University of Glasgow

School of Computing Science

Lab 1 Tasks: Solutions

A proposed E-R diagram is provided:

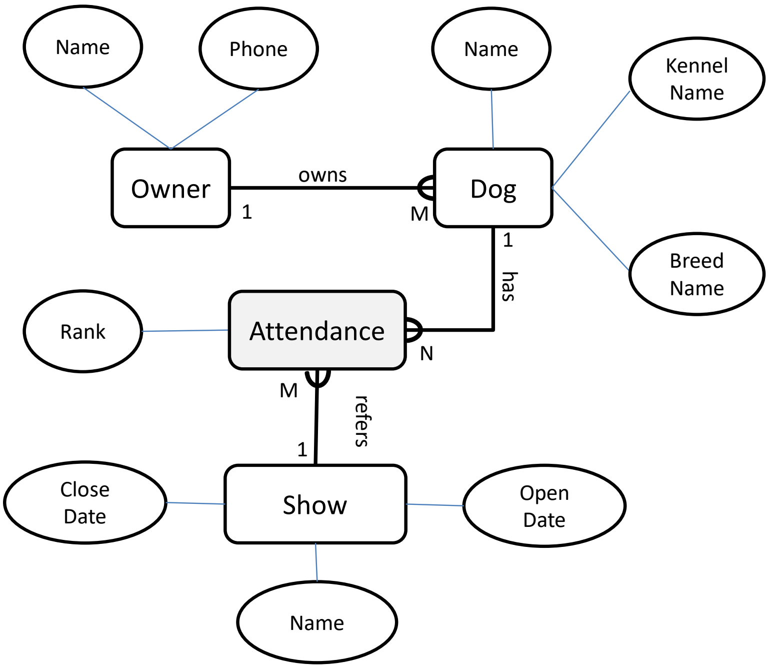
- **Owner (Entity)** of a dog has a name (Attribute) and contact details, e.g., phone number (Attribute).
- **Dog (Entity)** has a name (Attribute), an owner (associated with the Entity Owner), a breed name (Attribute) and a kennel (Attribute). In this proposed solution, we assume that a dog has a *unique* owner. Surely, we can assume that a dog might have more than one owner, which only changes the cardinality degree.
- **Show (Entity)** has a name, an opening and closing date. The show is identified by the combination of its name and its opening date.

A dog can attend many shows, where in each show, the dog has a specific performance assessed by the ranking. On the other hand, a show can be attended by many dogs. Thus, we derive a many-to-many relationship between: Dog and Show. However, we need to 'store' the ranking information of *a specific dog attending a specific show*. That is, we need to introduce a new attribute 'rank', which can take one of the values e.g., 1: gold, 2: silver, 3: bronze, or 4: none, and then *associate this ranking to a specific dog attending a specific show*. In this case, we introduce a *relationship entity* **Attendance**, which is used to associate the entities Dog and Show *via* two one-to-many relationships. This means that, we split the many-to-many relationship between Dog and Show into two one-to-many relationships: (Dog, Attendance) and (Attendance, Show). In addition, we attach the attribute 'rank' to the Attendance relationship entity.

Summarizing, we identified the following types of relationships:

- **One to many (1:M),** an owner may have more than one dog, but each dog has *only* one owner (in this proposed solution).
- **Many to many (N:M),** each dog attends several shows, and there are more than one dog participating in each show. We introduced the relationship-entity Attendance to associate a Dog with an Attendance to a specific Show. The Attendance stores information about the performance (rank) of the dog in the corresponding Show. Hence, we split this many-to-many relationship into two one-to-many relationships:
    - **One to many (1:N),** each *dog* is associated with many (N) *attendances* (each *attendance* corresponds to a specific *show*). That is, a dog has participated into N shows.
    - **One to many (1:M),** each *show* is associated with many (M) *attendances* (each *attendance* corresponds to a specific *dog*). That is, a show has been attended by M dogs.

Hence, the E-R Model is then:



Example: Let us instantiate the E-R diagram by adding some instances to the entities. Assume that there are three dogs: Bob, Alice and Jim, two owners: Phil and Chris, and two shows: Show1 and Show2. Bob is ranked 1st in Show1 and 2nd in Show2, Alice is ranked 3rd in Show1 only, Jim is ranked 2nd in Show1 and 1st in Show2. Chris owns Bob and Alice, while Phil owns Jim. Then, based on our conceptual model, we can 'populate' our entities with these facts/instances as follows:

- Entity: Owner; Instances: {Chris, Phil}
- Entity: Dog; Instances: {Bob, Alice, Jim}
- Entity: Show; Instances: {Show1, Show2}
- Entity: Attendance; Instances: {(Bob, Show1, 1st), (Bob, Show2, 2nd), (Alice, Show1, 3rd), (Jim, Show1, 2nd), (Jim, Show2, 1st)}

//note: not all the attributes of the entities are included in this example.

Now, these instances reflect the necessity of introducing the Attendance entity where we associate an instance of Dog with an instance of Show storing the corresponding ranking.