Task 1: Application

Semantic Application Idea:

"Dog Sport Connect" is a social application that helps dog owners find sports venues, events, and like-minded individuals for joint training sessions. Using knowledge graphs, the application provides recommendations for choosing sports activities based on the dog's breed, age, and activity level. It also supports social connections between owners to share experiences, train together, and attend events.

The application simplifies access to organized training sessions and venues, aiming to create a community of active dog owners who wish to improve their dogs' fitness and connect with other dog enthusiasts.

Competency Questions for the Knowledge Graph:

1. What dog training venues are nearby?
2. Are there any events or competitions for *this* breed nearby?
3. Which dog owners are interested in agility training?
4. Where can I find groups for joint flyball training?
5. What sports are best for my Border Collie?
6. What sports venues are available on weekends?
7. What kind of sports training is popular among dog owners in my area?
8. Are there any trainers nearby offering frisbee lessons for dogs?
9. What training can be organized with minimal equipment?
10. Which muscles should be developed for a dog to play frisbee?

Key Features of the Application:

- Information about available venues and their features
- Schedule of upcoming events and the ability to join
- The ability to find other dog owners for joint workouts and sharing experiences
- Sports and event recommendations based on breed, age, and activity level of the dog
- Video tutorials, training tips, and access to professional trainers.

This application will help dog owners not only find suitable places for sports activities but also build a community of like-minded people, adding social value to the training.

Task 2: Ontology

Steps for Creating the Ontology:

1. Defining that the ontology will describe dog sports activities and social interactions, including venues, sports types, events, and relationships between owners
2. Determining key concepts such as "Dogs," "Venues," "Events," "Trainers", "Sport Activities" and "Owners" and their relationships, e.g., "Owner -> owns -> Dog" and "Dog -> participates in -> Sport Activities"
3. Creating a hierarchy of classes, adding data properties (e.g., age, breed) and object properties (e.g., hosts, attends) to ensure connectivity between concepts
4. Extending the ontology using OWL constructs like property constraints (e.g., minimum number of event participants), logical connectors, and class expressions to enhance semantic understanding
5. Adding labels and comments to each element to ensure clarity and enhance reusability in the future

6. Checking logic of the ontology by starting the Reasoner and fixing inconsistencies.

My ontology utilizes various OWL constructs, including property constraints, logical operators, disjoint classes, cardinalities, and compound properties.

The OWL features that were used:

I defined **cardinality constraints** by specifying that each dog must have at least one owner and that each event must take place at exactly one training venue.

I used **the intersection** to create new subclasses. For example, I created an AgilityTrainers subclass, which will be the intersection of the Agility class (where Agility is a subclass of Sports Activities) and Trainers class. This means that trainers in this subclass specialize in training dogs for Agility classes.

I defined a **symmetric property** for communication between trainers and owners, for example, the "cooperatesWith" property, which indicates that if a trainer works with a certain owner, then the owner also works with this trainer.

For the Dogs class, I created the "hasOwner» property, which has the range class "Owners", and the domain class "Dogs". This property is **asymmetric**, ensuring that the relationship is one-directional - if Dog A is owned by Owner B, Owner B cannot be owned by Dog A.

I used **compound properties** (Property Chain Axiom) to define relationships between objects through intermediate steps. For example, I created a property chain to indicate that if a dog attend an event, that automatically the owner of this dog also attends the event. This is done by combining the properties "hasOwner" and "ownerAttends", where the chain specifies that the owner of a dog attending an event must also be recognized as attending that event.

For the "isPartOf" property, I defined that events can be part of sports activities but cannot be part of themselves and can belong to only one activity. This is achieved using **functional** and **irreflexive properties**.

For the "hasExperience" property, I defined that the trainer must have at least 1 year of experience. This is an example of a **property constraint**, as it specifies the minimum value that the "hasExperience" property must have. Such a constraint ensures that all trainers meet a minimum level of qualification.

I made the "Meetup" subclass part of a disjoint union along with the subclasses "Charity Event", "Competition", "Seminar", "Training Session", and "Workshop" using **disjointness property**. This ensures that an event cannot be classified under multiple categories.

I used the "SubClass Of" property to define that the class "Akita" is a subclass of "Dogs". I also utilized the "isSuitableForActivity" property to indicate that Akitas are suitable for specific activities, such as "Weight Pulling", "Bikejoring", "Carting", and "Skijoring", representing **existential restrictions** in OWL.