The analysis of the beginning would thus yield
the notion of the unity of being and
not-being—or, in a more reflected form, the unity
of differentiatedness and non-differentiatedness, or
the identity of identity and non-identity.

Hegel
The Science of Logic
Outline

(1) The language of predicate logic with identity: \( \mathcal{L}_= \)

- Syntax
- Semantics
- Proof theory

(2) Formalisation in \( \mathcal{L}_= \)

- Numerical quantifiers
- Definite descriptions
The logicians’ sense of ‘identical’

In English, we use the words ‘identity’/‘identical’ in a number of different ways.
The logicians’ sense of ‘identical’

In English, we use the words ‘identity’/‘identical’ in a number of different ways.

Wider uses of ‘identity’/‘identical’

(1) Mancunians have a strong sense of cultural identity.
The logicians’ sense of ‘identical’

In English, we use the words ‘identity’/‘identical’ in a number of different ways.

**Wider uses of ‘identity’/‘identical’**

1. Mancunians have a strong sense of cultural identity.
2. Dr. Jekyll has multiple identities.
The logicians’ sense of ‘identical’

In English, we use the words ‘identity’/‘identical’ in a number of different ways.

Wider uses of ‘identity’/‘identical’

(1) Mancunians have a strong sense of cultural identity.
(2) Dr. Jekyll has multiple identities.
(3) Jedward are almost completely identical.
The logicians’ sense of ‘identical’

In English, we use the words ‘identity’/‘identical’ in a number of different ways.

<table>
<thead>
<tr>
<th>Wider uses of ‘identity’/‘identical’</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) Mancunians have a strong sense of cultural identity.</td>
</tr>
<tr>
<td>(2) Dr. Jekyll has multiple identities.</td>
</tr>
<tr>
<td>(3) Jedward are almost completely identical.</td>
</tr>
</tbody>
</table>

The sense of ‘identity’ used in (3) is sometimes called ‘qualitative identity’.
The logicians’ sense of ‘identical’

In English, we use the words ‘identity’/‘identical’ in a number of different ways.

**Wider uses of ‘identity’/‘identical’**

(1) Mancunians have a strong sense of cultural identity.

(2) Dr. Jekyll has multiple identities.

(3) Jedward are almost completely identical.

The sense of ‘identity’ used in (3) is sometimes called ‘qualitative identity’.

- (3) says that John and Edward are almost exactly similar in every respect.
8.1 Qualitative and Numerical Identity

The logicians’ sense of ‘identical’

In English, we use the words ‘identity’/‘identical’ in a number of different ways.

Wider uses of ‘identity’/‘identical’

(1) Mancunians have a strong sense of cultural identity.
(2) Dr. Jekyll has multiple identities.
(3) Jedward are almost completely identical.

The sense of ‘identity’ used in (3) is sometimes called ‘qualitative identity’.

(3) says that John and Edward are almost exactly similar in every respect.

None of these uses of ‘identical’ is the logicians’ use.
In logic, we always use ‘identical’ in the following strict sense

A is identical to B iff A is the very same thing as B
i.e. A and B are one and the same thing.
In logic, we always use ‘identical’ in the following strict sense

A is identical to B iff A is the very same thing as B
i.e. A and B are one and the same thing.

This is sometimes called ‘**numerical identity**’
In logic, we always use ‘identical’ in the following strict sense:

A is identical to B iff A is the very same thing as B
  i.e. A and B are one and the same thing.

This is sometimes called ‘numerical identity’

(Unless otherwise stated ‘identity’/‘identical’ henceforth mean numerical identity/numerically identical.)
In logic, we always use ‘identical’ in the following strict sense

A is identical to B iff A is the very same thing as B
i.e. A and B are one and the same thing.

This is sometimes called ‘numerical identity’

(Unless otherwise stated ‘identity’/‘identical’ henceforth mean numerical identity/numerically identical.)

Examples

- George Orwell is identical to Eric Arthur Blair
In logic, we always use ‘identical’ in the following strict sense

A is identical to B iff A is the very same thing as B
i.e. A and B are one and the same thing.

This is sometimes called ‘numerical identity’

(Unless otherwise stated ‘identity’/‘identical’ henceforth
mean numerical identity/numerically identical.)

**Examples**
- George Orwell is identical to Eric Arthur Blair
- Dr. Jekyll is identical to Mr. Hyde
In logic, we always use ‘identical’ in the following strict sense

A is identical to B iff A is the very same thing as B

i.e. A and B are one and the same thing.

This is sometimes called ‘numerical identity’

(Unless otherwise stated ‘identity’/‘identical’ henceforth mean numerical identity/numerically identical.)

Examples

- George Orwell is identical to Eric Arthur Blair
- Dr. Jekyll is identical to Mr. Hyde
- John is not identical to Edward
A third formal language

The new language makes a single addition to $\mathcal{L}_2$. 
A third formal language

The new language makes a single addition to $\mathcal{L}_2$.

**The language $\mathcal{L}_=$**

The language $\mathcal{L}_=$ of predicate logic with identity adds a single binary predicate letter to the language of predicate logic $\mathcal{L}_2$.

- $\mathcal{L}_=$ adds the identity predicate $=$ to $\mathcal{L}_2$
A third formal language

The new language makes a single addition to $\mathcal{L}_2$.

The language $\mathcal{L}_=$

The language $\mathcal{L}_=$ of predicate logic with identity adds a single binary predicate letter to the language of predicate logic $\mathcal{L}_2$.

- $\mathcal{L}_=$ adds the identity predicate $=$ to $\mathcal{L}_2$

$=$ differs from the other predicate letters in several ways.
A third formal language

The new language makes a single addition to $\mathcal{L}_2$.

The language $\mathcal{L}_=$

The language $\mathcal{L}_=$ of predicate logic with identity adds a single binary predicate letter to the language of predicate logic $\mathcal{L}_2$.

- $\mathcal{L}_=$ adds the identity predicate $=$ to $\mathcal{L}_2$

$=$ differs from the other predicate letters in several way.

- $P$, $R^2$, etc., are non-logical expressions.
  Different $\mathcal{L}_2$-structures interpret them differently.
A third formal language

The new language makes a single addition to $L_2$.

The language $L_=$

The language $L_=$ of predicate logic with identity adds a single binary predicate letter to the language of predicate logic $L_2$.

- $L_=$ adds the identity predicate $=$ to $L_2$

$=$ differs from the other predicate letters in several ways.

- $P$, $R^2$, etc., are non-logical expressions.
  Different $L_2$-structures interpret them differently.

- $=$ is treated as a logical expression.
  It always has the same interpretation in any structure.
A third formal language

The new language makes a single addition to $\mathcal{L}_2$.

**The language $\mathcal{L}_=$**

The language $\mathcal{L}_=$ of predicate logic with identity adds a single binary predicate letter to the language of predicate logic $\mathcal{L}_2$.

- $\mathcal{L}_=$ adds the identity predicate $\equiv$ to $\mathcal{L}_2$

$\equiv$ differs from the other predicate letters in several way.

- $P, R^2$, etc., are non-logical expressions.
  Different $\mathcal{L}_2$-structures interpret them differently.

- $\equiv$ is treated as a logical expression.
  It always has the same interpretation in any structure.

- Minor difference: we write $a \equiv b$ (rather than $\equiv ab$).
We make a slight change to the definition of atomic formula.

**Definition (atomic formulae of $\mathcal{L}_=$)**

All atomic formulae of $\mathcal{L}_2$ are atomic formulae of $\mathcal{L}_=$. Furthermore, if $s$ and $t$ are variables or constants, then $s = t$ is an atomic formula of $\mathcal{L}_=$.
8.2 The Syntax of $\mathcal{L}_=$

Syntax

We make a slight change to the definition of atomic formula.

**Definition (atomic formulae of $\mathcal{L}_=$)**

All atomic formulae of $\mathcal{L}_2$ are atomic formulae of $\mathcal{L}_=$. Furthermore, if $s$ and $t$ are variables or constants, then $s = t$ is an atomic formula of $\mathcal{L}_=$.

The definition of formula and sentence is otherwise just like the definition for $\mathcal{L}_2$.

**Examples**

- Atomic $\mathcal{L}_=$-formulae:
Syntax

We make a slight change to the definition of atomic formula.

**Definition (atomic formulae of $\mathcal{L}_\leq$)**

All atomic formulae of $\mathcal{L}_2$ are atomic formulae of $\mathcal{L}_\leq$. Furthermore, if $s$ and $t$ are variables or constants, then $s = t$ is an atomic formula of $\mathcal{L}_\leq$.

The definition of formula and sentence is otherwise just like the definition for $\mathcal{L}_2$.

**Examples**

- Atomic $\mathcal{L}_\leq$-formulae: $c = a$, $x = y$, $x = a$, $R_{2} ax$.
8.2 The Syntax of $\mathcal{L}_=$

**Syntax**

We make a slight change to the definition of atomic formula.

**Definition (atomic formulae of $\mathcal{L}_=$)**

All atomic formulae of $\mathcal{L}_2$ are atomic formulae of $\mathcal{L}_=$. Furthermore, if $s$ and $t$ are variables or constants, then $s = t$ is an atomic formula of $\mathcal{L}_=$.

The definition of formula and sentence is otherwise just like the definition for $\mathcal{L}_2$.

**Examples**

- Atomic $\mathcal{L}_=$-formulae: $c = a$, $x = y_3$, 

We make a slight change to the definition of atomic formula.

**Definition (atomic formulae of $\mathcal{L}_=$)**

All atomic formulae of $\mathcal{L}_2$ are atomic formulae of $\mathcal{L}_=$. Furthermore, if $s$ and $t$ are variables or constants, then $s = t$ is an atomic formula of $\mathcal{L}_=$.

The definition of formula and sentence is otherwise just like the definition for $\mathcal{L}_2$.

**Examples**

- Atomic $\mathcal{L}_=$-formulae: $c = a$, $x = y_3$, $x = a$, 
Syntax

We make a slight change to the definition of atomic formula.

Definition (atomic formulae of $\mathcal{L}_\leq$)

All atomic formulae of $\mathcal{L}_2$ are atomic formulae of $\mathcal{L}_\leq$.
Furthermore, if $s$ and $t$ are variables or constants, then $s = t$ is an atomic formula of $\mathcal{L}_\leq$.

The definition of formula and sentence is otherwise just like the definition for $\mathcal{L}_2$.

Examples

- Atomic $\mathcal{L}_\leq$-formulae: $c = a$, $x = y_3$, $x = a$, $R^2 ax$. 
8.2 The Syntax of $\mathcal{L}_\leq$

Syntax

We make a slight change to the definition of atomic formula.

**Definition (atomic formulae of $\mathcal{L}_\leq$)**

All atomic formulae of $\mathcal{L}_2$ are atomic formulae of $\mathcal{L}_\leq$. Furthermore, if $s$ and $t$ are variables or constants, then $s = t$ is an atomic formula of $\mathcal{L}_\leq$.

The definition of formula and sentence is otherwise just like the definition for $\mathcal{L}_2$.

**Examples**

- Atomic $\mathcal{L}_\leq$-formulae: $c = a$, $x = y_3$, $x = a$, $R^2 ax$.
- Complex $\mathcal{L}_\leq$-formulae:
Syntax

We make a slight change to the definition of atomic formula.

Definition (atomic formulae of $\mathcal{L}_=$)

All atomic formulae of $\mathcal{L}_2$ are atomic formulae of $\mathcal{L}_=$. Furthermore, if $s$ and $t$ are variables or constants, then $s = t$ is an atomic formula of $\mathcal{L}_=$.

The definition of formula and sentence is otherwise just like the definition for $\mathcal{L}_2$.

Examples

- Atomic $\mathcal{L}_=$-formulae: $c = a$, $x = y_3$, $x = a$, $R^2ax$.
- Complex $\mathcal{L}_=$-formulae: $\neg x = y$, 

Syntax

We make a slight change to the definition of atomic formula.

Definition (atomic formulae of $\mathcal{L}_=$)

All atomic formulae of $\mathcal{L}_2$ are atomic formulae of $\mathcal{L}_=$. Furthermore, if $s$ and $t$ are variables or constants, then $s = t$ is an atomic formula of $\mathcal{L}_=$.

The definition of formula and sentence is otherwise just like the definition for $\mathcal{L}_2$.

Examples

- Atomic $\mathcal{L}_=$-formulae: $c = a$, $x = y_3$, $x = a$, $R^2 ax$.
- Complex $\mathcal{L}_=$-formulae: $\neg x = y$, $\forall x (Rxy_2 \rightarrow y_2 = x)$. 
Semantics

Definition:
\[ L = \text{-structure} \]

An \( L = \text{-structure} \) is simply an \( L^2 \)-structure.

Why no change?
Structures interpret non-logical expressions like \( P \) and \( a \).
Structures do not interpret logical expressions like \( \neg \) and \( \forall x \).
The fixed interpretation of logical expressions is specified in the definition of satisfaction.

\[ |\neg \phi|_\alpha A = T \iff |\phi|_\alpha A = F \]

Similarly, \( = \) is treated as a logical expression, which is not assigned a semantic value by the structure.
The fixed interpretation of \( = \) is specified in the definition of satisfaction.
Semantics

The definition of structure is just the same as before.

**Definition: \( \mathcal{L}_= \)-structure**

An \( \mathcal{L}_= \)-structure is simply an \( \mathcal{L}_2 \)-structure.
Semantics

The definition of structure is just the same as before.

**Definition: $\mathcal{L}_\equiv$-structure**

An $\mathcal{L}_\equiv$-structure is simply an $\mathcal{L}_2$-structure.

Why no change?
Semantics

The definition of structure is just the same as before.

**Definition: $\mathcal{L}_\leq$-structure**

An $\mathcal{L}_\leq$-structure is simply an $\mathcal{L}_2$-structure.

Why no change?

- Structures interpret non-logical expressions like $P$ and $a$. 
Semantics

The definition of structure is just the same as before.

**Definition: $\mathcal{L}_{\equiv}$-structure**

An $\mathcal{L}_{\equiv}$-structure is simply an $\mathcal{L}_2$-structure.

Why no change?

- Structures interpret non-logical expressions like $P$ and $a$.
- Structures do not interpret logical expressions like $\neg$ and $\forall x$. 
Semantics

The definition of structure is just the same as before.

**Definition: \( \mathcal{L}_\equiv \)-structure**

An \( \mathcal{L}_\equiv \)-structure is simply an \( \mathcal{L}_2 \)-structure.

Why no change?

- Structures interpret non-logical expressions like \( P \) and \( a \).
- Structures do not interpret logical expressions like \( \neg \) and \( \forall x \).
- The fixed interpretation of logical expressions is specified in the definition of satisfaction.
The definition of structure is just the same as before.

**Definition: \( \mathcal{L}_\equiv \)-structure**

An \( \mathcal{L}_\equiv \)-structure is simply an \( \mathcal{L}_2 \)-structure.

Why no change?

- Structures interpret non-logical expressions like \( P \) and \( a \).
- Structures do not interpret logical expressions like \( \neg \) and \( \forall x \).
- The fixed interpretation of logical expressions is specified in the definition of satisfaction.

**e.g.** \( |\neg \phi|^\mathcal{A} = \top \text{ iff } |\phi|^\mathcal{A} = \bot \)
Semantics

The definition of structure is just the same as before.

**Definition: \( \mathcal{L}_\equiv \)-structure**

An \( \mathcal{L}_\equiv \)-structure is simply an \( \mathcal{L}_2 \)-structure.

Why no change?

- Structures interpret non-logical expressions like \( P \) and \( a \).
- Structures do not interpret logical expressions like \( \neg \) and \( \forall x \).
- The fixed interpretation of logical expressions is specified in the definition of satisfaction.
  
  e.g. \(|\neg \phi|^\alpha_A = T \iff |\phi|^\alpha_A = F\)

- Similarly \( = \) is treated as a logical expression, which is not assigned a semantic value by the structure.
Semantics

The definition of structure is just the same as before.

**Definition: \( L_\equiv\)-structure**

An \( L_\equiv\)-structure is simply an \( L_2\)-structure.

Why no change?

- Structures interpret non-logical expressions like \( P \) and \( a \).
- Structures do not interpret logical expressions like \( \neg \) and \( \forall x \).
- The fixed interpretation of logical expressions is specified in the definition of satisfaction.
  
  e.g. \( |\neg \phi|_A^\alpha = T \) iff \( |\phi|_A^\alpha = F \)

- Similarly \( = \) is treated as a logical expression, which is not assigned a semantic value by the structure.

- The fixed interpretation of \( = \) is specified in the definition of satisfaction.
Let $\mathcal{A}$ be an $\mathcal{L}_\equiv$-structure (i.e. an $\mathcal{L}_2$-structure).
Truth in $\mathcal{A}$ is defined just as before with one addition:
Let \( \mathcal{A} \) be an \( \mathcal{L}_{=} \)-structure (i.e. an \( \mathcal{L}_2 \)-structure).

Truth in \( \mathcal{A} \) is defined just as before with one addition:

**Definition: satisfaction of identity statements**

\[
(ix) \quad |s = t|_\mathcal{A}^\alpha = T \text{ if and only if } |s|_\mathcal{A}^\alpha = |t|_\mathcal{A}^\alpha.
\]
Let $\mathcal{A}$ be an $\mathcal{L}_-$-structure (i.e. an $\mathcal{L}_2$-structure).
Truth in $\mathcal{A}$ is defined just as before with one addition:

**Definition: satisfaction of identity statements**

(ix) $|s=t|_{\mathcal{A}}^\alpha = T$ if and only if $|s|_{\mathcal{A}}^\alpha = |t|_{\mathcal{A}}^\alpha$.

**Note:** $\equiv$ is used in both $\mathcal{L}_-$ and the metalanguage.
Let $\mathcal{A}$ be an $\mathcal{L}_=$-structure (i.e. an $\mathcal{L}_2$-structure).

Truth in $\mathcal{A}$ is defined just as before with one addition:

**Definition: satisfaction of identity statements**

(ix) $|s = t|_\mathcal{A}^\alpha = T$ if and only if $|s|_\mathcal{A}^\alpha = |t|_\mathcal{A}^\alpha$.

**Note:** $=$ is used in both $\mathcal{L}_=$ and the metalanguage.
Let $\mathcal{A}$ be an $\mathcal{L}_=$-structure (i.e. an $\mathcal{L}_2$-structure).
Truth in $\mathcal{A}$ is defined just as before with one addition:

**Definition: satisfaction of identity statements**

$|s=t|_A^\alpha = T$ if and only if $|s|_A^\alpha = |t|_A^\alpha$.

**Note:** $=$ is used in both $\mathcal{L}_=$ and the metalanguage.
Let $\mathcal{A}$ be an $\mathcal{L}_-$-structure (i.e. an $\mathcal{L}_2$-structure).
Truth in $\mathcal{A}$ is defined just as before with one addition:

**Definition: satisfaction of identity statements**

(ix) $|s = t|^\alpha_{\mathcal{A}} = T$ if and only if $|s|^\alpha_{\mathcal{A}} = |t|^\alpha_{\mathcal{A}}$.

**Note:** $=$ is used in both $\mathcal{L}_-$ and the metalanguage.

The other definitions from Chapter 5 carry over directly to $\mathcal{L}_-$.

- Valid
- Logical truth
- Contradiction
- Logically equivalent
- Semantically consistent

These are defined just as before replacing ‘$\mathcal{L}_2$’ with ‘$\mathcal{L}_-$’.
Worked example

∀x ∀y x = y isn’t logically true.
Worked example

∀x ∀y x = y isn’t logically true.

Counterexample: let A be an L¬-structure with domain {1, 2}. 
Worked example

∀x ∀y x = y isn’t logically true.

Counterexample: let $\mathcal{A}$ be an $\mathcal{L}_{\leq}$-structure with domain $\{1, 2\}$.

Proof.
**Worked example**

\[ \forall x \forall y x = y \text{ isn’t logically true.} \]

Counterexample: let \( A \) be an \( \mathcal{L}_= \)-structure with domain \( \{1, 2\} \).

Proof. Let \( \alpha \) be an assignment over \( \mathcal{A} \).
Worked example

\( \forall x \forall y x = y \) isn’t logically true.

Counterexample: let \( \mathcal{A} \) be an \( \mathcal{L}_{=}-\)structure with domain \( \{1, 2\} \).

Proof. Let \( \alpha \) be an assignment over \( \mathcal{A} \).
Sufficient to prove (STP:) \( \forall x \forall y x = y \) is false in \( \mathcal{A} \) under \( \alpha \).
Worked example

∀x ∀y x = y isn’t logically true.

Counterexample: let A be an L_= -structure with domain {1, 2}.

Proof. Let α be an assignment over A.
Sufficient to prove (STP:) ∀x ∀y x = y is false in A under α. 

Now: |∀x∀y x = y|_α^A = T iff |∀y x = y|_β^A = T for every β differing from α at most in x.
Worked example

∀x ∀y x = y isn’t logically true.

Counterexample: let $\mathcal{A}$ be an $\mathcal{L}_-$-structure with domain $\{1, 2\}$.

Proof. Let $\alpha$ be an assignment over $\mathcal{A}$.
Sufficient to prove (STP:) $\forall x \forall y x = y$ is false in $\mathcal{A}$ under $\alpha$.

Now: $|\forall x \forall y x = y|_{\mathcal{A}}^\alpha = \text{F}$ iff $|\forall y x = y|_{\mathcal{A}}^\beta = \text{F}$ for some $\beta$

differing from $\alpha$ at most in $x$.
Worked example

$\forall x \forall y \ x = y$ isn’t logically true.

Counterexample: let $\mathcal{A}$ be an $\mathcal{L}_-$-structure with domain $\{1, 2\}$.

Proof. Let $\alpha$ be an assignment over $\mathcal{A}$.
Sufficient to prove (STP:) $\forall x \forall y \ x = y$ is false in $\mathcal{A}$ under $\alpha$.

Now: $|\forall x \forall y \ x = y|^\alpha_\mathcal{A} = F$ iff $|\forall y \ x = y|^\beta_\mathcal{A} = F$ for some $\beta$
differing from $\alpha$ at most in $x$.

STP: $|\forall y \ x = y|^\beta_\mathcal{A} = F$ for some assignment $\beta$ differing from $\alpha$
at most in $x$.
**Worked example**

∀x ∀y x = y isn’t logically true.

Counterexample: let A be an L_= -structure with domain {1, 2}.

Proof. Let α be an assignment over A.

Sufficient to prove (STP:) ∀x ∀y x = y is false in A under α.

Now: |∀x ∀y x = y|_A^α = F iff |∀y x = y|_A^β = F for some β differing from α at most in x.

STP: |∀y x = y|_A^β = F for some assignment β differing from α at most in x.

But: |∀y x = y|_A^β = T iff |x = y|_A^γ = T for every γ differing from β at most in y.
Worked example

\(\forall x \forall y x = y\) isn’t logically true.

Counterexample: let \(\mathcal{A}\) be an \(\mathcal{L}_\leq\)-structure with domain \(\{1, 2\}\).

Proof. Let \(\alpha\) be an assignment over \(\mathcal{A}\).
Sufficient to prove (STP:) \(\forall x \forall y x = y\) is false in \(\mathcal{A}\) under \(\alpha\).

Now: \(|\forall x \forall y x = y\rangle^\alpha_A = F\) iff \(|\forall y x = y\rangle^\beta_A = F\) for some \(\beta\)
differing from \(\alpha\) at most in \(x\).

STP: \(|\forall y x = y\rangle^\beta_A = F\) for some assignment \(\beta\) differing from \(\alpha\)
at most in \(x\).

But: \(|\forall y x = y\rangle^\beta_A = F\) iff \(|x = y\rangle^\gamma_A = F\) for some \(\gamma\) differing
from \(\beta\) at most in \(y\).
Worked example

\( \forall x \forall y x = y \) isn’t logically true.

Counterexample: let \( A \) be an \( \mathcal{L}_{=}-\)structure with domain \( \{1, 2\} \).

Proof. Let \( \alpha \) be an assignment over \( A \).
Sufficient to prove (STP:) \( \forall x \forall y x = y \) is false in \( A \) under \( \alpha \).

Now: \( |\forall x \forall y x = y|_A^\alpha = F \iff |\forall y x = y|_A^\beta = F \) for some \( \beta \)
differing from \( \alpha \) at most in \( x \).

STP: \( |\forall y x = y|_A^\beta = F \) for some assignment \( \beta \) differing from \( \alpha \)
at most in \( x \).

But: \( |\forall y x = y|_A^\beta = F \iff |x = y|_A^\gamma = F \) for some \( \gamma \) differing
from \( \beta \) at most in \( y \).

STP: \( |x = y|_A^\gamma = F \) for some \( \gamma \) differing from \( \beta \) in at most \( y \).
Worked example

∀x ∀y x = y isn’t logically true.

Counterexample: let A be an L_=-structure with domain {1, 2}.

Proof. Let α be an assignment over A.
Sufficient to prove (STP:) ∀x ∀y x = y is false in A under α.

Now: \(|∀x ∀y x = y|_A^α = F\) iff \(|∀ y x = y|_A^β = F\) for some β differing from α at most in x.

STP: \(|∀ y x = y|_A^β = F\) for some assignment β differing from α at most in x.

But: \(|∀ y x = y|_A^β = F\) iff \(|x = y|_A^γ = F\) for some γ differing from β at most in y.

STP: \(|x = y|_A^γ = F\) for some γ differing from α in at most x and y.
Worked example

\( \forall x \forall y x = y \) isn’t logically true.

Counterexample: let \( A \) be an \( L_{=}-\)structure with domain \( \{1, 2\} \).

Proof. Let \( \alpha \) be an assignment over \( A \).

Sufficient to prove (STP:) \( \forall x \forall y x = y \) is false in \( A \) under \( \alpha \).

Now: \( |\forall x \forall y x = y|^{\alpha \ A} = F \) iff \( |\forall y x = y|^{\beta \ A} = F \) for some \( \beta \) differing from \( \alpha \) at most in \( x \).

STP: \( |\forall y x = y|^{\beta \ A} = F \) for some assignment \( \beta \) differing from \( \alpha \) at most in \( x \).

But: \( |\forall y x = y|^{\beta \ A} = F \) iff \( |x = y|^{\gamma \ A} = F \) for some \( \gamma \) differing from \( \beta \) at most in \( y \).

STP: \( |x = y|^{\gamma \ A} = F \) for some \( \gamma \) differing from \( \alpha \) in at most \( x \) and \( y \).

So: Let \( \gamma \) assign \( x \) to 1 and \( y \) to 2 (otherwise agreeing with \( \alpha \)
### Worked example

∀x ∀y x = y isn’t logically true.

### Counterexample: let A be an \( \mathcal{L}_=- \)-structure with domain \( \{1, 2\} \).

Proof. Let \( \alpha \) be an assignment over \( A \).

Sufficient to prove (STP:) \( \forall x \forall y x = y \) is false in \( A \) under \( \alpha \).

**Now:** \( |\forall x \forall y x = y|_A^\alpha = F \) iff \( |\forall y x = y|_A^\beta = F \) for some \( \beta \) differing from \( \alpha \) at most in \( x \).

**STP:** \( |\forall y x = y|_A^\beta = F \) for some assignment \( \beta \) differing from \( \alpha \) at most in \( x \).

**But:** \( |\forall y x = y|_A^\beta = F \) iff \( |x = y|_A^\gamma = F \) for some \( \gamma \) differing from \( \beta \) at most in \( y \).

**STP:** \( |x = y|_A^\gamma = F \) for some \( \gamma \) differing from \( \alpha \) in at most \( x \) and \( y \).

**So:** Let \( \gamma \) assign \( x \) to 1 and \( y \) to 2 (otherwise agreeing with \( \alpha \))

Then \( |x|_\gamma \neq |y|_\gamma \);
8.3 Semantics

**Worked example**

\( \forall x \forall y \, x = y \) isn’t logically true.

**Counterexample:** let \( \mathcal{A} \) be an \( \mathcal{L}_{=} \)-structure with domain \( \{1, 2\} \).

**Proof.** Let \( \alpha \) be an assignment over \( \mathcal{A} \).

Sufficient to prove (STP:) \( \forall x \, \forall y \, x = y \) is false in \( \mathcal{A} \) under \( \alpha \).

**Now:** \( |\forall x \forall y \, x = y|^{\alpha}_{\mathcal{A}} = F \) iff \( |\forall y \, x = y|^{\beta}_{\mathcal{A}} = F \) for some \( \beta \) differing from \( \alpha \) at most in \( x \).

**STP:** \( |\forall y \, x = y|^{\beta}_{\mathcal{A}} = F \) for some assignment \( \beta \) differing from \( \alpha \) at most in \( x \).

**But:** \( |\forall y \, x = y|^{\beta}_{\mathcal{A}} = F \) iff \( |x = y|^{\gamma}_{\mathcal{A}} = F \) for some \( \gamma \) differing from \( \beta \) at most in \( y \).

**STP:** \( |x = y|^{\gamma}_{\mathcal{A}} = F \) for some \( \gamma \) differing from \( \alpha \) in at most \( x \) and \( y \).

**So:** Let \( \gamma \) assign \( x \) to 1 and \( y \) to 2 (otherwise agreeing with \( \alpha \))

Then \( |x|^{\gamma} \neq |y|^{\gamma} \); so \( |x = y|^{\gamma}_{\mathcal{A}} = F \).
Worked example

\(\forall x \forall y \ x = y\) isn’t logically true.

Counterexample: let \(\mathcal{A}\) be an \(\mathcal{L}_=-\)-structure with domain \(\{1, 2\}\).

Proof. Let \(\alpha\) be an assignment over \(\mathcal{A}\).

Sufficient to prove (STP:) \(\forall x \forall y \ x = y\) is false in \(\mathcal{A}\) under \(\alpha\).

Now: \(|\forall x \forall y \ x = y|_\mathcal{A}^\alpha = F\) iff \(|\forall x = y|_\mathcal{A}^\beta = F\) for some \(\beta\) differing from \(\alpha\) at most in \(x\).

STP: \(|\forall y x = y|_\mathcal{A}^\beta = F\) for some assignment \(\beta\) differing from \(\alpha\) at most in \(x\).

But: \(|\forall y x = y|_\mathcal{A}^\beta = F\) iff \(|x = y|_\mathcal{A}^\gamma = F\) for some \(\gamma\) differing from \(\beta\) at most in \(y\).

STP: \(|x = y|_\mathcal{A}^\gamma = F\) for some \(\gamma\) differing from \(\alpha\) in at most \(x\) and \(y\).

So: Let \(\gamma\) assign \(x\) to 1 and \(y\) to 2 (otherwise agreeing with \(\alpha\))

Then \(|x|_\gamma \neq |y|_\gamma\); so \(|x = y|_\mathcal{A}^\gamma = F\).

QED
Proof theory

Natural Deduction for $\mathcal{L}_=$ has the same rules as Natural Deduction for $\mathcal{L}_2$ with the addition of rules for $=$.
Natural Deduction for $\mathcal{L}_=$ has the same rules as Natural Deduction for $\mathcal{L}_2$ with the addition of rules for $\equiv$.

**$\equiv$Intro**

Any assumption of the form $t=t$ where $t$ is a constant can and must be discharged.
8.4 Proof Rules for Identity

Proof theory

Natural Deduction for $\mathcal{L}_=$ has the same rules as Natural Deduction for $\mathcal{L}_2$ with the addition of rules for $=.$

$=\text{Intro}$

Any assumption of the form $t = t$ where $t$ is a constant can and must be discharged.

A proof with an application of $=\text{Intro}$ looks like this:

$$[t = t]$$

$$\vdots$$
Proof theory

Natural Deduction for $\mathcal{L}_=$ has the same rules as Natural Deduction for $\mathcal{L}_2$ with the addition of rules for $\equiv$.

$\equiv$Intro

Any assumption of the form $t = t$ where $t$ is a constant can and must be discharged.

A proof with an application of $\equiv$Intro looks like this:

\[
[t = t] \\
\vdots
\]

Example: prove $\vdash \forall z (z = z)$
Proof theory

Natural Deduction for $\mathcal{L}_\equiv$ has the same rules as Natural Deduction for $\mathcal{L}_2$ with the addition of rules for $\equiv$.

$\equiv$Intro

Any assumption of the form $t = t$ where $t$ is a constant can and must be discharged.

A proof with an application of $\equiv$Intro looks like this:

\[
\begin{array}{c}
[t = t] \\
\vdots
\end{array}
\]

Example: prove $\vdash \forall z(z = z)$

\[
a = a
\]
Proof theory

Natural Deduction for $\mathcal{L}_=$ has the same rules as Natural Deduction for $\mathcal{L}_2$ with the addition of rules for $\equiv$.

$\equiv$Intro

Any assumption of the form $t = t$ where $t$ is a constant can and must be discharged.

A proof with an application of $\equiv$Intro looks like this:

\[
[t = t] \\
\vdots
\]

Example: prove $\vdash \forall z(z = z)$

\[
[a = a]
\]
Proof theory

Natural Deduction for $\mathcal{L}_=$ has the same rules as Natural Deduction for $\mathcal{L}_2$ with the addition of rules for $=$. 

$=\text{Intro}$

Any assumption of the form $t=t$ where $t$ is a constant can and must be discharged.

A proof with an application of $=\text{Intro}$ looks like this:

\[
\frac{[t=t]}{\vdash \forall z(z = z)}
\]

Example: prove $\vdash \forall z(z = z)$

\[
\frac{[a = a]}{\forall z(z = z)}
\]
If $s$ and $t$ are constants, the result of appending $\phi[t/v]$ to a proof of $\phi[s/v]$ and a proof of $s=t$ or $t=s$ is a proof of $\phi[t/v]$.

\[
\begin{array}{c}
\vdots \\
\phi[s/v] \\
\phi[t/v] \quad \vdots \\
\hline \\
\phi[t/v] \quad =\text{Elim} \\
\end{array}
\]

\[
\begin{array}{c}
\vdots \\
\phi[s/v] \\
\phi[t/v] \\
\hline \\
\phi[t/v] \\
\quad =\text{Elim} \\
\end{array}
\]

\[
\begin{array}{c}
\vdots \\
\phi[s/v] \\
\phi[t/v] \\
\hline \\
\phi[t/v] \\
\quad =\text{Elim} \\
\end{array}
\]

\[
\begin{array}{c}
\vdots \\
\phi[s/v] \\
\phi[t/v] \\
\hline \\
\phi[t/v] \\
\quad =\text{Elim} \\
\end{array}
\]
Worked example: prove the following.

\[ \vdash \forall x \forall y (Rxy \to (x = y \to Ryx)) \]
Worked example: prove the following.

\[ \vdash \forall x \forall y (Rxy \rightarrow (x = y \rightarrow Ryx)) \]

\[ Rab \]
Worked example: prove the following.

\[ \vdash \forall x \forall y (Rxy \rightarrow (x = y \rightarrow Ryx)) \]

\[ Rab \quad a = b \]

\[ \vdash \forall x \forall y (Rxy \rightarrow (x = y \rightarrow Ryx)) \]

\[ \vdash \forall x \forall y (Rxy \rightarrow (x = y \rightarrow Ryx)) \]

\[ \vdash \forall x \forall y (Rxy \rightarrow (x = y \rightarrow Ryx)) \]

\[ \vdash \forall x \forall y (Rxy \rightarrow (x = y \rightarrow Ryx)) \]
Worked example: prove the following.
\[ \vdash \forall x \forall y (Rxy \rightarrow (x = y \rightarrow Ryx)) \]

\[
\begin{array}{c}
Rab \\
a = b
\end{array} \quad \begin{array}{c}
Raa
\end{array}
\]

\[
\vdots \quad \vdots
\]

\[
\begin{array}{c}
\phi[s/v] \\
\phi[t/v]
\end{array} \quad s = t \quad \begin{array}{c}
=\text{Elim}
\end{array}
\]

\[
\vdots \quad \vdots
\]

\[
\begin{array}{c}
\phi[s/v] \\
\phi[t/v]
\end{array} \quad t = s \quad \begin{array}{c}
=\text{Elim}
\end{array}
\]
Worked example: prove the following.

\[ \vdash \forall x \forall y (Rxy \rightarrow (x = y \rightarrow Ryx)) \]

\[
\begin{align*}
Rab & \quad a = b \\
\hline
Raa & \quad a = b
\end{align*}
\]

\[
\begin{align*}
\phi[s/v] & \quad s = t \\
\hline
\phi[t/v] & = \text{Elim}
\end{align*}
\]

\[
\begin{align*}
\phi[s/v] & \quad t = s \\
\hline
\phi[t/v] & = \text{Elim}
\end{align*}
\]
Worked example: prove the following.

\[ \vdash \forall x \forall y (Rxy \rightarrow (x = y \rightarrow Ryx)) \]

\[
\begin{align*}
&\frac{Rab}{Raa} \quad a = b \\
\frac{Raa}{Rba} \quad a = b
\end{align*}
\]

\[
\begin{align*}
&\frac{\phi[s/v]}{\phi[t/v]} \quad s = t \quad \text{=Elim} \\
&\frac{\phi[s/v]}{\phi[t/v]} \quad t = s \quad \text{=Elim}
\end{align*}
\]
Worked example: prove the following.

\[ \vdash \forall x \forall y (Rxy \rightarrow (x = y \rightarrow Ryx)) \]

\[
\begin{align*}
Rab & \quad a = b \\
\hline
Raa & \quad a = b \\
\hline
Rba & \quad a = b \\
\hline
\end{align*}
\]

\[ a = b \rightarrow Rba \]
Worked example: prove the following.

\[ \vdash \forall x \forall y (Rxy \to (x = y \to Ryx)) \]

\[
\begin{align*}
Rab & \quad [a = b] \\
\hline
\quad Raa & \quad a = b \\
\hline
\quad Rba & \\
\hline
\quad a = b \to Rba
\end{align*}
\]
Worked example: prove the following.
\[ \vdash \forall x \forall y (Rxy \rightarrow (x = y \rightarrow Ryx)) \]

\[
\begin{align*}
Rab & \quad [a = b] \\
\hline
Raa & \quad [a = b] \\
\hline
Rba & \\
\hline
a = b \rightarrow Rba
\end{align*}
\]
Worked example: prove the following.

\[ \vdash \forall x \forall y \ (Rxy \rightarrow (x = y \rightarrow Ryx)) \]

\[
\begin{align*}
Rab & \quad [a = b] \\
Raa & \quad [a = b] \\
\frac{}{Rba} \\
\frac{Rba}{a = b \rightarrow Rba} \\
Rab & \rightarrow (a = b \rightarrow Rba)
\end{align*}
\]
Worked example: prove the following.

\[ \vdash \forall x \forall y (Rxy \rightarrow (x = y \rightarrow Ryx)) \]

\[
\begin{array}{c}
[ Rab ] [ a = b ] \\
\hline
Raa [ a = b ] \\
\hline
Rba \\
\hline
a = b \rightarrow Rba \\
\hline
Rab \rightarrow (a = b \rightarrow Rba)
\end{array}
\]
Worked example: prove the following.

\[ \vdash \forall x \forall y (Rxy \rightarrow (x = y \rightarrow Ryx)) \]

\[
\begin{array}{c}
\text{[Rab]} \quad \text{[a = b]} \\
\hline
Raa \quad \text{[a = b]} \\
\hline
Rba \\
\hline
a = b \rightarrow Rba \\
\hline
Rab \rightarrow (a = b \rightarrow Rba) \\
\hline
\forall y (Ray \rightarrow (a = y \rightarrow Rya))
\end{array}
\]
Worked example: prove the following.

\[ \vdash \forall x \forall y (Rxy \to (x = y \to Ryx)) \]

\[
\begin{array}{c}
\begin{array}{c}
[Rab] \\
[a = b]
\end{array}
\hline
Raa \\
[a = b]
\end{array}
\begin{array}{c}
Rba \\
[a = b]
\hline
a = b \to Rba \\
Rab \to (a = b \to Rba)
\end{array}
\]

\[
\forall y (Ray \to (a = y \to Rya))
\]

\[
\forall x \forall y (Rxy \to (x = y \to Ryx))
\]
Adequacy

Soundness and Completeness still hold.
Adequacy

Soundness and Completeness still hold.

Let $\Gamma$ be a set of $\mathcal{L}_=$-sentences and $\phi$ an $\mathcal{L}_=$-sentence.

**Theorem (adequacy)**

$\Gamma \vdash \phi$ if and only if $\Gamma \models \phi$. 
Formalisation with identity

Using = one can formalise ‘is [identical to]’ in English.

**Formalise:**

William II is Wilhelm II.

**Formalisation:** $a = b$.

**Dictionary:** $a$: William II. $b$: Wilhelm II.
Formalisation with identity

Using = one can formalise ‘is [identical to]’ in English.

Formalise:
William II is Wilhelm II.

Formalisation: \( a = b \).
Dictionary: \( a \): William II. \( b \): Wilhelm II.

Note: don’t confuse the ‘is’ of identity with the ‘is’ of predication.
Formalisation with identity

Using = one can formalise ‘is [identical to]’ in English.

**Formalise:**
William II is Wilhelm II.

**Formalisation:**  \( a = b. \)
**Dictionary:**  \( a: \) William II.  \( b: \) Wilhelm II.

**Note:** don’t confuse the ‘is’ of identity with the ‘is’ of predication.

**Formalise:**
Wilhelm II is an emperor.
Formalisation with identity

Using $=$ one can formalise ‘is [identical to]’ in English.

**Formalise:**

William II is Wilhelm II.

Formalisation: $a = b$.


**Note:** don’t confuse the ‘is’ of identity with the ‘is’ of predication.

**Formalise:**

Wilhelm II is an emperor.

Formalisation: $Ea$.

Dictionary: $a$: Wilhelm. $E$: ... is an emperor.
Formalisation with identity

Using = one can formalise ‘is [identical to]’ in English.

Formalise:
William II is Wilhelm II.

Formalisation: $a = b$.

Note: don’t confuse the ‘is’ of identity with the ‘is’ of predication.

Formalise:
Wilhelm II is an emperor.

Formalisation: $Ea$.
Dictionary: $a$: Wilhelm. $E$: ... is an emperor.

Here ‘is’ forms part of the predicate ‘is an emperor.’
Identity can also be used to formalise numerical quantifiers.
Identity can also be used to formalise numerical quantifiers.

Dictionary: $P$: ... is a perfect being.

**Formalise**

(1) There are at least two perfect beings.
Identity can also be used to formalise numerical quantifiers.

Dictionary: $P$: … is a perfect being.

Formalise

(1) There are at least two perfect beings.
Incorrect formalisation: $\exists x \exists y (Px \land Py)$.
Identity can also be used to formalise numerical quantifiers.

Dictionary: $P$: ... is a perfect being.

**Formalise**

(1) There are at least two perfect beings.
Incorrect formalisation: $\exists x \exists y (Px \land Py)$.
Correct formalisation: $\exists x \exists y (Px \land Py \land \neg x = y)$.
Identity can also be used to formalise numerical quantifiers.

Dictionary: \( P: \ldots \text{ is a perfect being.} \)

**Formalise**

(1) There are at least two perfect beings.

Incorrect formalisation: \( \exists x \exists y (Px \land Py) \).

Correct formalisation: \( \exists x \forall y (Px \land Py \land \neg x = y) \).

(2) There is at most one perfect being.
Identity can also be used to formalise numerical quantifiers.

**Dictionary:** $P$: ... is a perfect being.

**Formalise**

(1) There are at least two perfect beings.
Incorrect formalisation: $\exists x \exists y (Px \land Py)$.
Correct formalisation: $\exists x \exists y (Px \land Py \land \neg x = y)$.

(2) There is at most one perfect being.
Formalisation: $\neg \exists x \exists y (Px \land Py \land \neg x = y)$. 
Identity can also be used to formalise numerical quantifiers.

Dictionary: \( P: \ldots \) is a perfect being.

### Formalise

(1) There are at least two perfect beings.
Incorrect formalisation: \( \exists x \exists y (Px \land Py) \).
Correct formalisation: \( \exists x \exists y (Px \land Py \land \neg x = y) \).

(2) There is at most one perfect being.
Formalisation: \( \neg \exists x \exists y (Px \land Py \land \neg x = y) \).
Alternative formalisation: \( \forall x \forall y ((Px \land Py) \rightarrow x = y) \).
Identity can also be used to formalise numerical quantifiers.

Dictionary: \( P: \ldots \text{ is a perfect being.} \)

### Formalise

1. There are at least two perfect beings.
   Incorrect formalisation: \( \exists x \exists y (Px \land Py) \).
   Correct formalisation: \( \exists x \exists y (Px \land Py \land \neg x = y) \).

2. There is at most one perfect being.
   Formalisation: \( \neg \exists x \exists y (Px \land Py \land \neg x = y) \).
   Alternative formalisation: \( \forall x \forall y ((Px \land Py) \rightarrow x = y) \).

3. There is exactly one perfect being.
Identity can also be used to formalise numerical quantifiers.

Dictionary: $P$: ... is a perfect being.

**Formalise**

(1) There are at least two perfect beings.
Incorrect formalisation: $\exists x \exists y (P_x \land P_y)$.
Correct formalisation: $\exists x \exists y (P_x \land P_y \land \neg x = y)$.

(2) There is at most one perfect being.
Formalisation: $\neg \exists x \exists y (P_x \land P_y \land \neg x = y)$.
Alternative formalisation: $\forall x \forall y ((P_x \land P_y) \rightarrow x = y)$.

(3) There is exactly one perfect being.
Formalisation: $\exists x P_x \land \forall x \forall y ((P_x \land P_y) \rightarrow x = y)$.
Identity can also be used to formalise numerical quantifiers.

Dictionary: $P$: \ldots is a perfect being.

**Formalise**

(1) There are at least two perfect beings.
Incorrect formalisation: $\exists x \exists y (P_x \land P_y)$.
Correct formalisation: $\exists x \exists y (P_x \land P_y \land \neg x = y)$.

(2) There is at most one perfect being.
Formalisation: $\neg \exists x \exists y (P_x \land P_y \land \neg x = y)$.
Alternative formalisation: $\forall x \forall y ((P_x \land P_y) \rightarrow x = y)$.

(3) There is exactly one perfect being.
Formalisation: $\exists x P_x \land \forall x \forall y ((P_x \land P_y) \rightarrow x = y)$.
Alternative formalisation: $\exists x (P_x \land \forall y (P_y \rightarrow y = x))$. 
Definite descriptions

Examples of definite descriptions:

- ‘the Queen’
- ‘Bellerophon’s winged horse’
- ‘the author of Ulysses’
Definite descriptions

**Examples of definite descriptions:**

- ‘the Queen’
- ‘Bellerophon’s winged horse’
- ‘the author of Ulysses’

In $\mathcal{L}_2$: the best we can do is to formalise definite descriptions as constants.
Definite descriptions

Examples of definite descriptions:

- ‘the Queen’
- ‘Bellerophon’s winged horse’
- ‘the author of Ulysses’

In $\mathcal{L}_2$: the best we can do is to formalise definite descriptions as constants.

But this isn’t perfect...
Example

Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.
<table>
<thead>
<tr>
<th>Example</th>
<th>Not valid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.</td>
<td>Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.</td>
</tr>
</tbody>
</table>
### Example

<table>
<thead>
<tr>
<th>Not valid</th>
</tr>
</thead>
</table>

Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.

The obvious formalisation with constants is valid.
<table>
<thead>
<tr>
<th>Example</th>
<th>Not valid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bellerophon’s winged horse isn’t real; so there is something</td>
<td></td>
</tr>
<tr>
<td>that is Bellerophon’s winged horse.</td>
<td></td>
</tr>
</tbody>
</table>

The obvious formalisation with constants is valid.

Formalisation: premiss: $\neg Rb$. Conclusion: $\exists x (x = b)$.

Dictionary: $R$: ...is real. $b$: Bellerophon’s winged horse.
Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.

The obvious formalisation with constants is valid.

Formalisation: premiss: $\neg Rb$. Conclusion: $\exists x (x = b)$.

Dictionary: $R$: ...is real. $b$: Bellerophon’s winged horse.

\[ b = b \]
### Example

<table>
<thead>
<tr>
<th>Not valid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.</td>
</tr>
</tbody>
</table>

The obvious formalisation with constants is valid.

| Formalisation: premiss: \( \neg Rb \). Conclusion: \( \exists x (x = b) \). |
| Dictionary: \( R \): ...is real. \( b \): Bellerophon’s winged horse. |

\[ [b = b] \]
Example

Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.

The obvious formalisation with constants is valid.

Formalisation: premiss: \( \neg Rb \). Conclusion: \( \exists x (x = b) \).

Dictionary: \( R \): ...is real. \( b \): Bellerophon’s winged horse.

\[
\frac{[b = b]}{\exists x (x = b)}
\]
### Example

Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.

The obvious formalisation with constants is valid.

**Formalisation:**
- Premiss: \( \neg Rb \).
- Conclusion: \( \exists x (x = b) \).

**Dictionary:**
- \( R \): ...is real.
- \( b \): Bellerophon’s winged horse.

\[
\frac{b = b}{\exists x (x = b)}
\]

(In fact: the conclusion is a logical truth.)
### Example

Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.

The obvious formalisation with constants is valid.

**Formalisation:**
- **Premiss:** \( \neg Rb \).
- **Conclusion:** \( \exists x (x = b) \).

**Dictionary:**
- \( R \): ... is real.
- \( b \): Bellerophon’s winged horse.

\[
\begin{align*}
[b = b] \\
\hline
\exists x (x = b)
\end{align*}
\]

(In fact: the conclusion is a logical truth.)

**Source of the trouble:**
- \( L_\equiv \)-constants always refer to an object in a \( L_\equiv \)-structure.
- definite descriptions may fail to pick out a unique object.
Russell’s theory of descriptions.

There’s a better way to formalise definite descriptions in $\mathcal{L}_{=}$.
Russell’s theory of descriptions.

There’s a better way to formalise definite descriptions in $\mathcal{L}_\equiv$.

Formalise:
The author of Ulysses wrote Dubliners.
Russell’s theory of descriptions.

There’s a better way to formalise definite descriptions in $L_\equiv$.

Formalise:

The author of Ulysses wrote Dubliners.

Russell analyses this as the conjunction of two claims.

(i) There is exactly one author of Ulysses
(ii) and it wrote Dubliners.
Russell’s theory of descriptions.

There’s a better way to formalise definite descriptions in $L_\subseteq$.

Formalise:
The author of Ulysses wrote Dubliners.

Russell analyses this as the conjunction of two claims.

(i) There is exactly one author of Ulysses
(ii) and it wrote Dubliners.

Dictionary: A: ...is an author of Ulysses.
W: ...wrote Dubliners.
Russell’s theory of descriptions.

There’s a better way to formalise definite descriptions in \( L_{\equiv} \).

**Formalise:**

The author of Ulysses wrote Dubliners.

Russell analyses this as the conjunction of two claims.

\[(i)\] There is exactly one author of Ulysses
\n\[(ii)\] and it wrote Dubliners.

**Dictionary:**
A: …is an author of Ulysses.
W: …wrote Dubliners.

**Formalisation:**
\[\exists x (Ax \land \forall y (Ay \rightarrow y = x))\]
Russell’s theory of descriptions.

There’s a better way to formalise definite descriptions in $L_\subseteq$.

**Formalise:**

The author of Ulysses wrote Dubliners.

Russell analyses this as the conjunction of two claims.

(i) There is exactly one author of Ulysses

(ii) and it wrote Dubliners.

Dictionary: A: ...is an author of Ulysses.
W: ...wrote Dubliners.

Formalisation: $\exists x (Ax \land \forall y (Ay \rightarrow y = x) \land Wx)$
8.4 Uses of identity

Formalise:
Bellerophon’s winged horse isn’t real.

R: ... is real. B: ... is a winged horse belonging to Bellerophon.
Formalise:

Bellerophon’s winged horse isn’t real.

\( R: \ldots \) is real.  \( B: \ldots \) is a winged horse belonging to Bellerophon.

On Russell’s view this can have two readings.
Formalise:

Bellerophon’s winged horse isn’t real.

\( R: \ldots \text{is real. } B: \ldots \text{is a winged horse belonging to Bellerophon.} \)

On Russell’s view this can have two readings.

Paraphrase 1: (i) there is exactly one winged horse belonging to Bellerophon and (ii) it is not real.
Formalise:

Bellerophon’s winged horse isn’t real.

\[ R: \ldots \text{is real. } B: \ldots \text{is a winged horse belonging to Bellerophon.} \]

On Russell’s view this can have two readings.

Paraphrase 1: (i) there is exactly one winged horse belonging to Bellerophon and (ii) it is not real.

Formalisation 1: \[ \exists x (Bx \land \forall y (By \rightarrow y = x) \land \neg Rx) . \]
**Formalise:**

Bellerophon’s winged horse isn’t real.

\[ R: \ldots \text{is real. } B: \ldots \text{is a winged horse belonging to Bellerophon.} \]

On Russell’s view this can have two readings.

**Paraphrase 1:** (i) there is exactly one winged horse belonging to Bellerophon and (ii) it is not real.

**Formalisation 1:** \[ \exists x (Bx \land \forall y (By \rightarrow y = x) \land \neg Rx) \].

Dubious: this is true only if there are non-real things.
Formalise:

Bellerophon’s winged horse isn’t real.

*R*: ...is real.  *B*: ...is a winged horse belonging to Bellerophon.

On Russell’s view this can have two readings.

Paraphrase 1: (i) there is exactly one winged horse belonging to Bellerophon and (ii) it is not real.

Formalisation 1: $\exists x (Bx \land \forall y (By \rightarrow y = x) \land \neg Rx)$.

Dubious: this is true only if there are non-real things.

Paraphrase 2: It’s not the case that (i) there is exactly one winged horse belonging to Bellerophon and (ii) it is real).
8.4 Uses of identity

**Formalise:**

Bellerophon’s winged horse isn’t real.

*R: ...is real.  B: ...is a winged horse belonging to Bellerophon.*

On Russell’s view this can have two readings.

Paraphrase 1: (i) there is exactly one winged horse belonging to Bellerophon and (ii) it is not real.

Formalisation 1: \( \exists x (Bx \land \forall y(By \rightarrow y = x) \land \neg Rx) \).

Dubious: this is true only if there are non-real things.

Paraphrase 2: It’s not the case that (i) there is exactly one winged horse belonging to Bellerophon and (ii) it is real.

Formalisation 2: \( \neg \exists x (Bx \land \forall y(By \rightarrow y = x) \land Rx) \).
Example

Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.
### Example

<table>
<thead>
<tr>
<th>Not valid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.</td>
</tr>
</tbody>
</table>

We can capture its non-validity by using the second formalisation of the premiss.
### Example

Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.

We can capture its non-validity by using the second formalisation of the premiss.

<table>
<thead>
<tr>
<th>Dictionary:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$R$:</td>
<td>...is real.</td>
</tr>
<tr>
<td>$B$:</td>
<td>...is a winged horse belonging to Bellerophon.</td>
</tr>
</tbody>
</table>
**Example**

Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.

We can capture its non-validity by using the second formalisation of the premiss.

**Dictionary:**

- $R$: ...is real.
- $B$: ...is a winged horse belonging to Bellerophon.

**Formalisation**

Premiss: $\neg \exists x (Bx \land \forall y (By \rightarrow y = x) \land Rx)$. 
### Example

Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.

We can capture its non-validity by using the second formalisation of the premiss.

**Dictionary:**
- \( R \): ...is real.
- \( B \): ...is a winged horse belonging to Bellerophon.

### Formalisation

**Premiss:** \( \neg \exists x \left( Bx \land \forall y(By \rightarrow y = x) \land Rx \right) \).

**Conclusion:** \( \exists x Bx \).
Example

Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.

We can capture its non-validity by using the second formalisation of the premiss.

Dictionary:  
- $R$: ...is real.
- $B$: ...is a winged horse belonging to Bellerophon.

Formalisation

Premiss: $\neg \exists x (Bx \land \forall y (By \rightarrow y = x) \land Rx)$.

Conclusion: $\exists x Bx$.

The structure $A$ is a counterexample to this argument.

$$D_A = \{x : x \text{ is a horse}\}; \; |B|_A = \emptyset.$$
8.4 Uses of identity

**Example**

Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.

We can capture its non-validity by using the second formalisation of the premiss.

**Dictionary:**  
- \( R \): ...is real.  
- \( B \): ...is a winged horse belonging to Bellerophon.

**Formalisation**

**Premiss:**  
\[ \neg \exists x (Bx \land \forall y (By \rightarrow y = x) \land Rx). \]

**Conclusion:**  
\[ \exists x Bx. \]

The structure \( \mathcal{A} \) is a counterexample to this argument.

\[ D_{\mathcal{A}} = \{ x : x \text{ is a horse}\}; \ |B|_{\mathcal{A}} = \emptyset. \]
**Example**

Bellerophon’s winged horse isn’t real; so there is something that is Bellerophon’s winged horse.

We can capture its non-validity by using the second formalisation of the premiss.

Dictionary:  
- \(R\): …is real.  
- \(B\): …is a winged horse belonging to Bellerophon.

**Formalisation**

Premiss: \(\neg \exists x (Bx \land \forall y (By \rightarrow y = x) \land Rx)\).

Conclusion: \(\exists x Bx\).

The structure \(\mathcal{A}\) is a counterexample to this argument.

\[D_\mathcal{A} = \{x : x \text{ is a horse}\}; \ |B|_\mathcal{A} = \emptyset.\]

(It doesn’t matter what the extension of \(R\) is here.)
Multiple descriptions

We deal with these much like multiple quantifiers.

**Formalise**

The author of Ulysses likes the author of the Odyssey

Dictionary: U: ...is an author of Ulysses
O: ...is an author of the Odyssey. L: ...likes ...
Multiple descriptions

We deal with these much like multiple quantifiers.

**Formalise**

The author of Ulysses likes the author of the Odyssey

Dictionary:  
\[ U: \ldots \text{is an author of Ulysses} \]
\[ O: \ldots \text{is an author of the Odyssey. } L: \ldots \text{likes } \ldots \]

It’s helpful to break this into two steps.

**Partial formalisation:**

\[ \exists x_1 (U x_1 \land \forall y_1 (U y_1 \rightarrow y_1 = x_1) \land x_1 \text{ likes the author of the Odyssey}) \]
Multiple descriptions

We deal with these much like multiple quantifiers.

Formalise

The author of Ulysses likes the author of the Odyssey

Dictionary: U: \ldots is an author of Ulysses
O: \ldots is an author of the Odyssey. L: \ldots likes \ldots

It’s helpful to break this into two steps.

Partial formalisation:

\[ \exists x_1 (U x_1 \land \forall y_1 (U y_1 \rightarrow y_1 = x_1) \land x_1 \text{ likes the author of the Odyssey}) \]

It remains to formalise ‘\( x_1 \) likes the author of the Odyssey’. 
<table>
<thead>
<tr>
<th>x₁ likes the author of the Odyssey</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paraphrase: the author of the Odyssey is liked by x₁.</td>
</tr>
</tbody>
</table>
$x_1$ likes the author of the Odyssey

Paraphrase: the author of the Odyssey is liked by $x_1$.

Formalisation: $\exists x_2 (Ox_2 \land \forall y_2 (Oy_2 \rightarrow y_2 = x_2) \land Lx_1 x_2)$. 
\( x_1 \) **likes the author of the Odyssey**

Paraphrase: the author of the Odyssey is liked by \( x_1 \).

Formalisation: \( \exists x_2 \left( Ox_2 \land \forall y_2 (Oy_2 \rightarrow y_2 = x_2) \land Lx_1 x_2 \right) \).

Finally, we put this together with what we had before.

**The author of Ulysses likes the author of the Odyssey**

\[
\exists x_1 \left( Ux_1 \land \forall y_1 (Uy_1 \rightarrow y_1 = x_1) \land x_1 \text{ likes the author of the Odyssey} \right).
\]
$x_1$ likes the author of the Odyssey

Paraphrase: the author of the Odyssey is liked by $x_1$.

Formalisation: $\exists x_2 (Ox_2 \land \forall y_2 (Oy_2 \rightarrow y_2 = x_2) \land Lx_1x_2)$.

Finally, we put this together with what we had before.

The author of Ulysses likes the author of the Odyssey

$\exists x_1 (Ux_1 \land \forall y_1 (Uy_1 \rightarrow y_1 = x_1) \land x_1 \text{ likes the author of the Odyssey})$. 
$x_1$ likes the author of the Odyssey

Paraphrase: the author of the Odyssey is liked by $x_1$.

Formalisation: $\exists x_2 (Ox_2 \land \forall y_2 (Oy_2 \rightarrow y_2 = x_2) \land Lx_1x_2)$.

Finally, we put this together with what we had before.

The author of Ulysses likes the author of the Odyssey

$\exists x_1 (Ux_1 \land \forall y_1 (Uy_1 \rightarrow y_1 = x_1)$

$\land x_1$ likes the author of the Odyssey).

$\exists x_1 (Ux_1 \land \forall y_1 (Uy_1 \rightarrow y_1 = x_1)$

$\land \exists x_2 (Ox_2 \land \forall y_2 (Oy_2 \rightarrow y_2 = x_2) \land Lx_1x_2))$. 
\( x_1 \) likes the author of the Odyssey

Paraphrase: the author of the Odyssey is liked by \( x_1 \).

Formalisation: \( \exists x_2 (Ox_2 \land \forall y_2 (Oy_2 \rightarrow y_2 = x_2) \land Lx_1 x_2) \).

Finally, we put this together with what we had before.

The author of Ulysses likes the author of the Odyssey

\[ \exists x_1 (Ux_1 \land \forall y_1 (Uy_1 \rightarrow y_1 = x_1) \land x_1 \text{ likes the author of the Odyssey}) \).

\[ \exists x_1 (Ux_1 \land \forall y_1 (Uy_1 \rightarrow y_1 = x_1) \land \exists x_2 (Ox_2 \land \forall y_2 (Oy_2 \rightarrow y_2 = x_2) \land Lx_1 x_2)) \). \]
$x_1$ likes the author of the Odyssey

Paraphrase: the author of the Odyssey is liked by $x_1$.

Formalisation: $\exists x_2 (Ox_2 \land \forall y_2 (Oy_2 \rightarrow y_2 = x_2) \land Lx_1 x_2)$.

Finally, we put this together with what we had before.

The author of Ulysses likes the author of the Odyssey

$\exists x_1 (Ux_1 \land \forall y_1 (Uy_1 \rightarrow y_1 = x_1) \land x_1$ likes the author of the Odyssey).

$\exists x_1 (Ux_1 \land \forall y_1 (Uy_1 \rightarrow y_1 = x_1) \land \exists x_2 (Ox_2 \land \forall y_2 (Oy_2 \rightarrow y_2 = x_2) \land Lx_1 x_2))$. 
8.4 Uses of identity

**x₁ likes the author of the Odyssey**

Paraphrase: the author of the Odyssey is liked by $x₁$.

Formalisation: $\exists x₂(Ox₂ \land \forall y₂(Oy₂ \rightarrow y₂ = x₂) \land Lx₁x₂)$. 

Finally, we put this together with what we had before.

**The author of Ulysses likes the author of the Odyssey**

$\exists x₁(Ux₁ \land \forall y₁(Uy₁ \rightarrow y₁ = x₁)$

$\land x₁ \text{ likes the author of the Odyssey}$.

$\exists x₁(Ux₁ \land \forall y₁(Uy₁ \rightarrow y₁ = x₁)$

$\land \exists x₂(Ox₂ \land \forall y₂(Oy₂ \rightarrow y₂ = x₂) \land Lx₁x₂))$. 
Logical constants

\(\neg, \land, \lor, \rightarrow, \leftrightarrow, \forall, \exists\) and = are our only logical expressions.

This raises two questions:

Q1: What's special about these expressions?

A1: Alfred Tarski proposes to analyse topic neutrality in terms of 'permutation invariance'. Roughly, logical expressions are the ones whose meaning is insensitive to which object is which. See Tarski 'What are Logical Notions?' History and Philosophy of Logic 7, 143–154.
Logical constants

\neg, \land, \lor, \rightarrow, \leftrightarrow, \forall, \exists \text{ and } = \text{ are our only logical expressions.}

This raises two questions:
Logical constants

\(\neg, \land, \lor, \to, \leftrightarrow, \forall, \exists\) and \(=\) are our only logical expressions. This raises two questions:

**Q1** What’s special about these expressions?
Logical constants

\( \neg, \land, \lor, \rightarrow, \leftrightarrow, \forall, \exists \) and \( = \) are our only logical expressions.

This raises two questions:

**Q1** What’s special about these expressions?

**A1** Alfred Tarski proposes to analyse topic neutrality in terms of ‘permutation invariance’
Logical constants

\( \neg, \land, \lor, \rightarrow, \leftrightarrow, \forall, \exists \) and \( = \) are our only logical expressions. 45

This raises two questions:

Q1 What’s special about these expressions?

A1 Alfred Tarski proposes to analyse topic neutrality in terms of ‘permutation invariance’

- Roughly: logical expressions are the ones whose meaning is insensitive to which object is which.
Logical constants

¬, ∧, ∨, →, ↔, ∀, ∃ and = are our only logical expressions.

This raises two questions:

Q1 What’s special about these expressions?
A1 Alfred Tarski proposes to analyse topic neutrality in terms of ‘permutation invariance’
  - Roughly: logical expressions are the ones whose meaning is insensitive to which object is which.
  - See Tarski ‘What are Logical Notions?’ *History and Philosophy of Logic* 7, 143–154.
Q2 What happens if we add more logical constants?
Q2  What happens if we add more logical constants?
A2  This is the business of philosophical logic.
Q2  What happens if we add more logical constants?
A2  This is the business of philosophical logic.

<table>
<thead>
<tr>
<th>Extension of $\mathcal{L}_2$</th>
<th>New logical expressions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generalised quantifiers</td>
<td>more than half</td>
</tr>
<tr>
<td></td>
<td>infinitely many, etc.</td>
</tr>
<tr>
<td>Modal logic</td>
<td>It is necessarily the case that</td>
</tr>
<tr>
<td></td>
<td>It is possibly the case that</td>
</tr>
<tr>
<td>Deontic logic</td>
<td>It is obligatory that</td>
</tr>
<tr>
<td></td>
<td>It is permissible that</td>
</tr>
</tbody>
</table>
**Q2** What happens if we add more logical constants?

**A2** This is the business of philosophical logic.

<table>
<thead>
<tr>
<th>Extension of $\mathcal{L}_2$</th>
<th>New logical expressions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generalised quantifiers</td>
<td>more than half</td>
</tr>
<tr>
<td></td>
<td>infinitely many, etc.</td>
</tr>
<tr>
<td>Modal logic</td>
<td>It is necessarily the case that</td>
</tr>
<tr>
<td></td>
<td>It is possibly the case that</td>
</tr>
<tr>
<td>Deontic logic</td>
<td>It is obligatory that</td>
</tr>
<tr>
<td></td>
<td>It is permissible that</td>
</tr>
</tbody>
</table>

See the finals paper 127: Philosophical Logic.
fin