CS2102 Database Systems

Slides adapted from Prof. Chan Chee Yong

LECTURE 04

ENTITY RELATIONSHIP DATA MODEL

PREVIOUS LECTURE

Null values

Three-valued logic system: TRUE, FALSE, UNKNOWN

x	*y*	*x AND y*	*x OR y*	*NOT x*
FALSE FALSE FALSE	FALSE UNKNOWN TRUE	FALSE FALSE FALSE	FALSE UNKNOWN TRUE	TRUE
UNKNOWN UNKNOWN UNKNOWN	FALSE UNKNOWN TRUE	FALSE UNKNOWN UNKNOWN	UNKNOWN UNKNOWN TRUE	UNKNOWN
TRUE TRUE TRUE	FALSE UNKNOWN TRUE	FALSE UNKNOWN TRUE	TRUE TRUE TRUE	FALSE

x	*x IS NULL*	*x IS NOT NULL*
null	TRUE	FALSE
non-null	FALSE	TRUE

x	*y*	*x IS DISTINCT FROM y*
null	null	FALSE
null	non-null	TRUE
non-null	null	TRUE
non-null	non-null	x <> y

PREVIOUS LECTURE

Data definition language (DDL)

Create table

```
CREATE TABLE [ IF NOT EXISTS ] table_name ( [
  { column_name data_type
      [ column_constraints [ ... ] ]
    | table_constraints }
  [, ...]
] );
```

Drop table syntax

```
DROP TABLE [ IF EXISTS ] table_name
```

PREVIOUS LECTURE

Data definition language (DDL)

Alter table

- Add/remove/modify columns
  - `ALTER TABLE students ALTER COLUMN dept DROP DEFAULT;`
  - `ALTER TABLE students DROP COLUMN dept;`
  - `ALTER TABLE students ADD COLUMN faculty varchar(20);`
  - etc
- Add/remove constraints
- etc

PREVIOUS LECTURE

Data manipulation language (DML)

Insert into syntax

```
INSERT INTO table_name
[ ( column_name [, ...] ) ]
VALUES ( { expression | DEFAULT } [, ...] );
```

Delete from syntax

```
DELETE FROM table_name
[ WHERE condition ];
```

Update syntax

```
UPDATE table_name
SET column_name = { expression | DEFAULT }
[ WHERE condition ];
```

PREVIOUS LECTURE

Simple queries

Basic syntax

- Basic form of SQL query consists of *three clauses*

```
SELECT [ DISTINCT ]  select_list  -- select clause
FROM                 from_list    -- from clause
[ WHERE              condition ]  -- where clause
```

- `select_list` specifies columns to be included in output
- `from_list` specifies list of relations
- `condition` specifies conditions on relations

- **Output:** relation generated from `from_list` containing attributes based on `select_list` that satisfies `condition`
  - Output relation could contain duplicate record if `DISTINCT` is not used in the `SELECT` clause

- Entity Relationship Diagram
  - ER model
  - Relationship constraints
  - Participation constraints
  - Weak entity sets
- ER to SQL
  - ER diagram to SQL
  - Additional ER concepts
  - ER design and relational mapping

Overview

- Entity Relationship Diagram
  - ER model
  - Relationship constraints
  - Participation constraints
  - Weak entity sets
- ER to SQL
  - ER diagram to SQL
  - Additional ER concepts
  - ER design and relational mapping

Entity Relationship Diagram

SELF-READ

Conceptual data models

Introduction

- **Entity-relationship (ER) model**
  - Developed by Peter Chen in 1976
  - Designed for conceptual data model specifications
  - Most common data model used for database design
- **Unified modelling language (UML)**
  - Developed by Grady Booch & James Rumbaugh in 1997
  - Goes beyond conceptual data modelling
    - Software design specifications
  - Standardized by Object Management Group (OMG)

Database design process

Steps

1. **Requirement analysis**
   - Find out the data/application/performance requirement of the enterprise
2. **Conceptual database design**
   - Capture data requirements using a conceptual schema
3. **Logical database design**
   - Map conceptual schema to logical schema supported by DBMS
4. **Schema refinement**
   - Improve logical schema design using data constraints
5. **Physical database design**
   - Use performance requirements to design physical schema
6. **Application & security design**
   - Specify access control policies

Database design process

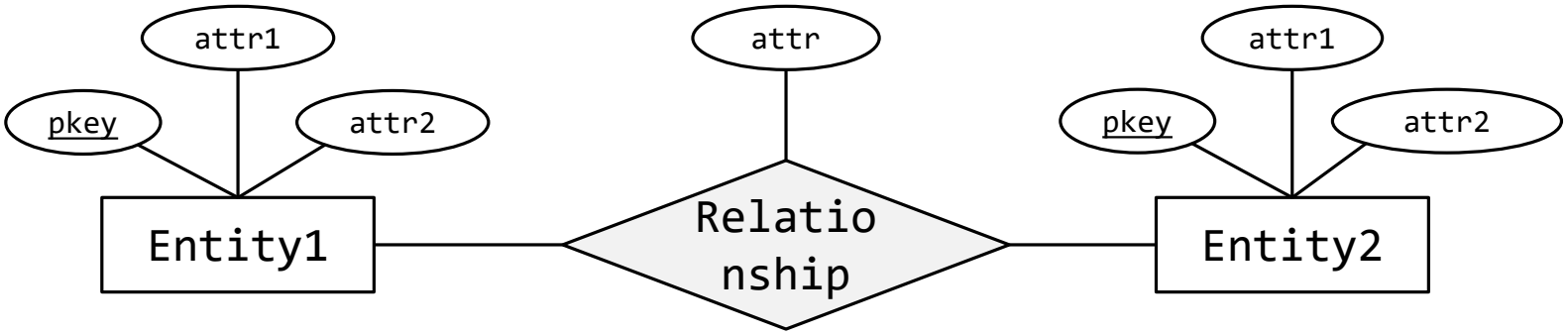
Requirement analysis

I would like my customers to be able to browse my catalog of books and place orders over the Internet. Currently, I take orders over the phone. I have mostly corporate customers who call me and give me the ISBN number of a book and a quantity; they often pay by credit card. If I don't have enough copies in stock, I order additional copies and delay the shipment until the new copies arrive; I want to ship a customer's entire order together. My catalog includes all the books I sell. For each book, the catalog contains its ISBN number, title, author, purchase price, sales price, and the year the book was published. Most of my customers are regulars, and I have records with their names and addresses. New customers have to call me first and establish an account before they can use my website. On my new website, customers should first identify themselves by their unique customer identification number. Then they should be able to browse my catalog and to place orders online.

ER model

Introduction

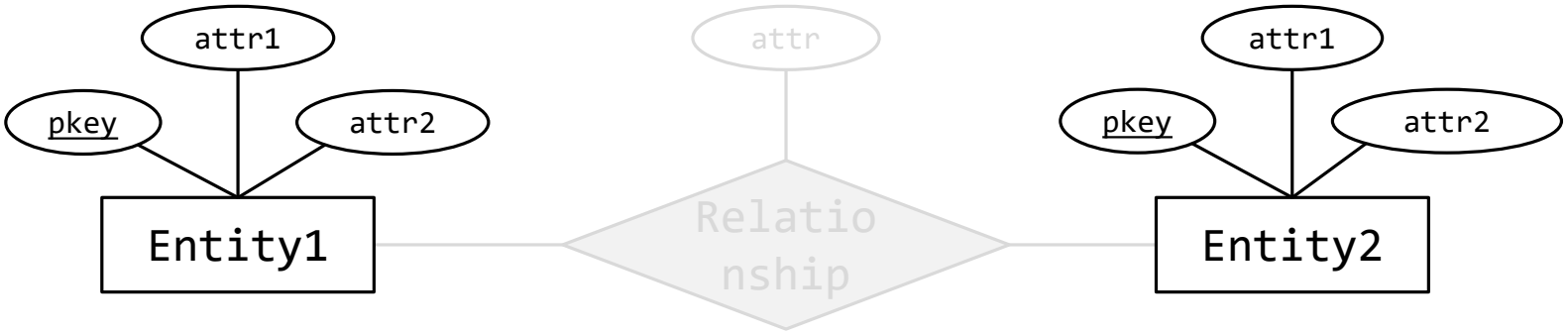
- Data is described in terms of entities and their relationships
- Information about entities and relationships are described using attributes
- Certain data constraints are represented using additional annotations
- ER schemas are presented as ER diagrams



ER model

Basic

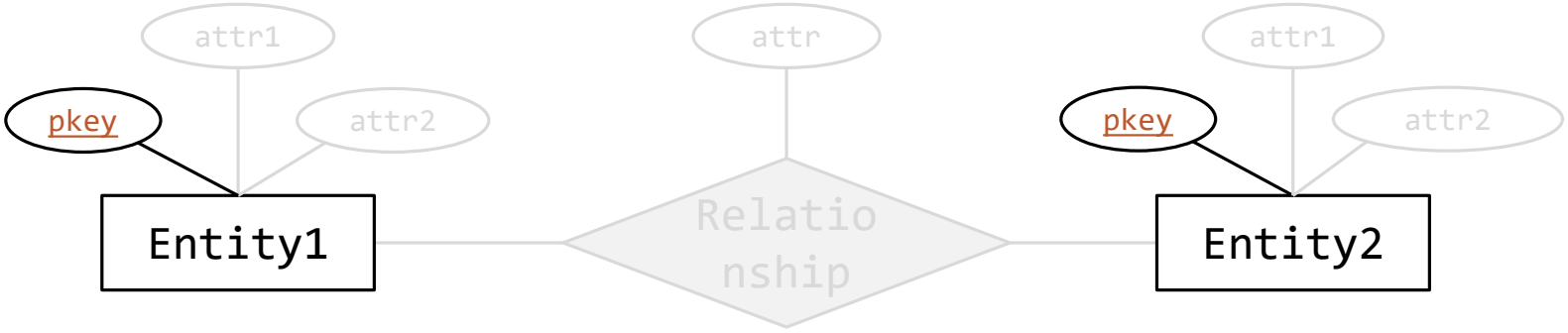
- **Entity** *Real-world object distinguishable from other objects*
- **Attribute** *Specific information describing an entity*
  each attribute has an atomic domain (e.g, integer, string)
  - *Represented by ovals*
- **Entity set** *A collection of similar entities*
  - *Represented by rectangles*



ER model

Keys

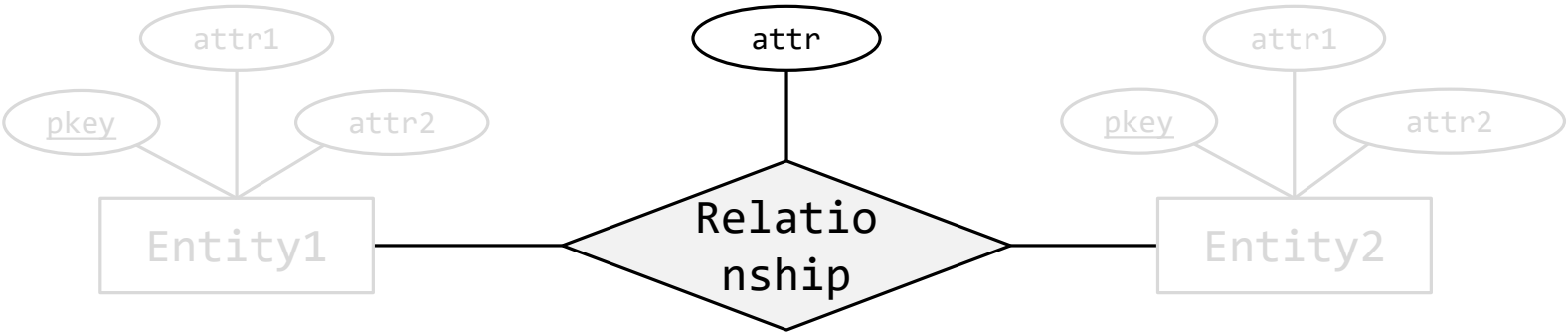
- Each entity set has a key (*i.e., minimal set of attributes whose values uniquely identify an entity*)
- An entity set could have multiple keys called candidate keys
- One of the candidate keys is chosen as the primary key
- The attributes that formed a primary key are underlined
- Example: `sid` in `Students` and `cid` in `Courses`



ER model

Relationships

- **Relationship** — *An association among two or more entities*
  attributes are used to describe information about relationships
- **Relationship set** — *A collection of similar relationships*
  - *Represented by diamonds*



ER model

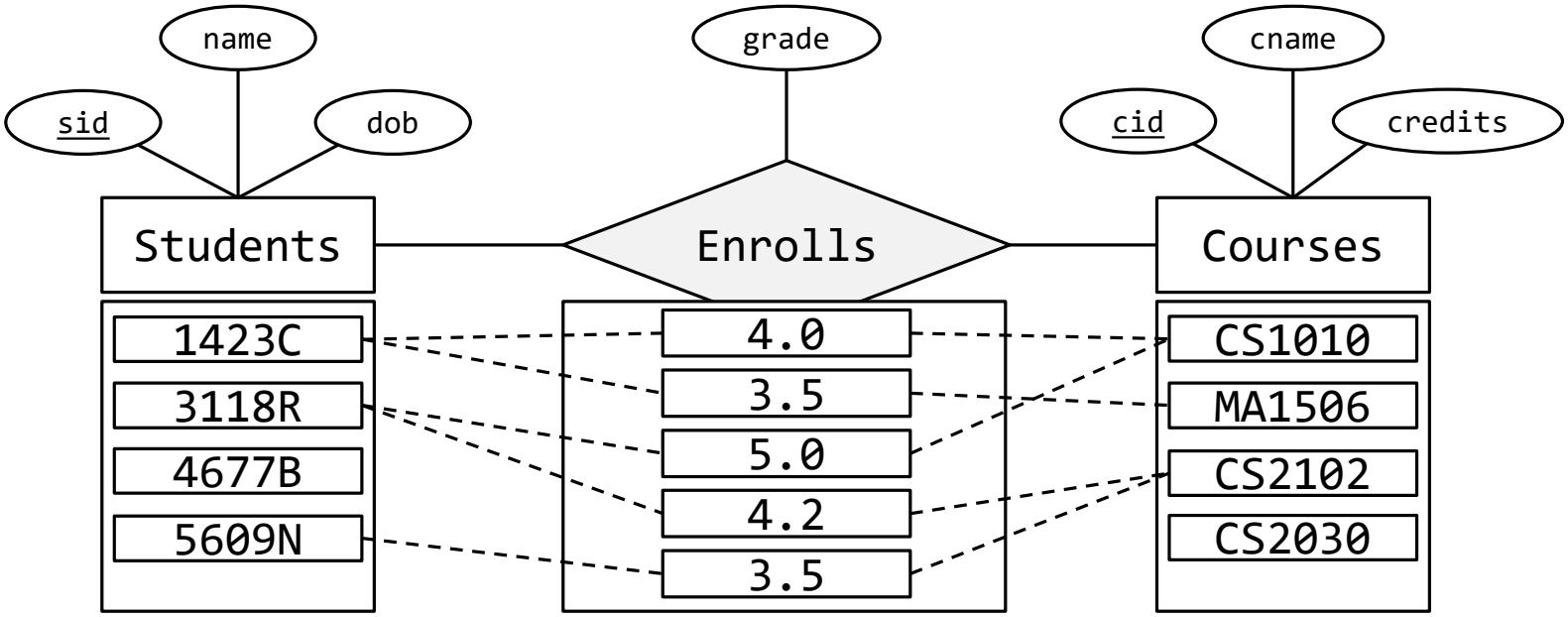
Entities, relationships, and attributes

I would like my customers to be able to browse my catalog of books and place orders over the Internet. Currently, I take orders over the phone. I have mostly corporate customers who call me and give me the ISBN number of a book and a quantity; they often pay by credit card. If I don't have enough copies in stock, I order additional copies and delay the shipment until the new copies arrive; I want to ship a customer's entire order together. My catalog includes all the books I sell. For each book, the catalog contains its ISBN number, title, author, purchase price, sales price, and the year the book was published. Most of my customers are regulars, and I have records with their names and addresses. New customers have to call me first and establish an account before they can use my website. On my new website, customers should first identify themselves by their unique customer identification number. Then they should be able to browse my catalog and to place orders online.

Relationship constraints

Many-to-many relationship sets

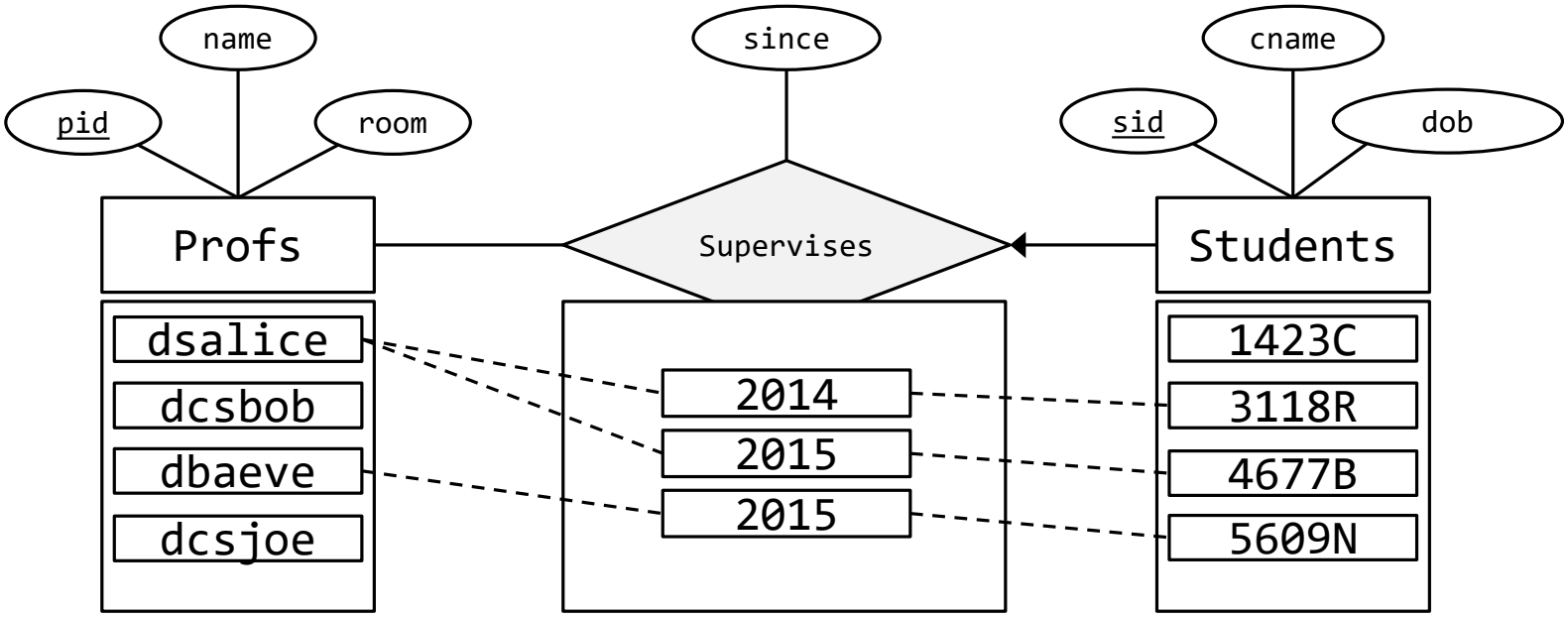
- Example
  - Each student can enroll in *0 or more* courses
  - Each course can be enrolled by *0 or more* students



Relationship constraints

Key constraints: one-to-many relationships

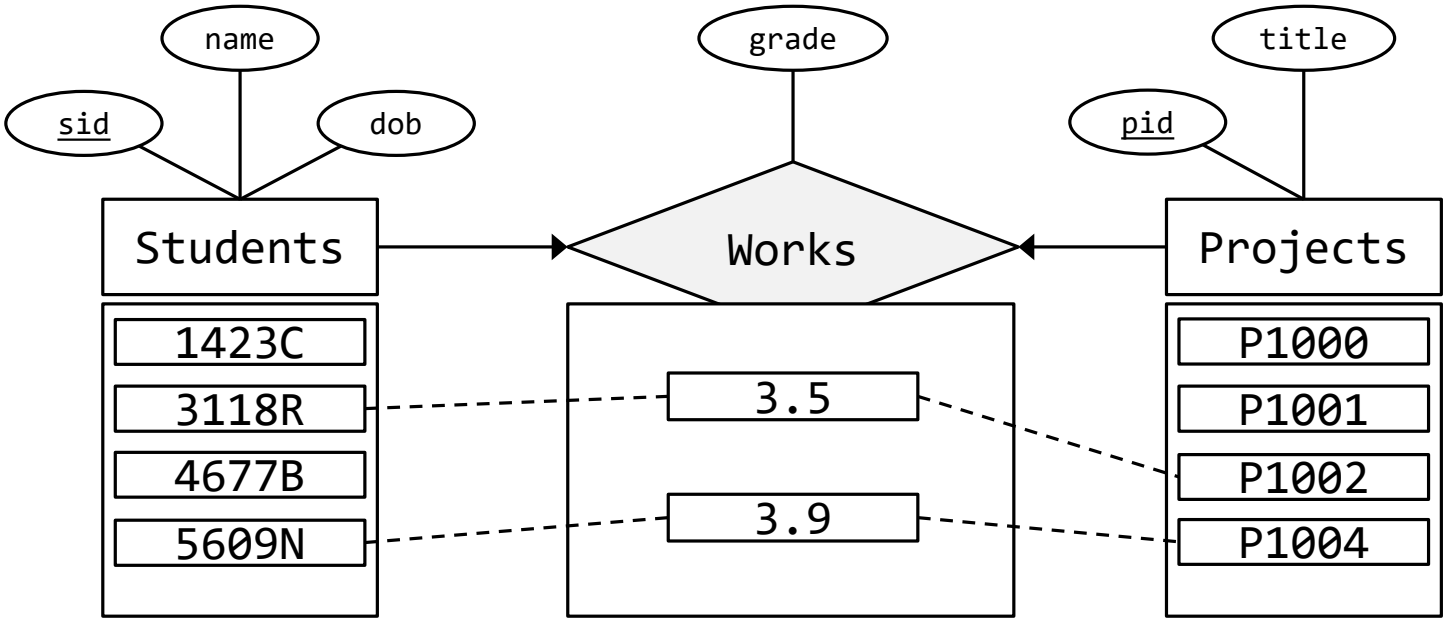
- Let R be a relationship set that involves entity set E
- Key constraint on E w.r.t. R
  - Each instance E can participate in *at most one* instance of R
- Example:
  - Each professor can supervise *0 or more* students
  - Each student can be supervised by *at most one* professor



Relationship constraints

Key constraints: one-to-one relationships

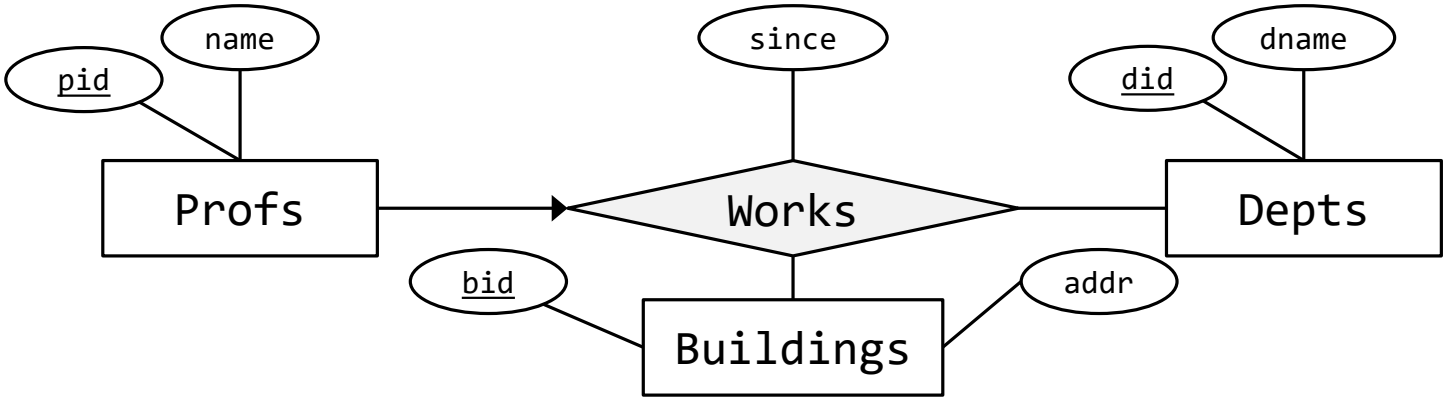
- Example:
  - Each student can work on *at most one* project
  - Each project can be worked on by *at most one* student

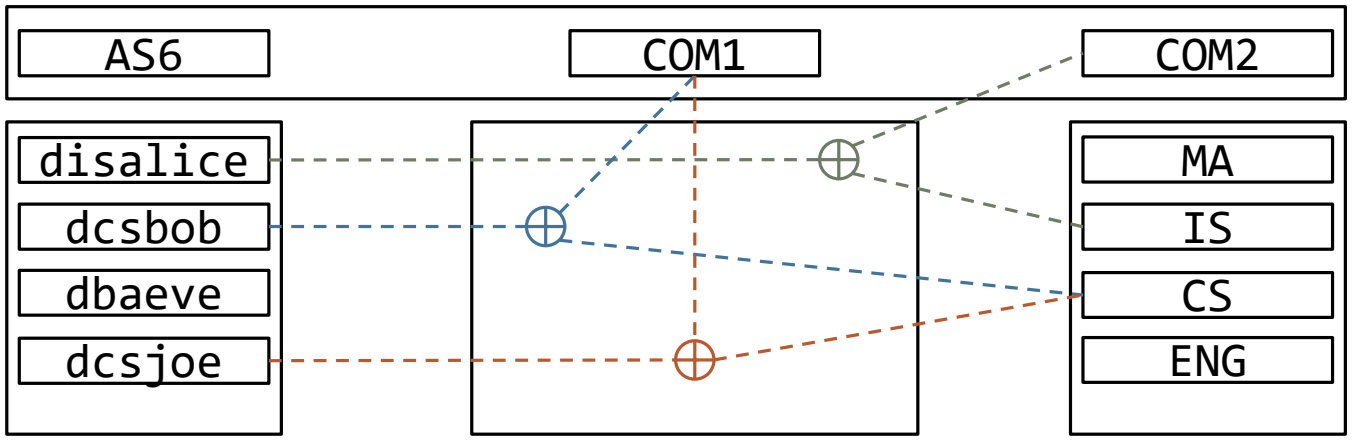


Relationship constraints

Key constraints: N-ary relationships

- Example:
  - Each professor can work in *at most one* department located at *some* building

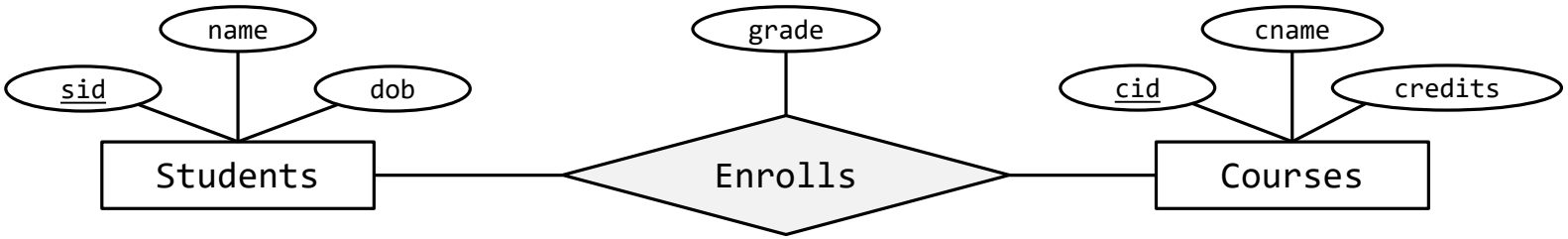




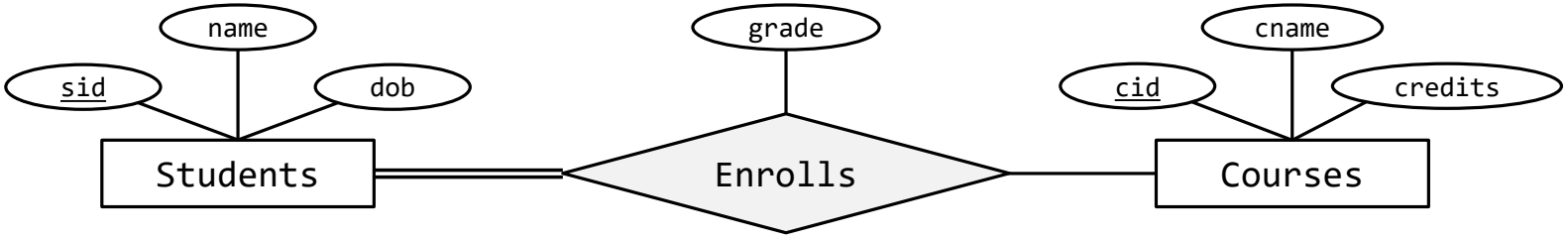
Participation constraints

Introduction

- **Participation constraints:** is the participation of an entity set in a relationship set *mandatory*?
  - **Partial participation constraint**
    - Example: each student can enroll in *0 or more* courses



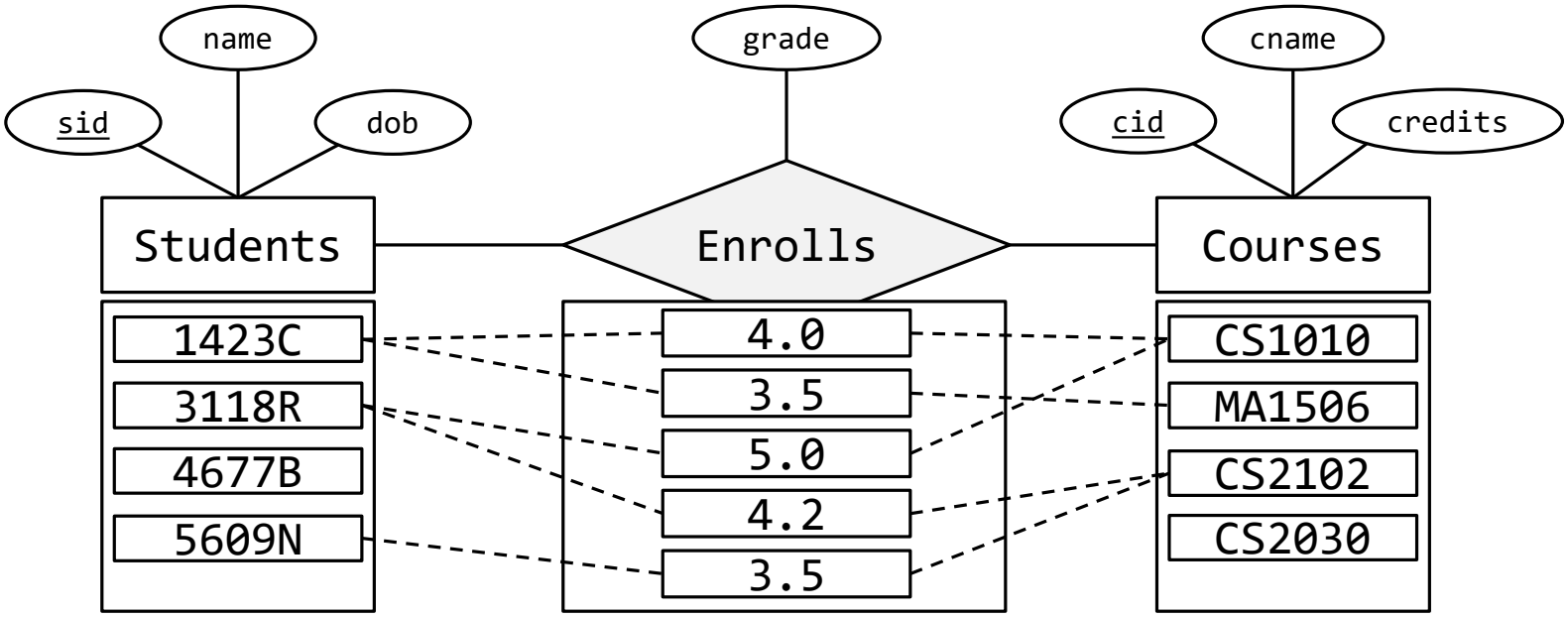
  - **Total participation constraint**
    - Example: each student must enroll in *at least one* course



Participation constraints

Partial participation constraints

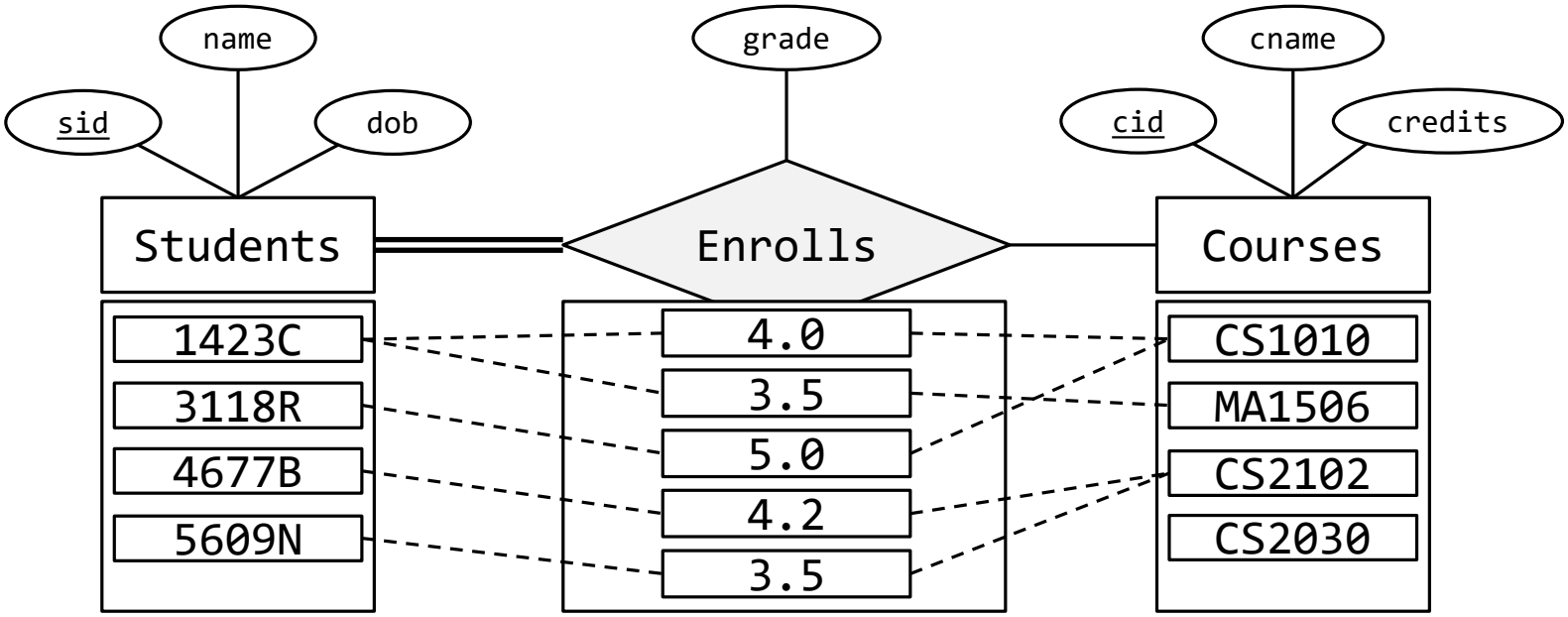
- Example
  - Each student can enroll in *0 or more* courses
  - Each course can be enrolled by *0 or more* students



Participation constraints

Total participation constraints

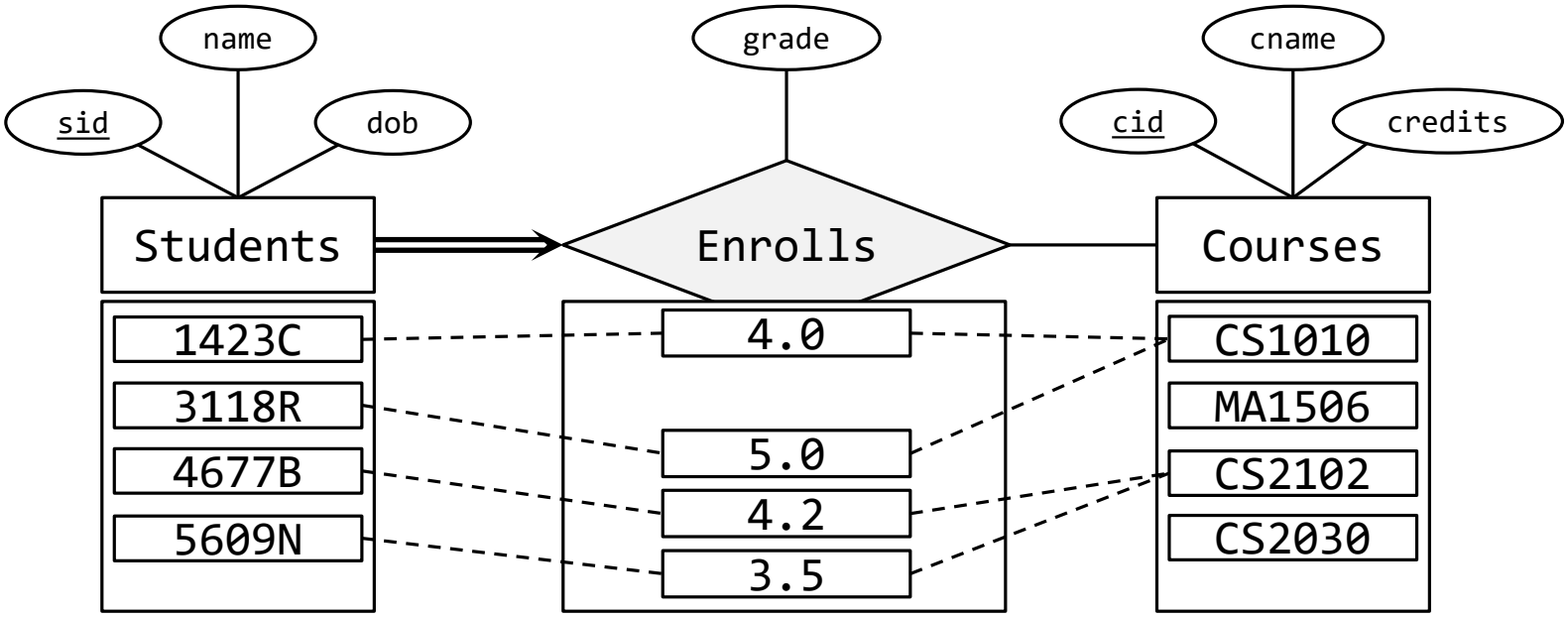
- Example
  - Each student must enroll in *at least one* courses
  - Each course can be enrolled by *0 or more* students



Participation constraints

Key & Total participation constraints

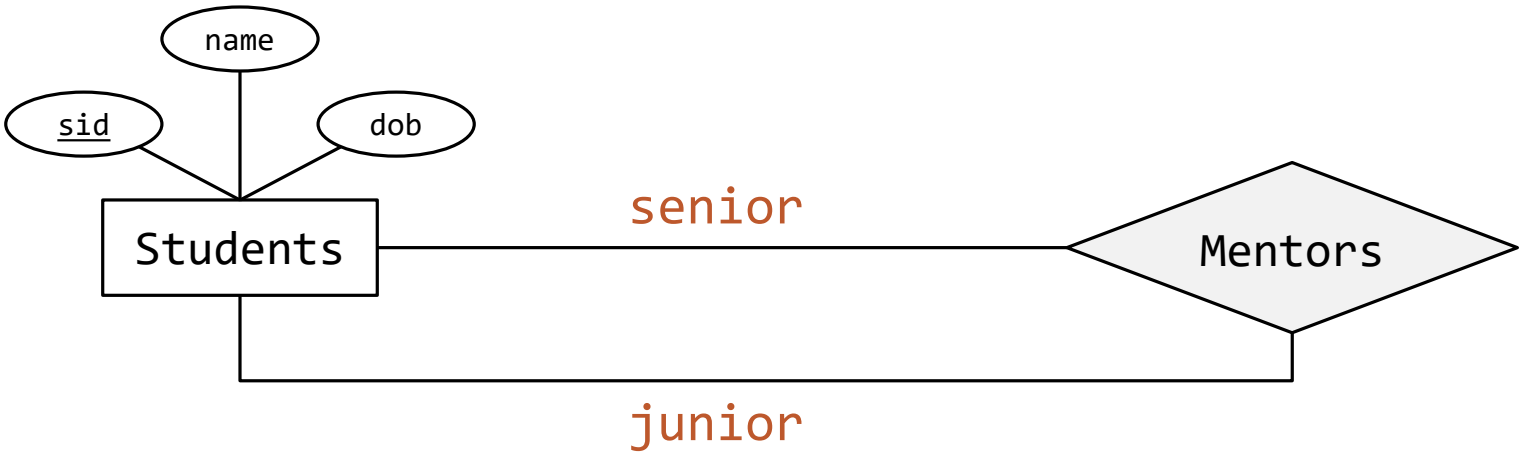
- Example
  - Each student must enroll in *at most one* & *at least one* courses
  - Each course can be enrolled by *0 or more* students



Roles in relationship

Roles

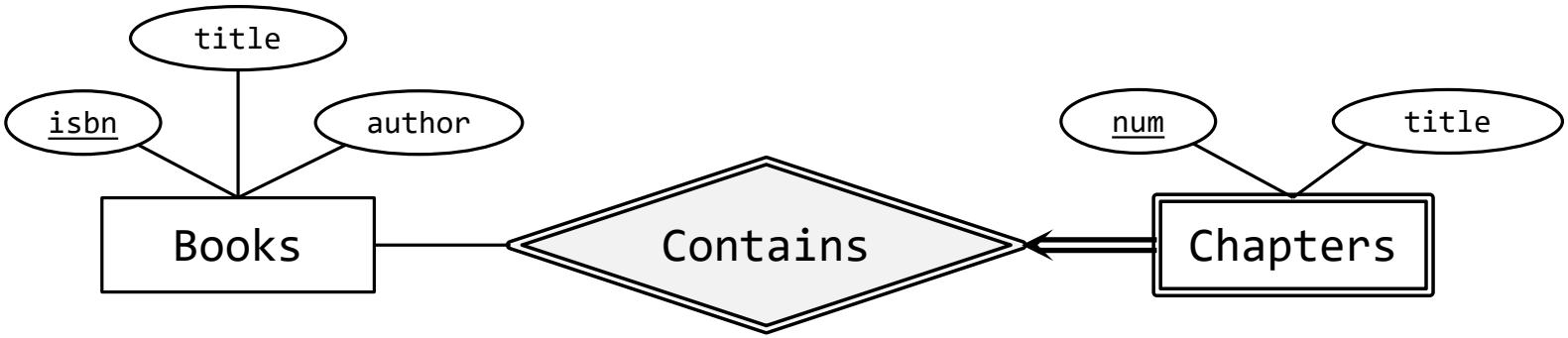
- Roles are used when one entity set appears *two or more times* in a relationship set



Weak entity sets

Definition

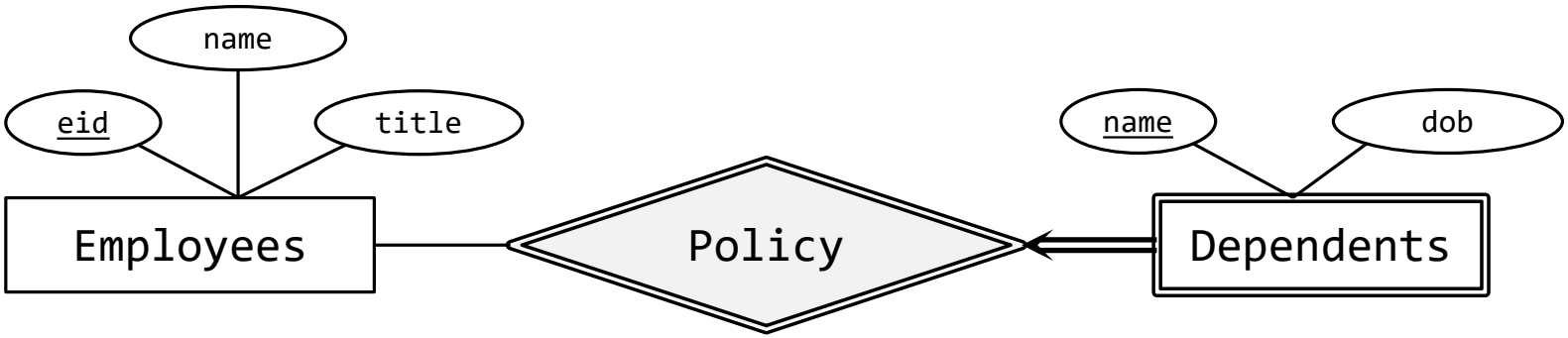
- Weak entity set is an entity set that *does not have its own key*
- A weak entity can only be uniquely identified by considering the *primary key of another entity* (called identifying owner)
  - There must be a *many-to-one relationship* (called identifying relationship) from the weak entity set to an owner entity set
  - Weak entity set must have *total participation* in identifying relationship



Weak entity sets

Properties

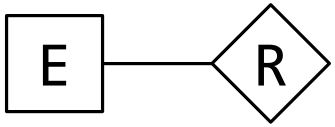
- Partial key of a weak entity set is a set of attributes of weak entity set that uniquely identifies a weak entity *for a given owner entity*
- A weak entity's existence is *dependent on the existence* of its owner entity
- Weak entity sets are represented by double-lined rectangles
- Identifying relationship sets are represented by double-lined diamonds



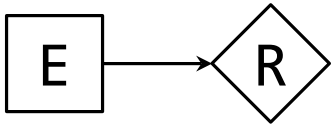
Summary

Relationship constraints

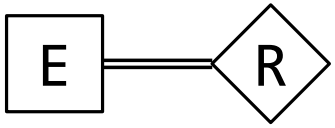
- Many-to-many — Each instance of E participates in *0 or more* instance of R



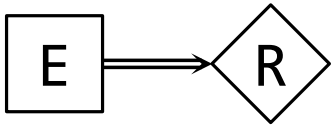
- Key — Each instance of E participates in *at most 1* instance of R



- Total — Each instance of E participates in *at least 1* instance of R



- Key & total — Each instance of E participates in *exactly one* instance of R



- Weak entity — E is a weak entity set with identifying owner E' and identifying relationship set R

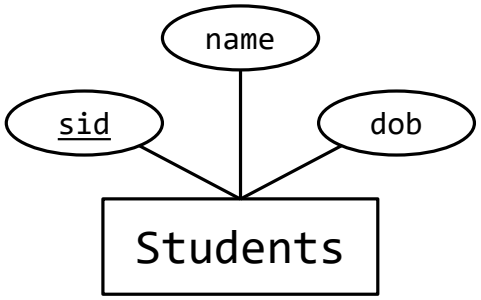
- Entity Relationship Diagram
  - ER model
  - Relationship constraints
  - Participation constraints
  - Weak entity sets
- ER to SQL
  - ER diagram to SQL
  - Additional ER concepts
  - ER design and relational mapping

ER to SQL

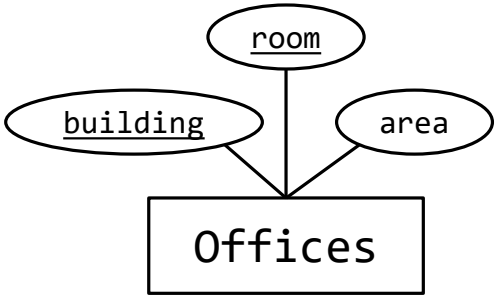
ER diagram to SQL

Entity sets

```
CREATE TABLE Students (
   sid      integer,
   name     varchar(30),
   dob      date,
   PRIMARY KEY(sid)
);
```

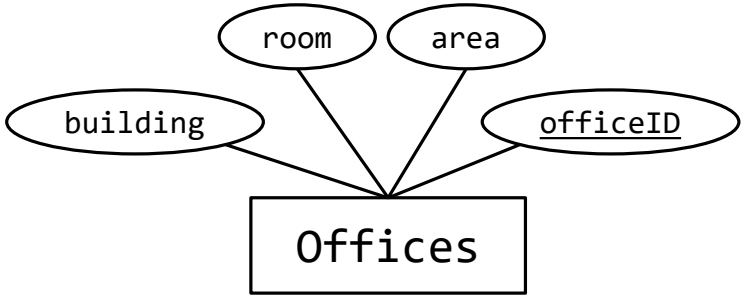



```
CREATE TABLE Offices (
   building      char(10),
   room          integer,
   area          varchar(20),
   PRIMARY KEY(building,room)
);
```



ER diagram to SQL

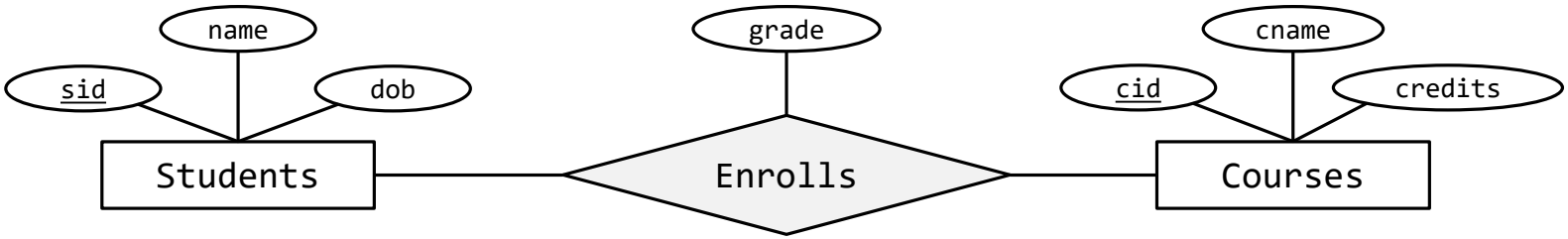
Entity sets with candidate keys




```
CREATE TABLE Offices (
  officeID     char(10) PRIMARY KEY,
  building     char(10) NOT NULL,
  room         integer  NOT NULL,
  area         varchar(20),
  UNIQUE(building,room)
);
```

ER diagram to SQL

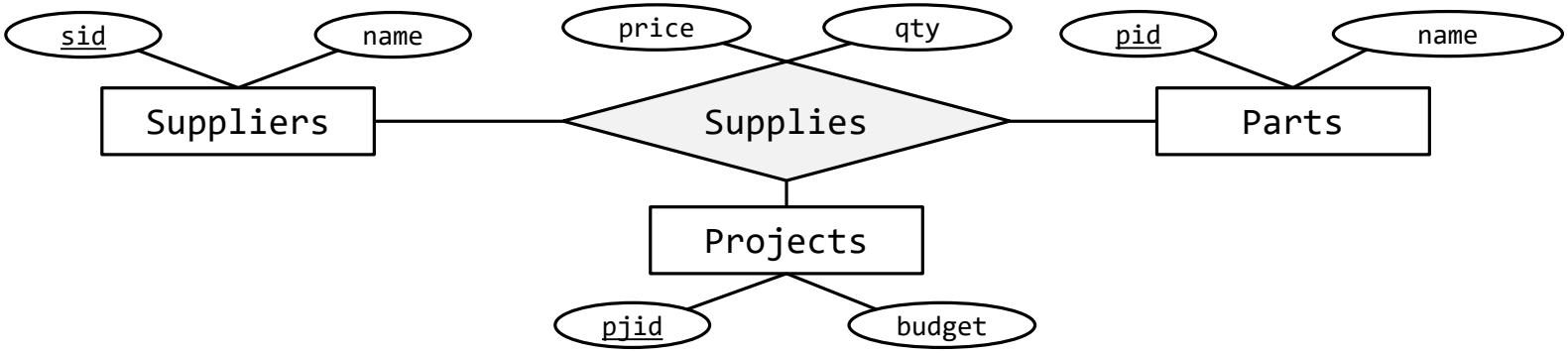
Relationship sets without constraints




```
CREATE TABLE Enrolls (
  sid     integer REFERENCES Students,
  cid     char(5) REFERENCES Parts,
  qty     numeric,
  PRIMARY KEY(sid,cid)
);
```

ER diagram to SQL

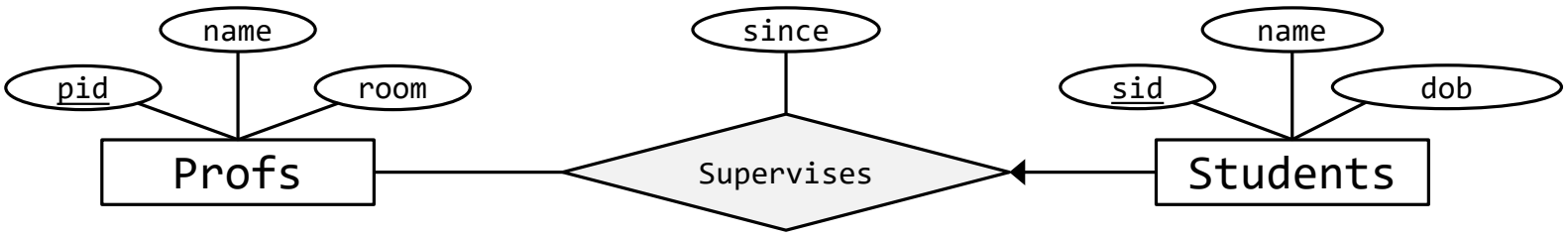
Relationship sets without constraints




```
CREATE TABLE Supplies (
  sid     char(10) REFERENCES Suppliers,
  pid     char(10) REFERENCES Parts,
  pjid    char(10) REFERENCES Projects,
  price   numeric,
  qty     integer,
  PRIMARY KEY(sid,pid,pjid)
);
```

ER diagram to SQL

Relationship sets with key constraints



First approach

Represent `Supervises` with a *separate table*

- `Profs (pid, name, room)`
- `Students (sid, name, dob)`
- `Supervises (sid, pid, since)`

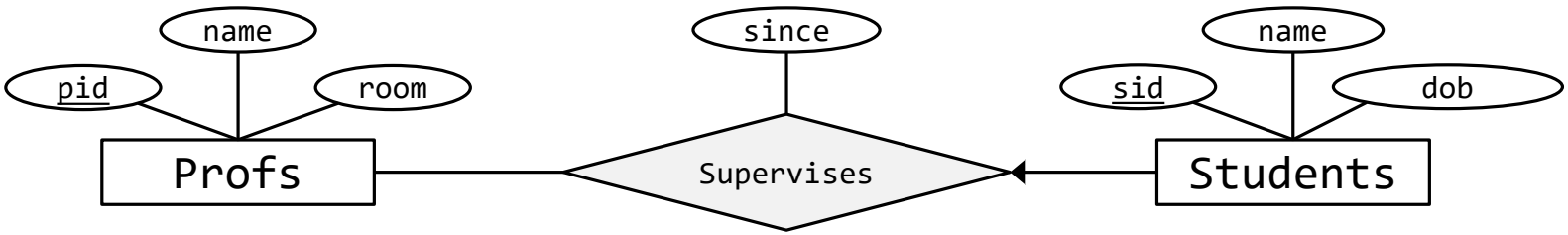
Second approach

Combine `Supervises` & `Students` into a *single table*

- `Profs (pid, name, room)`
- `SupervisedStudents ( sid, name, dob, pid, since )`

ER diagram to SQL

Relationship sets with key constraints



First approach

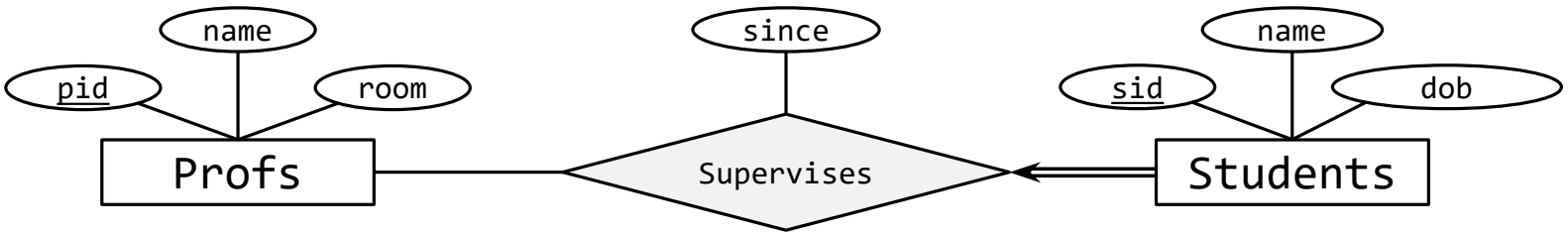
```
CREATE TABLE Supervises (
  sid    integer,
  pid    char(7),
  since  date,
  PRIMARY KEY(sid),
  FOREIGN KEY(sid) REFERENCES
    Students,
  FOREIGN KEY(pid) REFERENCES
    Profs
);
```

Second approach

```
CREATE TABLE SupervisedStudents (
  sid    integer,
  name   varchar(30),
  dob    date,
  pid    char(7),
  since  date,
  PRIMARY KEY(sid),
  FOREIGN KEY(pid) REFERENCES
    Profs
);
```

ER diagram to SQL

Relationship sets with key & total constraints



First approach

```
CREATE TABLE Supervises (
  sid    integer,
  pid    char(7),
  since  date,
  PRIMARY KEY(sid),
  FOREIGN KEY(sid) REFERENCES
    Students,
  FOREIGN KEY(pid) REFERENCES
    Profs
);
```

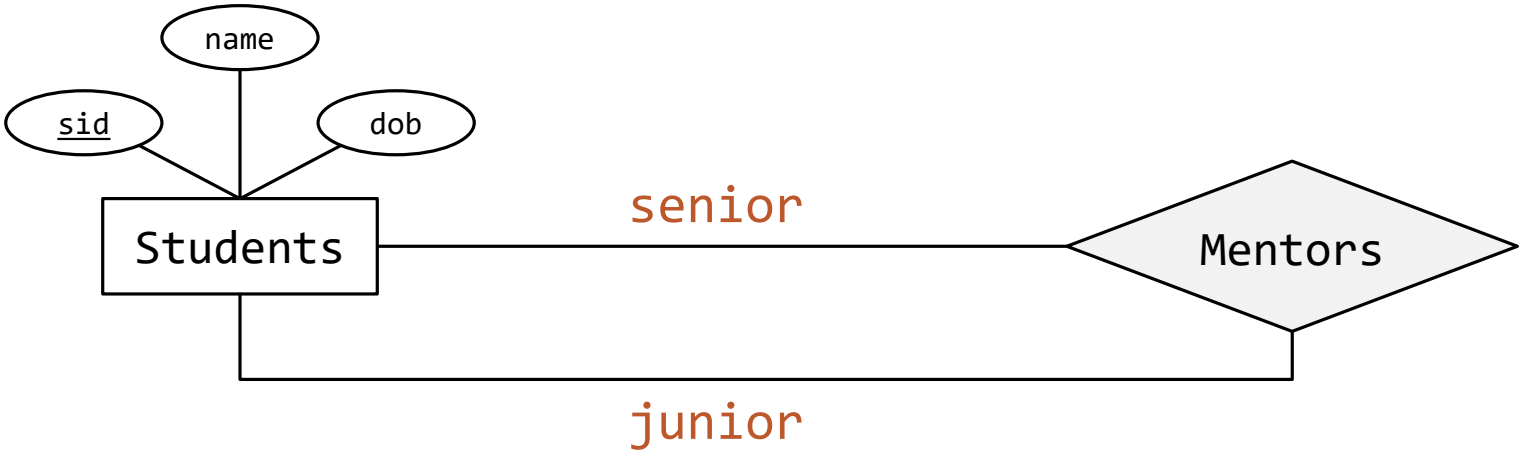
Total participation constraint of Students w.r.t Supervises is not captured!

Second approach

```
CREATE TABLE SupervisedStudents (
  sid    integer,
  name   varchar(30),
  dob    date,
  pid    char(7) NOT NULL,
  since  date,
  PRIMARY KEY(sid),
  FOREIGN KEY(pid) REFERENCES
    Profs
);
```

ER diagram to SQL

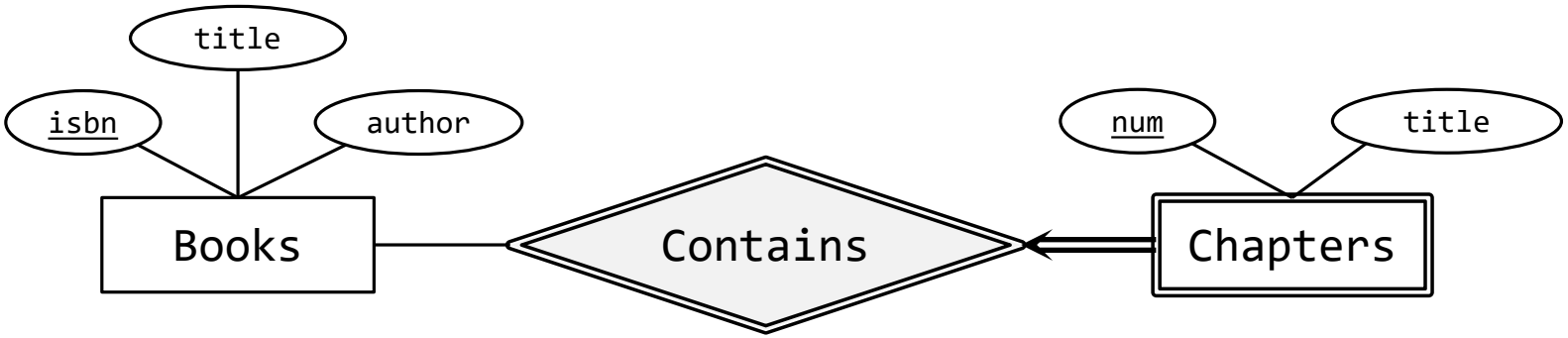
Roles in relationships




```
CREATE TABLE Mentors (
  seniorSID  integer,
  juniorSID  integer,
  PRIMARY KEY(seniorSID, juniorSID),
  FOREIGN KEY(seniorSID) REFERENCES Students(sid),
  FOREIGN KEY(juniorSID) REFERENCES Students(sid)
);
```

ER diagram to SQL

Weak entity sets



Books table

```
CREATE TABLE Books (
  isbn    char(30),
  title   char(50),
  author  char(60),
  PRIMARY KEY(isbn)
);
```

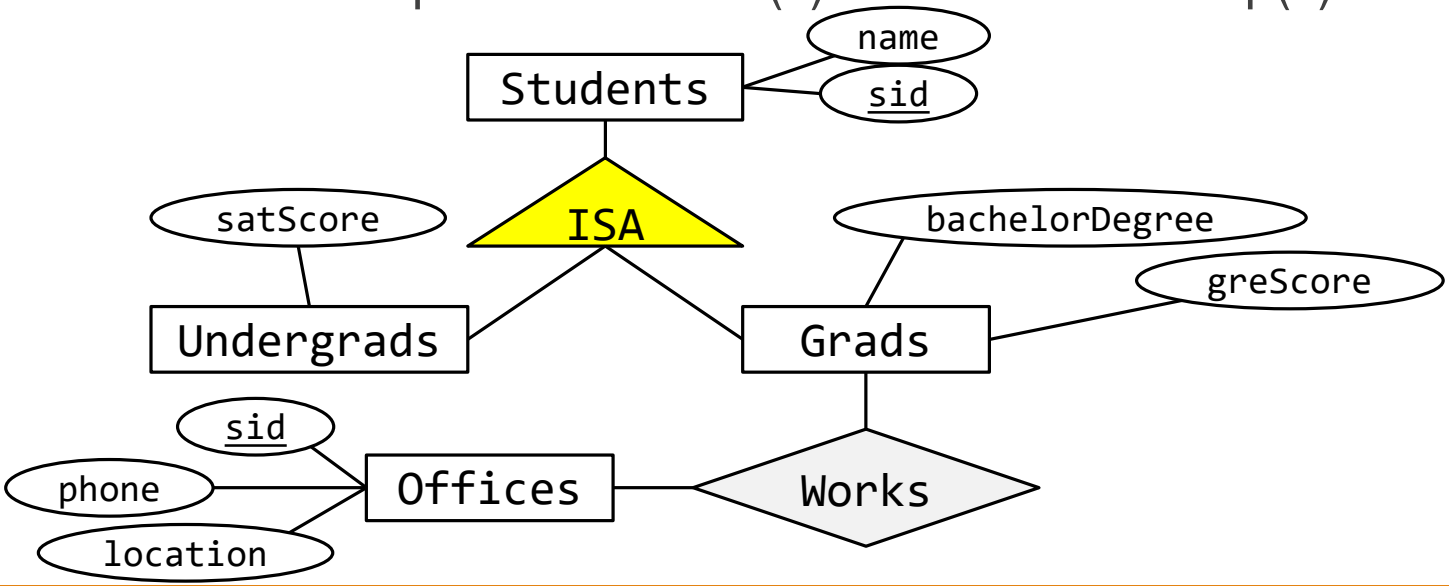
BookChapters table

```
CREATE TABLE BookChapters (
  num     char(30),
  title   char(50),
  isbn    char(60),
  PRIMARY KEY(num,isbn),
  FOREIGN KEY(isbn)
    REFERENCES Books
    ON DELETE cascade
);
```

Additional ER concepts

ISA hierarchies

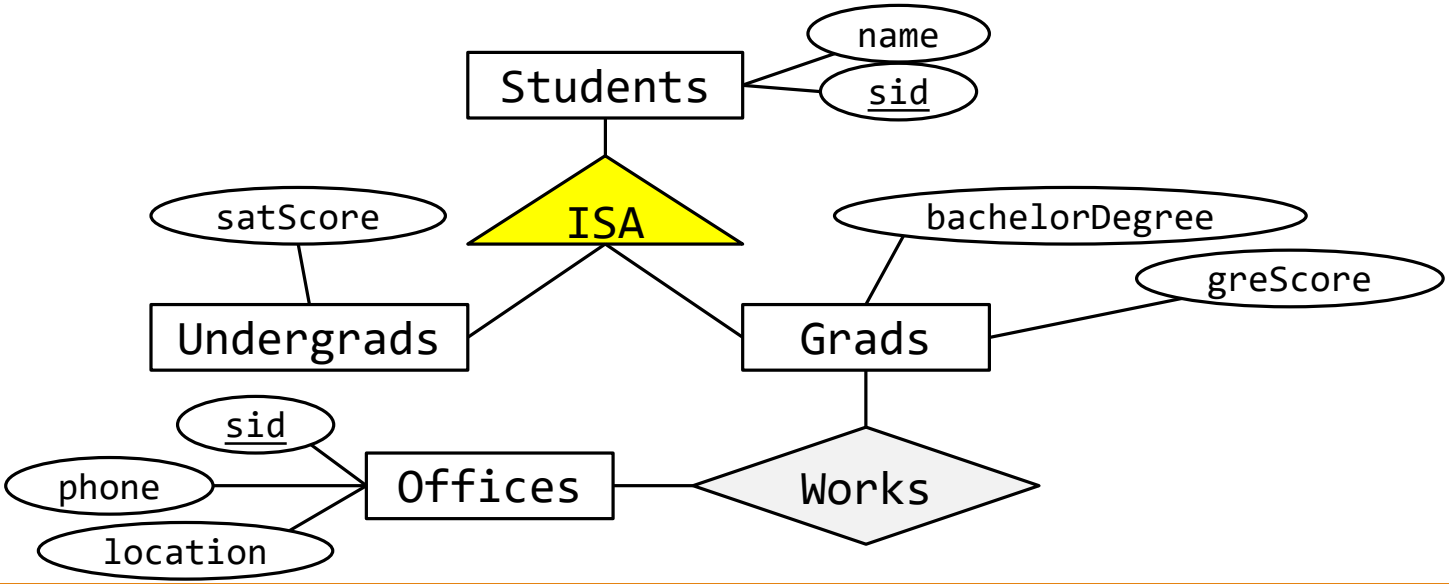
- Based on “is-a” relationship in OOP
  - Subclass-superclass relationship
  - Describing an entity sets into subclasses
- Every entity in a subclass entity set is an entity in its superclass entity set
- Each subclass has specific attribute(s) and/or relationship(s)



Additional ER concepts

ISA hierarchies

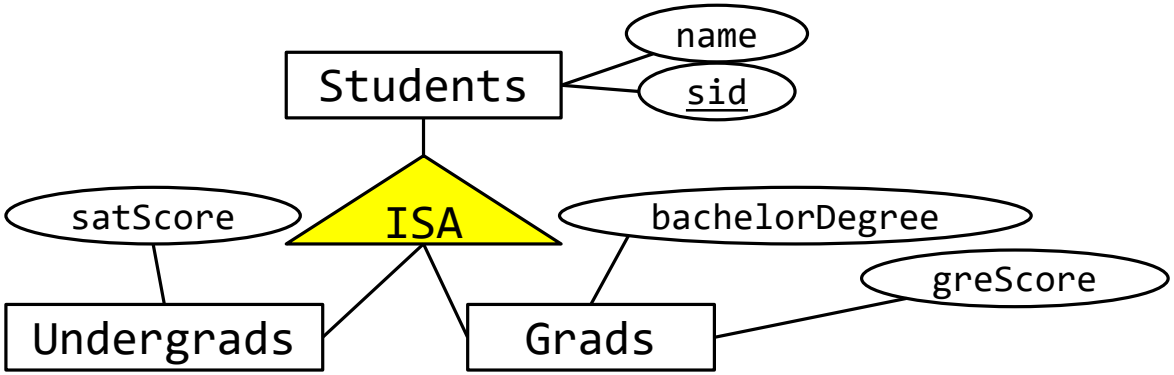
- Constraints:
  - Overlap constraints: can entity belong to multiple subclasses?
    - Satisfied if entity in superclass *could belong to multiple subclasses*
  - Covering constraints: does an entity in a superclass have to belong to some subclass?
    - Satisfied if every entity in a superclass *has to belong to some subclass*



Additional ER concepts

ISA hierarchies

- **Approach #1**: one relation per subclass or superclass

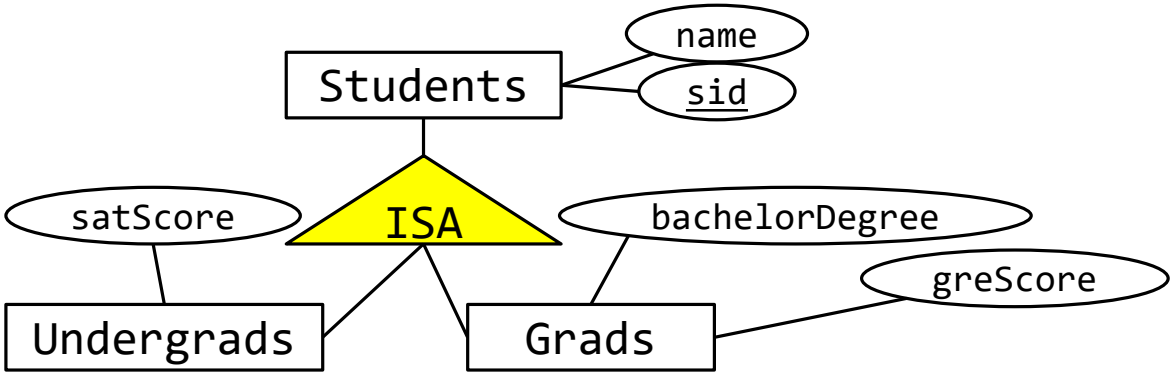



```
CREATE TABLE Students (
  sid       integer PRIMARY KEY,
  name      char(30) );

CREATE TABLE Undergrads (
  sid       integer PRIMARY KEY REFERENCES Students
            ON DELETE cascade,
  satScore  numeric );

CREATE TABLE Grads (
  sid       integer PRIMARY KEY REFERENCES Students
            ON DELETE cascade,
  greScore  numeric );
```

Additional ER concepts

ISA hierarchies

- **Approach #2**: one relation per subclass



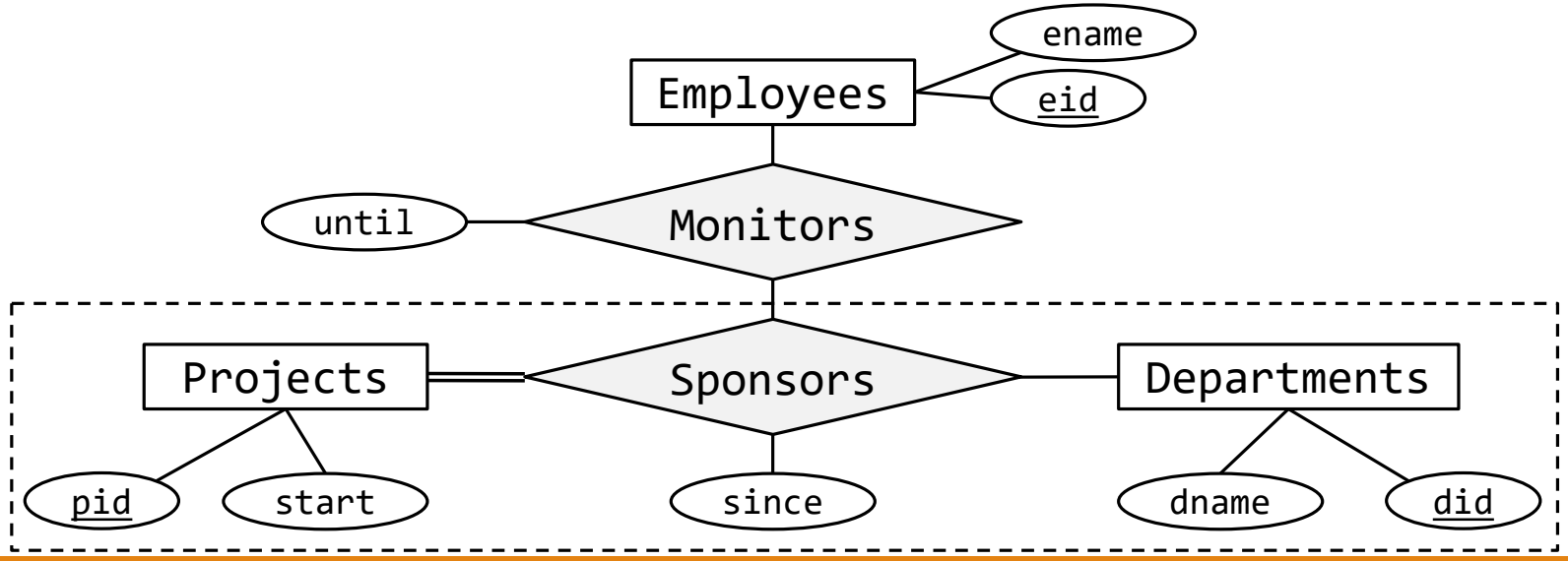
Applicable if covering constraint is satisfied

```
CREATE TABLE Undergrads (
  sid       integer PRIMARY KEY,

  satScore  numeric );

CREATE TABLE Grads (
  sid       integer PRIMARY KEY,

  greScore  numeric );
```

Additional ER concepts

Aggregation

- How to model a relationship between entities & relationships?
- Example:
  - Every project is sponsored by at least one department
  - Each sponsorship has a “since” attribute & might be monitored by 0 or more employees
  - Each monitoring has an “until” attribute



Additional ER concepts

Aggregation

- Relational mapping

```
CREATE TABLE Projects (
  pid       char(20),
  start     date,
  PRIMARY KEY(pid)
);
```

```
CREATE TABLE Departments (
  did       char(20),
  dname     char(30),
  PRIMARY KEY(did)
);
```

```
CREATE TABLE Sponsors (
  pid       char(20)
    REFERENCES Projects,
  did       char(30),
    REFERENCES Departments,
  since     date,
  PRIMARY KEY(pid,did)
);
```

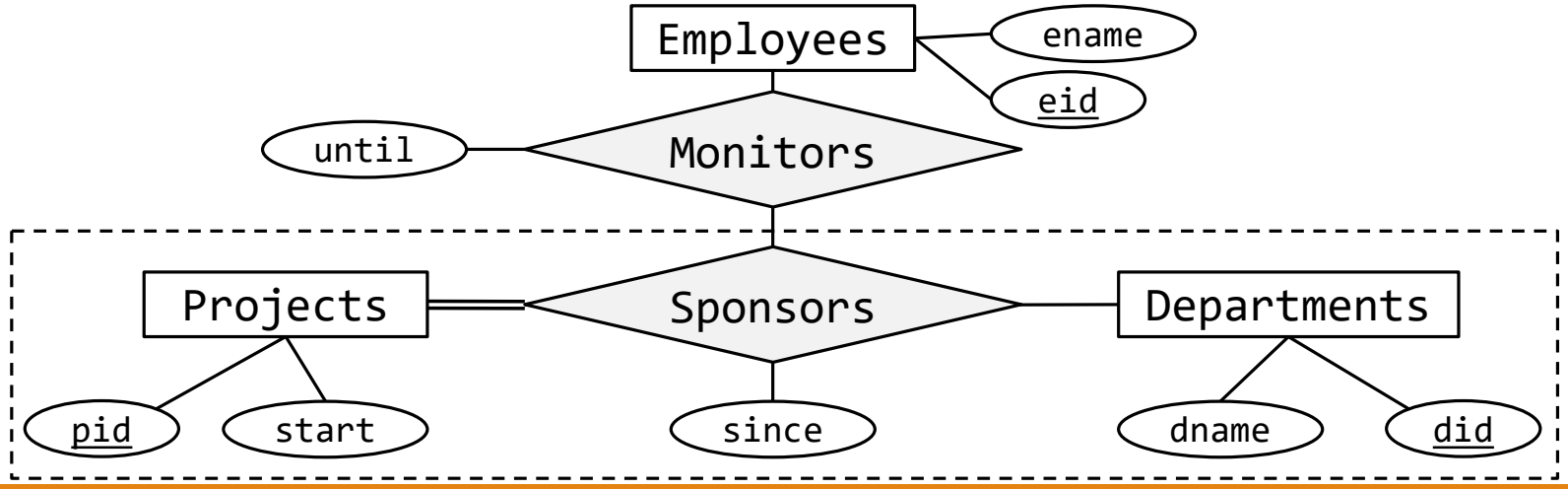
```
CREATE TABLE Employees (
  eid       char(20),
  ename     char(30),
  PRIMARY KEY(eid)
);
```

Additional ER concepts

Aggregation

- Relational mapping

```
CREATE TABLE Monitors (
  eid       char(20) REFERENCES Employees,
  pid       char(30),
  did       char(30),
  until     date,
  PRIMARY KEY(eid,pid,did),
  FOREIGN KEY(pid,did) REFERENCES Sponsors(pid,did)
);
```



ER design and relational mapping

Guidelines for ER design

- ER design should capture as many of the application's constraints as possible
- ER design must not impose any constraint that is not required in the application

Guidelines for relational mapping

- Relational schema should enforce as many of the application's constraints as possible using column/table constraints
- Relational schema must not impose any constraint that is not required in the application

Summary

- ER model has expressive constructs for conceptual data design
  - Concepts: *entities; relationships; attributes; weak entities; ISA hierarchies; aggregation*
  - Constraints: *key constraints; participation constraints*
- ER design is subjective
- Rules for mapping entity-relationship model to relational model
  - Entity & relationship sets
  - Key constraints
  - Participation constraints
  - Relationship roles
  - Weak entity sets
  - ISA hierarchies
  - Aggregation