Logic and Modelling

— Miscellaneous —

Jörg Endrullis

VU University Amsterdam

Databases, Queries and Predicate Logic

Queries in Databases: Relational Model

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

Queries in Databases: Relational Model

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

Relational Model

The table is modelled by a ternary predicate *Person*:

Person(101, Ann, Amsterdam)

Likewise, the predicate holds for every row of the table.

Queries in Databases: Relational Model

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

Relational Model

The table is modelled by a ternary predicate *Person*:

$$Person(101, \text{Ann}, \text{Amsterdam})$$

Likewise, the predicate holds for every row of the table.

We can formulate **queries using free variables**.

The names of people living in Amsterdam:

Queries in Databases: Relational Model

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

Relational Model

The table is modelled by a ternary predicate *Person*:

$$\mathit{Person}(101, \text{Ann}, \text{Amsterdam})$$

Likewise, the predicate holds for every row of the table.

We can formulate **queries using free variables**.

The names of people living in Amsterdam:

$$\{\, y \mid \exists i\ \mathit{Person}(i, y, \text{Amsterdam}) \,\}$$

Queries in Databases: Relational Model

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

Relational Model

The table is modelled by a ternary predicate *Person*:

$$\mathit{Person}(101, \text{Ann}, \text{Amsterdam})$$

Likewise, the predicate holds for every row of the table.

We can formulate **queries using free variables**.

The names of people living in Amsterdam:

$$\{\, y \mid \exists i\ \mathit{Person}(i, y, \text{Amsterdam}) \,\} = \{\, \text{Ann}, \text{Mia} \,\}$$

Queries in Databases: Tuple Model

Tuple Model

In this model, the **rows are objects** themselves (tuples).

Queries in Databases: Tuple Model

Tuple Model

In this model, the **rows are objects** themselves (tuples).

A unary predicate *Person* tells if a row belongs to table Persons:

$$Person(x)$$

Queries in Databases: Tuple Model

Tuple Model

In this model, the **rows are objects** themselves (tuples).

A unary predicate *Person* tells if a row belongs to table Persons:

$$Person(x)$$

Functions symbols for every column of the table:

$$\text{Pid}(x) \qquad \text{Name}(x) \qquad \text{City}(x)$$

Queries in Databases: Tuple Model

Tuple Model

In this model, the **rows are objects** themselves (tuples).

A unary predicate *Person* tells if a row belongs to table Persons:

$$Person(x)$$

Functions symbols for every column of the table:

$$\text{Pid}(x) \qquad \text{Name}(x) \qquad \text{City}(x)$$

(project the tuple/row to one of its arguments $\langle\, y_1, \ldots, y_n \,\rangle \mapsto y_i$)

Queries in Databases: Tuple Model

Tuple Model

In this model, the **rows are objects** themselves (tuples).

A unary predicate *Person* tells if a row belongs to table Persons:

$$Person(x)$$

Functions symbols for every column of the table:

$$\text{Pid}(x) \qquad \text{Name}(x) \qquad \text{City}(x)$$

(project the tuple/row to one of its arguments $\langle\, y_1, \ldots, y_n \,\rangle \mapsto y_i$)

We can formulate **queries using the function symbols**.

The names of people living in Amsterdam:

Queries in Databases: Tuple Model

Tuple Model

In this model, the **rows are objects** themselves (tuples).

A unary predicate *Person* tells if a row belongs to table Persons:

$$Person(x)$$

Functions symbols for every column of the table:

$$\text{Pid}(x) \qquad \text{Name}(x) \qquad \text{City}(x)$$

(project the tuple/row to one of its arguments $\langle\, y_1, \ldots, y_n \,\rangle \mapsto y_i$)

We can formulate **queries using the function symbols**.

The names of people living in Amsterdam:

$$\{\, y \mid \exists x \left(Person(x) \wedge \text{Name}(x) = y \wedge \text{City}(x) = \text{Amsterdam} \right) \}$$

Queries in Databases: Tuple Model

Tuple Model

In this model, the **rows are objects** themselves (tuples).

A unary predicate *Person* tells if a row belongs to table Persons:

$$Person(x)$$

Functions symbols for every column of the table:

$$\text{Pid}(x) \qquad \text{Name}(x) \qquad \text{City}(x)$$

(project the tuple/row to one of its arguments $\langle\, y_1, \ldots, y_n \,\rangle \mapsto y_i$)

We can formulate **queries using the function symbols**.

The names of people living in Amsterdam:

$$\{\, y \mid \exists x \left(Person(x) \wedge \text{Name}(x) = y \wedge \text{City}(x) = \text{Amsterdam} \right) \}$$

or equivalently

$$\{\, \text{Name}(x) \mid Person(x) \wedge \text{City}(x) = \text{Amsterdam} \,\}$$

Queries in Databases

Query: the names of people living in Amsterdam.

Relational Model

$$\{\ y \mid \exists i\ \mathit{Person}(i, y, \text{Amsterdam})\ \}$$

Tuple Model

$$\{\ \text{Name}(x) \mid \mathit{Person}(x) \wedge \text{City}(x) = \text{Amsterdam}\ \}$$

Queries in Databases

Query: the names of people living in Amsterdam.

Relational Model

$$\{\, y \mid \exists i\ \mathit{Person}(i, y, \text{Amsterdam}) \,\}$$

Tuple Model

$$\{\, \text{Name}(x) \mid \mathit{Person}(x) \wedge \text{City}(x) = \text{Amsterdam} \,\}$$

SQL Query

```
SELECT Name
FROM Person
WHERE City = 'Amsterdam'
```

Multiple Tables

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

CHATS

CID	P1	P2	Date
1	102	101	01.04.14
2	103	102	05.04.14
3	104	101	09.05.14
4	102	104	29.11.14

Query: PIDs of all people that have chatted with everyone in the persons table (except themselves).

Multiple Tables

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

CHATS

CID	P1	P2	Date
1	102	101	01.04.14
2	103	102	05.04.14
3	104	101	09.05.14
4	102	104	29.11.14

Query: PIDs of all people that have chatted with everyone in the persons table (except themselves).

Relational Model

Multiple Tables

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

CHATS

CID	P1	P2	Date
1	102	101	01.04.14
2	103	102	05.04.14
3	104	101	09.05.14
4	102	104	29.11.14

Query: PIDs of all people that have chatted with everyone in the persons table (except themselves).

Relational Model

$\{ y \mid$

$\}$

Multiple Tables

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

CHATS

CID	P1	P2	Date
1	102	101	01.04.14
2	103	102	05.04.14
3	104	101	09.05.14
4	102	104	29.11.14

Query: PIDs of all people that have chatted with everyone in the persons table (except themselves).

Relational Model

$$\{ y \mid \exists y_2, y_3 \; \mathit{Person}(y, y_2, y_3) \}$$

Multiple Tables

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

CHATS

CID	P1	P2	Date
1	102	101	01.04.14
2	103	102	05.04.14
3	104	101	09.05.14
4	102	104	29.11.14

Query: PIDs of all people that have chatted with everyone in the persons table (except themselves).

Relational Model

$$\{\, y \mid \exists y_2, y_3\ \mathit{Person}(y, y_2, y_3) \\ \wedge\ \forall x\ (\\ \rightarrow \\ )\ \}$$

Multiple Tables

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

CHATS

CID	P1	P2	Date
1	102	101	01.04.14
2	103	102	05.04.14
3	104	101	09.05.14
4	102	104	29.11.14

Query: PIDs of all people that have chatted with everyone in the persons table (except themselves).

Relational Model

$$\{\, y \mid \exists y_2, y_3\ \mathit{Person}(y, y_2, y_3) \wedge \forall x \,(\, x \neq y \wedge \exists x_2, x_3\ \mathit{Person}(x, x_2, x_3) \rightarrow \,) \,\}$$

Multiple Tables

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

CHATS

CID	P1	P2	Date
1	102	101	01.04.14
2	103	102	05.04.14
3	104	101	09.05.14
4	102	104	29.11.14

Query: PIDs of all people that have chatted with everyone in the persons table (except themselves).

Relational Model

$$\{\, y \mid \exists y_2, y_3\ \mathit{Person}(y, y_2, y_3) \wedge \forall x \left(x \neq y \wedge \exists x_2, x_3\ \mathit{Person}(x, x_2, x_3) \rightarrow \exists c_1, c_2\ (\mathrm{Chat}(c_1, y, x, c_2) \vee \mathrm{Chat}(c_1, x, y, c_2))\right) \}$$

Multiple Tables

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

CHATS

CID	P1	P2	Date
1	102	101	01.04.14
2	103	102	05.04.14
3	104	101	09.05.14
4	102	104	29.11.14

Query: PIDs of all people that have chatted with everyone in the persons table (except themselves).

Tuple Model

Multiple Tables

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

CHATS

CID	P1	P2	Date
1	102	101	01.04.14
2	103	102	05.04.14
3	104	101	09.05.14
4	102	104	29.11.14

Query: PIDs of all people that have chatted with everyone in the persons table (except themselves).

Tuple Model

$$\{ \text{PID}(y) \mid \mathit{Person}(y) \wedge \forall x \left(\mathit{Person}(x) \wedge x \neq y \rightarrow \right) \}$$

Multiple Tables

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

CHATS

CID	P1	P2	Date
1	102	101	01.04.14
2	103	102	05.04.14
3	104	101	09.05.14
4	102	104	29.11.14

Query: PIDs of all people that have chatted with everyone in the persons table (except themselves).

Tuple Model

$$\{ \text{PID}(y) \mid \mathit{Person}(y) \wedge \forall x \left(\mathit{Person}(x) \wedge x \neq y \rightarrow \exists c \left(\mathit{Chat}(c) \wedge \right. \right. \left. \left. \right) \right) \}$$

Multiple Tables

PERSONS

PID	Name	City
101	Ann	Amsterdam
102	Joe	Utrecht
103	Rick	Leipzig
104	Mia	Amsterdam

CHATS

CID	P1	P2	Date
1	102	101	01.04.14
2	103	102	05.04.14
3	104	101	09.05.14
4	102	104	29.11.14

Query: PIDs of all people that have chatted with everyone in the persons table (except themselves).

Tuple Model

$$\{\, \mathrm{PID}(y) \mid \mathit{Person}(y) \wedge \forall x \left(\mathit{Person}(x) \wedge x \neq y \rightarrow \exists c \left(\mathit{Chat}(c) \wedge \left((\mathrm{P1}(c) = \mathrm{PID}(x) \wedge \mathrm{P2}(c) = \mathrm{PID}(y)) \vee (\mathrm{P1}(c) = \mathrm{PID}(y) \wedge \mathrm{P2}(c) = \mathrm{PID}(x)) \right) \right) \right) \}$$

Multiple Tables

Query: PIDs of all people that have chatted with everyone in the persons table (except themselves).

SQL Query

```
SELECT Y.PID
FROM Person as Y
WHERE NOT EXISTS (
  SELECT *
  FROM Person as X
  WHERE Y.PID != X.PID
    AND NOT EXISTS (
      SELECT *
      FROM Chats as C
      WHERE ( C.P1 = Y.PID AND C.P2 = X.PID )
         OR ( C.P1 = X.PID AND C.P2 = Y.PID )
    )
)
```

There is no FORALL in SQL. Recall $\forall x\ \phi(x) \equiv \neg\exists x\ \neg\phi(x)$!

Relations as Query Results

Query: PID and Name of everybody who has chatted.

Relations as Query Results

Query: PID and Name of everybody who has chatted.

Relational Model

$$\{ \langle x, y \rangle \mid \qquad \}$$

Relations as Query Results

Query: PID and Name of everybody who has chatted.

Relational Model

$$\{ \langle x, y \rangle \mid \exists z \; \mathit{Person}(x, y, z) \}$$

Relations as Query Results

Query: PID and Name of everybody who has chatted.

Relational Model

$$\{ \langle x, y \rangle \mid \exists z \; \mathit{Person}(x, y, z) \\ \wedge \; \exists c_1, c_2, c_3 \; (\mathrm{Chat}(c_1, x, c_2, c_3) \vee \mathrm{Chat}(c_1, c_2, x, c_3)) \}$$

Relations as Query Results

Query: PID and Name of everybody who has chatted.

Relational Model

$$\{ \langle x, y \rangle \mid \exists z\, \mathit{Person}(x, y, z) \\ \wedge\ \exists c_1, c_2, c_3\, (\mathrm{Chat}(c_1, x, c_2, c_3) \vee \mathrm{Chat}(c_1, c_2, x, c_3)) \}$$

Tuple Model

$$\{ \langle \mathrm{PID}(p), \mathrm{Name}(p) \rangle \mid \mathit{Person}(p) \\ \}$$

Relations as Query Results

Query: PID and Name of everybody who has chatted.

Relational Model

$$\{ \langle x, y \rangle \mid \exists z\, \mathit{Person}(x, y, z) \\ \wedge\ \exists c_1, c_2, c_3\, (\mathrm{Chat}(c_1, x, c_2, c_3) \vee \mathrm{Chat}(c_1, c_2, x, c_3)) \}$$

Tuple Model

$$\{ \langle \mathrm{PID}(p), \mathrm{Name}(p) \rangle \mid \mathit{Person}(p) \\ \wedge\ \exists c\, (\mathrm{Chat}(c) \wedge (\mathrm{P1}(c) = \mathrm{PID}(p) \vee \mathrm{P2}(c) = \mathrm{PID}(p))) \}$$

Relations as Query Results

Query: PID and Name of everybody who has chatted.

Relational Model

$$\{ \langle x, y \rangle \mid \exists z\, \mathit{Person}(x, y, z) \\ \wedge\ \exists c_1, c_2, c_3\, (\mathsf{Chat}(c_1, x, c_2, c_3) \vee \mathsf{Chat}(c_1, c_2, x, c_3)) \}$$

Tuple Model

$$\{ \langle \mathsf{PID}(p), \mathsf{Name}(p) \rangle \mid \mathit{Person}(p) \\ \wedge\ \exists c\, (\mathsf{Chat}(c) \wedge (\mathsf{P1}(c) = \mathsf{PID}(p) \vee \mathsf{P2}(c) = \mathsf{PID}(p))) \}$$

SQL Query

```
SELECT P.PID, P.Name FROM Person as P
WHERE EXISTS (
  SELECT * FROM Chats as C
  WHERE C.P1 = P.PID OR C.P2 = P.PID
)
```

Exercises

Exercises

Explain the meaning of $\models$ and $\vdash$ for predicate logic.

Exercises

Explain the meaning of $\models$ and $\vdash$ for predicate logic.

$\phi_1, \ldots, \phi_n \models \psi$ means that all models $\mathcal{M}$ and environments ℓ that make $\phi_1, \ldots, \phi_n$ true, also make ψ true.

Exercises

Explain the meaning of $\models$ and $\vdash$ for predicate logic.

$\phi_1, \ldots, \phi_n \models \psi$ means that all models $\mathcal{M}$ and environments ℓ that make $\phi_1, \ldots, \phi_n$ true, also make ψ true.

$\phi_1, \ldots, \phi_n \vdash \psi$ means ψ is derivable using natural deduction starting from premises $\phi_1, \ldots, \phi_n$.

Exercises

Explain the meaning of $\models$ and $\vdash$ for predicate logic.

$\phi_1, \ldots, \phi_n \models \psi$ means that all models $\mathcal{M}$ and environments ℓ that make $\phi_1, \ldots, \phi_n$ true, also make ψ true.

$\phi_1, \ldots, \phi_n \vdash \psi$ means ψ is derivable using natural deduction starting from premises $\phi_1, \ldots, \phi_n$.

Explain soundness/correctness and completeness.

Exercises

Explain the meaning of $\models$ and $\vdash$ for predicate logic.

$\phi_1, \ldots, \phi_n \models \psi$ means that all models $\mathcal{M}$ and environments ℓ that make $\phi_1, \ldots, \phi_n$ true, also make ψ true.

$\phi_1, \ldots, \phi_n \vdash \psi$ means ψ is derivable using natural deduction starting from premises $\phi_1, \ldots, \phi_n$.

Explain soundness/correctness and completeness.

Soundness means that everything derivable by natural deduction is also semantically entailed:

$$\Gamma \vdash \phi \implies \Gamma \models \phi$$

Exercises

Explain the meaning of $\models$ and $\vdash$ for predicate logic.

$\phi_1, \ldots, \phi_n \models \psi$ means that all models $\mathcal{M}$ and environments ℓ that make $\phi_1, \ldots, \phi_n$ true, also make ψ true.

$\phi_1, \ldots, \phi_n \vdash \psi$ means ψ is derivable using natural deduction starting from premises $\phi_1, \ldots, \phi_n$.

Explain soundness/correctness and completeness.

Soundness means that everything derivable by natural deduction is also semantically entailed:

$$\Gamma \vdash \phi \implies \Gamma \models \phi$$

Completeness means that the derivation rules are strong enough to derive everything that is semantically entailed:

$$\Gamma \models \phi \implies \Gamma \vdash \phi$$

Exercises

Assume you want to disprove

$$\phi \vdash \psi$$

How can the soundness or completeness theorem help?

Exercises

Assume you want to disprove

$$\phi \vdash \psi$$

How can the soundness or completeness theorem help?

To show that there is no possible proof might be difficult.

Exercises

Assume you want to disprove

$$\phi \vdash \psi$$

How can the soundness or completeness theorem help?

To show that there is no possible proof might be difficult.

It is easier to give a counter-model.

Exercises

Assume you want to disprove

$$\phi \vdash \psi$$

How can the soundness or completeness theorem help?

To show that there is no possible proof might be difficult.

It is easier to give a counter-model.

That is, a model $\mathcal{M}$ and environment ℓ such that

$$\mathcal{M} \models_\ell \phi \qquad \text{and} \qquad \mathcal{M} \not\models_\ell \psi$$

Exercises

Assume you want to disprove

$$\phi \vdash \psi$$

How can the soundness or completeness theorem help?

To show that there is no possible proof might be difficult.

It is easier to give a counter-model.

That is, a model $\mathcal{M}$ and environment ℓ such that

$$\mathcal{M} \models_\ell \phi \qquad \text{and} \qquad \mathcal{M} \not\models_\ell \psi$$

Then we know that $\phi \not\models \psi$.

Exercises

Assume you want to disprove

$$\phi \vdash \psi$$

How can the soundness or completeness theorem help?

To show that there is no possible proof might be difficult.

It is easier to give a counter-model.

That is, a model $\mathcal{M}$ and environment ℓ such that

$$\mathcal{M} \models_\ell \phi \qquad \text{and} \qquad \mathcal{M} \not\models_\ell \psi$$

Then we know that $\phi \not\models \psi$.

By the soundness we have

$$\phi \vdash \psi \implies \phi \models \psi$$

Exercises

Assume you want to disprove

$$\phi \vdash \psi$$

How can the soundness or completeness theorem help?

To show that there is no possible proof might be difficult.

It is easier to give a counter-model.

That is, a model $\mathcal{M}$ and environment ℓ such that

$$\mathcal{M} \models_\ell \phi \qquad \text{and} \qquad \mathcal{M} \not\models_\ell \psi$$

Then we know that $\phi \not\models \psi$.

By the soundness we have

$$\phi \vdash \psi \implies \phi \models \psi$$

Hence we conclude $\phi \nvdash \psi$.

Exercises

Meaning of the symbols:

a : Alice $\qquad\qquad$ $L(x, y)$: x likes y

Translate the following sentences into predicate logic:

Everybody who likes someone, also likes Alice.

Exercises

Meaning of the symbols:

a : Alice $\qquad\qquad$ $L(x, y)$: x likes y

Translate the following sentences into predicate logic:

Everybody who likes someone, also likes Alice.

$$\forall x \left(\exists y \; L(x, y) \rightarrow L(x, a) \right)$$

Exercises

Meaning of the symbols:

a : Alice $\qquad\qquad$ $L(x, y)$: x likes y

Translate the following sentences into predicate logic:

Everybody who likes someone, also likes Alice.

$$\forall x \left(\exists y \; L(x, y) \rightarrow L(x, a) \right)$$

The brackets are important. A typical **mistake** in the exam:

$$\forall x \; \exists y \left(L(x, y) \rightarrow L(x, a) \right)$$

Exercises

Meaning of the symbols:

a : Alice $\qquad\qquad$ $L(x, y)$: x likes y

Translate the following sentences into predicate logic:

Everybody who likes someone, also likes Alice.

$$\forall x \left(\exists y \; L(x, y) \rightarrow L(x, a) \right)$$

The brackets are important. A typical **mistake** in the exam:

$$\forall x \; \exists y \left(L(x, y) \rightarrow L(x, a) \right)$$

What does this formula mean?

Exercises

Meaning of the symbols:

a : Alice $\qquad$ $L(x, y)$: x likes y

Translate the following sentences into predicate logic:

Everybody likes at most 2 *other* people.

Exercises

Meaning of the symbols:

a : Alice $\qquad\qquad$ $L(x, y)$: x likes y

Translate the following sentences into predicate logic:

Everybody likes at most 2 *other* people.

Note that **at most** means that it can be 0, 1 or 2 !

Exercises

Meaning of the symbols:

a : Alice $\qquad\qquad$ $L(x, y)$: x likes y

Translate the following sentences into predicate logic:

Everybody likes at most 2 *other* people.

Note that **at most** means that it can be 0, 1 or 2 !

$$\forall x\, \exists y_1\, \exists y_2\, \forall z\, \big(\, L(x, z) \wedge z \neq x \;\rightarrow\; z = y_1 \vee z = y_2 \,\big)$$

Exercises

Meaning of the symbols:

a : Alice $\qquad\qquad$ $L(x, y)$: x likes y

Translate the following sentences into predicate logic:

Everybody likes at most 2 *other* people.

Note that **at most** means that it can be 0, 1 or 2 !

$$\forall x\ \exists y_1\ \exists y_2\ \forall z\ \big(\ L(x,z) \wedge z \neq x\ \rightarrow\ z = y_1 \vee z = y_2\ \big)$$

or, equivalently

$$\begin{aligned}\forall x\ \forall y_1\ \forall y_2\ \forall y_3\ \big(\ & L(x,y_1) \wedge L(x,y_2) \wedge L(x,y_3) \\ & \wedge x \neq y_1 \wedge x \neq y_2 \wedge x \neq y_3 \\ & \rightarrow y_1 = y_2 \vee y_2 = y_3 \vee y_1 = y_3\ \big)\end{aligned}$$

Examples

Give a counter-model for:

$$\forall x\, \exists y\, R(x,y),$$
$$\forall x\, \forall y\, \forall z\, \big(R(x,y) \wedge R(y,z) \rightarrow R(x,z)\big)$$
$$\models \quad \exists x\, R(x,x)$$

Examples

Give a counter-model for:

$$\forall x\, \exists y\, R(x,y),$$
$$\forall x\, \forall y\, \forall z\, \big(R(x,y) \wedge R(y,z) \rightarrow R(x,z)\big)$$
$$\models \quad \exists x\, R(x,x)$$

What do the premises it mean?

Examples

Give a counter-model for:

$$\forall x\, \exists y\, R(x,y),$$
$$\forall x\, \forall y\, \forall z\, \big(R(x,y) \wedge R(y,z) \rightarrow R(x,z)\big)$$
$$\models \quad \exists x\, R(x,x)$$

What do the premises it mean?

(a) Every object has a successor.

Examples

Give a counter-model for:

$$\forall x\, \exists y\, R(x,y),$$
$$\forall x\, \forall y\, \forall z\, \big(R(x,y) \wedge R(y,z) \rightarrow R(x,z)\big)$$
$$\models \quad \exists x\, R(x,x)$$

What do the premises it mean?

(a) Every object has a successor.

(b) The successor-relation is transitive.

Examples

Give a counter-model for:

$$\forall x\, \exists y\, R(x,y),$$
$$\forall x\, \forall y\, \forall z\, \big(R(x,y) \wedge R(y,z) \rightarrow R(x,z)\big)$$
$$\models \quad \exists x\, R(x,x)$$

What do the premises it mean?

(a) Every object has a successor.

(b) The successor-relation is transitive.
Hence any n-step successor is an immediate successor.

Examples

Give a counter-model for:

$$\forall x\, \exists y\, R(x,y),$$
$$\forall x\, \forall y\, \forall z\, \big(R(x,y) \wedge R(y,z) \rightarrow R(x,z)\big)$$
$$\models \quad \exists x\, R(x,x)$$

What do the premises it mean?

(a) Every object has a successor.

(b) The successor-relation is transitive.
Hence any n-step successor is an immediate successor.

What does the conclusion mean?

Examples

Give a counter-model for:

$$\forall x\, \exists y\, R(x,y),$$
$$\forall x\, \forall y\, \forall z\, \big(R(x,y) \wedge R(y,z) \rightarrow R(x,z)\big)$$
$$\models \quad \exists x\, R(x,x)$$

What do the premises it mean?

(a) Every object has a successor.

(b) The successor-relation is transitive.
Hence any n-step successor is an immediate successor.

What does the conclusion mean?

- There is an object that is its own successor.

Examples

Give a counter-model for:

$$\forall x\, \exists y\, R(x,y),$$
$$\forall x\, \forall y\, \forall z\, \big(R(x,y) \wedge R(y,z) \rightarrow R(x,z)\big)$$
$$\models \quad \exists x\, R(x,x)$$

What do the premises it mean?

(a) Every object has a successor.

(b) The successor-relation is transitive.
Hence any n-step successor is an immediate successor.

What does the conclusion mean?

- There is an object that is its own successor.

Can there be finite counter-models?

Examples

Give a counter-model for:

$$\forall x\, \exists y\, R(x,y),$$
$$\forall x\, \forall y\, \forall z\, \big(R(x,y) \wedge R(y,z) \rightarrow R(x,z)\big)$$
$$\models \quad \exists x\, R(x,x)$$

What do the premises it mean?

(a) Every object has a successor.

(b) The successor-relation is transitive.
Hence any n-step successor is an immediate successor.

What does the conclusion mean?

- There is an object that is its own successor.

Can there be finite counter-models?

No, because by (a) there would by cycles, and by (b) every element on a cycle would be its own successor.

Examples

Give a counter-model for:

$$\forall x\, \exists y\, R(x,y),$$
$$\forall x\, \forall y\, \forall z\, \big(R(x,y) \wedge R(y,z) \rightarrow R(x,z)\big)$$
$$\models \quad \exists x\, R(x,x)$$

So, what could be an infinite counter-model?

Examples

Give a counter-model for:

$$\forall x\, \exists y\, R(x, y),$$
$$\forall x\, \forall y\, \forall z\, \big(R(x, y) \wedge R(y, z) \rightarrow R(x, z)\big)$$
$$\models \quad \exists x\, R(x, x)$$

So, what could be an infinite counter-model?

A counter-model:

- $A = \mathbb{N}$ (the universe is the set of natural numbers)
- $R = <$

Question Hour

Success with the Exam