Relational Model: Integrity Constraints

Kathleen Durant
CS 3200
Lesson 3A

Outline for today

- Representing constraints from the ERM in the Relational model
- Examples

Integrity Constraints

- Integrity Constraint (IC) is condition that must be true for *every* instance of the database; e.g., **domain constraints**.
  - ICs are specified when schema is defined.
  - ICs are checked when relations are modified.
- A legal instance of a relation is one that satisfies all specified ICs.
  - DBMS should not allow illegal instances.
- If the DBMS checks ICs, stored data is more faithful to real-world meaning.
- Avoids data entry errors

Key Constraints

- A set of fields is a key for a relation if :

  1. No two distinct tuples can have the same values in all key fields, and

  2. This is not true for any subset of the key.

  - Part 2 false? A superkey.
  - If there's >1 key for a relation, one of the keys is chosen to be the primary key.
    - E.g., sid is a key for Students.
    - What about student name?
  - The set {sid, gpa} is a superkey.

Specifying a primary key

- Primary key specified while creating a table
- CREATE TABLE Enrolled (sid CHAR(20), cid CHAR(40), grade CHAR(2), **PRIMARY KEY** (sid, cid))

Foreign Keys and Referential Integrity

- Foreign key: Set of fields in one relation that is used to `refer’ to a tuple in another relation.
  - Must correspond to primary key of the second relation.
  - Like a `logical pointer’. E.g., sid in Enrolled is a foreign key referring to Students:
  - Enrolled(sid int, cid char(20), grade char(2))
  - If all foreign key constraints are enforced, referential integrity is achieved, i.e., no dangling references.

Foreign Keys

- Only students listed in the Students relation should be allowed to enroll for courses.
- CREATE TABLE Enrolled (sid int, cid CHAR(20), grade CHAR(2), PRIMARY KEY (sid, cid), FOREIGN KEY (sid) REFERENCES Students)

Students

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

Enrolled

Sid	CId	Grade
55515	History 101	C
55516	Biology 220	A
55517	Anthro 320	B
55518	Music 101	A
55518	Music 101	A

Enforcing Referential Integrity

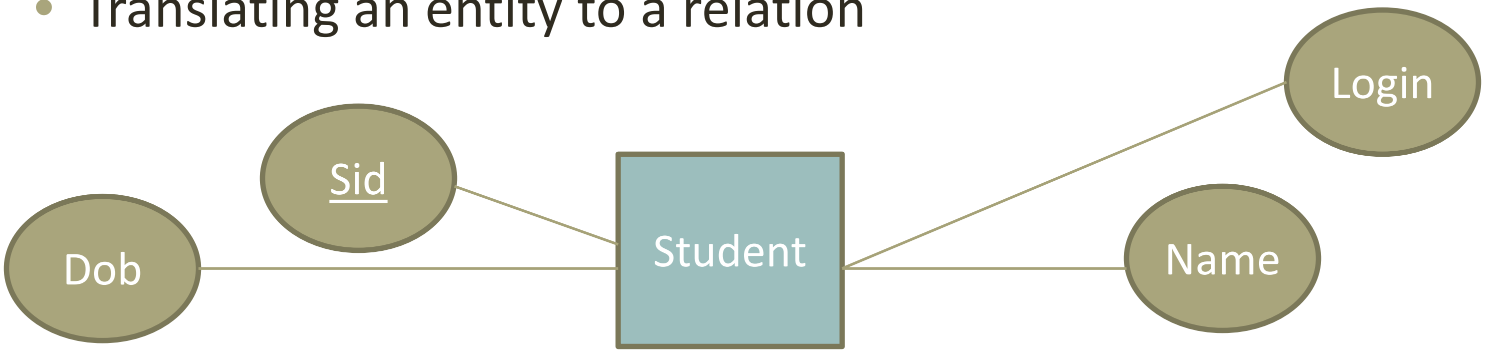
- Consider Students and Enrolled tables:
- sid in Enrolled is a foreign key that references the Students table.
- **What should be done if an Enrolled tuple with a nonexistent student id is inserted?**
  - Reject it.
- **What should be done if a Student,s tuple is deleted?**
  - **You have choices**
  1. Also delete all Enrolled tuples that refer to it.
  2. Disallow deletion of a Students tuple that is referred to.
  3. Set sid in Enrolled tuples that refer to it to a default sid.
     - (In SQL, also: Set sid in Enrolled tuples that refer to it to a special value null, denoting `unknown’ or `inapplicable’.)
- Similar if primary key of Students tuple is updated.

Specifying behavior on Referential Integrity violation

- Behavior specified at table create
  - No action (Reject action that violates constraint)
  - Update referring table (Update foreign key to the new value)
  - Set to NULL (Set all foreign keys to a NULL)
  - Set to a Default (Set all foreign keys to a default value)
- CREATE TABLE Enrolled (sid CHAR(20), cid CHAR(20), grade CHAR(2), PRIMARY KEY (sid, cid), **FOREIGN KEY (sid) REFERENCES Students ON DELETE CASCADE ON UPDATE SET DEFAULT**)

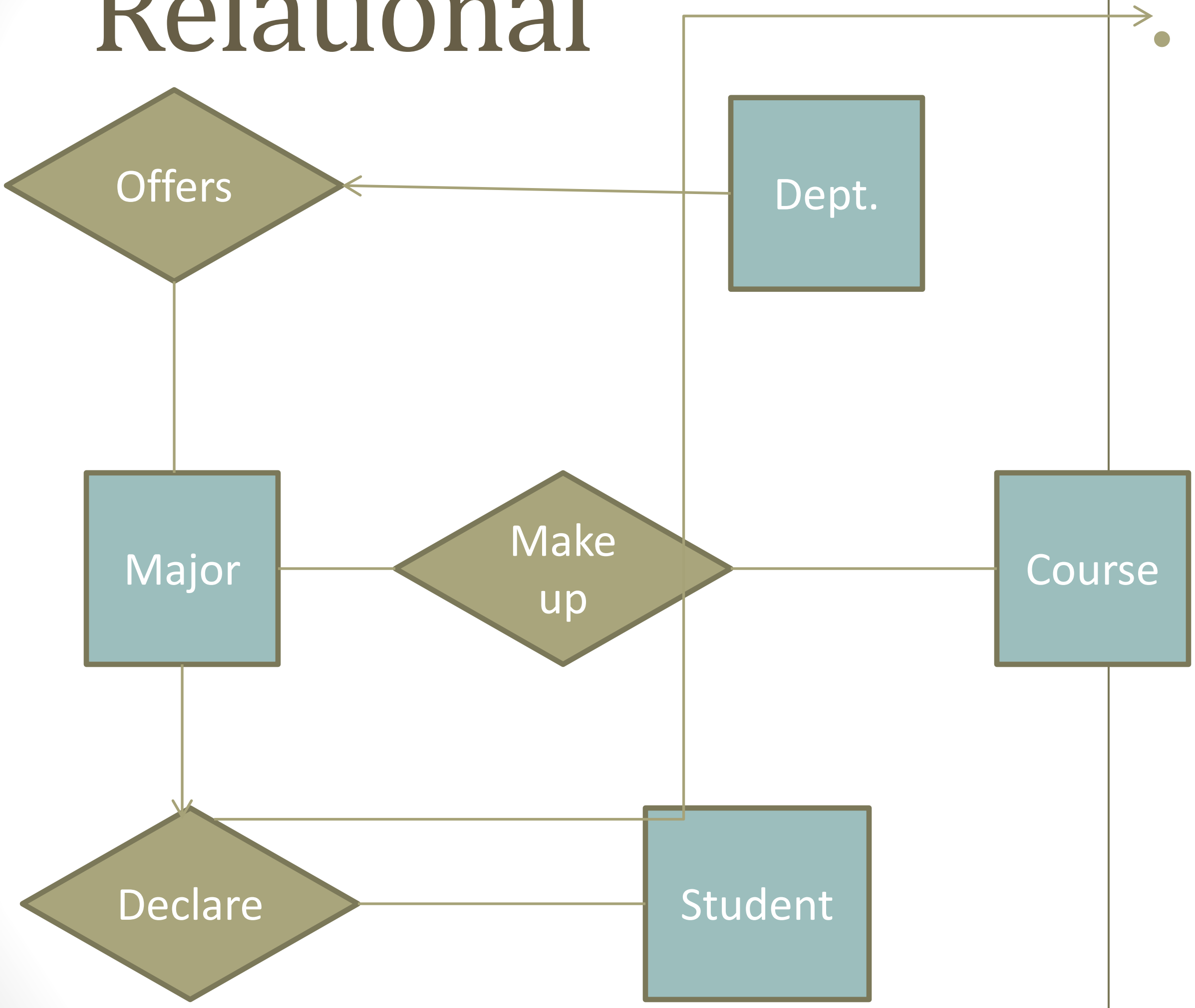
Logical DB design: ER to Relational

- Translating an entity to a relation



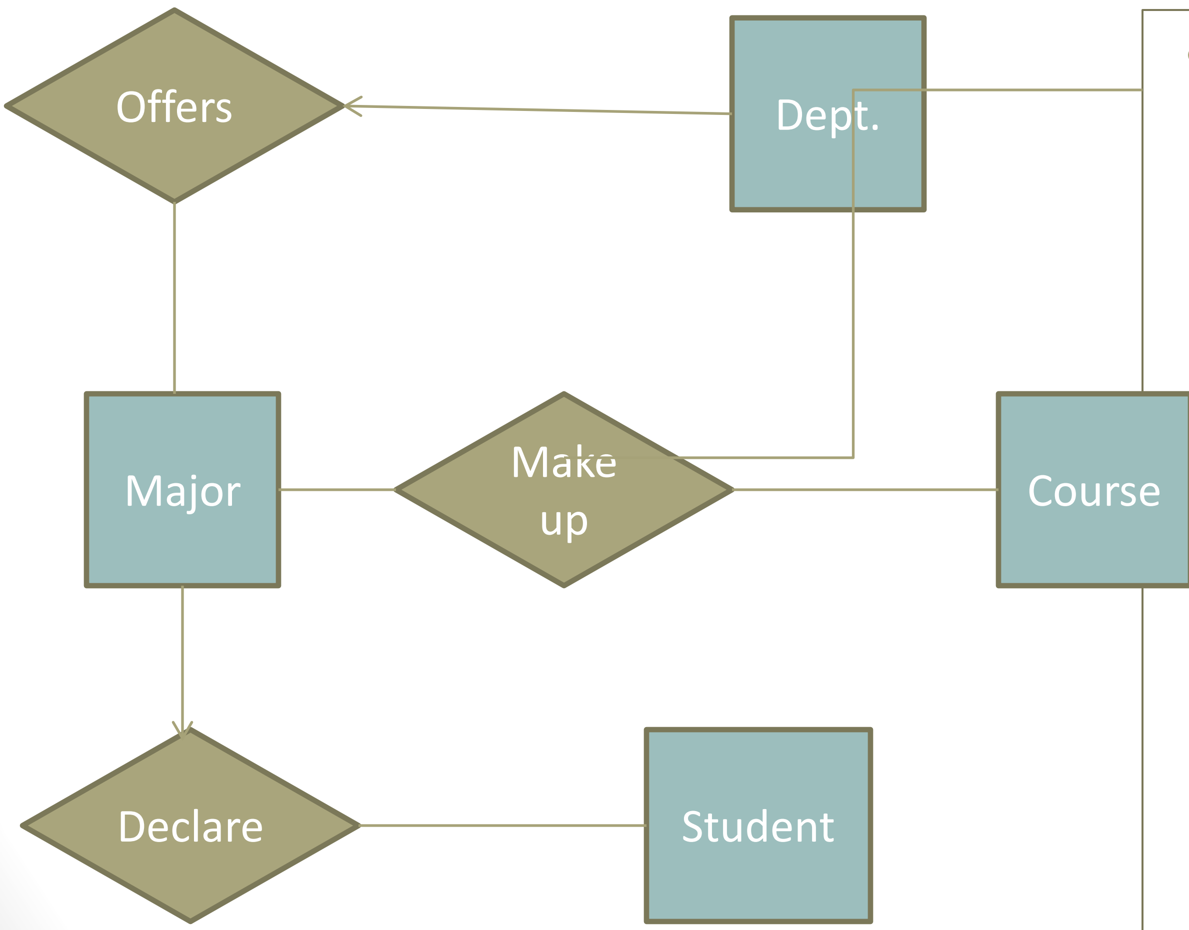
- Create table Student (sid int, Name char(20), Login(40), Dob date primary key Sid)
- In translating a relationship set to a relation, attributes of the relation must include:
  - Keys for each participating entity set (as foreign keys).
  - This set of attributes forms a superkey for the relation.
  - All descriptive attributes.

Logical DB design: ER to Relational



- Create table MajorCandidate (sid int, majorid int, yearGrad int, PRIMARY KEY (sid, majorid), foreign key sid REFERENCES Student on delete cascade on update cascade)

Total Participation Constraint

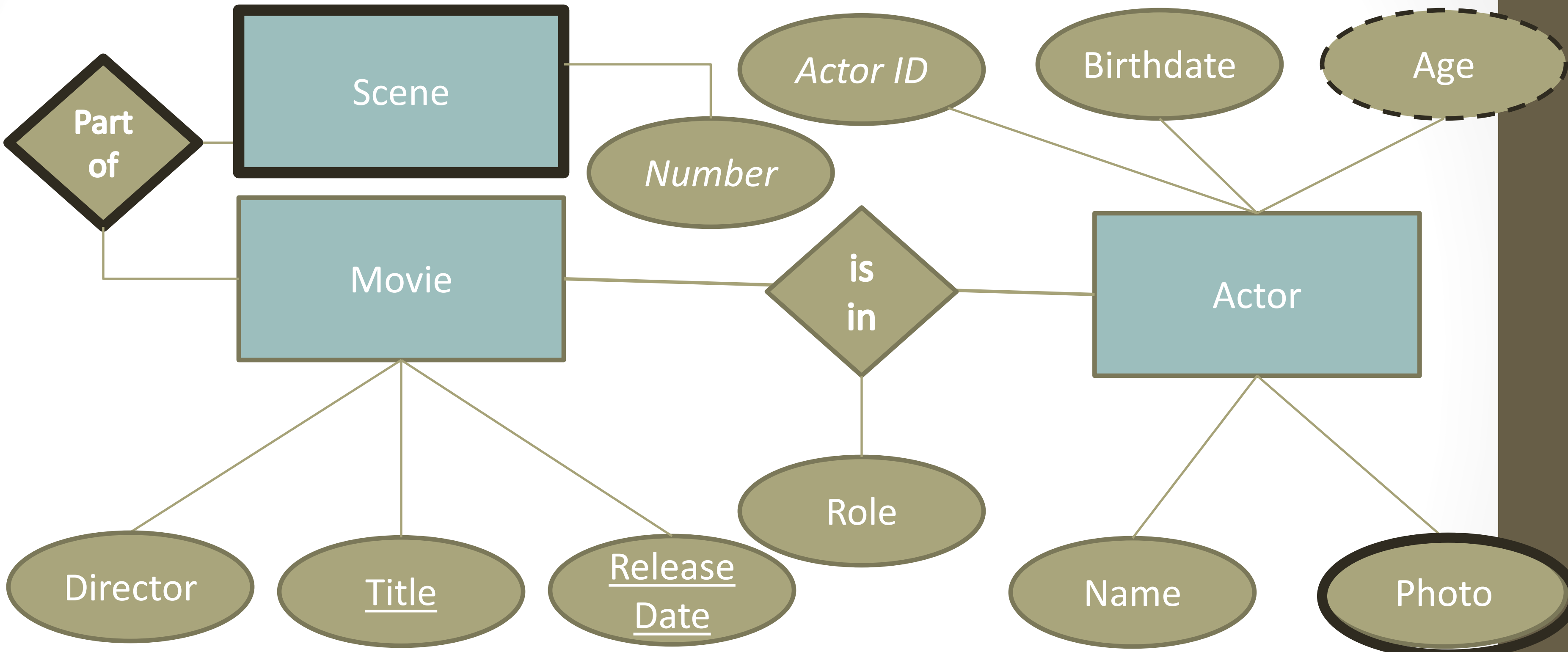


- CREATE TABLE CourseForMajor(Mid INTEGER not NULL, cid integer, PRIMARY KEY (cid), Required bool, FOREIGN KEY Mid, ON DELETE **NO ACTION**)

Total participation Constraint

- We can capture participation constraints for the combined entity+relationship relation.
- Ensure foreign key value is not null
- E.g., ‘every Mid value in Major also appears in a table Major
  - Constraint is across tables

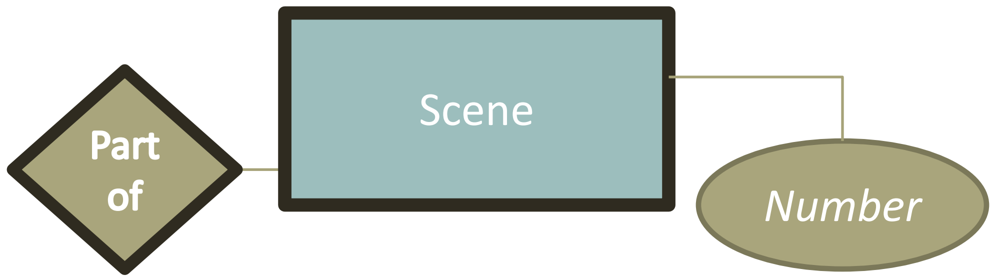
Weak Entity Constraint



- 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.

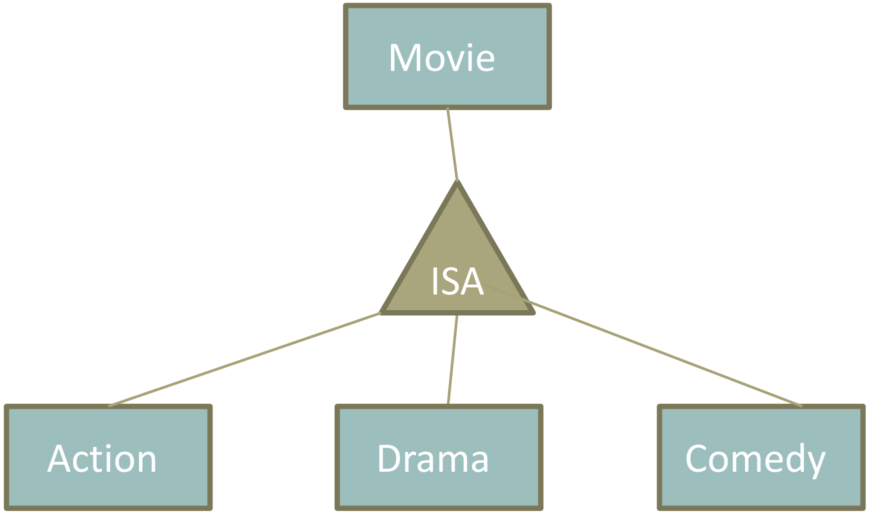
Weak entity set

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



- When the owner entity (Movie) is deleted, all owned weak entities (Scene) must also be deleted.
- CREATE TABLE MovieScene (MovieName char(20) ReleaseDate Date, SceneNumber int PRIMARY KEY (MovieName, ReleaseDate, SceneNumber), FOREIGN KEY (MovieName, ReleaseDate) REFERENCES Movies, ON DELETE CASCADE)

Translating ISA Relation



General approach:

- 4 relations: Movie, Action, Drama, Comedy
- Every movie is recorded in Movie
- For each genre extra information is stored in the corresponding table
- Must delete genre tuple if referenced movie tuple is deleted.
- Queries involving all movies: only access Movies.
- Queries on genre tables require a join to get some attributes.

Alternative: Just Genre tables

- Each movie must be in one of these two subclasses.

View

- A view is just a relation, but we store a definition, rather than a set of tuples.
- Views can be dropped using the DROP VIEW command.

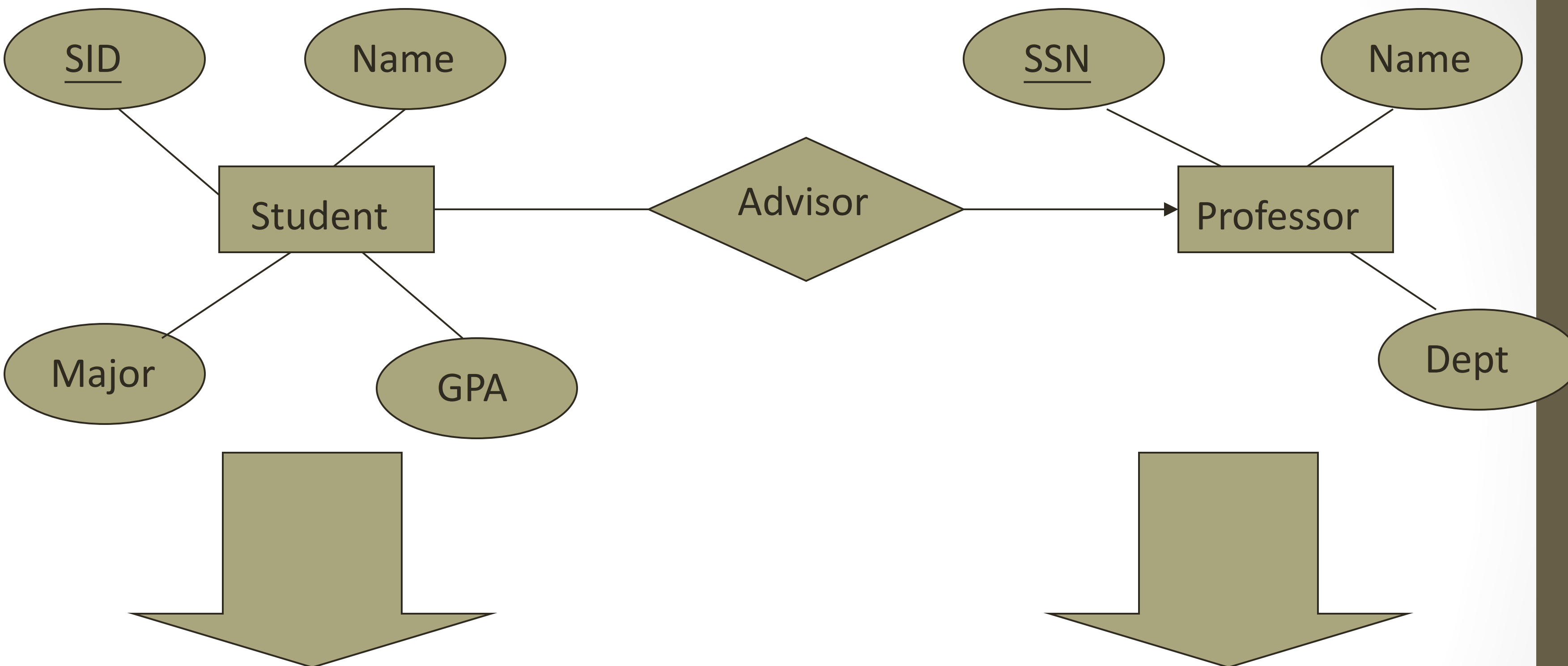
How to handle DROP TABLE if there's a view on the table?

- DROP TABLE command has options to let the user specify this.
- CREATE VIEW YoungActiveStudents (name, grade) AS SELECT S.name, E.grade FROM Students S, Enrolled E WHERE S.sid = E.sid and S.age<21

From ER Model to Relational Model

- Build a table for each entity set
- Build a table for each relationship set if necessary (more on this later)
- Make a column in the table for each attribute in the entity set
- Indivisibility Rule and Ordering Rule
- Primary Key

Example – Strong Entity Set



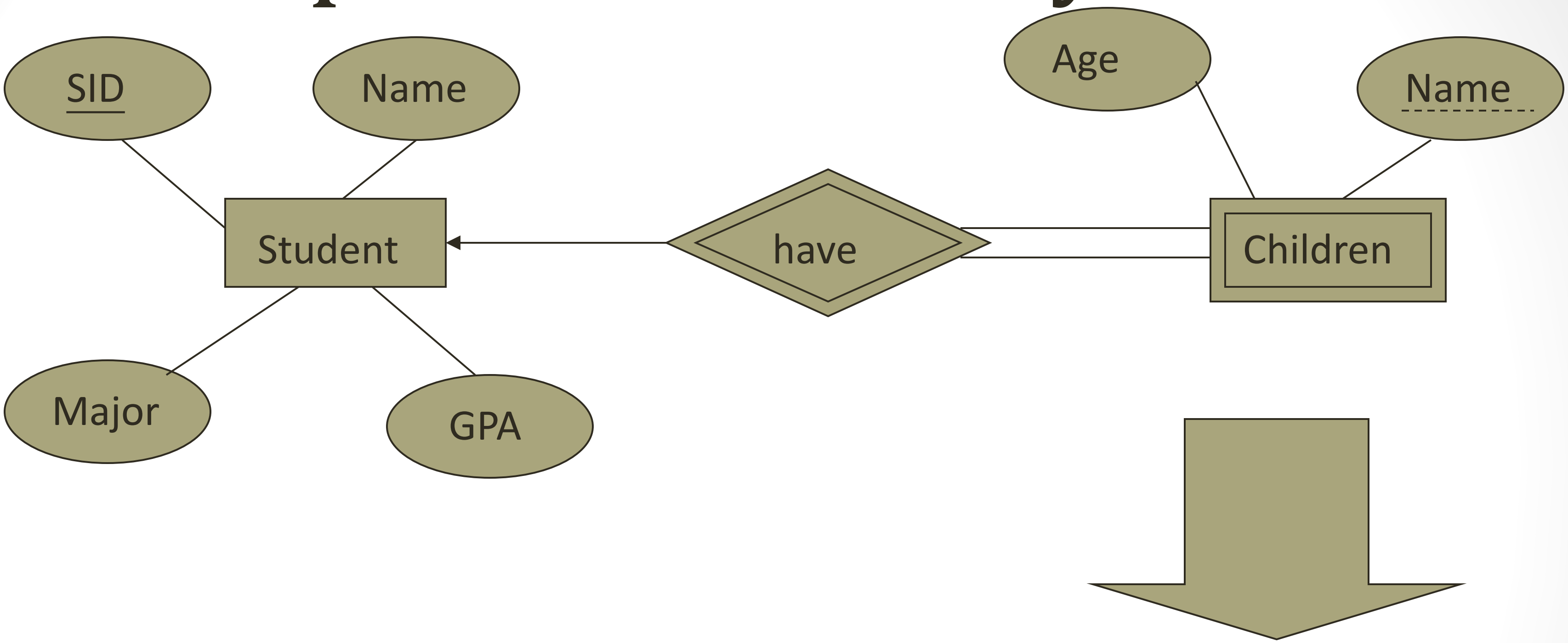
SID	Name	Major	GPA
1234	John	CS	2.8
5678	Mary	EE	3.6

SSN	Name	Dept
9999	Smith	Math
8888	Lee	CS

Representation of Weak Entity Set

- Weak Entity Set Cannot exists alone
- To build a table/schema for weak entity set
  - Construct a table with one column for each attribute in the weak entity set
  - Remember to include discriminator
  - Add the primary key of the Strong Entity Set (the entity set that the weak entity set is dependent on)
  - Primary Key of the weak entity set = Discriminator + foreign key

Example – Weak Entity Set



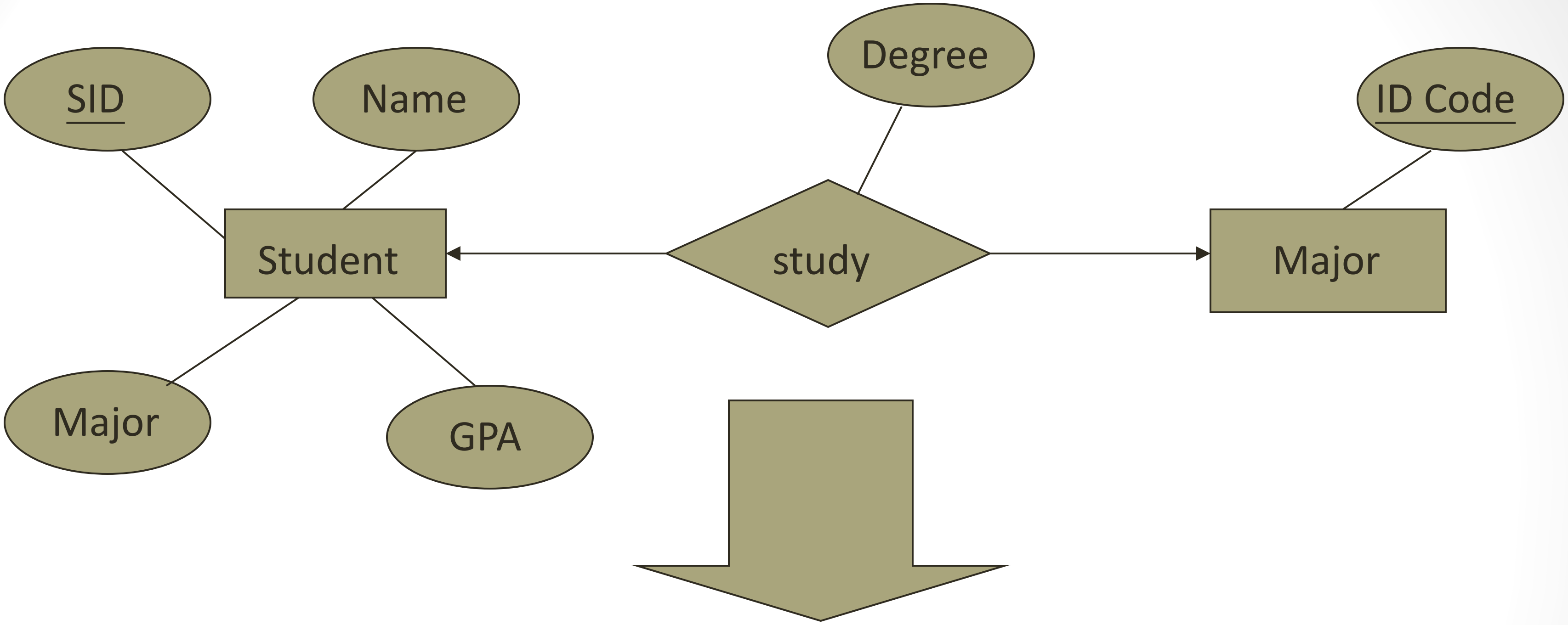
Age	Name	Parent_SID
10	Bart	1234
8	Lisa	5678

* Primary key of *Children* is *Parent_SID* + *Name*

Representation of Relationship Set

- Can be a separate relation
- Can be incorporated into the total participation entity set

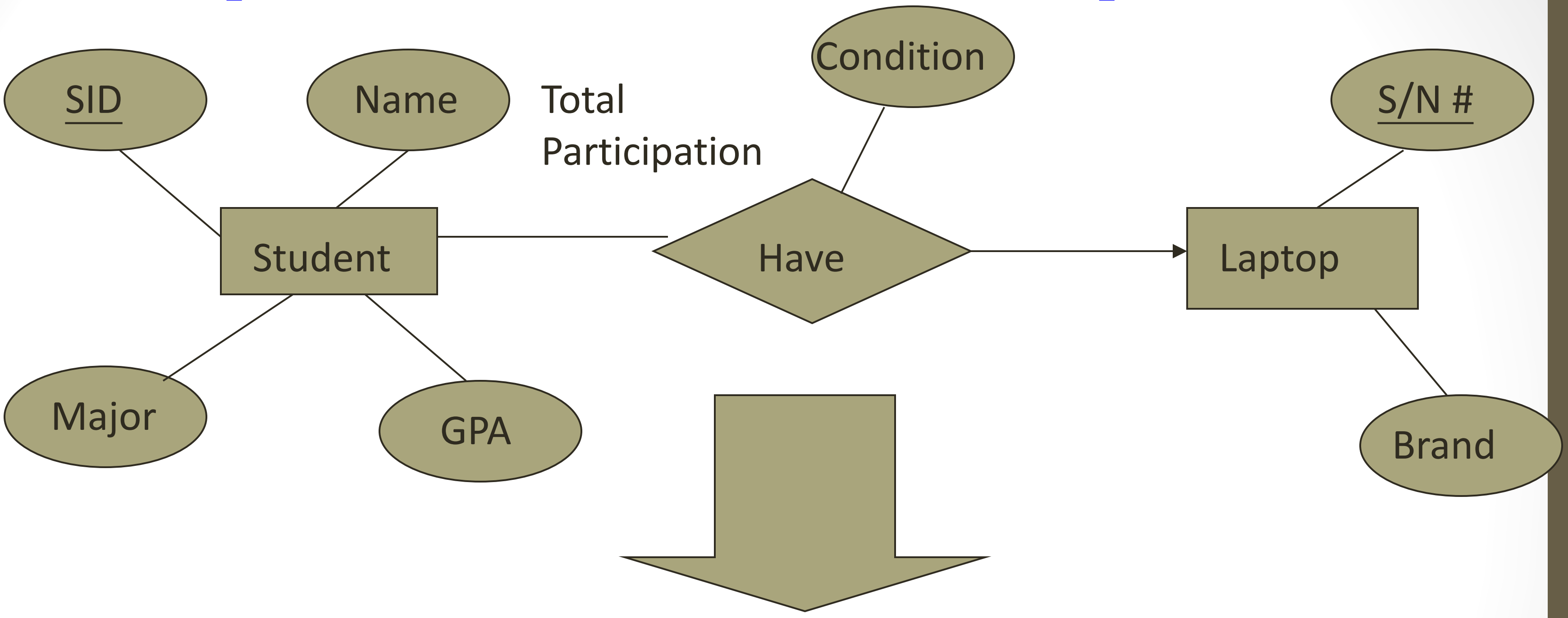
Example – One-to-One Relationship Set



<u>SID</u>	<u>Maj_ID Co</u>	S_Degree
9999	07	1234
8888	05	5678

* Primary key can be either *SID* or Maj_*ID_Co*

Example – One-to-One Relationship Set



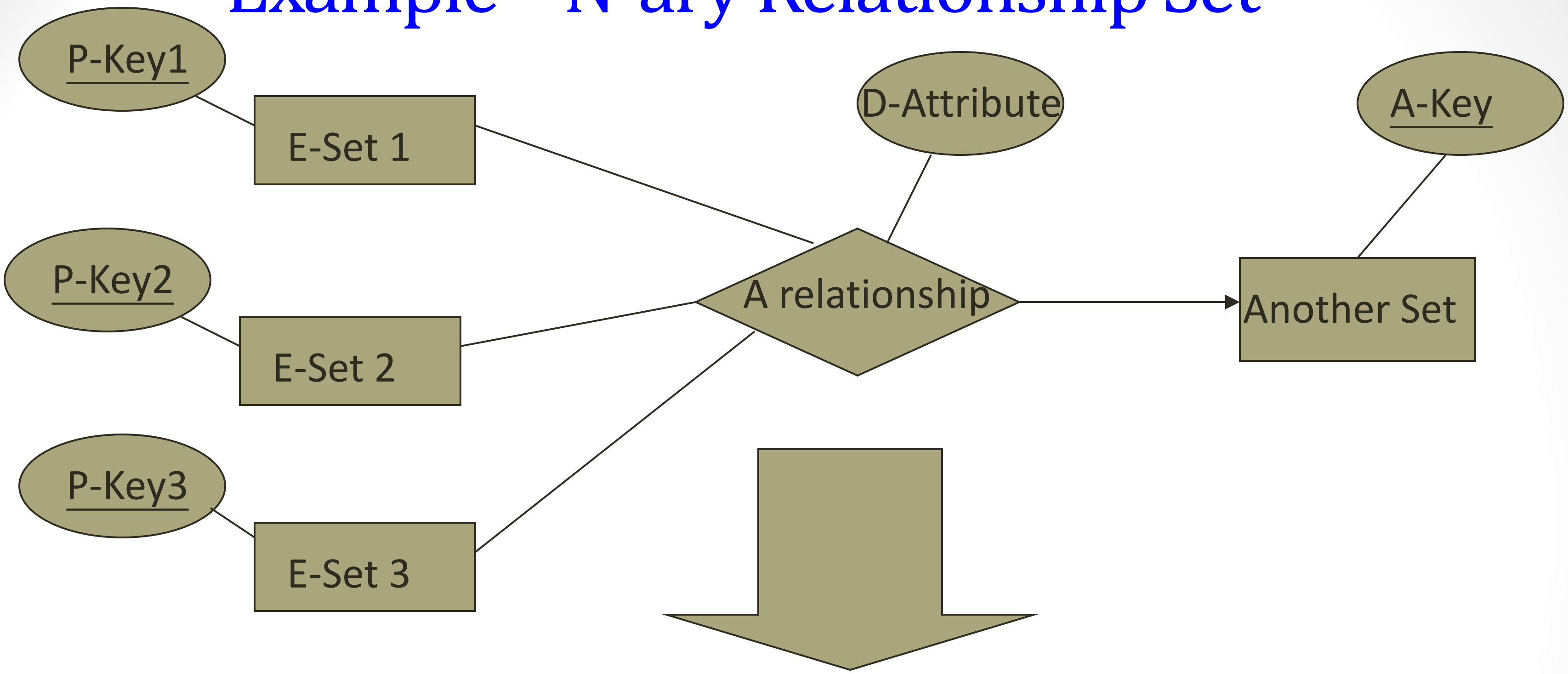
SID	Name	Major	GPA	LP_S/N	Hav_Cond
9999	Bart	Biology	2.0	123-456	Own
8888	Lisa	Physics	4.0	567-890	Loan

* Primary key can be either *SID* or *LP_S/N*

Representing Relationship Set
N-ary Relationship

- Intuitively Simple
  - Build a new table with as many columns as there are attributes for the union of the primary keys of all participating entity sets.
  - Augment additional columns for descriptive attributes of the relationship set (if necessary)
  - The primary key of this table is the union of all primary keys of entity sets that are on “many” side
  - That is it, we are done.

Example – N-ary Relationship Set



P-Key1	P-Key2	P-Key3	A-Key	D-Attribute
9999	8888	7777	6666	Yes
1234	5678	9012	3456	No

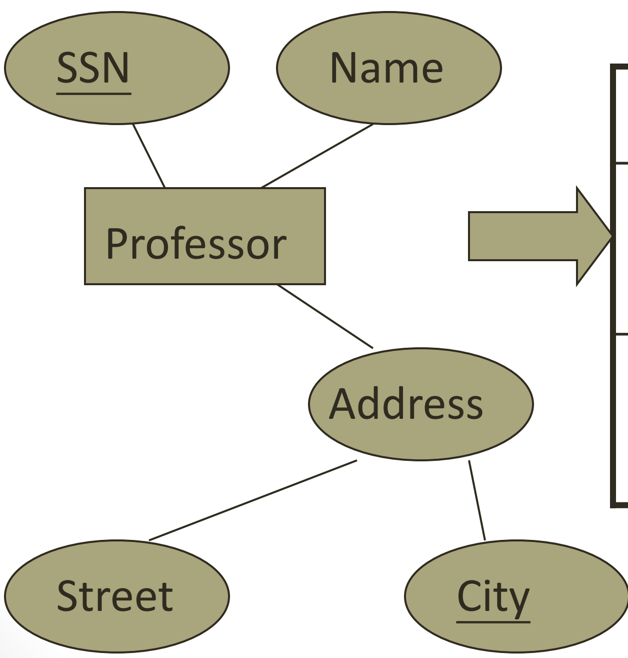
* Primary key of this table is *P-Key1 + P-Key2 + P-Key3*

Representing Relationship Set
Identifying Relationship

- This is what you have to know
  - You DON'T have to build a table/schema for the identifying relationship set once you have built a table/schema for the corresponding weak entity set
  - Reason:
    - A special case of one-to-many with total participation
    - Reduce Redundancy

Representing Composite Attribute

- Relational Model Indivisibility Rule Applies
- One column for each component attribute
- NO column for the composite attribute itself

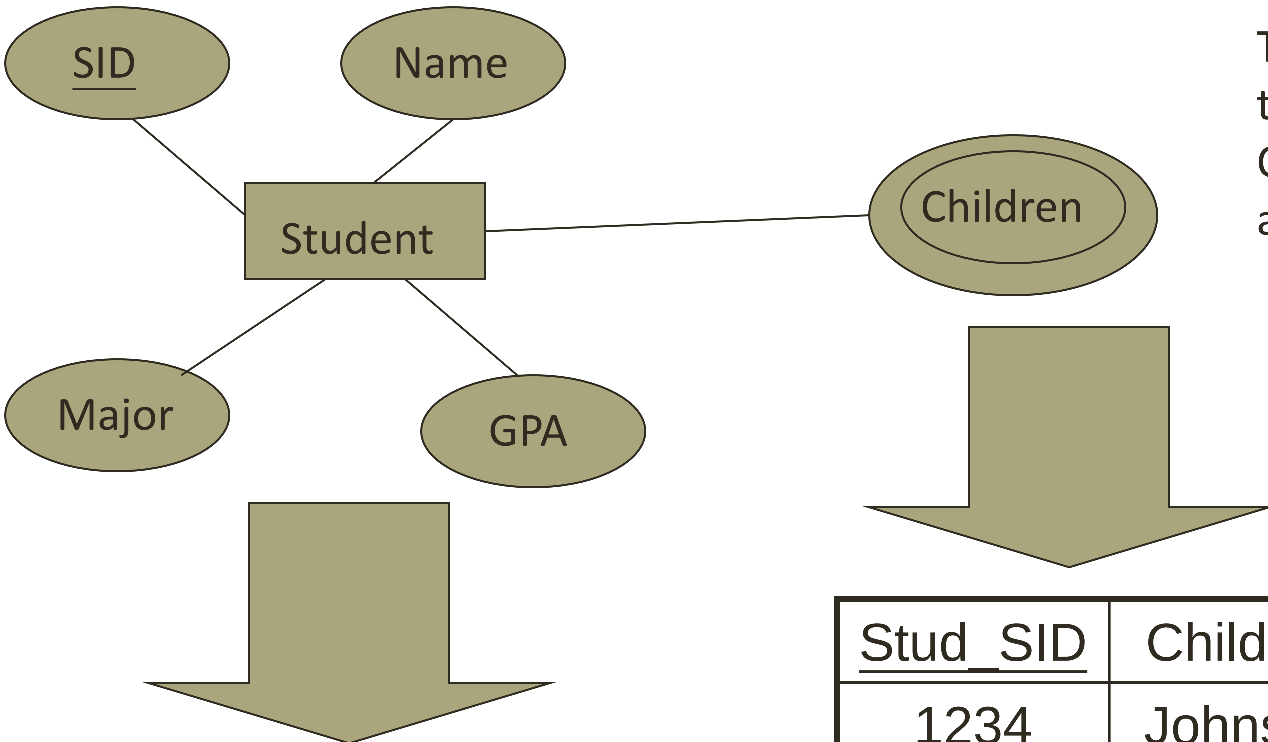


SSN	Name	Street	City
9999	Dr. Smith	50 1st St.	Fake City
8888	Dr. Lee	1 B St.	San Jose

Representing Multivalue Attribute

- For each multivalue attribute in an entity set/relationship set
  - Build a new relation schema with two columns
  - One column for the primary keys of the entity set/relationship set that has the multivalue attribute
  - Another column for the multivalue attributes. Each cell of this column holds only one value. So each value is represented as an unique tuple
  - Primary key for this schema is the union of all attributes

Example – Multivalue attribute



The primary key for this table is Student_SID + Children, the union of all attributes

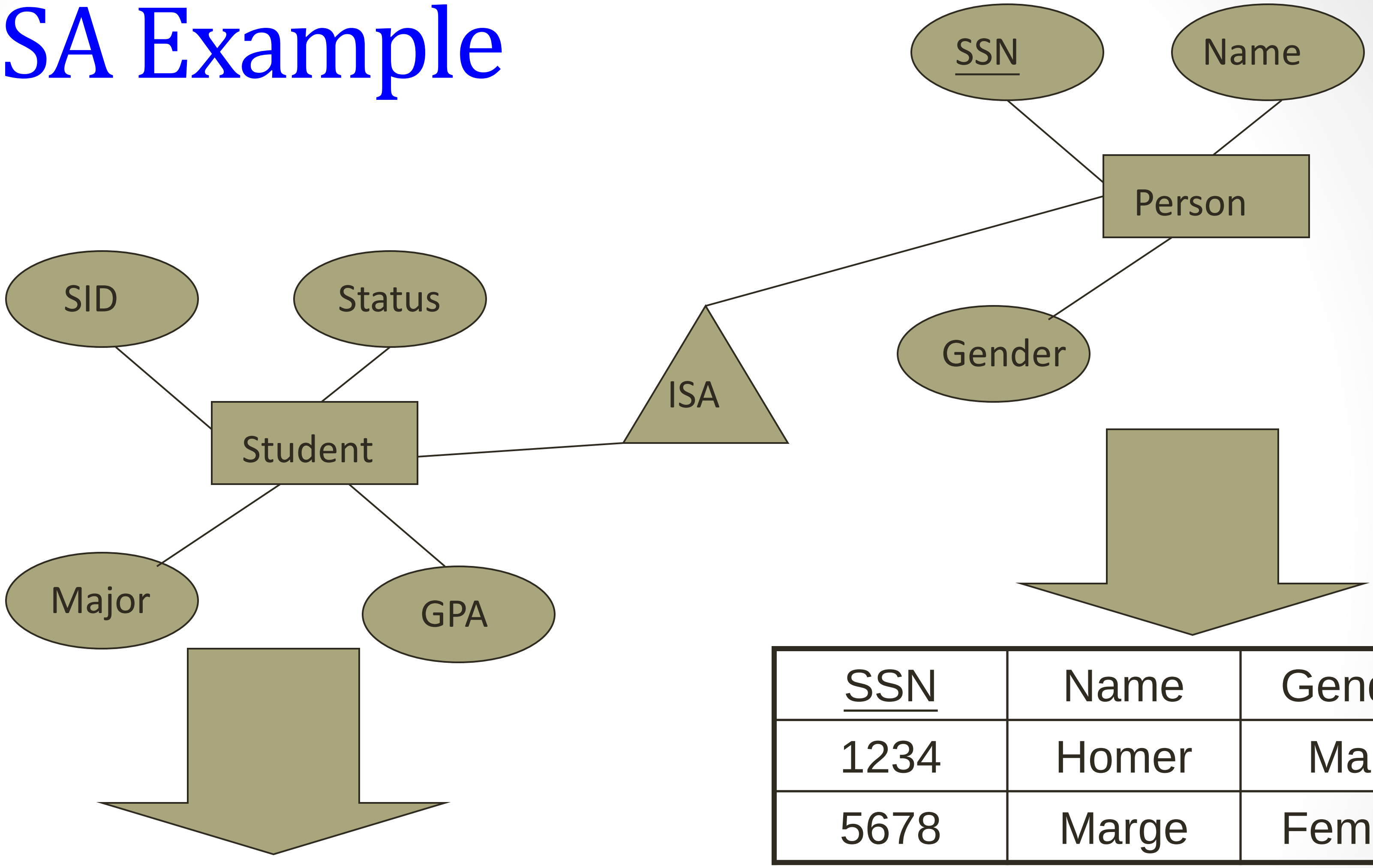
SID	Name	Major	GPA
1234	John	CS	2.8
5678	Homer	EE	3.6

Stud_SID	Children
1234	Johnson
1234	Mary
5678	Bart
5678	Lisa
5678	Maggie

Representing Class Hierarchy

- Two general approaches depending on disjointness and completeness
  - For non-disjoint and/or non-complete class hierarchy:
    - create a table for each super class entity set according to normal entity set translation method.
    - Create a table for each subclass entity set with a column for each of the attributes of that entity set plus one for each attributes of the primary key of the super class entity set
    - This primary key from super class entity set is also used as the primary key for this new table

ISA Example



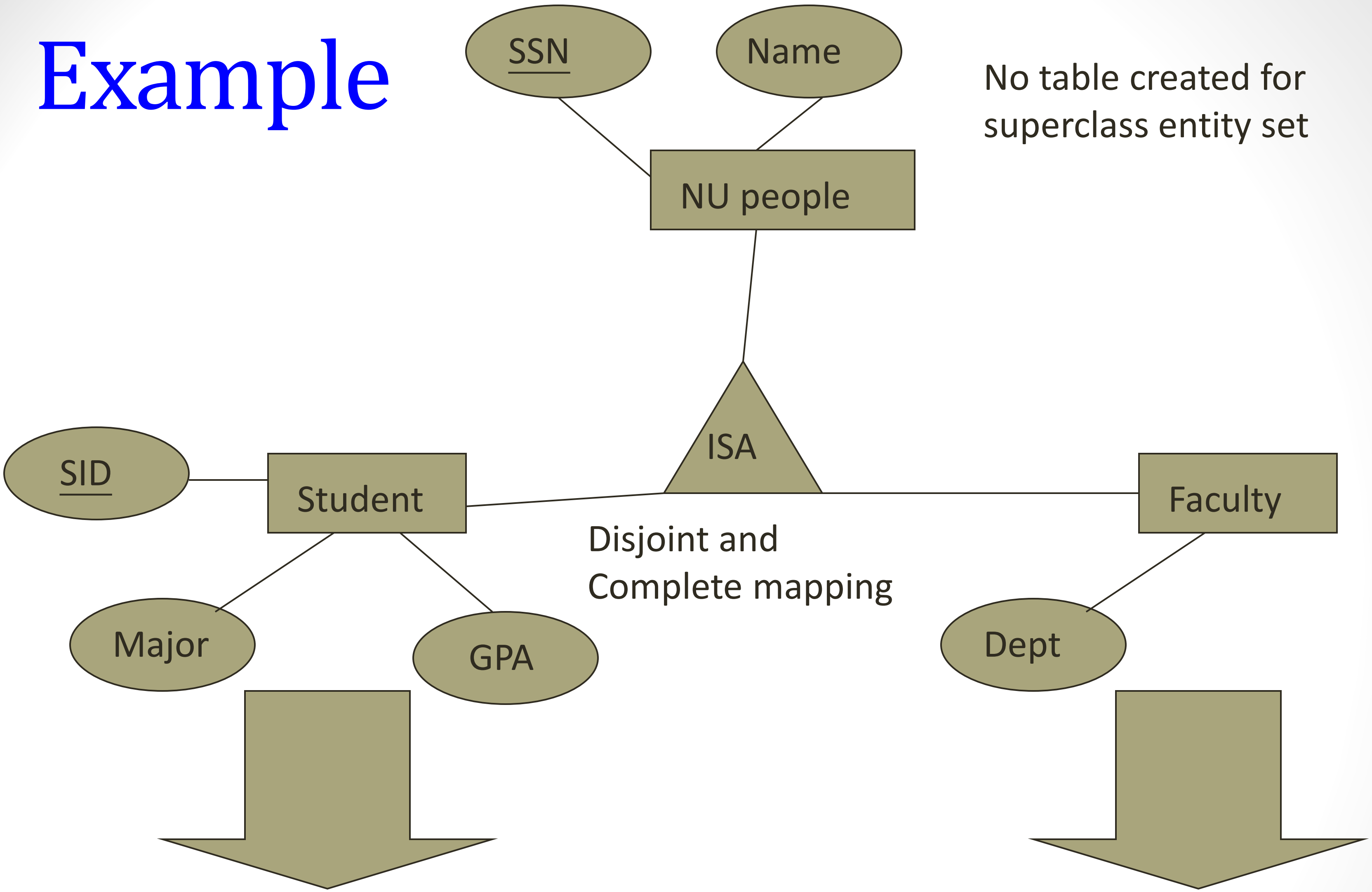
SSN	Name	Gender
1234	Homer	Male
5678	Marge	Female

SSN	SID	Status	Major	GPA
1234	9999	Full	CS	2.8
5678	8888	Part	EE	3.6

Representing Class Hierarchy

- Two general approaches depending on disjointness and completeness
  - For disjoint **AND** complete mapping class hierarchy:
  - DO NOT create a table for the super class entity set
  - Create a table for each subclass entity set include all attributes of that subclass entity set and attributes of the superclass entity set

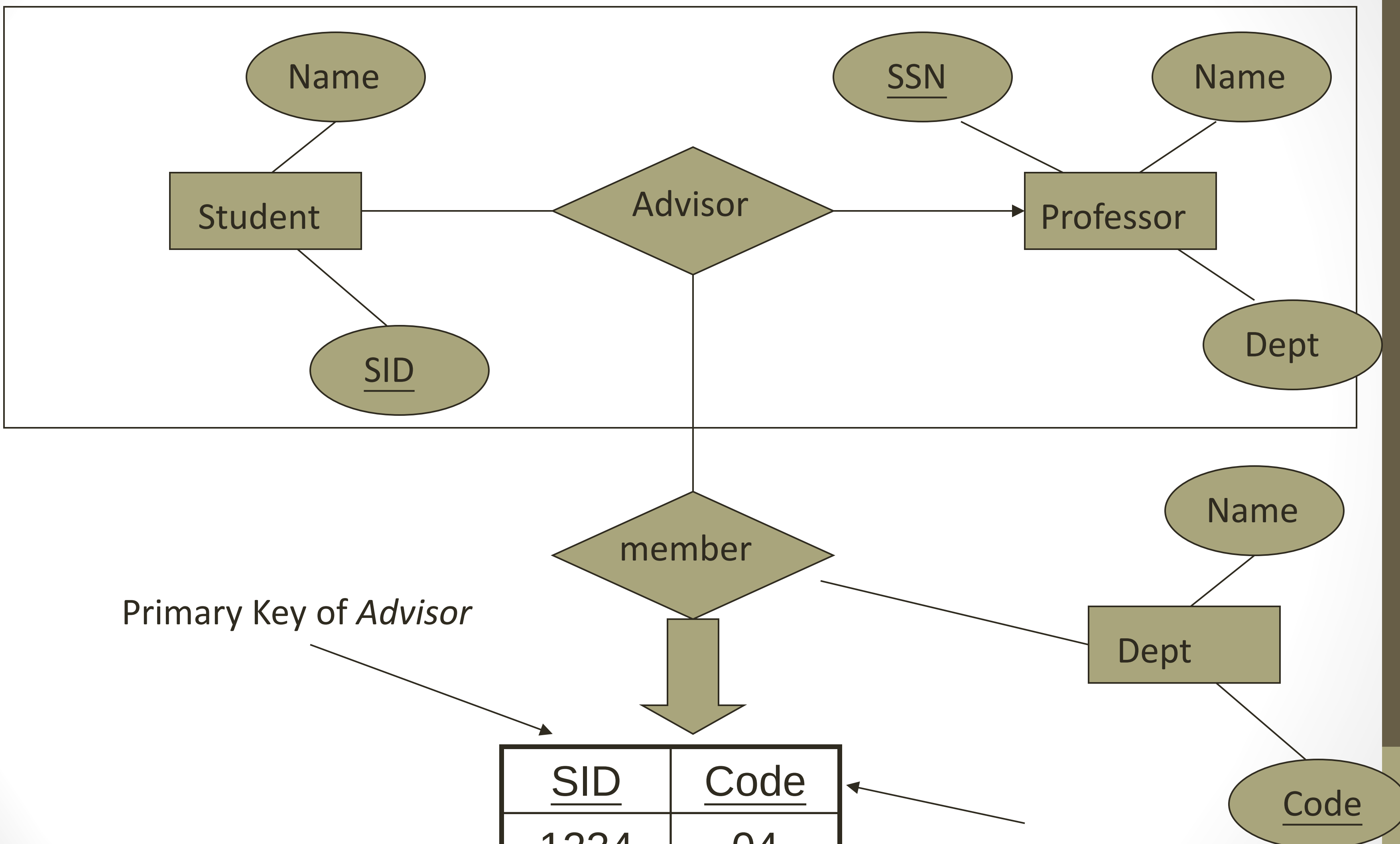
Example



SSN	Name	SID	Major	GPA
1234	John	9999	CS	2.8
5678	Mary	8888	EE	3.6

SSN	Name	Dept
1234	Homer	C.S.
5678	Marge	Math

Representing Aggregation



Primary Key of *Advisor*

<u>SID</u>	<u>Code</u>
1234	04
5678	08

Primary key of *Dept*

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 key (key constraints) and foreign keys (referential contraints)
  - In addition, we always have domain constraints.
- Rules to translate ER to relational model