kdeg
knowledge and data engineering group

Information Management and Data Engineering

CS4D2a – 4CSLL1 – CS3041

Conceptual Database Design

Entity Relationalship Modelling

Conceptual Design

- The process of *conceptual design* uses a high-level *conceptual data model* to create the *conceptual schema* for a database
  - In this case, the Relational Model
  - Part of the requirements analysis process
- Conceptual Schema
  - concise description of the data requirements of the user, including descriptions of entity types, relationships and constraints

Conceptual Design

- The conceptual schema does not include any implementation details
  - hence, can be used to communicate with non-technical users
  - used as a reference to ensure all data requirements are met, and that there are no conflicts
- Part of physical and logical data independence

Entity Relationship Model

- An abstract and high-level conceptual representation of information
- Entity Relationship Diagrams
  - Diagrammatic Notation of the ER Model
- Used to support the conceptual design of databases and help produce the conceptual schema
- Describes data as entities, relationships and attributes

Entities

- The basic object that an ER diagram represents is an *entity*
- An entity is a real world object with an independent existence
  - physical or conceptual
- Each entity has attributes which are the particular properties that describe the real world object

Attributes

- Several types of attributes occur and need to be modeled in the Entity-Relationship Model
  - *simple* versus *composite*
  - *single-valued* versus *multi-valued*
  - *stored* versus *derived*

Simple and Composite Attributes

- Composite attributes can be divided into smaller sub-parts
  - Address
- Attributes that are not divisible are called simple, or atomic, attributes.
  - Movie Certificate, Age…
- Composite attributes can be hierarchical
  - Apartment number, Building number, Street

Single and Multi-Valued Attributes

- Most attributes have a single attribute for each entity
  - PPS number, Age
  - Such attributes are called *single-valued*
- In some cases an attribute can have a set of values for an entity
  - Genre for a Movie, Colour for a Car
  - Some entities may have one value, others multiple
  - Such attributes are called *multi-valued*

Stored versus Derived Attributes

- In some cases, two or more attributes are related
  - Age and BirthDate
- Age can be calculated using today's date and a person's date of birth
  - Age is therefore called a *derived attribute*
  - It is said to be *derivable from* BirthDate
  - BirthDate is called a *stored attribute*

Stored versus Derived Attributes

- Some attributes can be derived from information in related entities, rather than attributes
  - If a number_of_employees attribute was associated with a CINEMA entity
  - This could be derived by totaling the number of employee entries stored in the EMPLOYEE entity

Entity Types and Sets

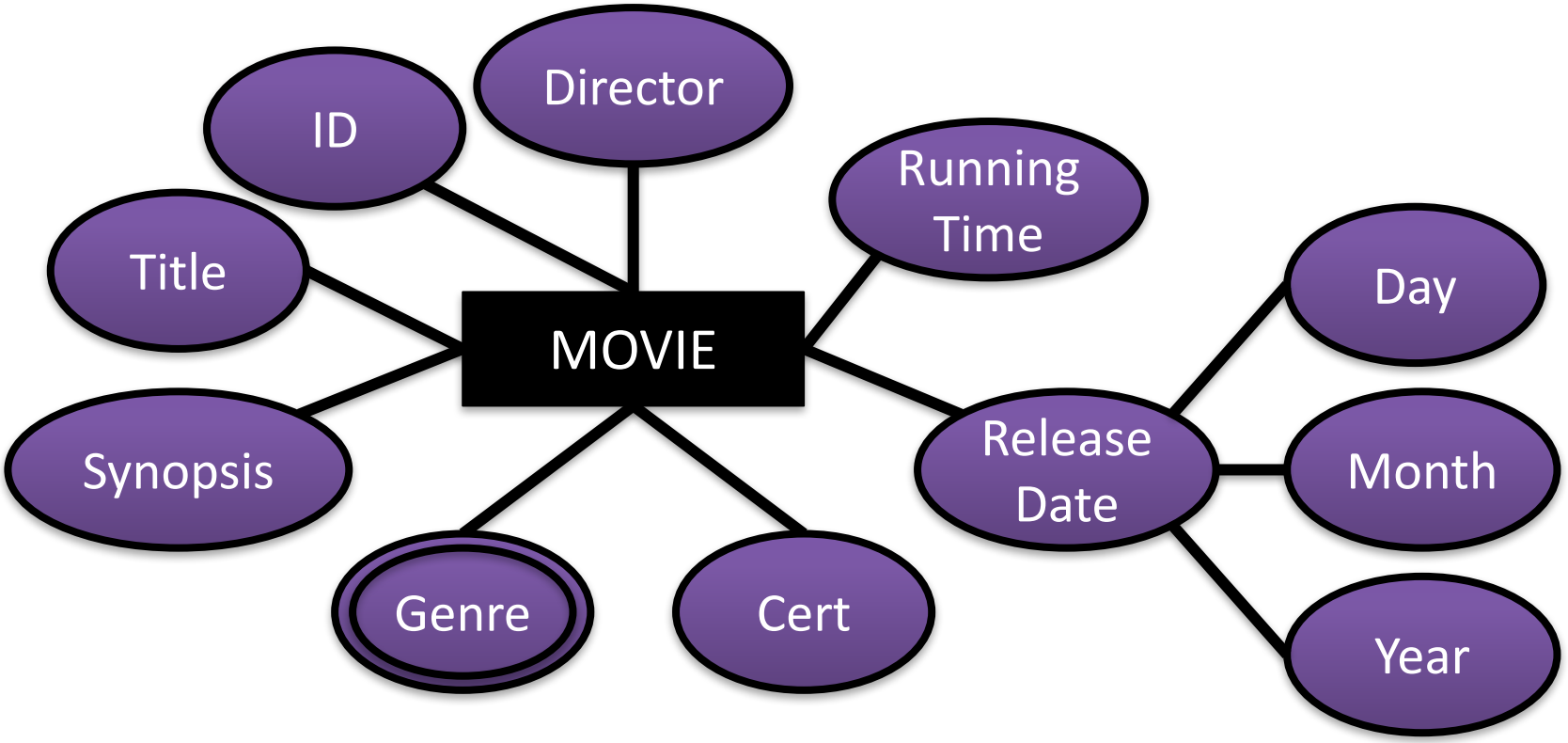
- ER diagrams don't show single instances of entities (or relations)
- They show *entity types*
  - An entity which is identified by its name and attributes
  - In a cinema database, MOVIE could be an entity type.
  - All movie entities share the same attributes but each instance has its own values for each attribute
- The collection of all instances of a particular entity type in a database is called an *entity set*

Entity Types and Sets

Entity Type	EMPLOYEE Name, Age, Salary	DEPARTMENT Name, Location, Manager
Entity Set	John Smith, 55, 80,000 Fred Brown, 40, 30,000 Judy Clark, 25, 45,000 … … …	Research, Dundrum, John Smith Sales, Dublin 1, Judy Clark Online, Finglas, Bob King … … …

Entity Types

- ER diagram notation of entity types:

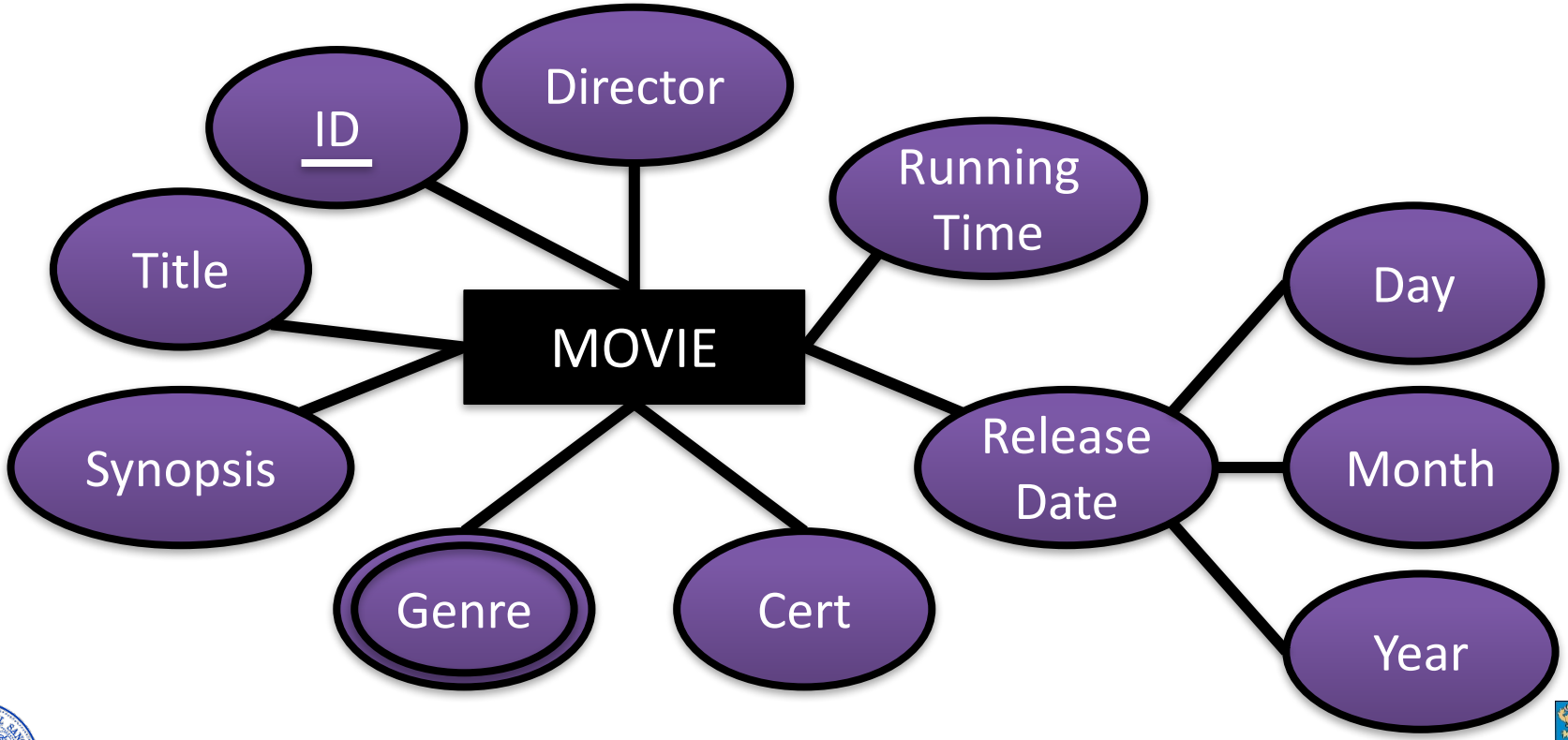


Key Attributes

- Each entity type usually has one or more attributes whose values are unique for each instance in the entity set
  - An attribute whose values are used to uniquely identify each entity is called the *key attribute*
    - ISBN, PPS, Student Number
  - More than one attribute can be used to form the key. In this case the combination must be unique
    - Composite key attribute

Key Attributes

- Key attributes are represented by underlining the attribute

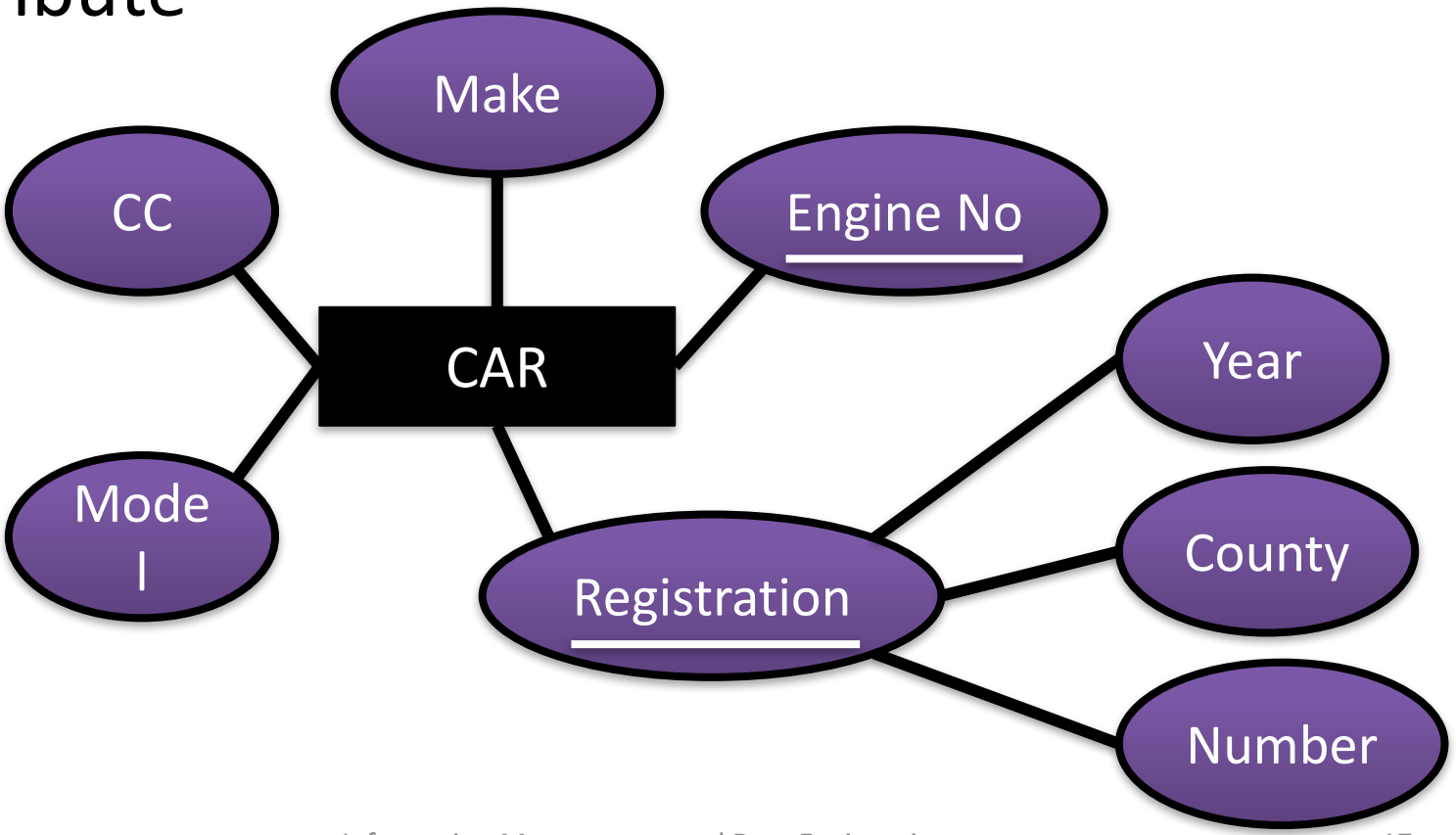


Key Attributes

- Specifying the key attribute places a uniqueness constraint on the entity type
  - this uniqueness constraint must hold for every instance of an entity in the entity set
- The key constraint is derived from the real world requirements that the database represents

Key Attributes

- Some entity types have more than one key attribute



Exercise

- Using the Entity Relationship diagram notation introduced on the previous slides, list all the entity types, and their associated attributes, required to model a chain of Movie Theatres

Exercise

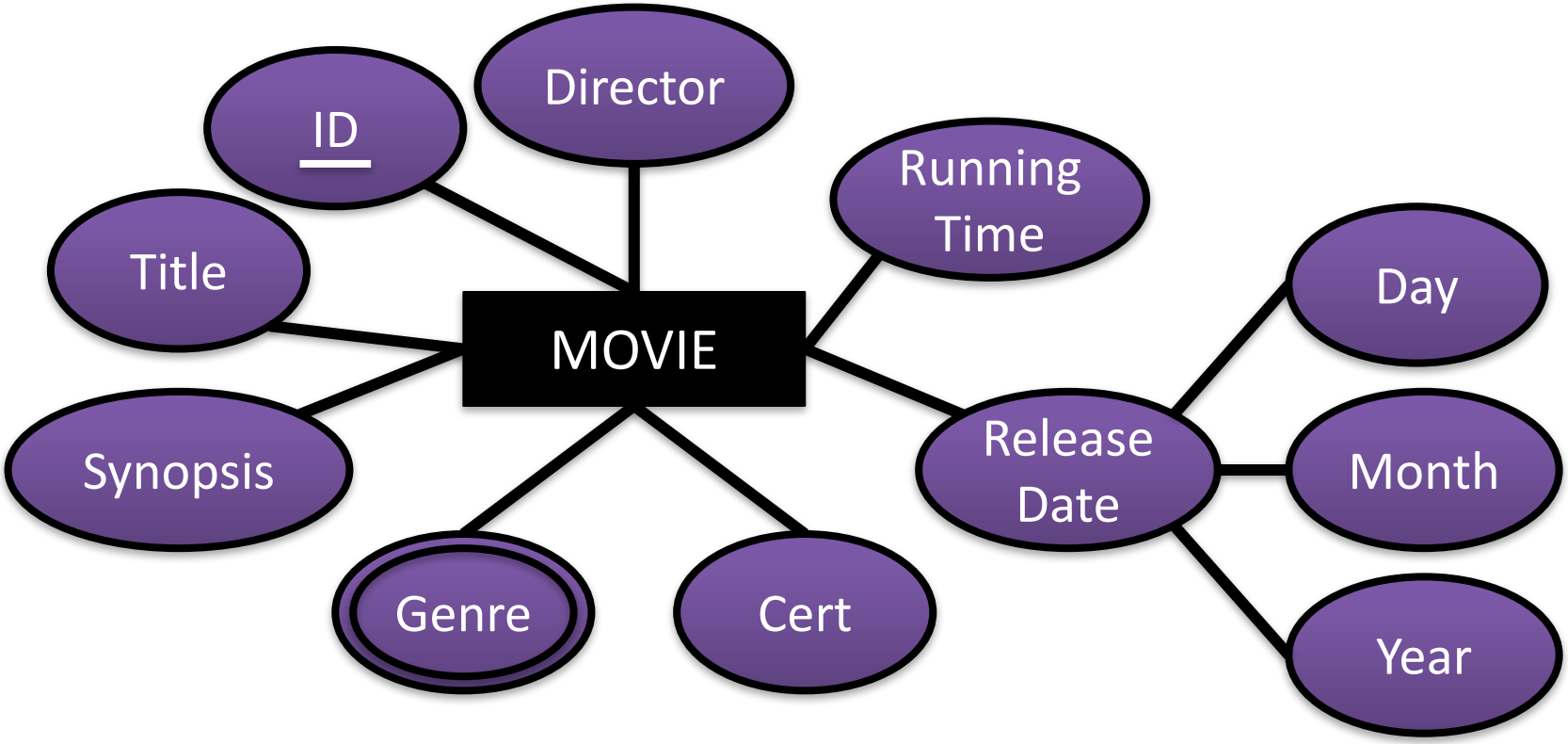
MOVIE

THEATRE

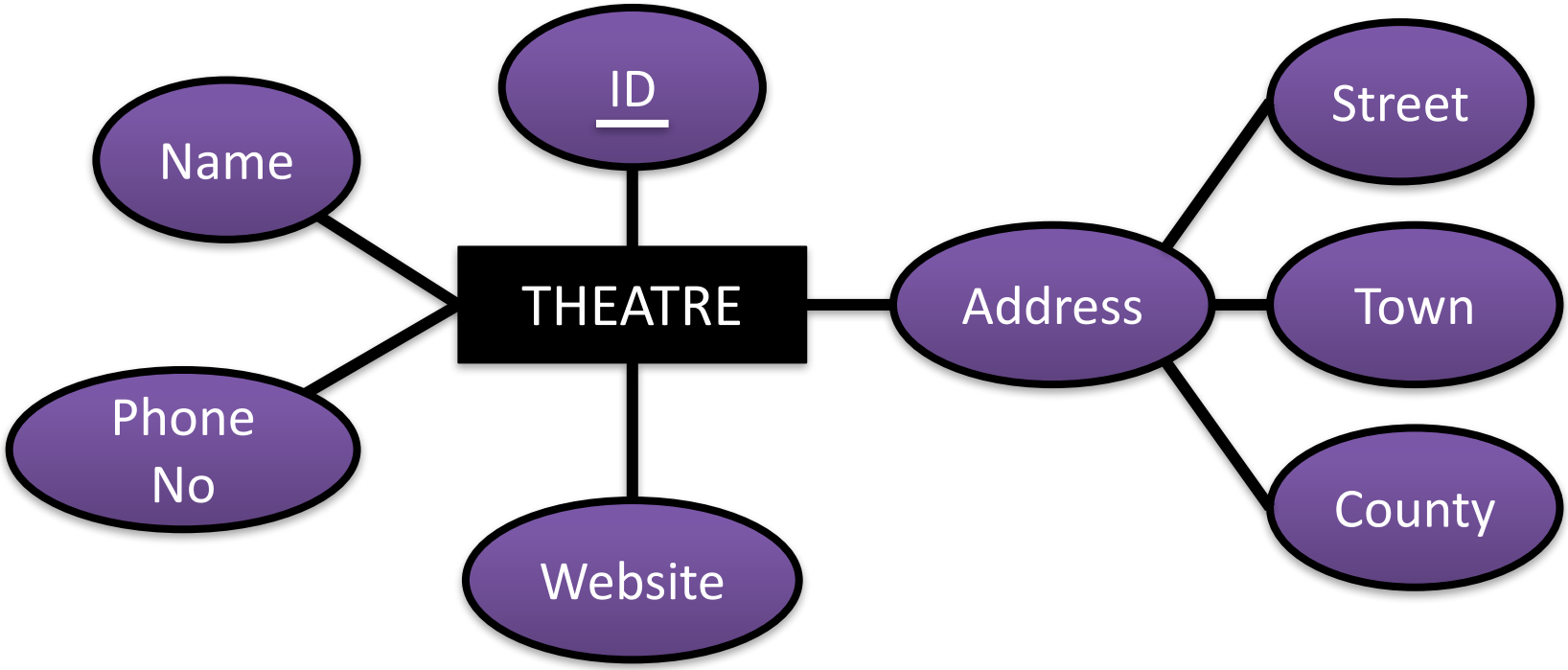
SCREEN

SCREENING

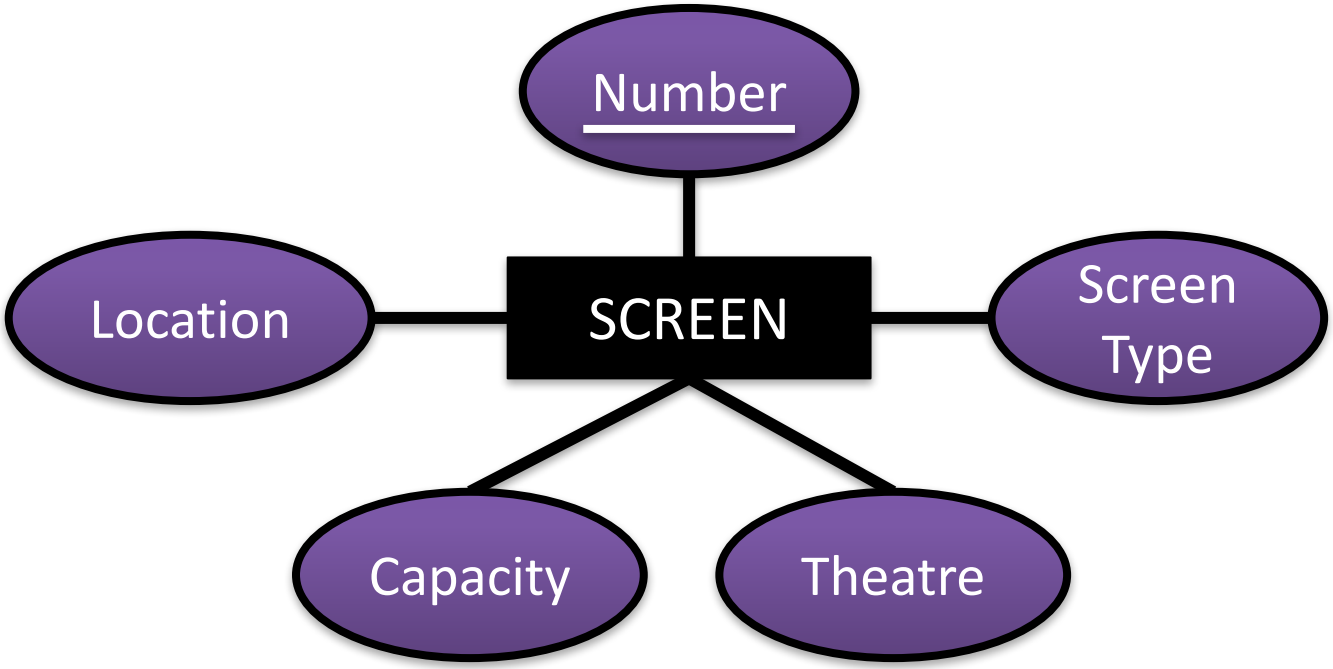
Exercise


Director
ID
Running Time
Title
MOVIE
Day
Release Date
Month
Synopsis
Genre
Cert
Year

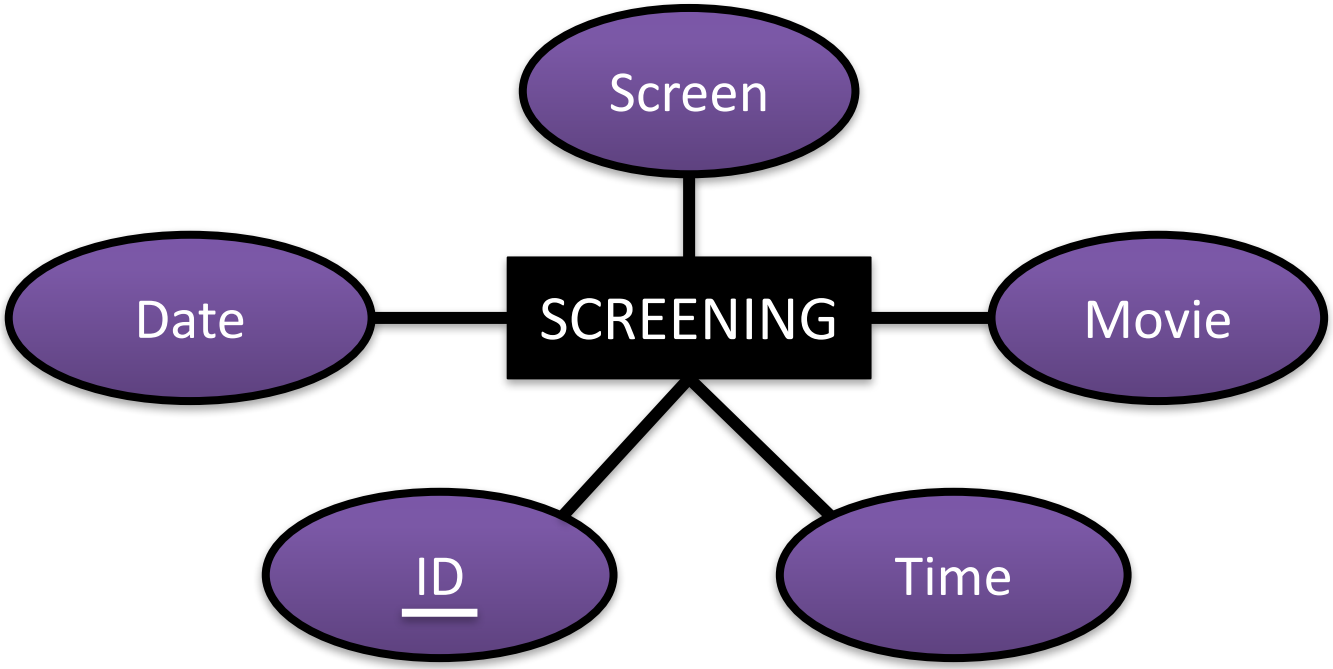
Exercise


ID
Name
Street
THEATRE
Address
Town
Phone No
County
Website

Exercise



Exercise



Relationships

- A *relationship* captures how two or more entity types are related to one another.
- Whenever an attribute of an entity type refers to another entity type, a relationship exists
  - There are a number of implicit relationships in our movie example – SCREEN and THEATRE, SCREENING and MOVIE, SCREENING and SCREEN
  - In the ER model, these references should not be represented as attributes, but as relationships

Relationships

- A relationship can be informally thought of as a verb, linking two or more nouns from the world you are trying to model
  - a “manages” relationship between an employee and a department
  - a “performs” relationship between an artist and a song
  - a “bores” relationship between a lecturer and a student
  - a “proved” relationship between a mathematician and a theorem

Relationship Types and Sets

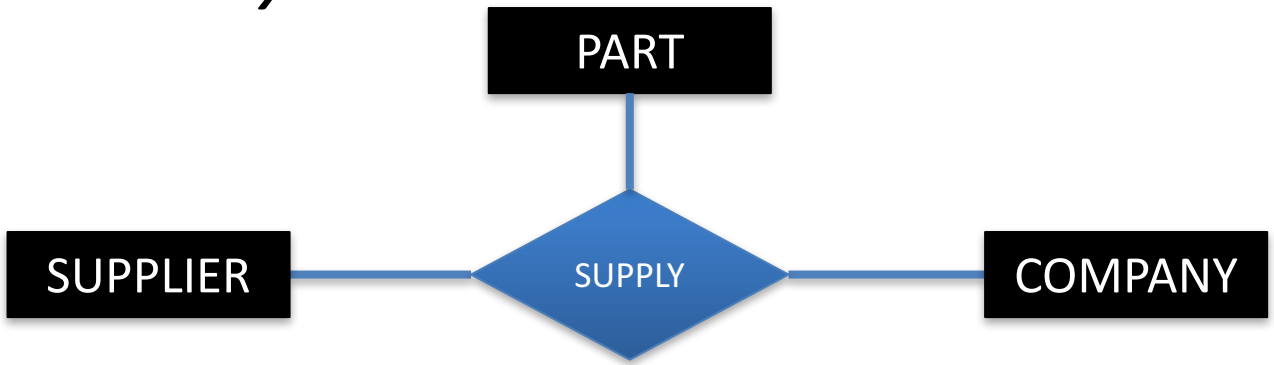
- As with entities, relationships have a *relationship type*, which is illustrated in an ER diagram
  - The collection of all instances of a particular relationship type in a database is called a *relationship set*
- Related entity types are said to *participate* in a relationship type
  - Each relationship instance, r_i, is an association of entities, where the association includes exactly one entity from each of the participating entity types

Relationship Types

- ER diagram notation of relationship types

Binary and Ternary Relationships

- The degree of a relationship type is the number of entity types that participate
  - The SHOW relationship is of degree two
- Relationship types of degree two are called *binary*, relationship types of degree three are called *ternary*



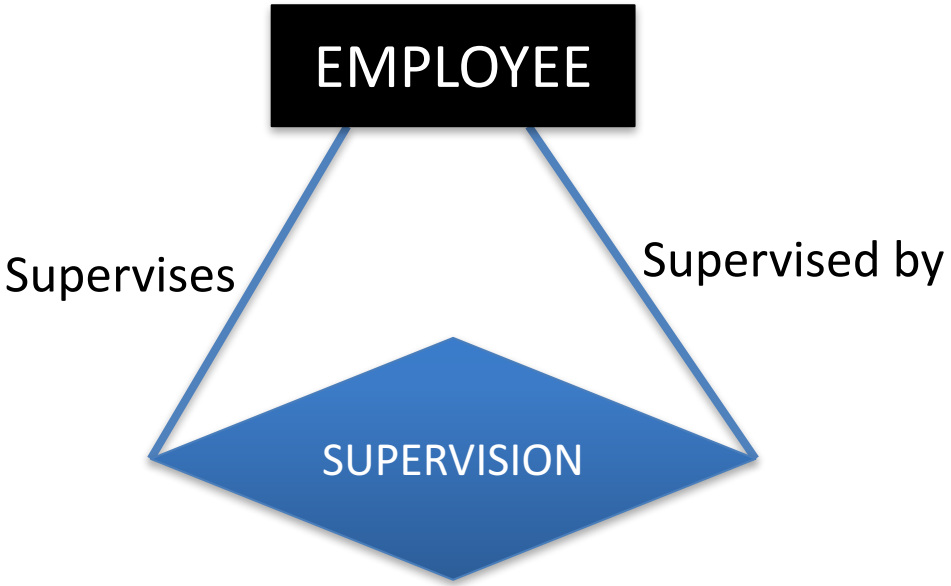
Relationship Roles

- Each entity type that participates in a relationship type plays a particular role
- A role name can be optionally added to an ER diagram to clearly identify what the relationship means

Recursive Relationships

- In some cases the same entity type participates more than once in a relationship type, in different roles
  - Such relationships are called *recursive relationships*
- In this case the relationship roles are key in distinguishing the role each participating entity plays

Recursive Relationships



- The SUPERVISION relationship relates an employee to a supervisor, where both employee and supervisor are members of the same entity set

Relationship Constraints

- Constraints limit the possible combination of entities that can participate in a relationship
- These constraints are determined by the real world requirements that are being modeled
- Two main types of relationship constraint
  - Cardinality Constraints
  - Participation Constraints

Cardinality Constraints

- Specify the *maximum* number of relationship instances that an entity can participate in
- Consider our SHOW relationship type
  - Cardinality ratio for MOVIE:SCREENINGS is 1:N
  - Each movie can be shown in many screenings
  - Each screening shows a maximum of one movie

Cardinality Constraints

- Possible cardinality ratios for binary relationships
  - 1:1 – One to One
  - 1:N – One to Many
  - M:N – Many to Many

Cardinality Constraints

- 1:1 One to One

- M:N Many to Many

Participation Constraints

- Specify the *minimum* number of relationship instances that an entity can participate in
- Two types of participation constraint
  - *total participation*
  - *partial participation*
- Formally, participation constraints specify if the existence of an entity depends on its being related to another entity via the relationship type

Participation Constraints

- Company policy states that every Employee MUST work for a department

- Then, an employee entity can only exist if they participate in at least one WORKS relationship between EMPLOYEE and DEPARTMENT

Participation Constraints

- The participation of EMPLOYEE in WORKS is called *total participation*
  - every entity in the total set of employee entities must be related to a department entity via the WORKS relationship type
- Total participation is also called *existence dependency*

Participation Constraints

- In the same company, every employee will not be responsible for managing a department

- Therefore, an employee entity can exist if they don't participate in the MANAGE relationship between EMPLOYEE and DEPARTMENT

Participation Constraints

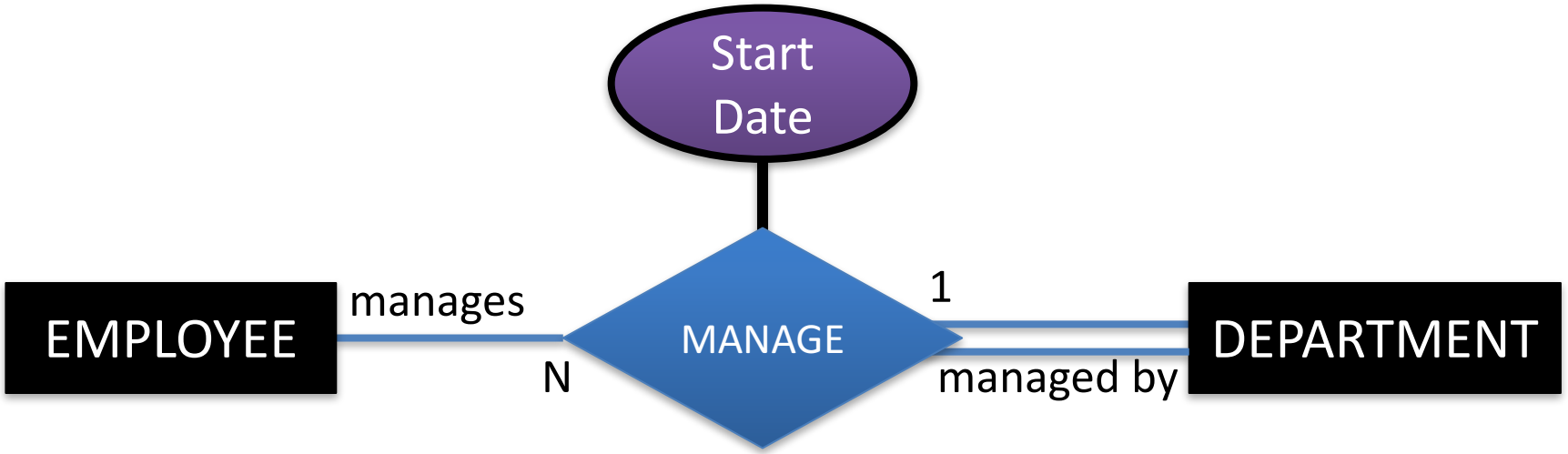
- The participation of EMPLOYEE in MANAGE is called *partial participation*
  - some, but not all, of the total set of employee entities are related to a department entity via the MANAGE relationship type
- Together, cardinality constraints and participation constraints are referred to as the *structural constrains* of a relationship type

Participation Constraints

- In ER diagrams, total participation is represented by a double line connecting the participating entity type to the relationship type
- Partial participation is represented by a single line

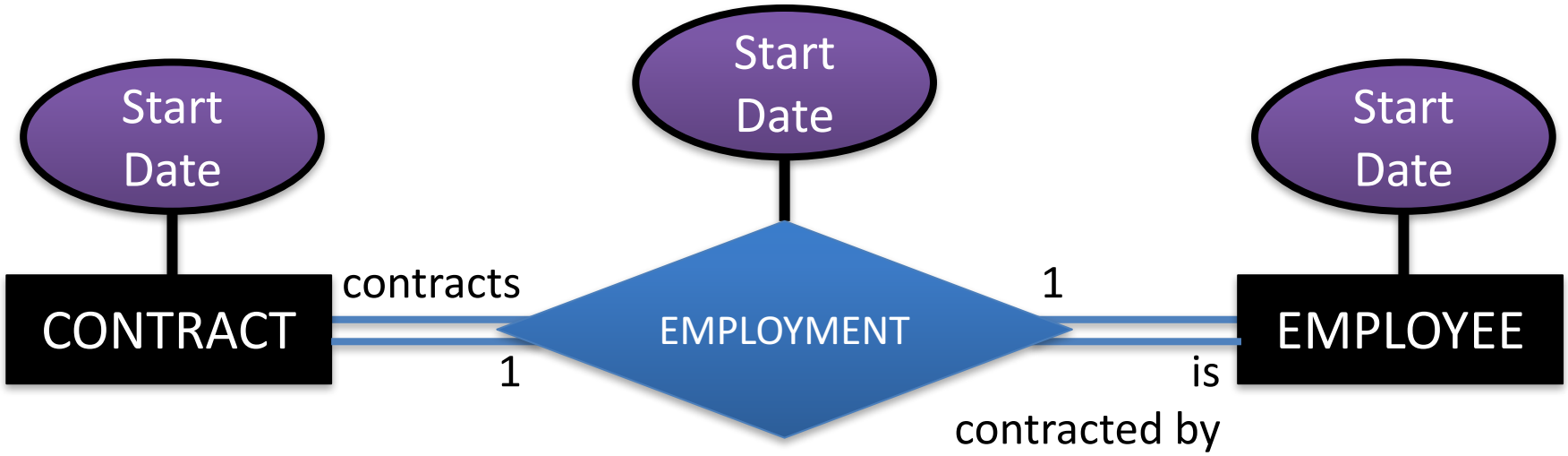
Relationship Type Attributes

- In the entity relationship model, relationship types can have attributes, similar to entity types



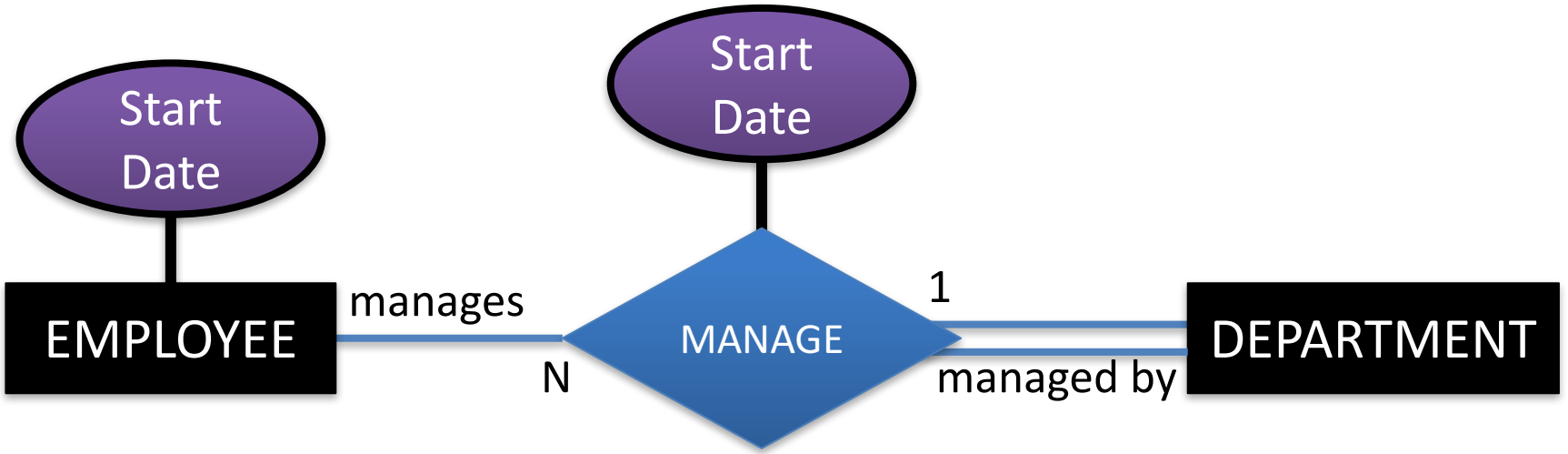
Relationship Type Attributes

- These relationship type attributes can be migrated to participating entities for relationships with a cardinality of 1:1 or 1:N



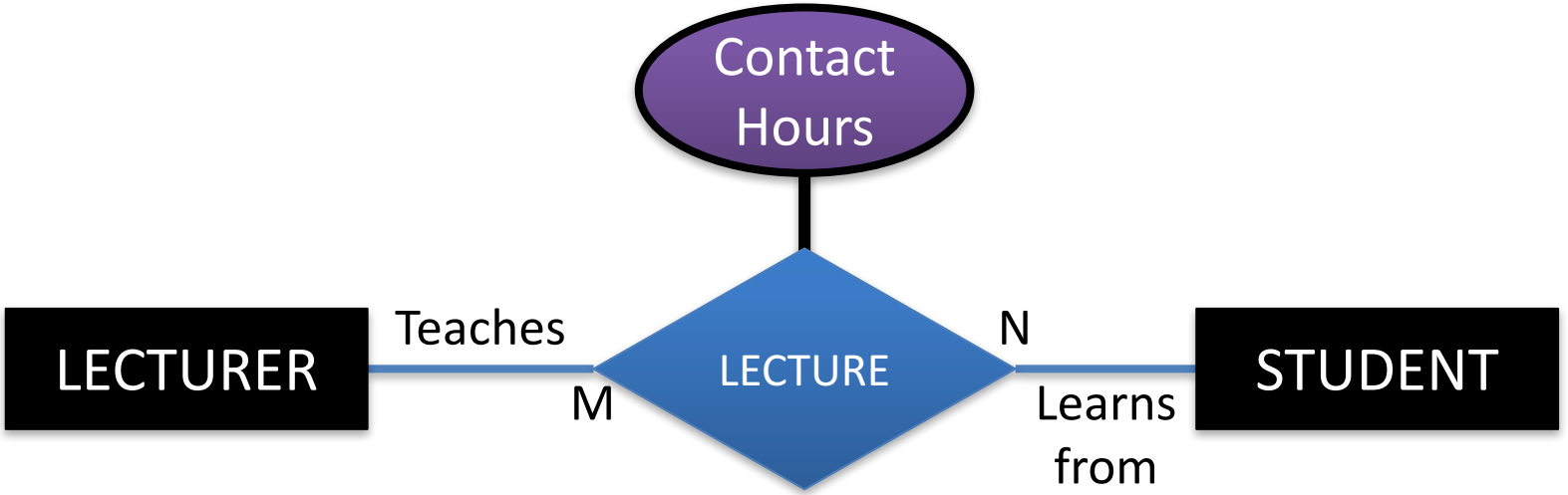
Relationship Type Attributes

- In relationship types with 1:N cardinality ratios, the relationship attribute can only be migrated to the N side of the relationship

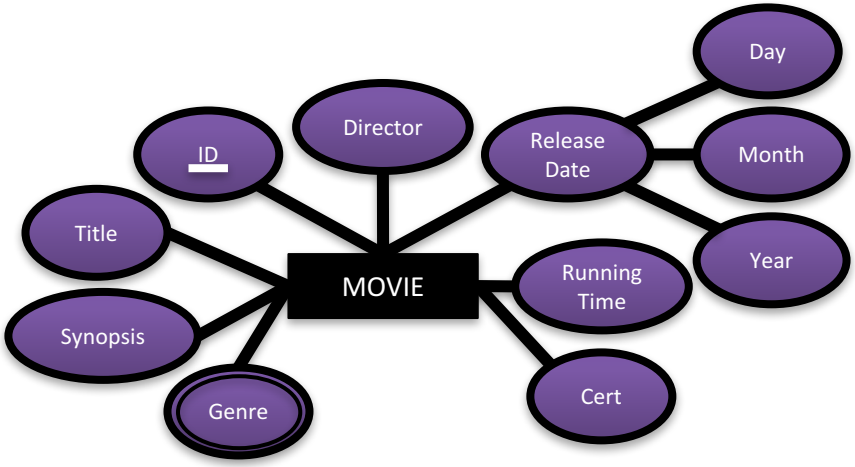


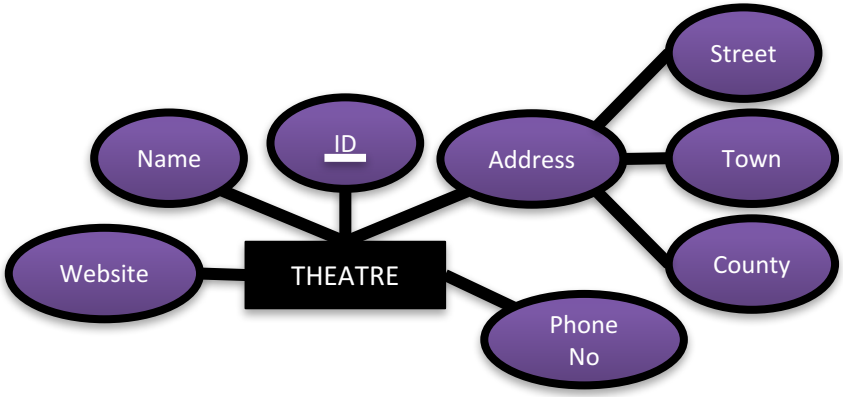
Relationship Type Attributes

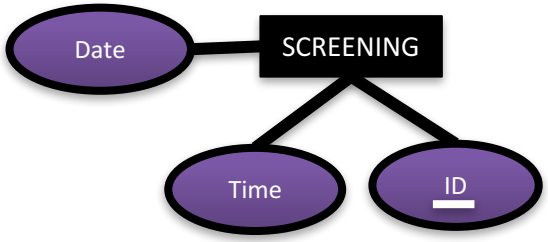
- In relationship types with N:M cardinality ratios, the attribute cannot be migrated and remains specified as a relationship attribute

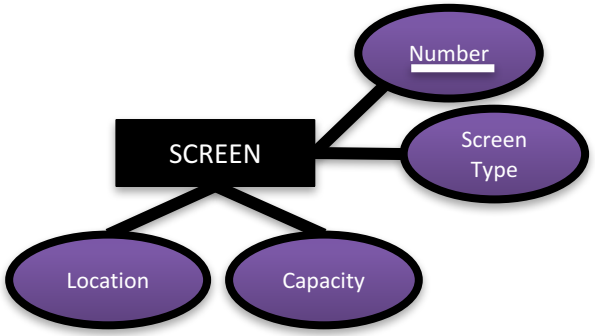


Cinema Example

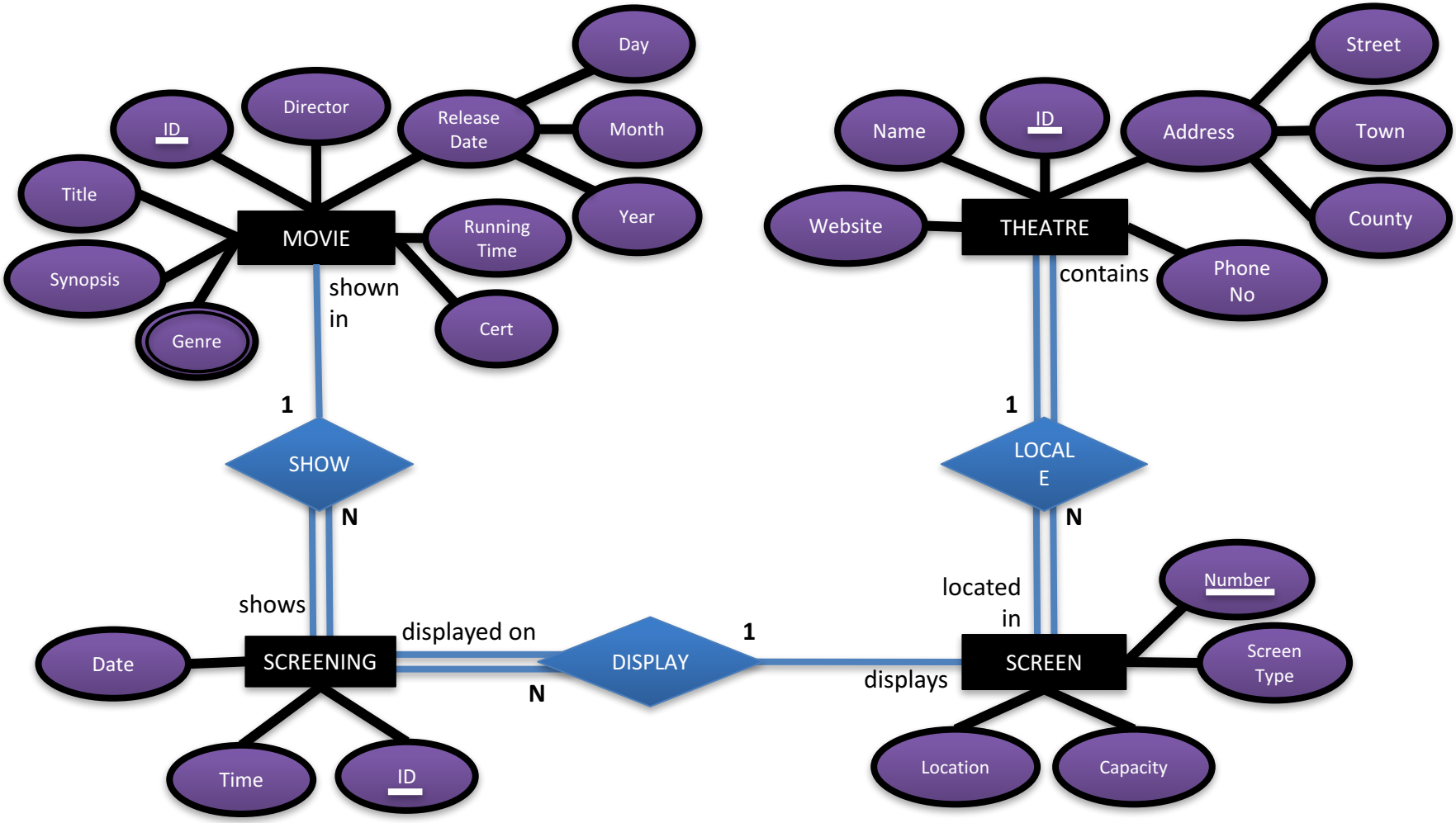








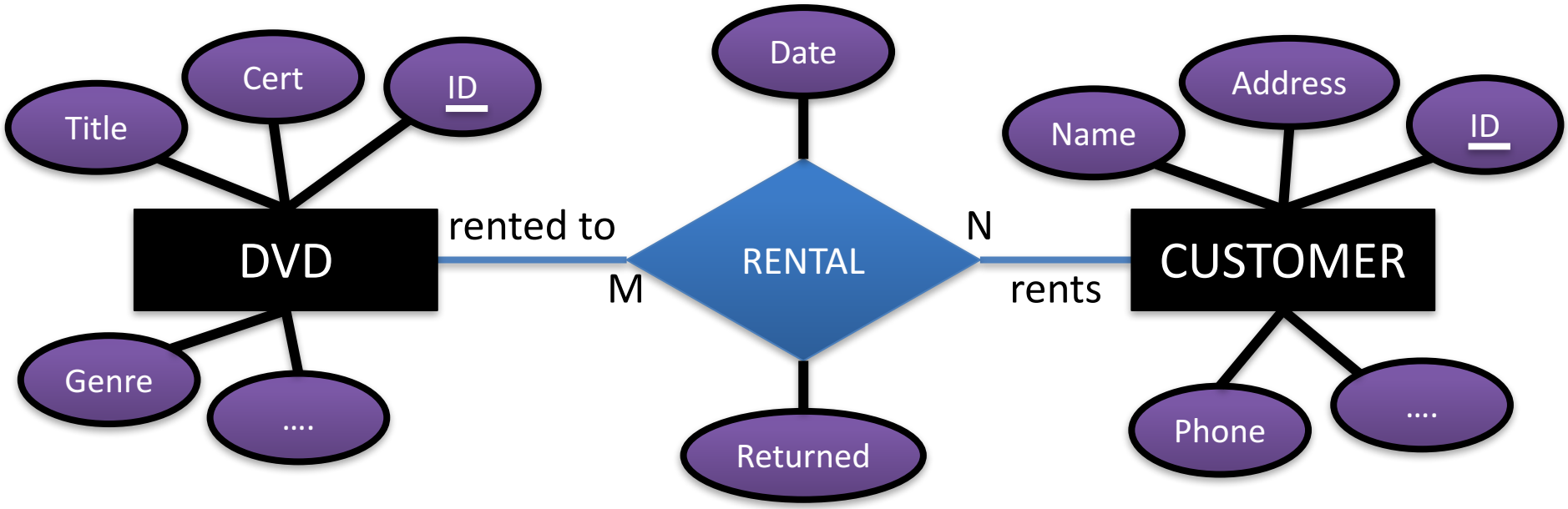
Cinema Example



Example

- Suppose you wish to model a DVD Rental shop called LotsoVision:
  - The DVD Rental shop needs to store information about the DVD's it rents (titles, cert, genre of DVD etc.), the customers that rent the DVDs (Name, Address, Phone, etc.)
  - A customer can rent many DVDs from the LotsoVision store. Each of LotsoVision's DVDs can be rented to many customers
- Draw the ER Diagram representing the entities, Atributes and Rental relationship.

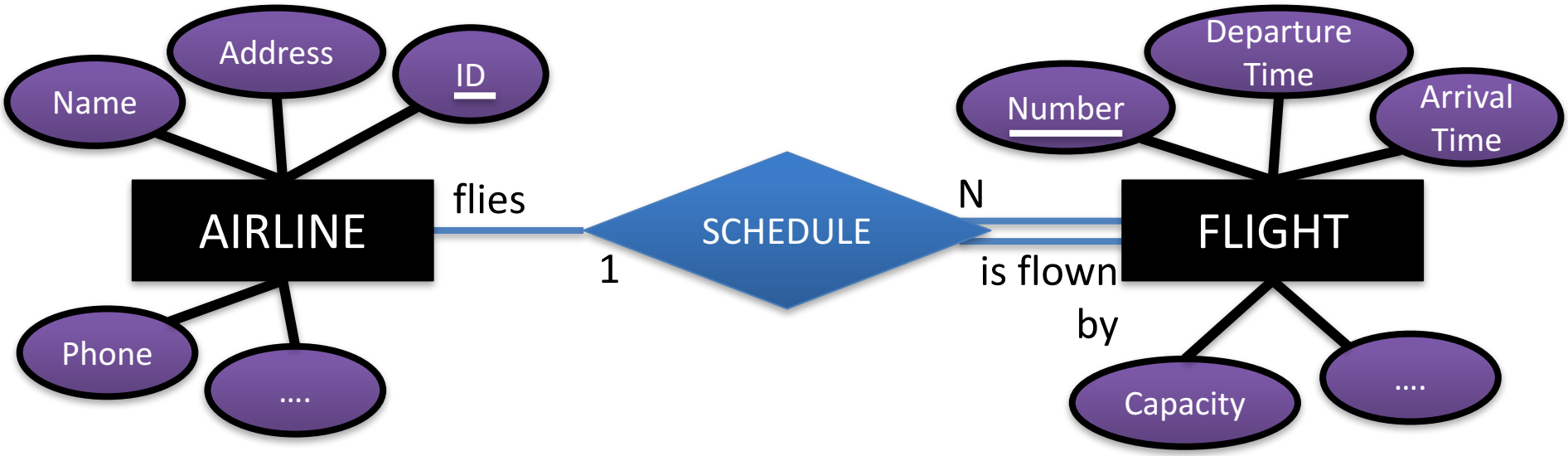
Example


Date
Cert
ID
Address
Title
Name
ID
rented to
N
DVD
RENTAL
CUSTOMER
M
rents
Genre
....
Returned
Phone
....

Example

- Suppose you want to represent an Airline company that schedules many flight.
  - Represent the Airline and flight entities and the schedule relationship using whatever attributes you think appropriate for each.
  - Note: An airline company flies many flights, but each flight is flown by only one airline

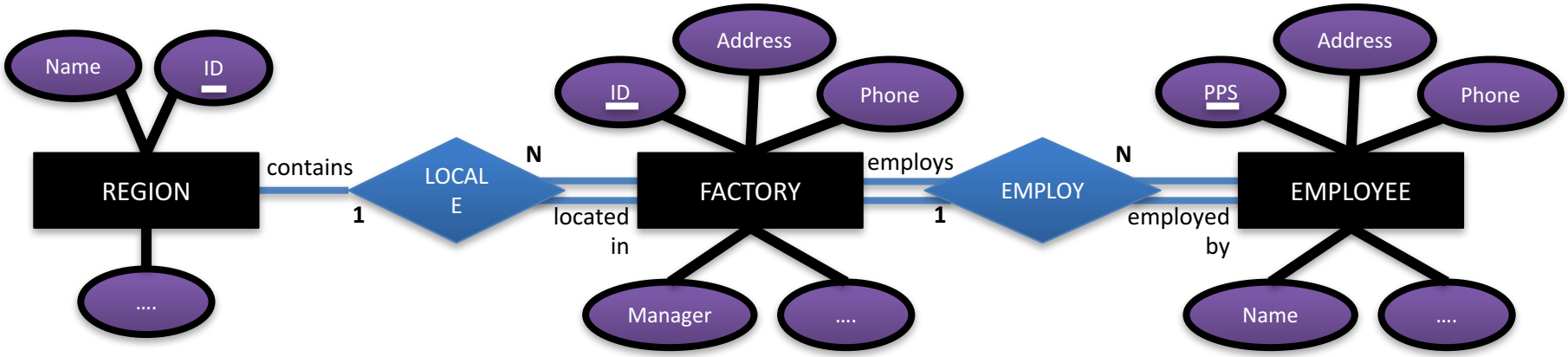
Example


Address
Name
ID
AIRLINE
flies
1
SCHEDULE
N
is flown
by
FLIGHT
Departure Time
Number
Arrival Time
Phone
....
Capacity
....

Examples

- The Globex Corporation operates many factories. Each of these factories is located in a region. Each region can have more than one Globex factory. Each factory employs many employees, but each of these employees is employed by only one factory.

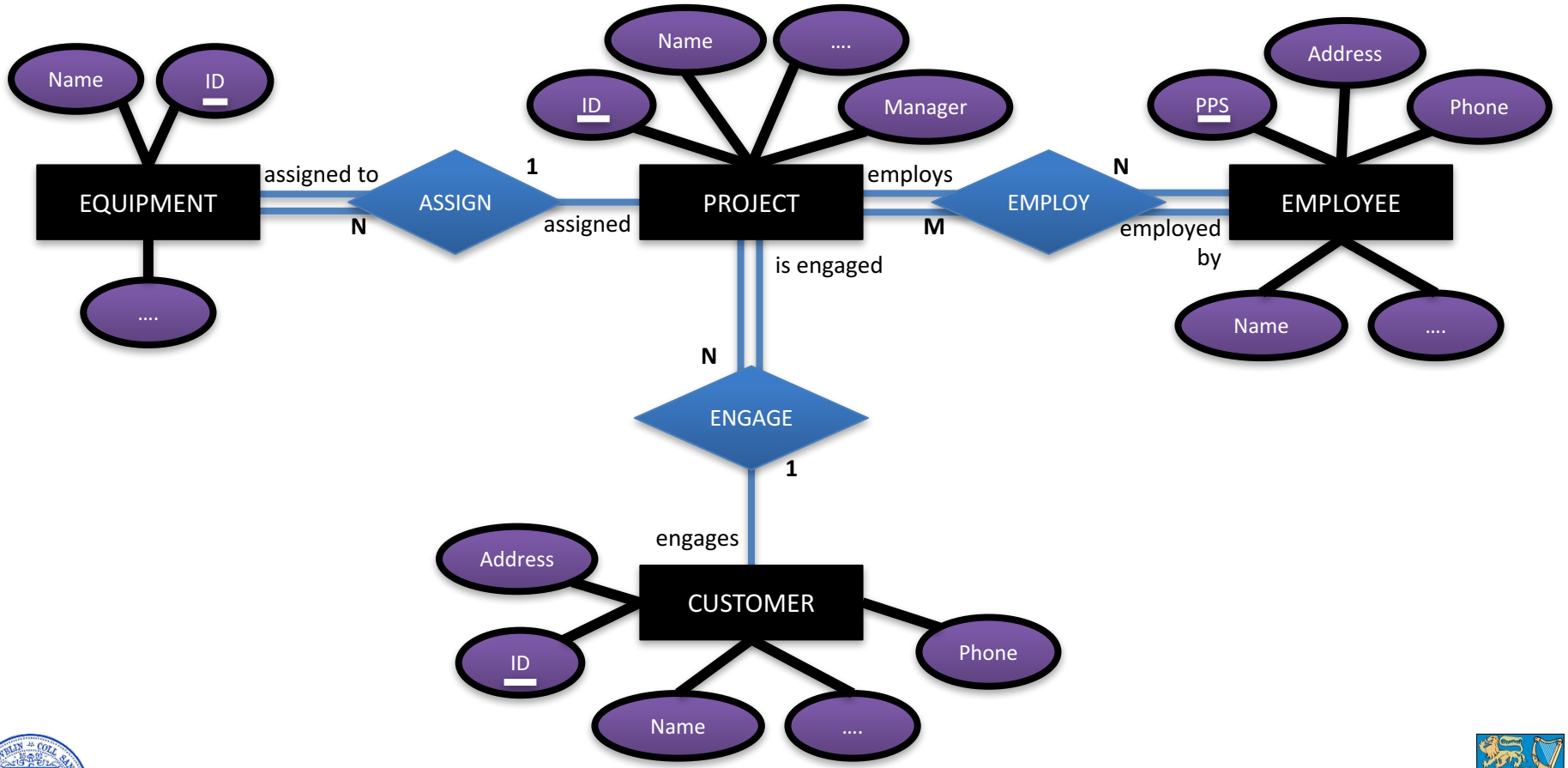
Example


Name
ID
REGION
....
contains
1
LOCAL
E
N
located
in
ID
Address
Phone
FACTORY
Manager
....
employs
1
EMPLOY
N
employed
by
PPS
Address
Phone
EMPLOYEE
Name
....

Exercise

- Dodgy Builders Construction Company is a building contractor that specialises in mid-range homes.
  - Dodgy Builder's has a number of customers and employees, handles a series of projects and owns lots of building equipment.
  - A customer can engage the company for work on more than one project.
  - Dodgy Builder employees can often work on more than one project at a time.
  - Building equipment is always assigned to only one project at a time.

Exercise


Name
ID
EQUIPMENT
....
assigned to
N
ASSIGN
1
assigned
Name
....
ID
Manager
PROJECT
employs
M
EMPLOY
N
employed by
Address
PPS
Phone
EMPLOYEE
Name
....
is engaged
N
ENGAGE
1
engages
Address
ID
CUSTOMER
Phone
Name
....

Entity Modelling Exercise

- Suppose we are modelling a customer database for a bank. For each customer the bank stores, the customer's id, name (firstname, middle initial lastname), phone numbers, date of birth, age, and the customer's address which includes the streetname, city and country name.
- Draw the entity relationship diagram (ER) for this entity

Modelling Exercise 2

- Suppose we are modelling a MOVIE database in which data is recorded about the movie industry. The data requirements are summarised as follows:
- Each movie is identified by title and year of release. Each movie has a length in minutes. Each has a production company and each is classified under one or more genres (e.g. horror, action…) Each movie has one or more directors and one or more actors appear in it. Each movie also has a plot outline. Finally each movie has zero or more quotes, each of which is spoken by a particular actor appearing in the movie.

Relationships as Attributes

- It is sometimes convenient to think of a relationship type in terms of attributes
  - as when initially modeling the entity types
- With binary relationships there are always two options for representing it as an attribute
  - Consider the “shows” relationship
  - We could use an attribute in Screening that refers to the Movie ID for the movie in question
  - or, we could use a multi-valued attribute in Movie that refers to all the screenings of that movie

Weak Entity Types

- In ER diagrams, entity types which don't have a key attribute are called *weak entity types*
- Entities of this type are identified by their relationship to specific entities from other entity types
  - this other entity type is called the identifying entity type
  - the relationship type that relates a weak entity type to its identifier is called the identifying relationship

Weak Entity Types

- A weak entity type always has a total participation constraint with relation to its identifying relationship
  - as a weak entity cannot be identified without its identifier entity