

Part 2

Entity Relationship Diagram (ERD)

- Example: Company Database
- Entity, Attribute, Relationship
- Structural constraints
- Weak entity types
- More ER Examples (Registration DB)
- More ER Examples (Bank DB)
- In class exercise 1: Zoo Database
- In class exercise 2: Library Database
- Alternative Notations for structural constraints (min,max) notation

Prepared By: Dr. Osama Al-Haj Hassan

Reference to: Fundamentals of Database Systems , Ramez Elmasri and Shamkant B. Navathe, Fifth Edition.

Example Database (Company DB)

- Suppose the outcome of requirements collection is as follows.
 - The company is organized into departments. Each department has a unique name, a unique number, and a particular employee who manages the department. We keep track of the start date when that employee began managing the department. A department may have several locations.

Example Database (Company DB)

- Suppose the outcome of requirements collection is as follows
 - A department controls a number of projects, each of which has a unique name, a unique number, and a single location.

Example Database (Company DB)

- Suppose the outcome of requirements collection is as follows
 - We store each employee's name, social security number, address, salary, gender, and birth date. An employee is assigned to one department but may work on several projects, which are not necessarily controlled by the same department. We keep track of the number of hours per week that an employee works on each project. We also keep track of the direct supervisor of each employee.

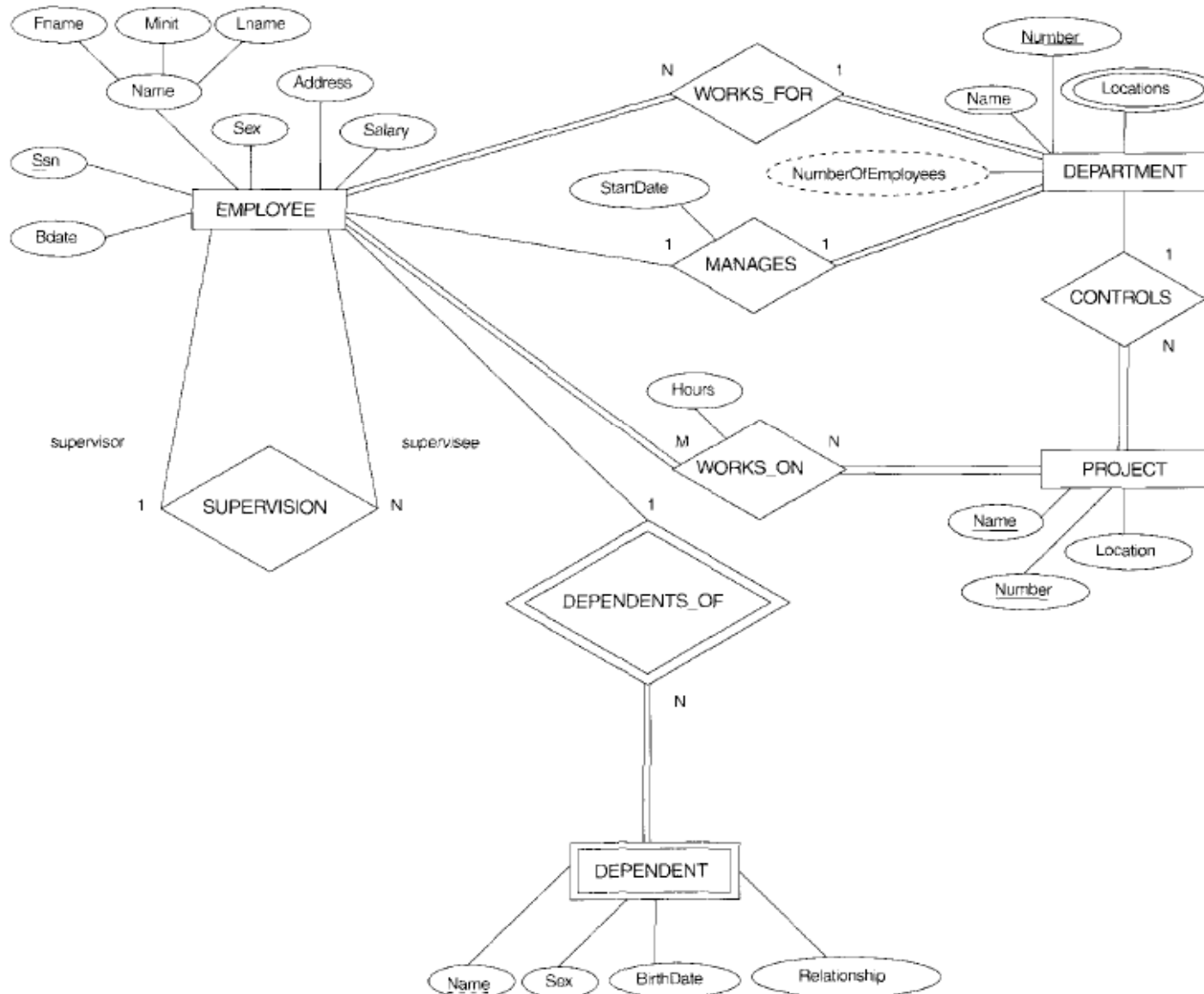
Example Database (Company DB)

- Suppose the outcome of requirements collection is as follows
 - We want to keep track of the dependents of each employee for insurance purposes. We keep each dependent's first name, gender, birth date, and relationship to the employee.

Example Database (Company DB)

- The previous requirements can be translated into the following schema represented as “Entity-Relationship (ER) Schema Diagram”.
- This ER diagram is shown in the next slide.

ER Schema diagram for the Company Database



Entities

- Entity:
 - Is a "thing" in the real world with an independent existence (eg. Department, employee, project, ... etc). (This is called Strong Entity)
 - It's ER Diagram notation:



ENTITY

- Example: Department Entity



Entity Instance

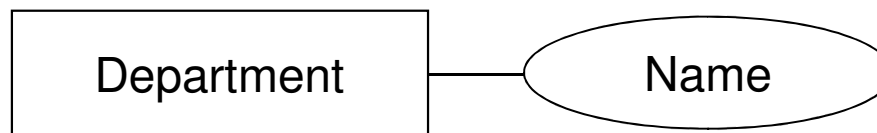
- An Entity is a general type that includes all instances
- An instance is an example of the entity
- Example:
 - Employee is an entity
 - Ahmad is an instance of entity Employee
- Example:
 - Department is an entity
 - CS Department is an instance of entity Department

Attributes

- Attribute:
 - Properties that describe entities (eg. Employee name, department name, ...etc).
 - It's ER Diagram notation



- Attribute “Value Set” or “Domain”
 - Data type associated with the attribute.
 - Example: EmployeeID is an integer.
 - Example: Grade can take letters “A,” “B,” “C,” “D”, and “F”.
- Attribute Example: Name is an attribute of Department

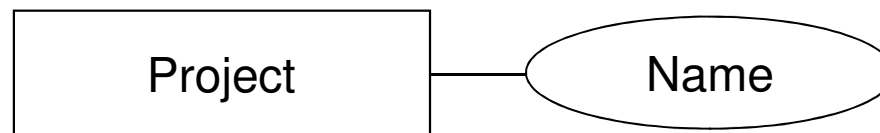


Attribute Types

- Simple (Atomic) Attribute
 - Attributes that are not divisible.
 - It's ER Diagram notation is the same as the general attribute you saw in the previous slide.



- Example: Project Name is a simple attribute of entity Project

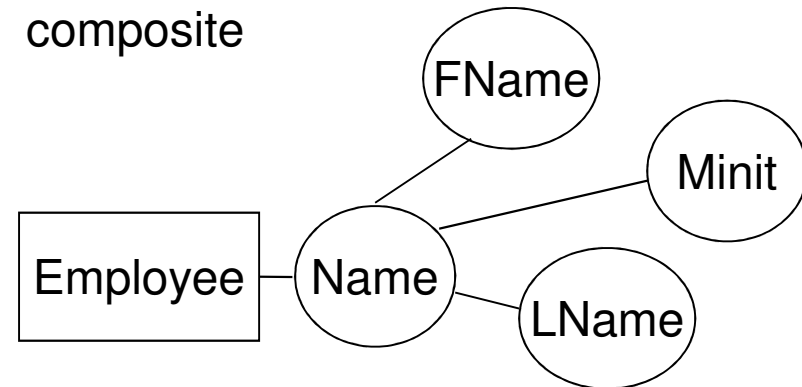


Attribute Types

- Composite Attribute
 - Can be divided into smaller subparts.
 - Example: Address can be derived into “city”, “street”, “building number”, and “apartment number”.
 - It’s ER-Diagram notation:

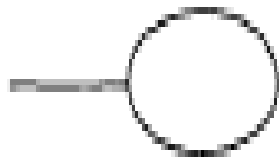


Example: Employee Name can be composite



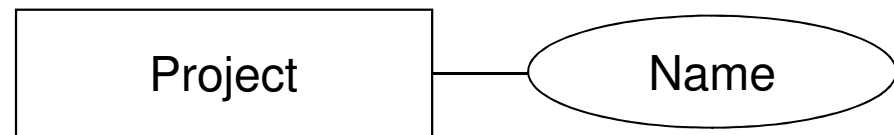
Attribute Types

- Single-Valued Attribute:
 - A **single** entity has a **single** value of that attribute.
 - Example: employee ID. A **single** employee has only one **single** ID.
 - Its ER Diagram notation is the same as a general attribute .



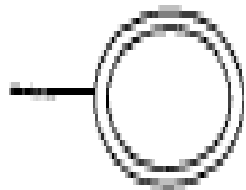
ATTRIBUTE

A **single** project has a **single** name



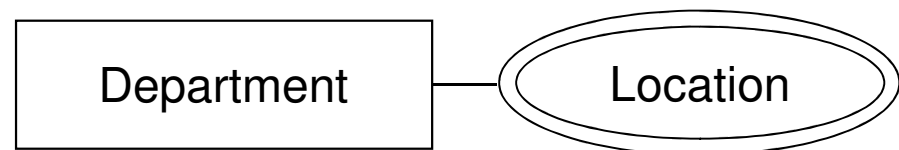
Attribute Types

- Multi-Valued Attribute:
 - A **single** entity has a **multiple** values of that attribute.
 - Example: Employee earned degree. A **single** employee can have **multiple** degrees (B.SC + MS + PhD degrees).
 - Its ER Diagram Notation is **two nested circles**.



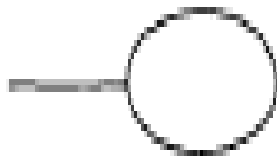
MULTIVALUED ATTRIBUTE

Example: A **single** department can have **multiple** locations



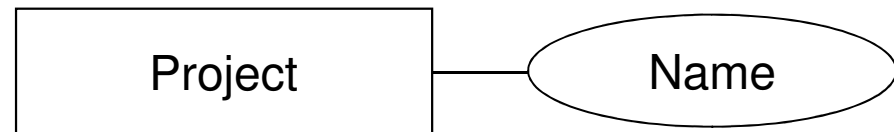
Attribute Types

- Stored Attribute
 - Cannot be derived from any other attribute.
 - Example: Birth Date.
 - Its ER-Diagram Notation is the same as the general attribute notation.



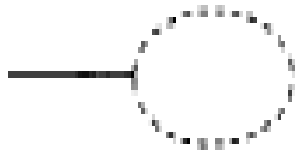
ATTRIBUTE

Example: Name is a stored attribute of project entity



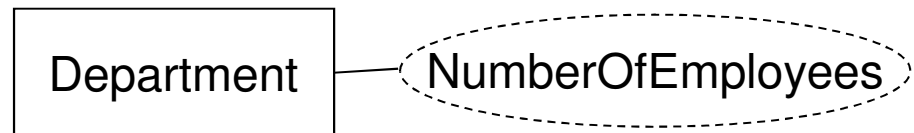
Attribute Types

- Derived Attribute
 - Can be derived from a “Stored Attribute”.
 - Example: employee age can be derived from his birth date and today’s date.
 - Its ER Diagram Notation is **dashed** circle.



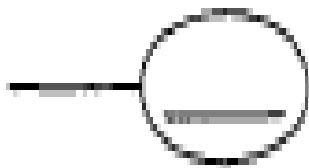
DERIVED ATTRIBUTE

Example: NumberOfEmployees
Is a derived attribute of entity Department.
It is derived from count of records.



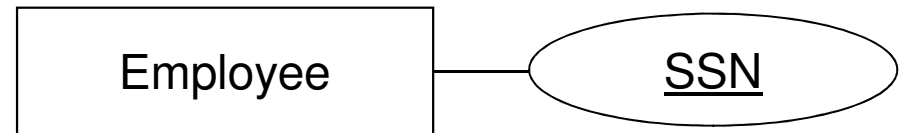
Attribute Types

- Key Attribute:
 - Its value uniquely identifies the entity it belongs to.
 - Example: employee ID uniquely identifies an employee.
 - Its ER Diagram Notation is an **underlined attribute name**.



KEY ATTRIBUTE

Example: SSN is a key attribute of entity Employee



Key Attribute Examples

- What is the key of the following entities?

Entity	Key
Student	Student_id
Employee	Employee_id
Course	Course_number
Section	section_id, course_number, semester, year
Bank branch	bank_id, branch_id
Employee Project	employee_id, project_id

Attribute Types

- Complex Attribute
 - Uses multi-level of nesting in a multi-valued or composite attributes.
 - Example:
 - PreviousDegrees of a STUDENT is a composite multi-valued attribute denoted by {PreviousDegrees(College, Year, Degree, Field)}.
 - Here we used “Multi-value” and “Composite”.
 - “Multi-value”: A single student can have multiple degrees.
 - “Composite”: each degree can be specified by several attributes (college, year, degree, field)

Note

- One attribute can have different types at the same time.
- For example: Project Name is:
 - Stored Attribute.
 - Key Attribute.
 - Single-Valued Attribute.
 - Simple Attribute.

Null Values

- Some attribute values could be optional or may be they are not crucial to have.
- For example, if you have an attribute **Hobbies**. It is OK for the value of this attribute to be missing. In this case, we call it “NULL” value.
- Key Attributes **cannot** be NULL because they uniquely identifies an entity, so they have to have a value.

More about “Key” Attributes

- A key uniquely identifies each entity in Entity Set.
- For example: StudentID is a key for student entity because each student has different (Unique) ID.
- Some times key attributes can be **composite** attributes. This happens when a single attribute cannot satisfy the “Uniqueness Requirement”.
Example:

SectionID, courseID, Semester, Year

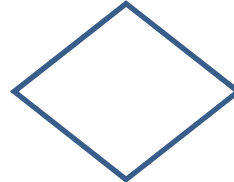
1	,	34234,	first,	2010
1	,	34234,	first,	2009
1	,	44478,	Sum,	2010

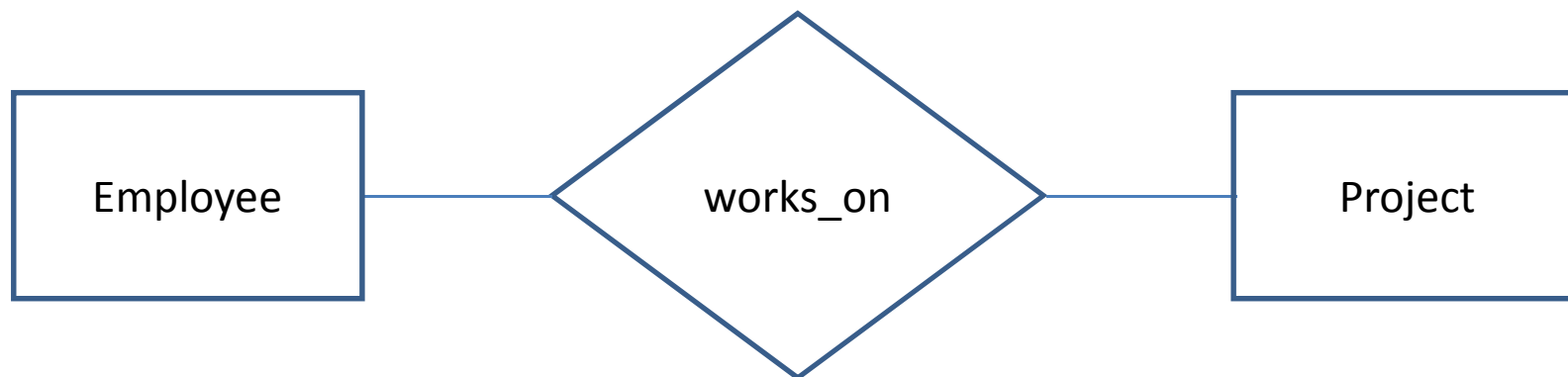
- In this example, Section entity has one key.
- This key consists of combination of 4 attributes.

Keys should be Minimal

- A key is minimal if it cannot be broken into smaller parts that work as a key.
- For example:
 - SectionID, courseID, Semester,Year
 - Is it Minimal?
 - Yes, because none of its smaller parts can work by itself as a key.
- Keys should be minimal.
- “studentID+studentAge”, is it minimal key of Student?
 - No, because “studentID” by itself works as a key.

Relationship

- A Relationship is an association between two or more entities.
- It is represented in ER diagram by using the following shape which is connected to participating entities. 
- Usually, a “verb” is written inside the relationship shape because verbs can express why the entities are connected with each other.

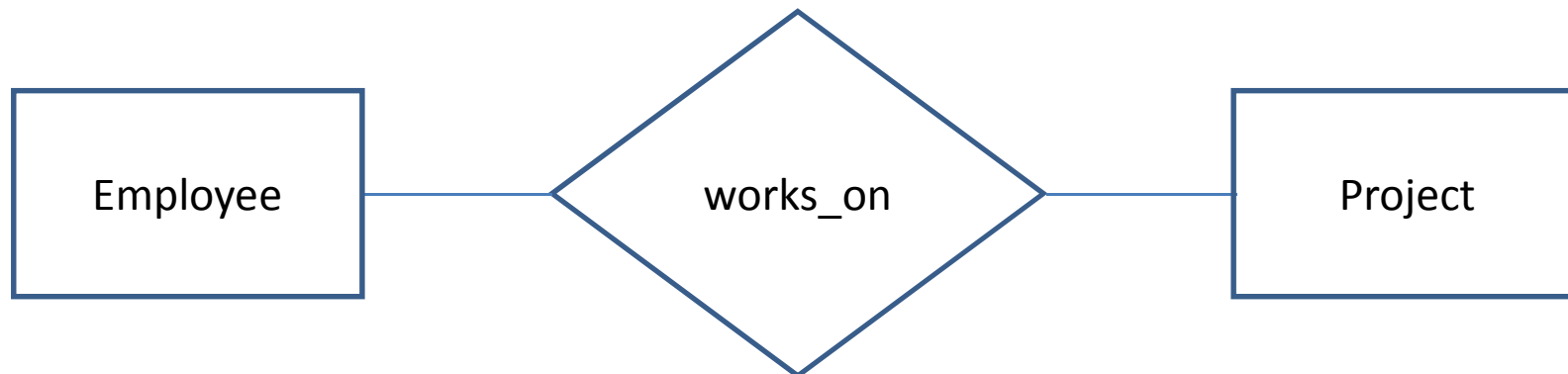


Relationship Degree

- Relationship degree is the number of participating entity types.
- Possible relationship degrees are
 - Binary Relationship: includes 2 entity types.
 - Ternary Relationship: includes 3 entity types.
 - N-ary Relationship: includes N entity types.

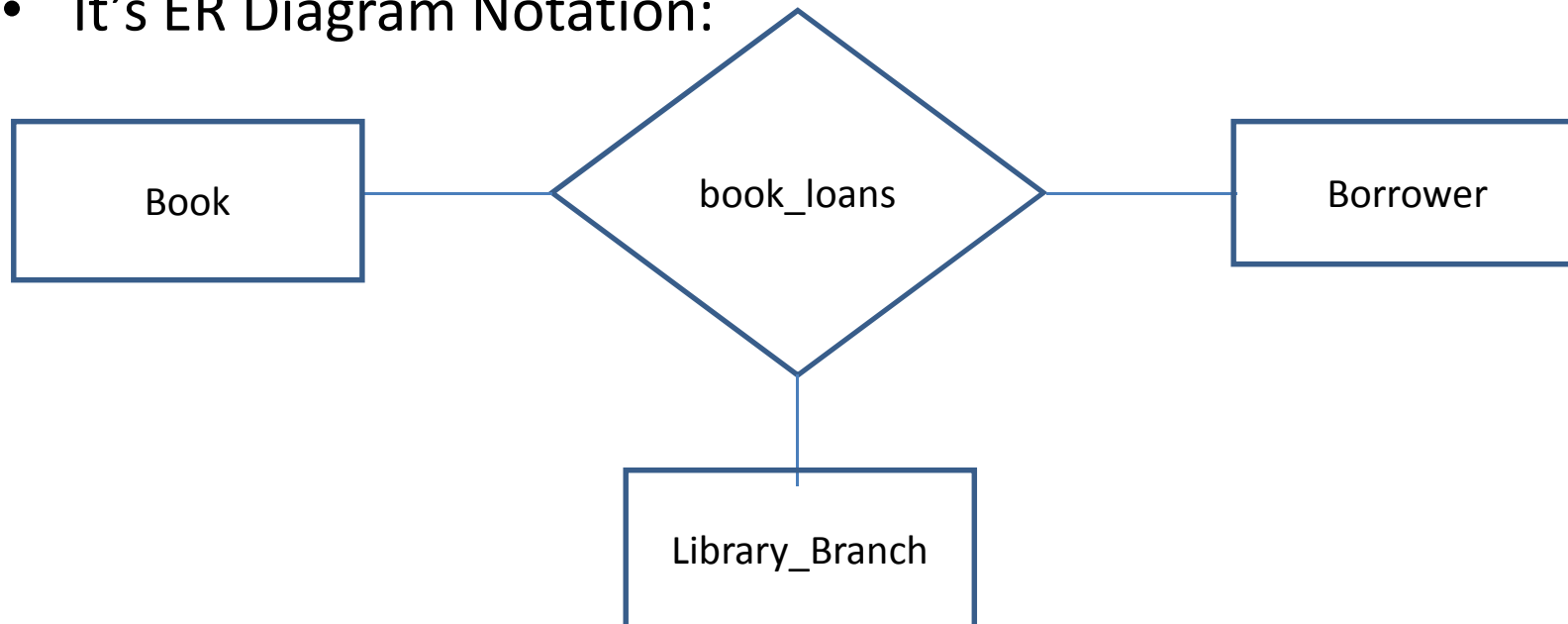
Relationship Degrees

- 1) Binary Relationship: Includes two entity types.
- 2) Example: Employee “works_on” Project.
- 3) Its notation in ER Diagram is as follows.



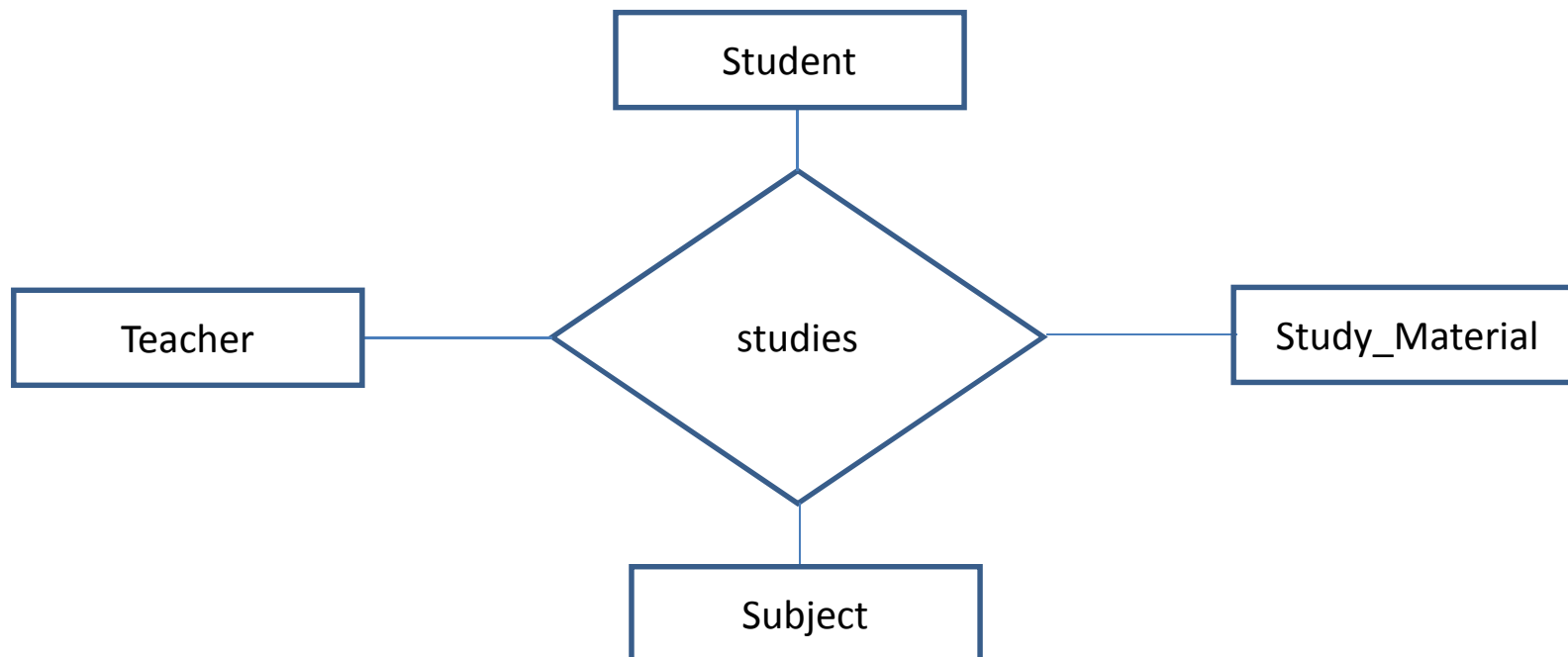
Relationship Degree

- Ternary relationship: includes 3 entity types
- Example: Book_loans: is a relationship that shows:
 - Each borrowed book (**Book**)
 - Who borrowed it (**Borrower**)
 - Which library branch the book was borrowed from (**Library Branch**)
- It's ER Diagram Notation:



Relationship Degrees

- N-ary Relationship: includes N entity types.
- Example: studies: is a 4-ary relationship that shows that a “student” studies a “subject” with a “teacher” and the help of “study material”.

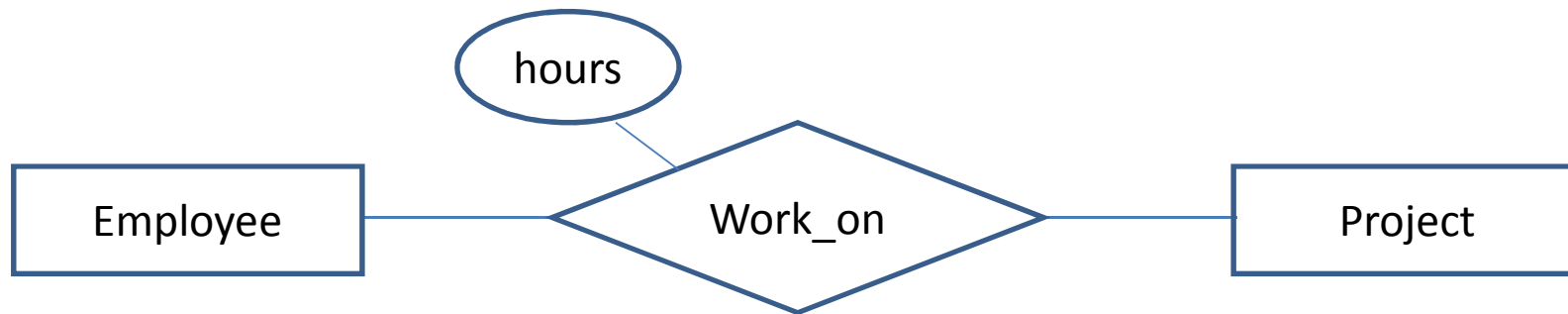


Relationship Attributes

- In some cases, a relationship type can have attributes.
- Usually, in these cases, the attribute does not belong to any of the participating entities (exclusively).
- Because of that, we add the attribute on the relationship.

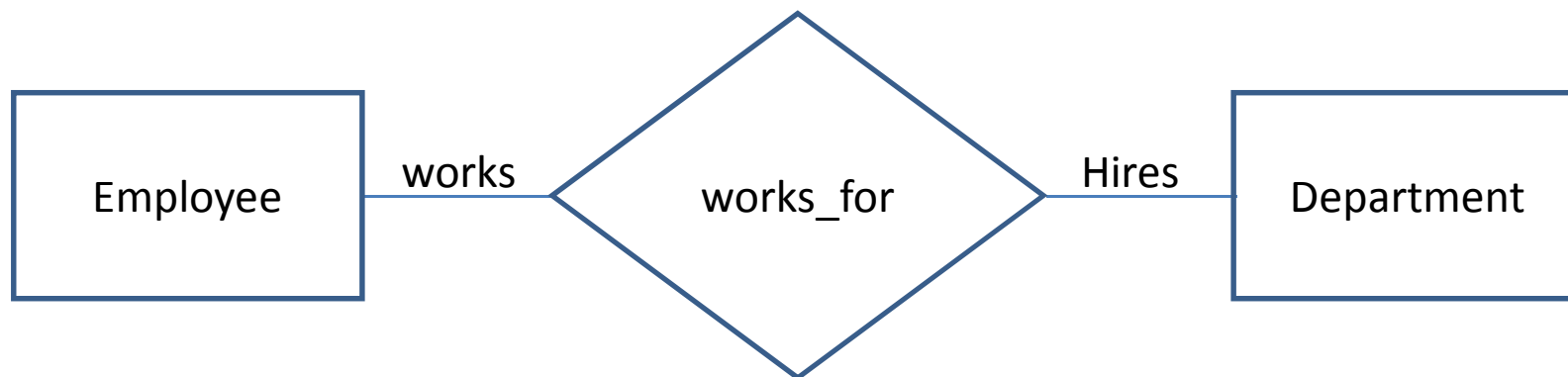
Relationship Attributes

- Examples:
 - start_date: is an attribute that specifies the start date of an employee as a manager of a department. It does not belong to employee or department exclusively. But, it belongs to both of them, therefore we place it on the relationship “manages”.
 - Hours: is an attribute that specifies the number of hours an employee works on a project. It does not belong to employee nor project exclusively. Therefore we place it on relationship “works_on”.



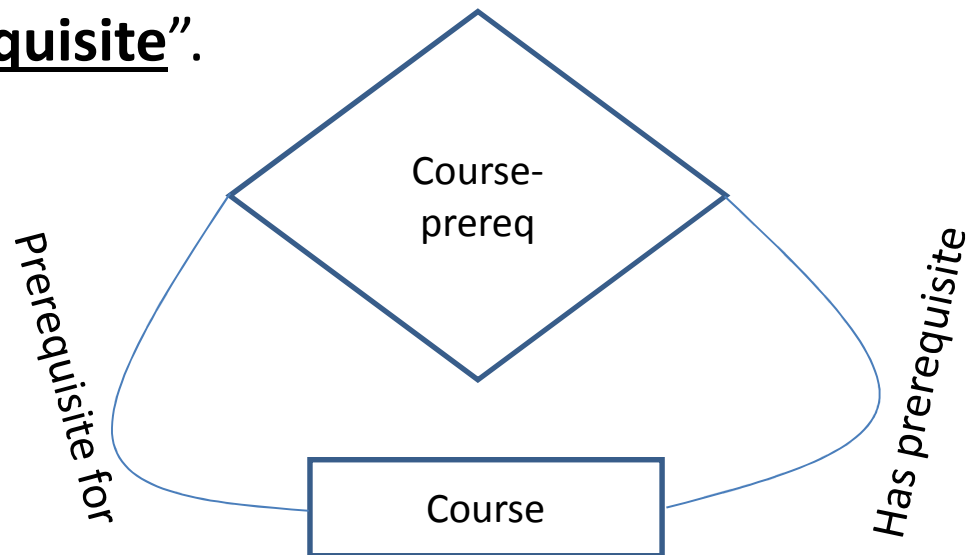
Entity Roles

- In any relationship, entity has a role that specifies what it does in a relationship.
- Example: In Employee “works_for” department relationship:
 - Employee Role: “worker” (works in department)
 - Department Role: “Employer” (employs employee)
- Entity roles can be written on relationship lines in ER Diagram. But they are **implicitly** known, so they are not necessary unless we have a recursive relationship (Look Next Slide).



Recursive Relationship

- Recursive Relationship is a relationship between an entity and itself.
- Example: Course_Prereq is a recursive relationship that shows courses and their prerequisite.
- Notice that entity role is important here because we need to know which course is a “prerequisite for” and which course “has prerequisite”.

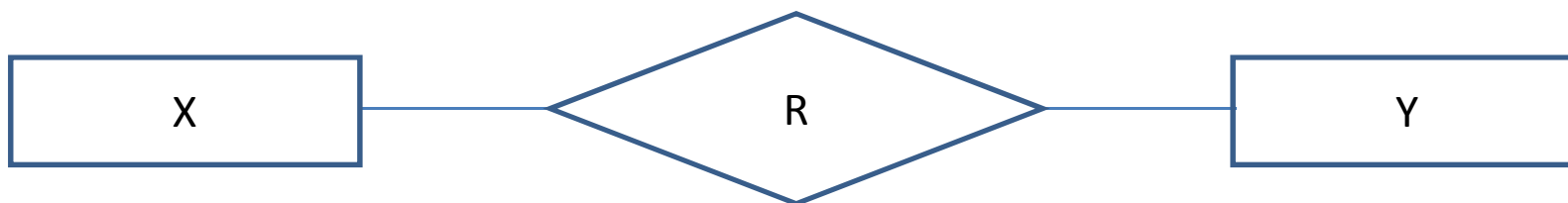


Constraints on Relationships

- Constraints should be reflected in ER diagram.
- They are called **Structural Constraints**.
- For example: Each employee works on “only one project”.
- Types of Structural Constraints:
 - Cardinality Ratio (Maximum Cardinality)
 - Participation (Minimum Cardinality)

Cardinality Ratio (Maximum Cardinality)

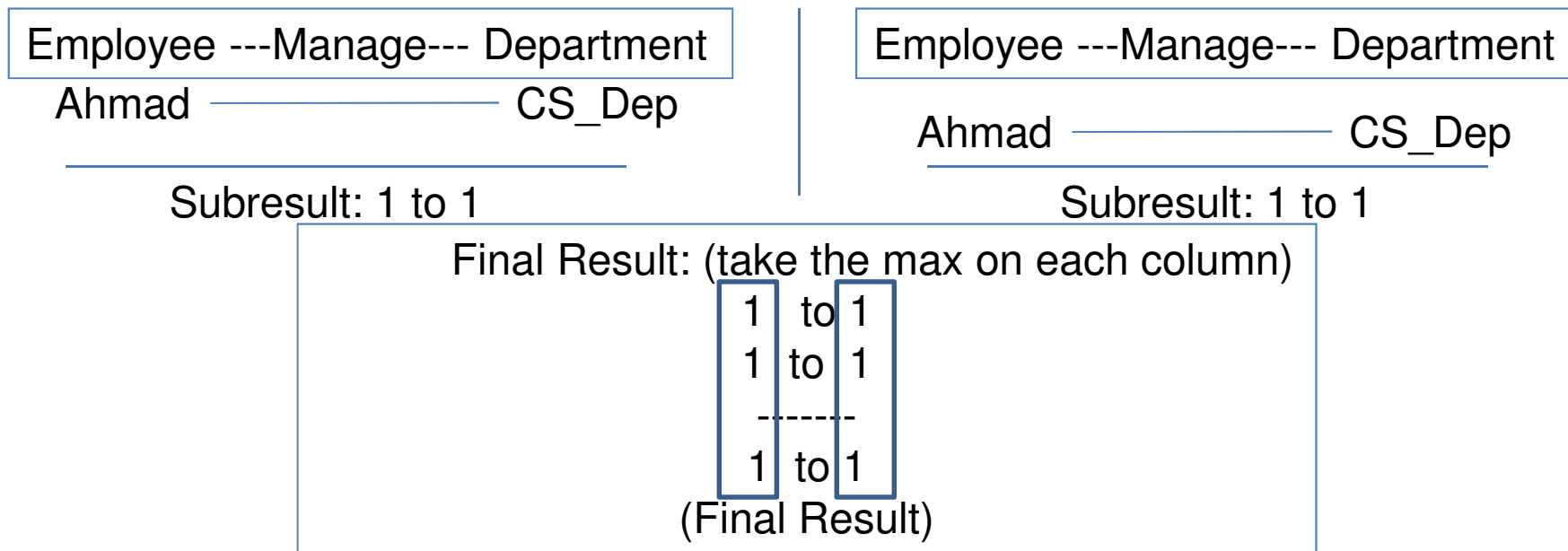
- Cardinality Ratio: is the maximum number of “relationship instances” that an entity can participate in. (Maximum Cardinality)
- Types of cardinality ratio :
 - 1:1 --- It is read as (one to one), 1 instance of entity x can be connected to only 1 instance of entity Y via relationship R and vice versa.
 - 1:N --- It is read as (one to many), 1 instance of entity x can be connected to N instances of entity Y via relationship R.
 - M:N --- It is read as (many to many), M instances of entity x can be connected to N instances of entity Y via relationship R and vice versa.



Cardinality Ratio (Maximum Cardinality) (1:1)

Examples:

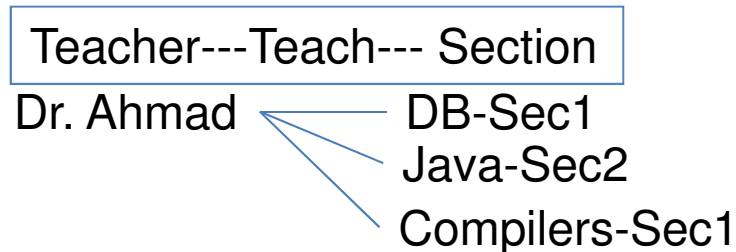
- One department is managed by only One employee.
- One employee can manage only One department.



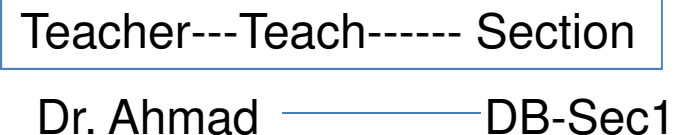
Cardinality Ratio (Maximum Cardinality) (1:N)

Examples:

- **One** teacher can teach **Many** sections
- **One** section is only taught by only **One** teacher



Subresult: 1 to N

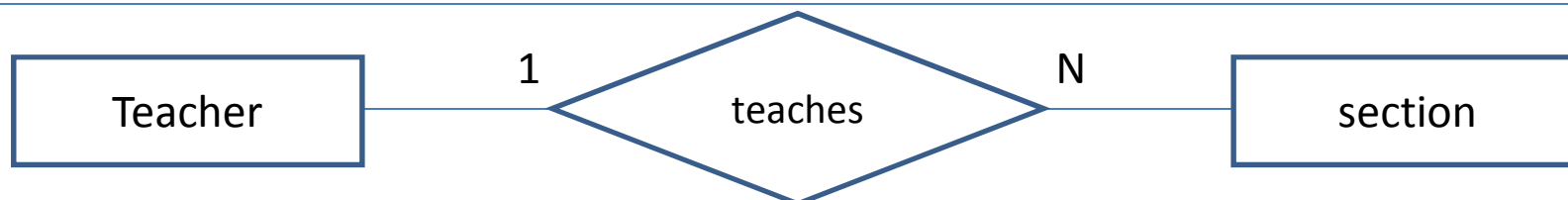


Subresult: 1 to 1

Final Result: (take the max on each column)

1 to N
1 to 1

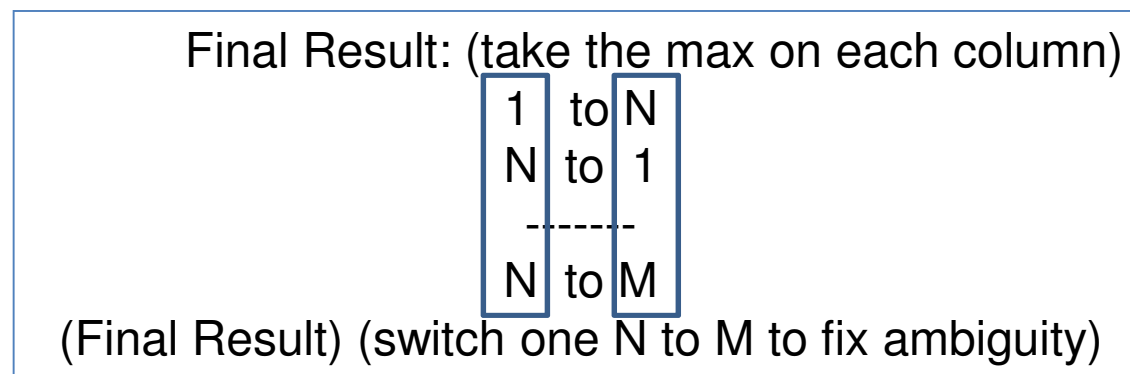
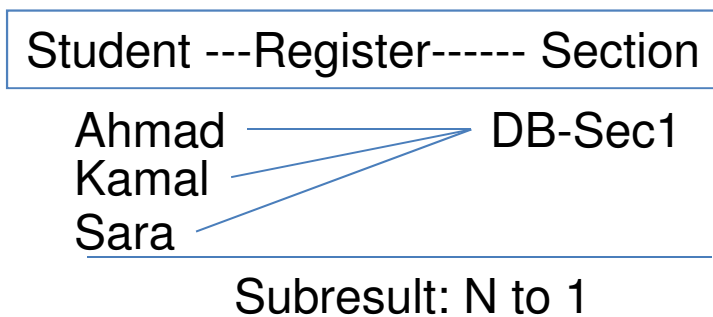
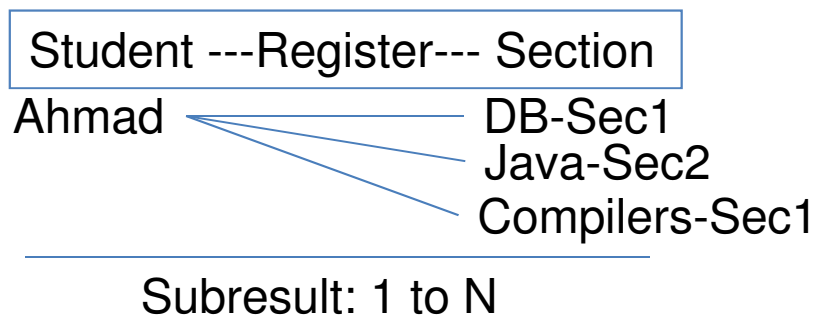
1 to N
(Final Result)



Cardinality Ratio (Maximum Cardinality) (M:N)

Examples:

- **One** student can register for **Many** sections
- **One** section can be registered for by **Many** students



Constraints on Relationships

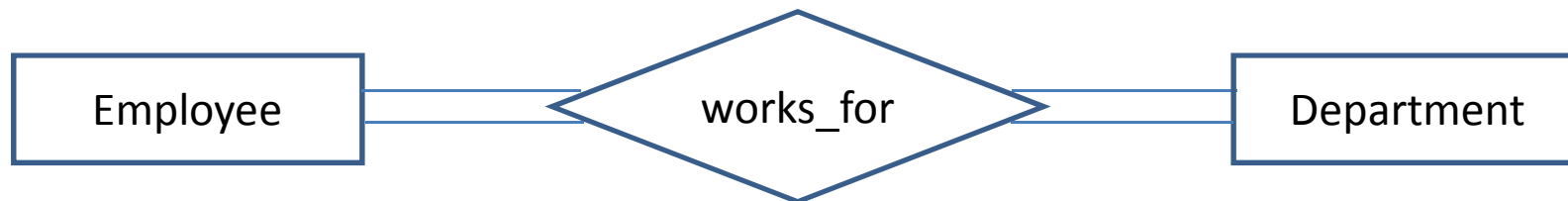
- Types of Structural Constraints:
 - Cardinality Ratio (Maximum Cardinality)
 - We already discussed this one.
 - Participation (Minimum Cardinality)
 - Now, we look into this.

Participation Constraints

- The participation constraint specifies whether the existence of an entity depends on it being related to another entity via the relationship type. This constraint specifies the “minimum” number of relationship instances that each entity can participate in.
- Two types of participation constraints:
 - Total Participation (Existence Dependency)
 - Partial Participation

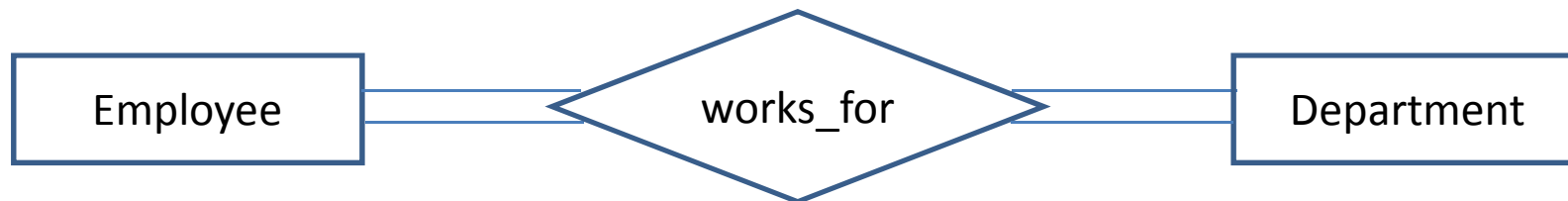
A way to think of participation constraints

- Example 1:
 - Employee “works for” department
 - Total Participation from **Employee side** (how?)
 - Assume that the **company has 3 employees**. Should **all** of the employees belong to at least one department ?
 - If the answer is yes: (Total Participation) (represented by two lines)
 - If the answer is No: (Partial Participation) (represented by one line)
 - In our example, the answer is “yes”



A way to think of participation constraints

- Example 1:
 - Employee “works for” department.
 - Total Participation from department side (how?)
 - Assume that the company has 3 departments. Should all departments have employees working in them ?
 - If the answer is yes: (Total Participation) (represented by two lines)
 - If the answer is No: (Partial Participation) (represented by one line)
 - In our example, the answer is “yes”



A way to think of participation constraints

- Example 2:
 - Employee “manages” department.
 - Partial Participation from **employee side** (how?)
 - Assume that the company has 3 employees. Should **all** employees manage departments ?
 - If the answer is yes: (Total Participation) (represented by two lines)
 - If the answer is No: (Partial Participation) (represented by one line)
 - In our example, the answer is “No” because some employees are not managers.



A way to think of participation constraints

- Example 2:
 - Employee “manages” department
 - Total Participation from **department side** (how?)
 - Assume that **company** has **3 departments**. Should **all** departments be managed by employees?
 - If the answer is yes: (Total Participation) (represented by two lines)
 - If the answer is No: (Partial Participation) (represented by one line)
 - In our example, the answer is “yes”



Weak Entity

- A weak entity: is an entity with a primary key that does not come from its own attributes.
- Strong entity: is an entity that does have a key attribute “from within its own attributes”.
- Weak entity is only identified by being related to another strong entity.
- This kind of relationship is called **identifying relationship**.
- Weak Entities are identified by a combination of:
 - **Partial Key**: Some attributes of weak entity.
 - **Strong Entity Key**: Key of strong entity that defines weak entity.
- **Weak entity key = partial key of weak entity + key of strong entity**

Weak Entity

- Example: Assume that employees can have dependents.
- By dependents we mean “Son”, “Daughter”, “Wife”, ... etc.
- Dependents are only identified through employees they belong to.
- For example: Employee Ahmad has a dependent, his daughter “Sara”. Sara is only identified by being related to Ahmad.
- Dependent can be identified by a combination of:
 - **Partial key**: may be “**First Name**” of dependent, assuming that dependents of the same employee do not have similar first name.
 - **Strong Entity Key**: Key of employee (**employeeID**)
- **Key of dependent is**: dependent name + employee ID

Weak Entity

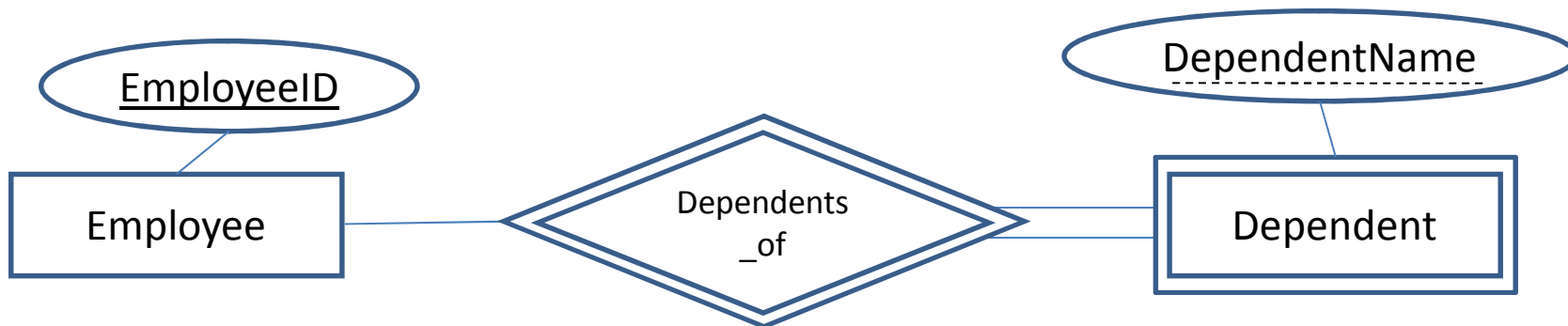
- For example assume employee with ID “365” has two dependents, his **daughter “Sara”** and his **son “Kamal”**. Also employee with ID 300 has a son “Kamal”, then dependents are identified as:

<u>EmployeeID</u>	<u>DependentName</u>	Relationship
365	Sara	Daughter
365	Kamal	Son
300	Kamal	Son

- Notice that partial key cannot work as key by itself.
 - in this example, the two employees have a son named “Kamal”. So, “kamal” (dependent name) cannot be a key for dependent. This is why it is called **partial key**.

Weak Entities

- Weak entities are represented in ER Diagram by a double lines in entity and relationship shapes.
- Also, because weak entities depend on a strong entity in order to exist, then weak entities **always** has “**total participation**” in the **identifying relationship**. (represented by double lines).
- In ER diagram, a **partial key** is underlined with a **dashed** line.



More Examples

(Registration DB)(Entities)

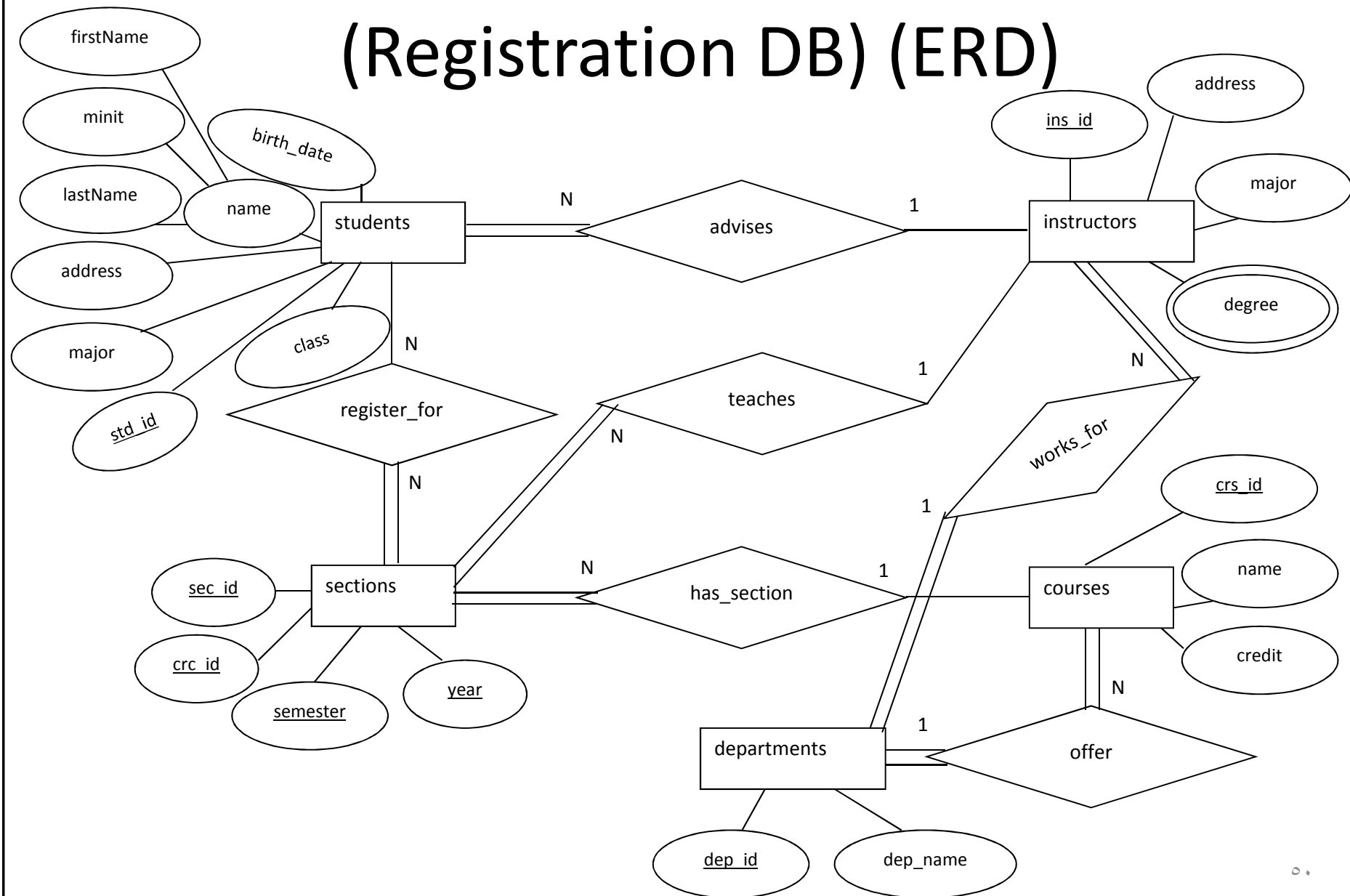
- **Student:** Each student has an id, a name that is composed of a first name, middle initial, and last name. Each student has an address, gender, major, class, and birth date.
- **Course:** Each course has an id, name, and credit hours.
- **Instructor:** Each instructor has an id, a name, address, major, and degree.
- **Department:** Each department has an id, and name.
- **Section:** Each section has an id which is “not” unique among other sections. Each section has a semester and year in which it was offered.

More Examples

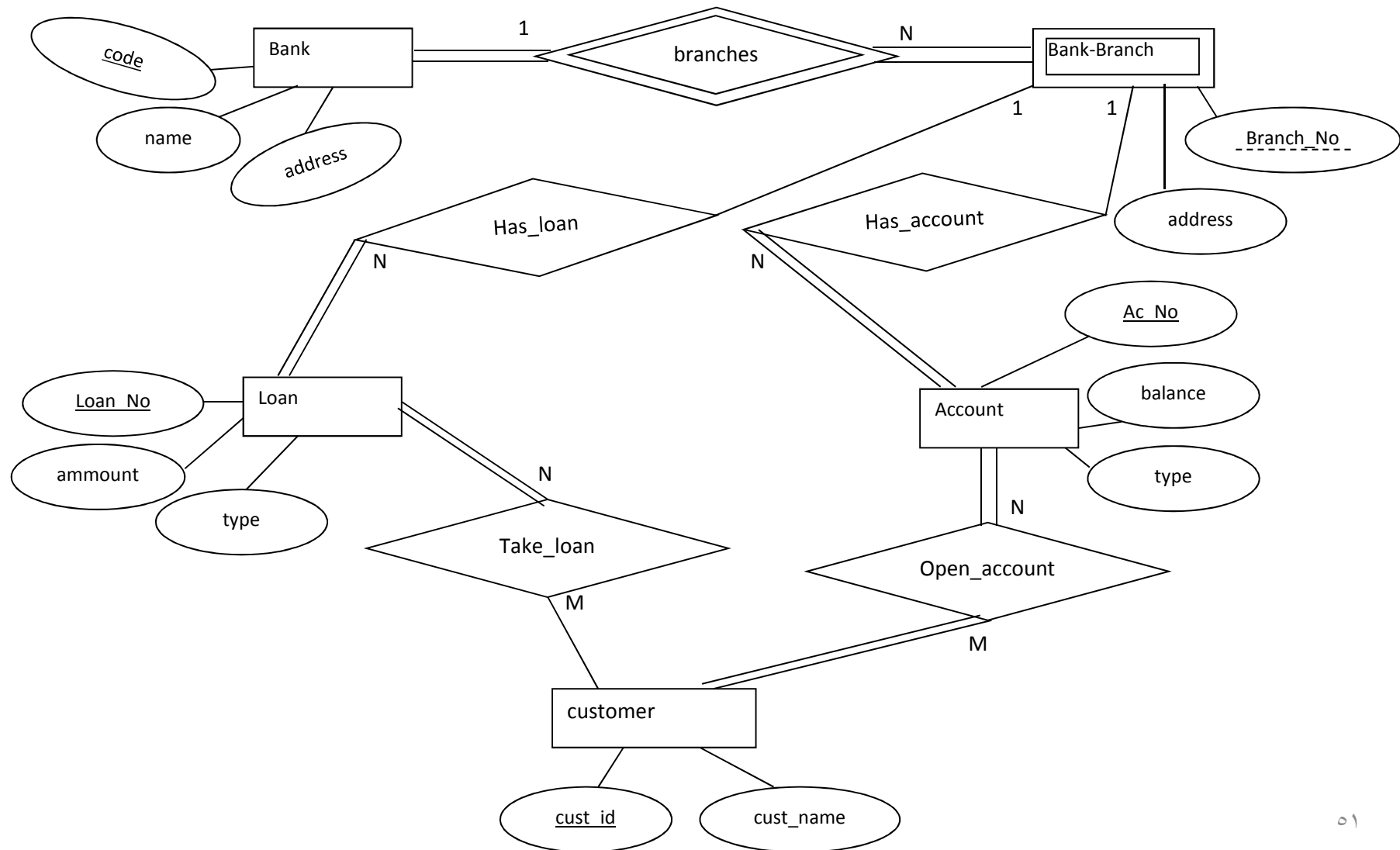
(Registration DB) (Relationship)

- Student “registers-for” Section
- Instructor “teaches” Section
- Course “has” Section
- Course is “offered by” Department
- Student “belongs to” a Department
- Course “belongs to” a Department
- Instructor “belongs to” a Department

More Examples (Registration DB) (ERD)



More Examples (Bank DB) (ERD)



Important Note

- Question: Assume you are designing ERD for one Library. Do you put entity “Library” in ERD?

Answer: **No**, remember that the entity is translated into a table in the database. In this case, the table would have only **one record** for the library (**Not Useful**)

Library

LibraryID	LibraryName
1	Abd Alhameed Shoman



In Class Exercises

- Design ERD for one zoo
- Design ERD for one Library
 - Assume each library has branches

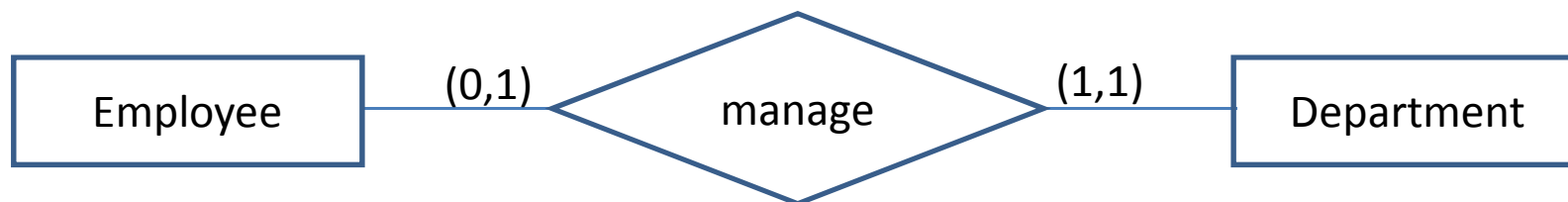
Alternative Notation for Structural Constraints

(Min,Max) Notation

- Instead of “cardinality ratio” & “participation constraints”, we can use (min,max) notation.
- For each participating entity type in a relationship, you specify a pair of numbers (min,max).
- (min,max) indicates the minimum and the maximum number of relationship instances an entity can participate in this relationship.

Alternative Notation for Structural Constraints (Min,Max) Notation

- Example:
 - Employee “manages” department.
 - From the employee side, the notation is (0,1). 0 is the minimum because some employees are **not managers**. 1 is the maximum because 1 given employee can be a manager for **at most 1 department**.
 - From the department side, the notation is (1,1) because each department **must** have a manager, and **at most** it has only **one manager**.



Alternative Notation for Structural Constraints

(Min,Max) Notation

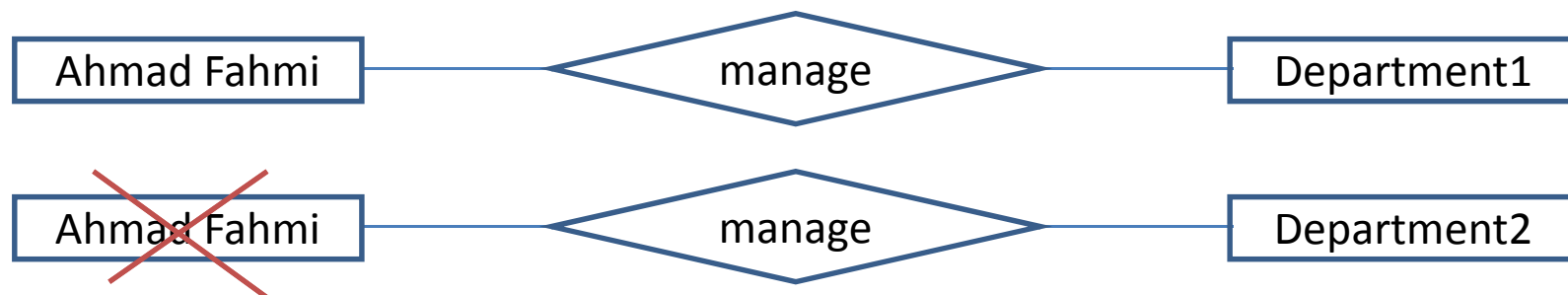
- From “Employee” side:
 - Suppose we have one employee Ahmad Fahmi.
 - What is the minimum number of times Ahmad Fahmi appears in manages relationship?
 - Answer: **zero**, because Ahmad Fahmi might not be a manager.
 - What is the maximum number of times Ahmad Fahmi appears in manages relationship?
 - Answer: **1**, because Ahmad Fahmi can be a manager for only one department.
 - Therefore, the final result from Employee side is (0,1)

Alternative Notation for Structural Constraints (Min,Max) Notation



- **Minimum** number of times for **Ahmad Fahmi** is zero because he might not be a manager

In this slide, we focus only on employee



- **Maximum** number of times for **Ahmad Fahmi** is 1 because he cannot be a manager for more than one department.

Alternative Notation for Structural Constraints

(Min,Max) Notation

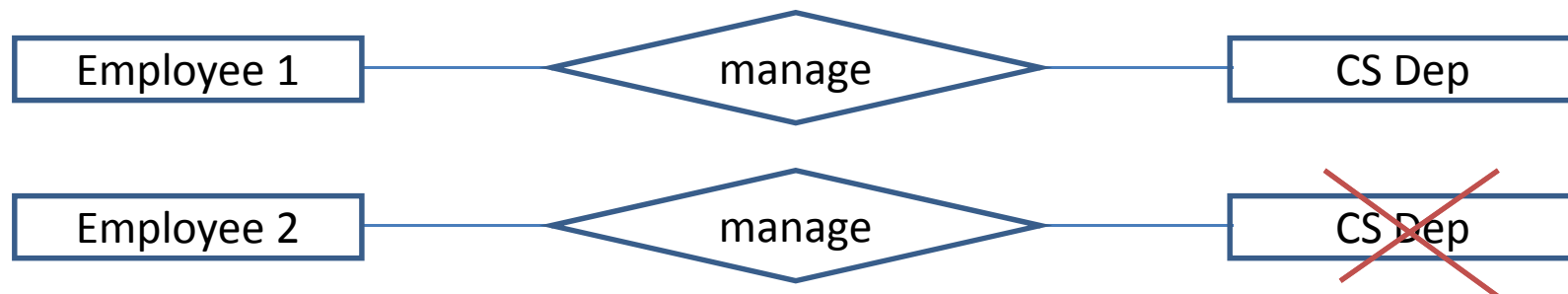
- From “Department” side:
 - Suppose we have one department CS Dep.
 - What is the minimum number of times CS Dep appears in manages relationship?
 - Answer: **1**, because CS Dep must have a manager.
 - What is the maximum number of times CS Dep appears in manages relationship?
 - Answer: **1**, because CS Dep can have only one manager.
 - Therefore, the final result from Department side is (1,1)

Alternative Notation for Structural Constraints (Min,Max) Notation



- **Minimum** number of times for **CS Dep** is 1 because it must have a manager

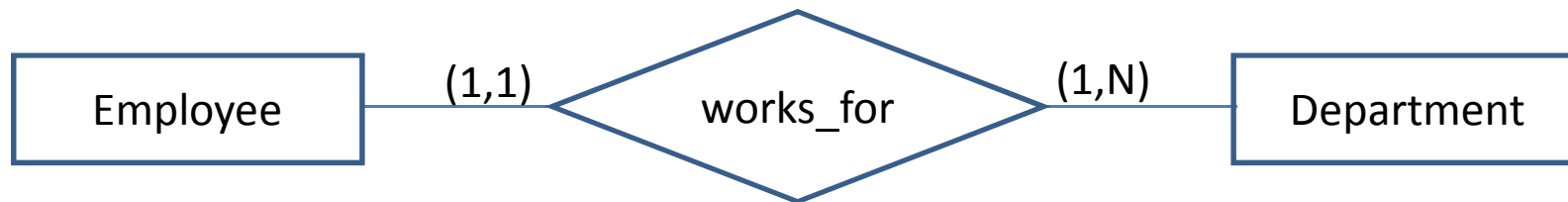
In this slide, we focus only on department



- **Maximum** number of times for **CS Dep** is 1 because it can have at most 1 manager

Alternative Notation for Structural Constraints (Min,Max) Notation

- Example:
 - Employee “works_for” department
 - From the employee side, the notation is (1,1). 1 is the minimum because each employees **must work** in a department. 1 is the maximum because 1 given employee can **work only in 1 department.**
 - From the department side, the notation is (1,n) because each department **must** have **at least** one employee, and **at most** it is can have n employees.

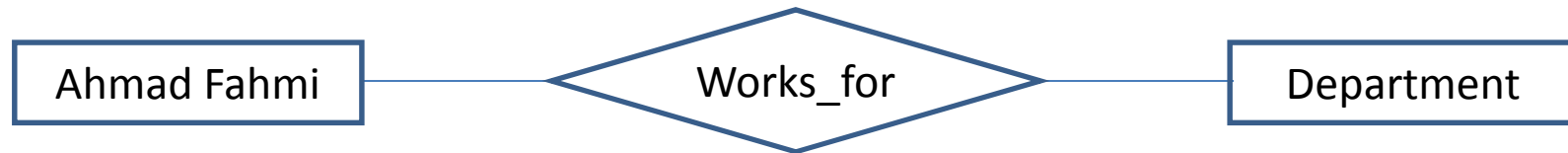


Alternative Notation for Structural Constraints

(Min,Max) Notation

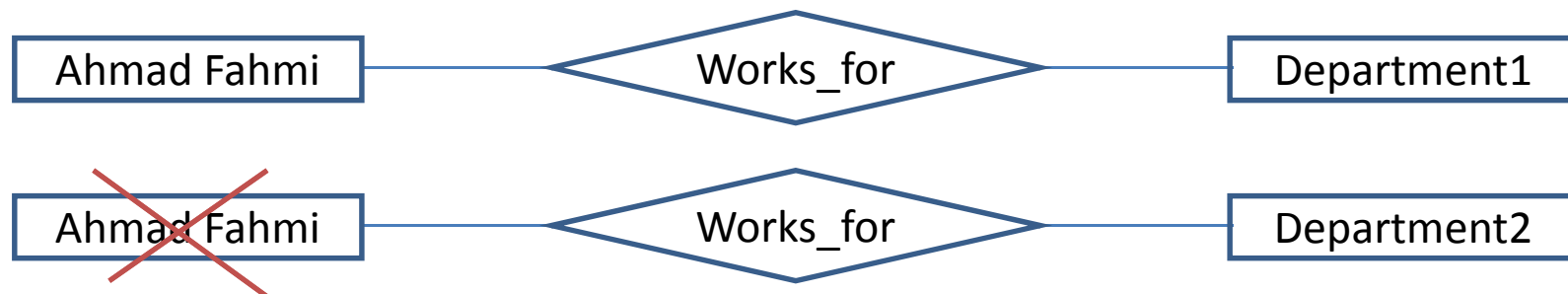
- From “Employee” side:
 - Suppose we have one employee Ahmad Fahmi.
 - What is the minimum number of times Ahmad Fahmi appears in work_for relationship?
 - Answer: **1**, because Ahmad Fahmi must work in some department.
 - What is the maximum number of times Ahmad Fahmi appears in works_for relationship?
 - Answer: **1**, because Ahmad Fahmi can can work for only one department.
 - Therefore, the final result from Employee side is (1,1)

Alternative Notation for Structural Constraints (Min,Max) Notation



- **Minimum** number of times for **Ahmad Fahmi** is 1 because he must work in at least 1 department

In this slide, we focus only on employee



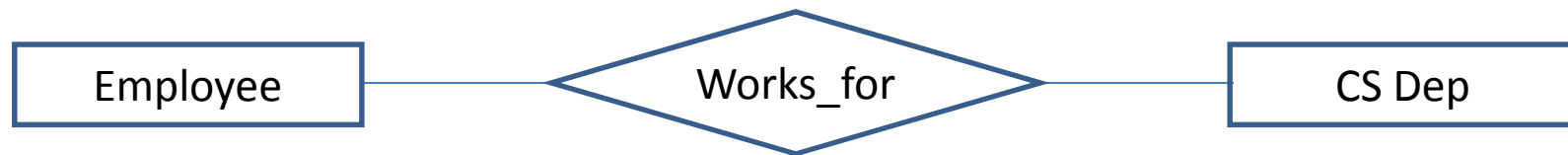
- **Maximum** number of times for **Ahmad Fahmi** is 1 because he can work for at most one department

Alternative Notation for Structural Constraints

(Min,Max) Notation

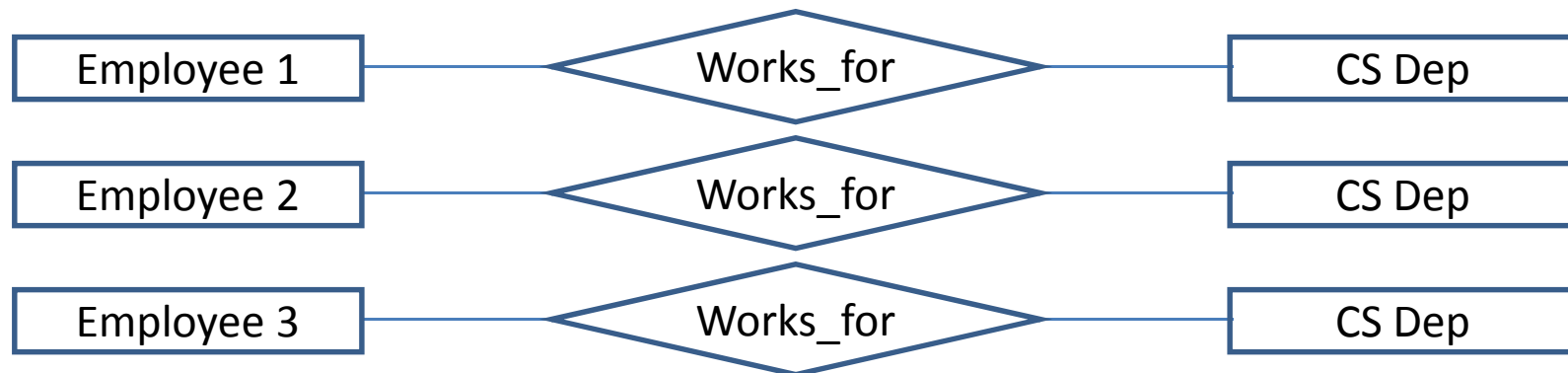
- From “Department” side:
 - Suppose we have one department CS Dep.
 - What is the minimum number of times CS Dep appears in works for relationship?
 - Answer: **1**, because CS Dep must have at least one employee.
 - What is the maximum number of times CS Dep appears in works for relationship?
 - Answer: **1**, because CS Dep can have many employees.
 - Therefore, the final result from Department side is (1,N)

Alternative Notation for Structural Constraints (Min,Max) Notation



- **Minimum** number of times for **CS Dep** is 1 because it must have at least one employee.

In this slide, we focus only on department



- **Maximum** number of times for **CS Dep** is "N" because at most "N" employees can work in CS Dep.