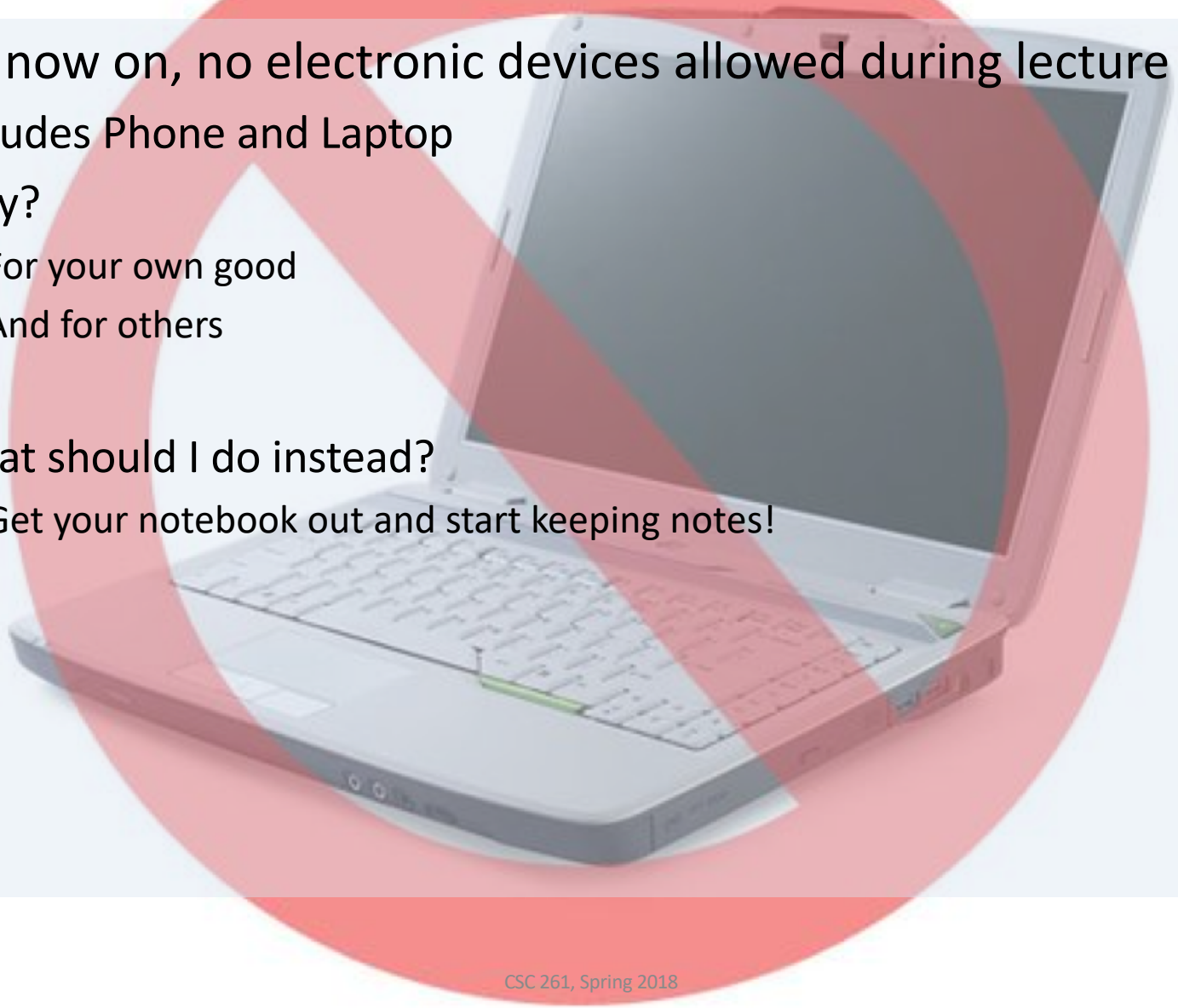
CSC 261/461 – Database Systems
Lecture 10

Spring 2018

Please put away all electronic devices

Announcements

- From now on, no electronic devices allowed during lecture
  - Includes Phone and Laptop
  - Why?
    - For your own good
    - And for others
  - What should I do instead?
    - Get your notebook out and start keeping notes!

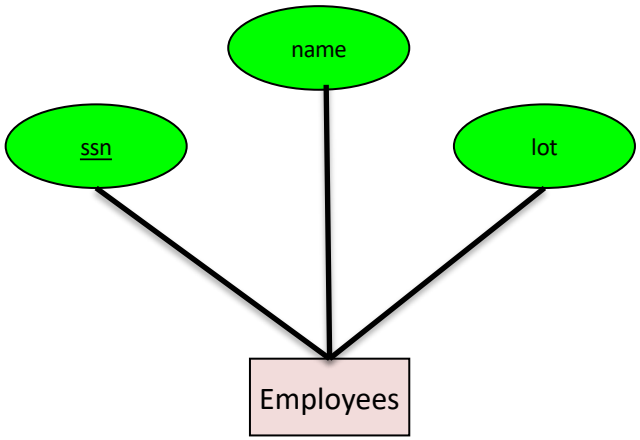


Design Theory (ER model to Relations)

ER Models to Relations

Please go through Chapter 9

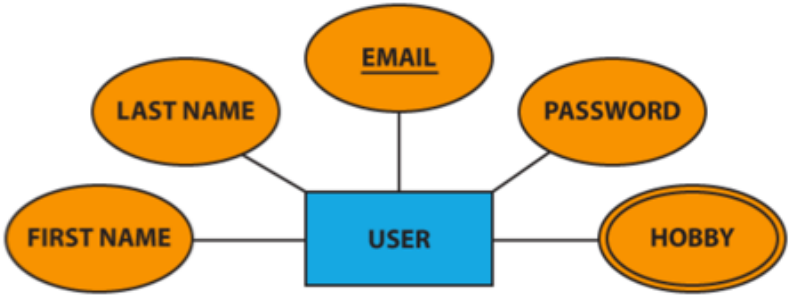
Entity Sets to Tables (Step 1)



ssn	name	lot
123-22-3333	Alex	23
234-44-6666	Bob	44
567-88-9787	John	12

```
CREATE TABLE Employees (   ssn char(11),
                           name varchar(30),
                           lot Integer,
                           PRIMARY KEY (ssn))
```

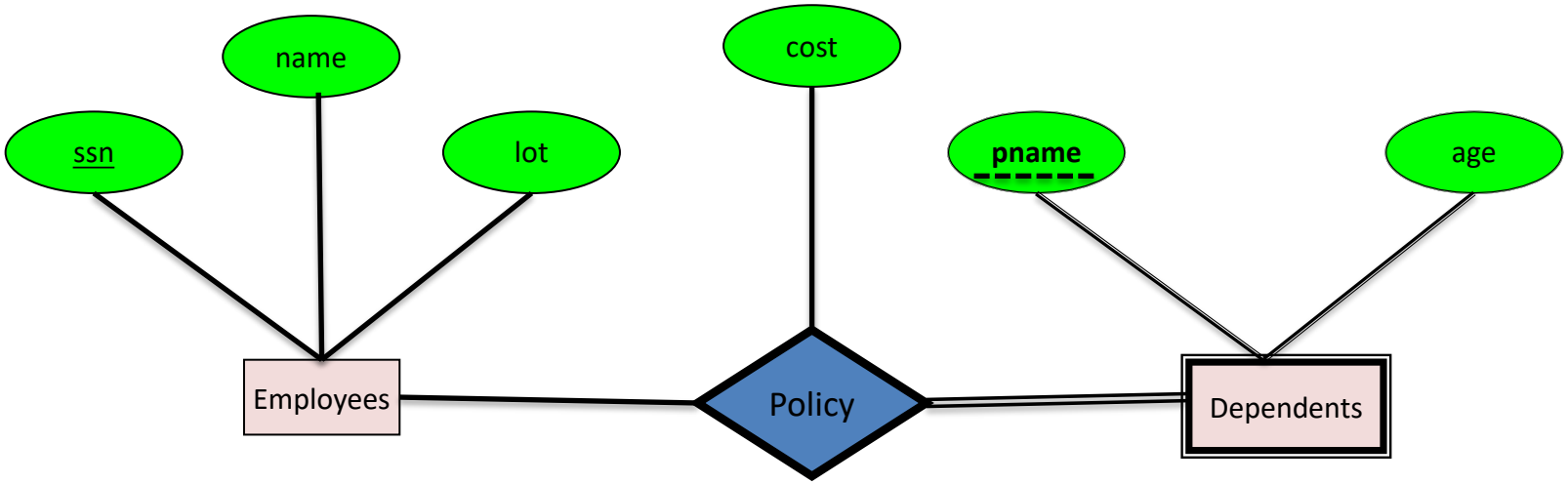
Multivalued Attribute



Multivalued Attribute

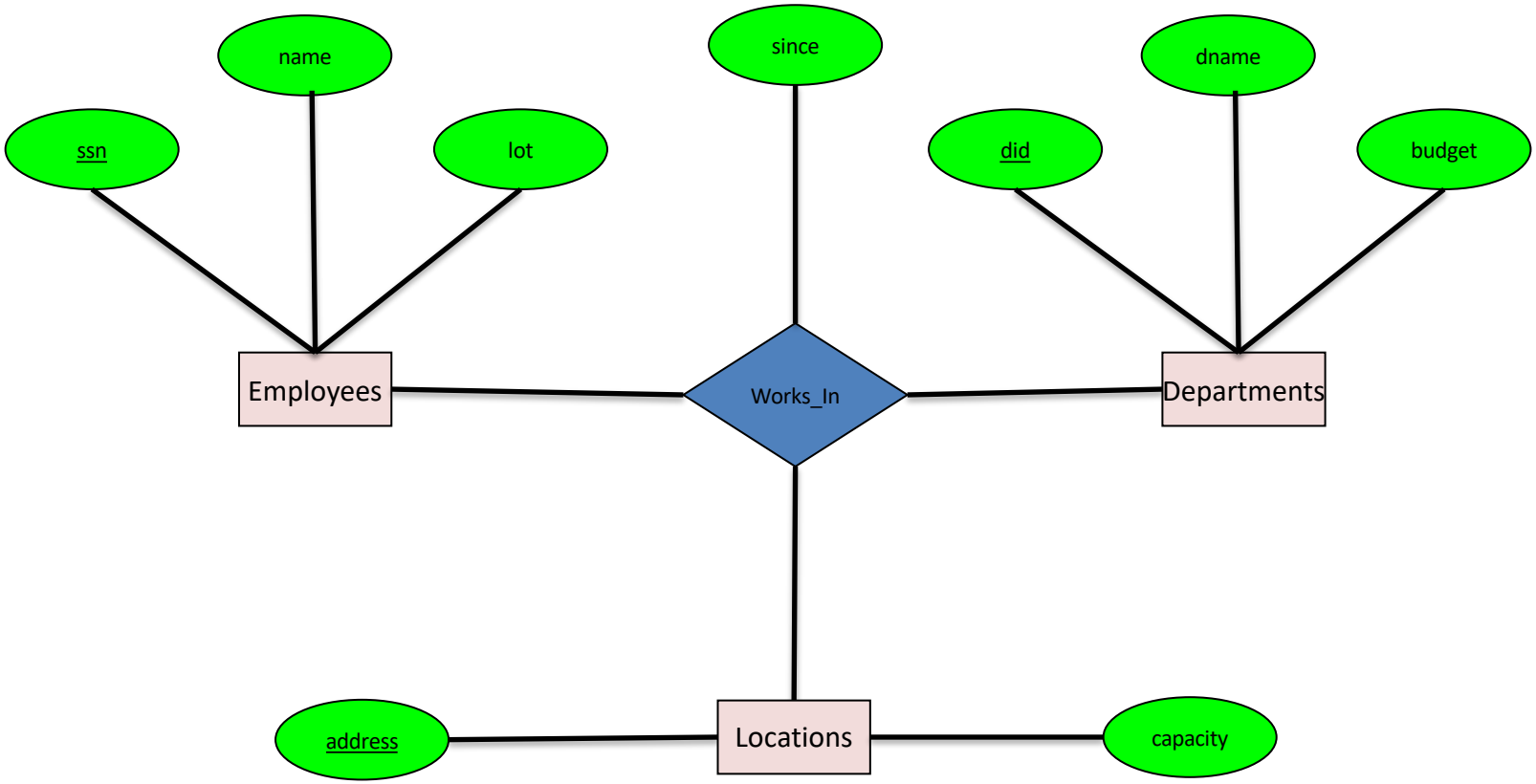
```
CREATE TABLE Hobby(
        Email varchar(30),
        Hobby varchar(30),
PRIMARY KEY (Email, Hobby),
FOREIGN KEY (Email) REFERENCES User(Email)
        ON DELETE CASCADE
)
```

Translating Weak Entity Sets




```
CREATE TABLE Dept_Policy(  pname varchar(30),
                           age integer,
                           cost float,
                           ssn char(11),
        PRIMARY KEY (pname, ssn),
        FOREIGN KEY (ssn) REFERENCES Employees
        ON DELETE CASCADE
)
```

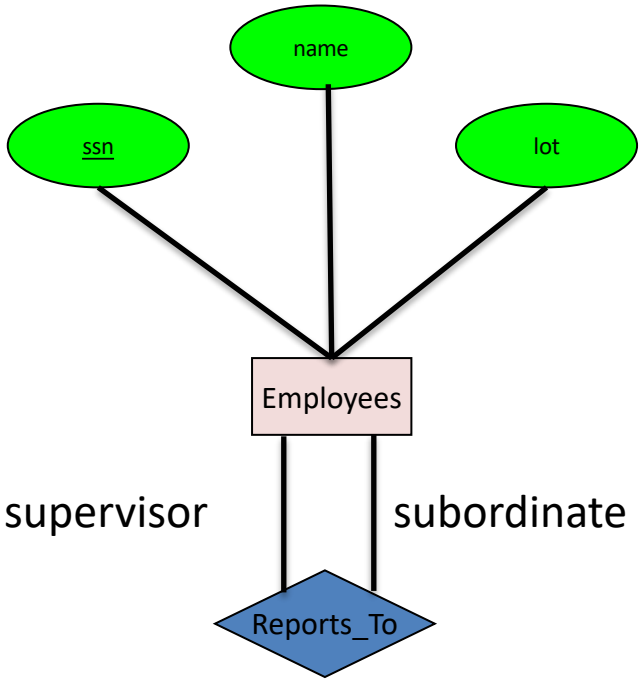
Relationship Sets (without Constraints) to Tables



Relationship Sets (without Constraints) to Tables

```
CREATE TABLE Works_in(     ssn char(11),
                           did integer (30),
address varchar(30),
since date,
PRIMARY KEY (ssn, did, address),
FOREIGN KEY (ssn) REFERENCES Employees,
FOREIGN KEY (address) REFERENCES Locations,
FOREIGN KEY (did) REFERENCES Departments,
)
```

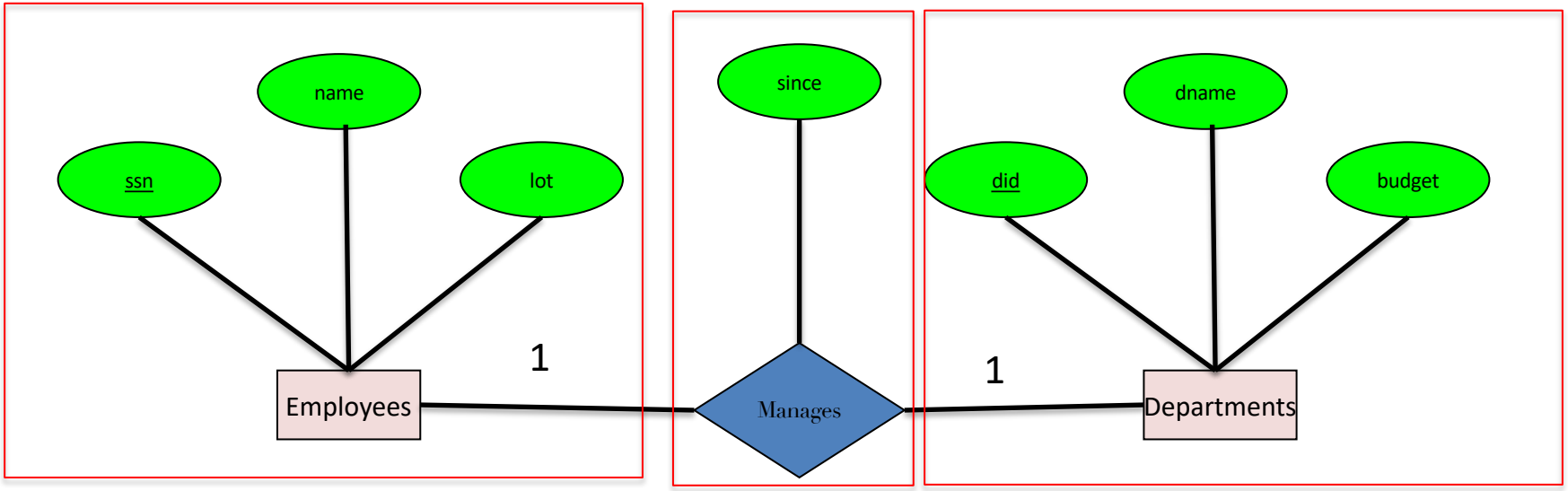
Relationship Sets (without Constraints) to Tables




```
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)
)
```

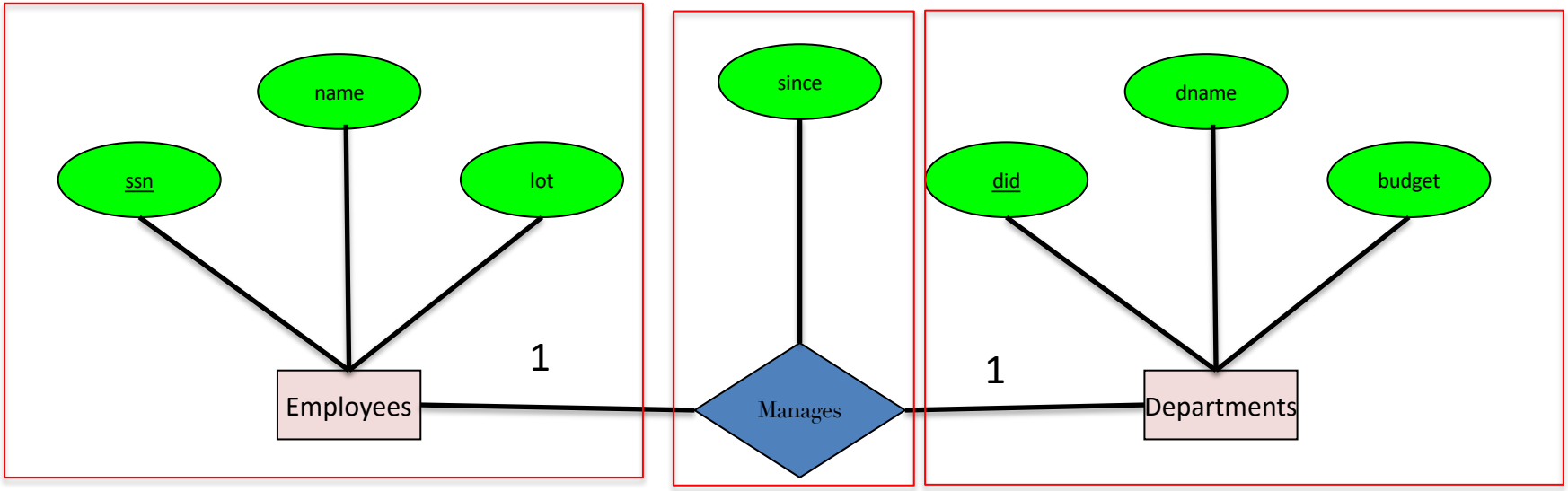
ONE-TO-ONE RELATIONSHIP

Relationship Sets to Tables (Option 1)



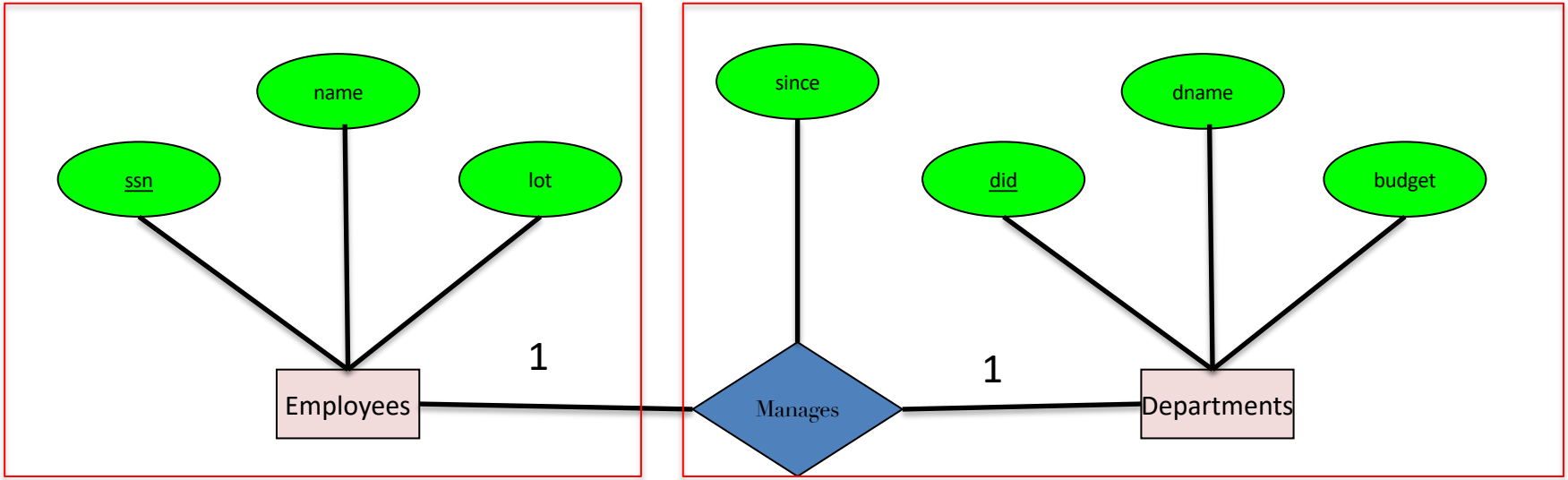

```
CREATE TABLE Manages(       ssn char(11),
                            did integer,
                            since date,
PRIMARY KEY (did),
FOREIGN KEY (ssn) REFERENCES Employees ON DELETE CASCADE,
FOREIGN KEY (did) REFERENCES Departments ON DELETE CASCADE,
)
```

Relationship Sets to Tables (Option 2)



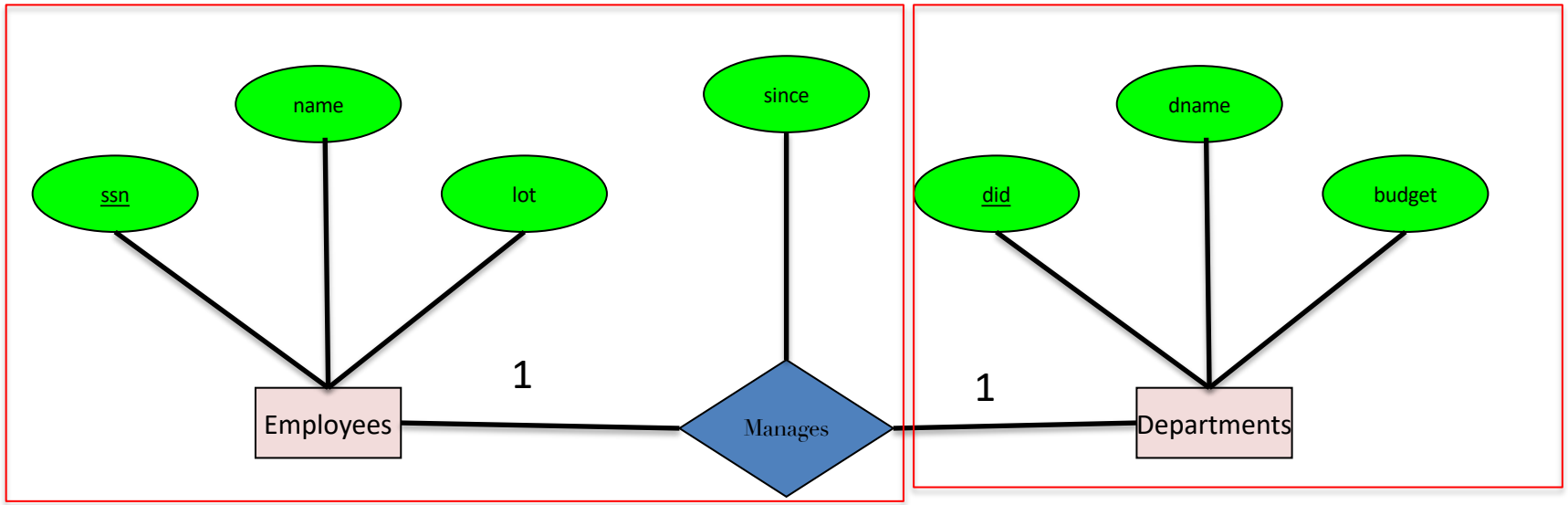

```
CREATE TABLE Manages(       ssn char(11),
                            did integer,
                            since date,
PRIMARY KEY (ssn),
FOREIGN KEY (ssn) REFERENCES Employees ON DELETE CASCADE,
FOREIGN KEY (did) REFERENCES Departments ON DELETE CASCADE,
)
```

Better way of doing it (Option 3)




```
CREATE TABLE Dept_Mgr(      did integer
                            dname varchar(30),
                            budget float(30),
                            ssn char(11),
                            since date,
PRIMARY KEY (did),
FOREIGN KEY (ssn) REFERENCES Employees ON DELETE SET NULL,
)
```

Possible but not recommended (Option 4)



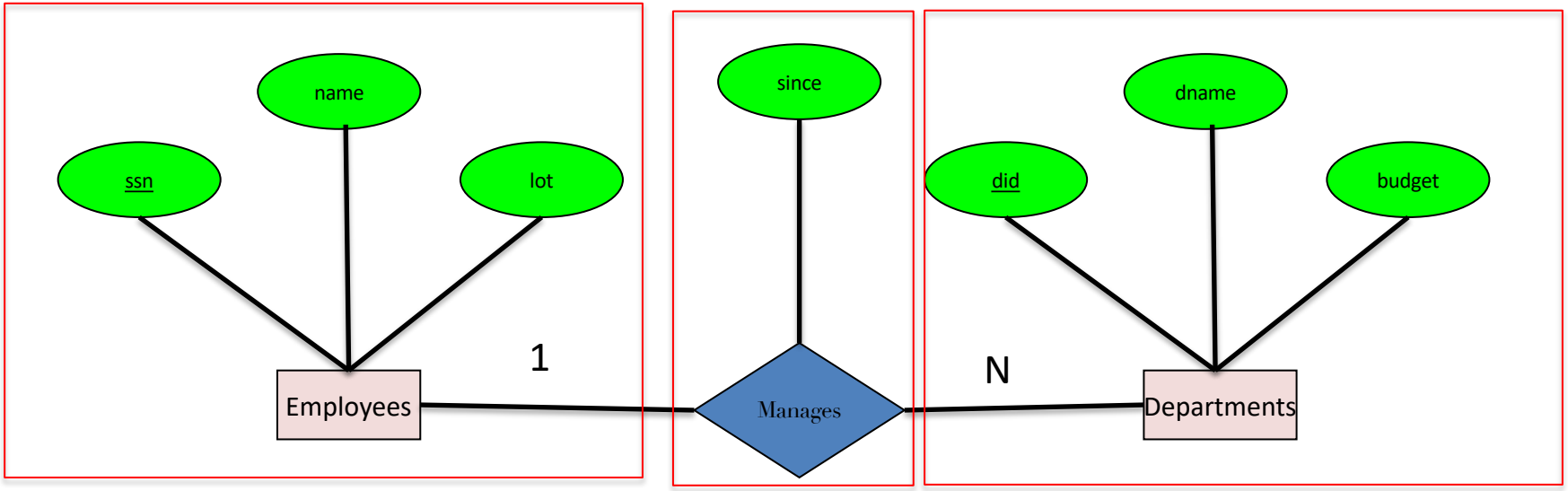

```
CREATE TABLE Employee_Mgr( ssn char(11),
                           name varchar(30),
                           lot integer,
                           since date,
                           did integer,

PRIMARY KEY (ssn),
FOREIGN KEY (did) REFERENCES Departments (did)
ON DELETE SET NULL,
)
```

Why Bad?

Many NULL Entries

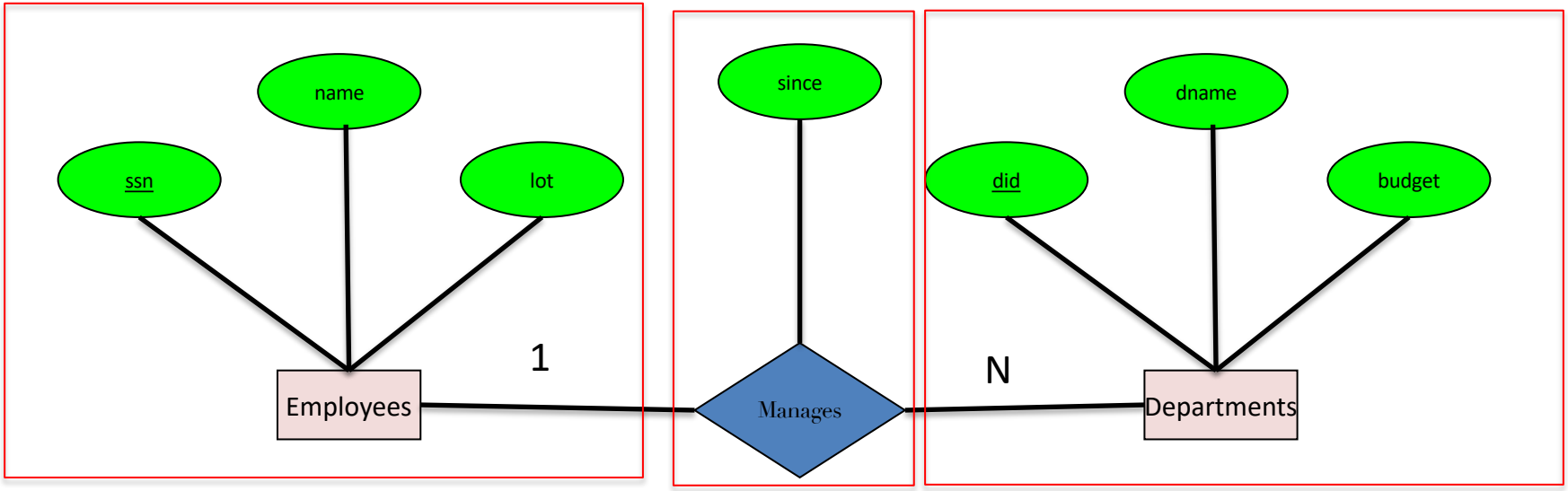
ONE-TO-MANY RELATIONSHIP

Relationship Sets to Tables (Option 1)




```
CREATE TABLE Manages(       ssn char(11),
                            did integer,
                            since date,
PRIMARY KEY (did),
FOREIGN KEY (ssn) REFERENCES Employees ON DELETE CASCADE,
FOREIGN KEY (did) REFERENCES Departments ON DELETE CASCADE,
)
```

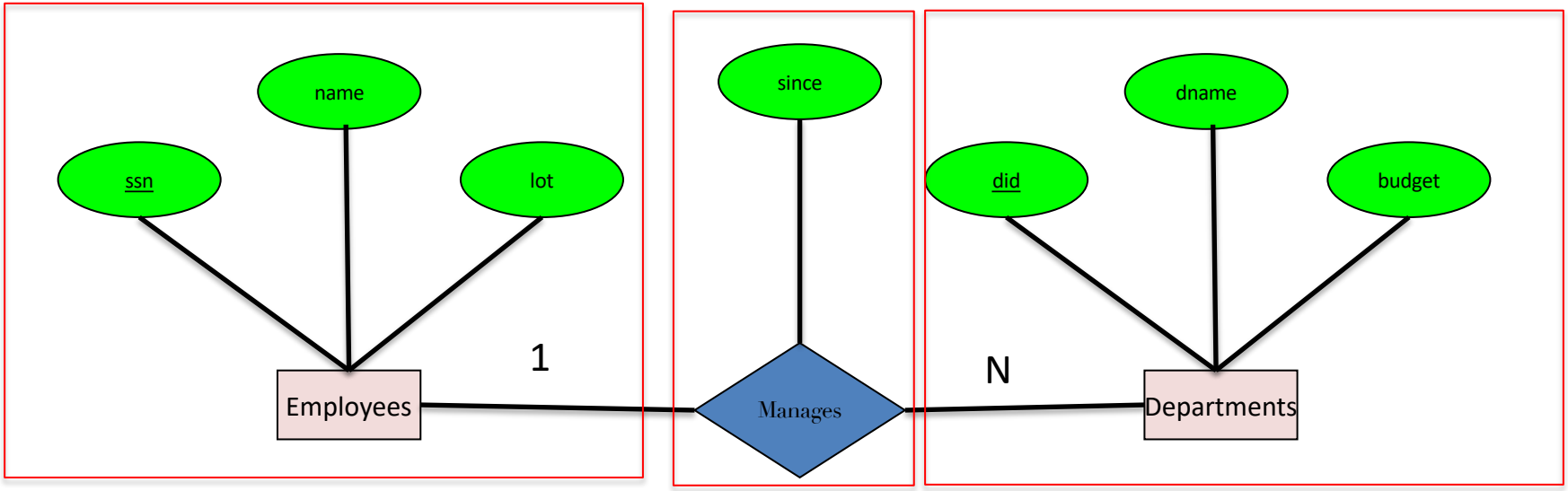
Relationship Sets to Tables (Option 2)




```
CREATE TABLE Manages(                    ar(11),
                                  did     ger,
                                  since   e,
PRIMARY KEY (ssn),
FOREIGN KEY (ssn) RE    RENCES    l    es ON DELETE CASCADE,
FOREIGN KEY (did) RE    NCES D    ents ON DELETE CASCADE,
)
```

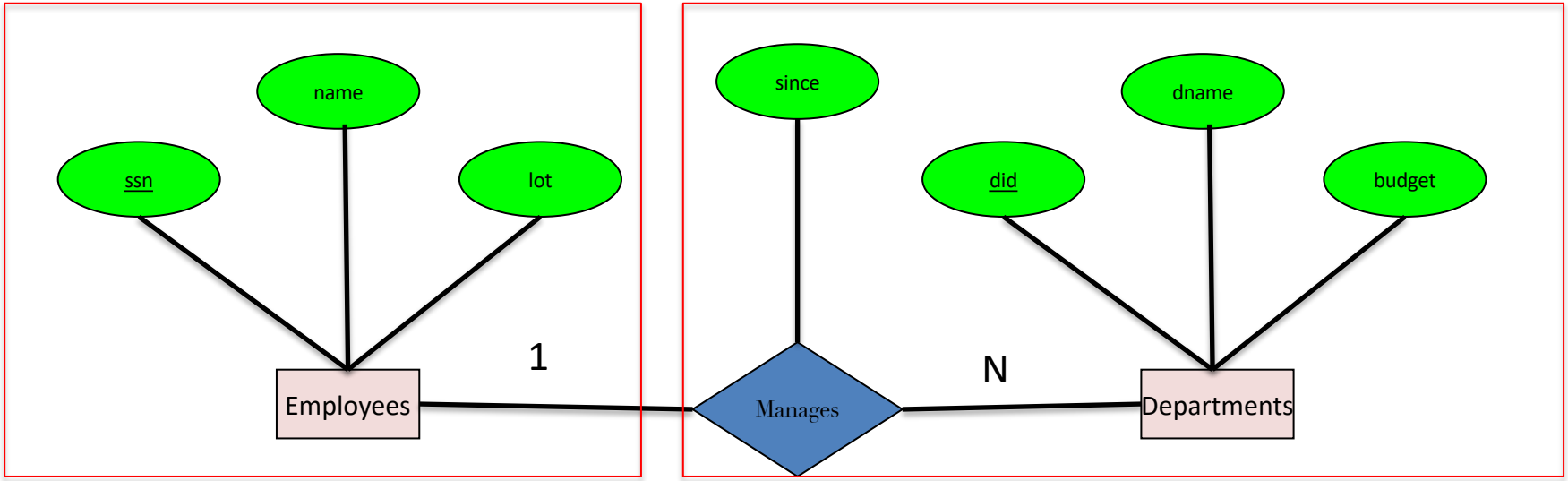

Wrong

Relationship Sets to Tables (Option 2 (2nd try))



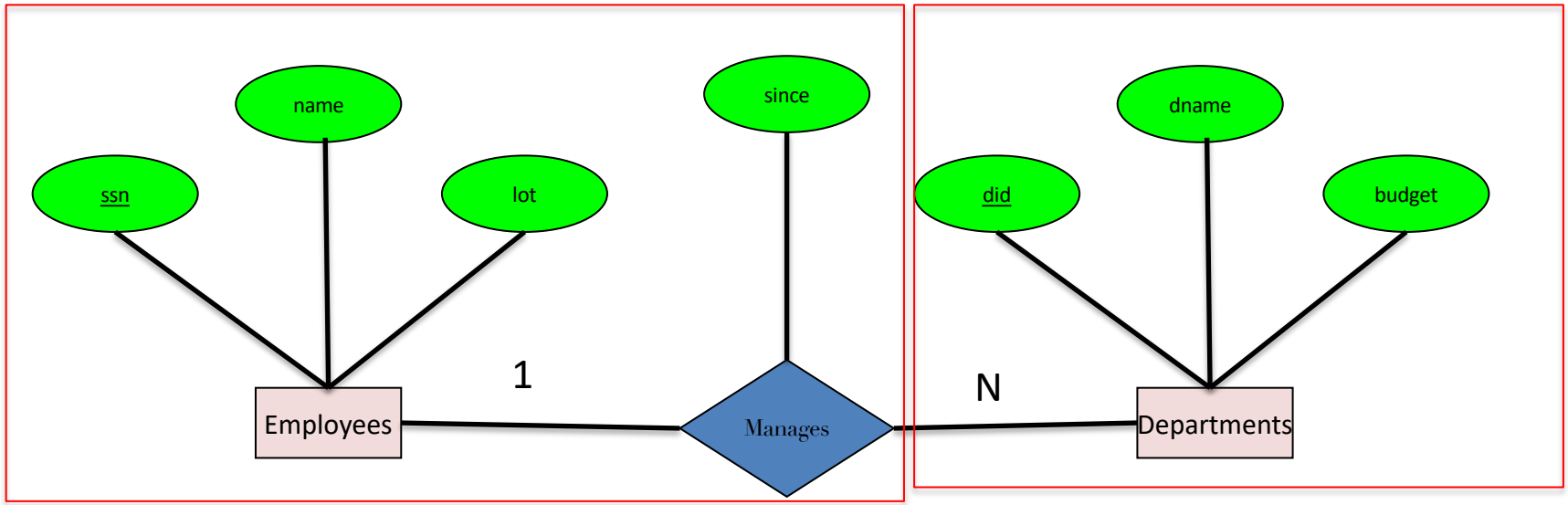

```
CREATE TABLE Manages(       ssn char(11),
                            did integer,
                            since date,
PRIMARY KEY (ssn, did),
FOREIGN KEY (ssn) REFERENCES Employees ON DELETE CASCADE,
FOREIGN KEY (did) REFERENCES Departments ON DELETE CASCADE,
)
```

Better way of doing it (Option 3)




```
CREATE TABLE Dept_Mgr(      did integer
                            dname varchar(30),
                            budget float(30),
                            ssn char(11),
                            since date,
PRIMARY KEY (did),
FOREIGN KEY (ssn) REFERENCES Employees ON DELETE SET NULL,
)
```

Not Possible (Option 4)



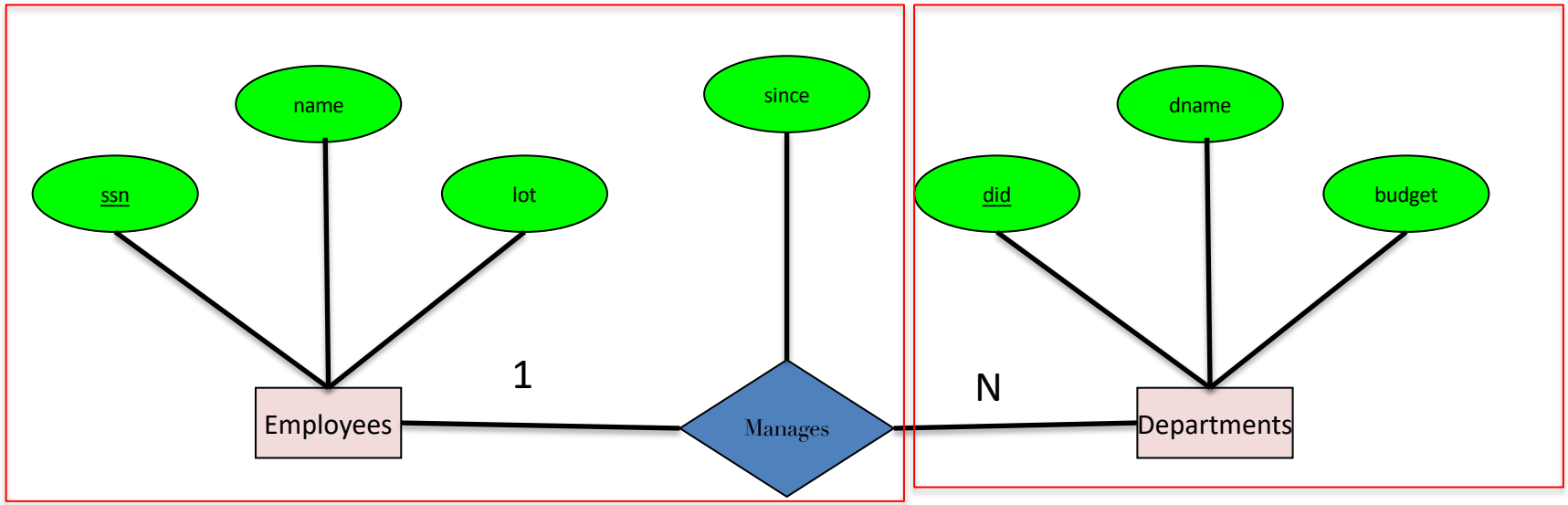

```
CREATE TABLE Employee_Mgr( ssn char(11),
                           archar(30),
                           lot  ger,
                           since  e,
                           int  er,

PRIMARY KEY (ssn),
FOREIGN KEY (did) REFE      rtments (did)
ON DELETE NO ACTION,
)
```

Why Not?

All relations are flat.

Wrong

What about this?

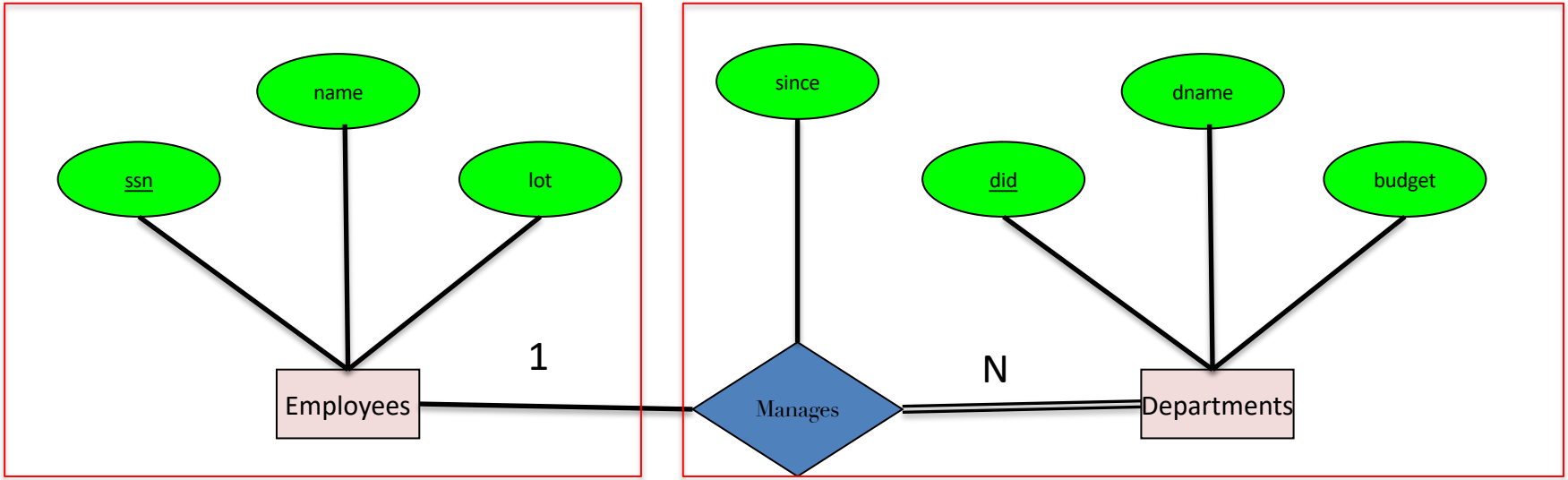



```
CREATE TABLE Employee_Mgr( ssn char(11),
                           name varchar(30),
                           lot integer,
                           since date,
                           did integer,

PRIMARY KEY (ssn, did),
FOREIGN KEY (did) REFERENCES Departments (did)
ON DELETE RESTRICT,
)
```

Possible. But…

Our Next topic…
It's a bad Functional Dependency

Should Department entity have Total participation?

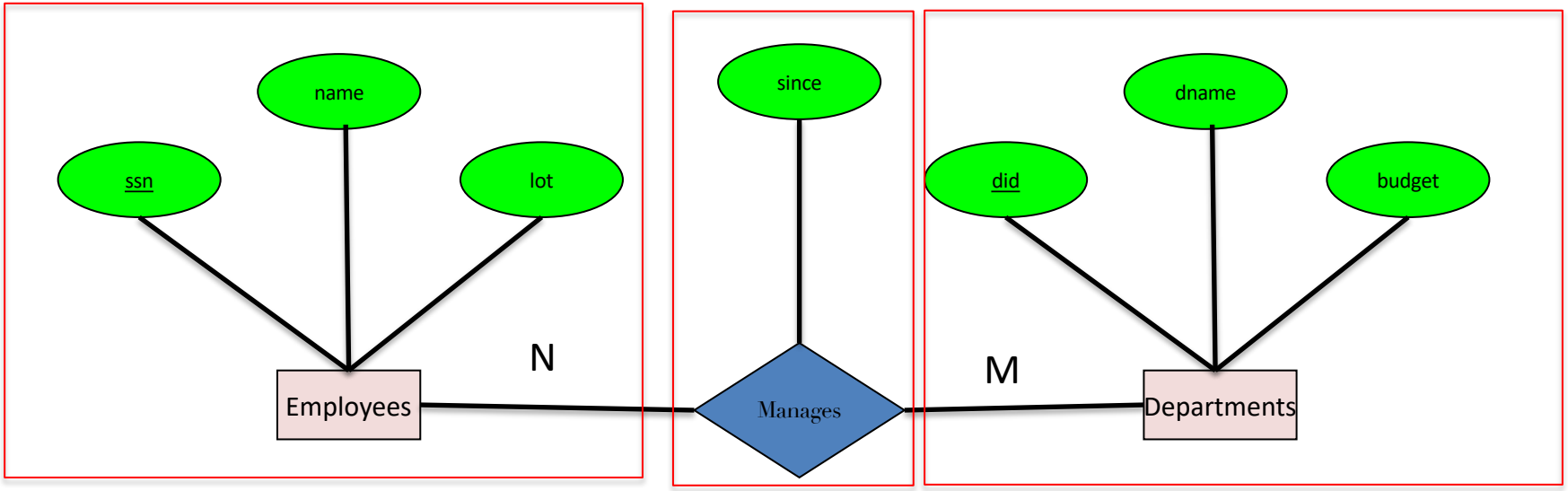



```
CREATE TABLE Dept_Mgr(      did integer
                            dname varchar(30),
                            budget float(30),
                            ssn char(11),
                            since date,
PRIMARY KEY (did),
FOREIGN KEY (ssn) REFERENCES Employees ON DELETE CASCADE,
)
```

No. We do not want to delete a department if the manager quits

MANY-TO-MANY RELATIONSHIP

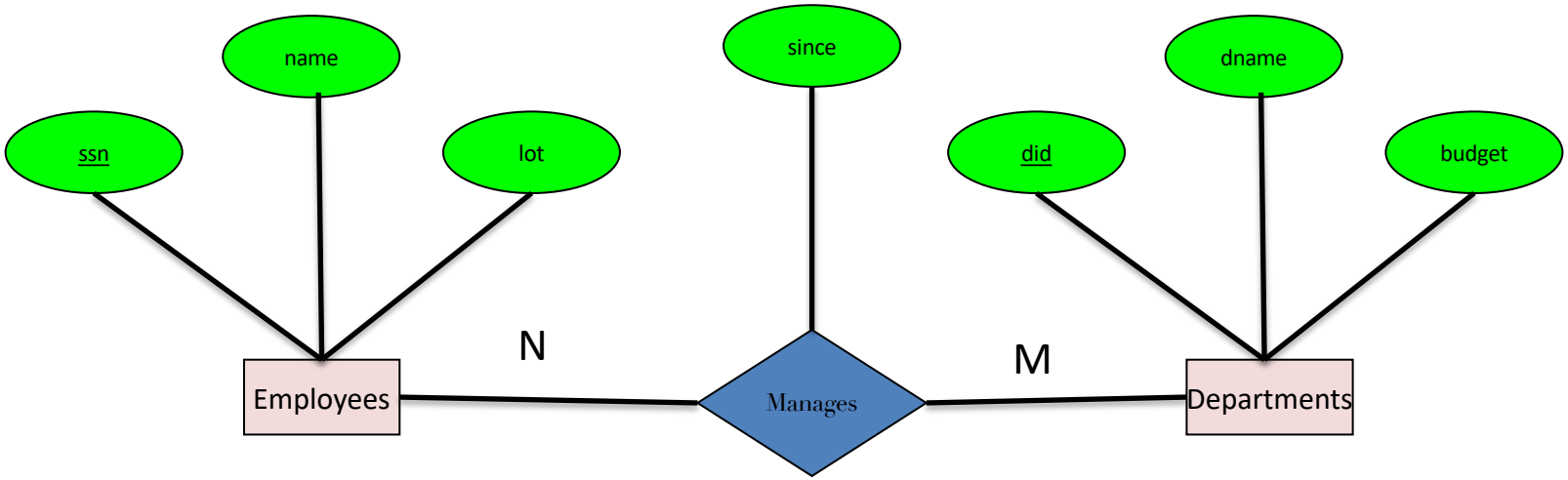
Relationship Sets to Tables (Only Option)



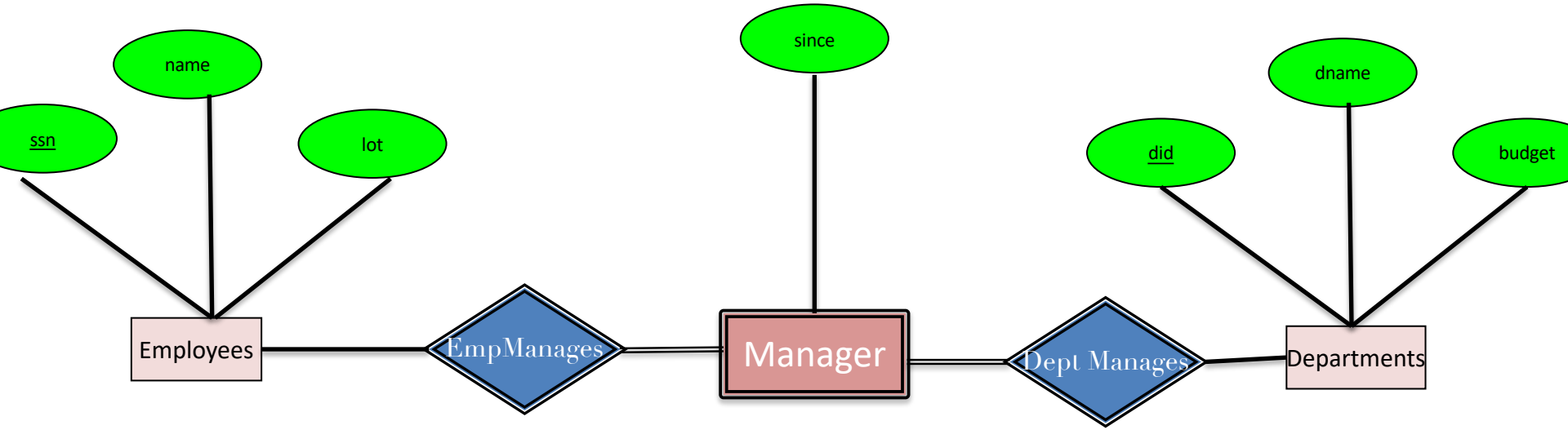

```
CREATE TABLE Manages(       ssn char(11),
                            did integer,
                            since date,
PRIMARY KEY (did, ssn),
FOREIGN KEY (ssn) REFERENCES Employees ON DELETE CASCADE,
FOREIGN KEY (did) REFERENCES Departments ON DELETE CASCADE,
)
```

MAPPING RELATIONSHIP INTO ENTITY SET

Version 1


name
since
dname
ssn
lot
did
budget
N
M
Employees
Manages
Departments

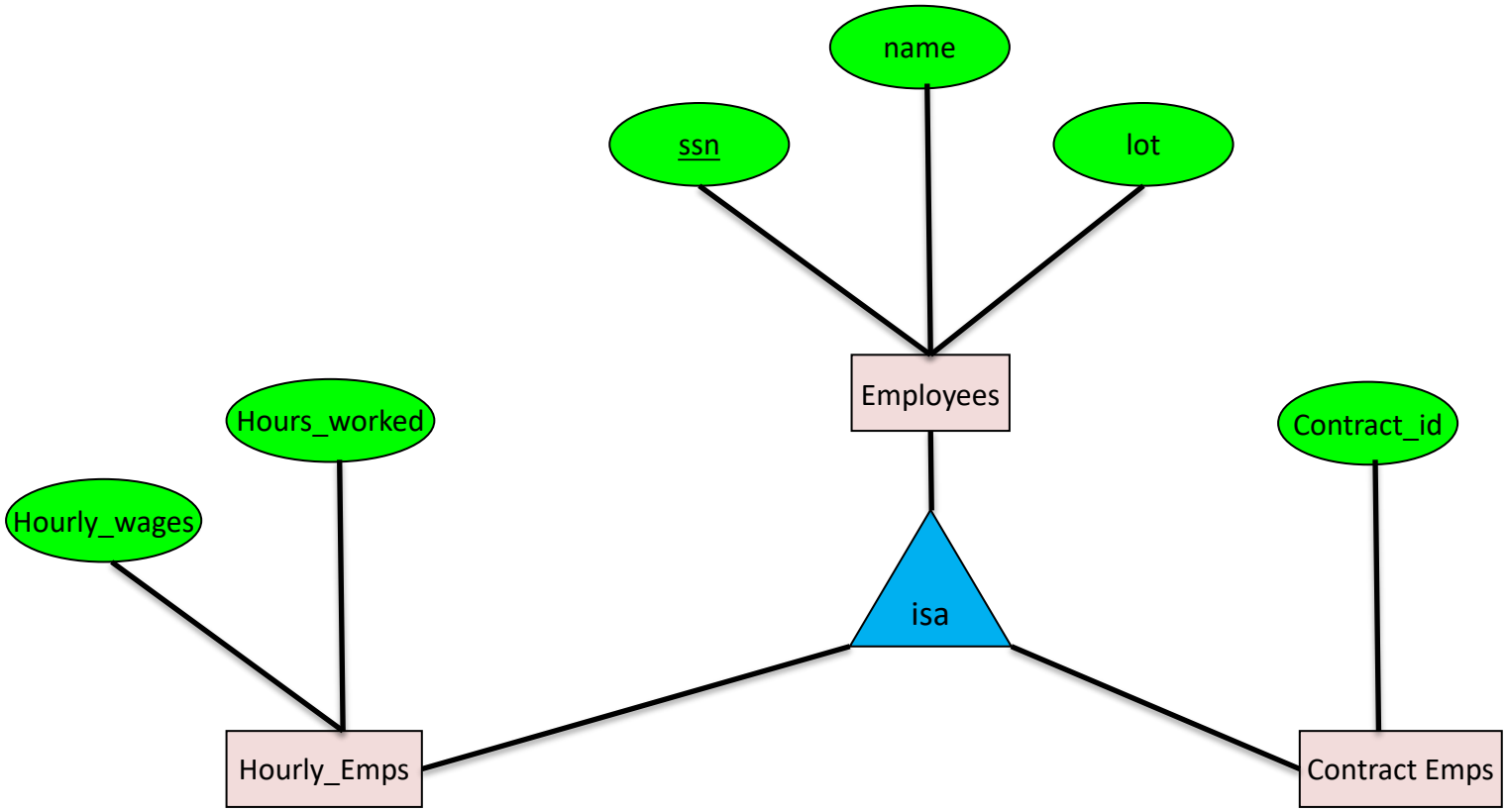
Version 2



If you ever face this, check if you could simplify

TRANSLATING CLASS HIERARCHIES

Translating Class Hierarchies



Two options

1. We can map each of the entity sets Employees, Hourly_Emps, and Contract_Emps to a distinct relation.
2. We can create just two relations, corresponding to Hourly_Emps and Contract_Emps

Both have their pros and cons

- Redundancy
- Performance

Notes on Project 1 Milestone 2

We did not cover Chapter 4 and you do not need to study it for exam or quizzes.

If you understand this isA concept and how to convert such an ER diagram into tables, that's fine.
And this is exactly what steps 8 and 9 are in Chapter 9

Hints for Project 2 Part 1

- `item.dat` contains nine fields (itemID, name, currently, buy_price, first_bid, started, ends, userID, and description)
- `user.dat` stores four fields (userID, rating, location, and country)
- `category.dat` stores only two fields (itemID, and category).
- `bid.dat` contains four fields (itemID, userID, time, and amount).

E/R Summary

- E/R diagrams are a visual syntax that allows technical and non-technical people to talk
  - For conceptual design
- Basic constructs: **entity**, **relationship**, and **attributes**
- A good design is faithful to the constraints of the application, but not overzealous
- The ER model to relation mapping is very important part of the database design.

Acknowledgement

- Some of the slides in this presentation are taken from the slides provided by the authors.

- Many of these slides are taken from cs145 course offered by Stanford University.