There is a real world with real structure. The program of mind has been trained on vast interaction with this world and so contains code that reflects the structure of the world and knows how to exploit it. This code contains representations of real objects in the world and represents the interactions of real objects.

You exploit the structure of the world to make decisions and take actions. Where you draw the line on categories, what constitutes a single object or a single class of objects for you, is determined by the program of your mind, which does the classification. This classification is not random but reflects a compact description of the world, and in particular a description useful for exploiting the structure of the world.

– Eric B. Baum [2004]

Knowledge Graphs and Ontologies

- Is there a flexible way to represent relations?
- How can knowledge/data bases be made to interoperate?

Choosing Individuals and Relations

How to represent: "Pen #5 is red."
suppose the pen is denoted by the constant $pen5$.

Choosing Individuals and Relations

How to represent: "Pen #5 is red."
suppose the pen is denoted by the constant *pen*5.

- $red(pen5)$. It's easy to ask "What's red?"
  Can't ask "what is the color of *pen*5?"

Choosing Individuals and Relations

How to represent: "Pen #5 is red."
suppose the pen is denoted by the constant *pen*5.

- $red(pen5)$. It's easy to ask "What's red?"
  Can't ask "what is the color of *pen*5?"
- $color(pen5, red)$. It's easy to ask "What's red?"
  It's easy to ask "What is the color of *pen*5?"
  Can't ask "What property of *pen*5 has value *red*?"

Choosing Individuals and Relations

How to represent: "Pen #5 is red."
suppose the pen is denoted by the constant *pen*5.

- $red(pen5)$. It's easy to ask "What's red?"
  Can't ask "what is the color of *pen*5?"
- $color(pen5, red)$. It's easy to ask "What's red?"
  It's easy to ask "What is the color of *pen*5?"
  Can't ask "What property of *pen*5 has value *red*?"
- $prop(pen5, color, red)$. It's easy to ask all these questions.

Choosing Individuals and Relations

How to represent: "Pen #5 is red."
suppose the pen is denoted by the constant *pen*5.

- *red*(*pen*5). It's easy to ask "What's red?"
  Can't ask "what is the color of *pen*5?"
- *color*(*pen*5, *red*). It's easy to ask "What's red?"
  It's easy to ask "What is the color of *pen*5?"
  Can't ask "What property of *pen*5 has value *red*?"
- *prop*(*pen*5, *color*, *red*). It's easy to ask all these questions.

prop(*Individual*, *Property*, *Value*) is the only relation needed:
called individual-property-value representation
or triple representation

Universality of *prop*

To represent "a is a parcel"

Universality of *prop*

To represent "a is a parcel"

- $prop(a, type, parcel)$, where *type* is a special property
  Then *parcel* is a class.

Universality of *prop*

To represent "a is a parcel"

- $prop(a, type, parcel)$, where $type$ is a special property
  Then $parcel$ is a class.
- $prop(a, parcel, true)$, where $parcel$ is a Boolean property
  Here parcel is the characteristic function of the class.

Reification

- To represent $scheduled(cs422, 2, 1030, cc208)$. "section 2 of course $cs422$ is scheduled at 10:30 in room $cc208$."

Reification

- To represent $scheduled(cs422, 2, 1030, cc208)$. "section 2 of course $cs422$ is scheduled at 10:30 in room $cc208$."
- Let $b123$ name the booking:

  $prop(b123, course, cs422).$

  $prop(b123, section, 2).$

  $prop(b123, time, 1030).$

  $prop(b123, room, cc208).$
- We have reified the booking.
- Reify means: to make into an individual.

Reification

- To represent $scheduled(cs422, 2, 1030, cc208)$. "section 2 of course $cs422$ is scheduled at 10:30 in room $cc208$."
- Let $b123$ name the booking:

  $prop(b123, course, cs422).$

  $prop(b123, section, 2).$

  $prop(b123, time, 1030).$

  $prop(b123, room, cc208).$
- We have reified the booking.
- Reify means: to make into an individual.
- What if we want to add the year?

Reification

- To represent $scheduled(cs422, 2, 1030, cc208)$. "section 2 of course $cs422$ is scheduled at 10:30 in room $cc208$."
- Let $b123$ name the booking:

  $prop(b123, course, cs422).$

  $prop(b123, section, 2).$

  $prop(b123, time, 1030).$

  $prop(b123, room, cc208).$
- We have reified the booking.
- Reify means: to make into an individual.
- What if we want to add the year?
- What if we want to add the instructor?

Semantic Networks / Knowledge Graphs

When you only have one relation, *prop*, it can be omitted without loss of information.
Logic:

$$prop(Individual, Property, Value) \qquad \text{or}$$

$$rdf(Individual, Property, Value)$$

Semantic Networks / Knowledge Graphs

When you only have one relation, *prop*, it can be omitted without loss of information.

Logic:

$$prop(Individual, Property, Value) \qquad \text{or}$$

$$rdf(Individual, Property, Value)$$

triple:

$$\langle Individual, Property, Value \rangle$$

Semantic Networks / Knowledge Graphs

When you only have one relation, *prop*, it can be omitted without loss of information.

Logic:

$prop(Individual, Property, Value)$ or

$rdf(Individual, Property, Value)$

triple:

$\langle Individual, Property, Value \rangle$

simple sentence:

Individual Property Value.

Subject Predicate Object.

Semantic Networks / Knowledge Graphs

When you only have one relation, *prop*, it can be omitted without loss of information.

Logic:

$prop(Individual, Property, Value)$ or

$rdf(Individual, Property, Value)$

triple:

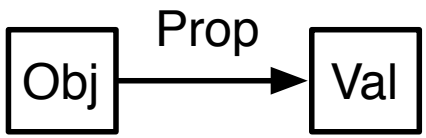
$\langle Individual, Property, Value \rangle$

simple sentence:

Individual Property Value.

Subject Predicate Object.

graphically:



Triples are universal representations of relations

All relations can be represented in terms of triples:

	...	P_j	...
	...	...	...
r_i	...	v_{ij}	...
	...	...	...

can be represented as

Triples are universal representations of relations

All relations can be represented in terms of triples:

	...	P_j	...
	...	...	...
r_i	...	v_{ij}	...
	...	...	...

can be represented as the triple $\langle r_i, P_j, v_{ij} \rangle$.

- r_i is either a primary key or a reified entity.

Triples are universal representations of relations

All relations can be represented in terms of triples:

	...	P_j	...
	...	...	...
r_i	...	v_{ij}	...
	...	...	...

can be represented as the triple $\langle r_i, P_j, v_{ij} \rangle$.

- r_i is either a primary key or a reified entity.
- Examples of reified entities: a booking, a marriage, flight number, transaction number, FIFA Wold Cup Final 2026.

Triples are universal representations of relations

All relations can be represented in terms of triples:

	...	P_j	...
	...	...	...
r_i	...	v_{ij}	...
	...	...	...

can be represented as the triple $\langle r_i, P_j, v_{ij} \rangle$.

- r_i is either a primary key or a reified entity.
- Examples of reified entities: a booking, a marriage, flight number, transaction number, FIFA Wold Cup Final 2026.

$prop(Entity, Property, Value)$ is the only relation needed: $\langle Entity, Property, Value \rangle$ triples, semantic network, entity relationship model, knowledge graphs, ...

Individuals and Identifiers

- A uniform resource identifier (URI) is a unique name that can be used to identify anything.
- A resource is anything that can be named.

Individuals and Identifiers

- A uniform resource identifier (URI) is a unique name that can be used to identify anything.
- A resource is anything that can be named.
- The modern unicode extension, is internationalized resource identifier (IRI)

Individuals and Identifiers

- A uniform resource identifier (URI) is a unique name that can be used to identify anything.
- A resource is anything that can be named.
- The modern unicode extension, is internationalized resource identifier (IRI)
- A IRI typically has the form of a uniform resource locator (URL), a web address, typically staring with `http://` or `https://`, because URLs are unique.

Individuals and Identifiers

- A uniform resource identifier (URI) is a unique name that can be used to identify anything.
- A resource is anything that can be named.
- The modern unicode extension, is internationalized resource identifier (IRI)
- A IRI typically has the form of a uniform resource locator (URL), a web address, typically staring with `http://` or `https://`, because URLs are unique.
- The IRI denotes the entity, not the web site; if someone uses the IRI they mean the individual denoted by the IRI.

Wikidata IRIs

- Wikidata (`https://www.wikidata.org`) is a free, collaborative knowledge graph with around 1.25 billion triples describing 100 million entities (as of 2022).

Wikidata IRIs

- Wikidata (`https://www.wikidata.org`) is a free, collaborative knowledge graph with around 1.25 billion triples describing 100 million entities (as of 2022).
- The soccer player Christine Sinclair is represented using the identifier "`http://www.wikidata.org/entity/Q262802`"

Wikidata IRIs

- Wikidata (`https://www.wikidata.org`) is a free, collaborative knowledge graph with around 1.25 billion triples describing 100 million entities (as of 2022).
- The soccer player Christine Sinclair is represented using the identifier "`http://www.wikidata.org/entity/Q262802`"
- The identifier "http://schema.org/name" is the property that gives the name of the subject

Wikidata IRIs

- Wikidata (`https://www.wikidata.org`) is a free, collaborative knowledge graph with around 1.25 billion triples describing 100 million entities (as of 2022).
- The soccer player Christine Sinclair is represented using the identifier "`http://www.wikidata.org/entity/Q262802`"
- The identifier "http://schema.org/name" is the property that gives the name of the subject
- "`http://www.wikidata.org/prop/direct/P27`" is the property "country of citizenship".

Wikidata IRIs

- Wikidata (`https://www.wikidata.org`) is a free, collaborative knowledge graph with around 1.25 billion triples describing 100 million entities (as of 2022).
- The soccer player Christine Sinclair is represented using the identifier "`http://www.wikidata.org/entity/Q262802`"
- The identifier "http://schema.org/name" is the property that gives the name of the subject
- "`http://www.wikidata.org/prop/direct/P27`" is the property "country of citizenship".
- Canada is "`http://www.wikidata.org/entity/Q16`"

Wikidata IRIs

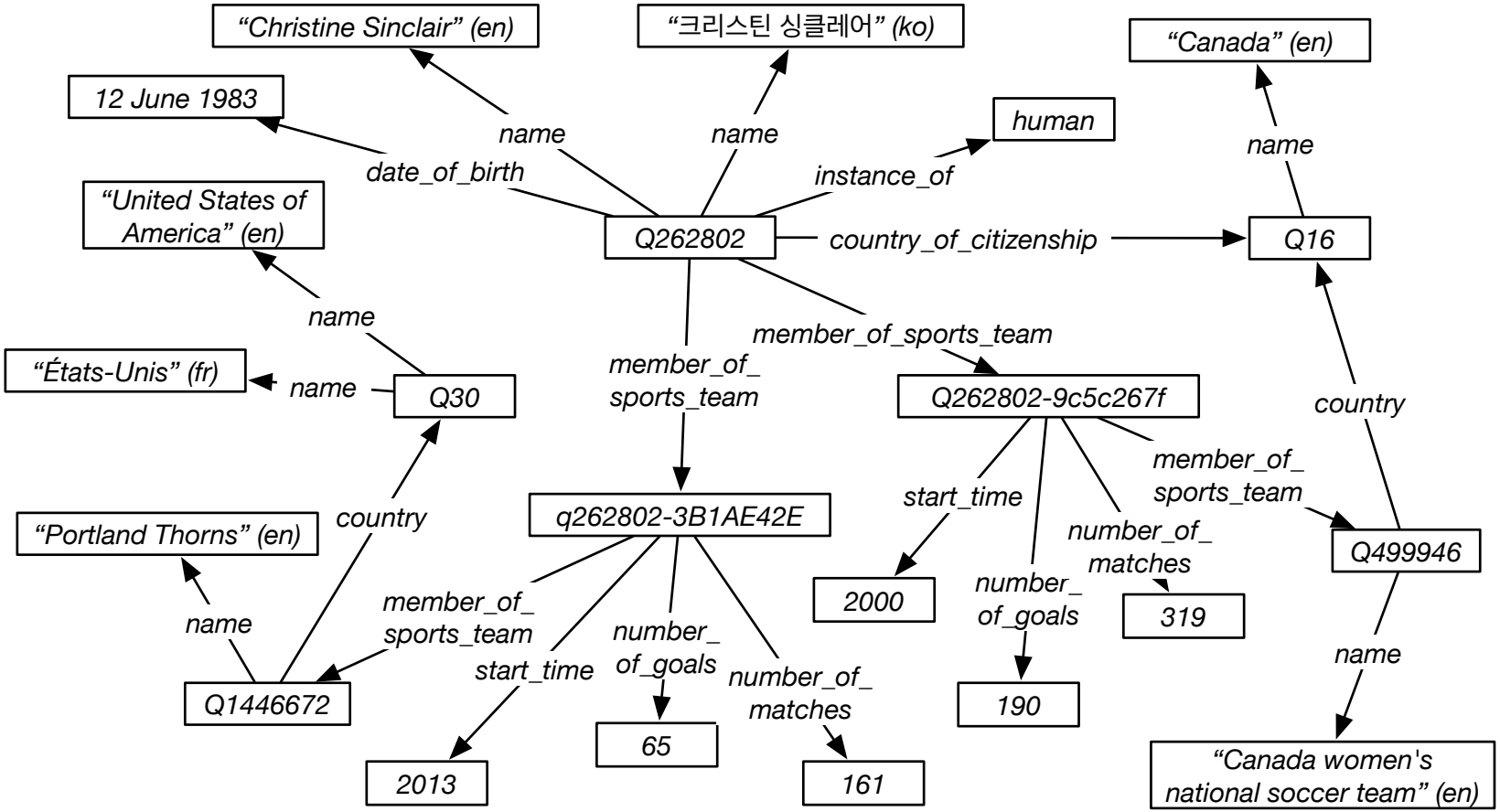
- Wikidata (`https://www.wikidata.org`) is a free, collaborative knowledge graph with around 1.25 billion triples describing 100 million entities (as of 2022).
- The soccer player Christine Sinclair is represented using the identifier "`http://www.wikidata.org/entity/Q262802`"
- The identifier "http://schema.org/name" is the property that gives the name of the subject
- "`http://www.wikidata.org/prop/direct/P27`" is the property "country of citizenship".
- Canada is "`http://www.wikidata.org/entity/Q16`"
- "Christine Sinclair is a citizen of Canada":

Wikidata IRIs

- Wikidata (`https://www.wikidata.org`) is a free, collaborative knowledge graph with around 1.25 billion triples describing 100 million entities (as of 2022).
- The soccer player Christine Sinclair is represented using the identifier "`http://www.wikidata.org/entity/Q262802`"
- The identifier "http://schema.org/name" is the property that gives the name of the subject
- "`http://www.wikidata.org/prop/direct/P27`" is the property "country of citizenship".
- Canada is "`http://www.wikidata.org/entity/Q16`"
- "Christine Sinclair is a citizen of Canada": *`/entity/Q262802`* *`/prop/direct/P27`* *`/entity/Q16`*

  but all starting with `http://www.wikidata.org`

Part of the Wikidata Knowledge Graph



Warning: naive methods to convert to triples don't work

Projecting onto pairs loses information:

- For example:
  Air Canada flies from New York to Vancouver
  Air Canada flies from Vancouver to Los Angeles

Warning: naive methods to convert to triples don't work

Projecting onto pairs loses information:

- For example:
  Air Canada flies from New York to Vancouver
  Air Canada flies from Vancouver to Los Angeles
- These are true triples:
  $\langle Air\ Canada, Flies\ From, New\ York \rangle$
  $\langle Air\ Canada, Flies\ To, Los\ Angeles \rangle$

Warning: naive methods to convert to triples don't work

Projecting onto pairs loses information:

- For example:
  Air Canada flies from New York to Vancouver
  Air Canada flies from Vancouver to Los Angeles
- These are true triples:
  $\langle Air\ Canada, Flies\ From, New\ York\rangle$
  $\langle Air\ Canada, Flies\ To, Los\ Angeles\rangle$
- However, Air Canada does not fly from New York to Los Angeles.
  The information about flights is lost!

Classes and Properties

- Primitive knowledge is knowledge that is specified explicitly.

Classes and Properties

- Primitive knowledge is knowledge that is specified explicitly.
- Derived knowledge is knowledge that can be inferred from primitive knowledge and other derived knowledge.

Classes and Properties

- Primitive knowledge is knowledge that is specified explicitly.
- Derived knowledge is knowledge that can be inferred from primitive knowledge and other derived knowledge.
- A standard way to use derived knowledge is to specify attributes (property–value pairs) for all members of a class.

Classes and Properties

- Primitive knowledge is knowledge that is specified explicitly.
- Derived knowledge is knowledge that can be inferred from primitive knowledge and other derived knowledge.
- A standard way to use derived knowledge is to specify attributes (property–value pairs) for all members of a class.
- Individuals inherit the attributes associated with the classes they are in.

Classes and Properties

- Primitive knowledge is knowledge that is specified explicitly.
- Derived knowledge is knowledge that can be inferred from primitive knowledge and other derived knowledge.
- A standard way to use derived knowledge is to specify attributes (property–value pairs) for all members of a class.
- Individuals inherit the attributes associated with the classes they are in.
- A natural kind is a class such that describing individuals using the class is more succinct than describing individuals without the class.

Classes and Properties

- Primitive knowledge is knowledge that is specified explicitly.
- Derived knowledge is knowledge that can be inferred from primitive knowledge and other derived knowledge.
- A standard way to use derived knowledge is to specify attributes (property–value pairs) for all members of a class.
- Individuals inherit the attributes associated with the classes they are in.
- A natural kind is a class such that describing individuals using the class is more succinct than describing individuals without the class.
- E.g., Specifying all *mammals* are warm blooded is more concise than specifying each individual is warm blooded.

Classes and Properties

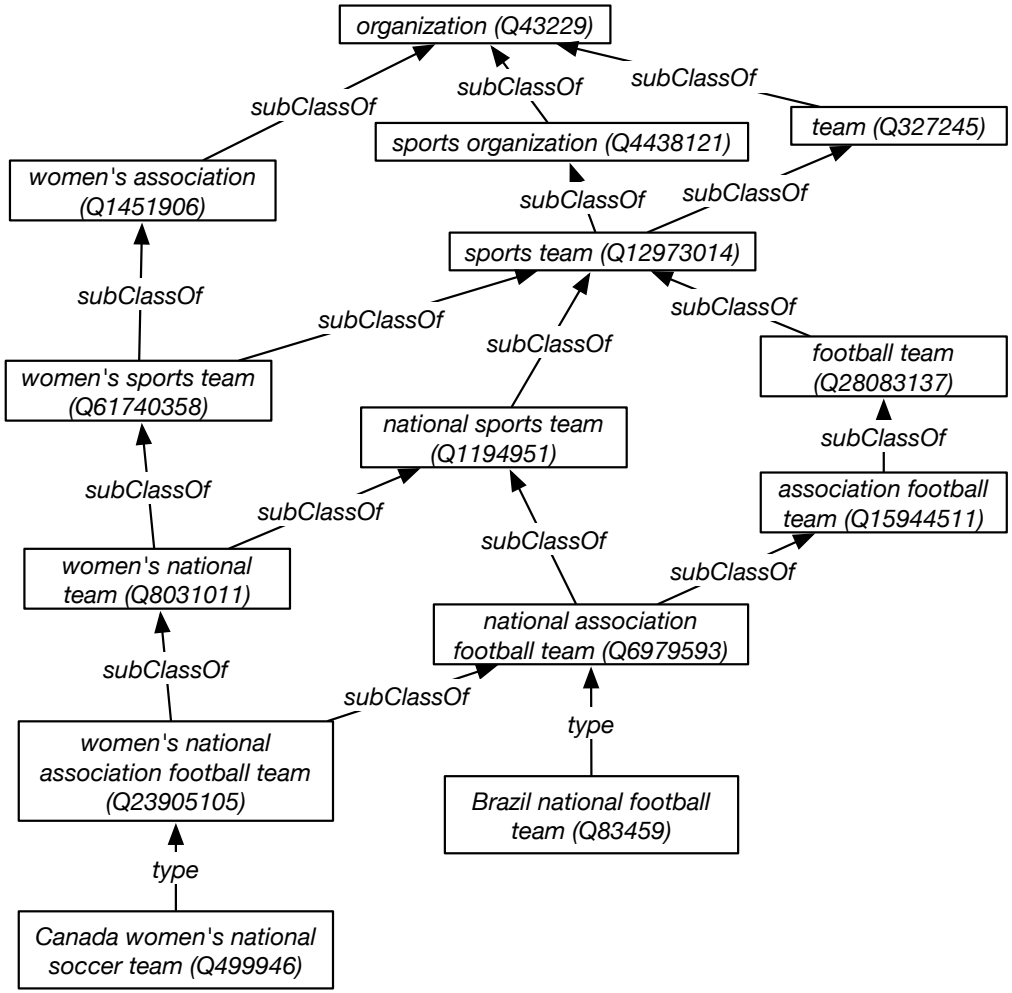
- Primitive knowledge is knowledge that is specified explicitly.
- Derived knowledge is knowledge that can be inferred from primitive knowledge and other derived knowledge.
- A standard way to use derived knowledge is to specify attributes (property–value pairs) for all members of a class.
- Individuals inherit the attributes associated with the classes they are in.
- A natural kind is a class such that describing individuals using the class is more succinct than describing individuals without the class.
- E.g., Specifying all *mammals* are warm blooded is more concise than specifying each individual is warm blooded.
- A class is the set of those actual and potential individuals that would be members of the class.
  E.g., class of chairs includes chairs that have not been build or may never be built.

Class Hierarchies

- Class S is a subclass of class C means S is a subset of C. $subClassOf(S, C) \wedge type(E, S) \rightarrow type(E, C)$.

Class Hierarchies

- Class S is a subclass of class C means S is a subset of C. $subClassOf(S, C) \wedge type(E, S) \rightarrow type(E, C)$.
- Part of the Wikidata class structure:



Property Hierarchies

- Property p_1 is a sub-property of property C if every pair related by p_1 is also related by p_2.
- $subPropertyOf(p_1, p_2) \wedge p_1(x, y) \rightarrow p_2(x, y)$

Property Hierarchies

- Property p_1 is a sub-property of property C if every pair related by p_1 is also related by p_2.
- $subPropertyOf(p_1, p_2) \wedge p_1(x, y) \rightarrow p_2(x, y)$

Some sub-properties in Wikidata are

- "member of sports team" (P54) is a sub-property of "member of" (P463)
- "member of" (P463) is a sub-property of "affiliation" (P1416)
- "affiliation" (P1416) is a sub-property of "part of" (P361)
- "part of" (P361) is a sub-property of "partially coincident with" (P1382)

Property

- The domain of a property is a class such that the subject of a triple with the property has to be in the class.
  $domain(p, C) \wedge p(x, y) \rightarrow type(x, C)$.

Property

- The domain of a property is a class such that the subject of a triple with the property has to be in the class.
  $domain(p, C) \wedge p(x, y) \rightarrow type(x, C)$.
- domain of "member of sports team" (P54) is

Property

- The domain of a property is a class such that the subject of a triple with the property has to be in the class.
  $domain(p, C) \wedge p(x, y) \rightarrow type(x, C)$.
- domain of "member of sports team" (P54) is "human" (Q5)

Property

- The domain of a property is a class such that the subject of a triple with the property has to be in the class.
  $domain(p, C) \wedge p(x, y) \rightarrow type(x, C)$.
- domain of "member of sports team" (P54) is "human" (Q5)
- The range of a property is a class such that the object of a triple with the property has to be in the class.
  $range(p, C) \wedge p(x, y) \rightarrow type(y, C)$.

Property

- The domain of a property is a class such that the subject of a triple with the property has to be in the class.
  $domain(p, C) \wedge p(x, y) \rightarrow type(x, C)$.
- domain of "member of sports team" (P54) is "human" (Q5)
- The range of a property is a class such that the object of a triple with the property has to be in the class.
  $range(p, C) \wedge p(x, y) \rightarrow type(y, C)$.
- range of "member of sports team" (P54) is

Property

- The domain of a property is a class such that the subject of a triple with the property has to be in the class.
  $domain(p, C) \wedge p(x, y) \rightarrow type(x, C)$.
- domain of "member of sports team" (P54) is "human" (Q5)
- The range of a property is a class such that the object of a triple with the property has to be in the class.
  $range(p, C) \wedge p(x, y) \rightarrow type(y, C)$.
- range of "member of sports team" (P54) is "sports team" (Q12973014)

Property

- The domain of a property is a class such that the subject of a triple with the property has to be in the class.
  $domain(p, C) \wedge p(x, y) \rightarrow type(x, C)$.
- domain of "member of sports team" (P54) is "human" (Q5)
- The range of a property is a class such that the object of a triple with the property has to be in the class.
  $range(p, C) \wedge p(x, y) \rightarrow type(y, C)$.
- range of "member of sports team" (P54) is "sports team" (Q12973014)
- Property p is functional means there is at most one object associated with any subject.
  $p(x, y_1) \wedge p(x, y_2) \rightarrow y_1 = y_2$.

Property

- The domain of a property is a class such that the subject of a triple with the property has to be in the class.
  $domain(p, C) \wedge p(x, y) \rightarrow type(x, C)$.
- domain of "member of sports team" (P54) is "human" (Q5)
- The range of a property is a class such that the object of a triple with the property has to be in the class.
  $range(p, C) \wedge p(x, y) \rightarrow type(y, C)$.
- range of "member of sports team" (P54) is "sports team" (Q12973014)
- Property p is functional means there is at most one object associated with any subject.
  $p(x, y_1) \wedge p(x, y_2) \rightarrow y_1 = y_2$.
- Is "member of sports team" (P54) functional?

Property

- The domain of a property is a class such that the subject of a triple with the property has to be in the class.
  $domain(p, C) \wedge p(x, y) \rightarrow type(x, C)$.
- domain of "member of sports team" (P54) is "human" (Q5)
- The range of a property is a class such that the object of a triple with the property has to be in the class.
  $range(p, C) \wedge p(x, y) \rightarrow type(y, C)$.
- range of "member of sports team" (P54) is "sports team" (Q12973014)
- Property p is functional means there is at most one object associated with any subject.
  $p(x, y_1) \wedge p(x, y_2) \rightarrow y_1 = y_2$.
- Is "member of sports team" (P54) functional?
- Is "date of birth" (P569) functional?

Property

- The domain of a property is a class such that the subject of a triple with the property has to be in the class.
  $domain(p, C) \wedge p(x, y) \rightarrow type(x, C)$.
- domain of "member of sports team" (P54) is "human" (Q5)
- The range of a property is a class such that the object of a triple with the property has to be in the class.
  $range(p, C) \wedge p(x, y) \rightarrow type(y, C)$.
- range of "member of sports team" (P54) is "sports team" (Q12973014)
- Property p is functional means there is at most one object associated with any subject.
  $p(x, y_1) \wedge p(x, y_2) \rightarrow y_1 = y_2$.
- Is "member of sports team" (P54) functional?
- Is "date of birth" (P569) functional?

These are not (currently) part of Wikidata!

Designing Classes

Aristotle [350 B.C.] suggested the definition if a class C in terms of:

- Genus: a super-class
- Differentia: the attributes that make members of the class C different from other members of the super-class

"If genera are different and co-ordinate, their differentiae are themselves different in kind. Take as an instance the genus 'animal' and the genus 'knowledge'. 'With feet', 'two-footed', 'winged', 'aquatic', are differentiae of 'animal'; the species of knowledge are not distinguished by the same differentiae. One species of knowledge does not differ from another in being 'two-footed'."

Aristotle, *Categories*, 350 B.C.

Designing Classes

The art of ranking things in genera and species is quite important, and greatly helps our judgment as well as our memory. . . . Order largely depends on it, and many good authors write in such a way that their whole account could be divided and subdivided according to a procedure related to genera and species. This helps one not merely to retain things in one's memory, but also to find them there. Writers who have laid out all sorts of notions under certain headings or categories have done something very useful.

– G. W. Leibniz [1705]

Designing Classes

To design classes based on Aristotelian definitions:

- For each class: determine a relevant superclass and then select those attributes that distinguish the class from other subclasses.
- Each attribute gives a property and a value.

Designing Classes

To design classes based on Aristotelian definitions:

- For each class: determine a relevant superclass and then select those attributes that distinguish the class from other subclasses.
- Each attribute gives a property and a value.
- For each property, define the domain to be most general class for which it makes sense.

Designing Classes

To design classes based on Aristotelian definitions:

- For each class: determine a relevant superclass and then select those attributes that distinguish the class from other subclasses.
- Each attribute gives a property and a value.
- For each property, define the domain to be most general class for which it makes sense.
- Make the range of the property another class that makes sense (perhaps requiring this range class to be defined, either by enumerating its values or by defining it using an Aristotelian definition).

Designing Classes

Aristotelian definitions result in class structure that is a lattice, and rarely a tree.

Designing Classes

Aristotelian definitions result in class structure that is a lattice, and rarely a tree.

Example: Consider definitions of rectangle, rhombus, and square:

- A quadrilateral is a planar figure made up of four straight sides.

Designing Classes

Aristotelian definitions result in class structure that is a lattice, and rarely a tree.

Example: Consider definitions of rectangle, rhombus, and square:

- A quadrilateral is a planar figure made up of four straight sides.
- A rectangle is a

Designing Classes

Aristotelian definitions result in class structure that is a lattice, and rarely a tree.

Example: Consider definitions of rectangle, rhombus, and square:

- A quadrilateral is a planar figure made up of four straight sides.
- A rectangle is a quadrilateral where

Designing Classes

Aristotelian definitions result in class structure that is a lattice, and rarely a tree.

Example: Consider definitions of rectangle, rhombus, and square:

- A quadrilateral is a planar figure made up of four straight sides.
- A rectangle is a quadrilateral where all inside angles are right angles (90°).

Designing Classes

Aristotelian definitions result in class structure that is a lattice, and rarely a tree.

Example: Consider definitions of rectangle, rhombus, and square:

- A quadrilateral is a planar figure made up of four straight sides.
- A rectangle is a quadrilateral where all inside angles are right angles (90°).
- A rhombus is a

Designing Classes

Aristotelian definitions result in class structure that is a lattice, and rarely a tree.

Example: Consider definitions of rectangle, rhombus, and square:

- A quadrilateral is a planar figure made up of four straight sides.
- A rectangle is a quadrilateral where all inside angles are right angles (90°).
- A rhombus is a quadrilateral where

Designing Classes

Aristotelian definitions result in class structure that is a lattice, and rarely a tree.

Example: Consider definitions of rectangle, rhombus, and square:

- A quadrilateral is a planar figure made up of four straight sides.
- A rectangle is a quadrilateral where all inside angles are right angles (90°).
- A rhombus is a quadrilateral where all four sides have the same length.

Designing Classes

Aristotelian definitions result in class structure that is a lattice, and rarely a tree.

Example: Consider definitions of rectangle, rhombus, and square:

- A quadrilateral is a planar figure made up of four straight sides.
- A rectangle is a quadrilateral where all inside angles are right angles (90°).
- A rhombus is a quadrilateral where all four sides have the same length.
- A square is a quadrilateral where

Designing Classes

Aristotelian definitions result in class structure that is a lattice, and rarely a tree.

Example: Consider definitions of rectangle, rhombus, and square:

- A quadrilateral is a planar figure made up of four straight sides.
- A rectangle is a quadrilateral where all inside angles are right angles (90°).
- A rhombus is a quadrilateral where all four sides have the same length.
- A square is a quadrilateral where all four sides have the same length and all inside angles are right angles.

Designing Classes

Aristotelian definitions result in class structure that is a lattice, and rarely a tree.

Example: Consider definitions of rectangle, rhombus, and square:

- A quadrilateral is a planar figure made up of four straight sides.
- A rectangle is a quadrilateral where all inside angles are right angles (90°).
- A rhombus is a quadrilateral where all four sides have the same length.
- A square is a quadrilateral where all four sides have the same length and all inside angles are right angles.

What is a most specific superclass of square?

Designing Classes

Aristotelian definitions result in class structure that is a lattice, and rarely a tree.

Example: Consider definitions of rectangle, rhombus, and square:

- A quadrilateral is a planar figure made up of four straight sides.
- A rectangle is a quadrilateral where all inside angles are right angles (90°).
- A rhombus is a quadrilateral where all four sides have the same length.
- A square is a quadrilateral where all four sides have the same length and all inside angles are right angles.

What is a most specific superclass of square?
A square is both a rectangle and a rhombus.

Designing Classes

Aristotelian definitions result in class structure that is a lattice, and rarely a tree.

Example: Consider definitions of rectangle, rhombus, and square:

- A quadrilateral is a planar figure made up of four straight sides.
- A rectangle is a quadrilateral where all inside angles are right angles (90°).
- A rhombus is a quadrilateral where all four sides have the same length.
- A square is a quadrilateral where all four sides have the same length and all inside angles are right angles.

What is a most specific superclass of square?
A square is both a rectangle and a rhombus.
Neither is more specific than the other.