

ER to Relational Mapping

Davood Rafiei

Original material Copyright 2001-2018
(some material from textbooks and other instructors)



ER Model: Overview

- The “world” is described in terms of
 - Entities
 - Relationships
 - Attributes
- Constraints and Complications
 - Key constraints
 - Participation constraints
 - Set-valued attributes
 - Weak entities
 - ISA hierarchies



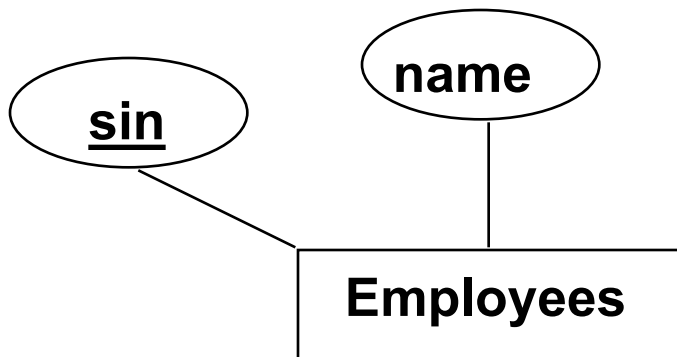
Relational Model: Overview

- Database: a set of relations (tables)
- For each table, we specify
 - Columns and the domain of each column,
 - Often the primary key and the foreign keys,
 - Other constraints (if any),
 - The way referential integrities must be enforced.



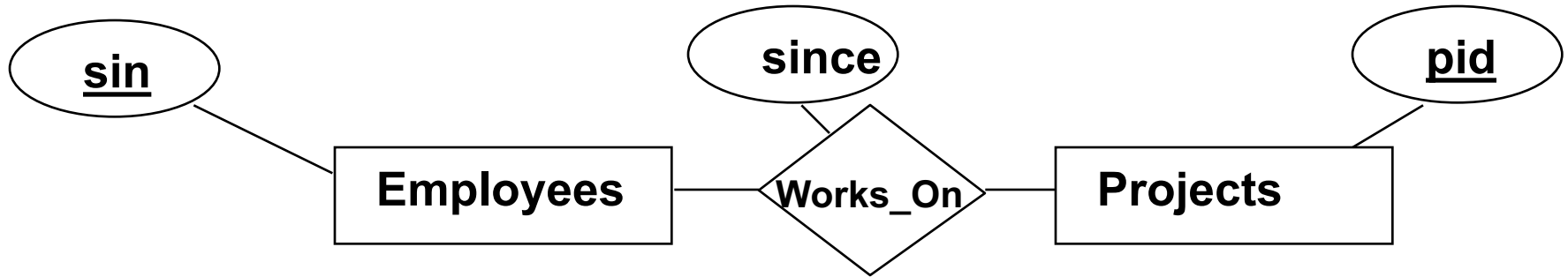
Entity Sets to Tables

- Entity sets to tables.



```
CREATE TABLE Employees  
  (sin CHAR(11),  
   name CHAR(20),  
   PRIMARY KEY (sin))
```

Relationship Sets to Tables



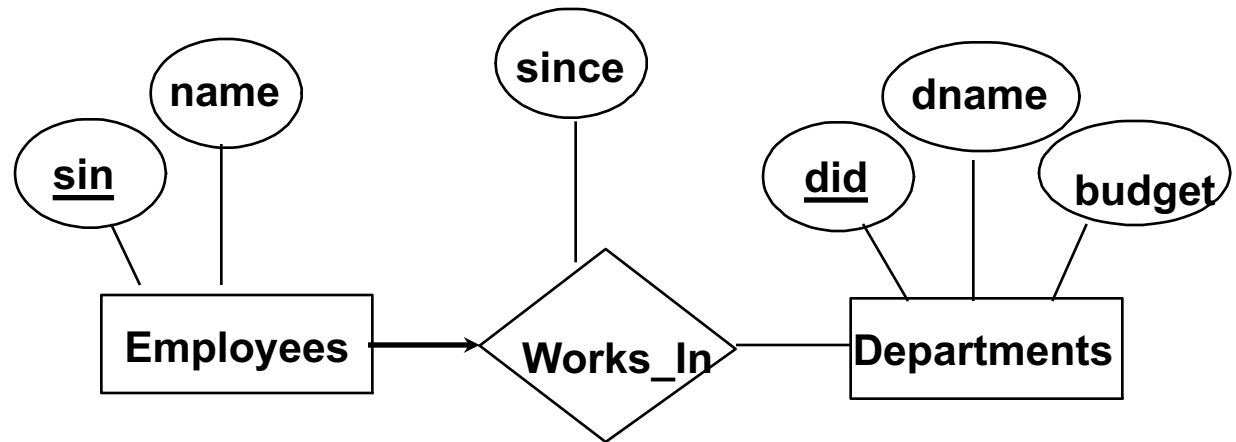
- **Constraint:** none.
- Attributes of the relation (table):
 - Key of every participating entity set (as foreign keys).
 - All descriptive attributes

```
CREATE TABLE Works_On(  
    sin CHAR(11),  
    pid INTEGER,  
    since DATE,  
    PRIMARY KEY (sin, pid),  
    FOREIGN KEY (sin)  
        REFERENCES Employees,  
    FOREIGN KEY (pid)  
        REFERENCES Projects)
```



Relationships with Key Constraints

- **Constraint:** each employee works in at most one department.



- Map the relationship to a table:
 - What is the key now?

```
CREATE TABLE Works_In (  
  sin CHAR(11),  
  did CHAR(3),  
  since DATE,  
  PRIMARY KEY (sin),  
  FOREIGN KEY (sin) REFERENCES Employees,  
  FOREIGN KEY (did) REFERENCES Departments)
```

Relationships with Key Constraints (Cont.)

■ Better mapping:

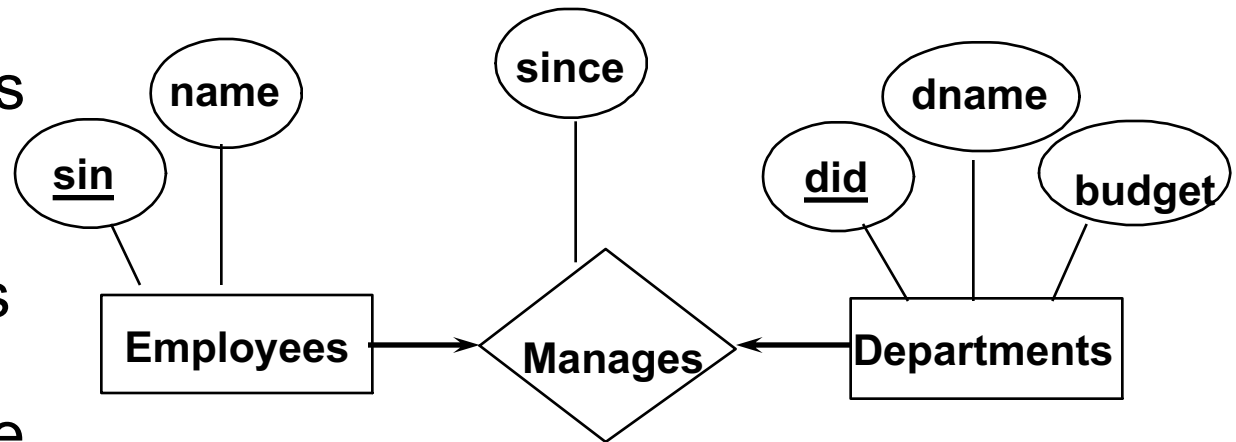
- Since each employee can work in at most one department, we could instead combine Works_In and Employees.
- Has one less table☺

```
CREATE TABLE Emp_Works(  
  sin CHAR(11),  
  name CHAR(20),  
  did CHAR(3),  
  since DATE,  
  PRIMARY KEY (sin),  
  FOREIGN KEY (did) REFERENCES Departments)
```



Relationships with Key Constraints (Cont.)

- **Constraint:** each employee manages at most one department and each department is managed by at most one employee.

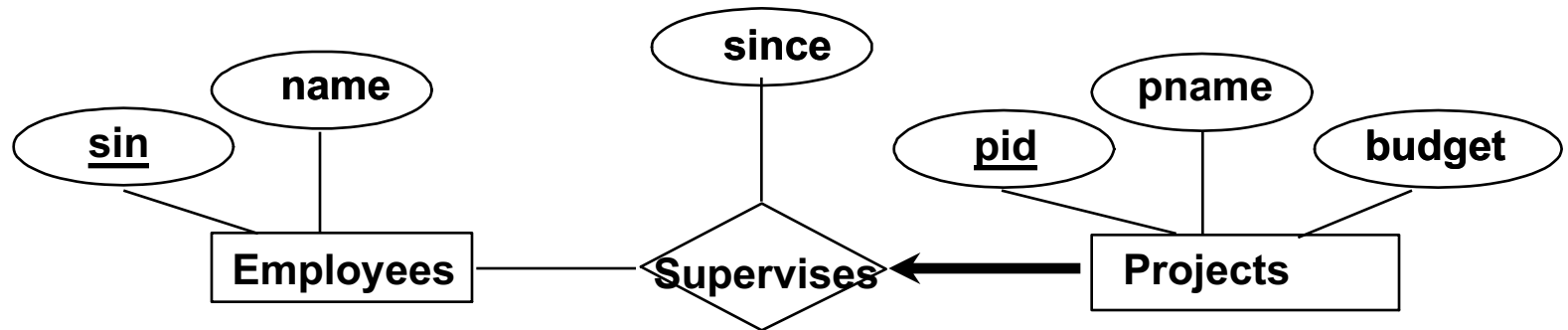


- We can combine **Manages** with **Departments**.
- We can also combine **Manages** with **Employees**.

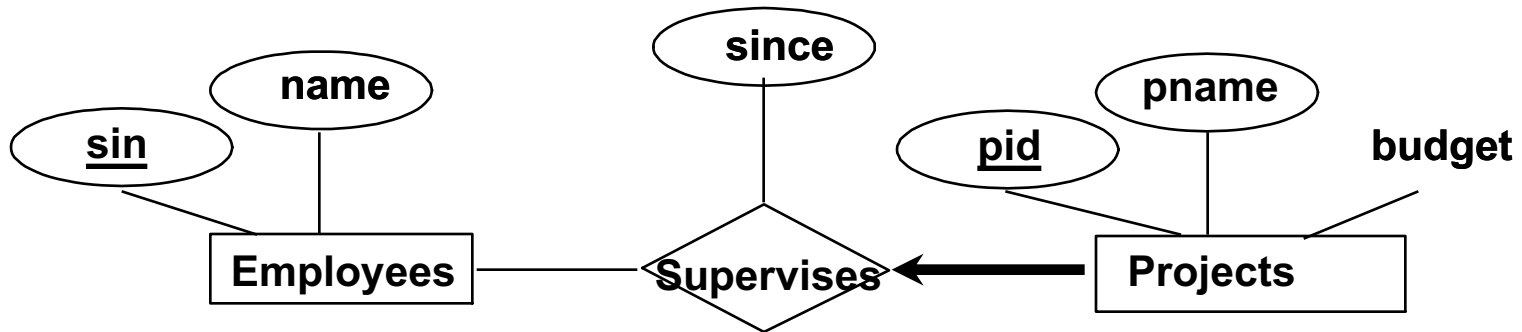
```
CREATE TABLE Dept(  
  did CHAR(3),  
  dname CHAR(20),  
  budget INTEGER,  
  mgr CHAR(11),  
  since DATE,  
  PRIMARY KEY (did),  
  FOREIGN KEY (mgr) REFERENCES Employees)
```


Review: Participation Constraints

- Does every project have a supervisor?
 - If so, this is a participation constraint: the participation of Projects in Supervises is said to be *total* (vs. *partial*).
 - ✓ Every *pid* value in Projects table must appear in a row of the Supervises table (with a non-null *sin* value!)



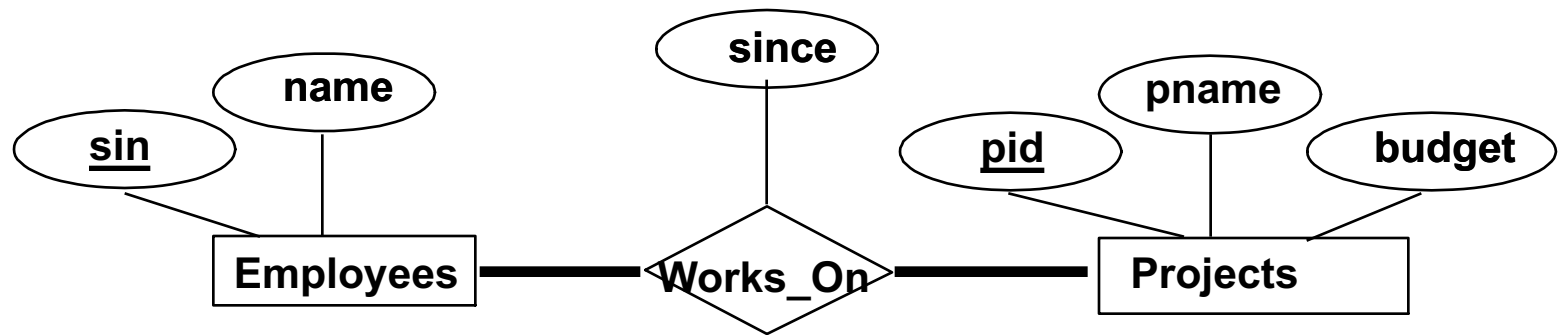
Participation Constraints



```
CREATE TABLE Proj_Supervises (  
  pid INTEGER,  
  pname CHAR(20),  
  budget REAL,  
  sin CHAR(11) NOT NULL,  
  since DATE,  
  PRIMARY KEY (pid),  
  FOREIGN KEY (sin) REFERENCES Employees  
  ON DELETE NO ACTION)
```

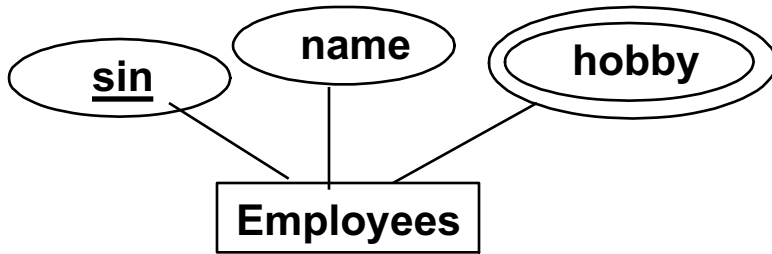
Participation Constraints (Cont.)

- How can we map Works_On relationship to a table and still keep the participation constraints?



- Can't without resorting to CHECK constraints (will be discussed later).

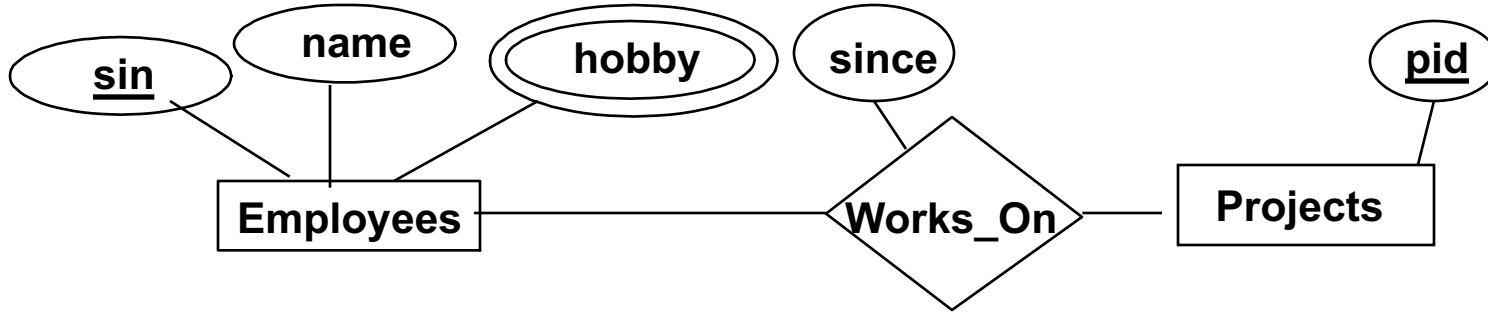
Set-Valued Attributes



```
CREATE TABLE Employees (  
    sin CHAR(11),  
    name CHAR(20),  
    hobby char(15),  
    PRIMARY KEY (sin, hobby) )
```

- Cannot store more than one value in a field!
- What is the key of the relation?
 - sin cannot be a key!
- The same problem arises in mapping a relationship with a set-valued attribute.

Set-Valued Attributes (Cont)



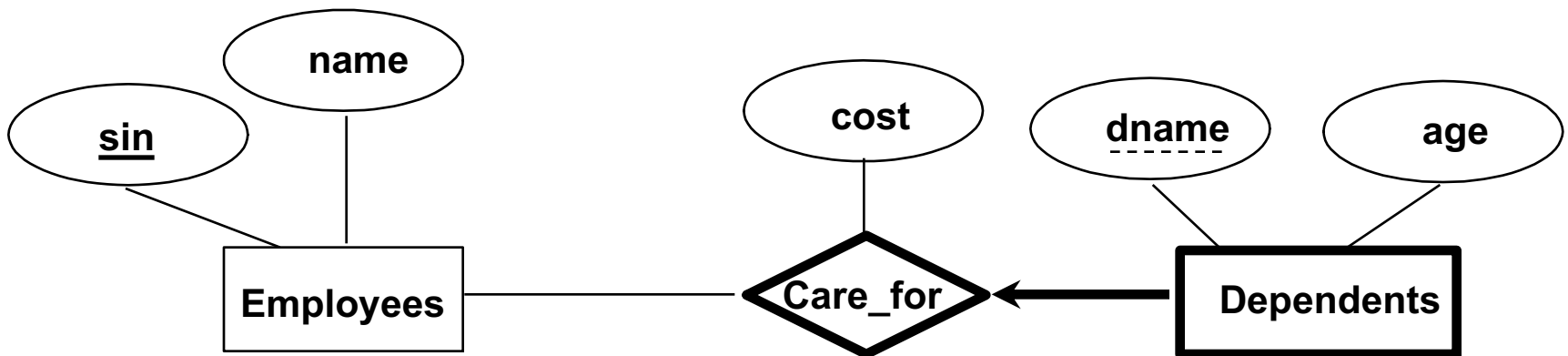
- Can sin reference employees any more?
- No. We cannot define sin as a foreign key any longer.

```
CREATE TABLE Works_On(  
  sin CHAR(11),  
  pid INTEGER,  
  since DATE,  
  PRIMARY KEY (sin, pid),  
  FOREIGN KEY (pid)  
    REFERENCES Projects,  
  FOREIGN KEY (sin)  
  REFERENCES Employees  
)
```



Review: Weak Entities

- A *weak entity* can be identified uniquely only by considering the primary key of another (*owner*) entity.
 - Owner entity set and weak entity set must participate in a one-to-many relationship set (1 owner, many weak entities).
 - Weak entity set must have total participation in this *identifying* relationship set.



Translating Weak Entity Sets

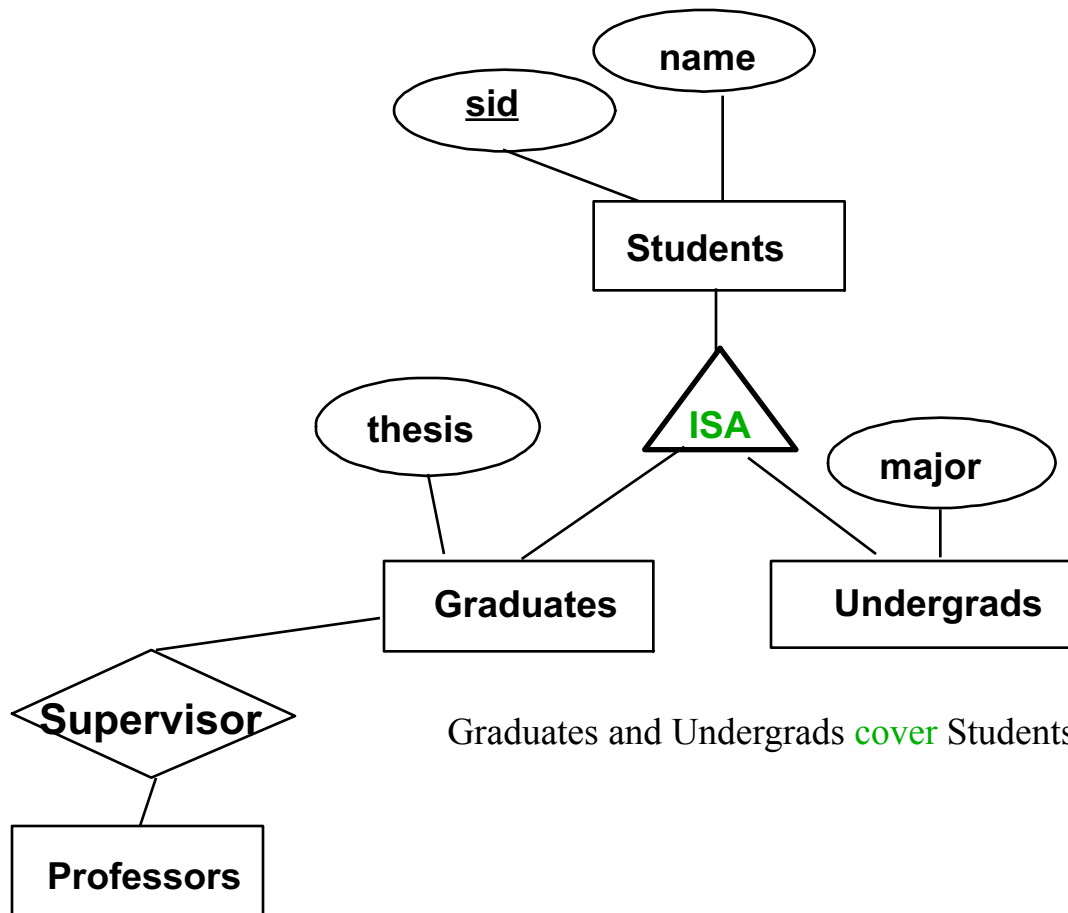
- Weak entity set and identifying relationship set are translated into a single table.

```
CREATE TABLE Dep_Care (  
    dname CHAR(20),  
    age INTEGER,  
    cost REAL,  
    sin CHAR(11) NOT NULL,  
    PRIMARY KEY (dname, sin),  
    FOREIGN KEY (sin) REFERENCES Employees  
    ON DELETE CASCADE)
```

- When the owner entity is deleted, all owned weak entities must also be deleted.



Review: ISA Hierarchies



Translating ISA Hierarchies to Relations

- **General approach:** 3 relations

- ✓ *Students(sid, name)*
- ✓ *Graduates(sid, thesis)*
- ✓ *Undergrads(sid, major)*

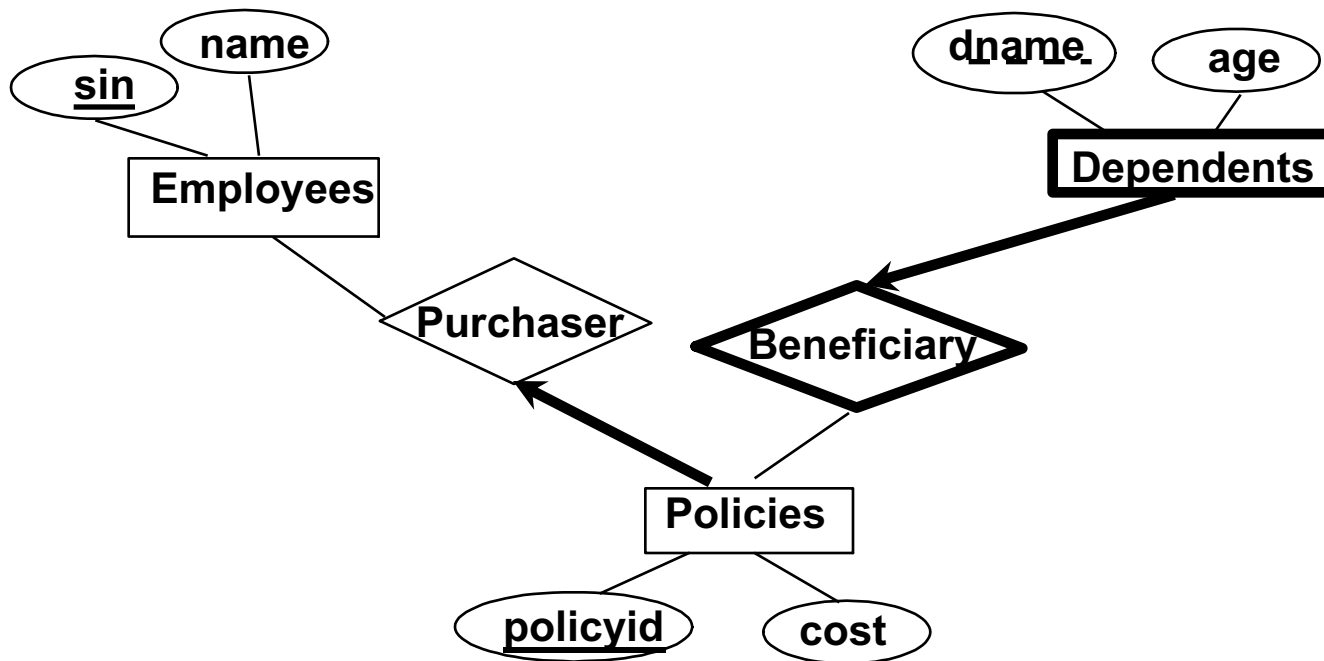
```
CREATE TABLE Undergrads (  
    sid CHAR(8) NOT NULL,  
    major CHAR(12),  
    PRIMARY KEY (sid),  
    FOREIGN KEY (sid) REFERENCES Students  
    ON DELETE CASCADE)
```

- **Alternative: Graduates and Undergrads**

- If Graduates and Undergrads Cover Students.
 - ✓ *Graduates(sid, name, thesis).*
 - ✓ *Undergrads(sid, name, major)*



Exercise: Map to Relations



Exercise: Answer

- The key constraints allow us to combine Purchaser with Policies and Beneficiary with Dependents.

```
CREATE TABLE Policies (  
    policyid INTEGER,  
    cost REAL,  
    sin CHAR(11) NOT NULL,  
    PRIMARY KEY (policyid),  
    FOREIGN KEY (sin) REFERENCES Employees  
        ON DELETE CASCADE)
```

- Participation constraints lead to NOT NULL constraints.

```
CREATE TABLE Dependents (  
    dname CHAR(20),  
    age INTEGER,  
    policyid INTEGER NOT NULL,  
    PRIMARY KEY (dname, policyid),  
    FOREIGN KEY (policyid) REFERENCES Policies  
        ON DELETE CASCADE)
```

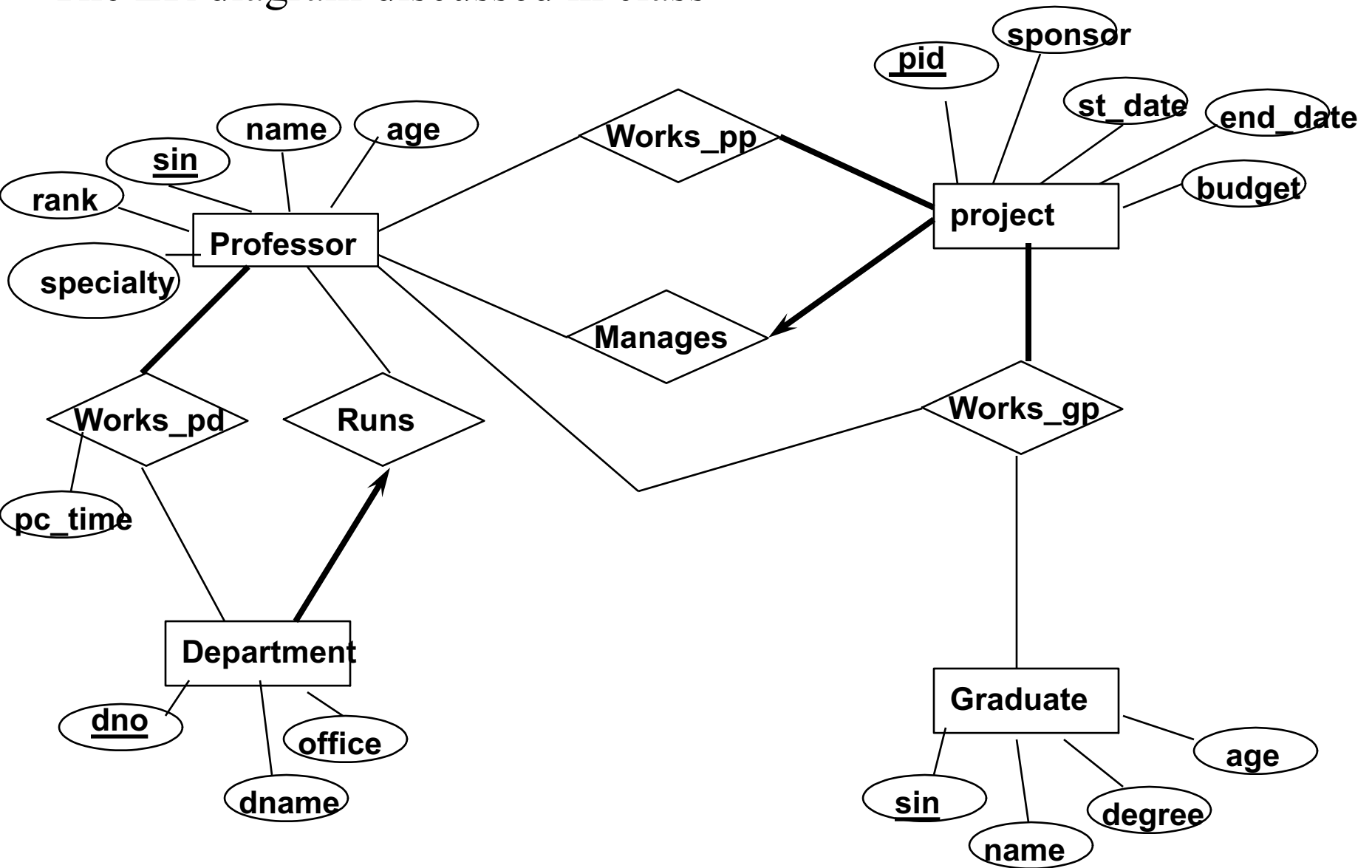


Relational Model: Summary

- A tabular representation of data.
- Simple and intuitive, currently the most widely used.
- Integrity constraints can be specified by the DBA, based on application semantics. DBMS checks for violations.
 - Two important ICs: primary and foreign keys
 - In addition, we *always* have domain constraints.
- Powerful and natural query languages exist.
- Rules to translate ER to relational model



The ER diagram discussed in class



Map it into relations...

