IIT PALAKKAD

Data Models

Jan-May 2025

Dr. Koninika Pal

What are Data Models

- Describing data, data relationship, data semantics, consistency constraints

- High-level or Conceptual data model
  - Entity-Relationship model
  - Relational data model
  - Object-relational data model
  - Semi-structure model
  - Hierarchical model
  - Network model
- Physical-level data model

IIT PALAKKAD

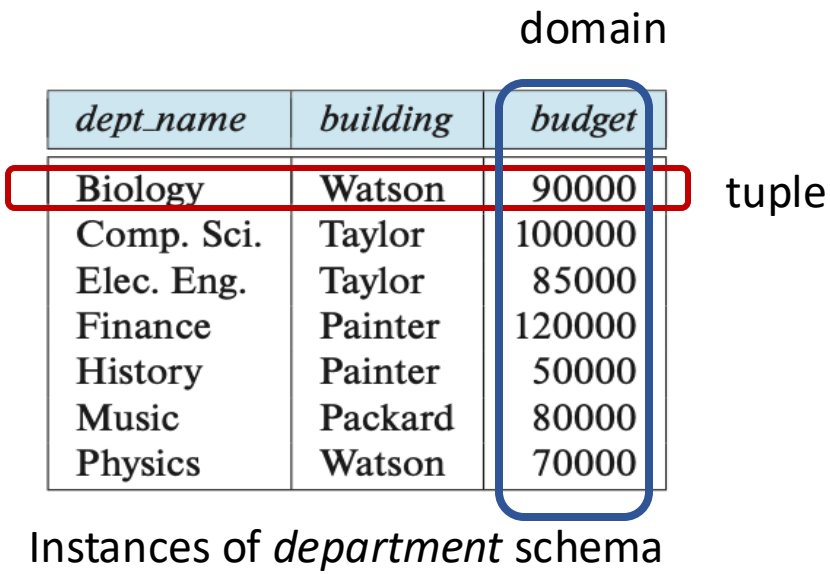
Relational Models

Relational Model

- Collection of relations (two-dimensional array of rows and columns, single valued entries, no duplicate rows)
  - Schema and Instances

    department (dept_name: String, building: String , budget: Integer)
  - Tuple and domain

- Integrity constraints



dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

Instances of *department* schema

Integrity Constraints

- Domain constraints
  - Atomic and null value
  - Unique value

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

- Key Constraints: distinguish tuples, Unique
  - Super key: set of one or multiple attributes that allows to identify the tuple uniquely in the relation , e.g., {dept_name}, {dept_name, building}
  - Candidate key: minimal Super key, e.g., dept_name
  - Primary key: the candidate key chosen by DBA based on infrequent updates on candidate key, e.g., dept_name

Integrity Constraints

- Foreign-key Constraints: check consistency of data

  e.g., *dept_name* referencing *department* relation is the foreign key in *course*

course_id	*title*	*dept_name*	*credits*
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

- General constraints: avoid errors during modification

  e.g., check range of *credits* (*3<=credits <= 5*)

Structural Query Language (SQL)

- Data Definition Language:
  - CREATE, ALTER, DROP, etc.
- Data Manipulation Language:
  - UPDATE, DELETE, INSERT, etc.

course_id	*title*	*dept_name*	*credits*
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

create table *department*
(*dept_name* **varchar** (20),
building **varchar** (15),
budget **numeric** (12,2),
primary key (*dept_name*));

create table *course*
(*course_id* **varchar** (7),
title **varchar** (50),
dept_name **varchar** (20),
credits **numeric** (2,0),
primary key (*course_id*),
foreign key (*dept_name*) **references** *department*);

IC with Structural SQL

- Defining integrity constraints

course_id	*title*	*dept_name*	*credits*
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3

create table *course*
(*course_id* **varchar** (7),
title **varchar** (50). **not null ,**
dept_name **varchar** (20) **not null ,**
credits **numeric** (2,0),
primary key (*course_id*),
foreign key (*dept_name*) **references** *department*);
check (*credits***>=3)**

- Enforce integrity constraints

insert into *course* (*course_id*, *title*, *dept_name*, *credits*)
values (*'CS-304'*, *'Modern Databases'*, *null*, *3*)

Rejected by DBMS

IC with Structural SQL

course_id	*title*	*dept_name*	*credits*
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
DS-204	Machine Learning	Data Sci.	3

Not a primary key in *department*

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

insert into *course* (*course_id, title, dept_name, credits*)
values *('DS-204', 'Machine Learning', Data Sci., 3)*

Rejected by DBMS

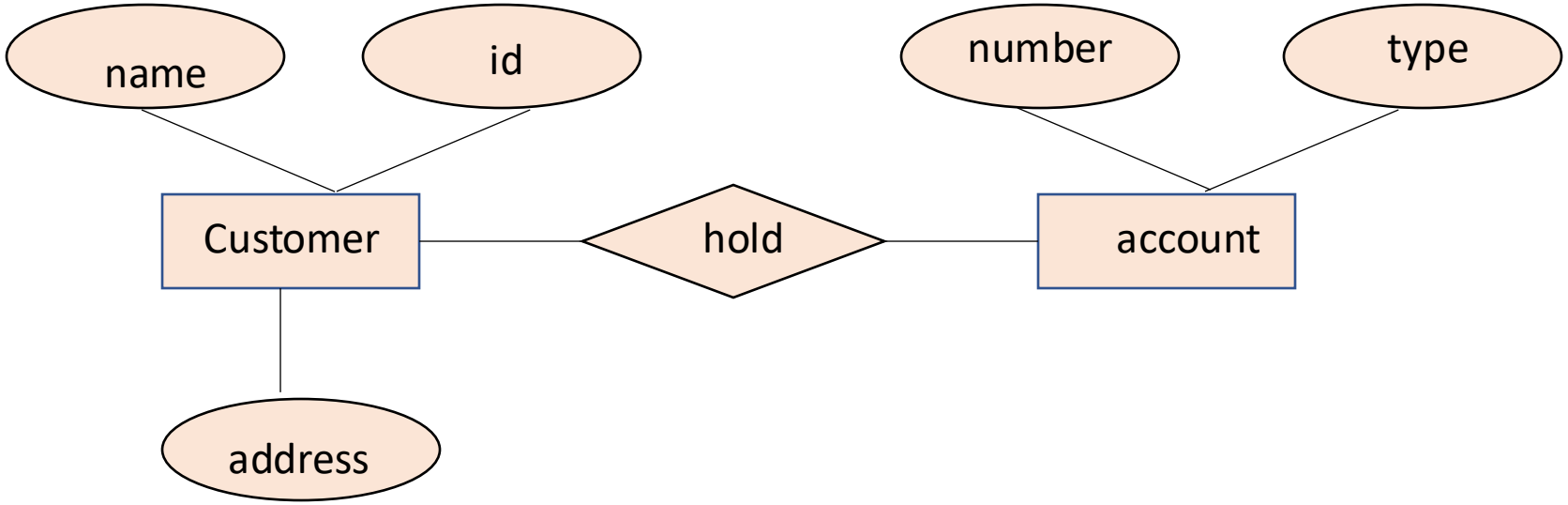
create table *course*
(*course_id* **varchar** (7),
title **varchar** (50),
dept_name **varchar** (20),
credits **numeric** (2,0),
primary key (*course_id*),
foreign key (*dept_name*) **references** *department*);

IIT PALAKKAD

Entity Relationship Models

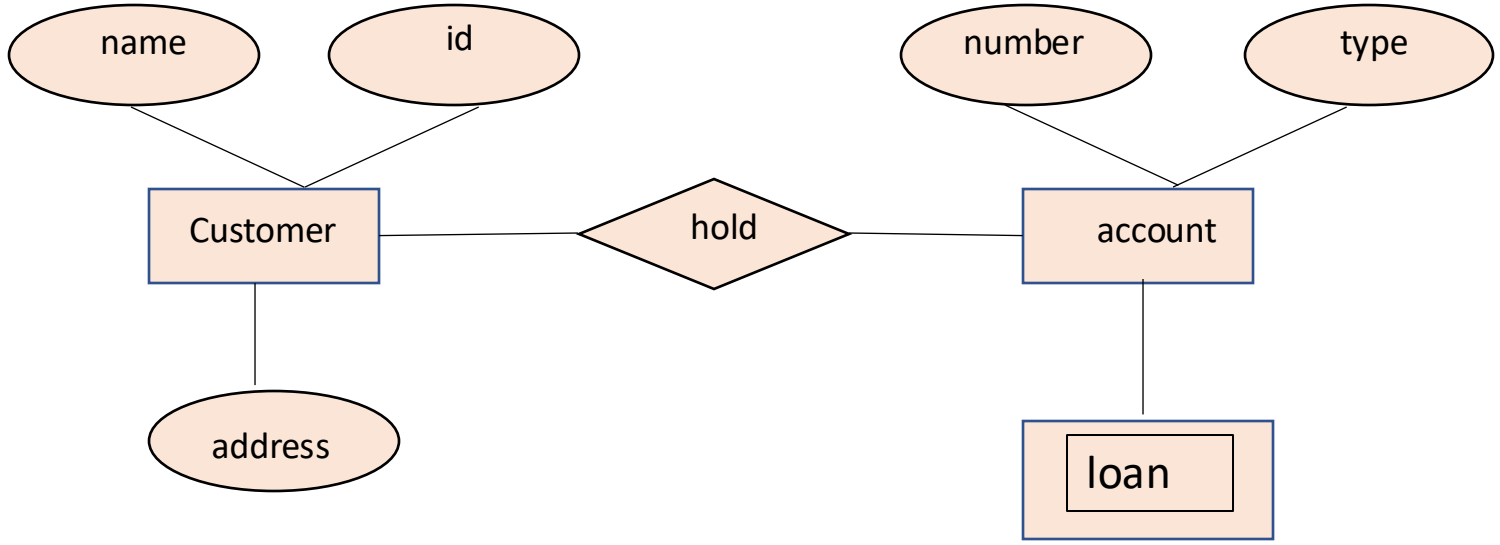
Entity-Relationship Model

- First step in database design
- Logical representation of data
  - ER model - Graphical representation of the model for easy understanding
- Components
  - Entities
  - Relationship



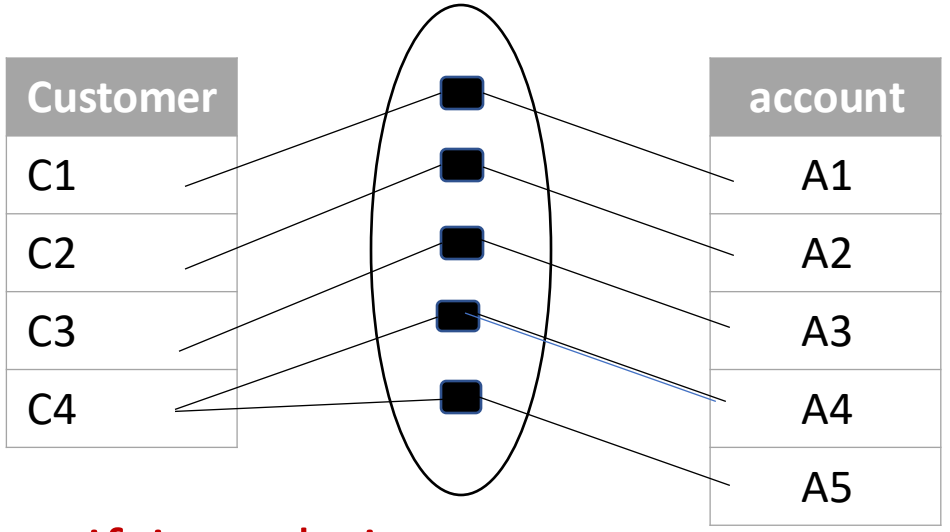
Entities

- Real world objects
- Weak Entities
- Attributes
  - Key attribute
  1. Composite
  2. Multivalued
  3. derived
- Entity set
  - Domain

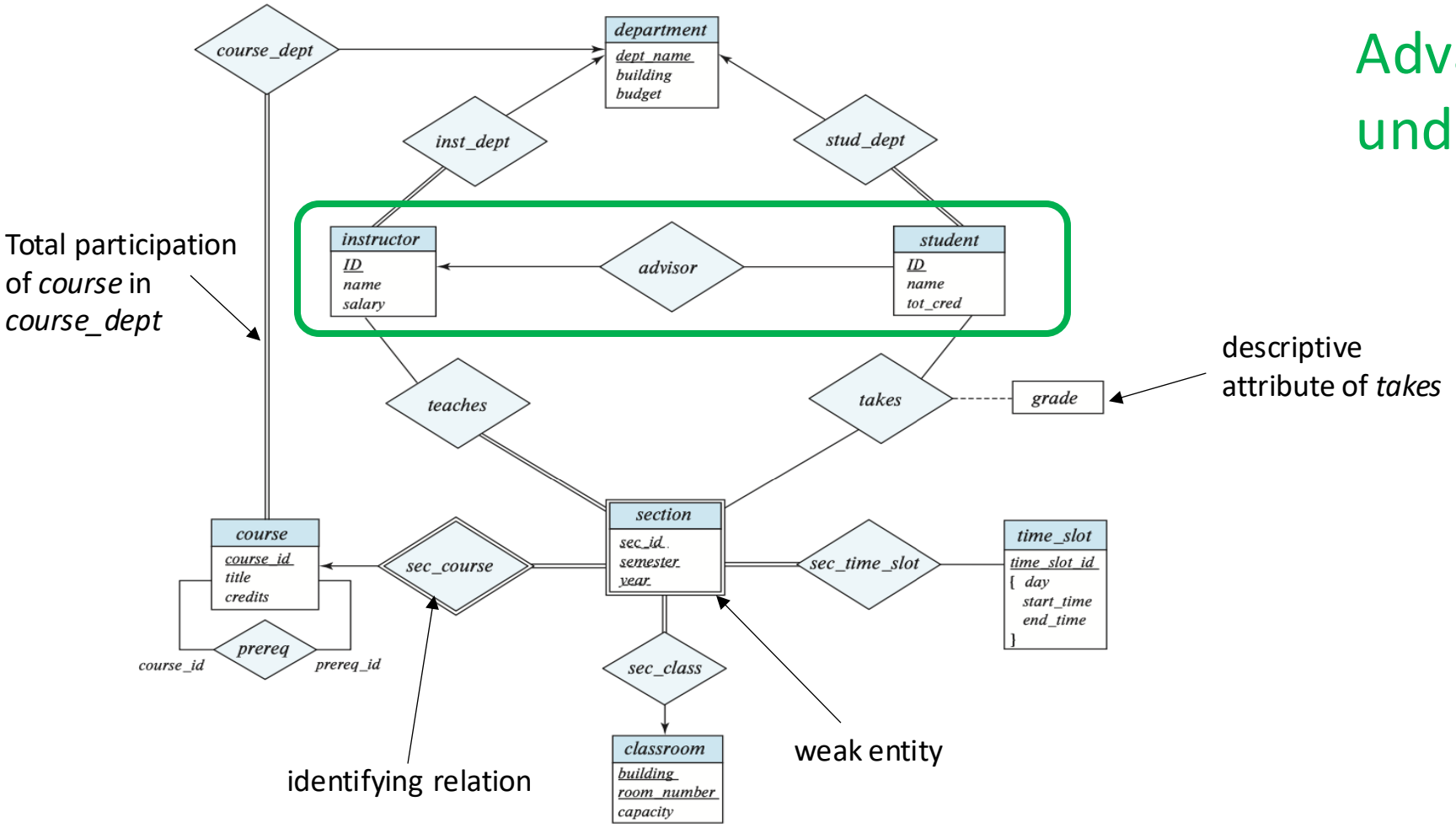


Relationships

- Association among entities
  - Descriptive attribute
- Relationship set
  - Total vs. partial
- Key Constraints
  - Mapping cardinalities
  - Weak entities: Total relationship with identifying relation
- Primary key for relations



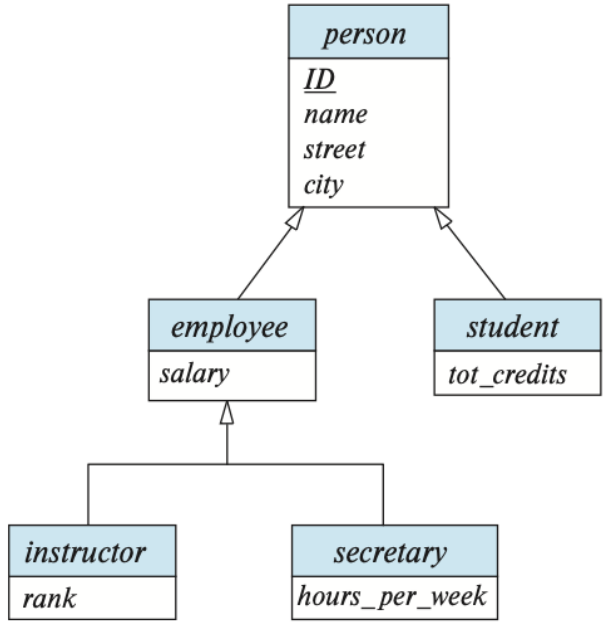
An Example of ER Diagram



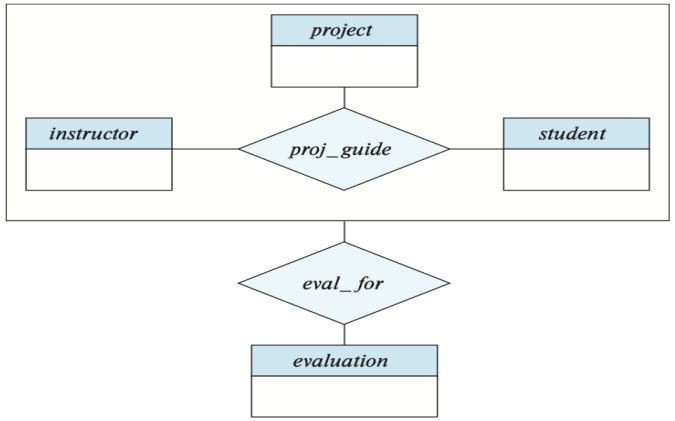
Advance features under class hierarchy

Advanced Features

- Class hierarchy among entities
  - ISA relationship
  - Attribute inheritance
  - Overlapping/disjoint
    - Based on coverage of entities between subclasses
  - Partial/total constraint on specialization
    - Based on coverage of superclass entities among subclasses



- Aggregation:
  - relationships as higher- level entities

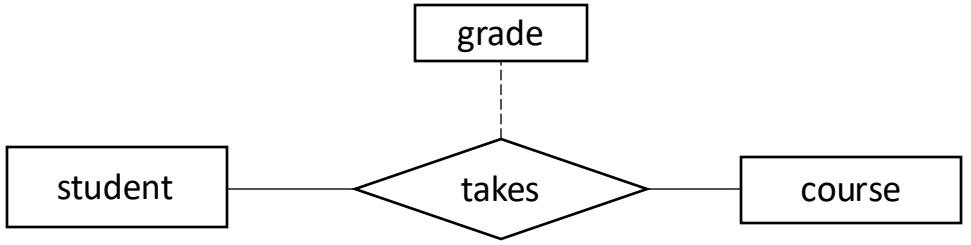


Design Issues with ER Models

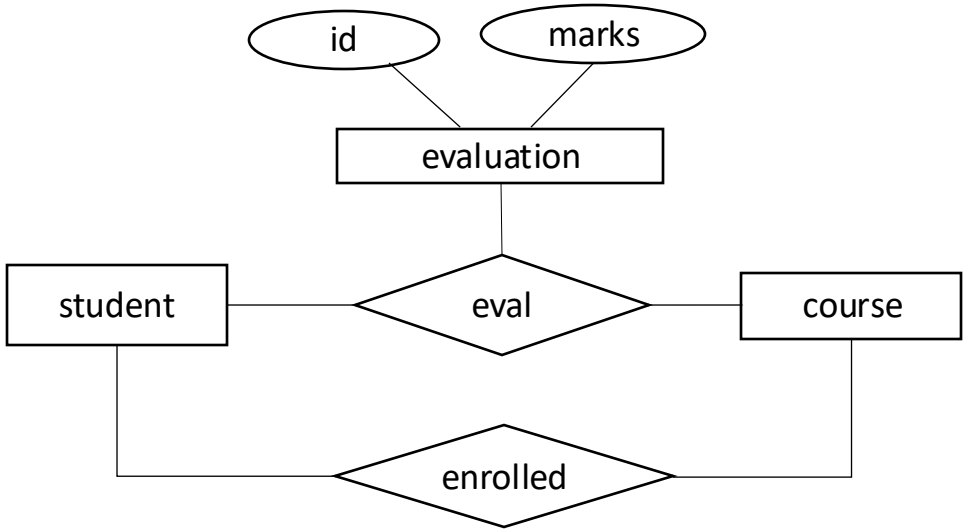
- Entity vs. Attribute

  e.g., *address* as an entity or an attribute of entity person

- Entity vs. Relationship
- Binary vs. Ternary relationship



Vs.

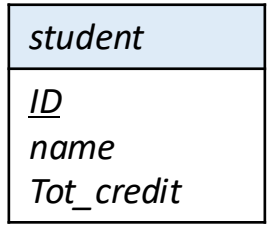


IIT PALAKKAD

Database Design to Relational Models

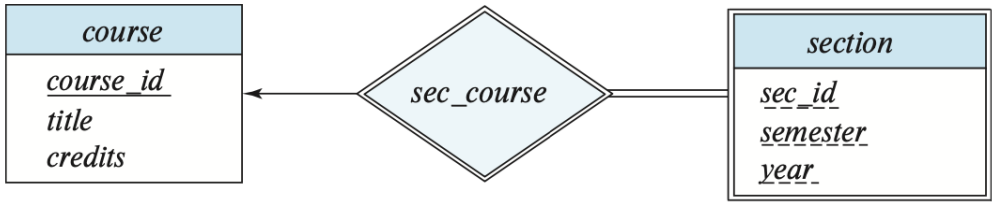
E-R model to Relational Model

- Entity set → table
  - Composite attribute -> new attributes
  - Multivalued attribute -> new relation



ID	name	tot_credit

- Weak entity set → table

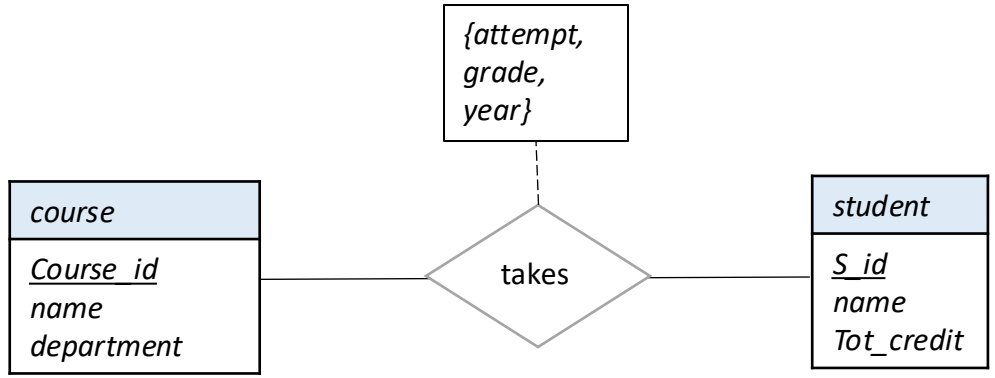


Sec_id	Course_id	semester	year

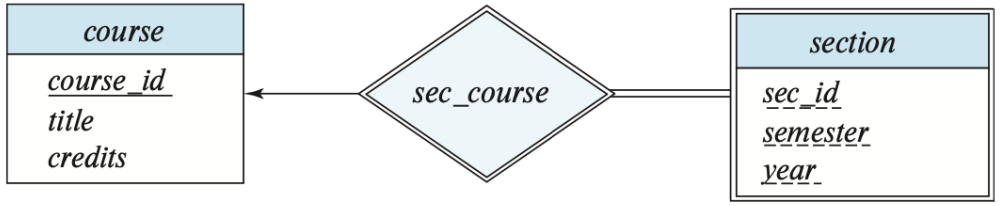
E-R model to Relational Model

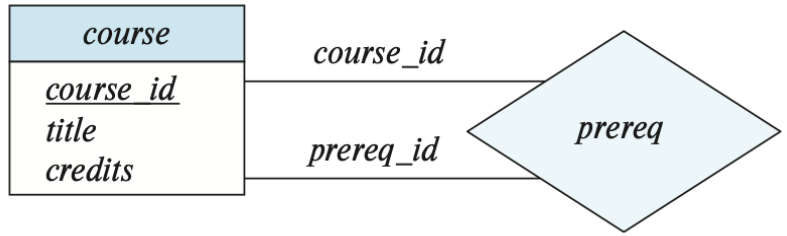
- Relationship → table

s_id	Course_id	attempt	year	grade



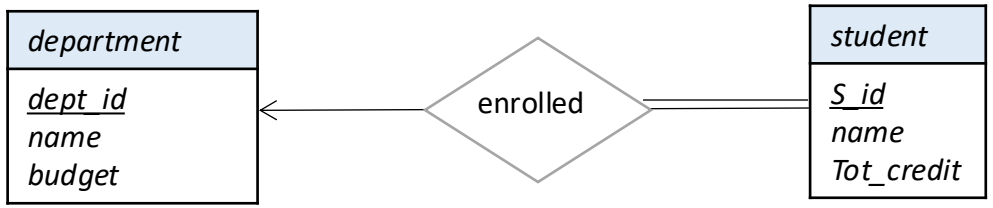
- Redundant relations should be removed





E-R model to Relational Model

- Total participation in relationship
  - many-to-one



- Creating single relation by combining entity-set (with total participation in the relationship) and relationship set

S_id	name	Tot_credit	Dept_id

E-R model to Relational Model

- Class-hierarchy → relational model
  - Each entity → relation
  - Pros: queries for common attribute can be handled by accessing single relation

stuff

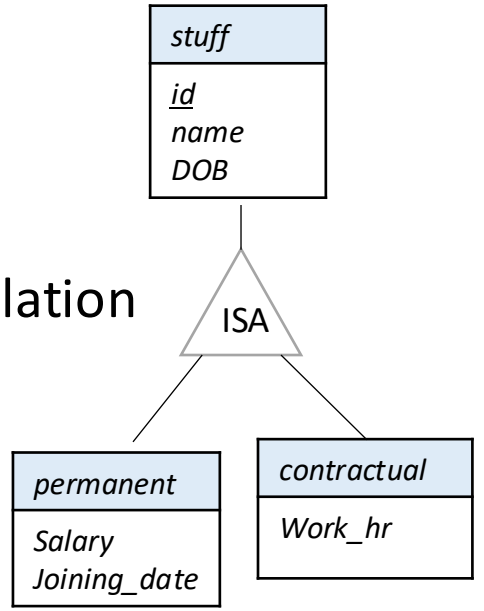
id	name	DOB

permanent

id	salary	Joinig_date

contractual

id	Work_hr



  - Lower-level entities → relation
  - Pros: queries for specific attribute can be handled by accessing single relation

permanent

id	name	DOB	salary	Joining_date

Contractual

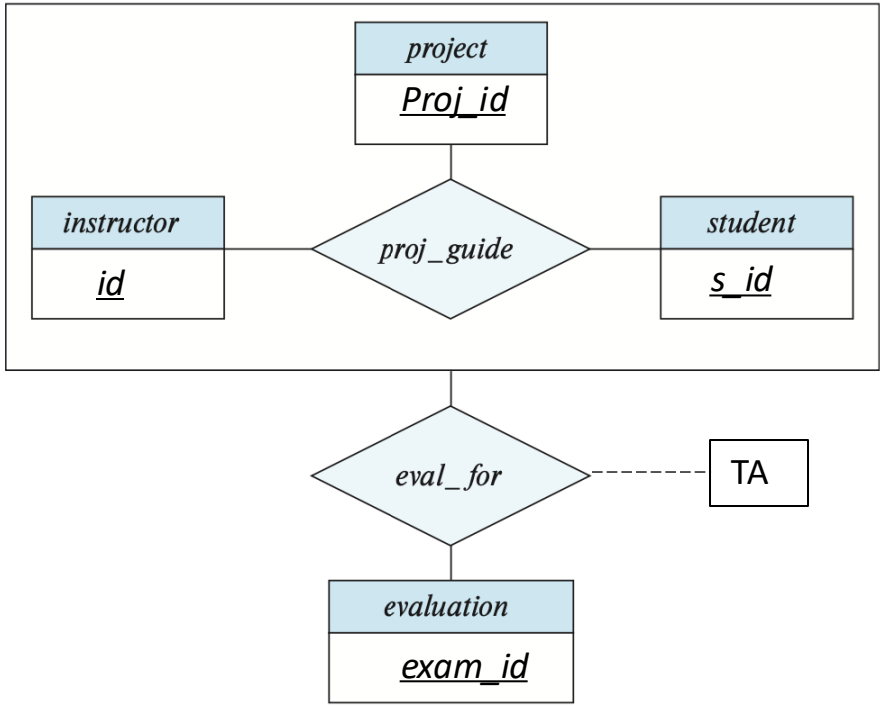
id	name	DOB	work_hr

- Query-dependant choices

E-R model to Relational Model

- ER model with Aggregation → relation
  - Follow the method for relationship → table

Proj_id	id	S_id	Exam_id	TA



- Why do we need relation for aggregated relationship set?
- When can we omit it?