the same thing. For example, :Winner and :Loser are disjoint, and :roger_federer and :rafael_nadal are different individuals.

Boolean Combinations of Classes Sometimes classes need to be combined in ways that go beyond subclass relations. For instance, we may want to define the class :Person to be the disjoint union of the classes :Female and :Male.

Local Scope of Properties rdfs:range states that the instances in the range of a property, say :plays, all belong to a certain class. Thus in RDFS we cannot declare range restrictions that differentiate between contexts. For example, we cannot say that tennis players play only tennis, while other people may play badminton.

Special Characteristics of Properties Sometimes it is useful to say that a property is *transitive*, such as :greater_than; *unique*, like :is_mother_of; or the *inverse* of another property, such as :eats and :is_eaten_by.

Cardinality Restrictions Sometimes we need to place restrictions on how many distinct values a property may or must take. For example, each person has exactly two parents, and a course is taught by at least one lecturer.

Consistency Once we can determine relations between classes, we may also want to determine conflicts between their definitions. Suppose we have declared :Fish and :Mammal to be disjoint classes. It is then an error to assert that :dolphin is an instance of both. A sufficiently expressive ontology language should allow us to detect these types of inconsistencies.

Finally, an ontology language must make it as *convenient* as possible to build sentences that make use of its expressiveness. For instance, a language is not very convenient if we would need to reiterate entire definitions every time we want to state that two classes are equivalent.