IN2090
Databases and data modelling

Today's topics:

- Data Modeling Using the Entity–Relationship (ER) Model
  - Ternary relationships
  - ER-to-Relational Mapping

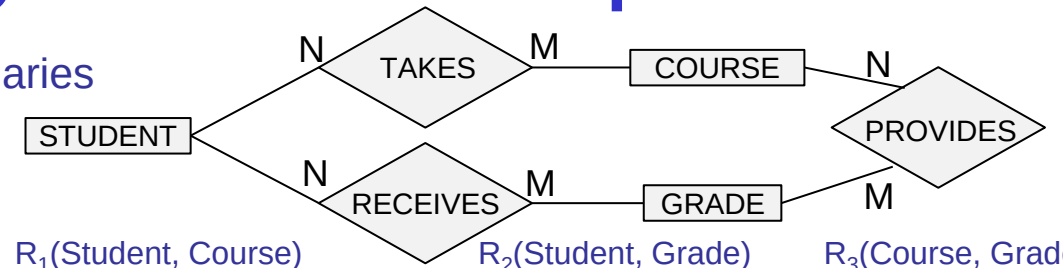
Equivalence of ternary and binary relationships

Ternary

R(Student, Course, Grade)

Student	Course	Grade
1	C1	A
2	C1	A
1	C2	B
~~1~~	~~C1~~	~~B~~
2	C2	A
~~1~~	~~C2~~	~~A~~

Binaries



R_1(Student, Course)

Student	Course
1	C1
2	C1
1	C2
2	C2

⋈

R_2(Student, Grade)

Student	Grade
1	A
2	A
1	B
~~2~~	~~A~~

⋈

R_3(Course, Grade)

Course	Grade
C1	A
~~C1~~	~~A~~
C2	B
C2	A

A ternary relation R(X, Y, Z) can be decomposed into:

- Two binary relations $R_1(X, Y)$ and $R_2(X, Z)$ if $R(X, Y, Z) = R_1(X, Y) \bowtie R_2(X, Z)$
- Three binary relations $R_1(X, Y)$, $R_2(X, Z)$, and $R_3(Y, Z)$ if $R(X, Y, Z) = R_1(X, Y) \bowtie R_2(X, Z) \bowtie R_3(Y, Z)$

In this case, the ternary relationship CANNOT have an equivalent, binary decomposition structure!

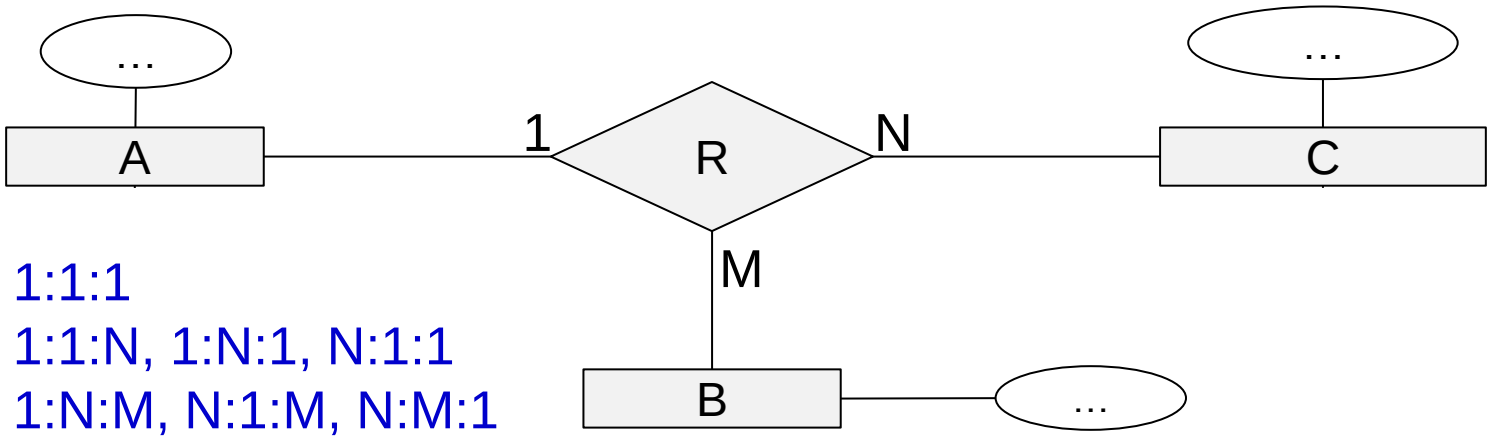
Student	Course	Grade
1	C1	A
1	C1	B
2	C1	A
1	C2	A
1	C2	B
2	C2	A

Student	Course	Grade
1	C1	A
2	C1	A
1	C2	A
1	C2	B
2	C2	A

Not allowed in the original ternary relation!

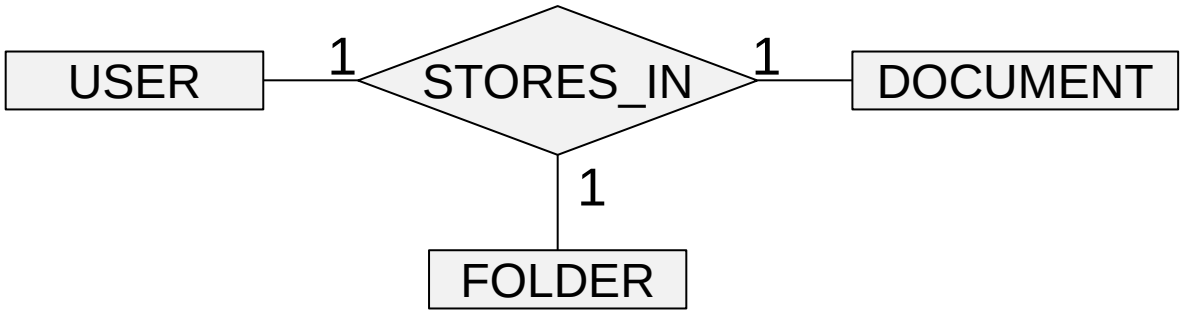
Ternary relationships

- Association among three entities
  - Typically required when binary relationships are not sufficient to accurately describe the semantics of the association
- In ER diagrams the three entities are attached to a single relationship diamond, and the connectivity of each entity is designated as either “one” (1) or “many” (N, M,…), e.g.,



1:1:1
1:1:N, 1:N:1, N:1:1
1:N:M, N:1:M, N:M:1
N:M:P

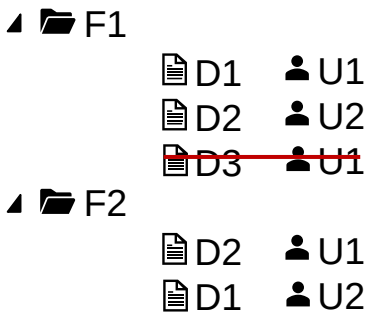
Forms of ternary relationships (1:1:1)



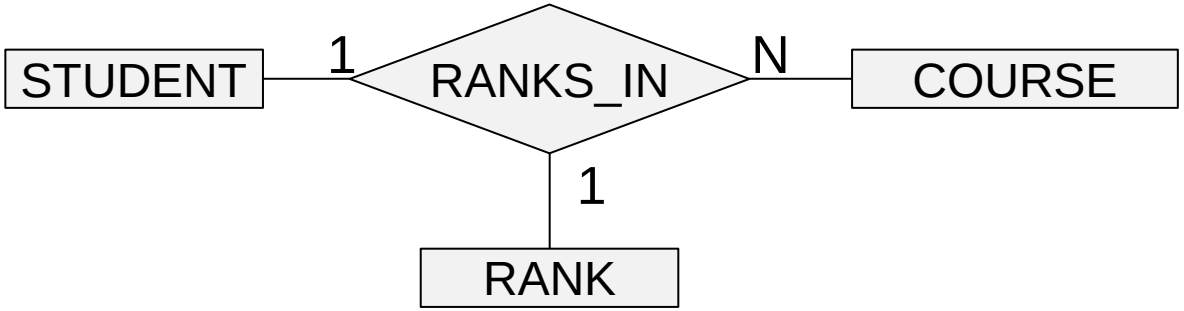
A user uses exactly one folder to store each document. A user may store many documents and maintain different folders for different documents. A folder can be used to store different documents given that each document in that folder belongs to different users. A document can be stored in different folders given that each such folder is used by different users.

User	Folder	Document
U1	F1	D1
U2	F1	D2
U1	F2	D2
~~U1~~	~~F1~~	~~D3~~
U2	F2	D1

- Each combination of user-document determines one folder
- Each combination of user-folder determines one document
- Each combination of folder-document determines one user



Forms of ternary relationships (1:1:N, 1:N:1, N:1:1)

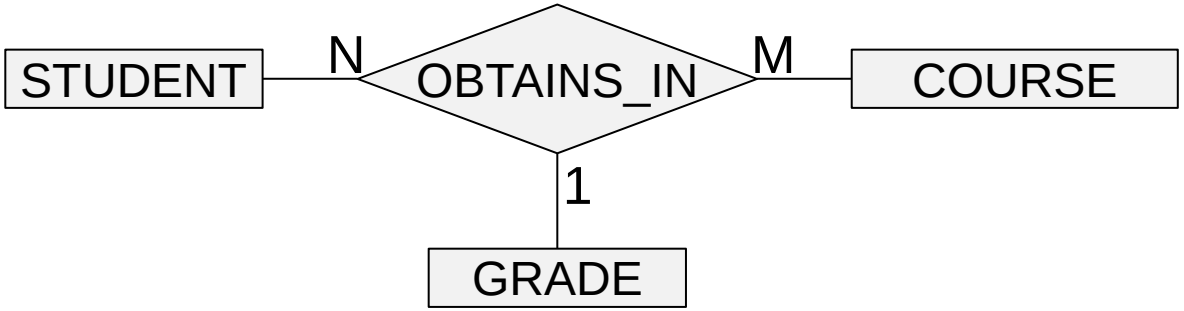


Each student in a course can only rank at one rank level but can rank at different rank levels for different courses. Each rank level in a course can be assigned to only one student and different students can be assigned to different ranks in that course.

- Each combination of course-rank determines one student
- Each combination of course-student determines one rank

Student	Course	Rank
1	C1	1
2	C1	2
1	C2	1
3	C1	3
2	C2	1

Forms of ternary relationships (1:N:M, N:1:M, N:M:1)

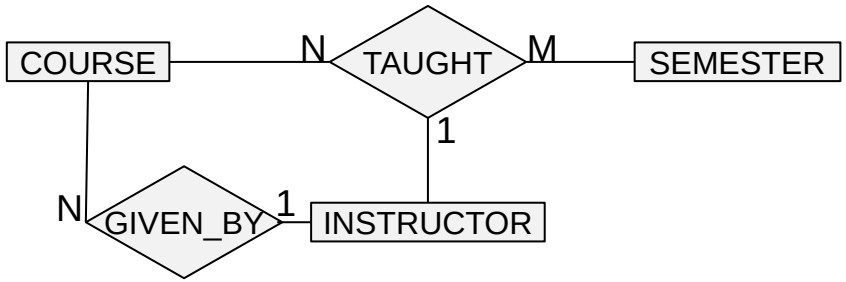


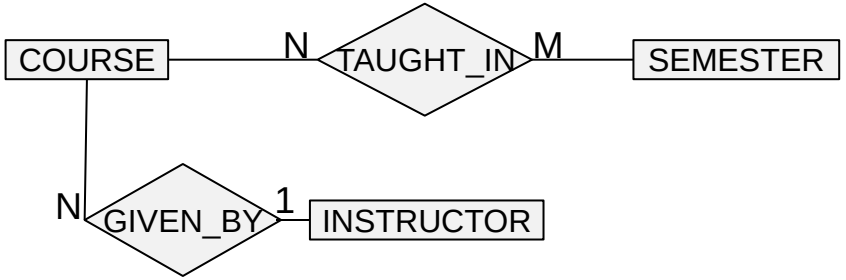
Each student taking a particular course receives exactly one grade for that course. The same grade in a course can be taken by many students in that course, and the same grade for a student can be taken in many courses by that student.

- Each combination of course-student determines one grade

Student	Course	Grade
1	C1	A
2	C1	A
1	C2	B
2	C2	A
~~1~~	~~C1~~	~~B~~

Example of situation when N:M:1 ternary relation is equivalent to binary relations





Course	Semester	Instructor
C1	S1	I1
C1	S2	I1
C2	S1	I1
C3	S1	I3
~~C1~~	~~S1~~	~~I2~~
~~C2~~	~~S3~~	~~I2~~

Course	Semester
C1	S1
C1	S2
C2	S1
C3	S1

⋈

Course	Instructor
C1	I1
~~C1~~	~~I1~~
C2	I1
C3	I3

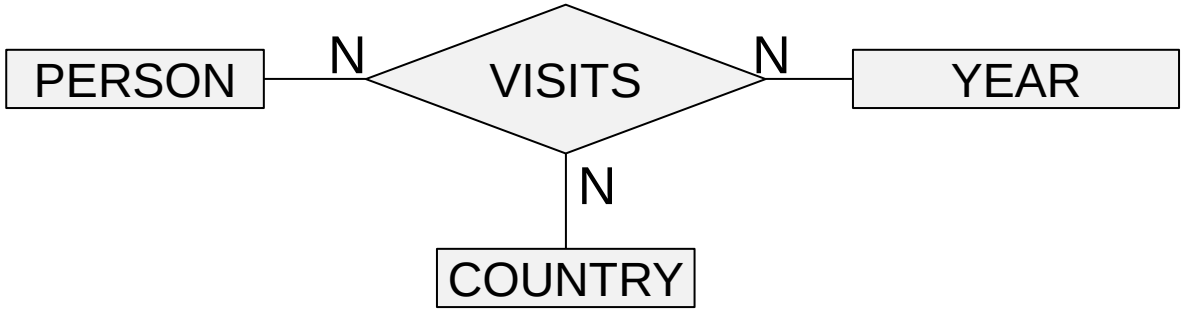
Same!

Course	Semester	Instructor
C1	S1	I1
C1	S2	I1
C2	S1	I1
C3	S1	I3

In this case (X:Y:Z) = (N:M:1) and (X:Z) = (N:1), the ternary relationship CAN have an equivalent, binary decomposition structure (XY)(XZ) !

In general two or three binary relationships cannot replace a ternary relationship, but they may do so under certain *additional* constraints!

Forms of ternary relationships (N:M:P)

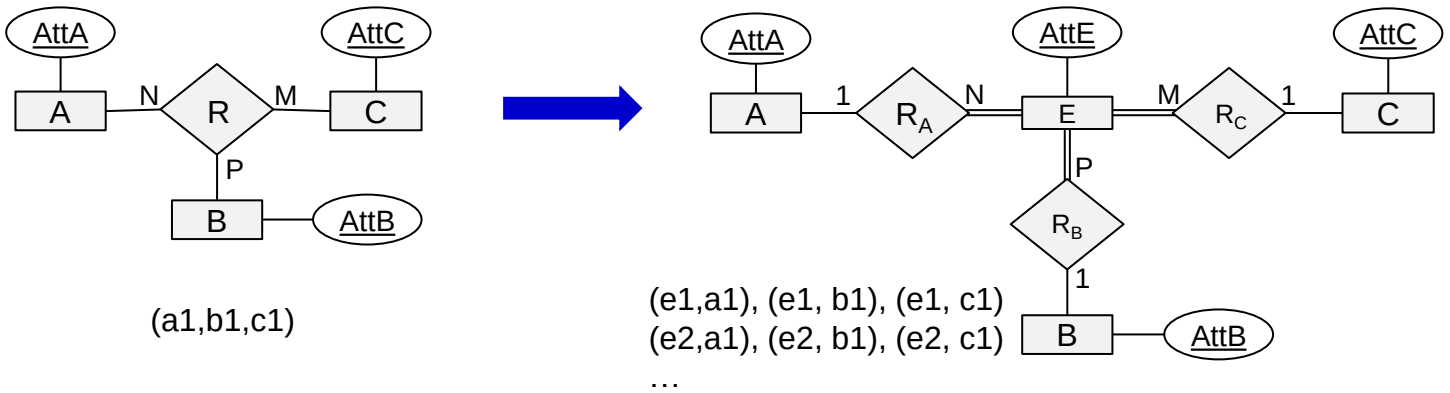


Many persons may visit many countries in many years.

Person	Country	Year
1	DE	1980
1	NO	2019
1	DE	2000
2	DE	1980
2	DE	1981
1	NO	20218

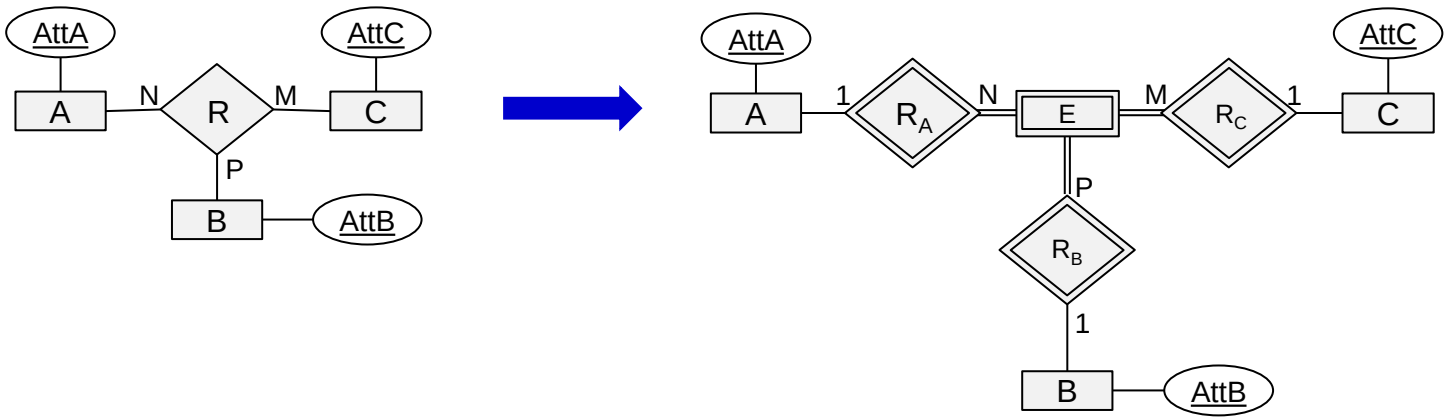
Representing ternary relationships using binary relationships

- Some database design tools permit only binary relationships
- Ternary relationships can be represented using binary relationships by creating an *artificial* entity type
  - Replace R by an artificial entity type E, and three relationship types: R_A(relating E and A), R_B(relating E and B), and R_C(relating E and C)
  - Create an identifying attribute for E (and add any attributes of R to E)
  - Add constraints (Note: translating all ternary constraints may not be possible)

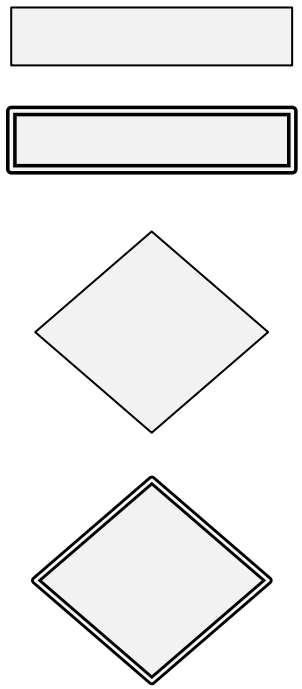


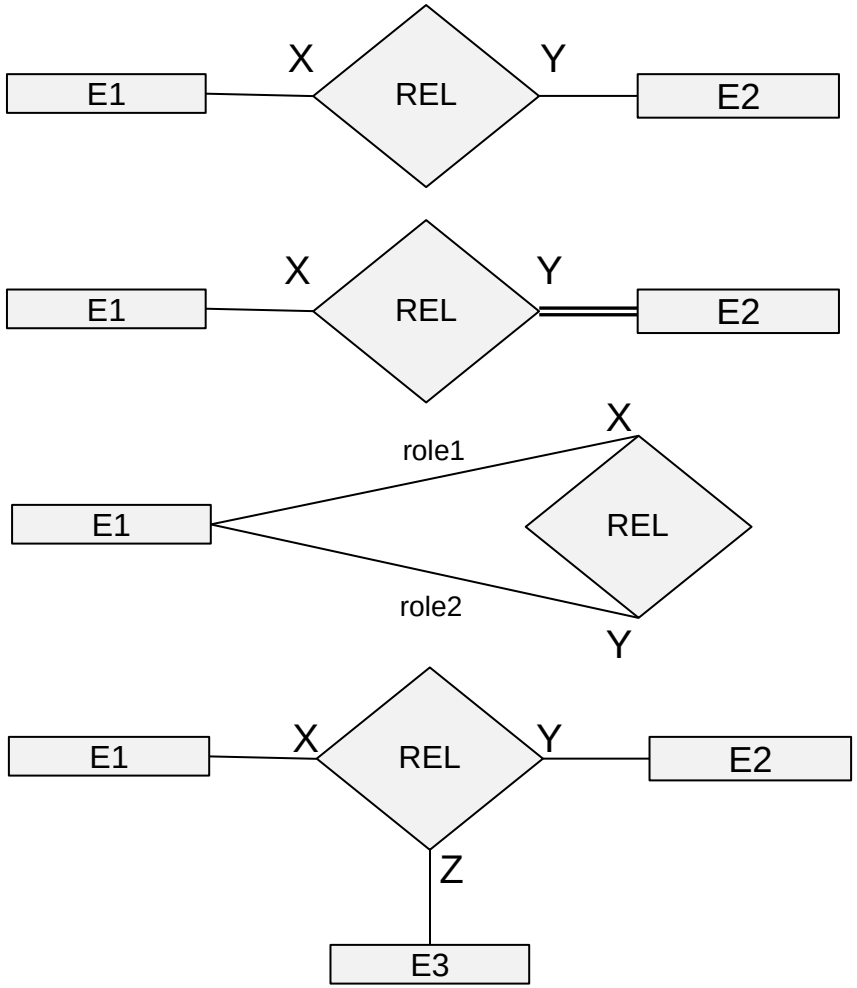
Representing ternary relationships using binary relationships (cont')

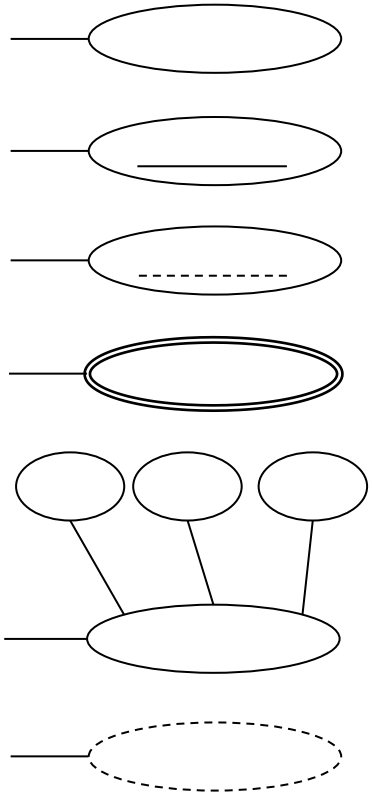
- Alternative representation using artificial *weak* entity type
  - Replace R by an artificial weak entity type E with no partial key and three identifying relationship types: R_A(relating E and A), R_B(relating E and B), and R_C(relating E and C)
  - An entity in the weak entity type E is identified by the combination of its three identifying entities from A, B, and C
  - Add constraints (Again, note that translating all ternary constraints may not be possible)



Summary ER modelling notation







Typical steps in ER modeling

ITERATE !!!

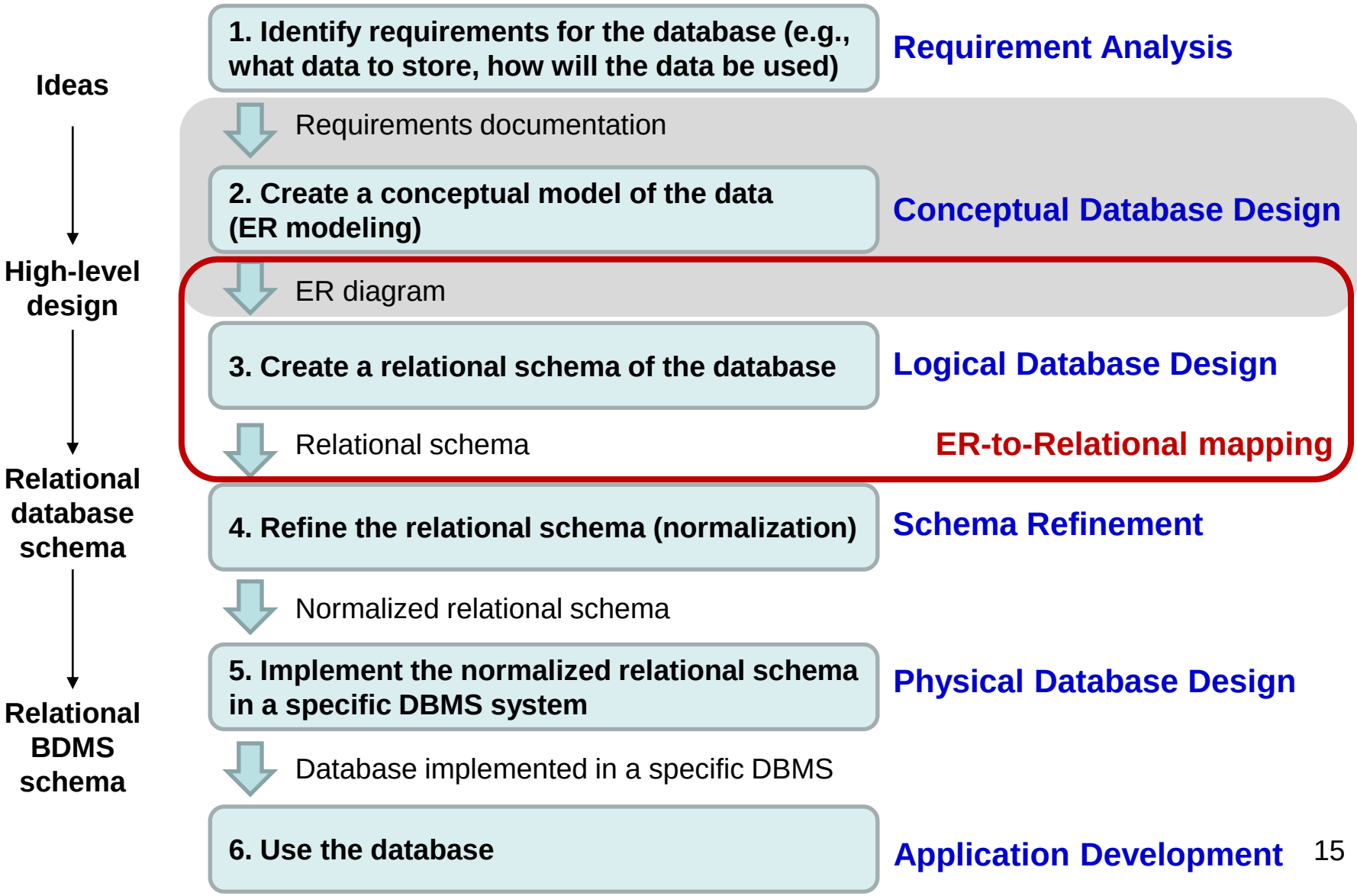
1. Identify entity types
   - Assign a singular noun to each entity type
2. Identify relationships between (among) entities
   - Identify binary relationships first
   - Use a meaningful verb for a relationship name
3. Identify relationship constraints
   - Cardinality constraints (1:1, 1:N, N:M, 1:1:1, 1:1:N, 1:M:M, N:M:P, …)
   - Participation constraints (Total, Partial)
4. Assign attributes to entity types and relationship types
   - Usually attributes come from nouns, adjectives or adverbs
5. Select identifiers (keys) for entity types
   - Weak entity: composite key
   - Regular entity: attribute key

ER modeling aspects

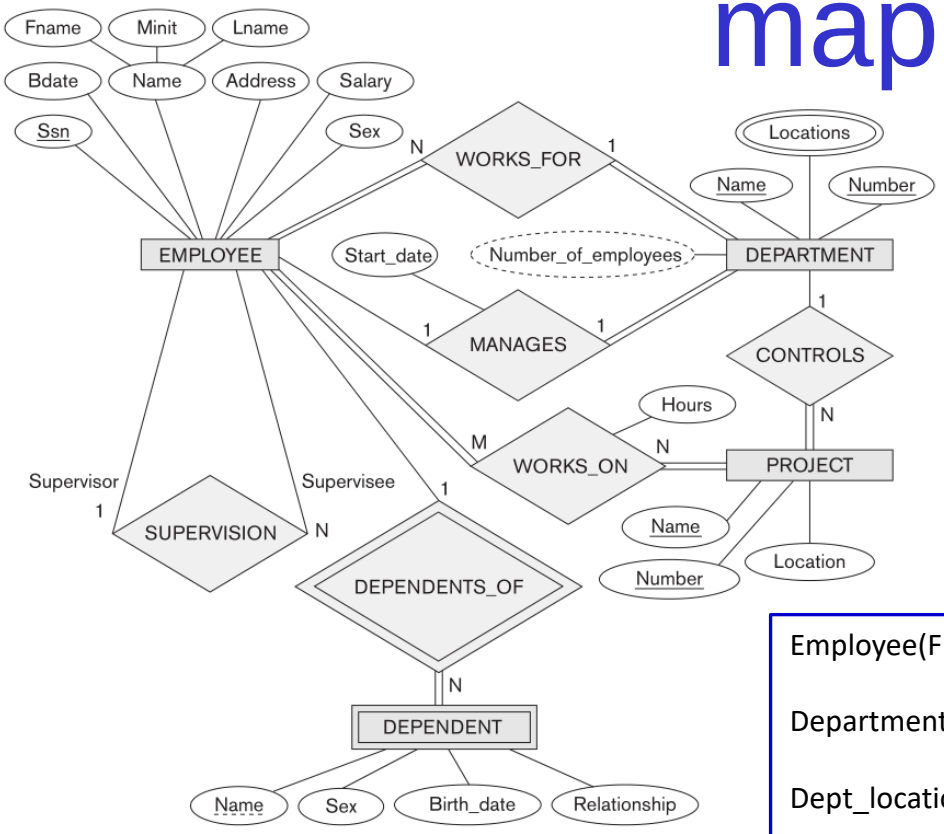
- Clearly state the database requirements before doing any conceptual modeling in ER
- Interact often with the end user of the database, to make sure all assumptions you make are also true for the user's view of the database
- Key principles: keep the conceptual model simple and faithful to the requirements specification, avoid redundancy, specify constraints
- Design choices: entity type vs. attribute, entity type vs. relationship type, ternary vs. multiple binary relationships
- ER design is subjective (there is no single best way to get a model right, though many ways to get it wrong!)

ER-to-Relational mapping

Building a database



Example ER-to-Relational mapping



Going from an ER model to a relational model (understandable by a relational DBMS)

Employee(Fname, Minit, Lname, Ssn, Bdate, Address, Sex, Salary, Super_ssn, Dno)

Department(Dname, Dnumber, Mgr_ssn, Mgr_start_date)

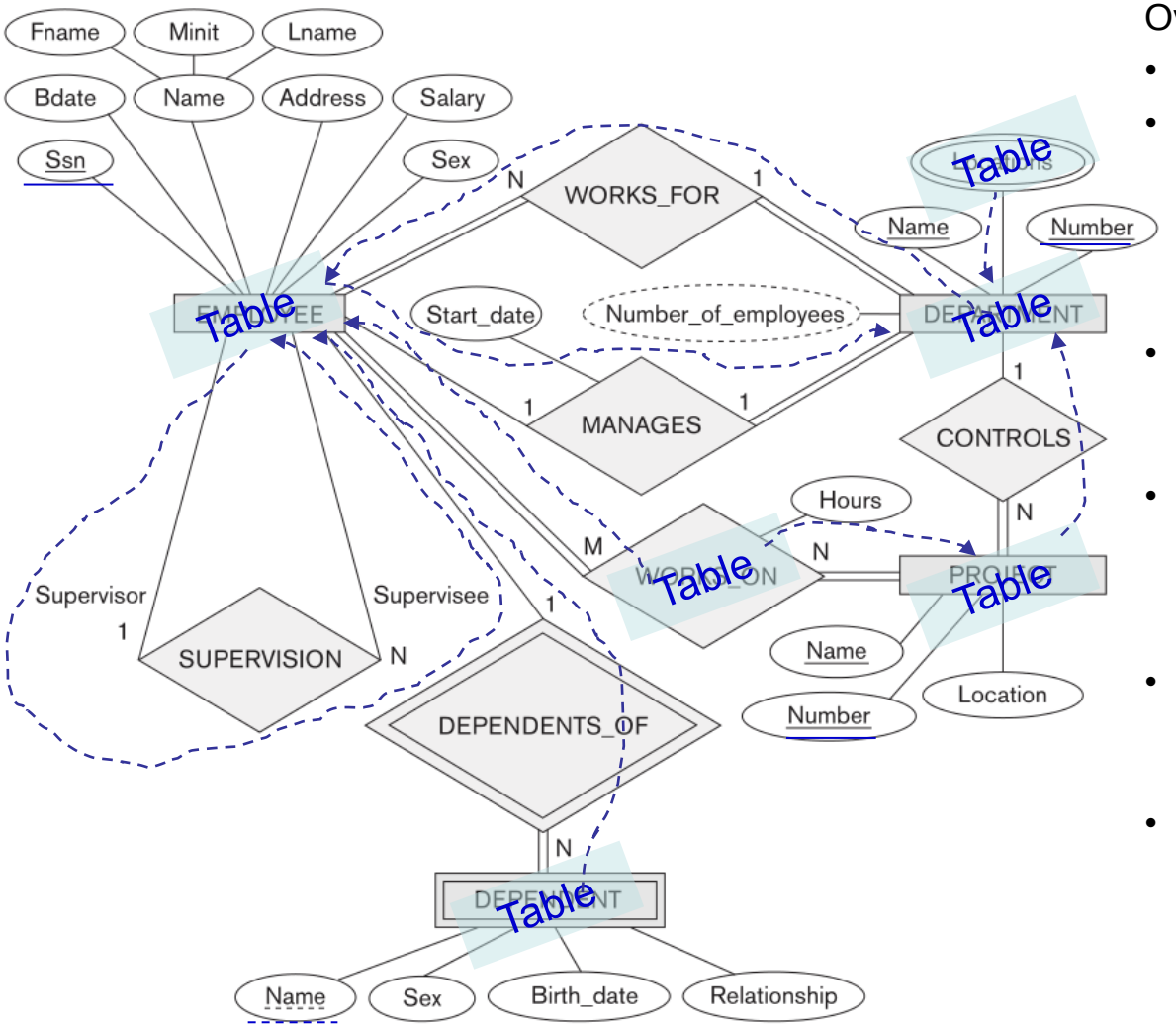
Dept_locations(Dnumber, Dlocation)

Project(Pname, Pnumber, Plocation, Dnum)

Works_on(Essn, Pno, Hours)

Dependent(Essn, Dependent_name, Sex, Bdate, Relationship)

ER-to-Relational mapping overview



Overview:

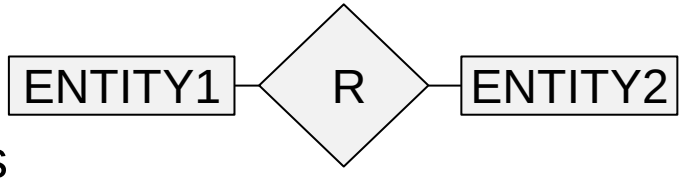
- Entities become relations (i.e., tables)
- Simple attributes and elements of composite attributes become attributes in relations (i.e., column headers in tables); key attributes become candidate keys
- Complex attributes become tables with foreign keys back to the parent entity table (its primary key)
- one-to-one or one-to-many relationships imply connections by primary key/foreign key pairs between tables
- many-to-many relationships become an “interconnect” table that simulates two equivalent one-to-many relationships
- Ternary relationships become an “interconnect” table with primary key/foreign key pairs to simulate actual relationships among attributes

ER-to-Relational mapping: A seven-step mapping procedure

Step 1: Mapping of Regular Entity Types

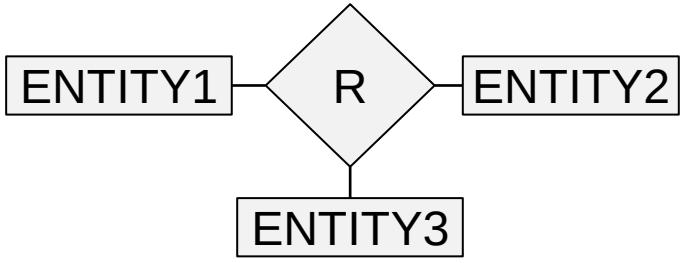
Step 2: Mapping of Weak Entity Types

Step 3: Mapping of Binary 1:1 Relation Types
Step 4: Mapping of Binary 1:N Relationship Types
Step 5: Mapping of Binary M:N Relationship Types



Step 6: Mapping of Multivalued attributes

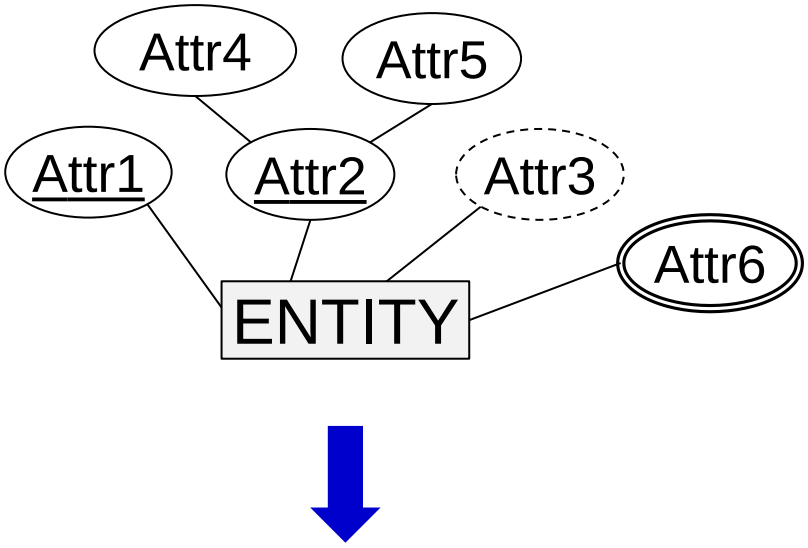
Step 7: Mapping of Ternary Relationship Types



Step 1: Mapping of Regular Entity Types

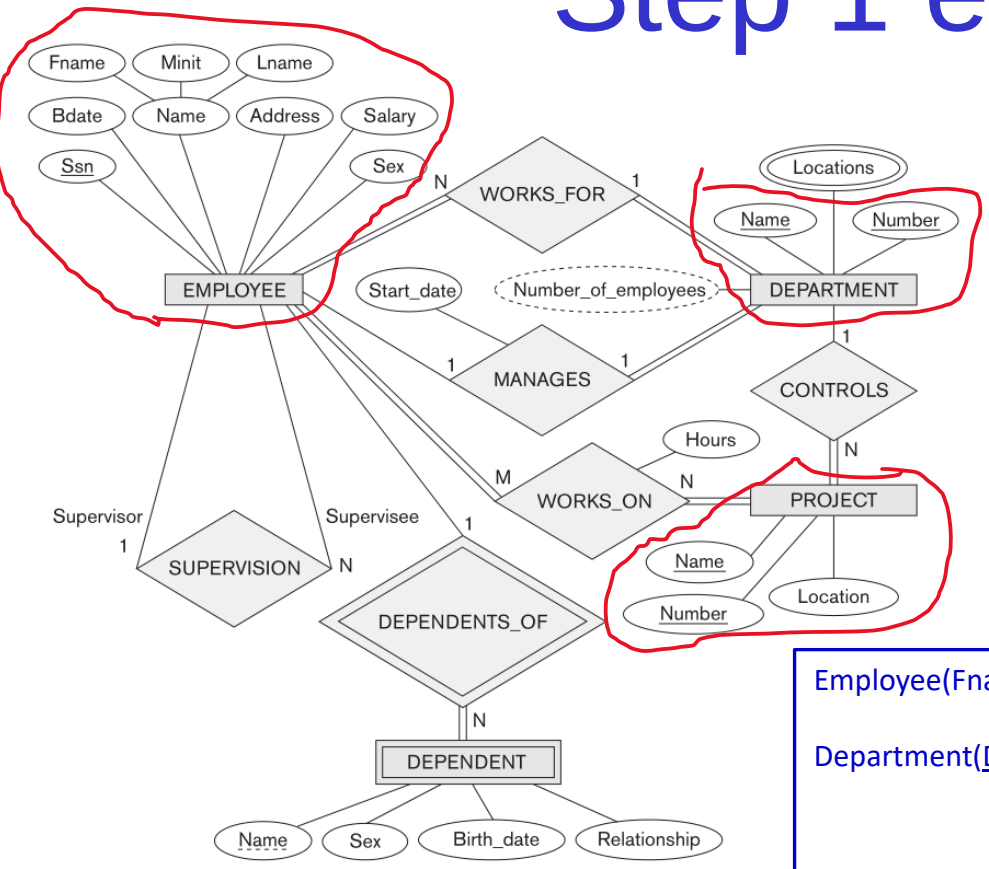
For each regular entity type:

1. Create a relation
2. For every simple attribute of the entity type, create an attribute of the relation
   - For composite attributes only their constituent atomic components are recorded
   - Derived attributes are not recorded
   - Multivalued attributes are treated later (Step 6)
3. For every key of the entity type create candidate keys in the relation, and choose a primary key



Rel(Attr1, Attr4, Attr5)

Step 1 example



Employee(Fname, Minit, Lname, Ssn, Bdate, Address, Sex, Salary)

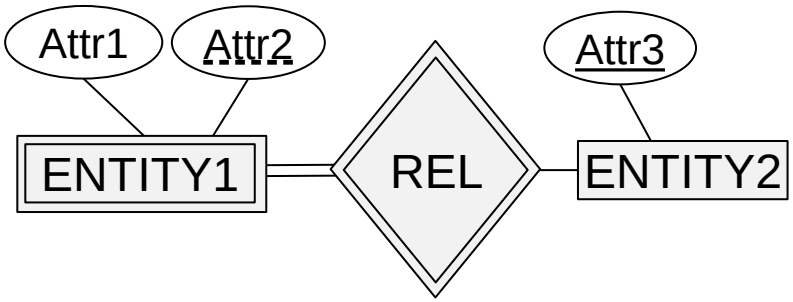
Department(Dname, Dnumber)

Project(Pname, Pnumber, Plocation)

Step 2: Mapping of Weak Entity Types

For each weak entity type:

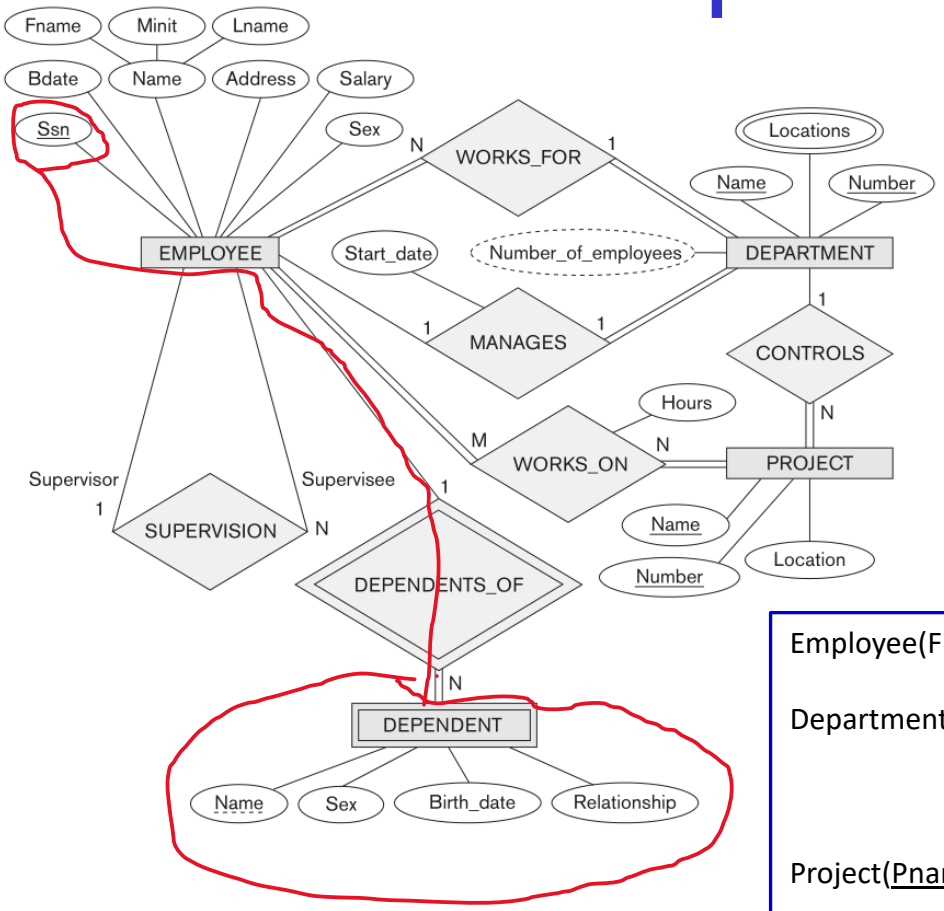
1. Create a relation
2. For every simple attribute of the entity type, create an attribute of the relation
   - For composite attributes only their constituent atomic components are recorded
3. Add the key(s) of the identifying (parent) entity type as attribute(s) in the relation
   - Include the added attribute(s) as foreign key(s) corresponding to the key attribute(s) of the parent entity type
4. The attribute(s) thus added plus the partial key(s) of the weak entity type form the primary key of the relation representing the weak entity type



RelE1(Attr1, Attr2, Attr3)

RelE2(Attr3)

Step 2 example



Employee(Fname, Minit, Lname, Ssn, Bdate, Address, Sex, Salary)

Department(Dname, Dnumber)

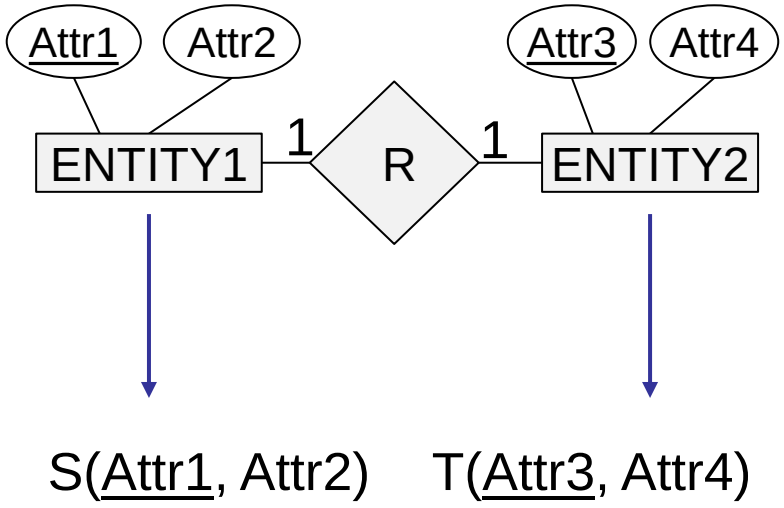
Project(Pname, Pnumber, Plocation)

Dependent(Essn, Dependent_name, Sex, Bdate, Relationship)

Step 3: Mapping of Binary 1:1 Relation Types

For each binary 1:1 relationship type R:

1. Identify the relations S and T that correspond to the entity types participating in R
2. Choose one of the three possible approaches:
   1. Option 1: Foreign key approach
   2. Option 2: Merged relationship approach
   3. Option 3: Relationship relation approach

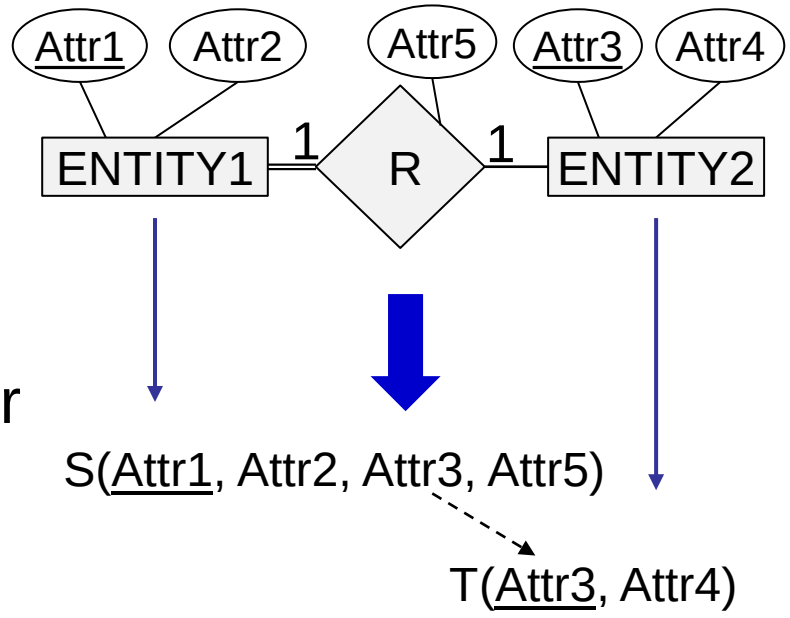


Note: Option 1 should be followed unless special conditions exist (Options 2 and 3)

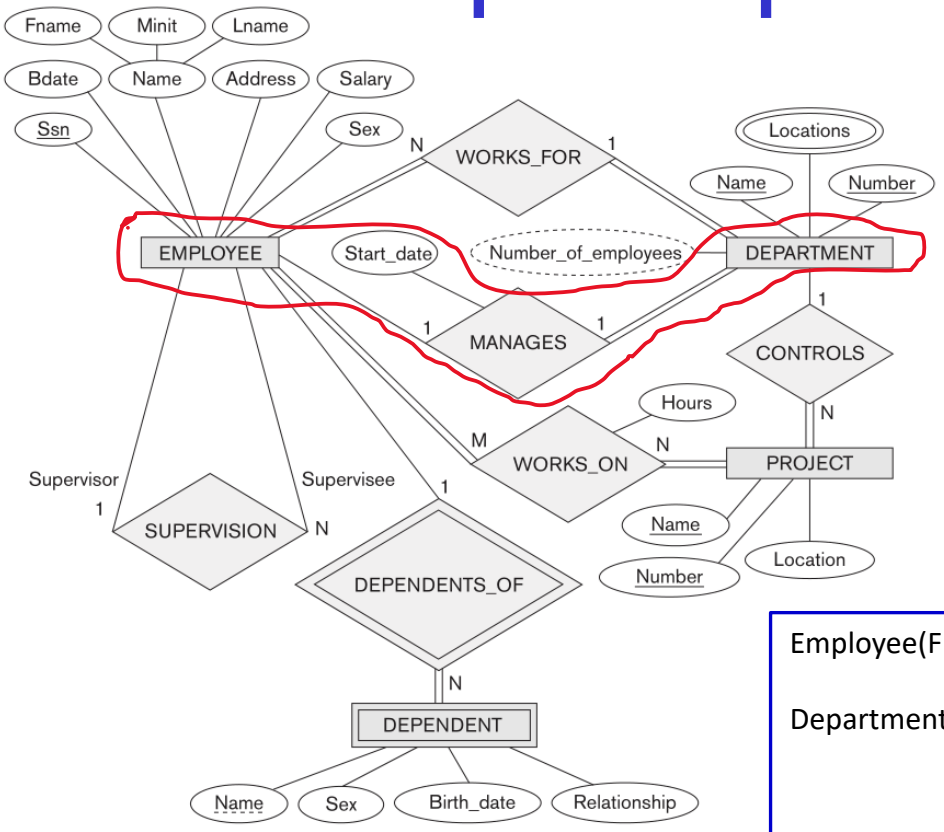
Step 3 Option 1 (Foreign key approach)

a. Choose one of the relations (e.g., S) and include as a foreign key in S the primary key of T
   - If one of the entity types has total participation in R, choose that one in the role of S

b. Include all the simple attributes (or simple components of composite attributes) of the 1:1 relationship type R as attributes of S



Step 3 Option 1 example



Employee(Fname, Minit, Lname, Ssn, Bdate, Address, Sex, Salary)

Department(Dname, Dnumber, Mgr_ssn, Mgr_start_date)

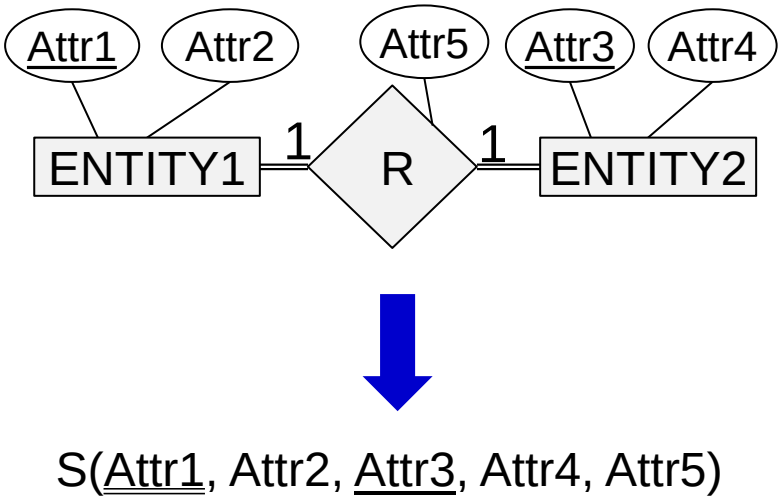
Project(Pname, Pnumber, Plocation)

Dependent(Essn, Dependent_name, Sex, Bdate, Relationship)

Step 3 Option 2 (Merged relationship approach)

a. When both entity types have total participation in R, merge the two entity types and the relationship into a single relation
   - The two relations corresponding to the two entity types will have the exact same number of tuples at all times, thus they can be merged into one

b. Choose one of the keys of the two entity types as a primary key in the resulting relation



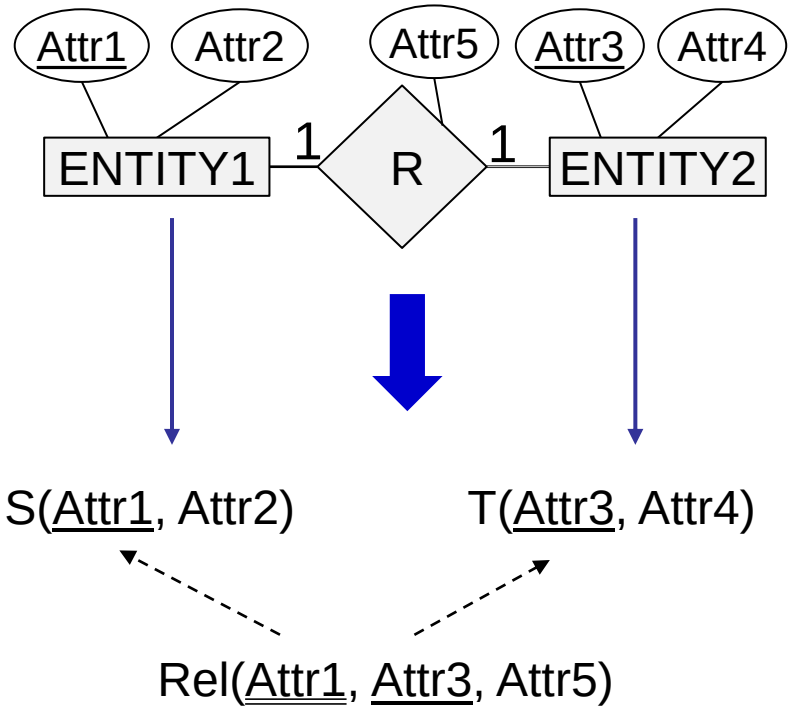
S(Attr1, Attr2, Attr3, Attr4, Attr5)

Step 3 Option 3 (Relationship relation approach)

a. Set up a third relation Rel (a.k.a. the "relationship relation") for the purpose of cross-referencing the primary keys of the two relations S and T representing the entity types

b. The relation Rel include in the primary key of relation Rel the attributes of S and T as foreign keys to S and T; and add to the attributes of Rel any other attributes of the relationship type R

c. The primary key of Rel will be one of the two foreign keys, and the other foreign key will be a candidate key of Rel



Step 4: Mapping of Binary 1:N Relationship Types

Two possible approaches:

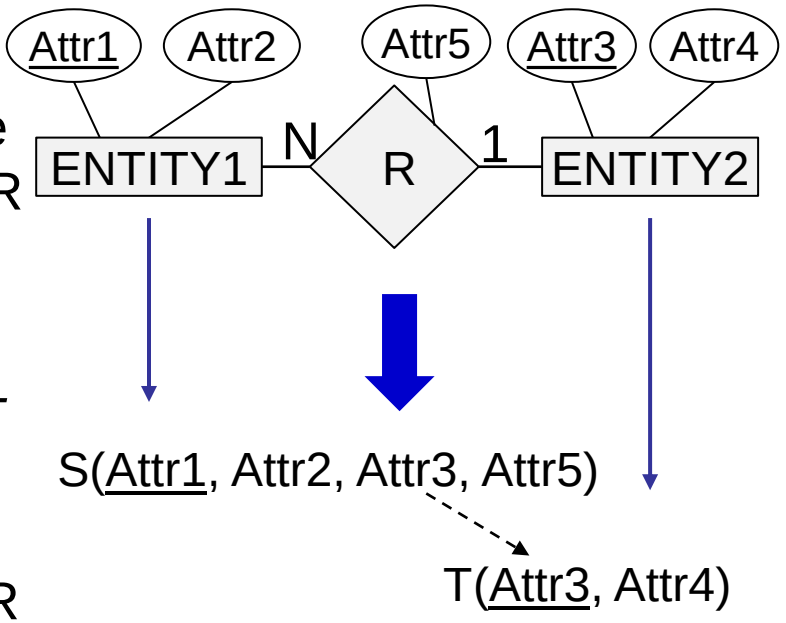
1. Option 1: Foreign key approach
2. Option 2: Relationship relation approach

Note: Option 1 is generally preferred as it reduces the number of resulting relations

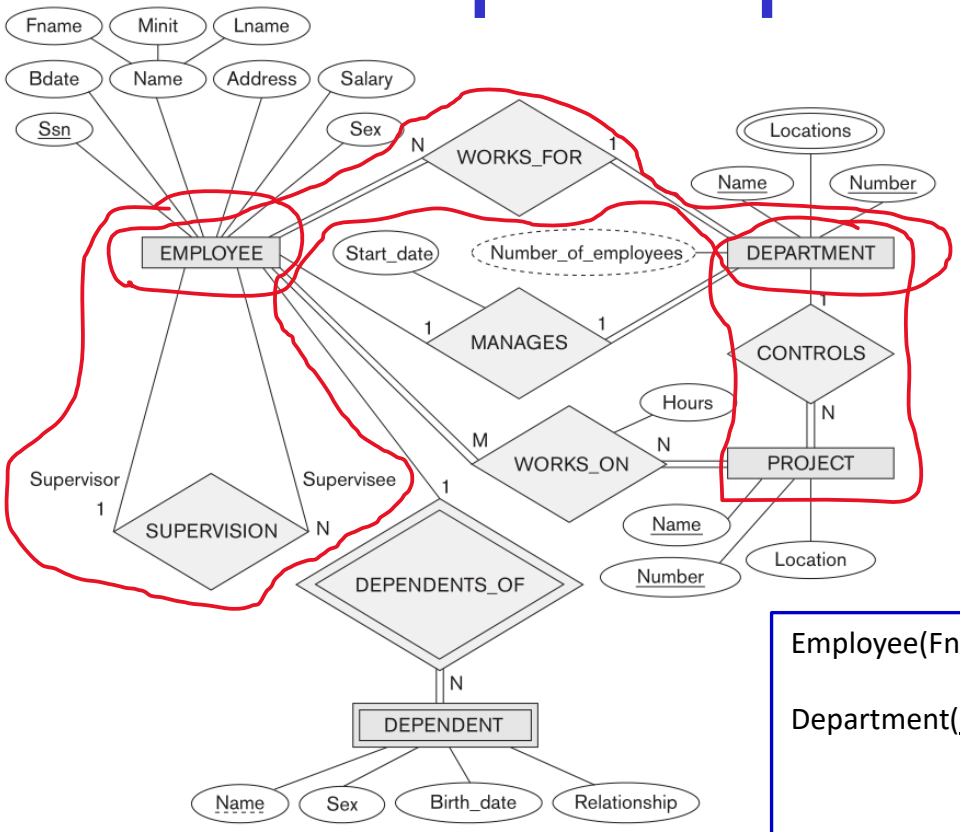
Step 4 Option 1 (Foreign key approach)

For each regular binary 1:N (or N:1) relationship type R:

a. Identify the relation S that represents the participating entity type at the *N-side* of R
b. Include as foreign key in S the primary key of the relation T that represents the other entity type participating in R (the *1-side* of R)
c. Include any simple attributes (or simple components of composite attributes) of R as attributes of S



Step 4 Option 1 example



Employee(Fname, Minit, Lname, Ssn, Bdate, Address, Sex, Salary, Super_ssn, Dno)

Department(Dname, Dnumber, Mgr_ssn, Mgr_start_date)

Project(Pname, Pnumber, Plocation, Dnum)

Dependent(Essn, Dependent_name, Sex, Bdate, Relationship)

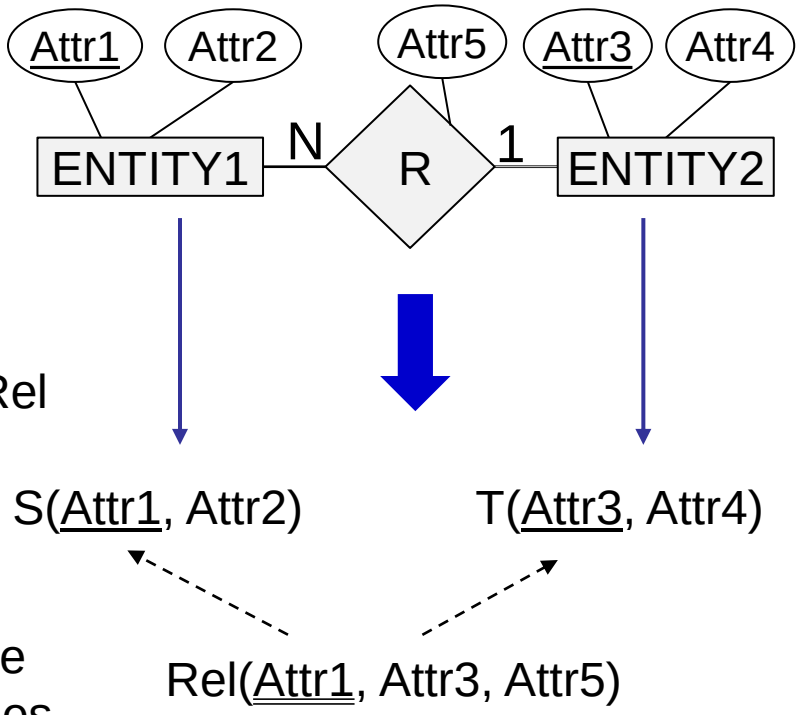
Step 4 Option 2 (Relationship relation approach)

For each regular binary 1:N (or N:1) relationship type R:

a. Identify the relation S that represents the participating entity type at the *N-side* of the relationship type

b. Create a separate relation Rel whose attributes are the primary keys of S and T, which will also be foreign keys to S and T; include any attributes of R as attributes of Rel

c. The primary key of Rel is the same as the primary key of S

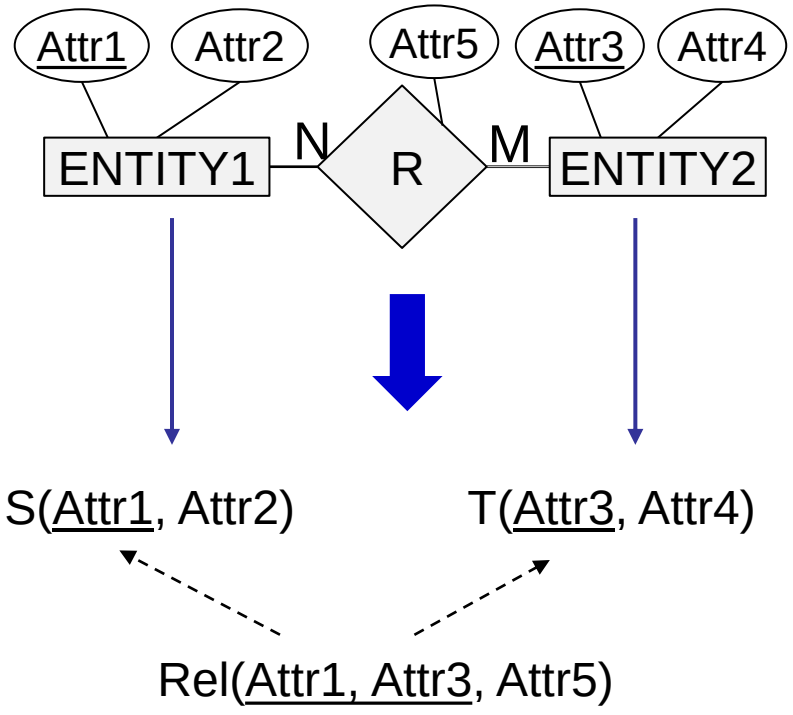


Note: Use this option if few tuples in S participate in the relationship to avoid excessive NULL values in the foreign key

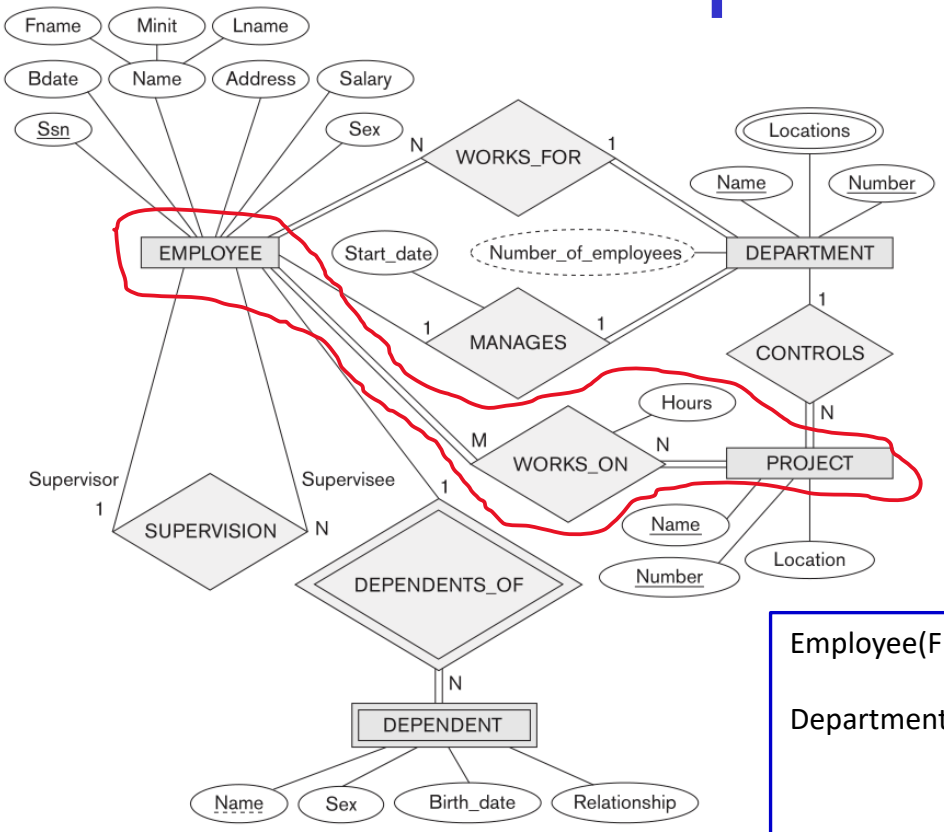
Step 5: Mapping of Binary M:N Relationship Types

For each binary M:N relationship type R:

1. Create a new relation Rel to represent R
2. Include as foreign key attributes in Rel the primary keys of the relations that represent the participating entity types (S and T); their combination will form the primary key of Rel
3. Also include any simple (or simple components of composite) attributes of R as attributes of Rel



Step 5 example



Employee(Fname, Minit, Lname, Ssn, Bdate, Address, Sex, Salary, Super_ssn, Dno)

Department(Dname, Dnumber, Mgr_ssn, Mgr_start_date)

Project(Pname, Pnumber, Plocation, Dnum)

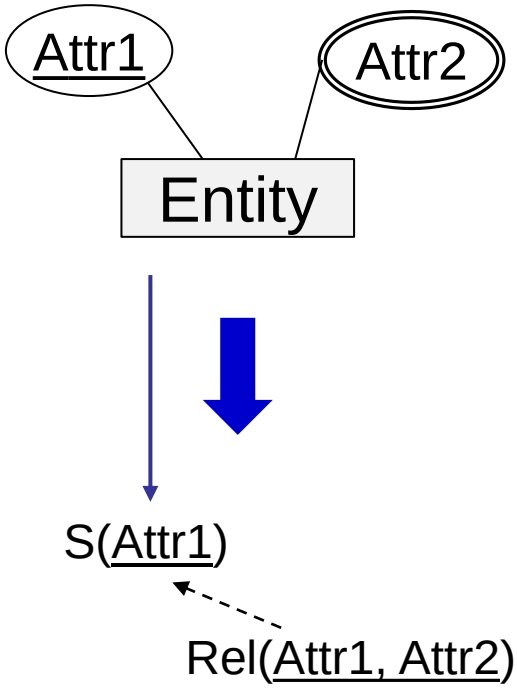
Works_on(Essn, Pno, Hours)

Dependent(Essn, Dependent_name, Sex, Bdate, Relationship)

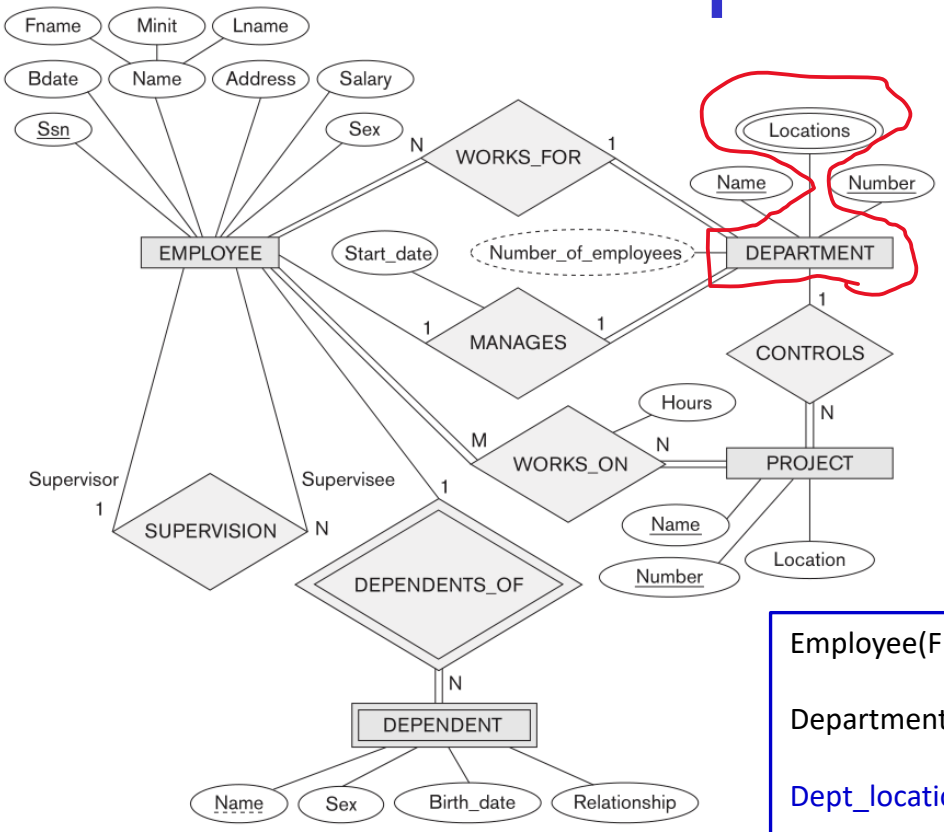
Step 6: Mapping of Multivalued attributes

For each multivalued attribute A:

1. Create a new relation Rel
2. Include an attribute corresponding to A, plus the primary key attribute K—as a foreign key in Rel—of the relation that represents the entity type (or relationship type) that has A as a multivalued attribute
3. The primary key of Rel is the combination of A and K
4. If the multivalued attribute is composite, include its simple components



Step 6 example



Employee(Fname, Minit, Lname, Ssn, Bdate, Address, Sex, Salary, Super_ssn, Dno)

Department(Dname, Dnumber, Mgr_ssn, Mgr_start_date)

Dept_locations(Dnumber, Dlocation)

Project(Pname, Pnumber, Plocation, Dnum)

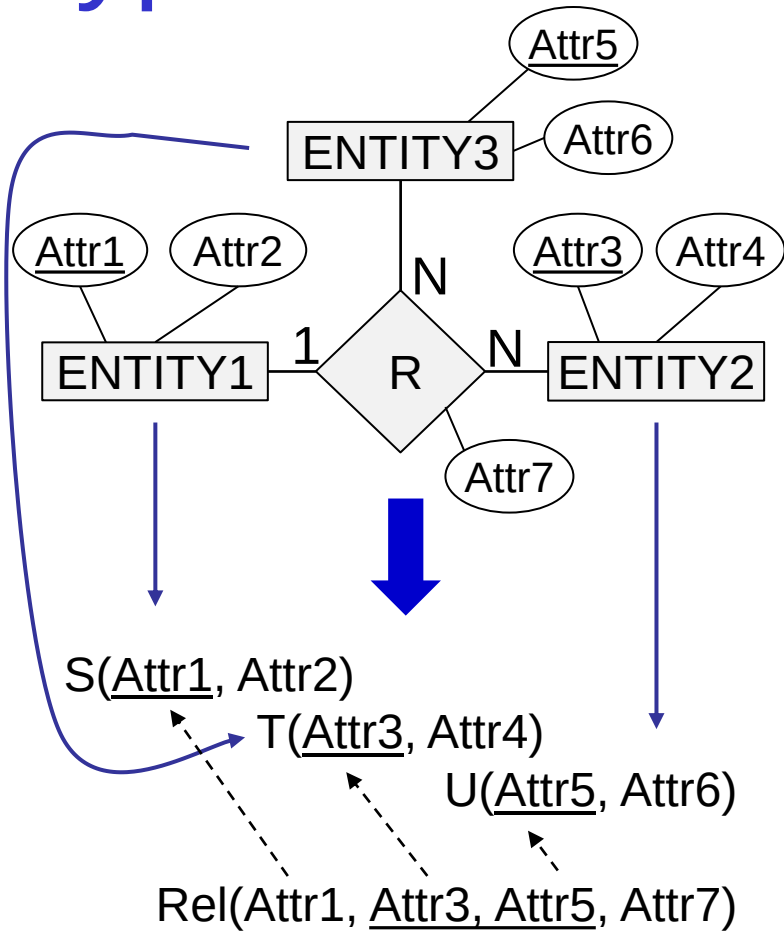
Works_on(Essn, Pno, Hours)

Dependent(Essn, Dependent_name, Sex, Bdate, Relationship)

Step 7: Mapping of Ternary Relationship Types

For each ternary relationship type R:

- Create a new relationship relation Rel to represent R
- Include as foreign key attributes in Rel the primary keys of the relations that represent the participating entity types (S, T, U); also include any simple (or simple components of) attributes of R as attributes of Rel
- Candidate keys in Rel are established as follows – if S:T:U is:
  - 1:1:1: three candidate keys formed by the combination of primary keys of (S,T), (S,U), and (T,U) respectively
  - 1:1:N: two candidate keys formed by the combination of primary keys of (S,U), and (T,U) respectively
  - 1:N:N: one candidate key formed by the combination of primary keys of (T,U)
  - N:N:N: one candidate key formed by the combination of primary keys of (S, T,U)



Correspondence between ER and relational models

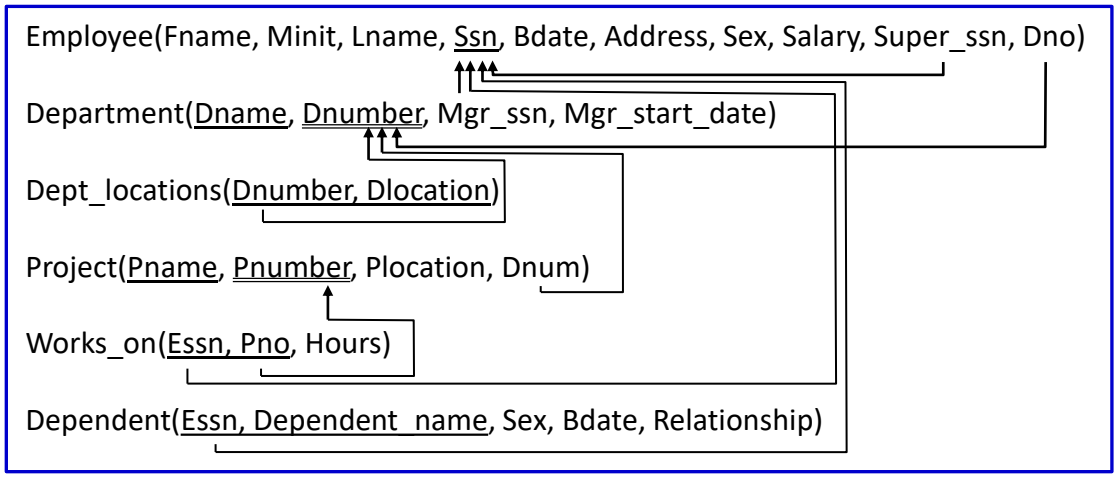
ER Model	**Relational Model**
Entity type	“Entity” relation
1:1 or 1:N relationship type	Foreign key (or “relationship” relation)
M:N relationship type	“Relationship” relation and two foreign keys
n-ary relationship type	“Relationship” relation and n foreign keys
Simple attribute	Attribute
Composite attribute	Set of simple component attributes
Multivalued attribute	Relation and foreign key
Value set	Domain
Key attribute	Primary (or secondary) key

Key difference between ER and relational models

- **Relationship types are not represented explicitly in the relational model**, but implicitly by having two attributes A and B, one a primary key and the other a foreign key (over the same domain) included in two relations S and T, respectively
- Two tuples in S and T are related when they have the same value for A and B
- By using the EQUIJOIN operation over S.A and T.B, we can combine all pairs of related tuples from S and T and materialize the relationship
  - M:N implies two joins
  - 1:N implies one or two joins
  - 1:1 implies zero, one or two joins
  - N-ary implies N joins

When multiple relationships need to be traversed, numerous join operations must be specified. The user must always be aware of the foreign key attributes in order to use them correctly in combining related tuples from two or more relations.

Example



Create a relation that includes the employee name, project name, and hours that the employee works on each project

$$\pi_{Fname,Minit,Lname,Pname,Hours}((Employee \bowtie_{Ssn=Essn} Works\_on) \bowtie_{Pno=Pnumber} Project))$$

- Does it make sense to join the Project and Dept_locations relations on the condition Dlocation=Plocation ?

Summary ER modeling

- Conceptual database deign is essential in building DBs
- Conceptual design follows requirements analysis
- ER is a popular choice for the conceptual design phase (some systems use the ER model for relational schema design)
- Basic ER concepts: entities, relationships, attributes, constraints
- ER models are usually easier to understand than relational models
- ER models express relationships explicitly
- Procedure to convert ER diagrams to relational schemas