SCHOOL OF
COMPUTER AND SECURITY SCIENCE

AUSTRALIA ECU EDITH COWAN UNIVERSITY

CSG1207/CSI5135: Systems and Database Design
Workshop 04

Background

Entity relationship modelling allows us to design and model a database in a visual manner. ER diagrams consist of entities, the relationships between them (1:1, 1:M, M:M) and the attributes of those relationships. Other aspects of ER modelling, such as cardinality, allow us to depict more detail and information in diagrams.

Enhanced ER modelling allows for the depiction of more sophisticated aspects of a system, particularly in relation to the relationships between entities. Multiple relationships occur when two entities are linked in more than one way, self-referencing relationships occur when a given entity has a recursive relationship with itself and supertypes and subtypes provide a sophisticated and efficient way to represent related entities.

Task 1

Answer the following review questions:

1. Name two possible situations where you may encounter a self-referencing relationship (besides those mentioned in the lecture) and draw a physical ER diagram for each.

2. What must be taken into account for attribute names when multiple relationships exist between two entities?

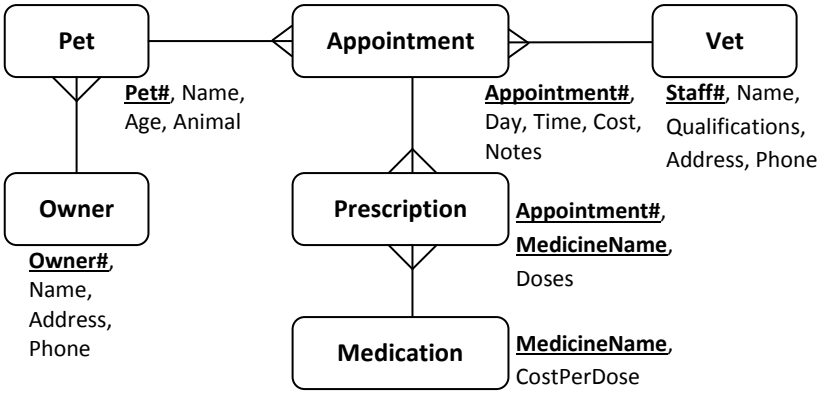
3. Create an ER diagram involving a supertype for a music festival ticket (consisting of a ticket#, sale date and name of ticket holder) which has two subtypes:
    - Single stage ticket (includes attributes of stage number and stage location)
    - Single show ticket (includes attributes of show number and start time)

4. What rules dictate the order of table creation and table dropping?

5. What is the purpose of a data dictionary?

6. Name and describe the three commonly known sub-languages of SQL.

Task 2



Given the ER diagram above, do the following:

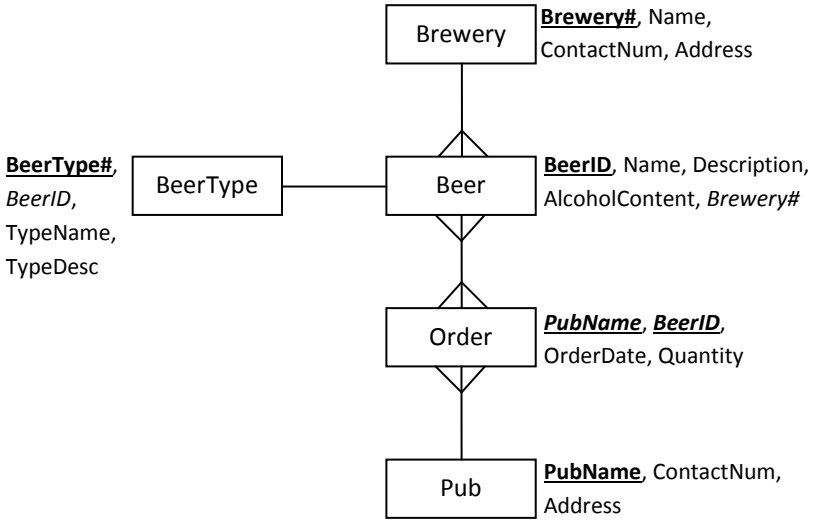
- **A.** Add cardinality to all relationships in the diagram, stating any assumptions you make
- **B.** Convert it to a physical diagram
- **C.** List two possible create and drop orders for the diagram
- **D.** Add an entity named AnimalType which contains a list of animal types (Cat, Dog, etc), which will have relationships to the Pet entity and the Vet entity. A pet can only be one animal type, but a vet may specialise in many animal types (and there may be multiple vets who specialise in a specific animal)

Task 3

Come up with a database scenario and create a physical ER diagram to model it. Aim to include at least 5 entities (including intermediary entities). Your diagram should include entities with multiple relationships, a self-referencing relationship, and supertypes/subtypes if you can think of an appropriate scenario.

Task 4

Your friend was asked to design a database for a chain of breweries to manage their beer and sales to pubs, but ended up sampling too much beer and came up with the following diagram.



The finished database design is due tomorrow, and your friend is too busy talking to a pile of beer kegs to finish it, so you've decided to help. You know the diagram has errors and is missing some things that should be shown. You know the following details about the scenario:

- A beer can be made in multiple breweries, not just one
- Orders can consist of more than one beer
- Breweries make more than one beer

Find and correct any errors in the diagram, and make sure that it incorporates the details that you know. The result should be a well-structured physical ER diagram including cardinality. Once this is complete, list the table creation and dropping order for the database, and try to create a data dictionary for it.

Task 5

You wish to create a database to store details of Pokémon. The database needs to keep track of Pokémon details, including the moves they can do and which Pokémon they evolve into. Draw a physical ER diagram to model this, using the following guidelines:

- Pokémon details include number, name, description and type (Fire, Water, Earth, etc)
- Move details include the name of the move and the amount of points it takes to use
- Pokémon can do many moves, and each move can be done by multiple Pokémon
- Different Pokémon learn the same move at a different level – this should be included
- Attacks are associated with a type, e.g. "Water Gun" is a Water type move
- You can assume a Pokémon can only evolve into one other Pokémon, and some do not evolve. This should be implemented as a self-referencing relationship