Logical Database Design
Mapping ERD to Relational - Part 1

Dr. Jeevani Goonetillake

UCSC

Objectives of Mapping

- How to address features from a local conceptual model that are not compatible with the relational model.

- How to derive a set of relations from a logical data model (ER/EER diagram).

Transforming ERD into Relations

Transforming (mapping) E-R diagrams to relations is a relatively straight forward process with a well-defined set of rule

Step 1: Map Strong Entities
2: Map Weak Entities
3: Map Binary Relationships
4: Map Unary Relationships
5: Map Complex Relationships
6: Map Supertype/Subtype Relationships
7: Map Categories

Transforming ERD into Relations

- Refine the local conceptual data model in addressing the features that are not compatible with the relational model. This involves:
  - M:N binary relationship types;
  - M:N recursive relationship types;
  - Complex relationship types;
  - Multi-valued attributes.

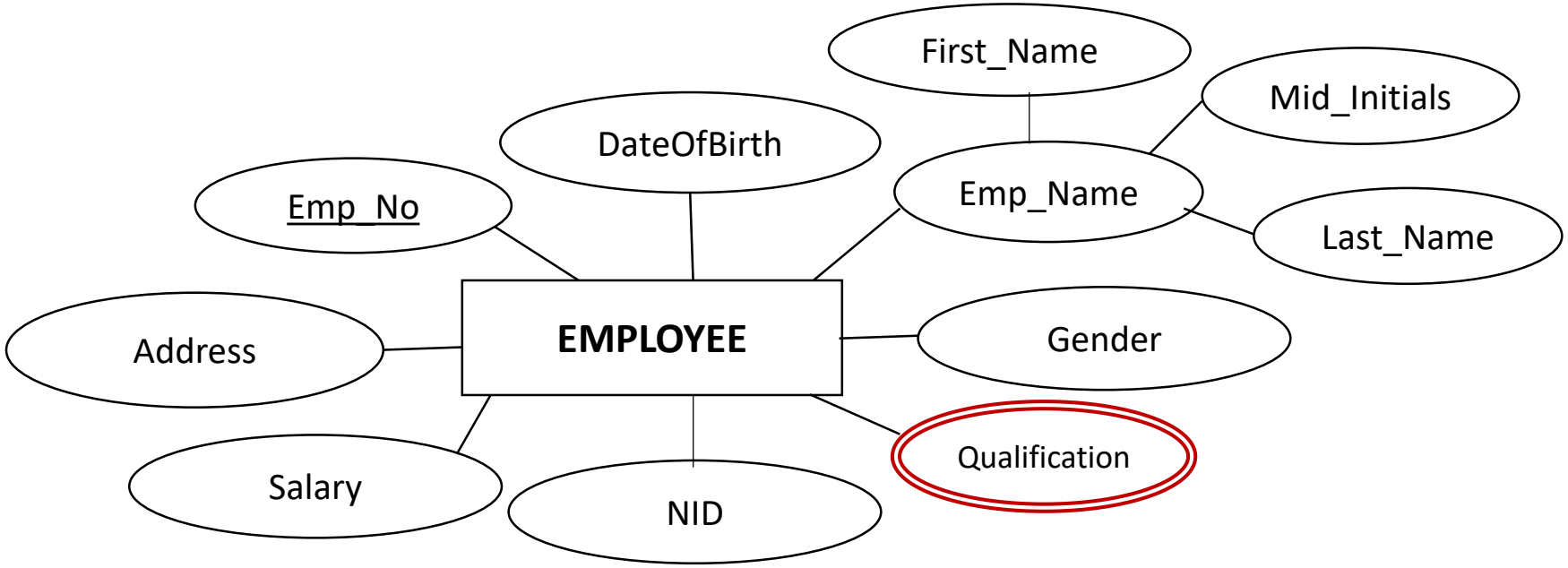
Map Strong Entity Types

Strong Entity Types

Create a relation that includes all simple attributes of that entity. For composite attributes, include only constituent simple attributes.

Entity	Relation
Entity name	Relation name
Simple attribute	Attribute of the relation
Entity identifier	Primary key of relation
Composite attribute	Component attributes

Map Strong Entity Types



Employee (<u>Emp_No</u>, NID, Address, Salary, Gender, DateOfBirth, First_Name, Mid_Initials, Last_Name)

Map Multi-valued Attributes

- Create new relation to represent multi-valued attribute and include primary key of (parent) entity in new relation, to act as a foreign key.

- Primary key of new relation is combination of the multi-valued attribute and the primary key of the (parent) entity.

  Emp_Qualification(Emp_No, Qualification)

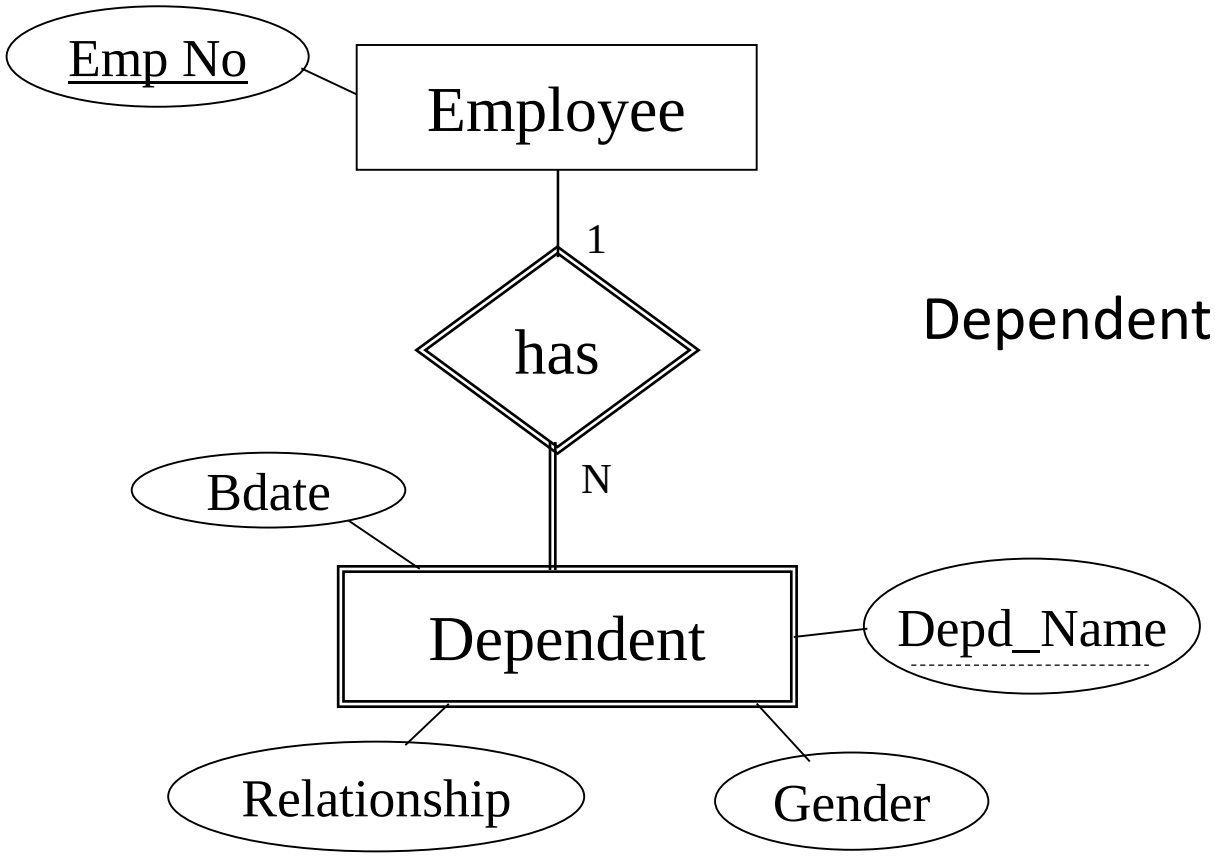
- The propagate (CASCADE) option for the referential triggered action should be specified on the foreign key in the new relation corresponding to the multivalued attribute for both ON UPDATE and ON DELETE.

Map Multi-valued Attributes

```
CREATE TABLE Emp_Qualification (
    Emp_No CHAR(05) NOT NULL,
    Qualification VARCHAR(25) NOT NULL,
 PRIMARY KEY (Emp_No , Qualification ),
 FOREIGN KEY (Emp_No) REFERENCES Employee(Emp_No)
                          ON DELETE CASCADE ON UPDATE CASCADE);
```

Map Weak Entity Types

- Create a relation that includes all simple attributes of that entity.
- Primary key is partially or fully derived from each owner entity.



Dependent(Emp_No, Depd_Name, Gender, Bdate, Relationship)

Map Weak Entity Types

- The propagate (CASCADE) option is enforced for the referential triggered action on the foreign key in the relation corresponding to the weak entity type, since a weak entity has an existence dependency on its owner entity. This can be used for both ON UPDATE and ON DELETE.

```
CREATE TABLE Dependent (
   Emp_No   CHAR(5)                NOT NULL,
   Depd_Name  VARCHAR(15)       NOT NULL,
   Gender CHAR,
   Bdate    DATE,
   Relationship  VARCHAR(8),
 PRIMARY KEY (Emp_No, Depd_Name ),
 FOREIGN KEY (Emp_No) REFERENCES EMPLOYEE(Emp_No)
                ON DELETE CASCADE    ON UPDATE CASCADE
 );
```

Map Binary Relationships

Rules depend on both the degree of the relationships and the cardinalities of the relationships.

- Map Binary One-to-Many (1:N) Relationships
- Map Binary Many-to-Many (M:N) Relationships
- Map Binary One-to-One (1:1) Relationships

1:N Binary Relationship Types

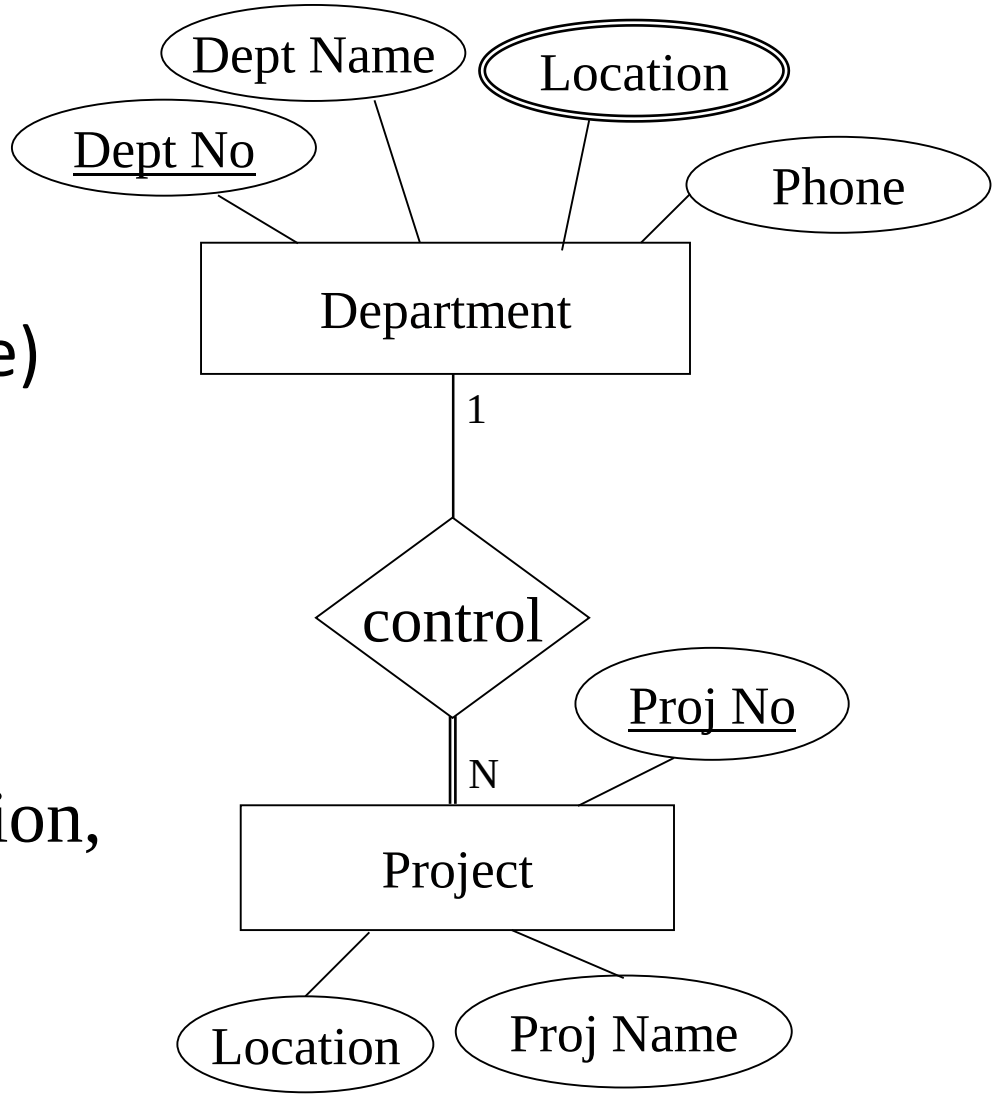
- Entity on ‘one side’ is designated the parent entity and entity on ‘many side’ is the child entity.
- Post copy of the primary key attribute(s) of parent entity into relation representing child entity, to act as a foreign key.
- Include any attributes of the relationship to the relation of the many side

1:N Binary Relationship Types

FK

Department(Dept_No, Dept_Name, Phone)

Project(Proj_No, Proj_Name, Location, ***Dept_No***)



1:N Binary Relationship Types

```
CREATE TABLE PROJECT (
        Pname   VARCHAR(15) NOT NULL,
        Proj_No  CHAR(05) NOT NULL,
        Plocation   VARCHAR(20),
        Dept_No  CHAR(03) NOT NULL,
        PRIMARY KEY (Proj_No),
        UNIQUE (Pname),
        FOREIGN KEY (Dept_No) REFERENCES Department(Dept_No)
                        ON DELETE SET DEFAULT ON UPDATE CASCADE );
```

1:1 Binary Relationship Types

- More complex as cardinality cannot be used to identify parent and child entities in a relationship.
- Instead, participation used to decide whether to combine entities into one relation or to create two relations and post copy of primary key from one relation to the other. Consider following:

i) total participation on both sides of 1:1 relationship;

ii) total participation on one side of 1:1 relationship;

iii) partial participation on both sides of 1:1 relationship.

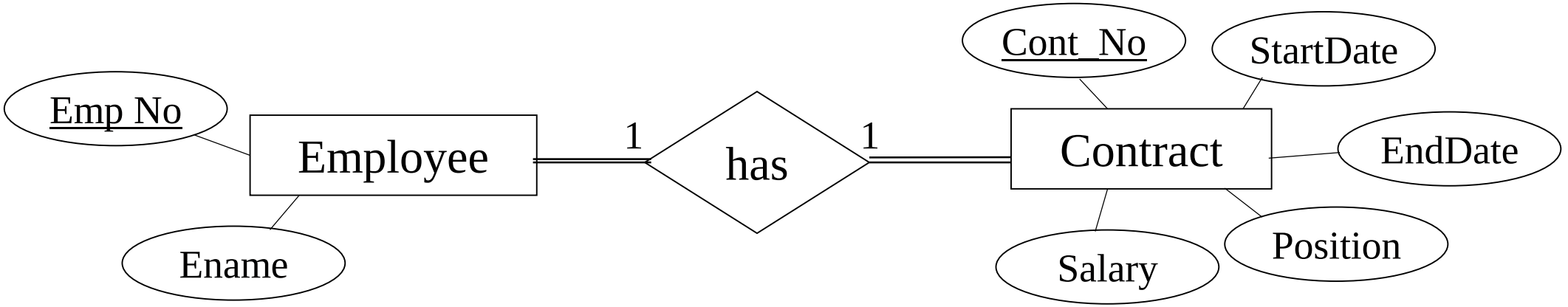
Total Participation On Both Sides Of 1:1 Relationship

Rule 1: It is often possible to subsume one entity type into the other.

Combine entities involved into one relation and choose one of the primary keys of original entities to be primary key of new relation, while other (if one exists) is used as an alternate key.

- The choice of which entity type subsumes the other depends on which is the most important entity type (more attributes, better key, semantic nature of them).
- The key of the subsumed entity type becomes a normal attribute.
- If there are any attributes in common, the duplicates are removed.
- The primary key of the new combined entity is usually the same as that of the original more important entity type.

Total Participation On Both Sides Of 1:1 Relationship



Rule 1:

The two entity types could be amalgamated into one.

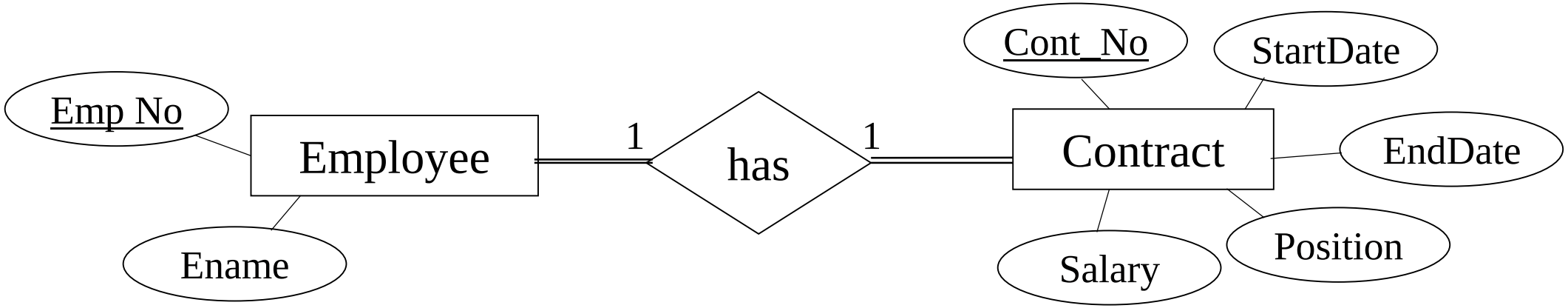
Employee(Emp_No, Ename, Cont_No, StartDate, EndDate, Position, Salary)

Total Participation On Both Sides Of 1:1 Relationship

Rule 2: Based upon how the entity types are participating in other relationships it is usually recommended to keep them as two separate relations.

- If two separate relations are maintained the primary key of one relation is included as the foreign key in the other.
- It does not matter which way around it is done but it should not have a foreign key in each entity.

Total Participation On Both Sides Of 1:1 Relationship



Rule 2:

Two entity types are kept apart, and a foreign key is used

Employee(Emp_No, Ename, Cont_No)

Contract(Cont_No, StartDate, EndDate, Position, Salary)

or

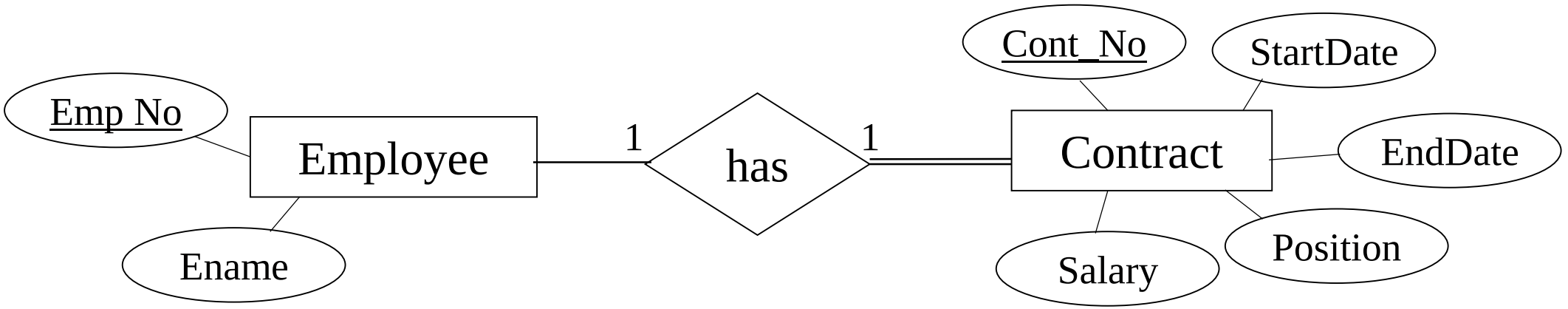
Employee(Emp_No, Ename)

Contract(Cont_No, StartDate, EndDate, Position, Salary, Emp_No)

Total Participation On One side Of a 1:1 Relationship

- Identify parent and child entities using participation constraints.
- Entity with partial participation is designated parent entity, and other entity designated child entity.
- A copy of the primary key of parent relation is placed in relation representing the child entity. This will avoid the insertion of null entries.
- If relationship has one or more attributes, these attributes should also be included in the child relation.

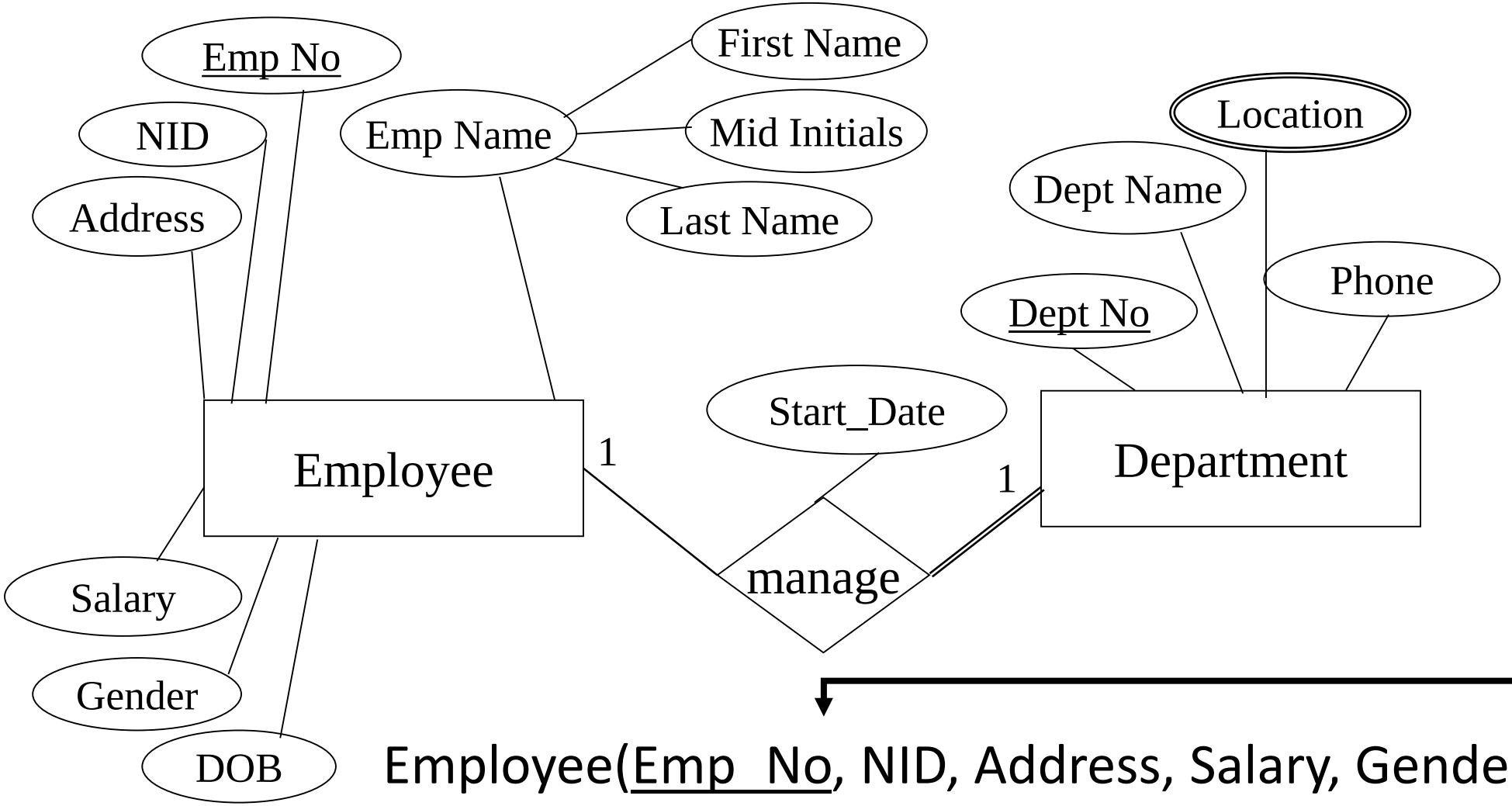
Total Participation On One side Of a 1:1 Relationship



Map the foreign key into Contract - *Emp_No* is mandatory thus never becomes null.

Employee(Emp_No, Ename)

Contract(Cont_No, StartDate, EndDate, Position, Salary, Emp_No)



FK

Employee(Emp_No, NID, Address, Salary, Gender, DOB, First_Name, Mid_Initials, Last_Name, Dept_No)

Department(Dept_No, Dept_Name, Phone, *Manager*, Start_Date)

Partial participation on both sides of a 1:1 relationship

- Designation of the parent and child entities is arbitrary unless we can find out more about the relationship.
- Consider employee Uses Car is a 1:1 relationship with partial participation on both sides. In this partial scenario it is assumed that majority of cars, but not all, are used by employees.
- Car entity, although partial, is closer to being total than Employee entity. Therefore designate Employee as parent entity and Car as child entity.
- Then the mapping is similar to total participation on one side of 1:1 relationship where a copy of the primary key of parent relation is placed in relation representing the child entity.

Map M:N Binary Relationship Types

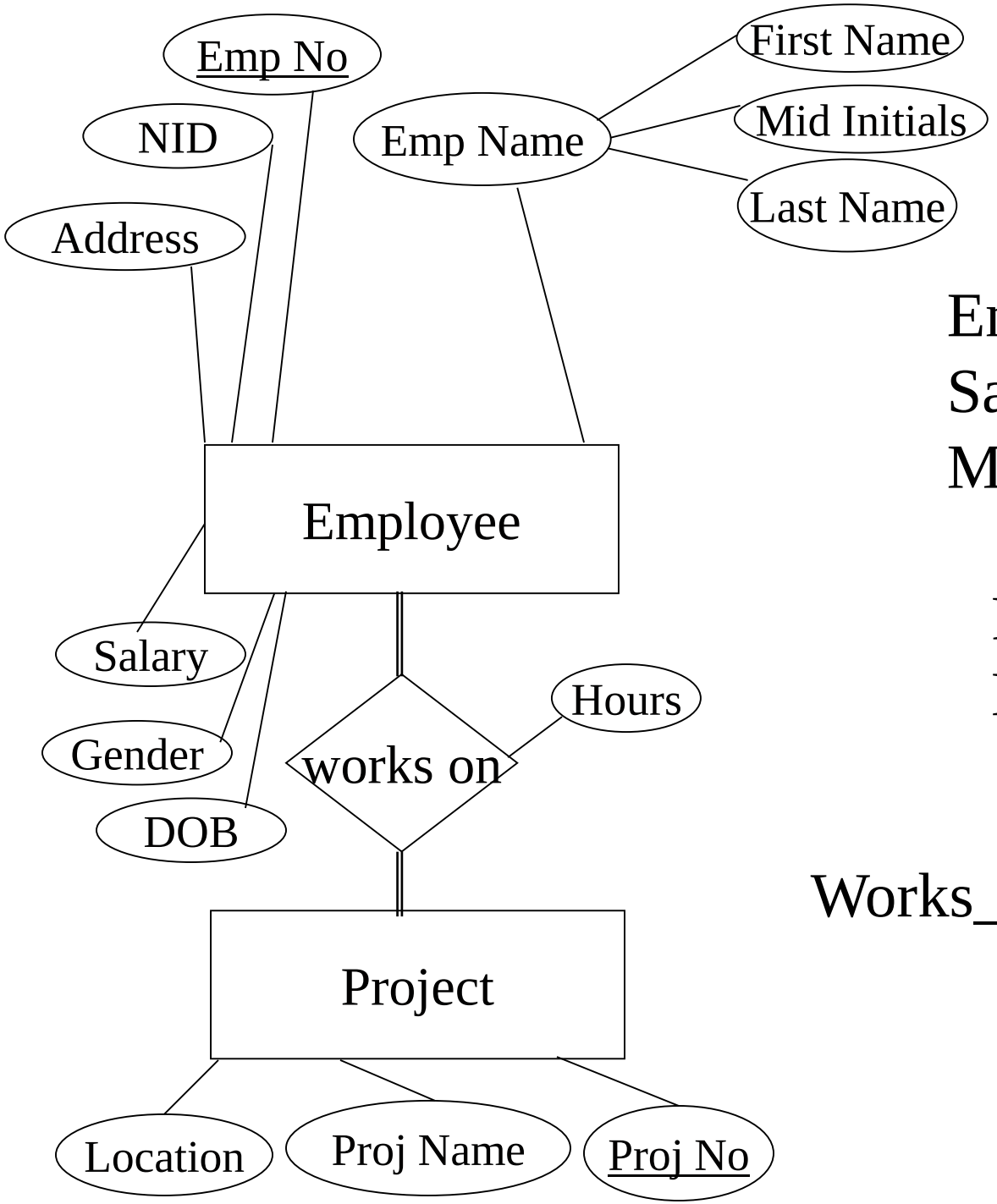
Create relation to represent relationship and include any attributes that are part of relationship.

- Post a copy of the primary key attribute(s) of the entities that participate in relationship into new relation, to act as foreign keys.
- These foreign keys will also form primary key of the new relation, possibly in combination with some of the attributes of the relationship.

Map M:N Binary Relationship Types

Create relation to represent relationship and include any attributes that are part of relationship.

- Post a copy of the primary key attribute(s) of the entities that participate in relationship into new relation, to act as foreign keys.
- These foreign keys will also form primary key of the new relation, possibly in combination with some of the attributes of the relationship.



FK/NN

Employee(Emp_No, NID, Address, Salary, Gender, DOB, First_Name, Mid_Initials, Last_Name, Dept_No)

Project(Proj_No, Proj_Name, Location, Dept_No)

FK/NN

Works_On(*Emp_No, Proj_No*, Hours)

Map M:N Binary Relationship Types

- The propagate (CASCADE) option for the referential triggered action should be specified on the foreign keys in the new relation corresponding to the relationship.
- Each relationship instance has an existence dependency on each of the entities it relates. This can be used for both ON UPDATE and ON DELETE.

```
CREATE TABLE Works_on(
   Emp_No CHAR(05) NOT NULL,
   Proj_No CHAR(03) NOT NULL,
   Hours DECIMAL(3,1) NOT NULL,
 PRIMARY KEY (Emp_No, Proj_No),
 FOREIGN KEY (Emp_No) REFERENCES Employee(Emp_No)
                          ON DELETE CASCADE ON UPDATE CASCADE ,
 FOREIGN KEY (Proj_No) REFERENCES Project(Proj_No)
                          ON DELETE CASCADE ON UPDATE CASCADE);
```