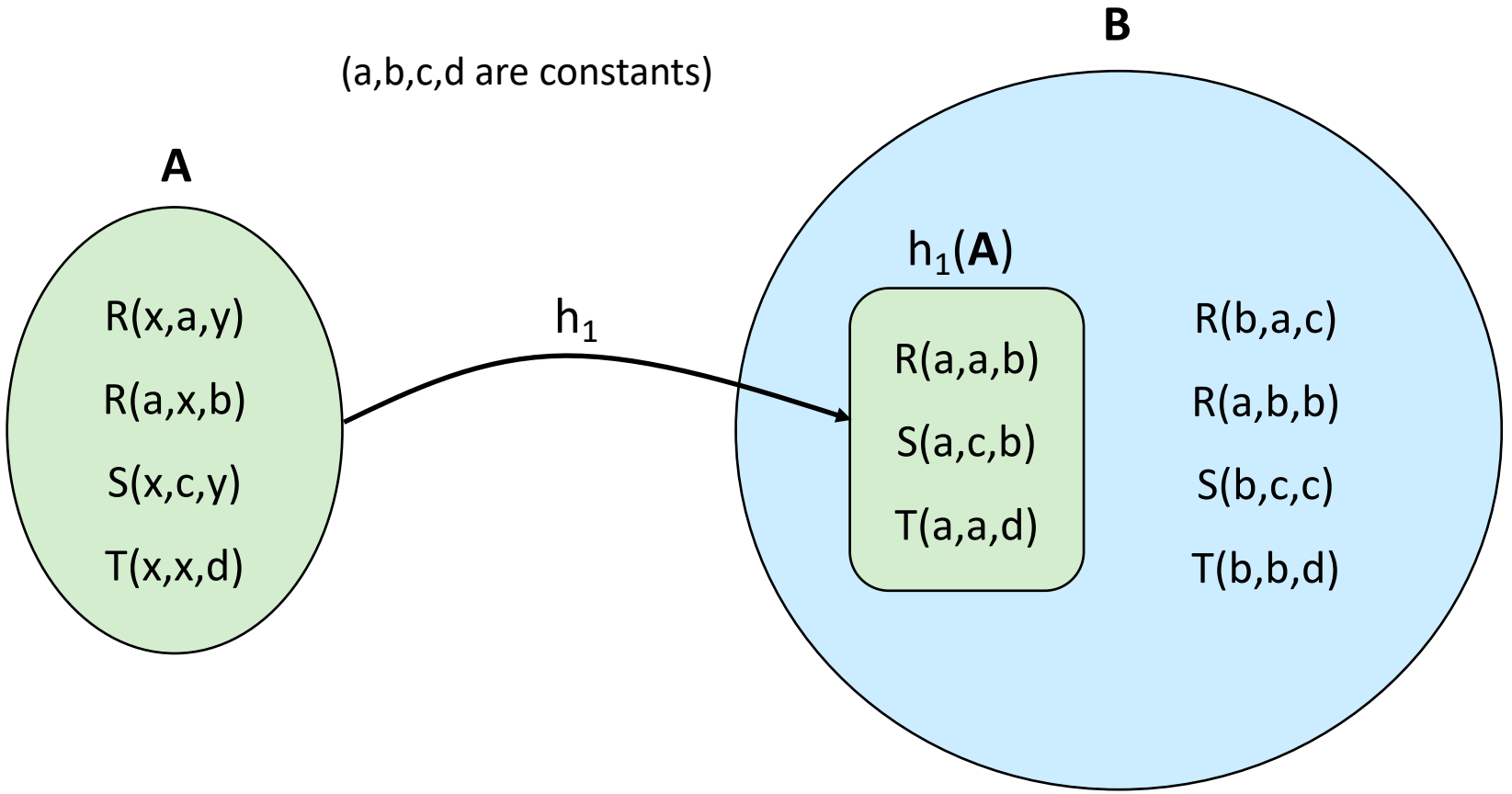
h : terms(**A**) → terms(**B**) that is the identity on constants

Homomorphism

(a,b,c,d are constants)

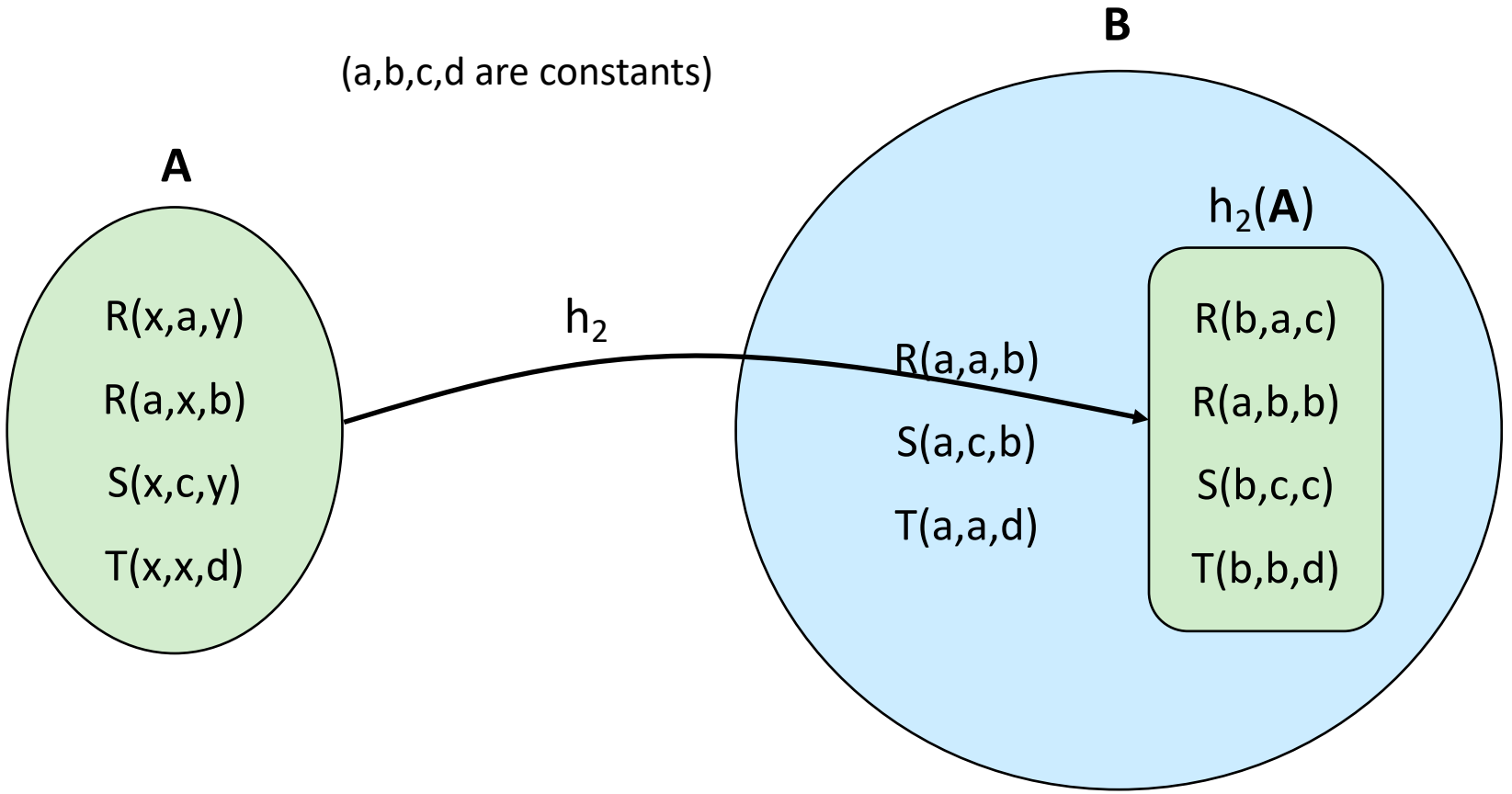


Homomorphism



$h_1 = \{a \mapsto a, b \mapsto b, c \mapsto c, d \mapsto d, x \mapsto a, y \mapsto b\}$

Homomorphism



$h_2 = \{a \mapsto a, b \mapsto b, c \mapsto c, d \mapsto d, x \mapsto b, y \mapsto c\}$

Find the Homomorphisms

$$\mathbf{S_1} = \{P(x_1,y_1), P(y_1,z_1), P(z_1,w_1)\}$$

$\mathbf{S_2} = \{P(x_2,y_2), P(y_2,z_2), P(z_2,x_2)\}$

$\mathbf{S_3} = \{P(x_3,y_3), P(y_3,x_3)\}$

$$\mathbf{S_4} = \{P(x_4,y_4), P(y_4,x_4), P(y_4,y_4)\}$$

$$\mathbf{S_5} = \{P(x_5,x_5)\}$$

Find the Homomorphisms

$\mathbf{S_1} = \{P(x_1,y_1), P(y_1,z_1), P(z_1,w_1)\}$

$\{x_1 \mapsto x_2, y_1 \mapsto y_2, z_1 \mapsto z_2, w_1 \mapsto x_2\}$

$\{x_1 \mapsto x_3, y_1 \mapsto y_3, z_1 \mapsto x_3, w_1 \mapsto y_3\}$

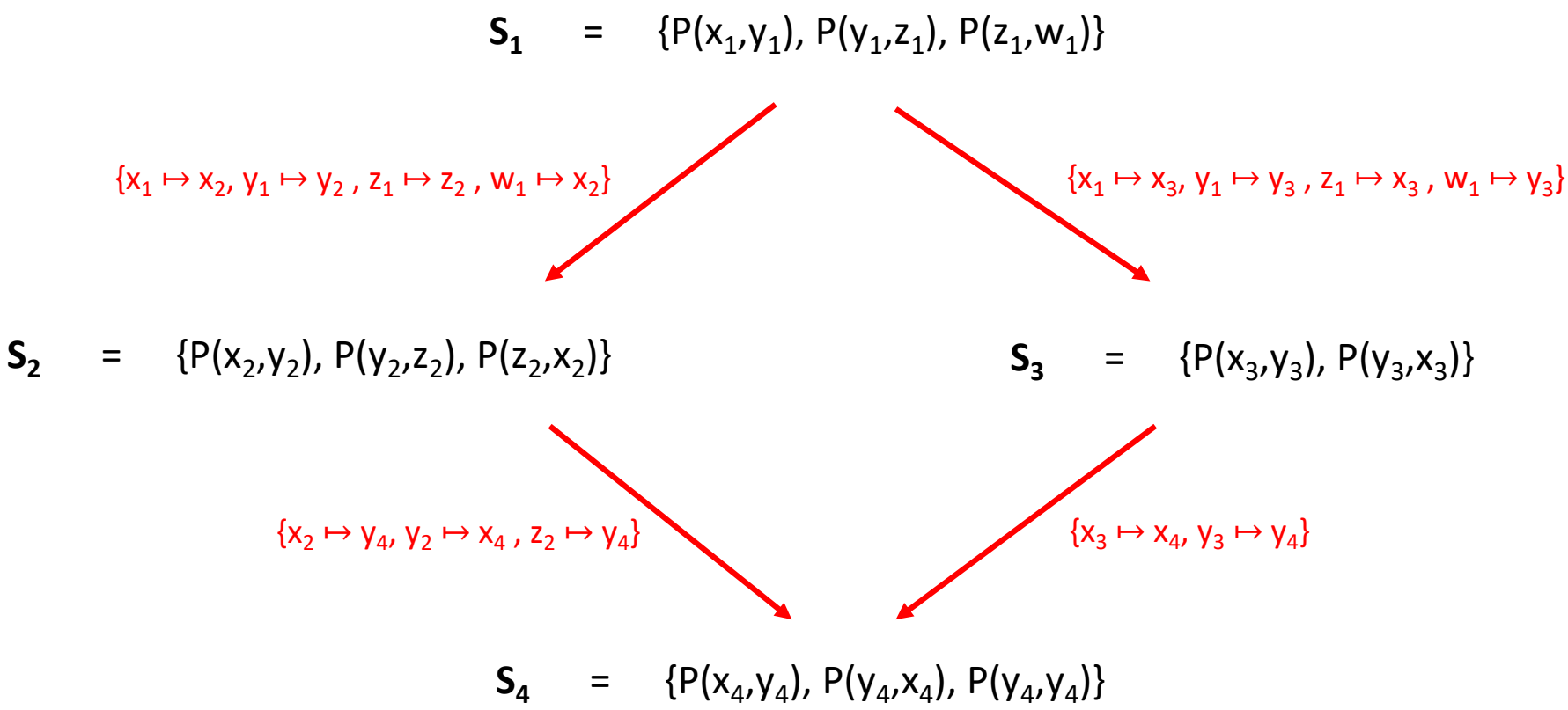
$\mathbf{S_2} = \{P(x_2,y_2), P(y_2,z_2), P(z_2,x_2)\}$

$\mathbf{S_3} = \{P(x_3,y_3), P(y_3,x_3)\}$

$\mathbf{S_4} = \{P(x_4,y_4), P(y_4,x_4), P(y_4,y_4)\}$

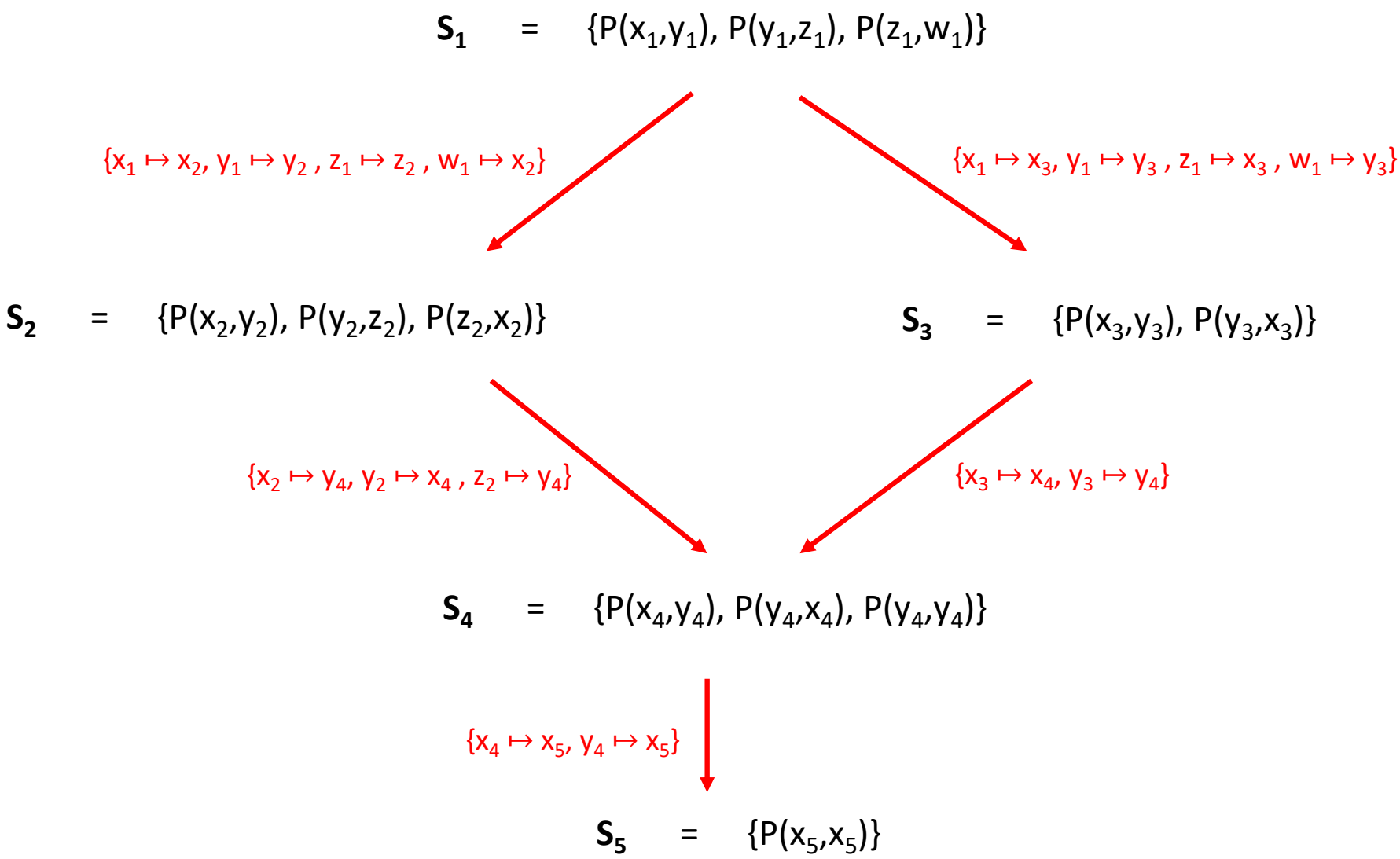
$\mathbf{S_5} = \{P(x_5,x_5)\}$

Find the Homomorphisms

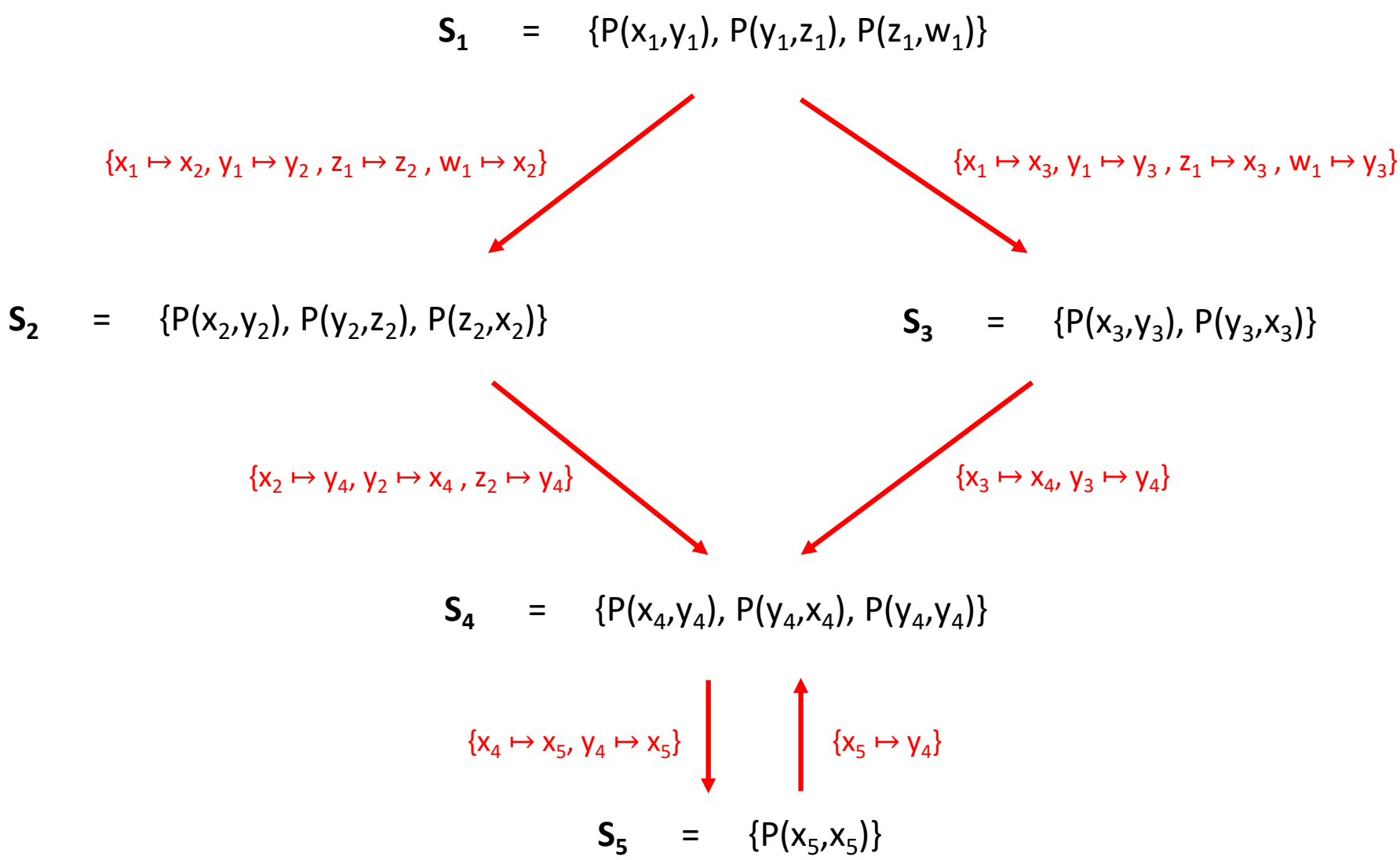


$S_5 = \{P(x_5,x_5)\}$

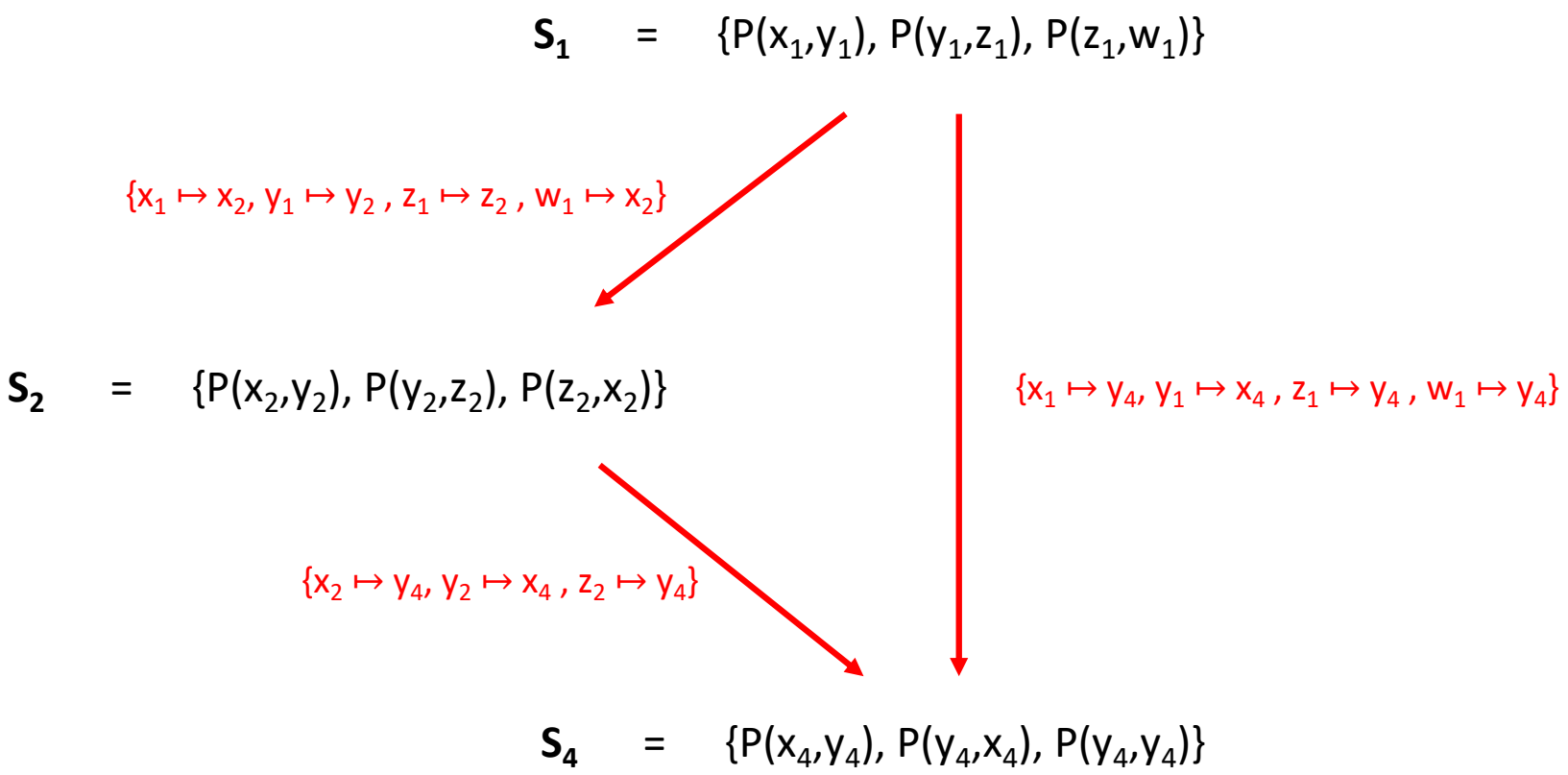
Find the Homomorphisms



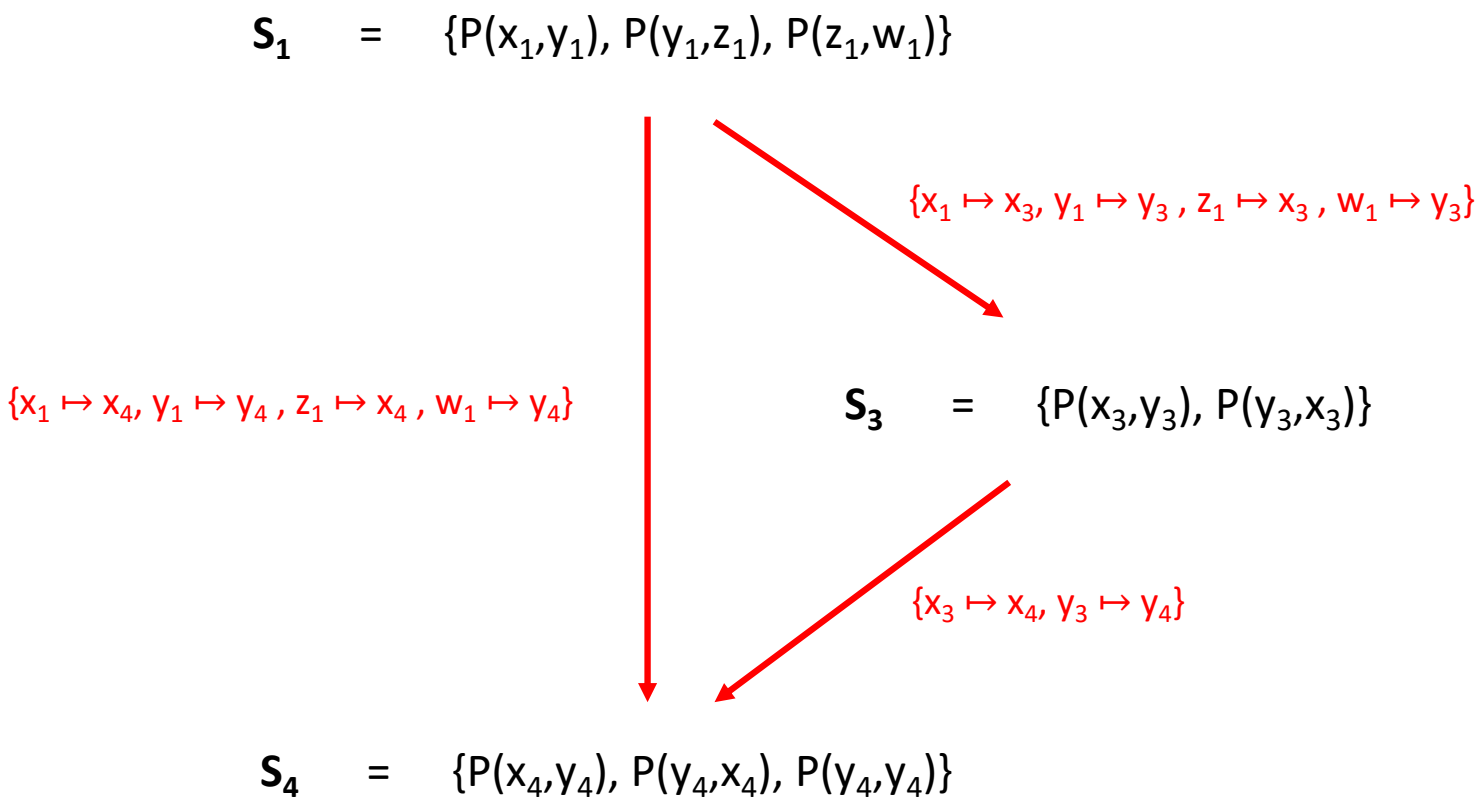
Find the Homomorphisms



Homomorphisms Compose



Homomorphisms Compose



Semantics of Conjunctive Queries

- A match of a conjunctive query $Q(x_1,\ldots,x_k)$:- body in a database D is a homomorphism h from the set of atoms body to the set of atoms D

- The answer to $Q(x_1,\ldots,x_k)$:- body over D is the set of k-tuples

$$Q(D) := \{(h(x_1),\ldots,h(x_k)) \mid h \text{ is a match of } Q \text{ in } D\}$$

- The answer consists of the witnesses for the distinguished variables of Q

Pattern Matching Problem

List the airlines that fly directly from London to Glasgow

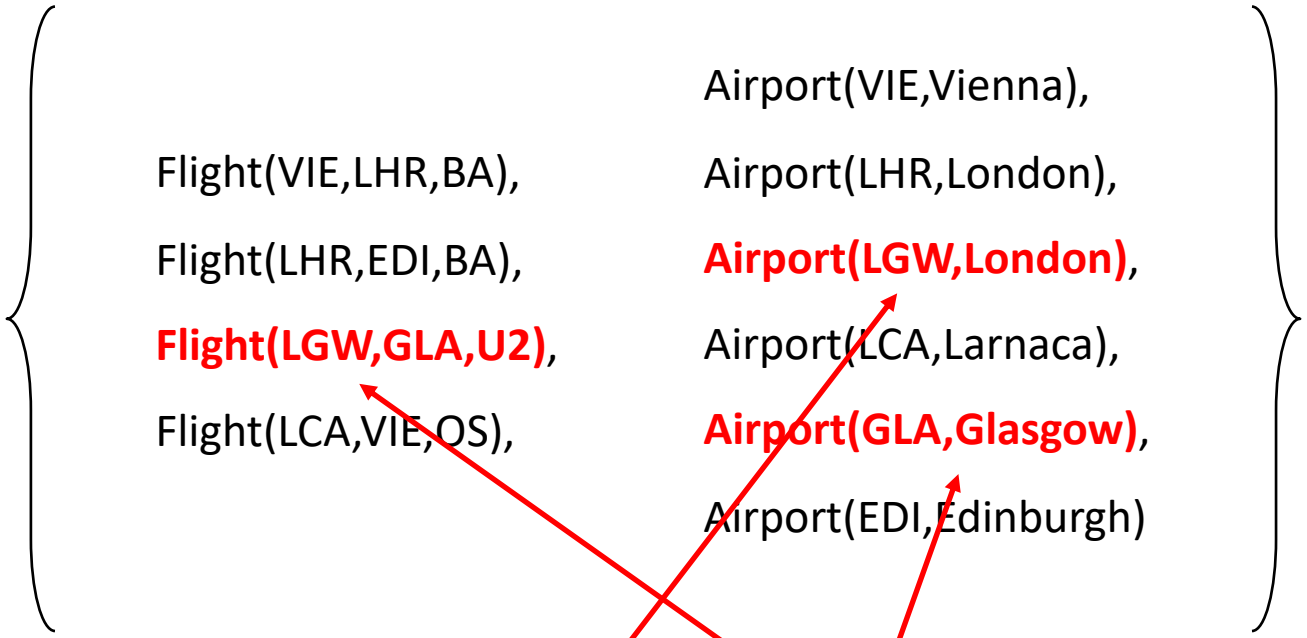
Flight(VIE,LHR,BA),
Flight(LHR,EDI,BA),
Flight(LGW,GLA,U2),
Flight(LCA,VIE,OS),

Airport(VIE,Vienna),
Airport(LHR,London),
Airport(LGW,London),
Airport(LCA,Larnaca),
Airport(GLA,Glasgow),
Airport(EDI,Edinburgh)

Q(z) :- Airport(x,London), Airport(y,Glasgow), Flight(x,y,z)

Pattern Matching Problem

List the airlines that fly directly from London to Glasgow



{x ↦ LGW, y ↦ GLA, z ↦ U2,

London ↦ London, Glasgow ↦ Glasgow}

Q(z) :- Airport(x,London), Airport(y,Glasgow), Flight(x,y,z)