

CSE 350: Advanced Data Structures and Indexes (Spring 2026)

Lecture 24: Entity Relational Model to Relational Model

4/23/2026



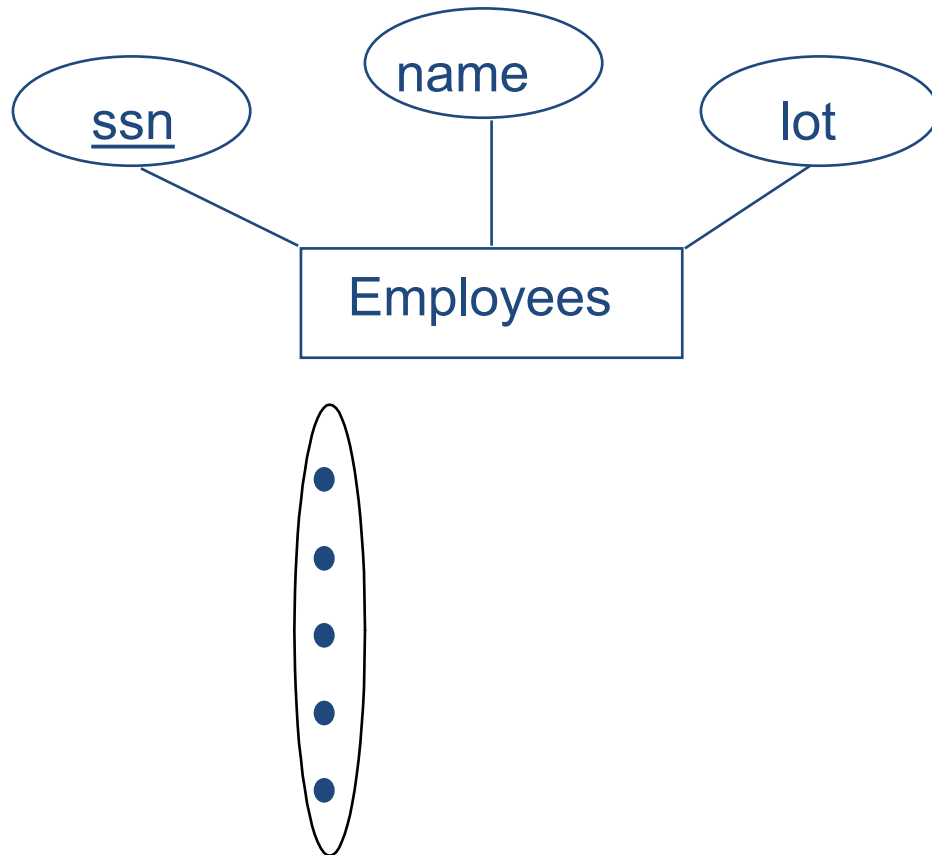
University at Buffalo

Department of Computer Science
and Engineering

School of Engineering and Applied Sciences

Logical DB Design: ER to Relational

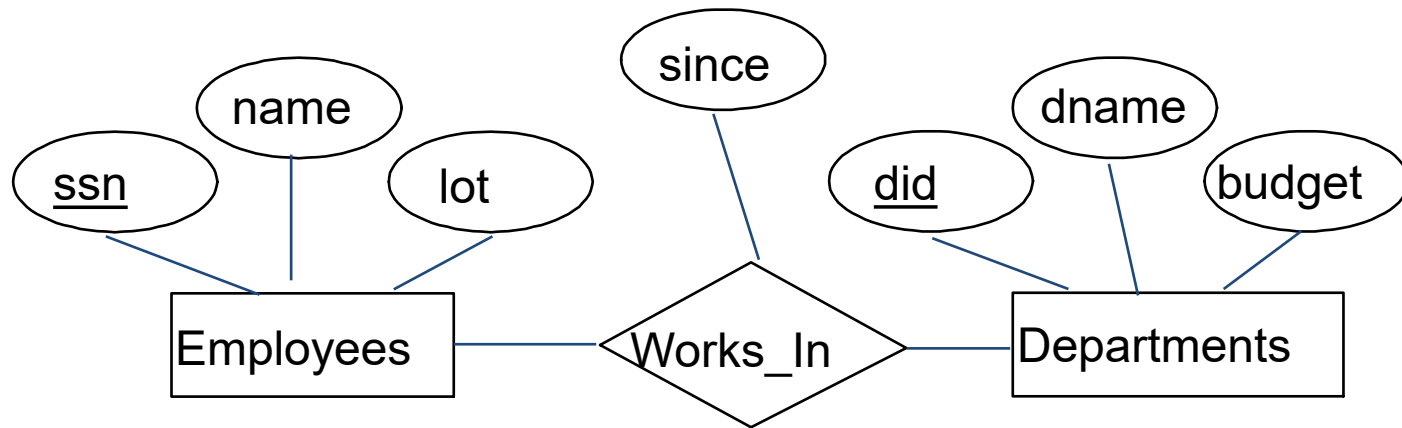
- Entity sets to tables.



ssn	name	lot
123-22-3666	Attishoo	48
231-31-5368	Smiley	22
131-24-3650	Smethurst	35

```
CREATE TABLE Employees  
(ssn CHAR(11),  
name CHAR(20),  
lot INTEGER,  
PRIMARY KEY (ssn))
```

Relationship Sets to Tables



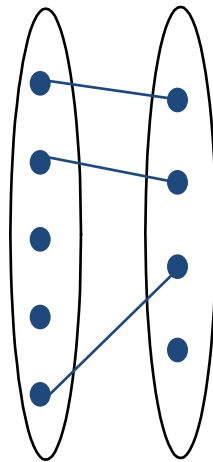
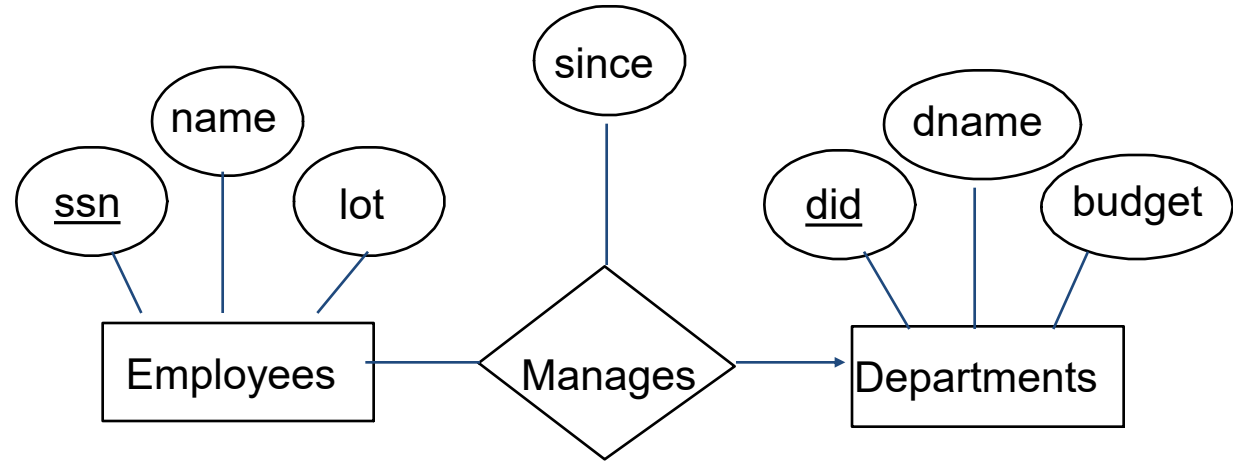
```
CREATE TABLE Works_In(  
  ssn CHAR(1),  
  did INTEGER,  
  since DATE,  
  PRIMARY KEY (ssn, did),  
  FOREIGN KEY (ssn)  
    REFERENCES Employees,  
  FOREIGN KEY (did)  
    REFERENCES Departments)
```

- In translating a **many-to-many** relationship set to a relation, attributes of the relation must include:
 - Keys for each participating entity set (as foreign keys).
 - This set of attributes forms a **superkey** for the relation.
 - All descriptive attributes.

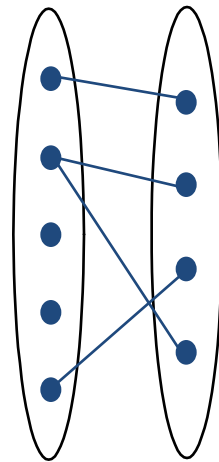
ssn	did	since
123-22-3666	51	1/1/91
123-22-3666	56	3/3/93
231-31-5368	51	2/2/92

Review: Key Constraints

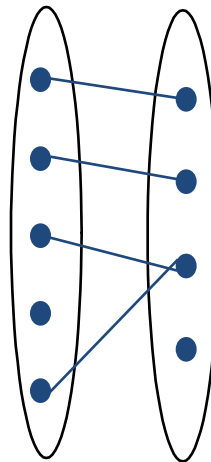
- Each dept has *at most one* manager, according to the key constraint on Manages.



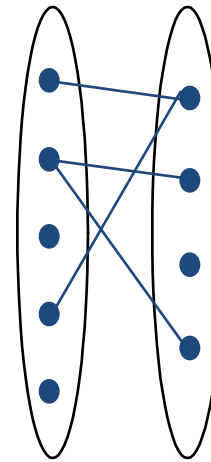
1-to-1



1-to Many



Many-to-1



Many-to-Many

Translation to relational model?

Translating ER Diagrams with Key Constraints

- Map relationship set to a table:
 - Note that `did` is the key now!
 - Separate tables for Employees and Departments.

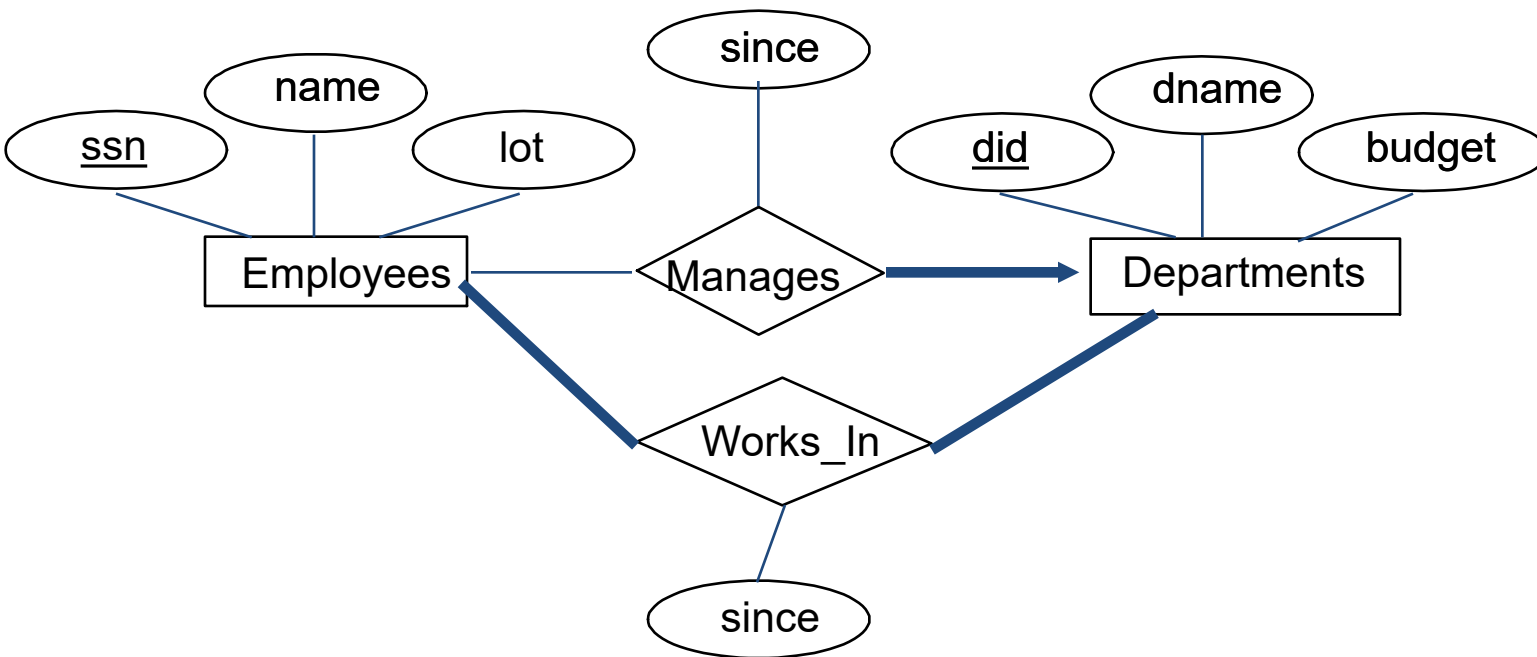
- Since each department has a unique manager, we could instead combine Manages and Departments.

```
CREATE TABLE Manages(  
  ssn CHAR(11),  
  did INTEGER,  
  since DATE,  
  PRIMARY KEY (did),  
  FOREIGN KEY (ssn) REFERENCES Employees,  
  FOREIGN KEY (did) REFERENCES Departments)
```

```
CREATE TABLE Dept_Mgr(  
  did INTEGER,  
  dname CHAR(20),  
  budget REAL,  
  ssn CHAR(11),  
  since DATE,  
  PRIMARY KEY (did),  
  FOREIGN KEY (ssn) REFERENCES Employees)
```

Review: Participation Constraints

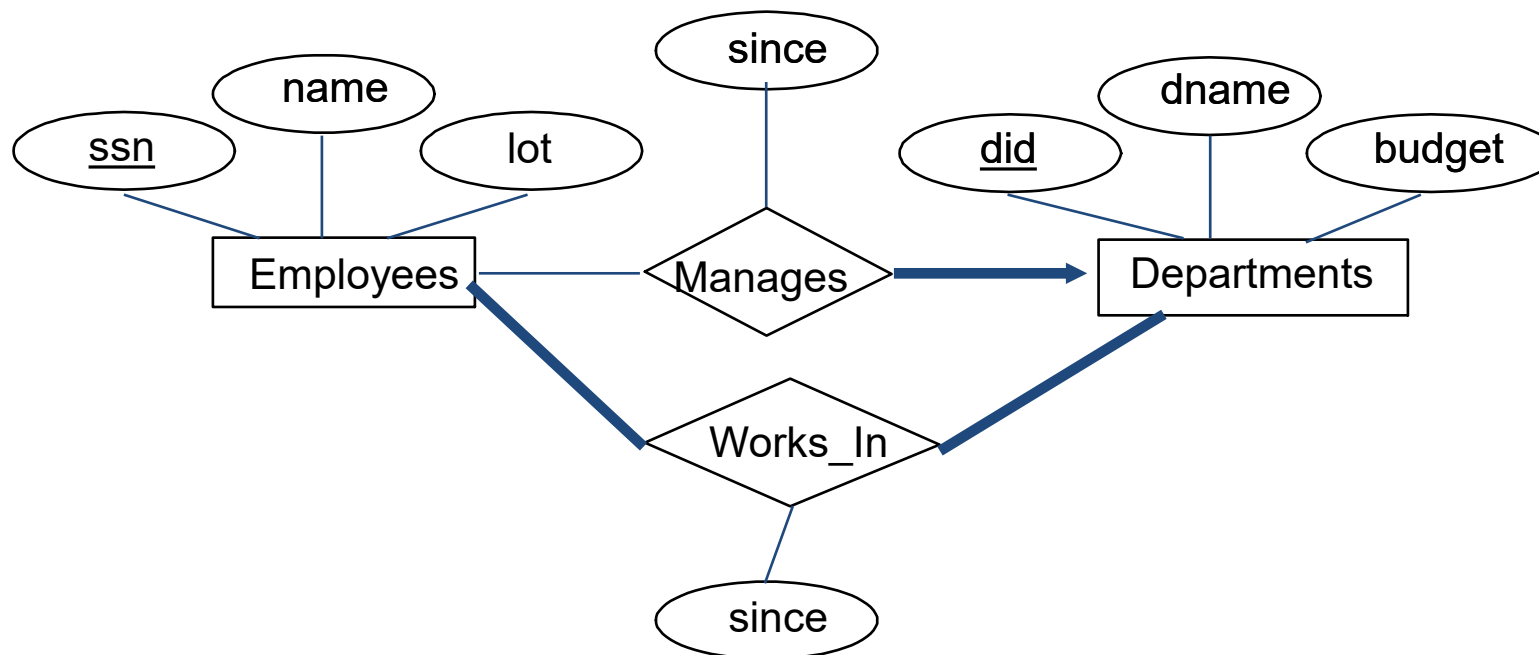
- Does every department have a manager?
 - If so, this is a participation constraint: the participation of Departments in Manages is said to be *total* (vs. *partial*).
 - Every *did* value in Departments table must appear in a row of the Manages table (with a non-null *ssn* value!)



```
CREATE TABLE Dept_Mgr(  
  did INTEGER,  
  dname CHAR(20),  
  budget REAL,  
  ssn CHAR(11) NOT NULL,  
  since DATE,  
  PRIMARY KEY (did),  
  FOREIGN KEY (ssn)  
    REFERENCES Employees,  
  ON DELETE NO ACTION)
```

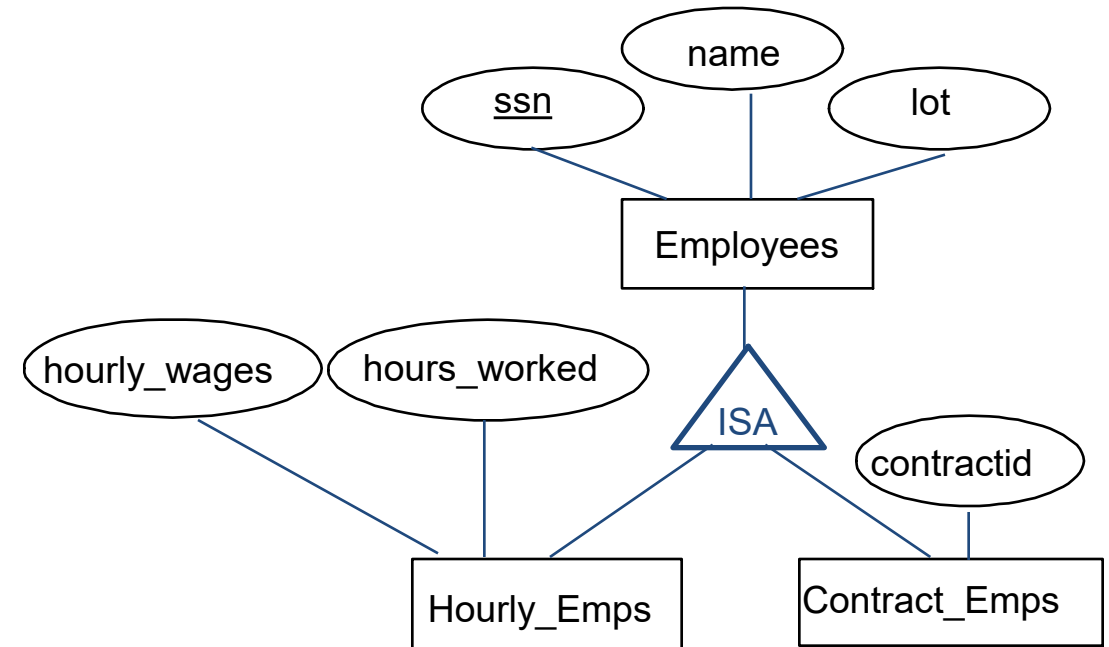
Participation Constraints in SQL

- We can capture participation constraints involving one entity set in a binary relationship, but little else (without resorting to CHECK constraints).



Review: ISA Hierarchies

- As in C++, or other PLs, attributes are inherited.
- If we declare A *ISA* B, every A entity is also considered to be a B entity.



- *Overlap constraints*: Can Joe be an Hourly_Emps as well as a Contract_Emps entity? (*Allowed/disallowed*)
- *Covering constraints*: Does every Employees entity also have to be an Hourly_Emps or a Contract_Emps entity? (*Yes/no*)

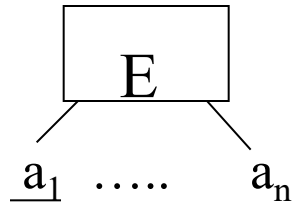
Translating ISA Hierarchies to Relations

- *General approach:*
 - 3 relations: Employees, Hourly_Emps and Contract_Emps.
 - *Hourly_Emps*: Every employee is recorded in Employees. For hourly emps, extra info recorded in Hourly_Emps (*hourly_wages*, *hours_worked*, *ssn*); must delete Hourly_Emps tuple if referenced Employees tuple is deleted).
 - Queries involving all employees easy, those involving just Hourly_Emps require a join to get some attributes.
- *Alternative: Just Hourly_Emps and Contract_Emps.*
 - *Hourly_Emps*: *ssn*, *name*, *lot*, *hourly_wages*, *hours_worked*.
 - Each employee must be in one of these two subclasses.

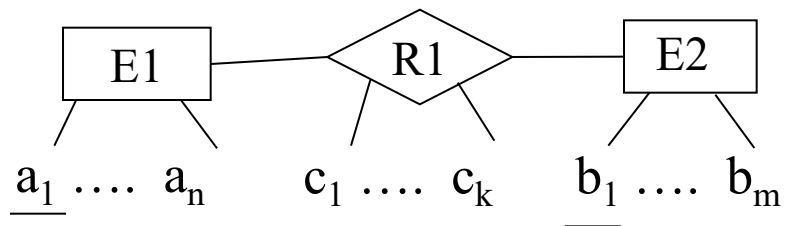
E/R to Relations

E/R diagram

Relational schema, e.g.
account=(bname, acct_no, bal)



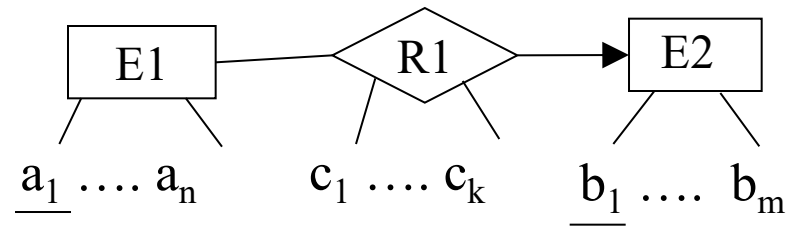
$E = (\underline{a_1}, \dots, a_n)$



$R1 = (\underline{a_1}, \underline{b_1}, c_1, \dots, c_k)$

More on relationships

- What about:

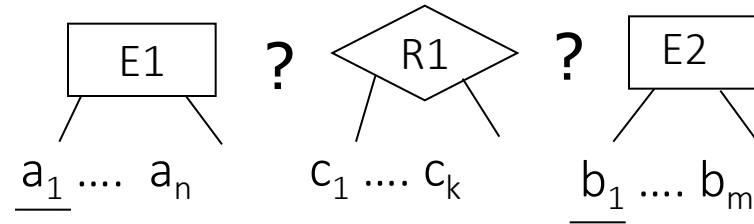


- Could have :

$$R1 = (a_1, \underline{b_1}, c_1, \dots, c_k)$$

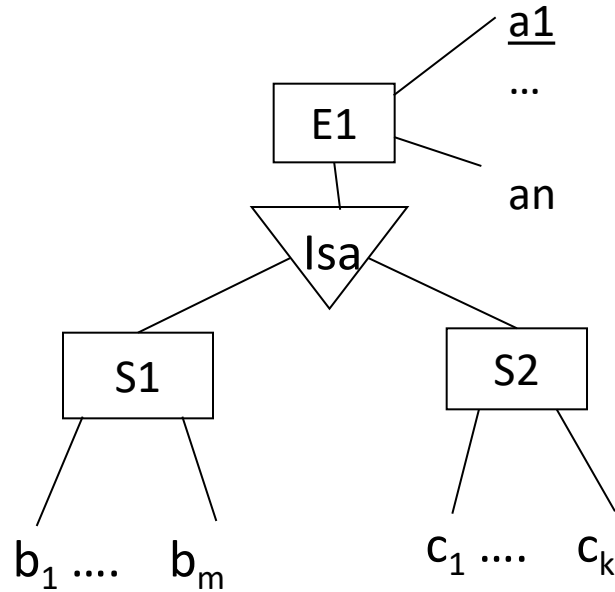
- Use b_1 as the key for R1, it is also the key for $E2 = (b_1, \dots, b_m)$
- Usual strategy:
 - ignore R1
 - Add a_1, c_1, \dots, c_k to E2 instead, i.e.
 - $E2 = (\underline{b_1}, \dots, b_m, a_1, c_1, \dots, c_k)$

More



	$E1 = (\underline{a_1}, \dots, a_n)$ $E2 = (\underline{b_1}, \dots, b_m)$ $R1 = (\underline{a_1}, \underline{b_1}, c_1, \dots, c_k)$
	$E1 = (\underline{a_1}, \dots, a_n)$ $E2 = (\underline{b_1}, \dots, b_m, a_1, c_1, \dots, c_k)$
	$E1 = (\underline{a_1}, \dots, a_n, b_1, c_1, \dots, c_k)$ $E2 = (\underline{b_1}, \dots, b_m,)$
	<p>Treat as n:1 or 1:m Foreign key also needs to be marked as a candidate key!</p>

E/R to Relational



Method 1: $E = (\underline{a_1}, \dots, a_n)$

$S1 = (\underline{a_1}, b_1, \dots, b_m)$

$S2 = (\underline{a_1}, c_1, \dots, c_k)$

Method 2:

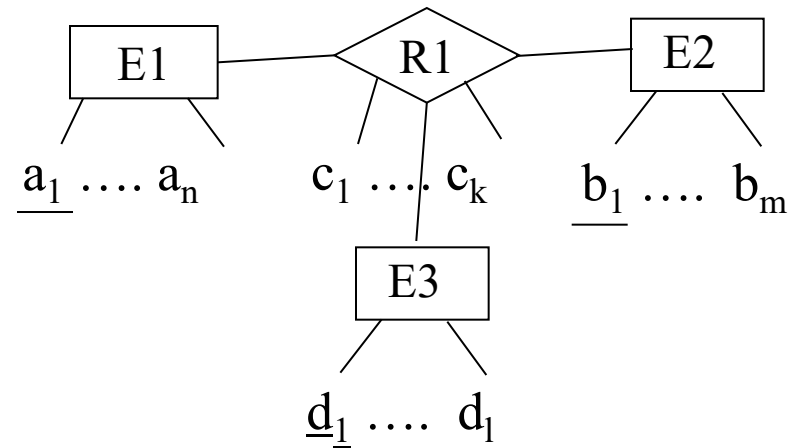
$S1 = (\underline{a_1}, \dots, a_n, b_1, \dots, b_m)$

$S2 = (\underline{a_1}, \dots, a_n, c_1, \dots, c_k)$

Q: When is method 2 not possible?

Tenary relationshipset:

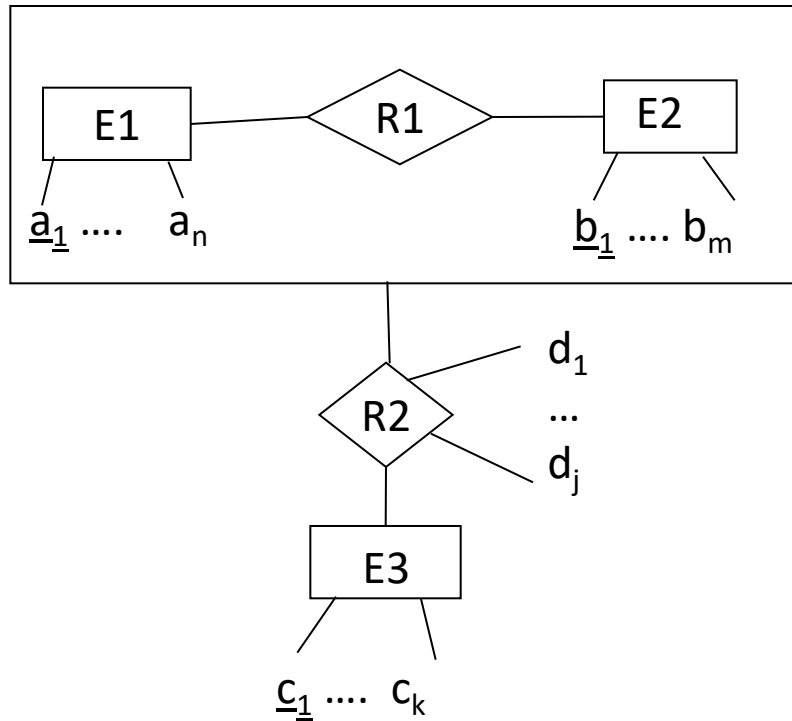
- What about tenary:



- Strategy:
 - $E1(\underline{a_1} \dots a_n) E2(\underline{b_1} \dots b_m) E3(\underline{d_1} \dots d_l)$
 - $R1(\underline{a_1}, \underline{b_1}, \underline{d_1}, c_1, \dots, c_k)$

E/R to Relational

- Aggregation



E1, R1, E2, E3 as before

$R2 = (\underline{c_1}, \underline{a_1}, \underline{b_1}, d_1, \dots, d_j)$

ER Model to Relation Model Mapping Summary

- Usually easier to understand than Relational
- Expresses relationships clearly
- Rules to convert ER-diagrams to Relational Schema
- Some systems use ER-model for schema design
- Many use ER-model as a step before creating relational tables