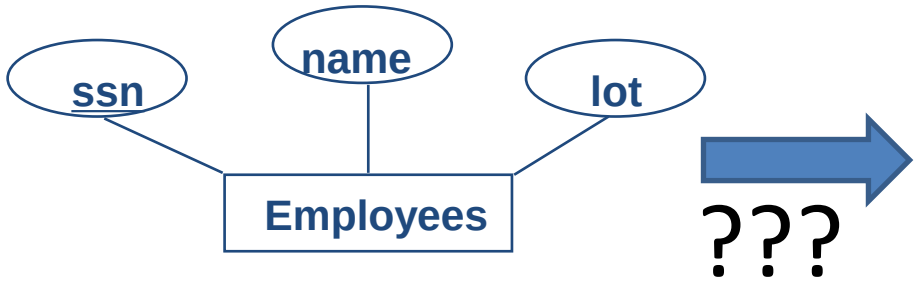
ER to Relational Mapping (Part I)

So Far We have



ER diagram

ssn	Name	lot
213456789	Adam	L2
123983109	Betty	L1

Relational table

How can we go from the ER-diagrams to relational tables?

From ER Model to Relational Model

Basic steps of converting ER model to relational tables:

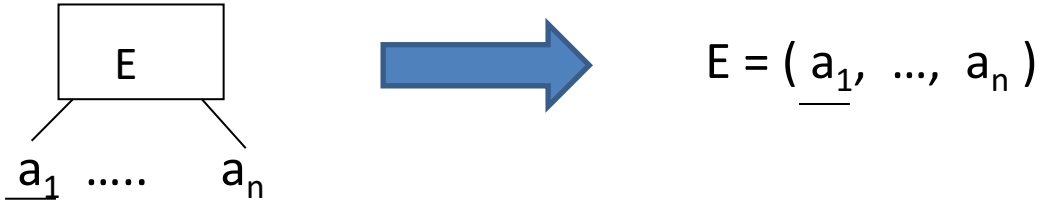
- For each entity set: construct a table
- For each relationship set: construct a table if necessary (some relationship sets do not need a table)
- For each table, decide its schema:
  - Decide the attributes of these tables
  - Decide primary key, candidate keys, and foreign keys of these tables

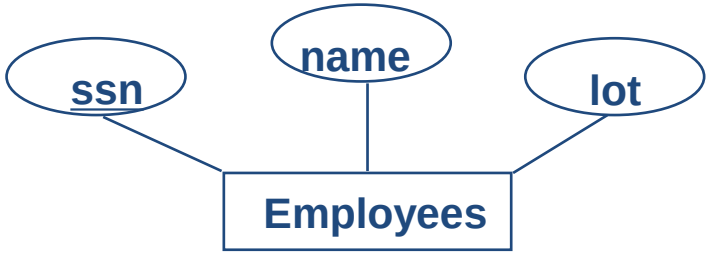
Today's lecture

- Translate ER diagrams into relational tables
  - Entity sets
    - Strong entity sets
    - Weak entity sets
  - Relationship sets

Translating Strong Entity Sets

- Rule

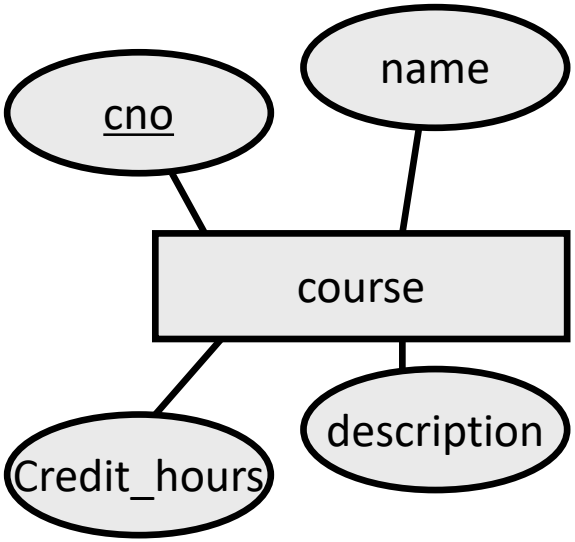




Schema: Employees (ssn, name, lot)

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

Exercise

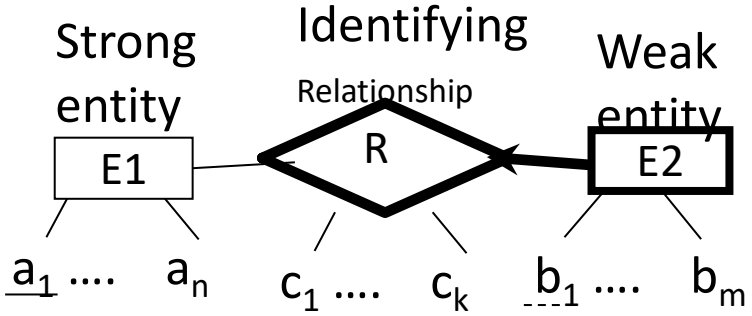


Translate this ER diagram to relational table.

Schema: course (cno, name, credit_hours, description)

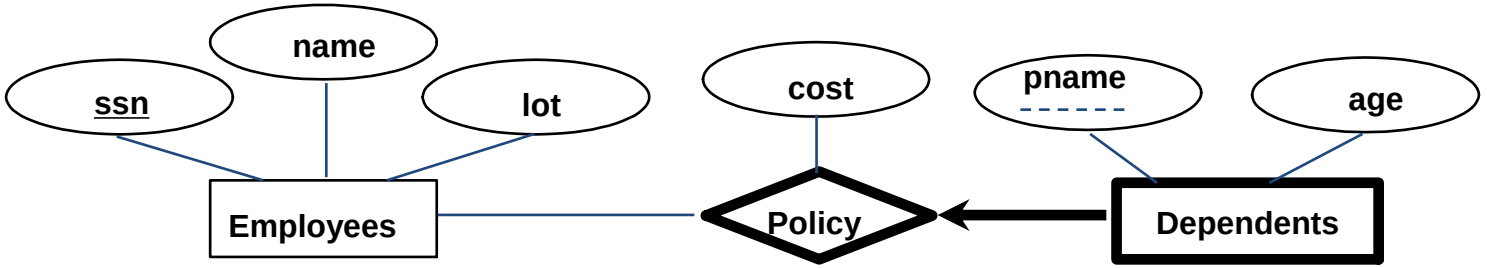
```
CREATE TABLE course (
                    cno INTEGER,
                    name CHAR(20),
                    credit_hours  INTEGER,
                    description CHAR(50),
                    PRIMARY KEY (cno)
);
```

Translating Weak Entity Sets



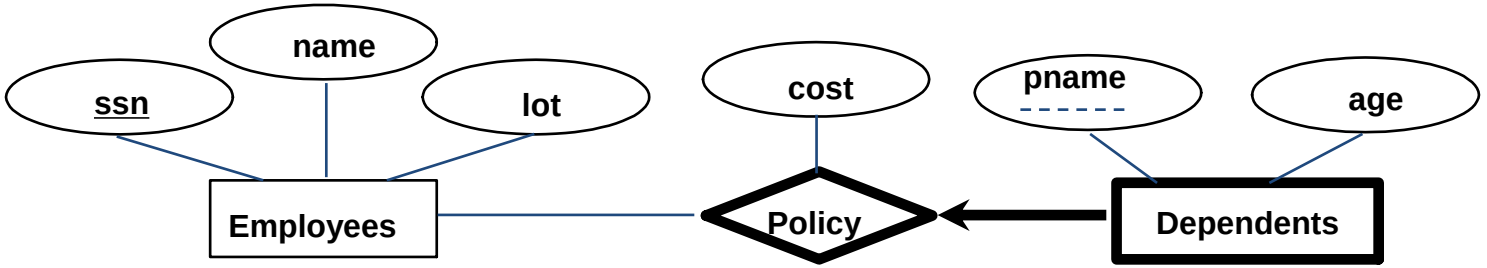
- Strong entity set (as usual): E1 = ($\underline{a_1}$, …, a_n)
  - Primary key (a1)
  - No foreign key
- Weak entity set: E2 = ($\underline{a_1}$, $\underline{b_1}$, …, b_m , c_1, …, c_k)
  - *Attributes*: $\underline{b_1}$, …, b_m (i.e., attributes of E2) + c_1 …. c_k (i.e., attributes of R) **+** a_1 (i.e., the primary key of E1).
  - *Primary key*: (a_1, b_1) - a composite key
  - *Foreign key:*
    - a_1 (reference table E1)
- There is NO table for the relation R.

Translating Weak Entity Sets: Example 1



Translate this ER diagram to relational tables.

Translating Weak Entity Sets: Example 1



Translate this ER diagram to relational tables.

Schema:

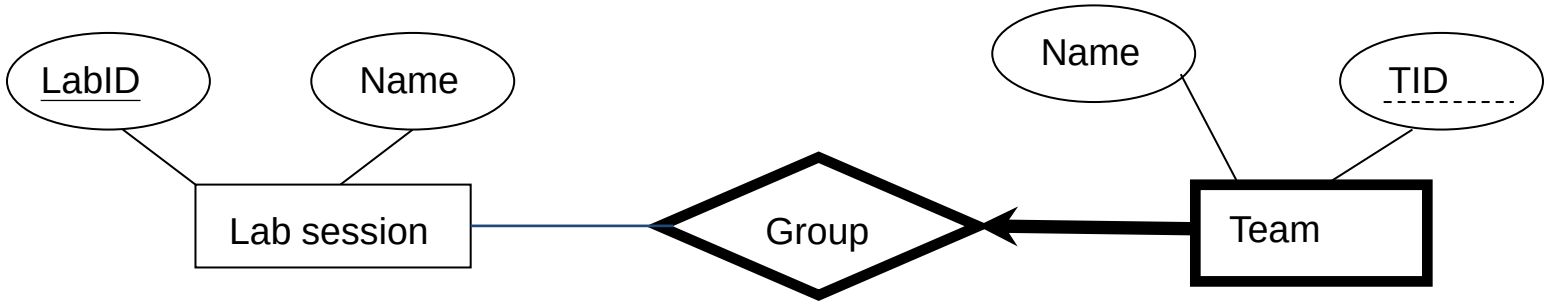
Strong entity: *Employees (ssn, name, lot),*

Weak entity: *Dep_Policy(pname, ssn_employee, age, cost):*

Foreign key ssn_employee ref. Emploees.ssn

```
CREATE TABLE  Dep_Policy (pname  CHAR(20),
                          age  INTEGER,
                          cost  FLOAT,
                          ssn_employee  CHAR(10),
                          PRIMARY KEY  (pname, ssn_employee),
                          FOREIGN KEY  (ssn_employee) REFERENCES Employees(ssn)
 );
```

Translating Weak Entity Sets: Example 2



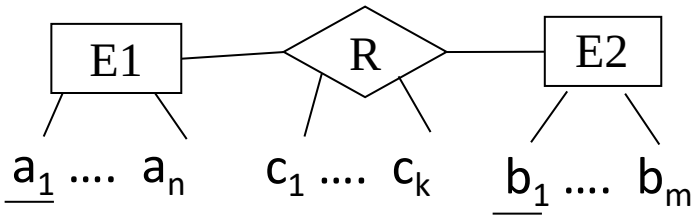
Write down the schema of *Lab* and *Team* tables (including the foreign keys)

Today's lecture

- Translate ER diagrams into relational tables
  - Entity sets
  - Relationship sets
    - Many-to-many cardinality constraint
    - One-to-many cardinality constraint
    - IsA relationship

Many-to-many (M-M) Relationship Sets (Binary relationships)

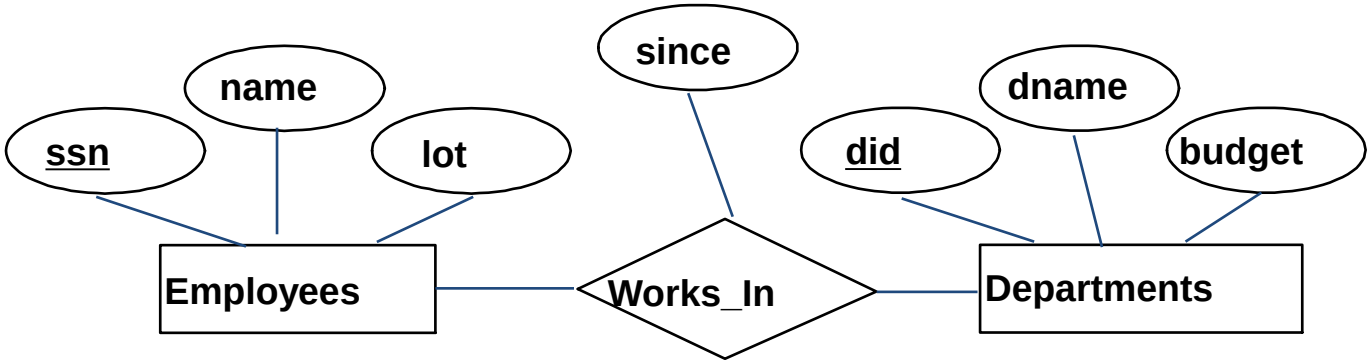
- Translate a binary M-M relationship set to a relation table R:
  - *Attributes*: <c_1 …. c_k> + a_1 + b_1 (i.e., the primary key of E1 & E2).
  - *Primary key*: (a_1, b_1) - a composite key
  - *Foreign keys:*
    - a_1 (reference table E1),
    - b_1 (reference table E2).



The ER diagram is mapped to 3 tables

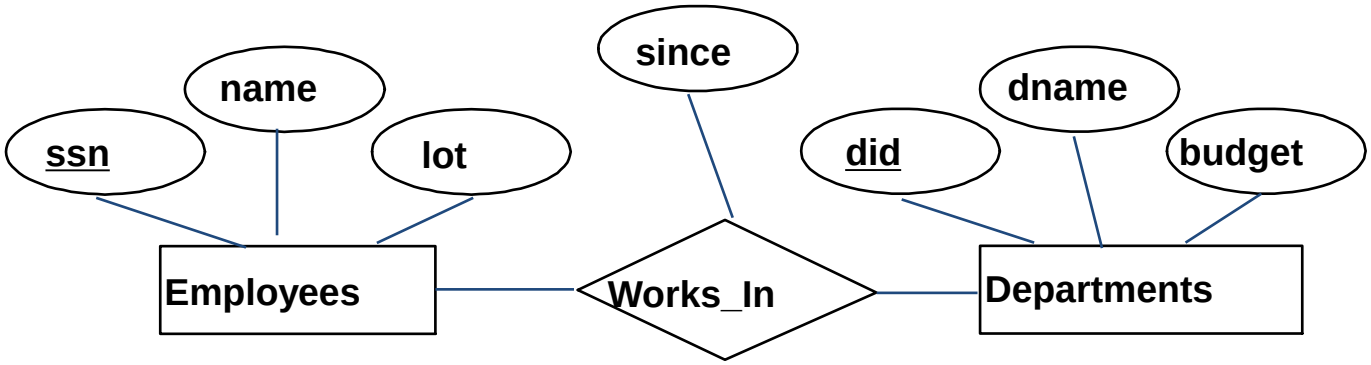
- 2 Tables for entity sets E1 & E2
  - E1($\underline{a_1}$, …, a_n)
  - E2($\underline{b_1}$, …, b_m)
- 1 Table for M-M relationship set R
  - R($\underline{a_1}$, $\underline{b_1}$, c_1, …, c_k) with 2 foreign keys

Example



1. What's the schema of *Works_In* table?
2. Write the SQL statement to construct the *Works_In* table

Example



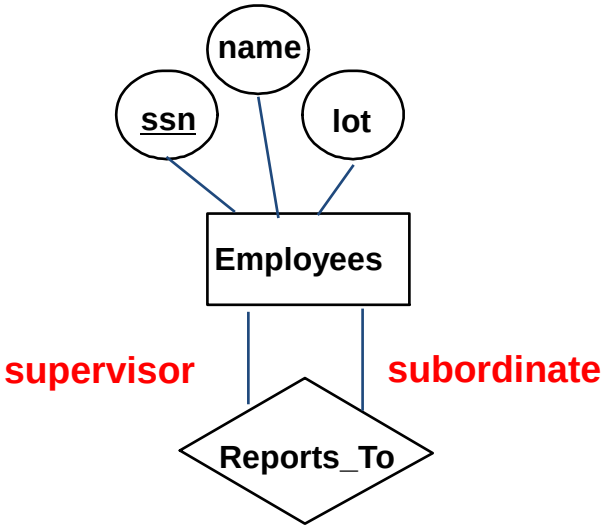
1. What's the schema of *Works_In* table?
2. Write the SQL statement to construct the *Works_In* table

Schema: Works_In(ssn, did, since) with two foreign keys: (ssn) references Employees.ssn, (did) references Departments.did

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

Relationship Sets with Different Roles

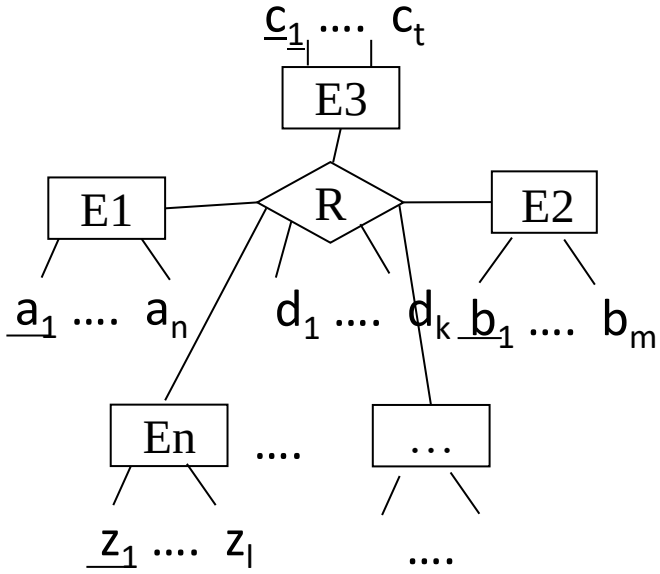
Write the SQL statement to construct the *Reports_To* table:



Schema: Employees(supervisor_ssn, subordinate_ssn) with two foreign keys: (supervisor_ssn) references Employees.ssn, (subordinate_ssn) references Employees.ssn

```
CREATE TABLE Reports_To(
         supervisor_ssn   CHAR(11),
         subordinate_ssn  CHAR(11),
         PRIMARY KEY (supervisor_ssn, subordinate_ssn),
         FOREIGN KEY (supervisor_ssn) REFERENCES Employees(ssn),
         FOREIGN KEY (subordinate_ssn)REFERENCES Employees(ssn)
);
```

Many-to-many (M-M Relationship Sets (More than 2 Relationships)



Step 1: Create n tables for *n* entity sets:

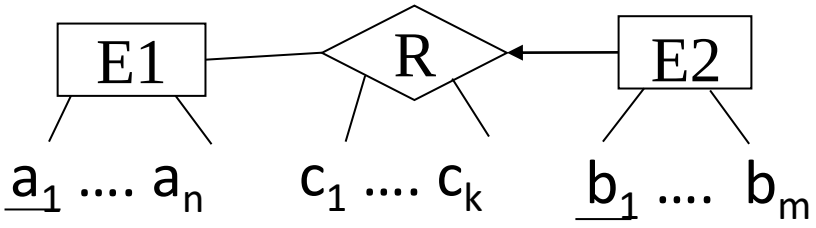
- E_1 (a_1, … a_n)
- E_2 (b_1, …b_m)
- E_3 (c_1, …c_t)
- …
- E_n(z_1, …z_l)

Step 2: create one table for the relationship R

$$R(a_1, b_1, c_1, \ldots z_1, d_1, \ldots, d_k)$$

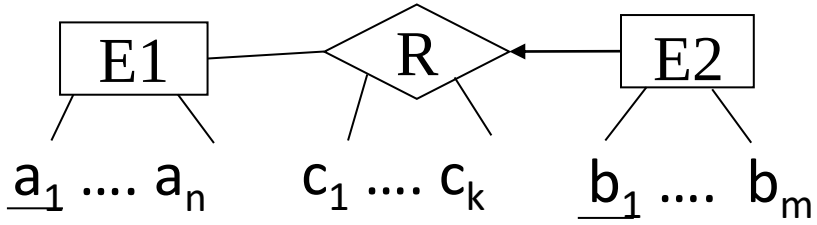
- Attributes: the primary keys a_1, b_1, c_1, …z_1, of n entity sets + attributes d_1, … d_k of R
- Primary key: (a_1, b_1, c_1, …z_1)
- n foreign keys:
  - a_1 (reference table E_1),
  - b_1 (reference table E_2),
  - c_1 (reference table E_3),
  - …
  - z_1 (reference table E_n).

Translate 1-to-Many (1-M) Relationship Set to Relational Table



- Two alternative solutions
  - Three-table solution: One table for E1, R, E2 each
  - Two-table solution: One table for E1 and E2 each. No table for R.

Solution 1: Translation of 1-M Relationship

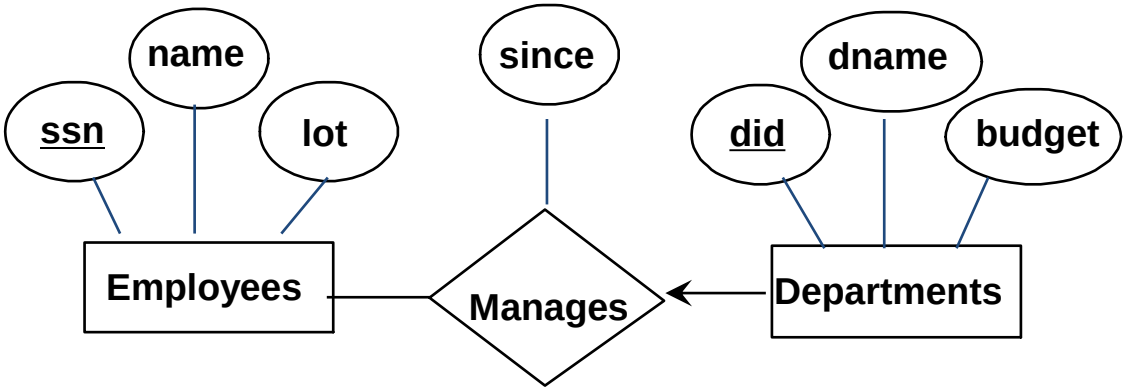


- Three-table solution
  - Two tables for E1 and E2 (as usual)
    - E_1 ($\underline{a_1}$, … a_n)
    - E_2 ($\underline{b_1}$, …b_m)
  - One table for R

$$R= (a_1, \underline{b_1}, c_1, \ \ldots, \ c_k)$$

    - The key of R: (b1). (Note: the key of R only contains the key of the entity set at Many-side)
    - R has two foreign keys:
      - a_1 (reference E1)
      - b_1 (reference E2)

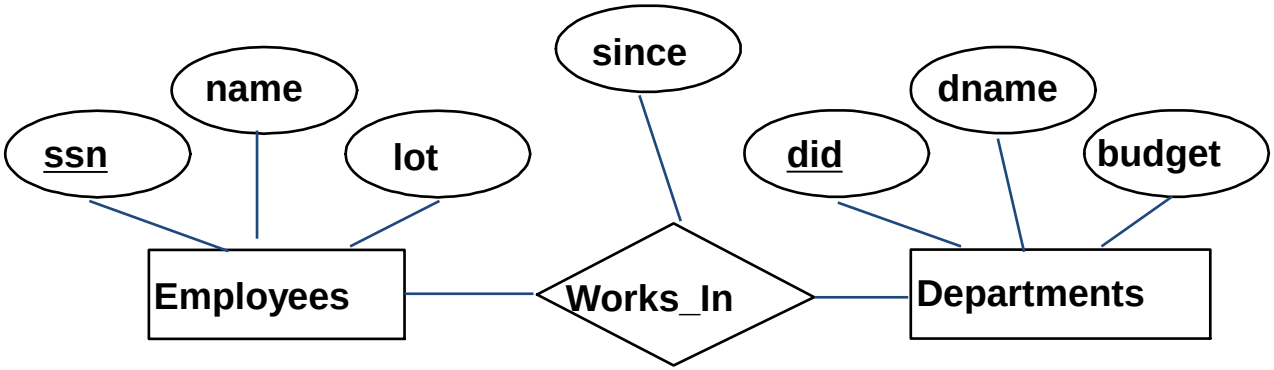
Example




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

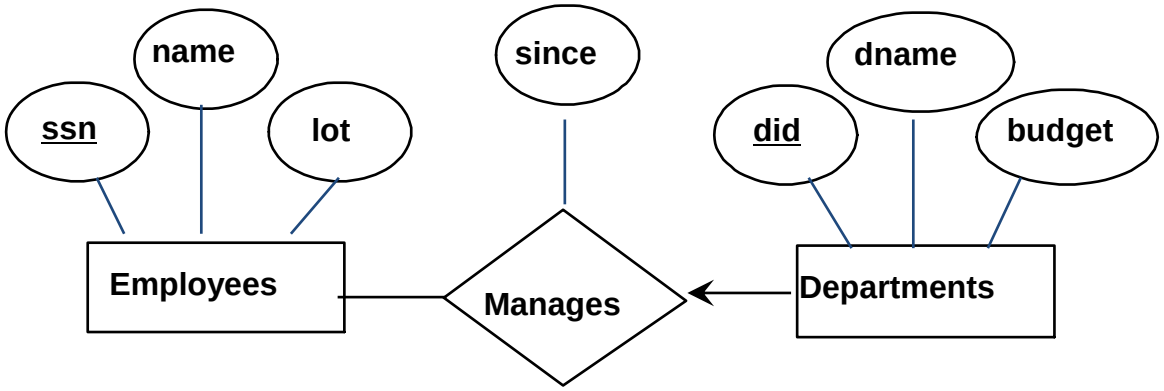
Note that the key only includes did!

Revisit the Example of M-M Relationship



SSN	did	Since
123-22-3666	1	01/01/2010
123-22-3666	2	02/03/2010
231-31-5368	1	01/01/2010

Why SSN can be removed from the Key in the 1-M relationship?



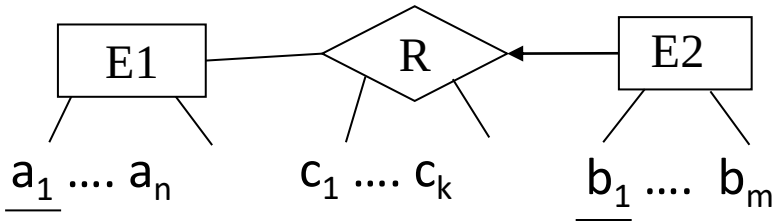
SSN	did	Since
123-22-3666	1	01/01/2010
123-22-3666	2	02/03/2010
231-31-5368	3	01/01/2010

Manages Relationship
(1-to-many relationship)
key: (did)

SSN	did	Since
123-22-3666	1	01/01/2010
123-22-3666	2	02/03/2010
231-31-5368	1	01/01/2010

Works-in Relationship
(Many-to-many relationship)
Key: (SSN, did)

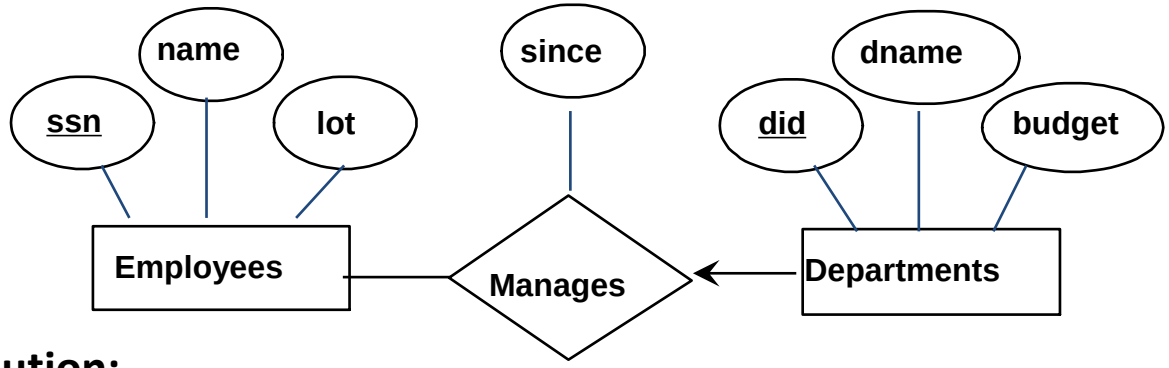
Solution 2: Translation of 1-M Relationship



Two-table solution:

- One table for E1 (entity set at 1-side)
  - E_1 ($\underline{a_1}$, … a_n)
- One table for E2 (entity set at Many-side)
  - E2=($\underline{b_1}$, …., b_m, a_1, c_1, …, c_k) (Note: E2 contains attributes of R and key of E1)
  - Foreign key of E2
    - a_1 (reference E1).
- No table for R

Example



Three-table solution:

- Employees (ssn, name, lot)
- Dept_Mgr (ssn, did, since)
- Departments (did, dname, budget)

SSN	did	Since
12345678	1	01/01/2010
12345678	2	02/03/2010
13452121	3	01/01/2010

Dept_Mgr table

did	dname	budget
1	HR	20000
2	Marketing	400000
3	IT	300000

Departments table

Two-table solution:

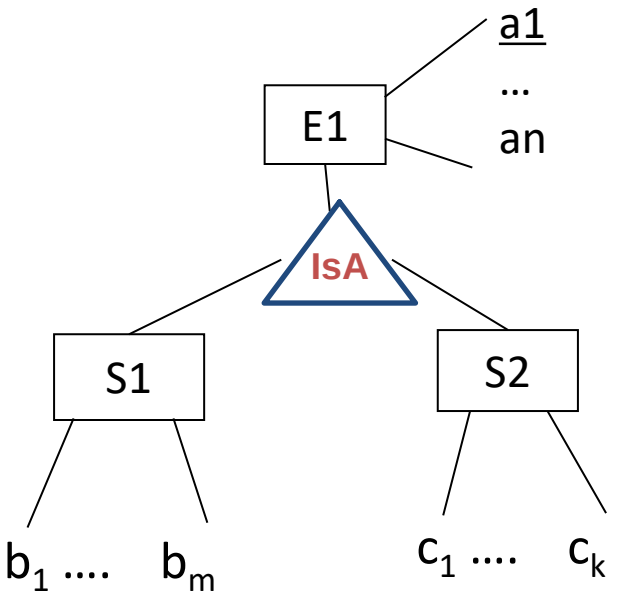
- Employees (ssn, name, lot)
- Departments (did, dname, budget, ssn, since)

did	dname	budget	SSN	Since
1	HR	20000	12345678	01/01/2010
2	Marketing	400000	12345678	02/03/2010
3	IT	300000	13452121	01/01/2010

Departments table

Note: in both solutions, Employees table is omitted as it is the same for both solutions.

Translating IsA to Relations



Solution 1:

- E1 ($\underline{a_1}$, …, a_n)
- S1 ($\underline{a_1}$, b_1, …, b_m)

Foreign key: a1 references E1(a1)

- S2($\underline{a_1}$, c_1 …, c_k)

Foreign key: a1 references E1(a1)

Solution 2:

- S1 ($\underline{a_1}$,…, a_n, b_1, …, b_m)
- S2 ($\underline{a_1}$, …, a_n, c_1 …, c_k)

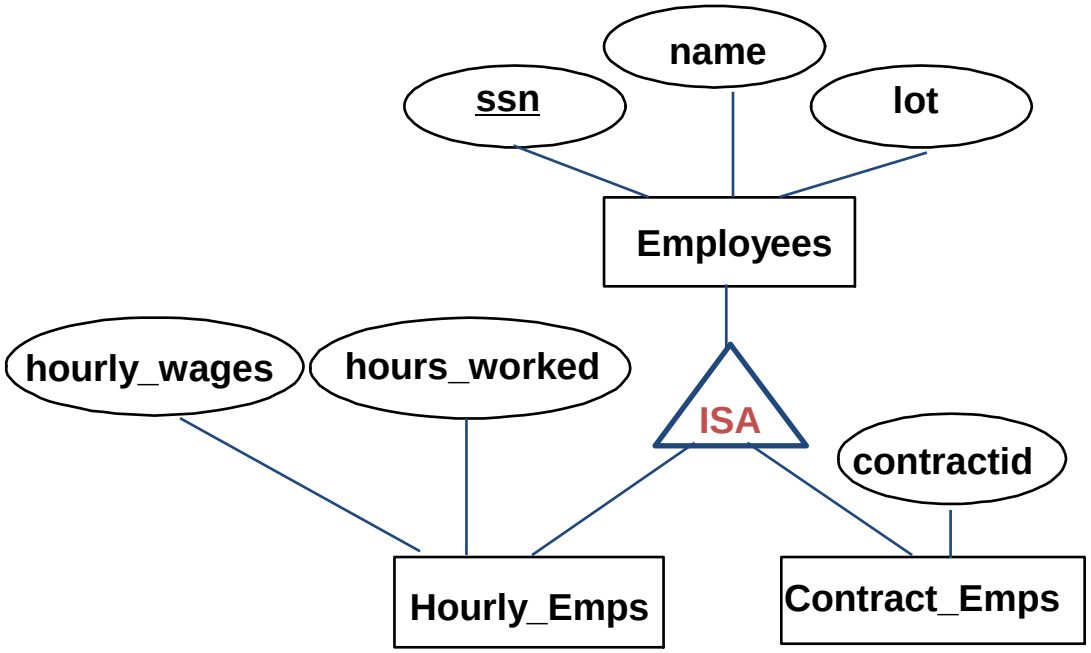
(no table for E1. All attributes of E1 are added to S1 and S2)

Q: When Solution 2 is wrong?

A: Solution 2 is wrong if IsA relationship does not have the ***covering constraint***

- In other words, there are some E1 entities that are neither S1 nor S2

Example: Translating ISA Hierarchies to Relations



- Solution 1*:*
  - 3 relations: Employees, Hourly_Emps and Contract_Emps.

- Solution 2: Just Hourly_Emps and Contract_Emps (if there is a covering constraint on ISA relationship).