The Auxiliary *FuncProp* Property

A functional property is a property that is a function: it relates a resource to at most one value. Functional properties are not a concept of RDF but are used in the axiomatization of other primitives.

The constant $FuncProp$ represents the class of all functional properties. P is a functional property if, and only if, it is a property, and there are no x, y_1, and y_2 such that $P(x, y_1)$, $P(x, y_2)$, and $y_1 \neq y_2$.

$$Type(?p, FuncProp) \longleftrightarrow$$
$$(Type(?p, Property) \wedge \forall ?r \forall ?v1 \forall ?v2$$
$$(PropVal(?p, ?r, ?v1) \wedge PropVal(?p, ?r, ?v2) \longrightarrow ?v1 = ?v2))$$

Reified Statements

The constant $Statement$ represents the class of all reified statements. All reified statements are resources, and $Statement$ is an instance of $Class$:

$$Type(?s, Statement) \longrightarrow Type(?s, Resource)$$
$$Type(Statement, Class)$$

A reified statement can be decomposed into the three parts of an RDF triple:

$$Type(?st, Statement) \longrightarrow$$
$$\exists ?p \exists ?r \exists ?v(PropVal(Predicate, ?st, ?p) \wedge$$
$$PropVal(Subject, ?st, ?r) \wedge PropVal(Object, ?st, ?v))$$

$Subject$, $Predicate$, and $Object$ are functional properties. That is, every statement has exactly one subject, one predicate, and one object:

$$Type(Subject, FuncProp)$$
$$Type(Predicate, FuncProp)$$
$$Type(Object, FuncProp)$$