

## Tutorial: Travel agency ontology

Use the Protégé editor to define a normalised ontology for use by a travel agency covering the following:

Hotel, restaurant, sports, luxury hotel, bed and breakfast, safari, activity, hiking, spa treatment, sunbathing, sightseeing, accommodation rating (three stars, etc.), campground, surfing.

**Build** a class hierarchy and indicate which classes in it are primitive and which are definable. Define the required relations, their properties, domains and ranges as well as individuals.

**Define** the following classes:

1. A two star hotel.
2. A spa resort (i.e., a destination offering a spa treatment).
3. A destination with sport activities but without safari.
4. A destination where all hotels have three star rating.
5. A destinations with at least three restaurants and at least four hotels.

# 1. Card sorting

hotel

restaurant

sports

luxury hotel

bed & breakfast

two stars

sunbathing

sightseeing

accommodation rating

three stars

campground

swimming pool

hiking

activity

spa treatment

safari

surfing

destination

## 2. Arrange Concepts/Properties into Hierarchy

self-standing	modifiers	relations	definable
<ul style="list-style-type: none"> <li>- accommodation               <ul style="list-style-type: none"> <li>- hotel                   <ul style="list-style-type: none"> <li>- luxury hotel</li> </ul> </li> <li>- b&amp;b</li> <li>- campground</li> </ul> </li> <li>- activity               <ul style="list-style-type: none"> <li>- sports                   <ul style="list-style-type: none"> <li>- surfing</li> <li>- hiking</li> </ul> </li> <li>- safari</li> <li>- sightseeing</li> <li>- relaxation                   <ul style="list-style-type: none"> <li>- sunbathing</li> <li>- spa treatment</li> </ul> </li> </ul> </li> <li>- restaurant</li> <li>- destination</li> </ul>	accommodation rating <ul style="list-style-type: none"> <li>- two stars</li> <li>- three stars</li> <li>- four stars</li> <li>- five stars</li> </ul> hotel facility <ul style="list-style-type: none"> <li>- swimming pool</li> <li>- meeting facilities</li> </ul>	hasAccommodation hasActivity hasRating hasRestaurant hasFacility	two star hotel ...

**NB.** All siblings in the hierarchy of self-standing entities are **disjoint**.

## Class Hierarchy represents an “IS-A” Relation

a class *A* is a **subclass** of *B* if **every** instance of *A* is also an instance of *B*

- hotels and B&Bs are accommodations  
therefore, *Hotel* and *B&B* may be regarded as subclasses of *Accommodation*
- hiking and surfing are sport activities  
therefore, *Hiking* and *Surfing* may be regarded as subclasses of *Sports*
- **however**, neither a restaurant is a hotel nor a swimming pool is a hotel  
therefore, neither *Restaurant* nor *SwimmingPool* can be a subclass of *Hotel*  
(although a hotel may have a restaurant/swimming pool)

## Relations

Destination <u>hasAccommodation</u> Accommodation	domain: Destination range: Accommodation
Destination <u>hasActivity</u> Activity	domain: Destination range: Activity
Destination or Hotel <u>hasRestaurant</u> Restaurant	domain: Destination $\cup$ Hotel range: Restaurant
Hotel <u>hasFacility</u> Facility	domain: Hotel range: Facility

## Modifiers: using value sets

Consider modifier *AccommodationRating*  
(one star, two stars, three stars, four stars, five stars)

- 5 individuals: oneStar, twoStars, . . . , fiveStars
- class AccommodationRating consisting of those 5 individuals
- **functional** property hasRating  
with domain Accommodation and range AccommodationRating