Databases

Database Design Using Entity-Relationship Model

João R. Campos

Bachelor in Informatics Engineering
Department of Informatics Engineering
University of Coimbra
2024/2025

From Previous Lesson(s)…

- Relational Model
  - Relational databases
  - Relation or table, tuple or row, attribute or column
  - Superkeys, candidate keys, primary key and foreign key
  - Integrity restrictions: primary key (entity), foreign key (referential), domain, …
- SQL
  - create table, alter table, and drop table
  - Query the database: select…
  - Cartesian product and joining tables
  - Aggregation functions
  - Modifying the data (insert, update, delete)

Outline

- Database Design Process
- Entity-Relationship Model
  - Entities and Entity Sets
  - Relationships and Relationship Sets
  - Mapping Cardinalities and Participation
  - Removing Redundant Attributes
- From E-R Diagrams to Relational Schemas
  - Obtaining Tables from Entity Sets and Relationship Sets
  - The *onda* Tool
  - Obtaining the SQL DDL commands

Register your presence at UCStudent!

These slides use the following book as reference: Abraham Silberschatz, Henry F. Korth and S. Sudarshan, "Database System Concepts", McGraw-Hill Education, Seventh Edition, 2019.

This class focuses mostly on **Chapter 6**

Database Design

- Creating a database application is a complex task, involving several aspects, such as:
  - Design of the database schema
  - Implementation of the programs that access and update the data
  - Design of a security scheme to control access to data
- For small applications, we may be able to decide directly on the relations that are needed (e.g., the employee database seen before)
- For complex applications, we need to follow a process in order to obtain the best model relational schema possible

Database Design Process

- Characterize the data needs of the prospective database users
- Conceptual design
  - Choose a modeling approach (e.g. E-R model, normalization) and apply the concepts of the chosen data model
  - Translate requirements into a conceptual schema (e.g. E-R diagram)
- Specification of functional requirements
  - Conceptual schema can be used to extract the functional requirements
  - Describe the kinds of operations (or transactions) that will be performed on the data
- Implementation of the database
  - Logical design: from the conceptual schema to the relational data model
  - Physical design: decision related to the the data model (e.g., indexes needed)

Design Alternatives

- There may be several design alternatives for the same problem
- We must ensure that we avoid two major pitfalls:
  - Redundancy: a bad design may result in repeated information
  - Incompleteness: a bad design may make certain aspects of the enterprise difficult or impossible to model
- Avoiding bad designs is not enough!
  - There may be a number of good designs from which we must choose

Why is this a problem?

Design Approaches

- Entity-Relationship model
  - Models an organization as a collection of entities and relationships
    - Entity: a “thing” or “object” in the enterprise that is distinguishable from other objects, and is described by a set of attributes
    - Relationship: an association among several entities
  - Represented diagrammatically by an entity-relationship diagram
- Normalization theory
  - Formalize what designs are bad, and test for them

Databases

Entities and Entity Sets
Relationships and Relationship Sets
Mapping Cardinalities and Participation
Removing Redundant Attributes

ENTITY-RELATIONSHIP MODEL

E-R Data Model

- Developed to facilitate database design by allowing specification of a schema that represents the overall logical structure of a database
- The E-R model is very useful in mapping real-world meanings and interactions onto a conceptual schema
- Employs three basic concepts:
  - Entity sets
  - Relationship sets
  - Attributes
- Has an associated diagrammatic representation, the E-R diagram
- Many tools based on the E-R model are available

Entities and Entity Sets

- An entity is a "thing" or "object" in the real world that is distinguishable from all other objects
  - e.g., each person in a university is an entity
- An entity has a set of properties, and the values for some set of properties must uniquely identify an entity
  - e.g., a person may have a person *id* property whose value uniquely identifies that person
    - The value 2018280021 for person *id* would uniquely identify one particular person in the university
- An entity set is a set of entities of the same type that share the same properties or attributes
  - The set of all people who are instructors can be defined as the entity set instructor

Attributes and Values

- Entity sets do not need to be disjoint
  - e.g., it is possible to define the entity set person consisting of all people in a university
    - A person entity may be an instructor entity, a student entity, both, or neither
- An entity is represented by a set of attributes
  - Descriptive properties possessed by each member of an entity set
  - e.g., possible attributes of the instructor entity set are *ID*, *name*, *dept_name*, and *salary*
  - Attributes may be simple or complex (composed by sub-attributes)
- Each entity has a value for each of its attributes
  - e.g., a particular instructor entity may have the value 12121 for ID, the value Wu for name, the value Finance for dept name, and the value 90000 for salary

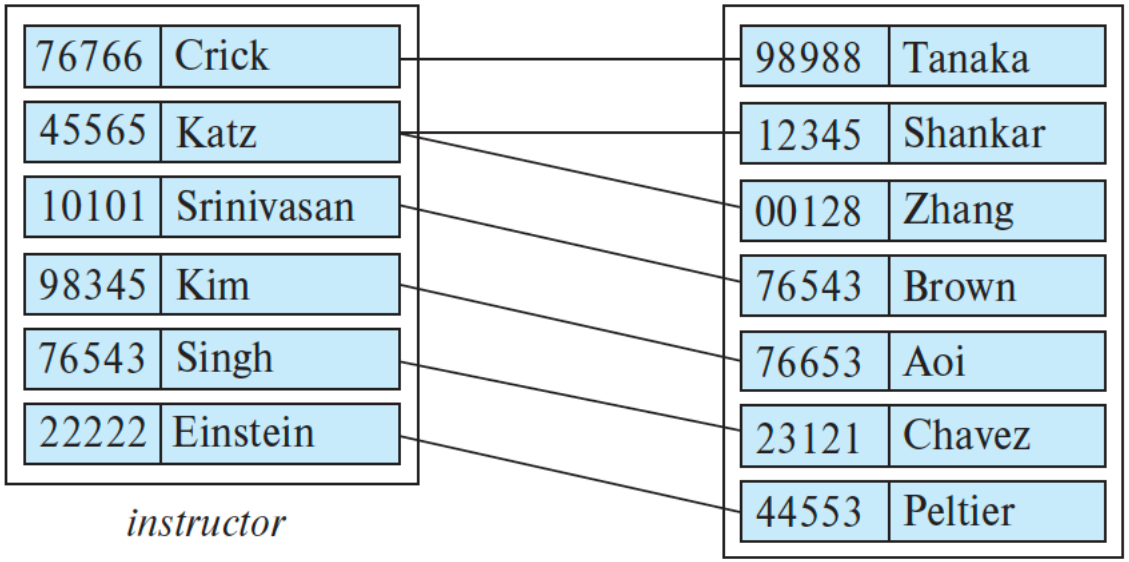
Entity Sets in the E-R Diagram…

instructor		
ID	VChr	PK
name	VChr	
salary	Number	

student		
ID	VChr	PK
name	VChr	
tot_cred	Number	

Relationships and Relationship Sets

- A relationship is an association among several entities
  - e.g., we can define a relationship *advisor* that associates instructor Katz with student Shankar, specifying that Katz is an advisor to student Shankar
- A relationship set is a set of relationships of the same type
  - Consider entity sets *instructor* and *student*: the relationship set *advisor* denotes the associations between students and the instructors who act as their advisors

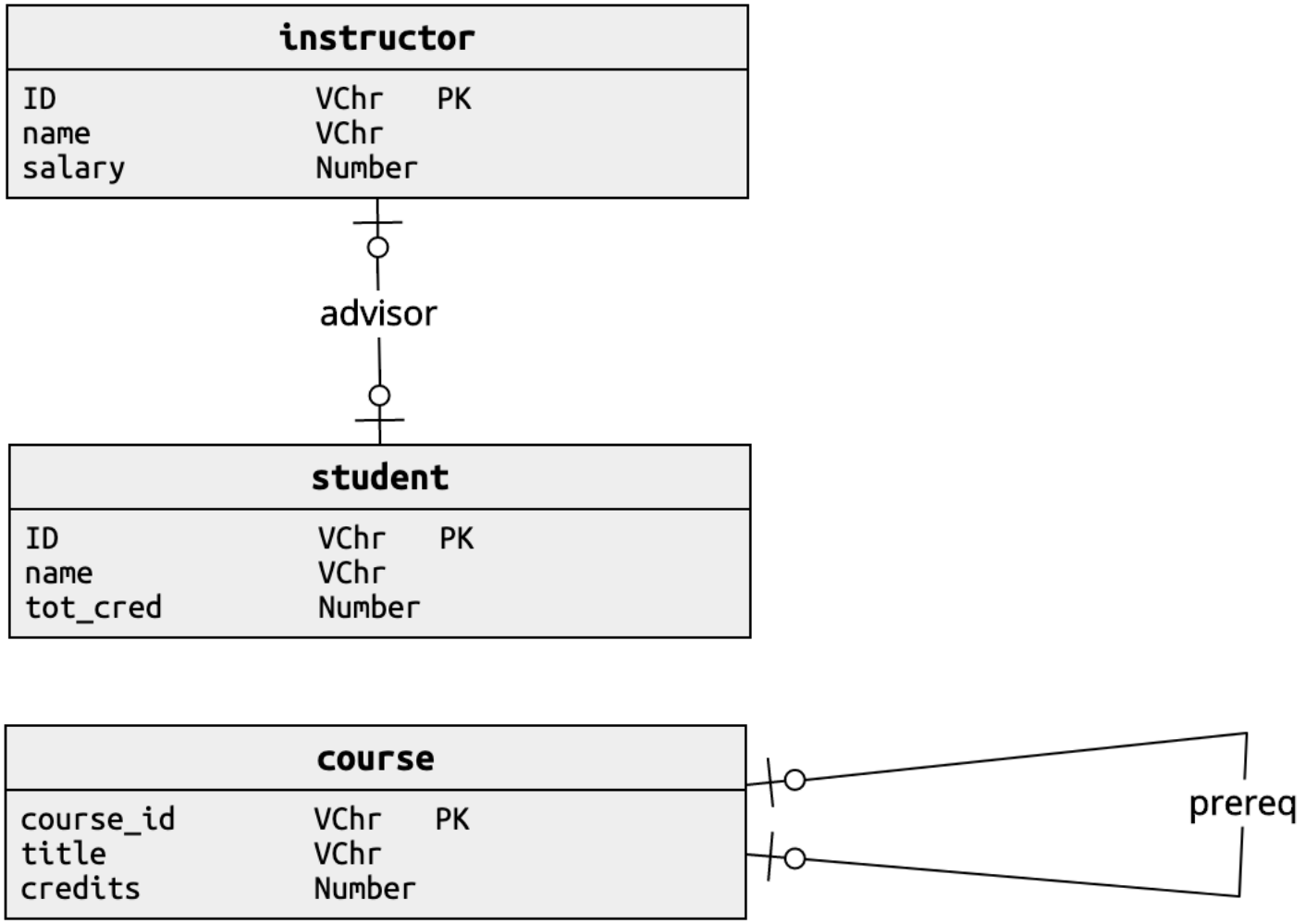


Source: A. Silberschatz, H. F. Korth and S. Sudarshan, "Database System Concepts", McGraw-Hill Education, Seventh Edition, 2019.

Relationships and Relationship Sets

- A relationship instance represents an association between the named entities in the real-world organization that is being modeled
  - e.g., the individual *instructor* entity Katz and the *student* entity Shankar participate in a relationship instance of *advisor*
- There may be several relationship sets between two entity sets
  - e.g., besides advising a student in the context of an internship, an instructor may also act as a tutor in a more general context
    - Relationship set *tutor* is thus different from relationship set *advisor*
- Relationship sets may be recursive when the same entity set participates more than once in the same relationship set

Relationship Sets in the E-R Diagram…

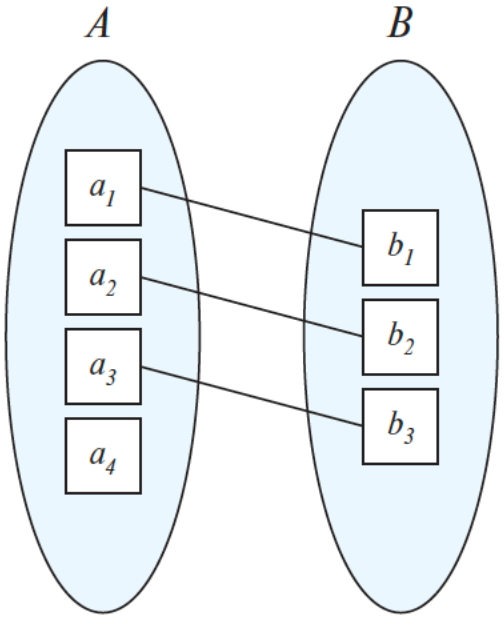


Attributes

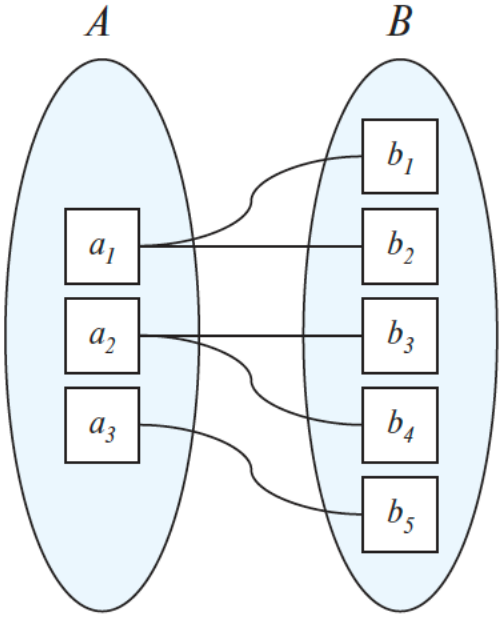
- For each attribute, there is a set of permitted values, called the domain, or value set, of that attribute
  - The domain of attribute *course_id* might be the set of all text strings of a certain length
  - The domain of attribute semester might be strings from the set: *{Fall, Winter, Spring, Summer}*
- An attribute can be characterized by the following attribute types:
  - Simple attributes (e.g., *student_ID*) and composite attributes (e.g., *address*)
  - Single-valued (e.g., *student_ID*) and multivalued (e.g., *dependent_names*)
  - Derived attributes
    - e.g. age can be derived from the date of birth; should it be stored?
- An attribute takes a null value when an entity does not have a value for it

Mapping Cardinalities

- Mapping cardinalities express the number of entities to which another entity can be associated via a relationship set
- For a binary relationship set *R* between entity sets *A* and *B*, the mapping cardinality must be one of the following:
  - One-to-one
  - One-to-many
  - Many-to-one
  - Many-to-many



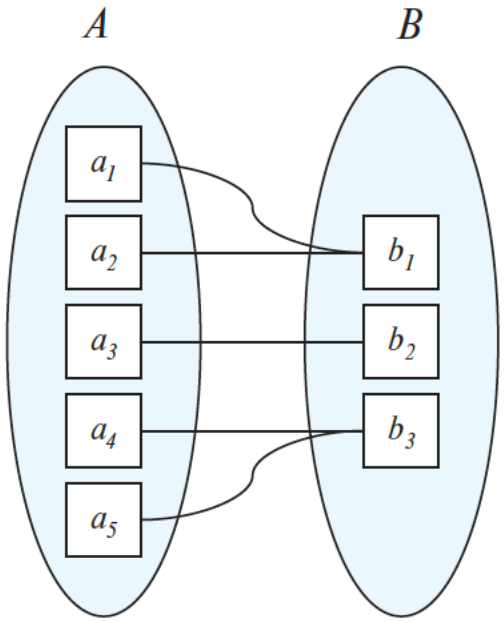
one instructor
one student



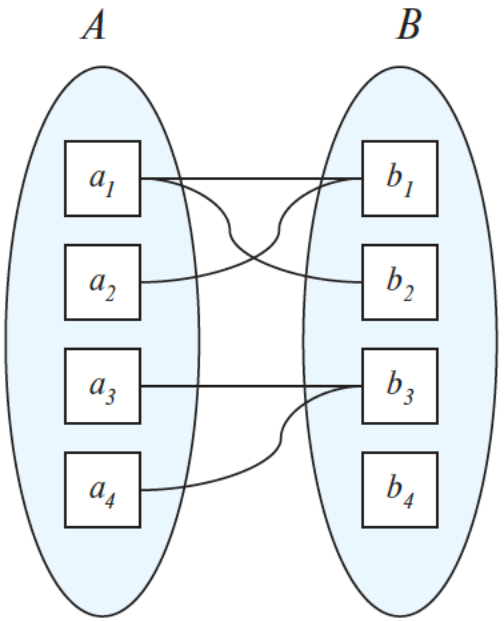
one instructor
many students

Source: A. Silberschatz, H. F. Korth and S. Sudarshan, "Database System Concepts", McGraw-Hill Education, Seventh Edition, 2019.

Mapping Cardinalities



many instructors
one student

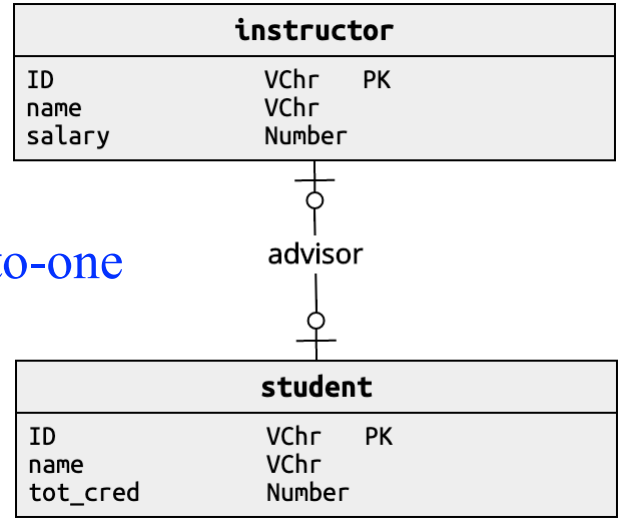


many instructors
many students

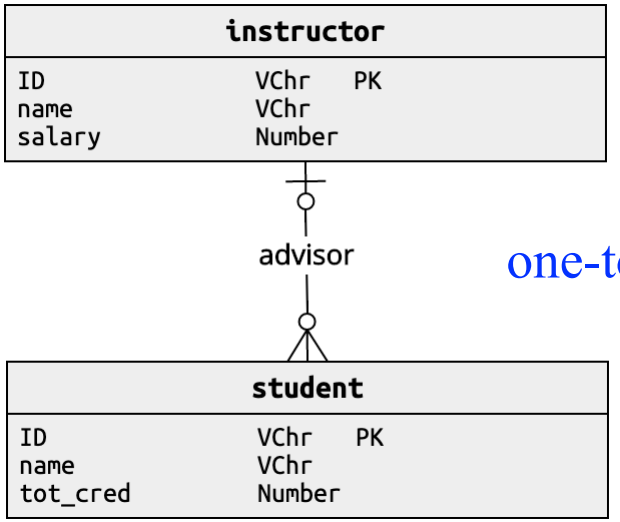
Source: A. Silberschatz, H. F. Korth and S. Sudarshan, "Database System Concepts", McGraw-Hill Education, Seventh Edition, 2019.

Cardinalities in the E-R Diagram…

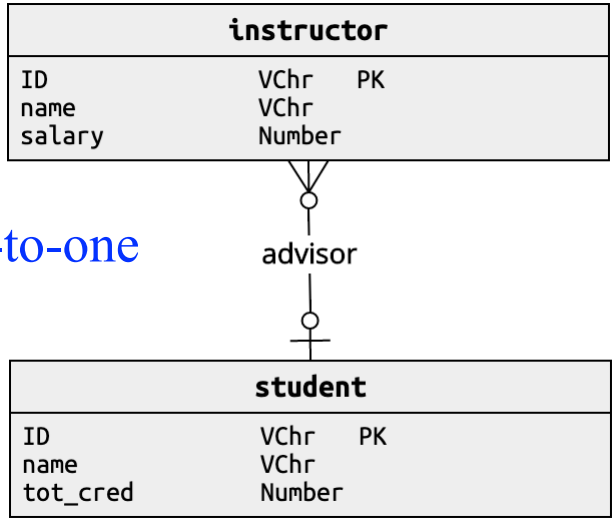
one-to-one



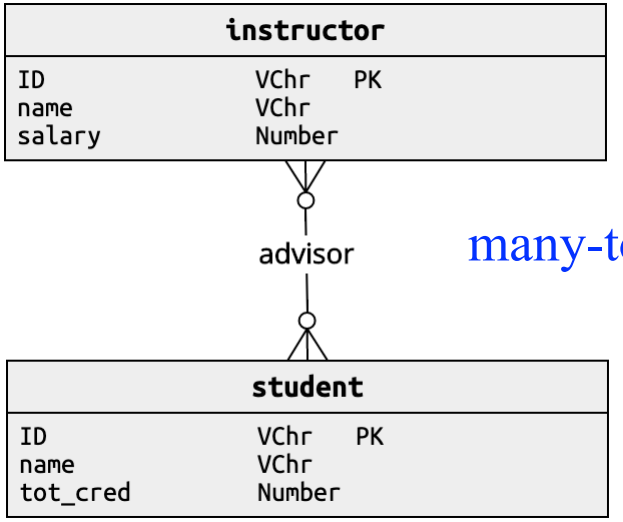
one-to-many



many-to-one

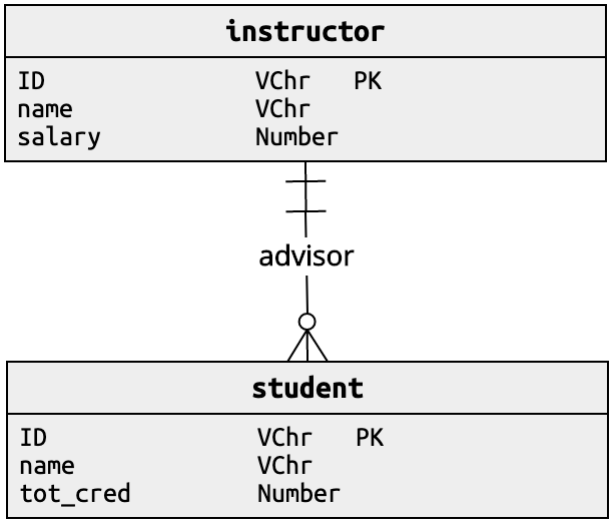


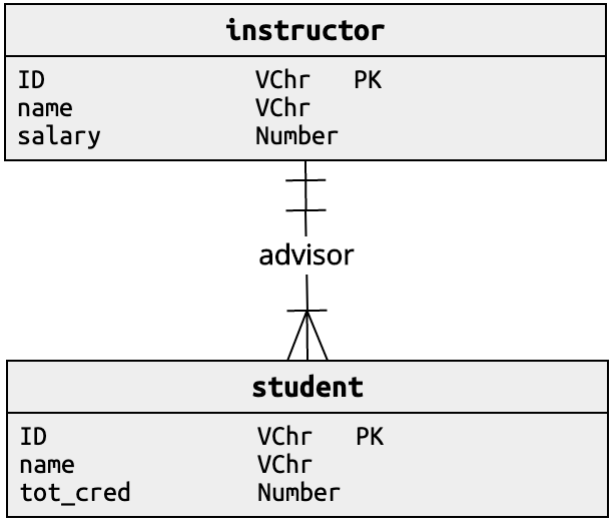
many-to-many



Total and Partial Participation

- Total participation: every entity in the entity set *E* must participate in at least one relationship in relationship set *R*
- Partial participation: if it is possible that some entities in *E* do not participate in relationships in *R*





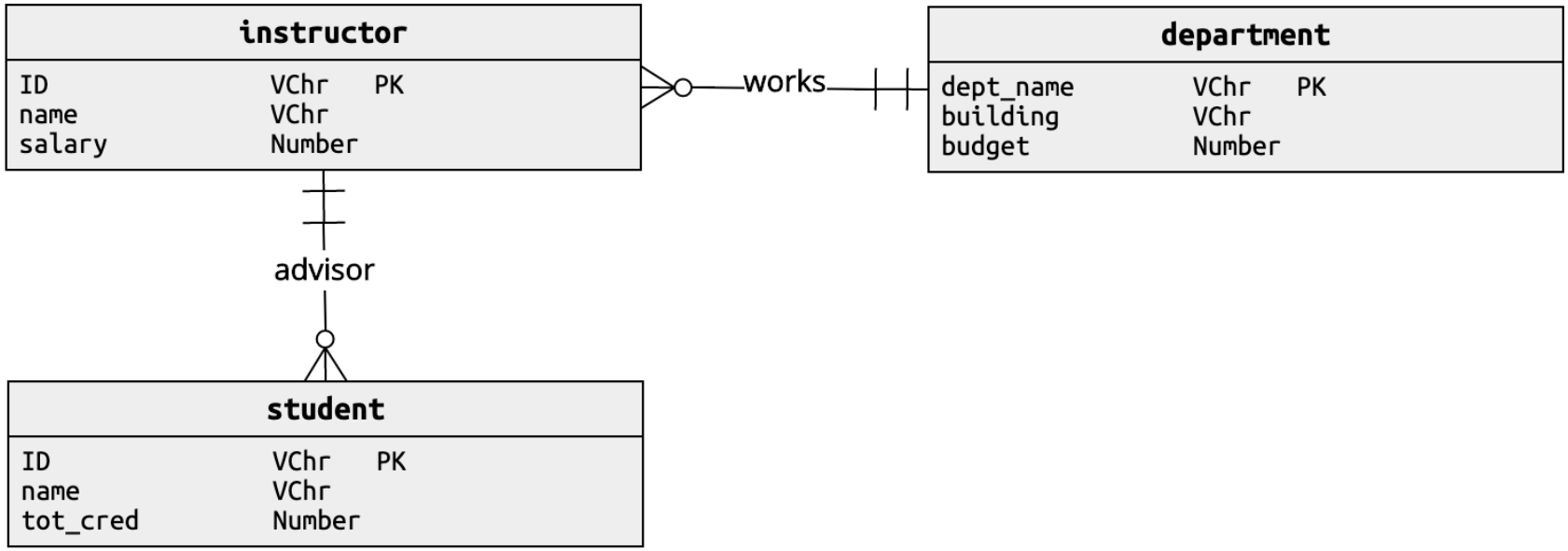
Defining the Primary Key

- The primary key is a set of attributes whose values univocally allow identifying a specific entity in the context of the organization
  - e.g., no two students have the same *student id* at the university
- Primary key and candidate key concepts from relations apply here in the same way
- How to select the primary key?

- Identical to the principles of relational PK discussed in previous class

Multiple Binary Relationships

- The same entity set may have several relationships sets with several other entity sets
  - e.g., entity set *A* may have a relationship set R_1 with entity set *B* and a relationship R_2 with entity set *C*



Removing Redundant Attributes

- A good E-R design does not include redundant attributes

instructor		
ID	VChr	PK
name	VChr	
dept_name	VChr	
salary	Number	

Removing Redundant Attributes

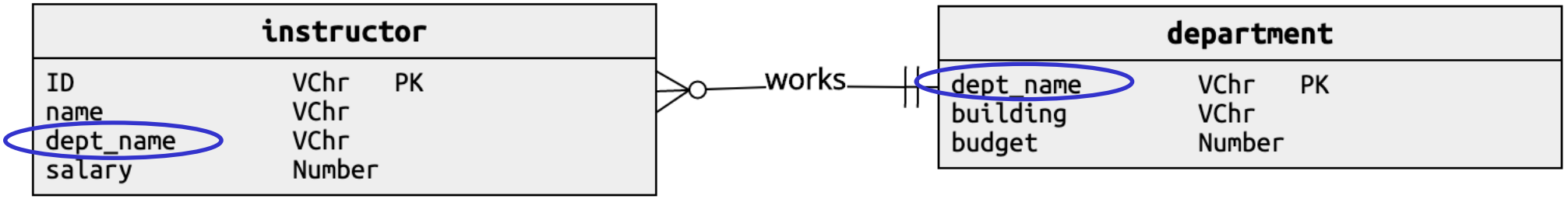
- A good E-R design does not include redundant attributes

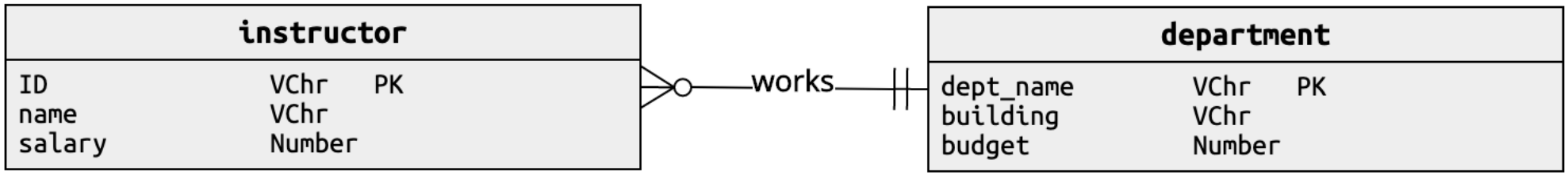
instructor		
ID	VChr	PK
name	VChr	
dept_name	VChr	
salary	Number	

department		
dept_name	VChr	PK
building	VChr	
budget	Number	

Removing Redundant Attributes

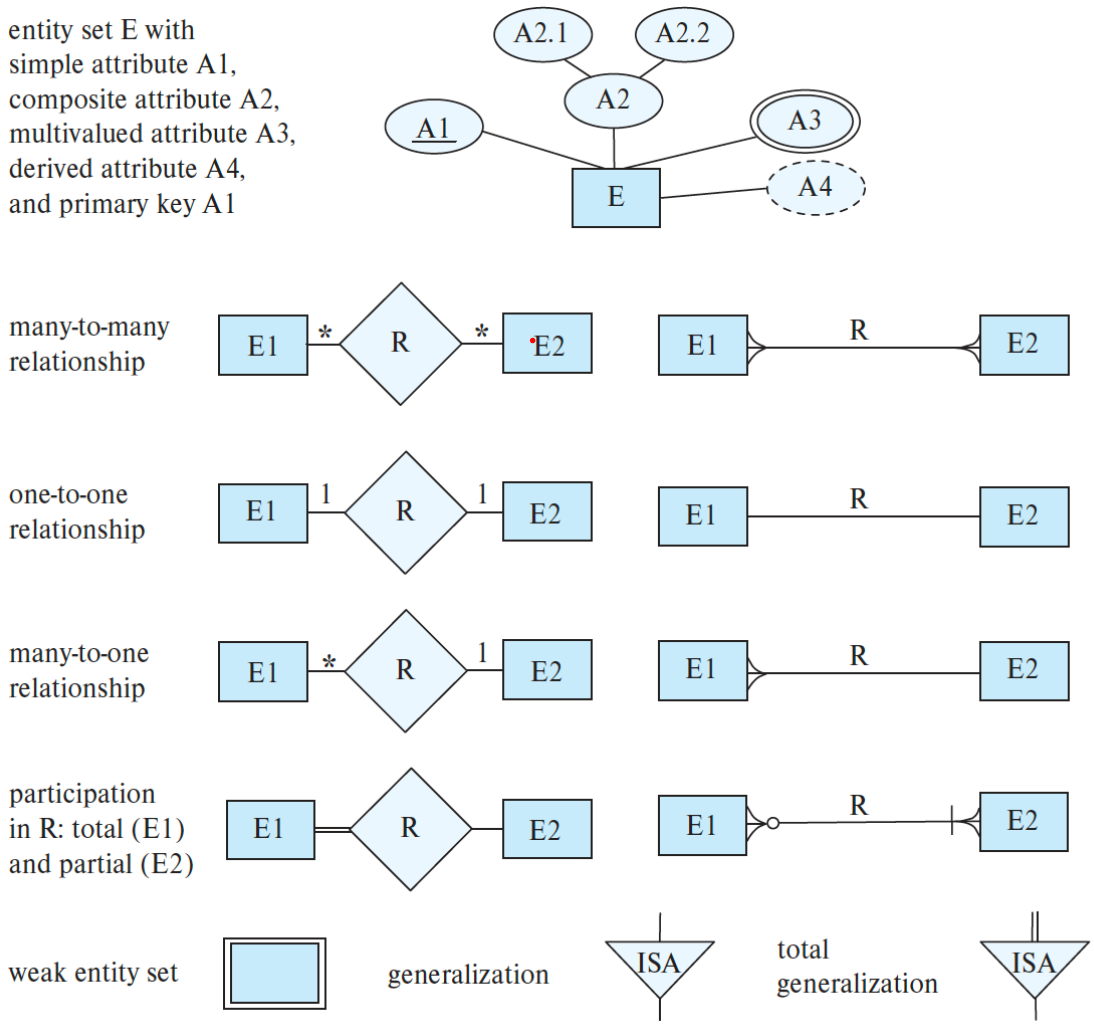
- A good E-R design does not include redundant attributes





Alternative Notations for E-R Diagrams

- There exist several notations for E-R Diagrams
- We are using the Crow's Foot Notation
- All notations are based in the concepts presented before

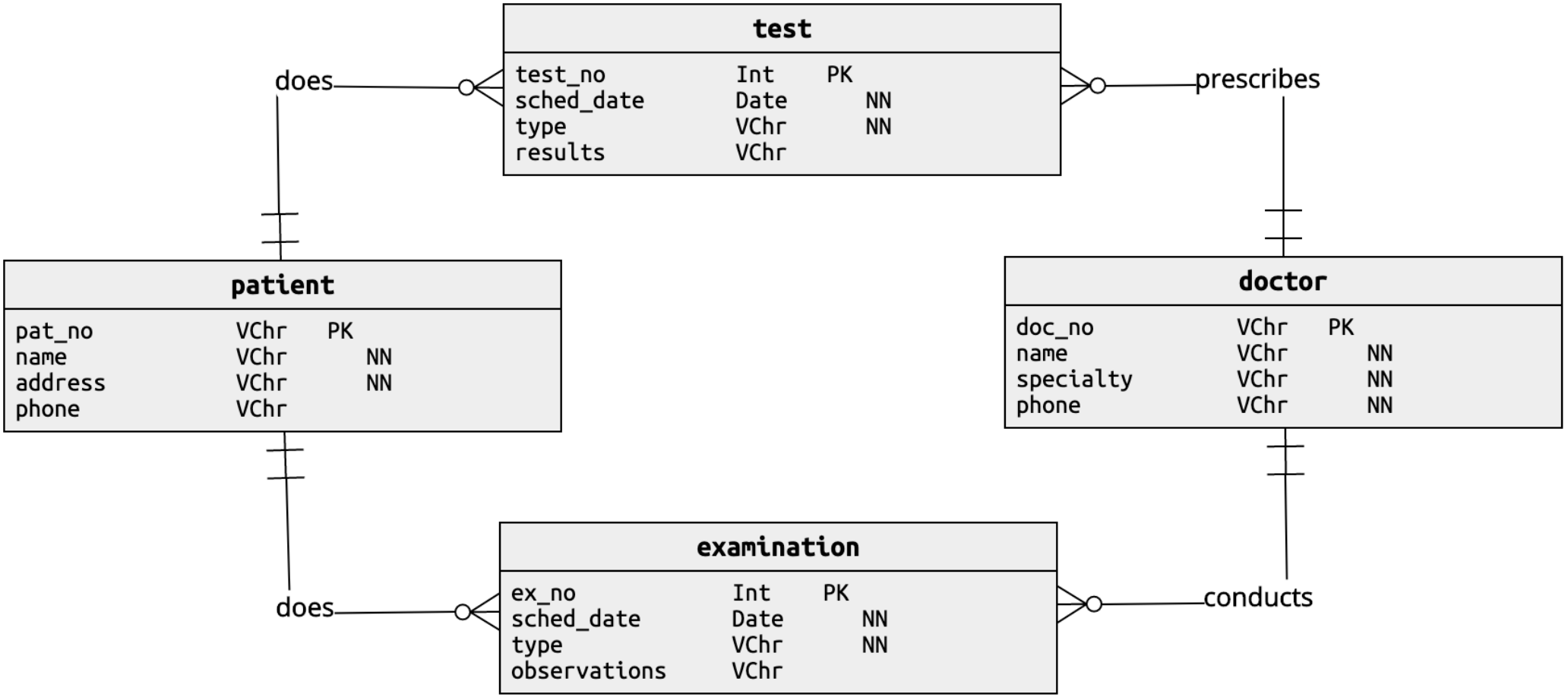


Source: A. Silberschatz, H. F. Korth and S. Sudarshan, "Database System Concepts", McGraw-Hill Education, Seventh Edition, 2019.

DEMO #1

- Assume the following:
  - *A small hospital needs to develop a new database application to manage patients. Over time, the hospital treats many patients and has a group of medical doctors. For each patient, the hospital wants a log of the various tests and examinations conducted. Tests are prescribed by doctors, while examinations are conducted by doctors.*
- TODO:
  - Entity sets?
  - Attributes of each entity?
  - Primary keys?
  - Relationship sets?
  - Cardinalities and participations?
  - E-R diagram…

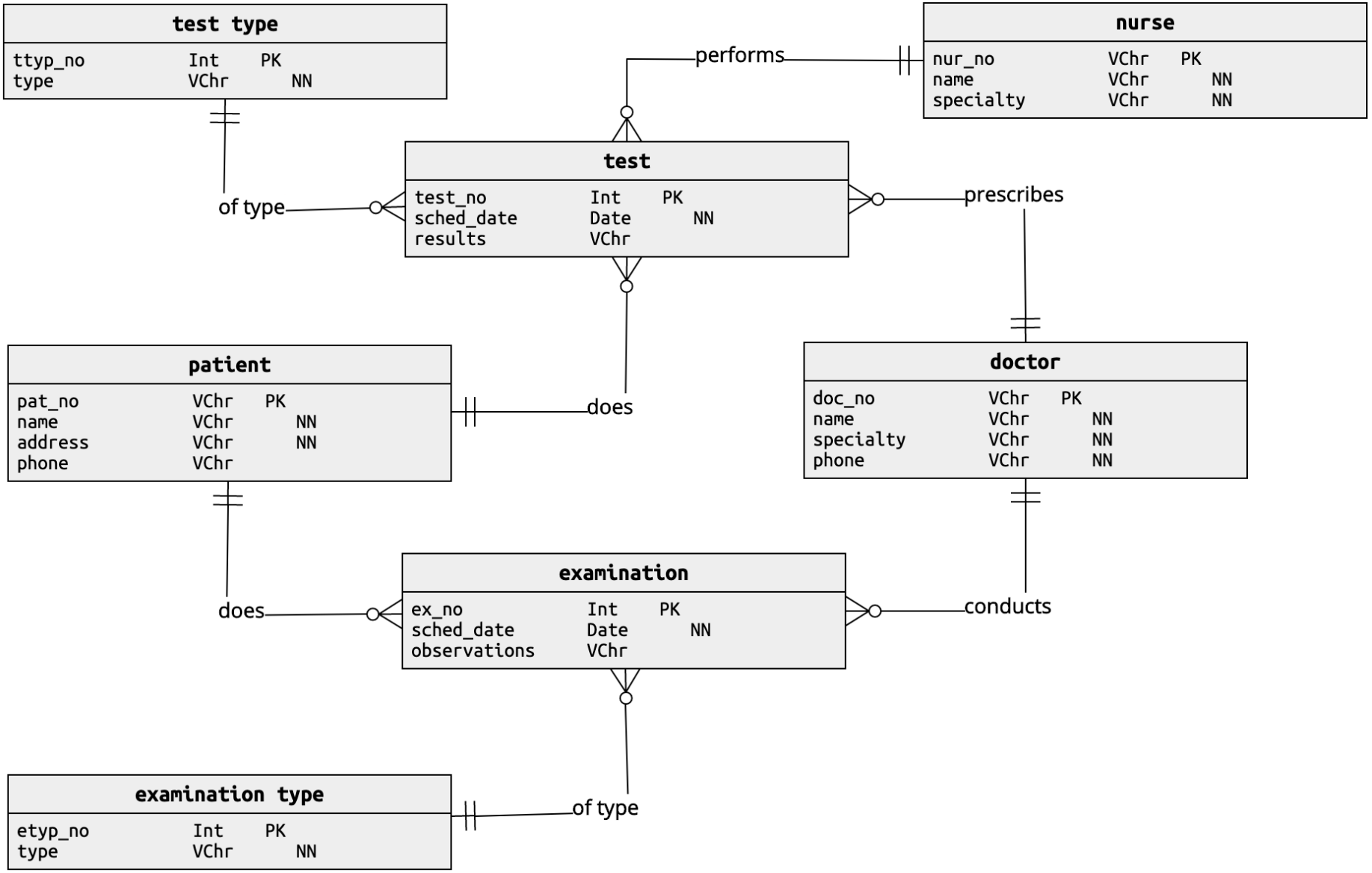
DEMO #1 – Potential E-R Diagram



DEMO #2

- Assume the following:
  - *A small hospital needs to develop a new database application to manage patients. Over time, the hospital treats many patients and has a group of medical doctors. For each patient, the hospital wants a log of the various tests and examinations conducted. Tests are prescribed by doctors, while examinations are conducted by doctors.*
- TODO:
  - Improve the E-R diagram from DEMO #1 considering:
    - Tests are performed by nurses
    - There is a predefined set of types of tests, which can evolve over time
    - There is a predefined set of types of examinations, which can evolve over time
  - Implement the diagram using the ***onda*** tool
    - **http://onda.dei.uc.pt (v4)**

DEMO #2 – Potential E-R Diagram



Obtaining Tables from Entity Sets and Relationship Sets
The ONDA Tool
Obtaining the SQL DDL commands

FROM E-R DIAGRAMS TO RELATIONAL SCHEMAS

The Goal

- Starting from the E-R Diagram, obtain the logical database design – relational schema
  - Relations, attributes, keys, constraints, etc.
  - Foreign-keys are generated during this process!
    - As seen before, no foreign-keys are represented in E-R diagrams (only relationship sets exist)
- Logical design may need to be fine-tuned (e.g., using normalization)
- The physical database design is built on top of this logical design
- SQL DDL commands can be automatically generated from the logical design in order to create the database
  - The SQL commands generated depend on the target database engine, as different engines implement slightly different versions of SQL

Strong Entity Sets → Relations

- Let E be a strong entity set with attributes a_1, $a2$,..., a_n
- This entity generates a relation schema E with n distinct attributes
  - Each tuple in relation E corresponds to one entity of the entity set E
- The primary key of the entity set serves as the primary key of the relation schema
  - This follows directly from the fact that each tuple corresponds to a specific entity in the entity set

entity set

department		
dept_name	VChr	PK
building	VChr	
budget	Number	

relation schema

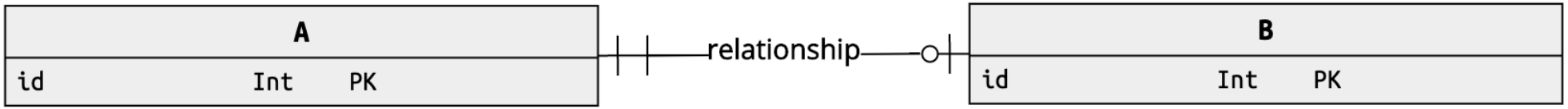
department		
dept_name	VChr	PK
building	VChr	
budget	Numeric	

Relationship Sets → Relations

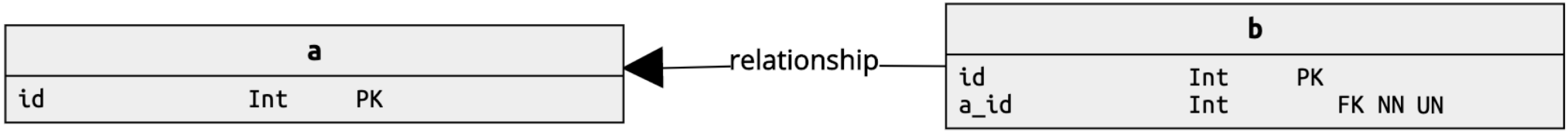
- Each strong entity set in the relationship set will lead to a relation
  - (there is one exception, which we will discuss next class)
- Depending on the mapping cardinalities and participation, several cases may occur
- An additional relation schema may (or may not) be needed
  - Primary key and foreign keys must be defined
- A foreign key may (or may not) be added to one of the relations originated by the strong entity sets
- Both foreign keys and new relations are needed to allow setting the relationships between tuples
  - We should avoid situations where foreign keys may have *null* values!

One-to-One Relationship Sets: Case #1

- Let's assume a one-to-one relationship set with partial participation on one entity set and full participation on the other



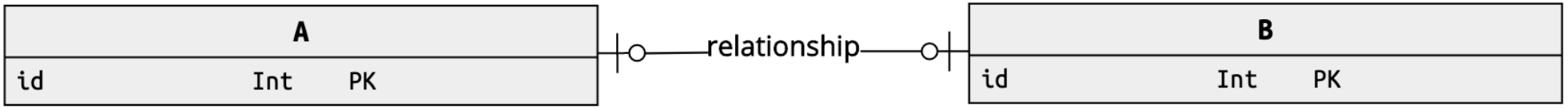
- What is the solution? We have the same three hypothesis:
  - Relation schema *A* includes a foreign key to relation schema *B*
  - Relation schema *B* includes a foreign key to relation schema *A*
  - A relation *A_B* is originated to allow the mapping between *A* and *B*



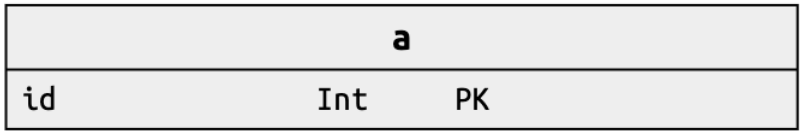
Why is not the third hypothesis a good one?

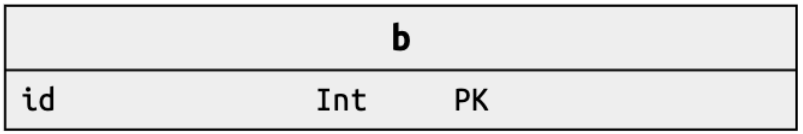
One-to-One Relationship Sets: Case #2

- Let's now assume a one-to-one relationship set with partial participation on both entity sets



- Each entity set leads to a relation schema

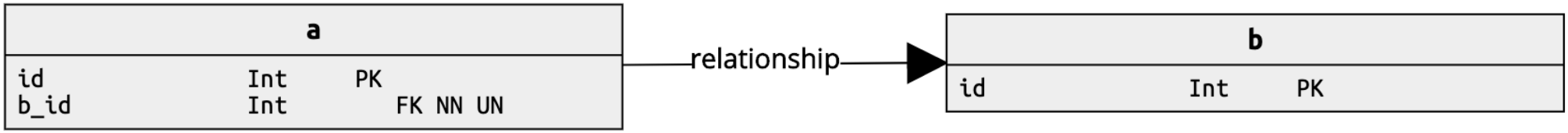




- What about the relationship set? What does it originate?

One-to-One Relationship Sets: Case #2

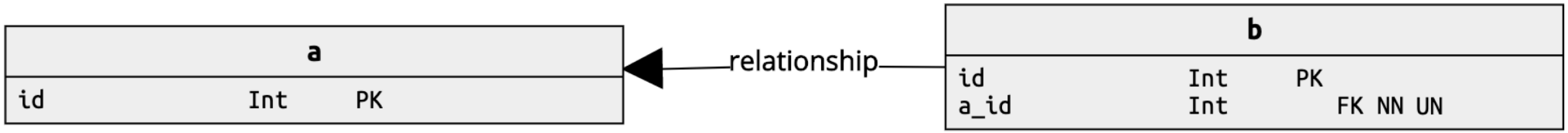
- First hypothesis: relation schema *A* includes a foreign key to relation schema *B*



- Is this a good solution? Will there be tuples in relation *A* where the foreign key *b_id* is *null*?
  - Yes! According to the E-R diagram, there is partial participation, so some *A* entities do not have a correspondence to a *B* entity

One-to-One Relationship Sets: Case #2

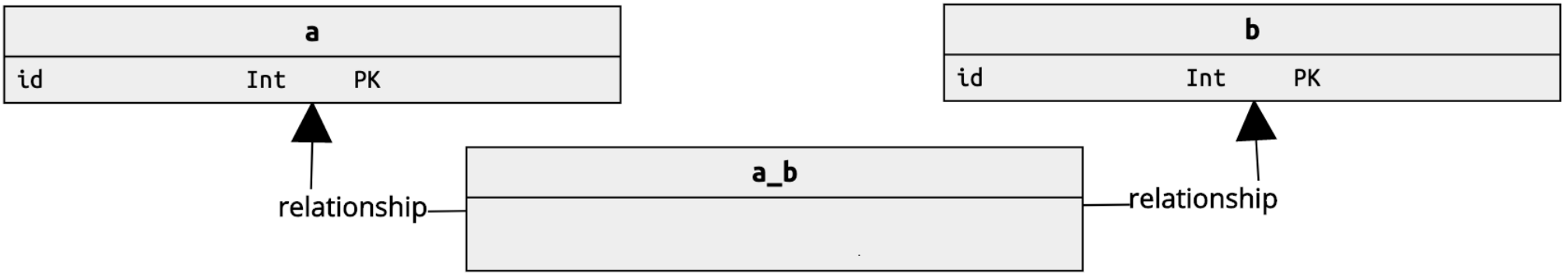
- Second hypothesis: relation schema *B* includes a foreign key to relation schema *A*



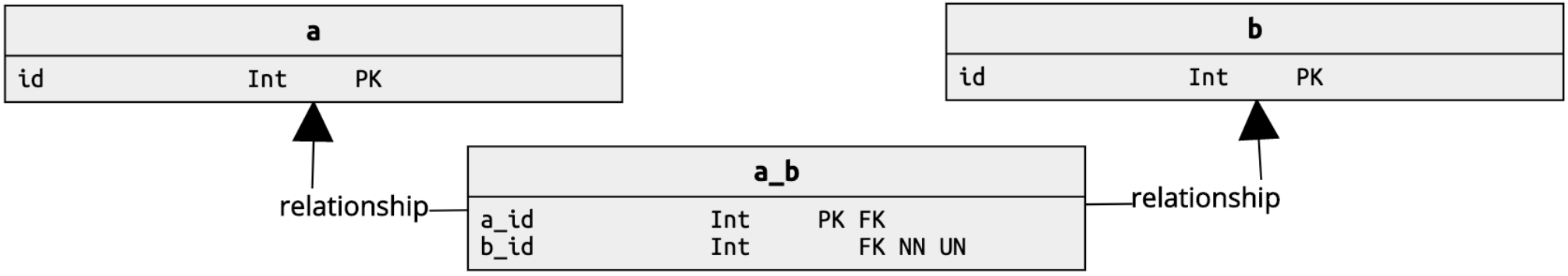
- Is this a good solution? Will there be tuples in relation *B* where the foreign key *a_id* is *null*?
  - Yes! According to the E-R diagram, there is partial participation, so some *B* entities do not have a correspondence to a *A* entity

One-to-One Relationship Sets: Case #2

- Third hypothesis: a relation *A_B* is originated to allow the mapping

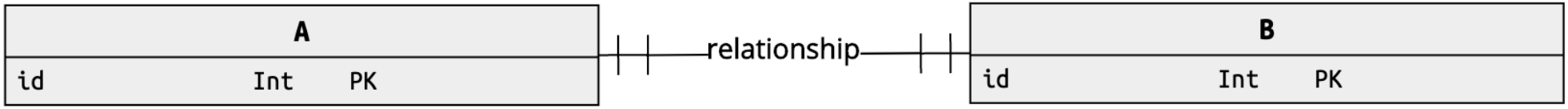


- What are the attributes of the new relation schema *A_B*?
- What is the primary key of *A_B*?
- Are there any foreign keys?



One-to-One Relationship Sets: Case #3

- Finally, let's assume a one-to-one relationship set with full participation on both entity sets

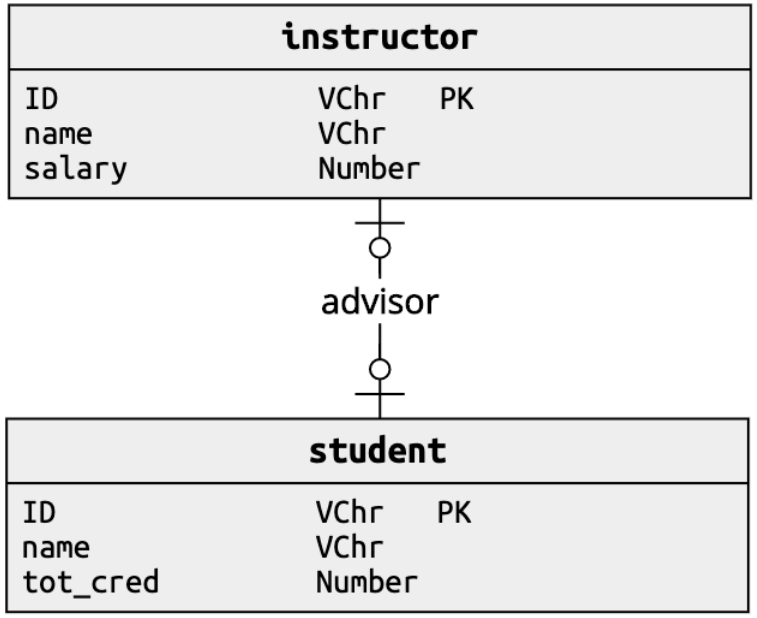


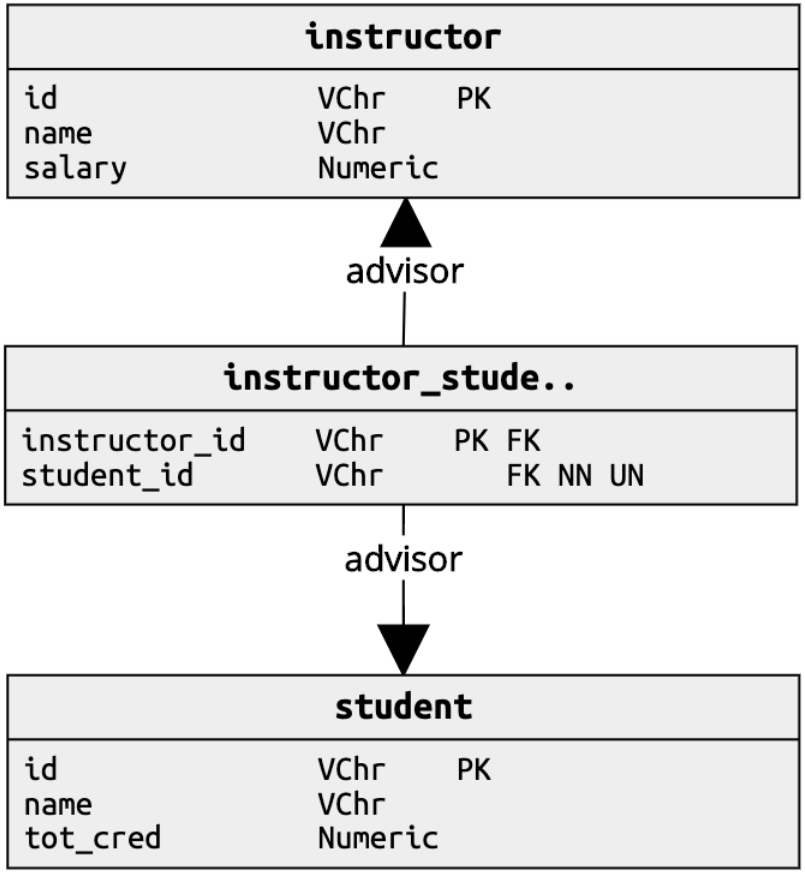
- What is the solution? We have the same three hypothesis:
  - Relation schema *A* includes a foreign key to relation schema *B*
  - Relation schema *B* includes a foreign key to relation schema *A*
  - A relation *A_B* is originated to allow the mapping between *A* and *B*

None of these is good!

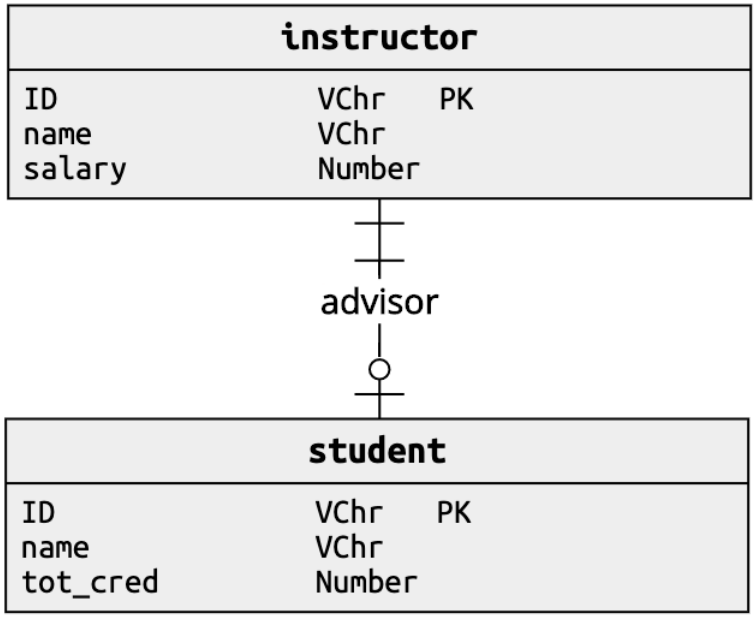
a_b		
id	Int	PK
b_id	Int	NN UN

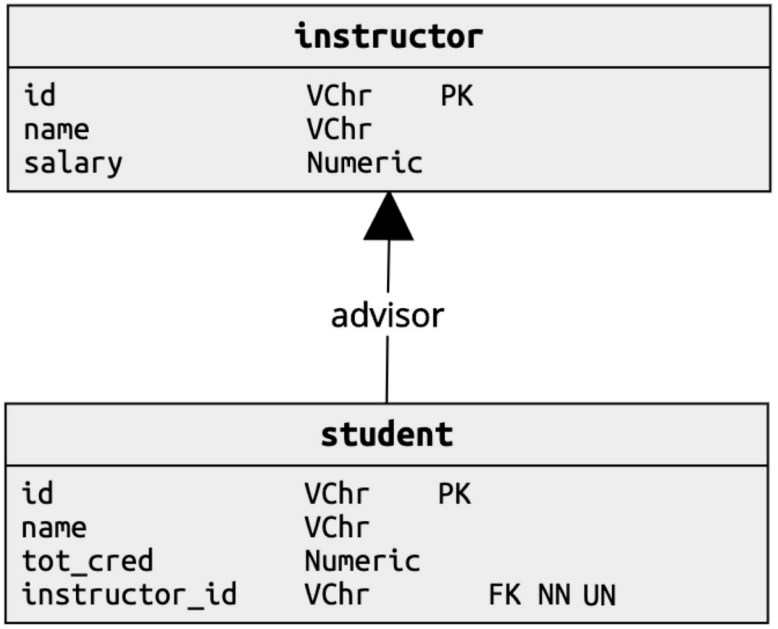
One-to-One Relationship Sets: Examples



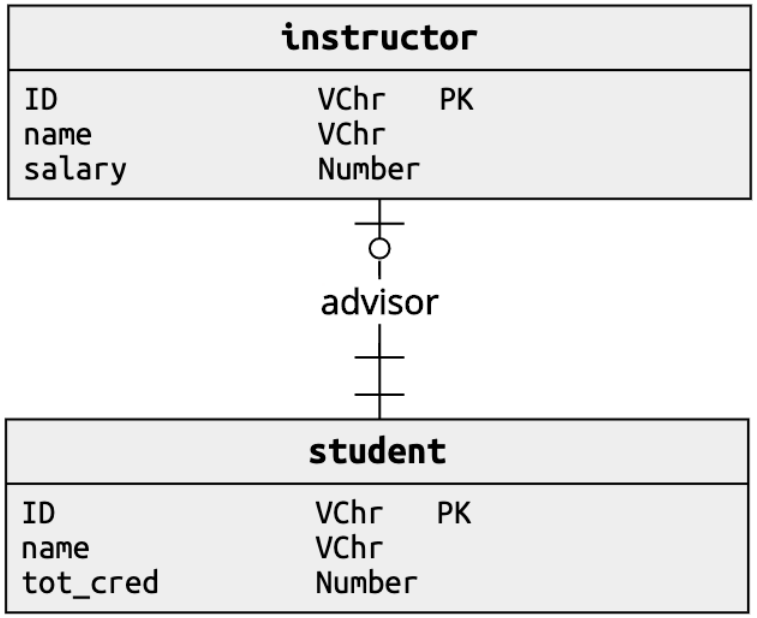


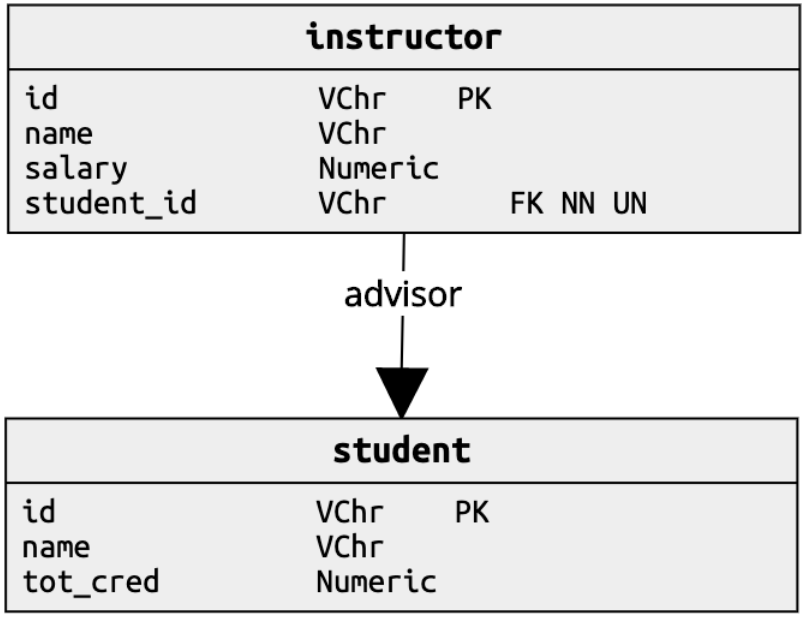
One-to-One Relationship Sets: Examples



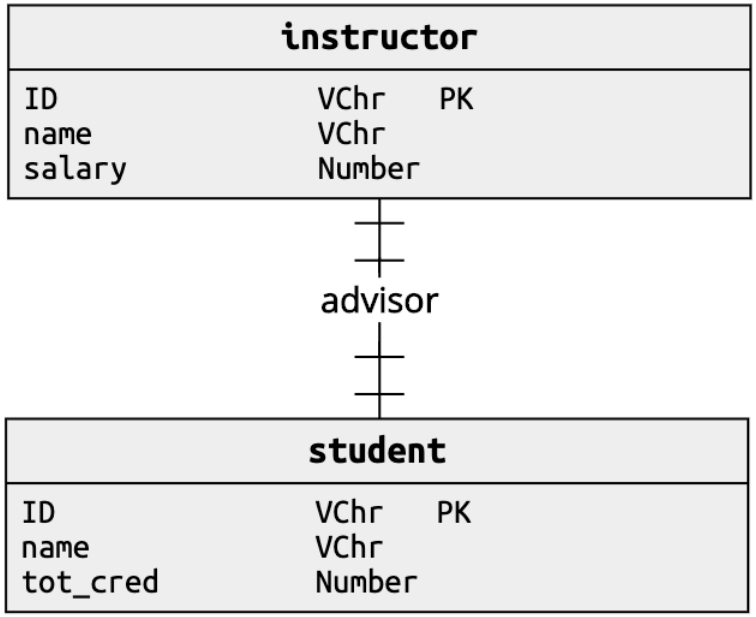


One-to-One Relationship Sets: Examples





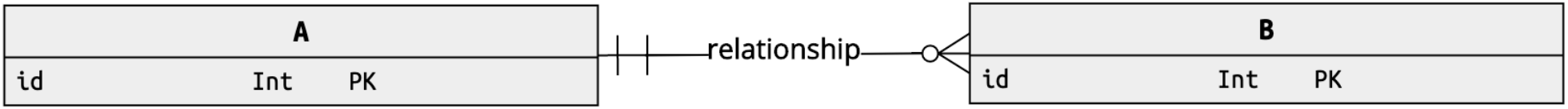
One-to-One Relationship Sets: Examples



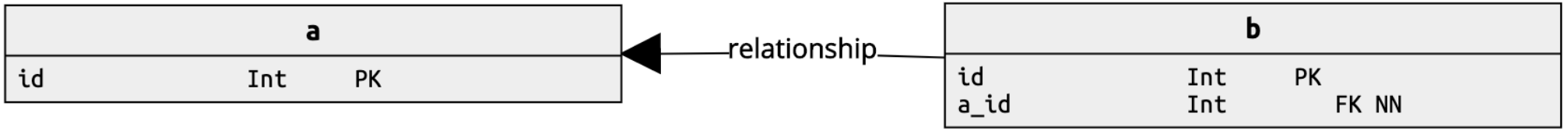
instructor_stude..		
id	VChr	PK
name	VChr	
salary	Numeric	
student_id	VChr	NN UN
student_name	VChr	
student_tot_cred	Numeric	

One-to-Many Relationship Sets: Case #1

- Let's assume a one-to-many relationship set with total participation on the *many* side

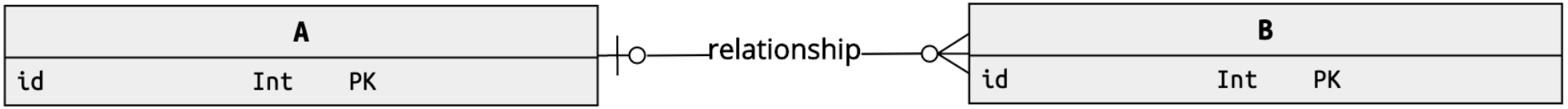


- What is the solution?

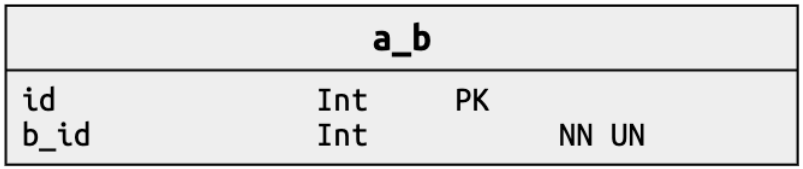


One-to-Many Relationship Sets: Case #2

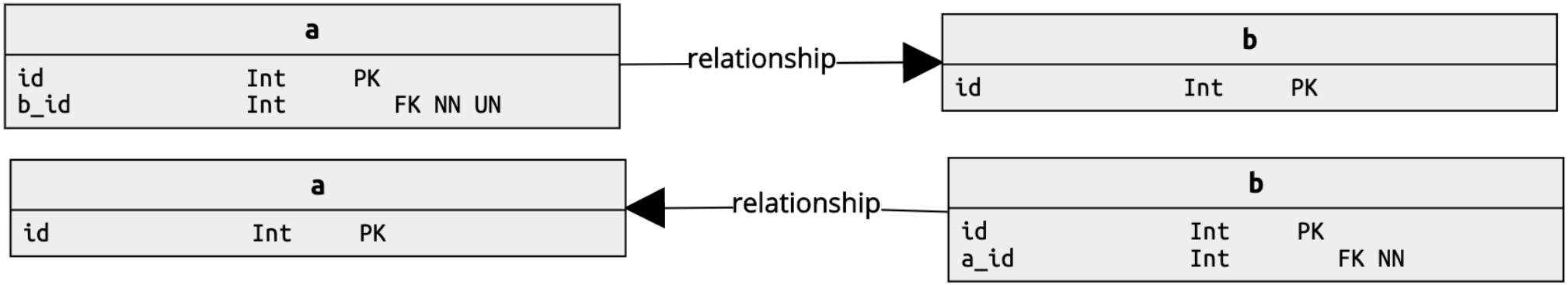
- Let's assume a one-to-many relationship set with partial participation on both entity sets



- Can this be represented with a single relation schema?

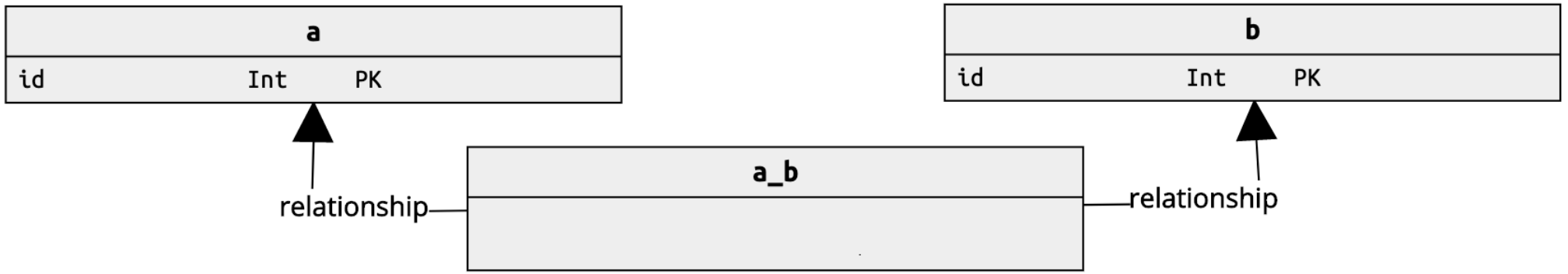


- What about with two tables?

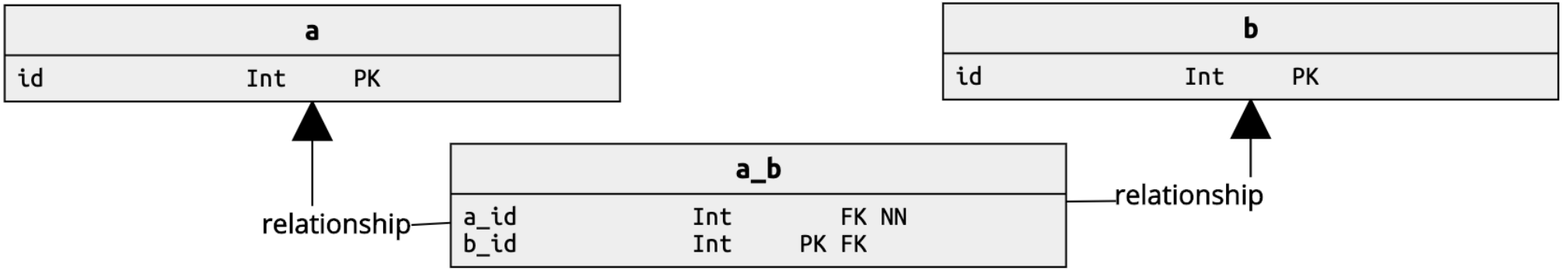


One-to-Many Relationship Sets: Case #2

- What about three tables?

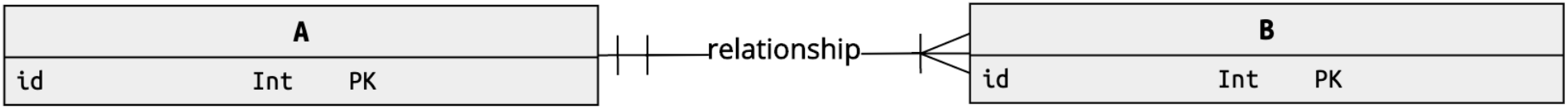


- What are the attributes of the new relation schema *A_B*?
- What is the primary key of *A_B*?
- Are there any foreign keys?

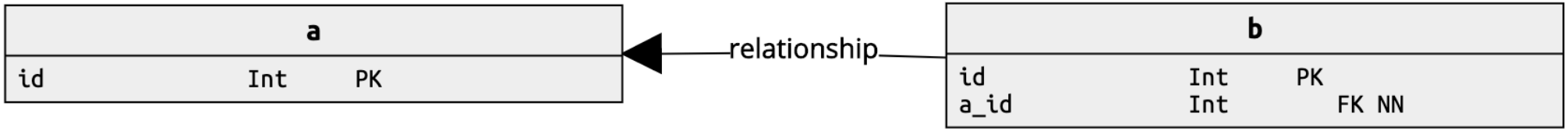


One-to-Many Relationship Sets: Case #3

- Finally, let's assume a one-to-many relationship set with total participation on both sides

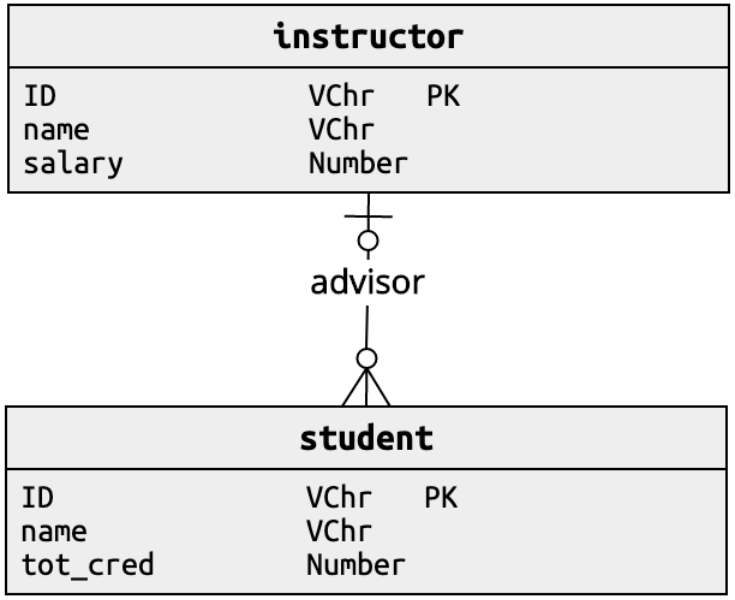


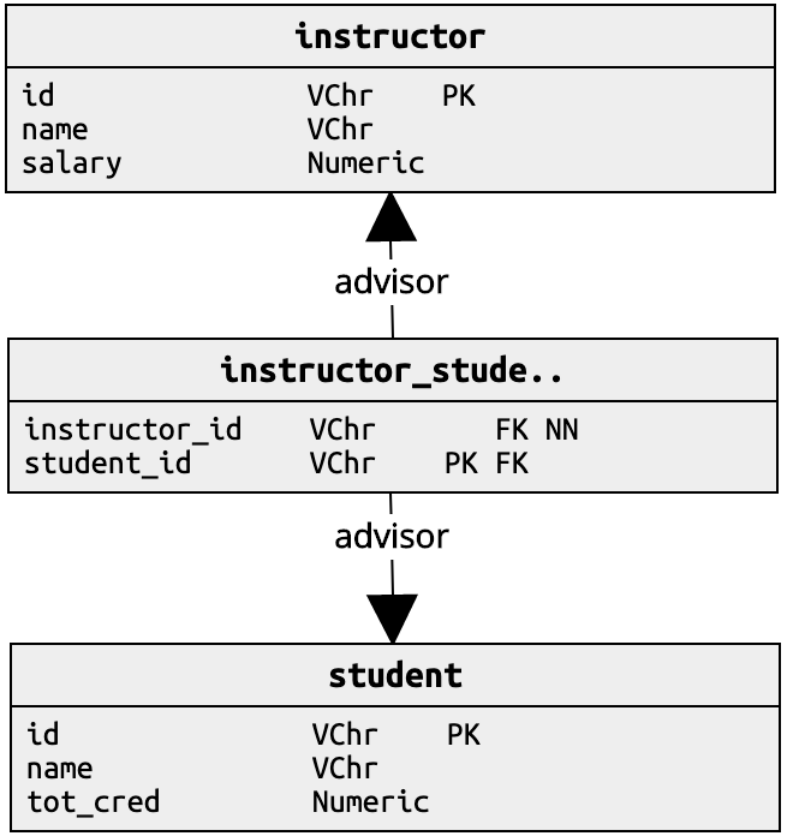
- Does this participation make any difference?



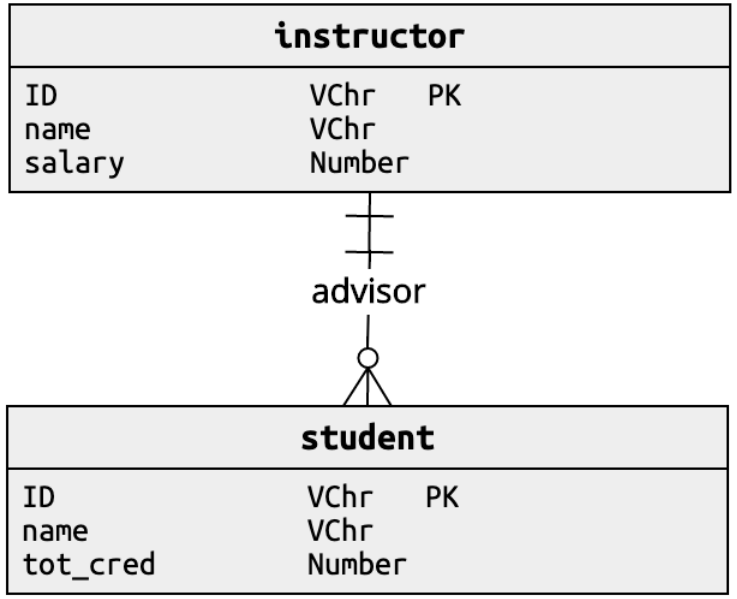
- So, why should it be represented in the E-R diagram?

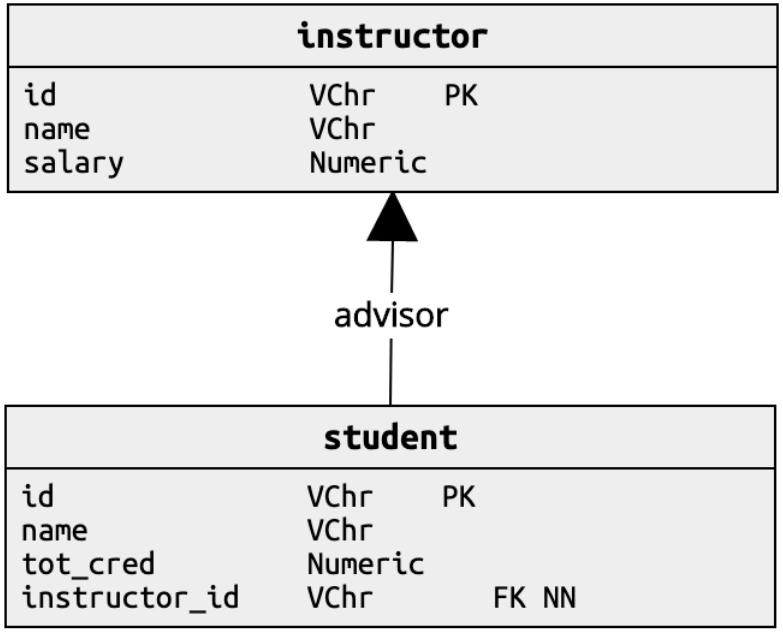
One-to-Many Relationship Sets: Examples



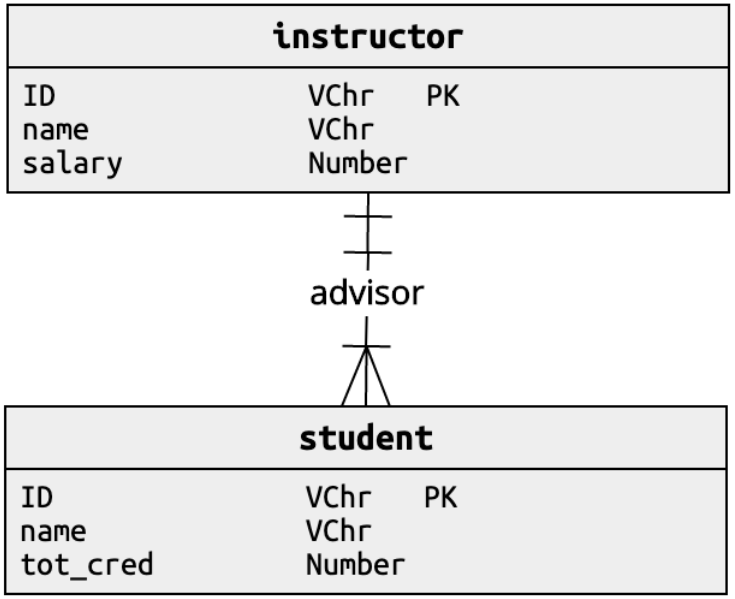


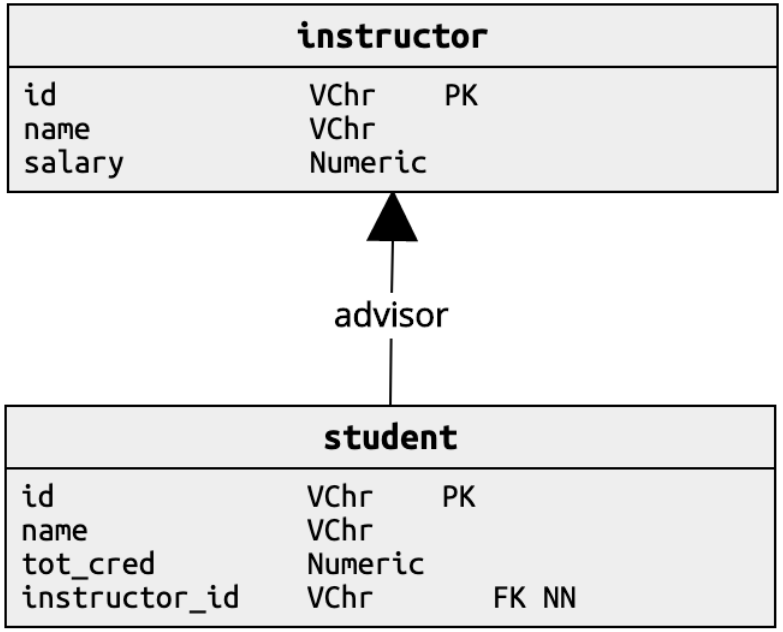
One-to-Many Relationship Sets: Examples



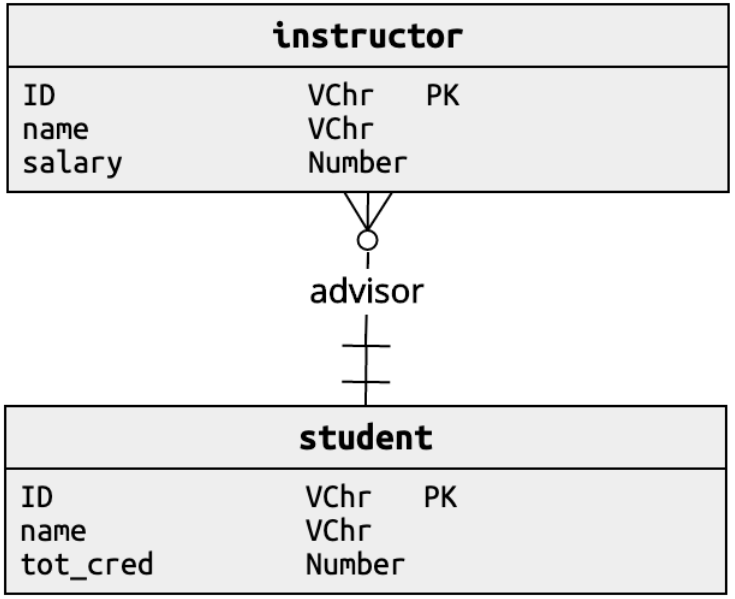


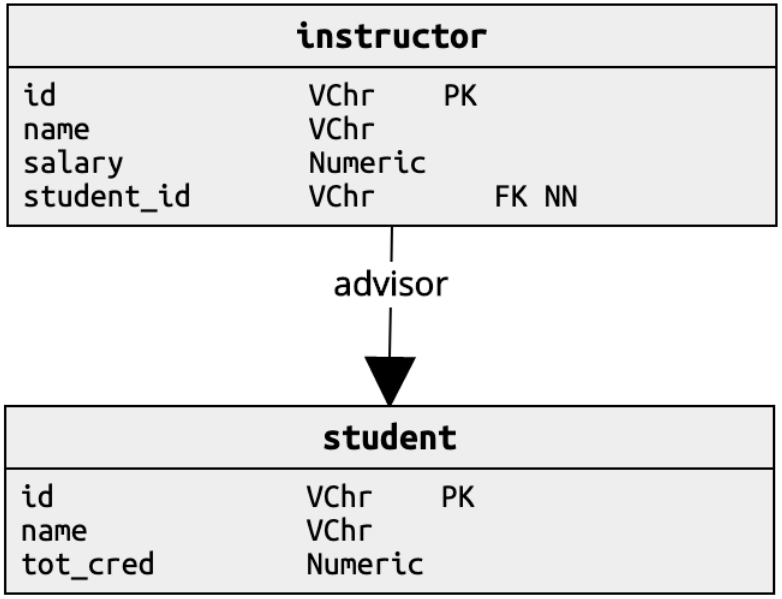
One-to-Many Relationship Sets: Examples





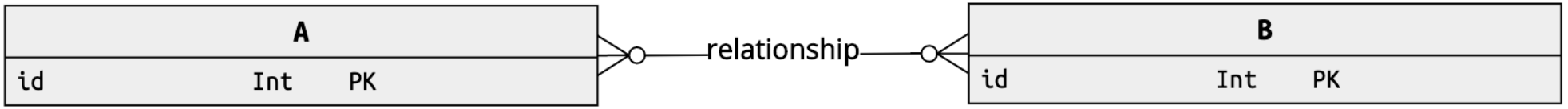
One-to-Many Relationship Sets: Examples



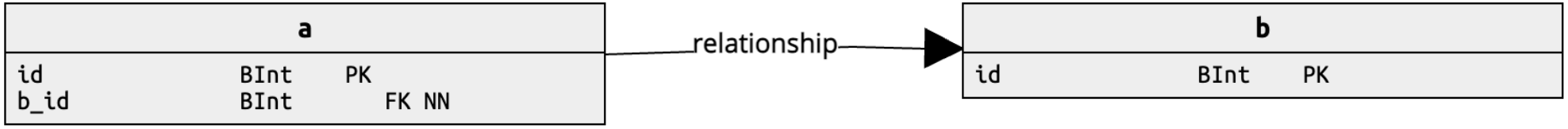


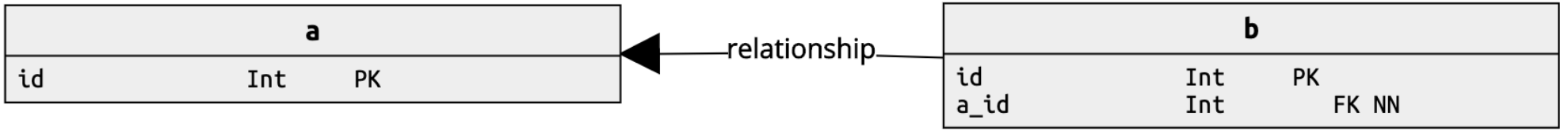
Many-to-Many Relationship Sets

- Let's assume a many-to-many relationship set with partial participation on both sides



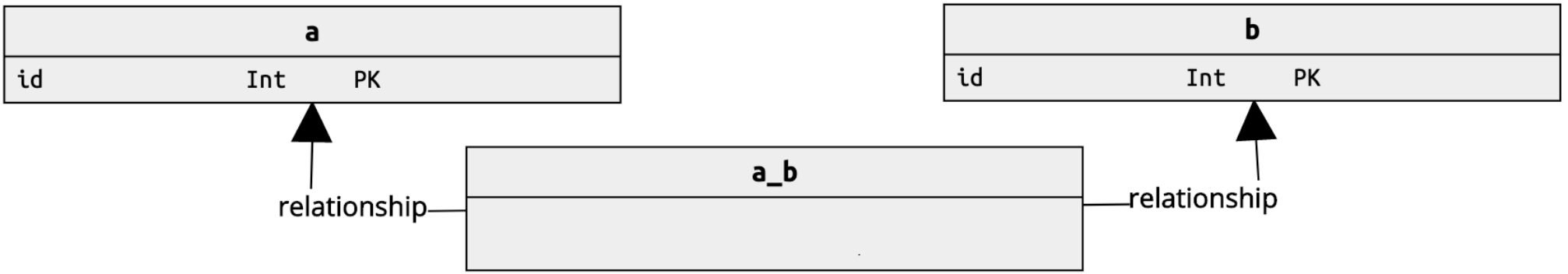
- Can it be represented with two relations?



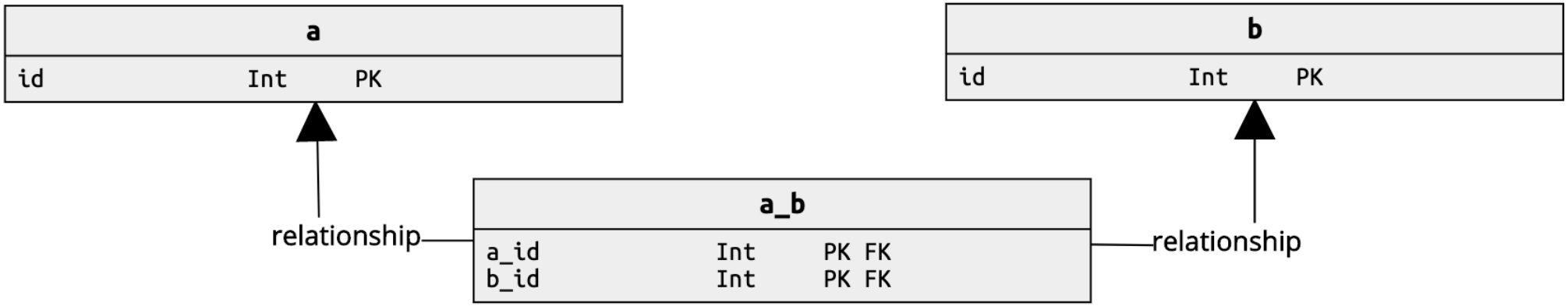


Many-to-Many Relationship Sets

- What about three relations?

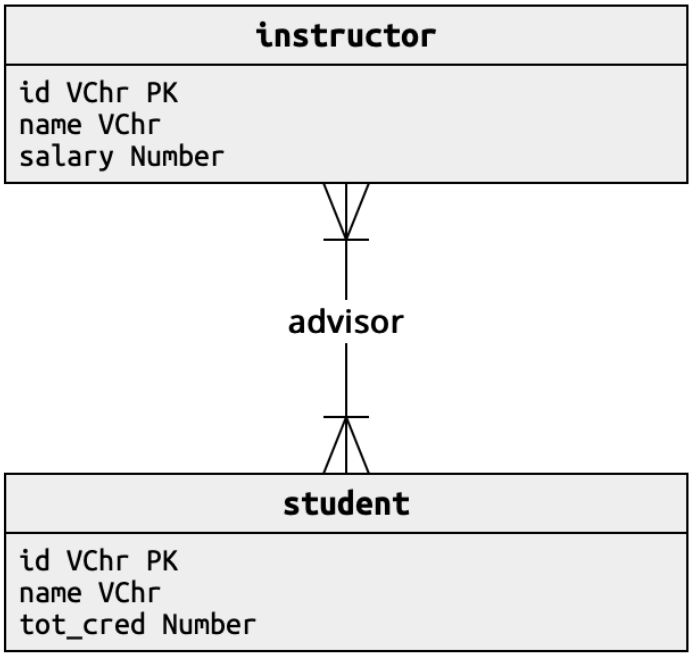


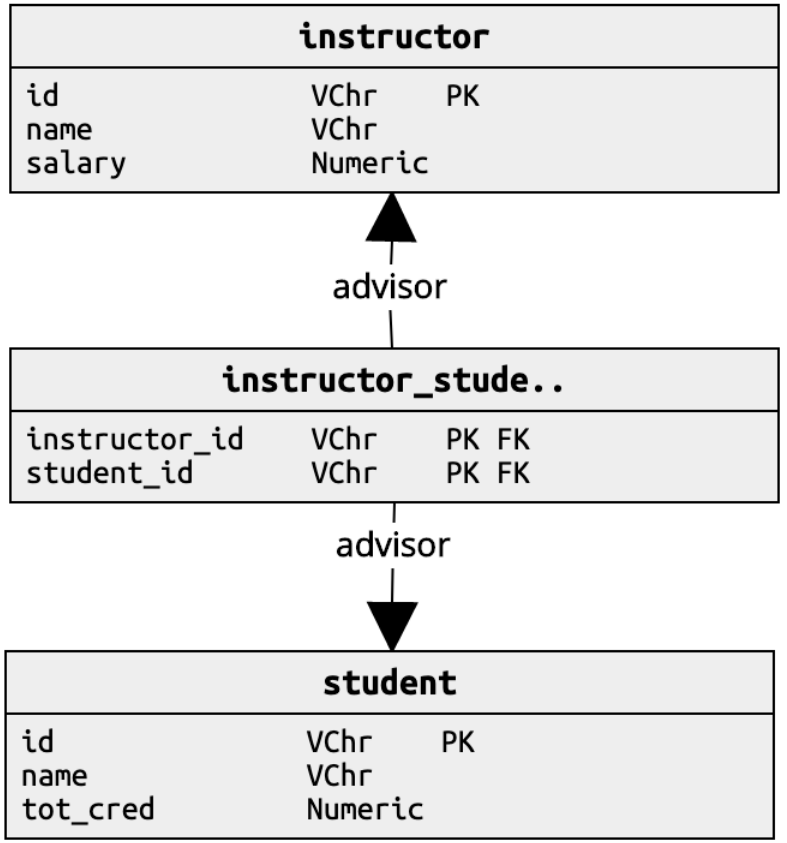
- What is the primary key of *A_B*?
- Are there any foreign keys?



Many-to-Many Relationship Sets

- Does total participation make any difference in many-to-many relationships?
- So, why should total participation be represented in the E-R diagram?





Take-Away(s)

- Design Process: conceptual-design, logical-design, physical-design
- Entity-relationship (E-R) data model
- Entity and entity set
- Relationship and relationship set: binary, recursive
- Mapping cardinality: one-to-one, one-to-many, many-to-many
- Total and partial participation
- E-R diagram
- E-R diagram to relational schemas: different cases depending on the mapping cardinality and participation
- *onda* tool (and many others) can be used to support the process

Next Lesson(s)

- Weak Entity Sets
- Attributes of Relationship Sets
- *n-ary* Relationship Sets
- Extended E-R features
  - Specialization
  - Generalization
  - Attribute Inheritance
  - Completeness Constraints
- Typical design issues

Q&A

Databases

Database Design Using Entity-Relationship Model

João R. Campos

Bachelor in Informatics Engineering
Department of Informatics Engineering
University of Coimbra
2024/2025