DB Week 3 Workshop

INFO20003 | Sandy Luo

Workshop Overview

01

Entity Relation (ER) Recap

02

Case Study: Conceptual & Logical models

03

Lab: Create Physical model

01

Entity

02

Weak Entity

03

Attribute

04

Business rules → Relationships

Entity Relation

Entity

- Real world “object”
- Unit that has a collection of information describing it
  - Every instance of this object is **unique** → has ID (**Primary Key**)

Concrete	Abstract
• User • Sale item • Classroom • Cinema	• Event • Subscription • Role • Currency

Are these entities? Concrete / abstract?

- Library Book
- Contract
- Speed
- Flight Reservation
- Sales Report
- Colour
- Bank Transaction
- Email Address
- Medical Prescription
- Department
- Temperature
- Happiness

01

Entity

02

Weak Entity

03

Attribute

04

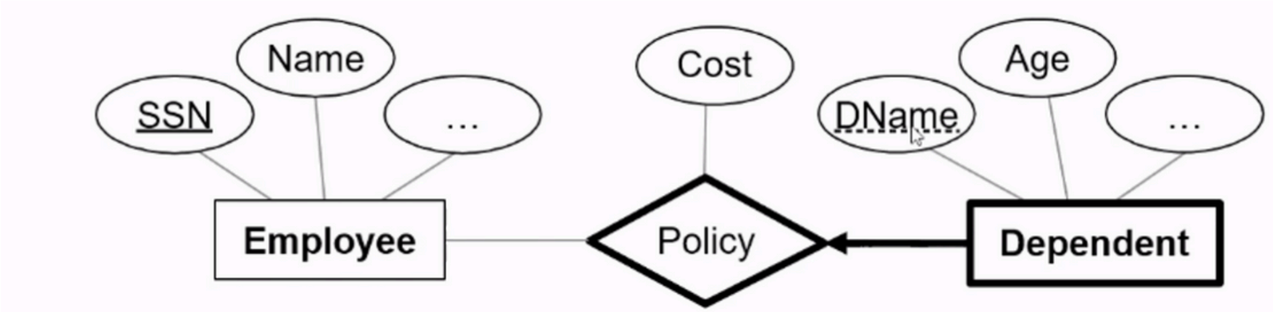
Business rules → Relationships

Entity Relation

Weak Entity

- Entity that **cannot be uniquely identified** alone
  - No ID (Primary Key), only ***partial key***

- Mandatory **identifying** relationship with ONE parent / owner entity
  - Weak = Need parent's primary key to be uniquely identified



e.g. insurance covered by company for family can only exist if still employed

Intuition: Can it be on its own in the DB?

Family Card ID: 123

Member 1: Finn
Member 2: Jake
Member 3: Fiona
Member 4: Cake

Entities:

- Family
- Member

Table of Families

ID	Start Date	Expiry Date	Payment card #
123	…	…	…
…	…	…	…

Table of Members

Member #	First Name	Type
1	Finn	Human
2	Jake	Dog
…	…	…

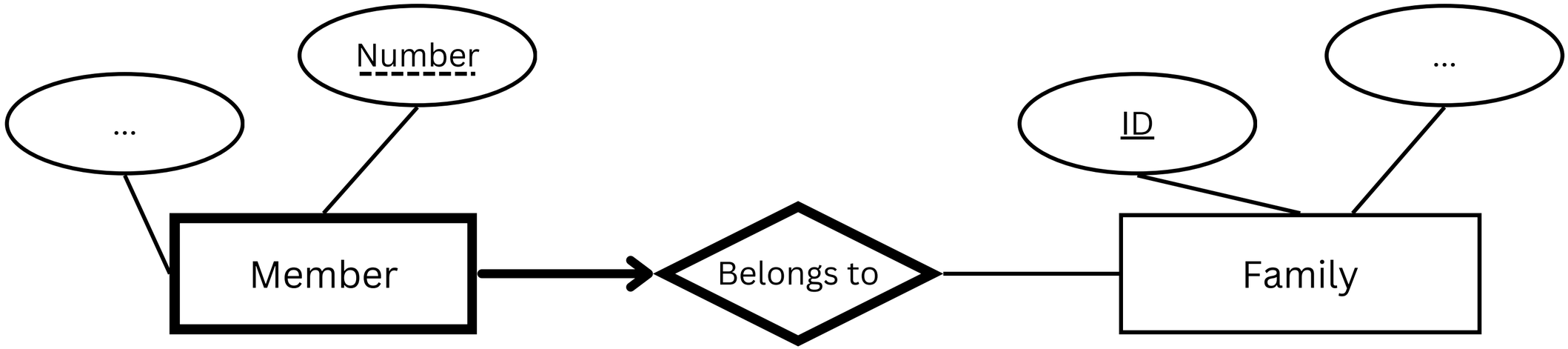
Is this instance (row) guaranteed to be unique?

No, it is only unique if we say “Member 1 of Family 123”.

So “Member” is a WEAK ENTITY!

Identifying entity = Family, Primary Key = ID, Partial Key = Member #

Intuition: Can it be on its own in the DB?



- Bolded: Weak entity, joining relationship (on weak side)
- Why the arrow?
- **Primary** key vs **Partial** key

01

Entity

02

Weak Entity

03

Attribute

04

Business rules → Relationships

Entity Relation

Attribute

- **Characteristics** describing an entity / relationship

- Each attribute has a range of permitted values:
  - name: # characters, alphanumeric…
  - age: small integer
  - address: # characters
  - …

01

Entity

02

Weak Entity

03

Attribute

04

Business rules → Relationships

Entity Relation

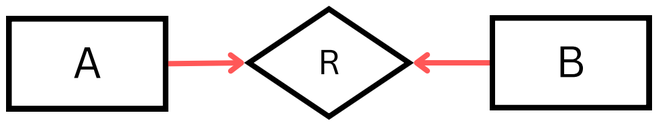
Business Rules

- Business rules define:
  - Entities
  - Attributes
  - **Relationships**
    - b/w two (+) entities
  - **Constraints**
    - Key constraints (max)
    - Participating constraints (

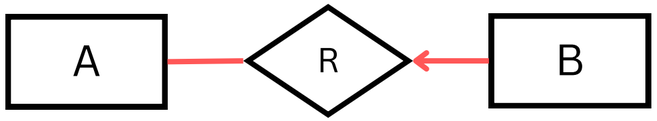
Key Constraint

- ***Maximum* # of associations** an entity can have in a relationship set

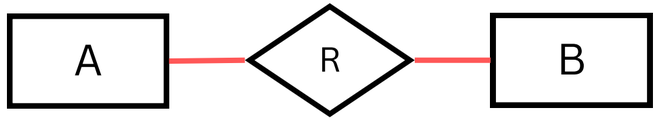
1. One-to-one
   - **One** A can only be associated w/ max. **one** B
   - Vice versa



2. One-to-many
   - **One** A can be associated w/ **many** B's
   - **One** B can only be associated w/ max. **one** A



3. Many-to-many
   - **One** A can be associated w/ **many** B's
   - Vice versa

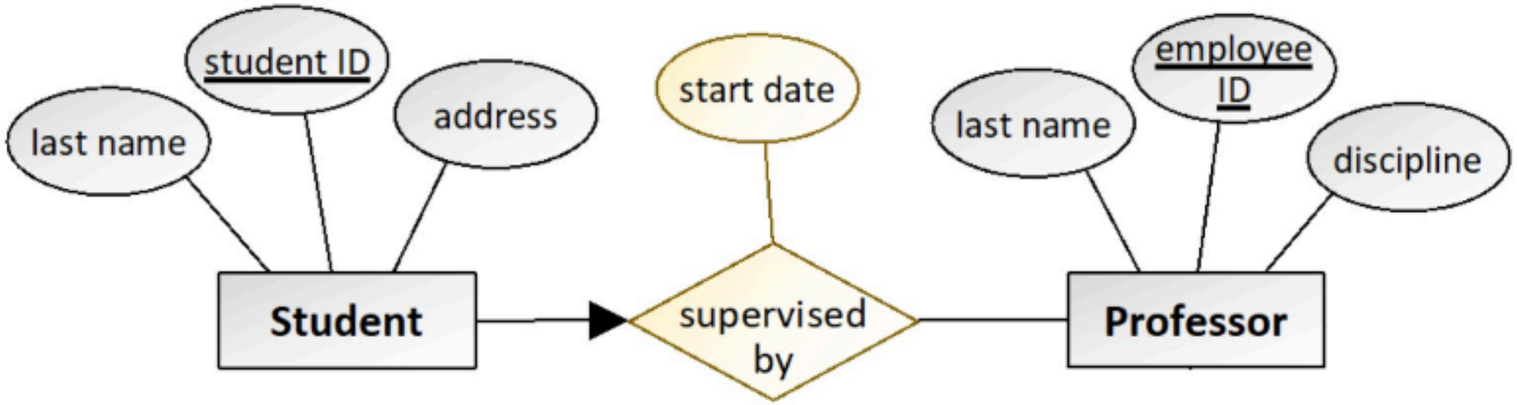


Key Constraints

- “A student is allowed to be supervised by at most one professor, but the professor on the other hand can supervise more than one student”

Key Constraint:

- One-to-many
- One: ?
- Many: ?

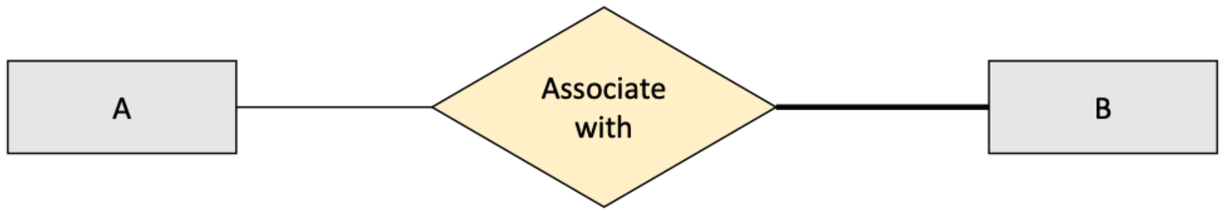


Participation Constraint

- ***Minimum* # of associations** an entity should have in a relationship set

1. **Total** Participation
   - **Every** entity in the entity set **MUST** take part in the relationship

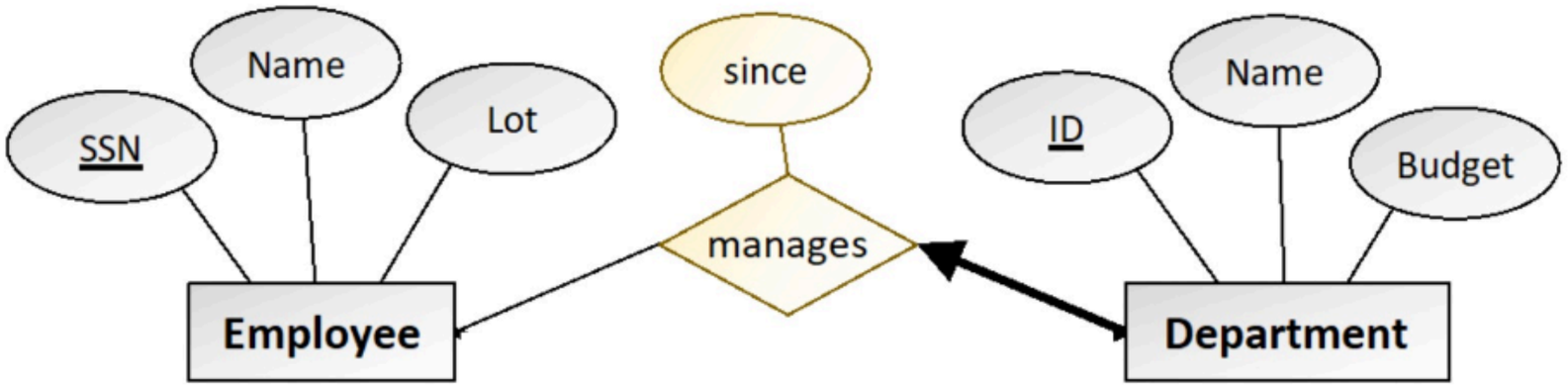
2. **Partial** Participation
   - Participation in relationship is **NOT compulsory**



A: partial participation
B: total participation (bold line)

Participation Constraint

- “All departments must have a manager, however not every employee manages a department”



Q2 -
Case Study

A cinema chain operates a number of cinemas. Each cinema has several screens, numbered starting from 1. The chain keeps track of the size (in feet) and seating capacity of every screen, as well as whether the screen offers the Gold Class experience.

The cinema chain owns hundreds of movie projectors – both film projectors (16 mm and 35 mm) and digital projectors (2D and 3D). The chain stores key information about each projector, namely its serial number, model number, resolution and hours of use. Each movie screen has space for a single projector; technicians must be able to identify which screen each projector is currently projecting onto.

A wide range of movies are shown at these cinemas. The system should keep track of the last time a movie was shown on a particular screen. The marketing department needs to know the movie's title and year of release, along with the movie's rating (G, PG, M, MA15+ or R18+).

Each cinema has a numeric ID, name and address. For cinemas that are not owned outright, the business also keeps track of yearly rent. The system needs to be able to generate weekly activity reports for the chain's chief operating officer.

Q2(a): Entities

- Cinema
- Screen
- Projector
- Movie

Why is “cinema chain” not an entity?

Q2(a): Business Rules

- Each cinema has several screens, numbered starting from 1
- Each movie screen has space for a single projector
- Technicians must be able to identify which screen each projector is currently projecting onto.
- The system should keep track of the last time a movie was shown on a particular screen

Q2(a): Attributes

- **Cinema** (ID, name, address, yearlyRent)
- **Screen** (number, size, seatingCapacity, goldClass)
- **Projector** (format [16 mm film/35 mm film/2D digital/3D digital], serialNumber, modelNumber, resolution, hoursUsed)
- **Movie** (title, yearReleased, rating)

Q2(b): Relationships

- **In Chen’s notation, form relationships b/w entities**

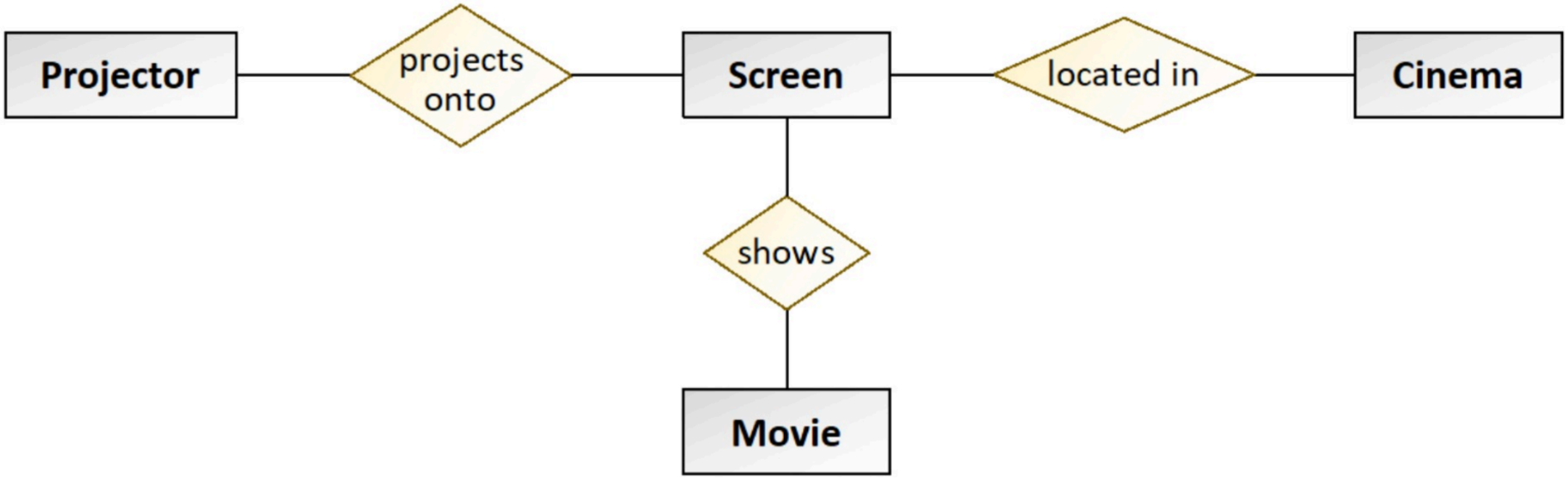
- Cinema
- Screen
- Projector
- Movie

Q2(b): Relationships

- **In Chen’s notation, form relationships b/w entities**

- Things to consider:
  - Don’t worry about attributes / constraints yet
  - How to name relationships

Q2(b): Relationships



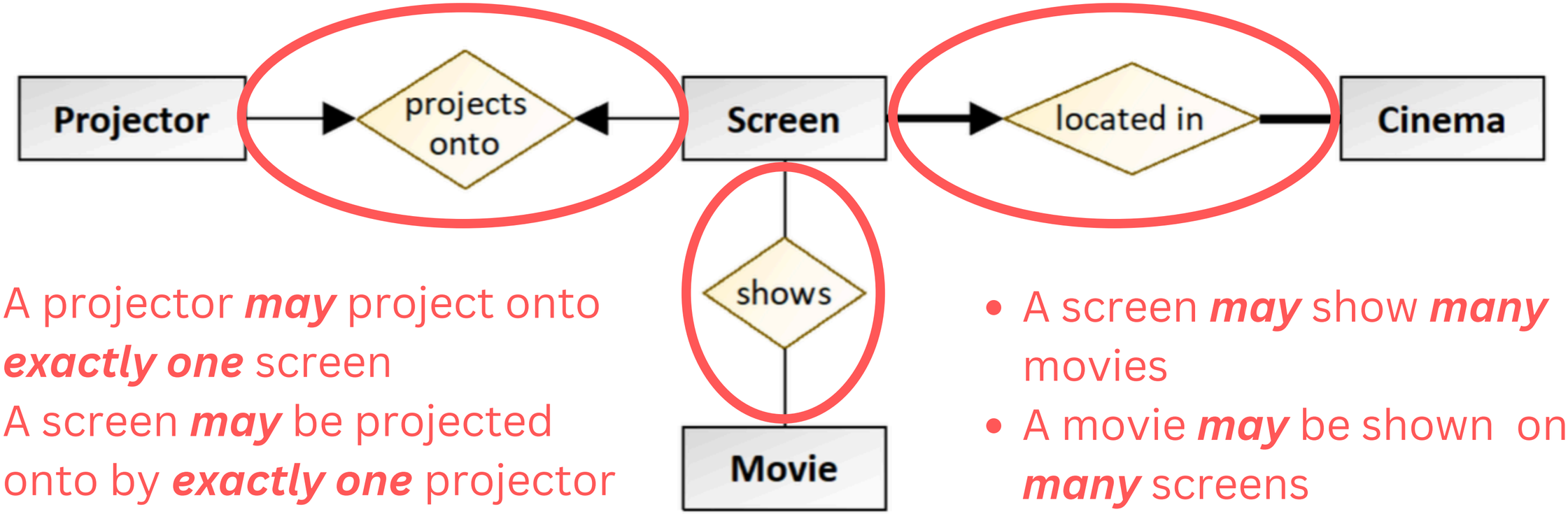
Q2(c): Constraints

- **Add key constraints and participation constraints**

- Recap:
  - KC: Maximum # of associations an entity can have in a relationship set
  - PC: Minimum # of associations an entity should have in a relationship set
  - What should be bolded?
  - Should there be arrows? What direction?

Q2(c): Constraints

- A screen ***must*** be located in ***exactly one*** cinema
- A cinema ***must*** contain ***at least one screen***



- A projector ***may*** project onto ***exactly one*** screen
- A screen ***may*** be projected onto by ***exactly one*** projector

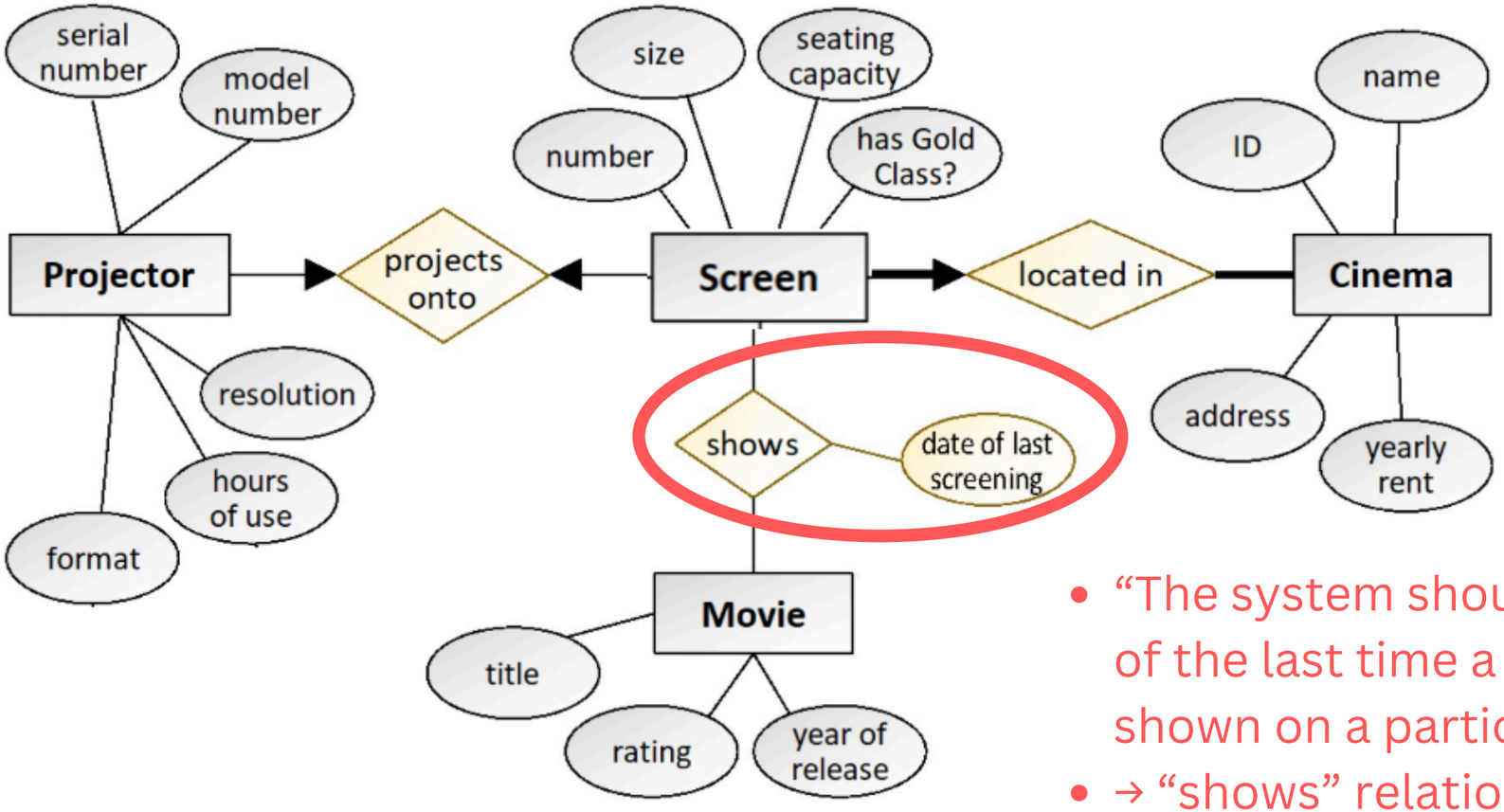
- A screen ***may*** show ***many*** movies
- A movie ***may*** be shown on ***many*** screens

Q2(d): Attributes

- **Add attributes to the entities and relationships**

- **Cinema** (ID, name, address, yearlyRent)
- **Screen** (number, size, seatingCapacity, goldClass)
- **Projector** (format [16 mm film/35 mm film/2D digital/3D digital], serialNumber, modelNumber, resolution, hoursUsed)
- **Movie** (title, yearReleased, rating)

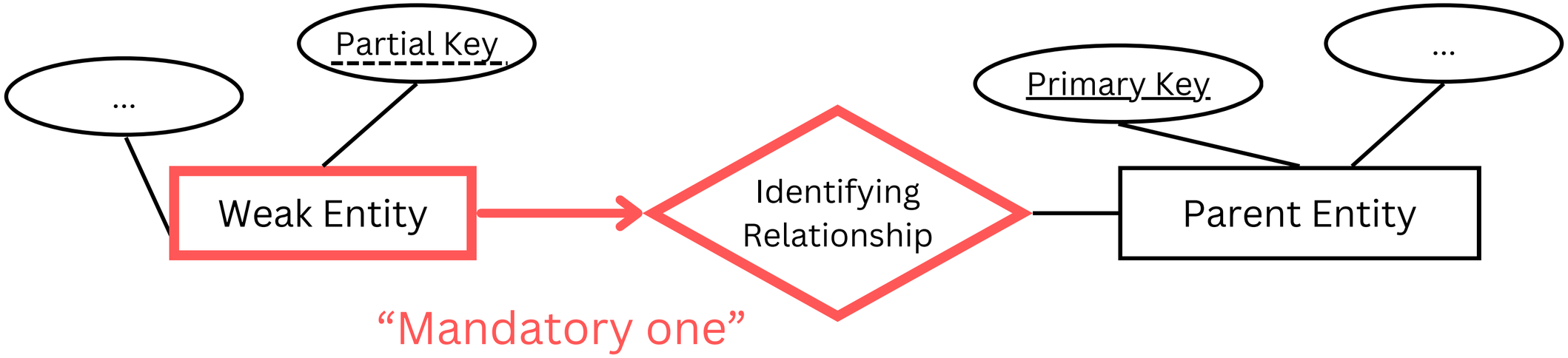
Q2(d): Attributes



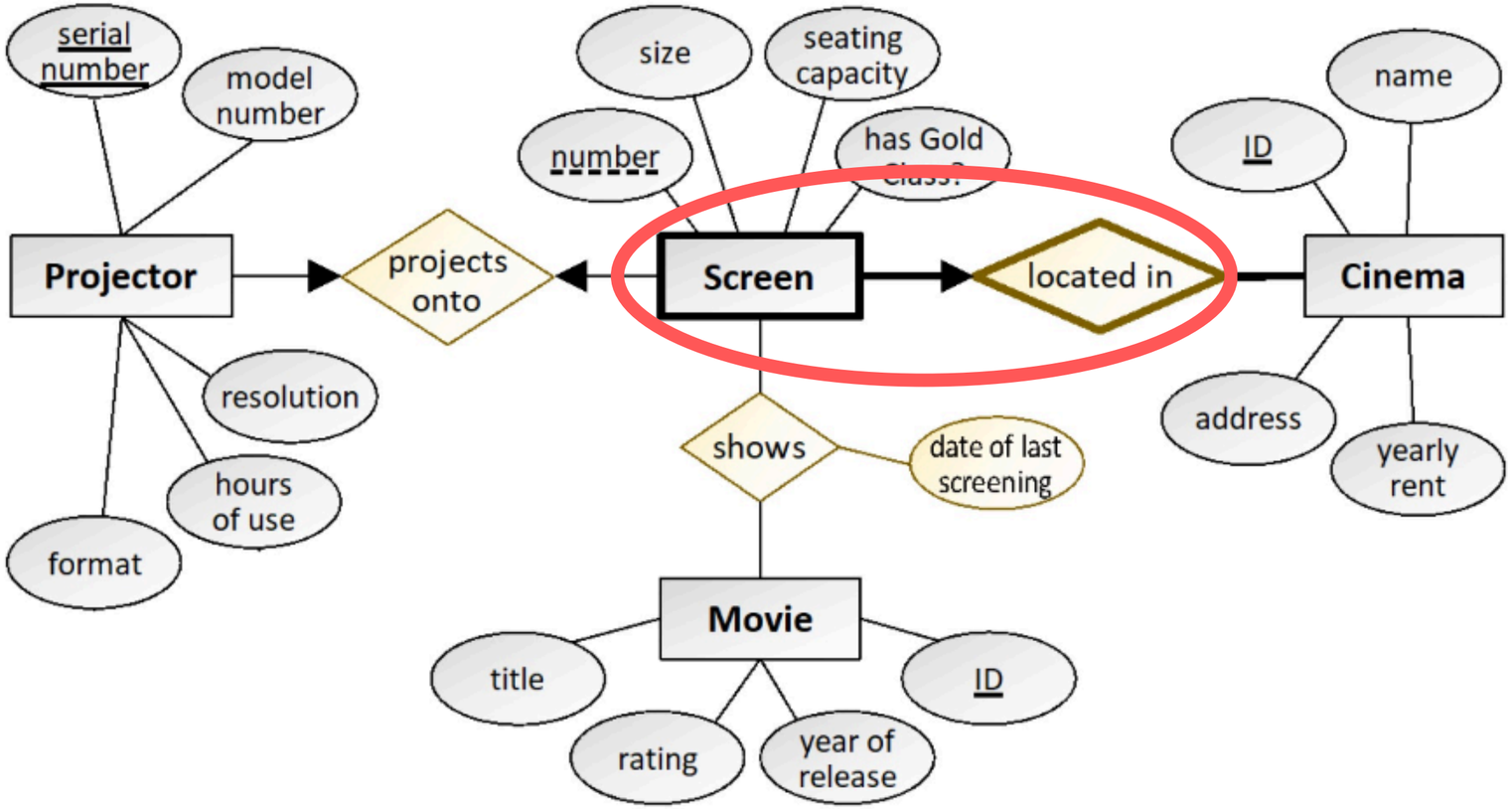
- “The system should keep track of the last time a movie was shown on a particular screen”
- → “shows” relationship (date of last screening)

Q2(e): Weak Entities

- Identify and mark **weak entities**, **identifying relationships** and **key attributes**



Q2(e): Weak Entities

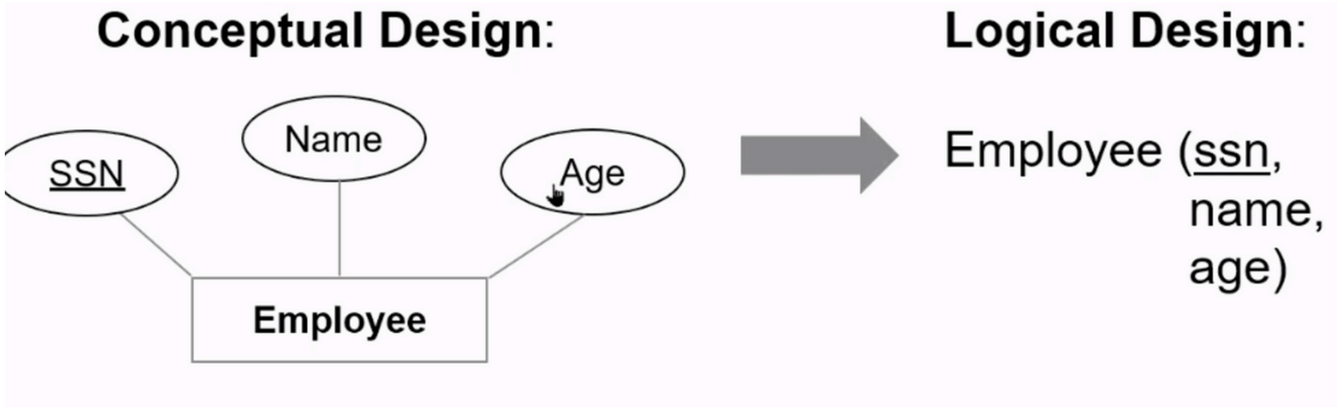


- No cinema, no screen
- Screens can only be identified by the cinema its located in
- **Screen = weak entity, “located in” = identifying relationship**

Q3 - Design Modelling

Q3(a): Conceptual → Logical

- Flatten conceptual entities to **relations**
  - Resolve multi-valued / composite attributes
  - **Resolve relationships** (many-to-many…)
- Entity name → CamelCase
- Attribute name → lower case



Q3(a): Resolve Relationships

1. One-to-one

- Add FK to **either table**
- Prefer to add to total participation end → reduce NULL value

2. One-to-many

- Add FK to the **‘one’** side

3. Many-to-many

- Create new entity → **associative entity**

Q3(a): Resolve Relationships

- One-to-one relationship:

Screen (ScreenNumber, Size, SeatingCapacity, HasGoldClass, ProjectorSerialNumber)
FK

Or you can add FK to the Projector table
(but Screen PK is {ScreenNumber, CinemaID}, see next slide)

Q3(a): Resolve Relationships

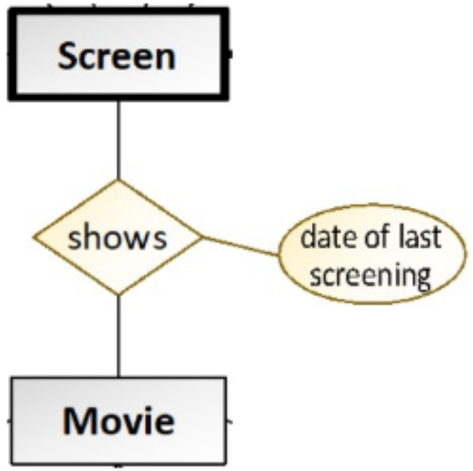
- One-to-many relationship:

To resolve the Cinema-Screen relationship, we add a foreign key on the Screen table. Because this is an **identifying relationship**, this foreign key, "cinema ID", will also be a primary key (known as a "primary foreign key"):

FK FK
Screen (CinemaID, ScreenNumber, Size, SeatingCapacity, HasGoldClass, ProjectorSerialNumber)

Q3(a): Resolve Relationships

- Many-to-many relationship:



FK FK FK
MovieScreening (CinemaID, ScreenNumber, MovieID, DateOfLastScreening)

Q3(a): Logical Model

Cinema (CinemaID, Name, Address, YearlyRent)

Screen (CinemaID [FK], ScreenNumber, Size, SeatingCapacity, HasGoldClass, ProjectorSerialNumber [FK])

MovieScreening (CinemaID [FK], ScreenNumber [FK], MovieID [FK], DateOfLastScreening)

Movie (MovieID, Title, YearOfRelease, Rating)

Projector (SerialNumber, Format, ModelNumber, Resolution, HoursOfUse)

Q3(b): Logical → Physical

- Add data types
  - **Foreign key** MUST have **same data type** as the related Primary Key

- Add NULL / NOT NULL constraint
  - **Primary Keys** should *always* be NOT NULL
  - **Foreign keys** should also be NOT NULL if relationship is **mandatory**

Q3(b): Physical