Normalisation | ER diagram

Satyam Govila

Analysing Database Design

Let’s consider a single large dataset having only single relation.

This large database defined as a single relation may result in data duplication.

Can you think of the disadvantages of having a large database with repetitive data?

Analysing Database Design

This repetition of data may result in:

- Making relations very large.
- Difficult to maintain and update data as it would involve searching many records in relation.
- Wastage and poor utilisation of disk space and resources.
- The likelihood of errors and inconsistencies increases.

How should we handle this problem?

Solution : Normalization

What is Normalization?

- Normalization is a process of decomposing the relations into smaller, simpler, and well-structured relations with fewer attributes.

- It is the process of organising the data in the database.

- It is used to minimise the data redundancy from a relation or set of relations and is also used to eliminate undesirable characteristics like Insertion, Update, and Deletion Anomalies.

- Normalization consists of a series of guidelines that helps to guide you in creating a good database structure.

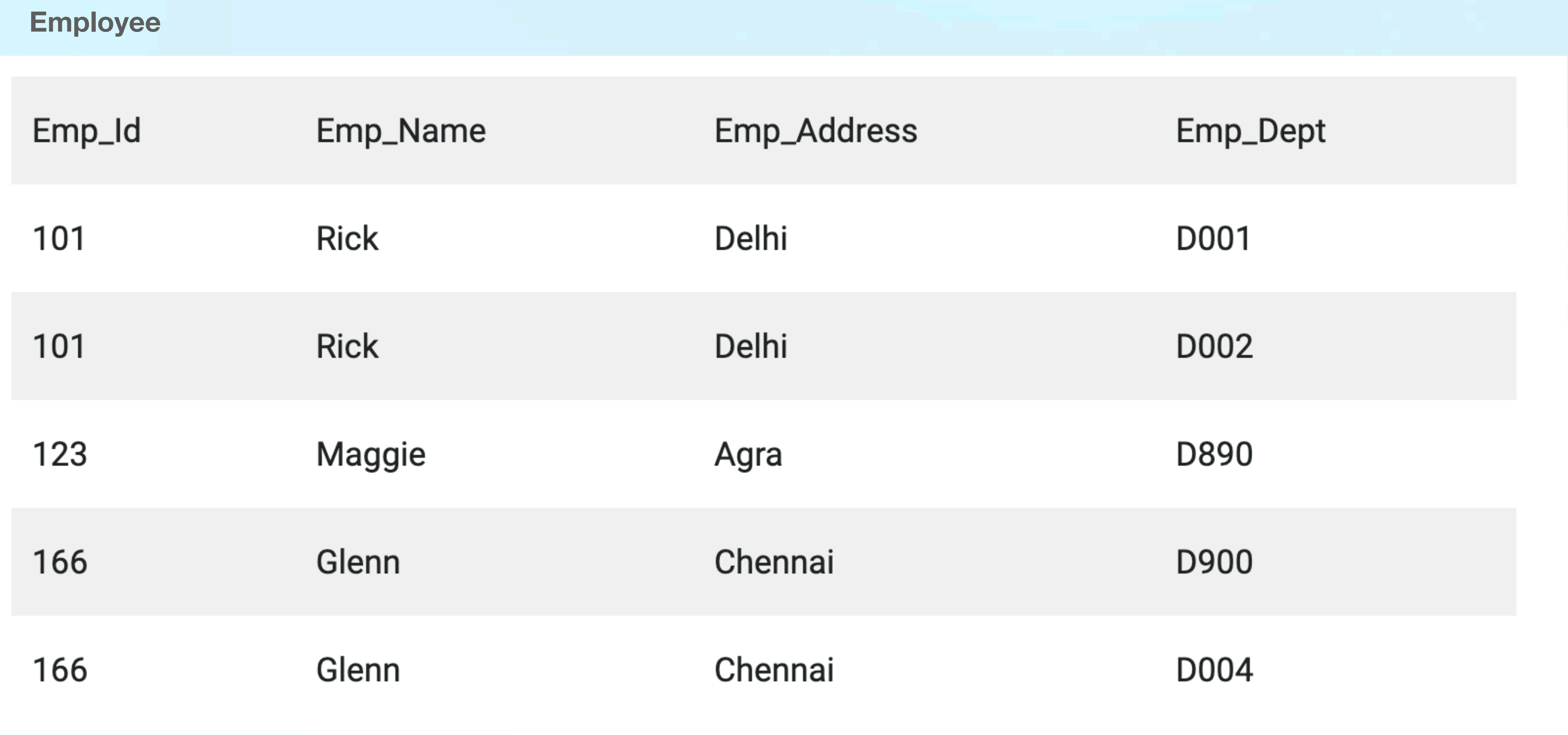
Anomalies in DBMS

Data modification anomalies can be categorised into three types:

- **Insertion Anomaly:** Insertion Anomaly refers to when one cannot insert a new tuple into a relationship due to lack of data.
- **Deletion Anomaly:** The delete anomaly refers to the situation where the deletion of data results in the unintended loss of some other important data.
- **Updatation Anomaly:** The update anomaly is when an update of a single data value requires multiple rows of data to be updated.

Let's understand anomalies with the help of example —>

Anomalies in DBMS



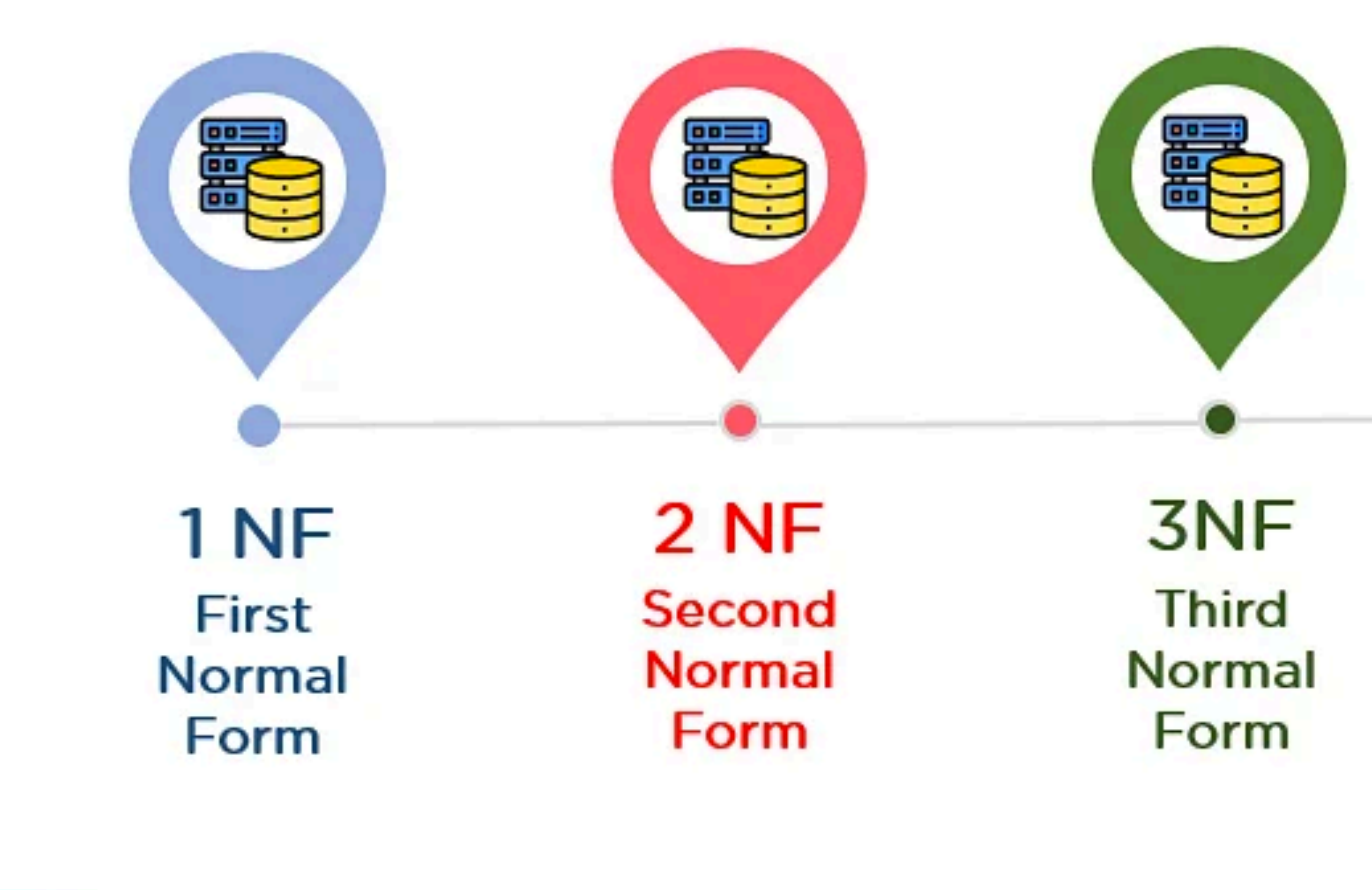
Employee

Emp_Id	Emp_Name	Emp_Address	Emp_Dept
101	Rick	Delhi	D001
101	Rick	Delhi	D002
123	Maggie	Agra	D890
166	Glenn	Chennai	D900
166	Glenn	Chennai	D004

Types of Anomalies:-

1. Update anomaly
2. Insert anomaly
3. Delete anomaly

Types of Normal Form



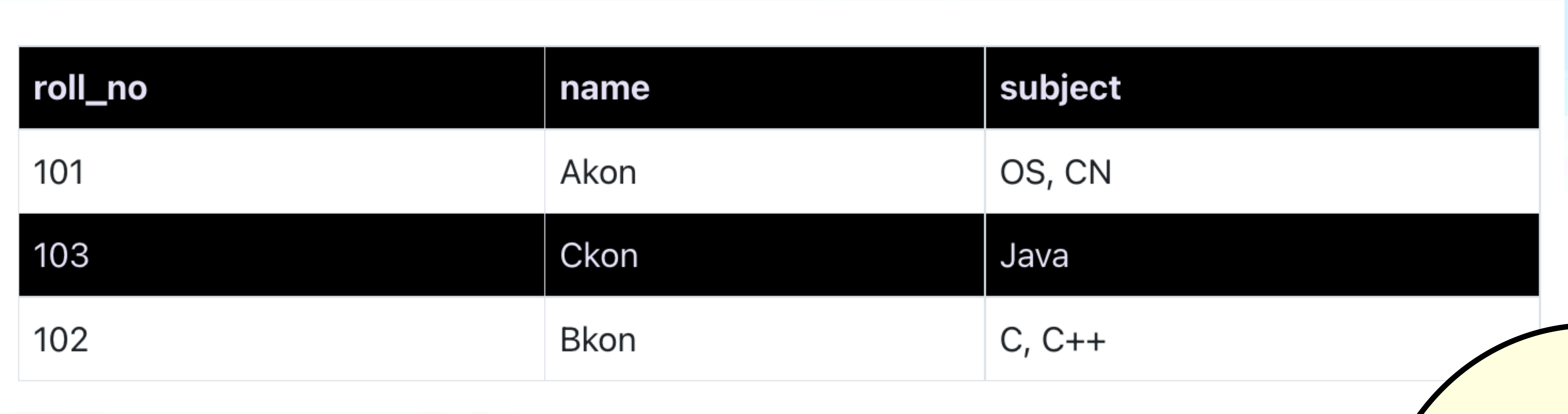
First Normal Form (1NF)

For a table to be in the First Normal Form, it should follow the following 4 rules:

1. It should only have single(atomic) valued attributes/columns.
2. Values stored in a column should be of the same domain
3. All the columns in a table should have unique names.
4. Order in which data is stored, does not matter.

1NF Example

Is the following table in 1NF?



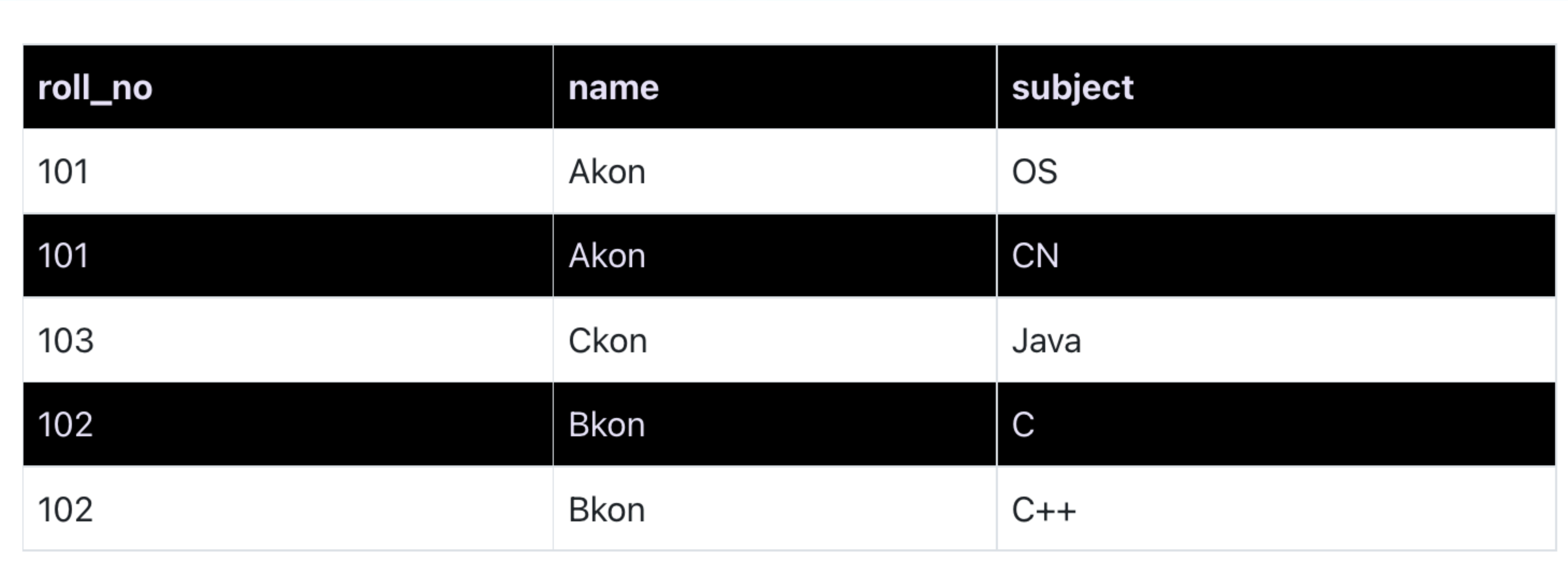
roll_no	name	subject
101	Akon	OS, CN
103	Ckon	Java
102	Bkon	C, C++

Rules for 1NF:-

1. **Single Valued Attributes**
2. **Attribute Domain should not change**
3. **Unique name for Attributes/Columns**
4. **Order doesn't matters**

1NF Example

Converting Table to 1NF…



roll_no	name	subject
101	Akon	OS
101	Akon	CN
103	Ckon	Java
102	Bkon	C
102	Bkon	C++

Second Normal Form (2NF)

For a table to be in the Second Normal Form,

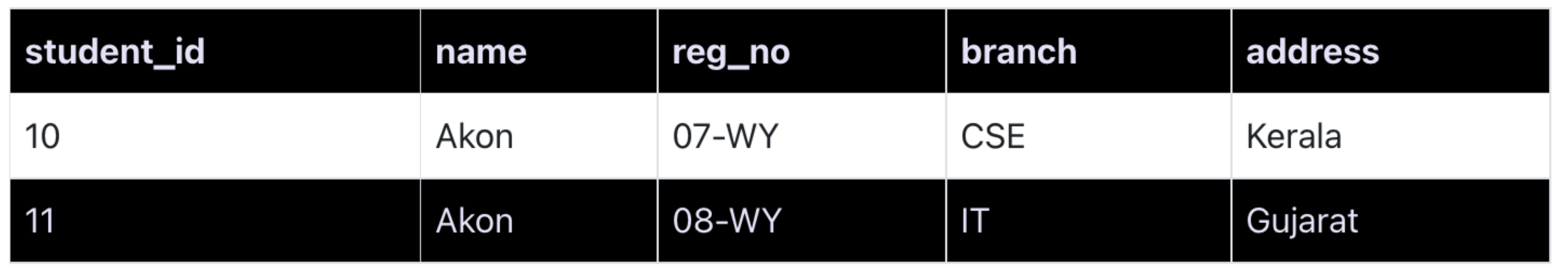
1. It should be in the First Normal form.
2. No Partial Dependency.

But what is partial dependency?

Partial Dependency

Let’s understand “dependency” first…

Table Name : Students
Primary Key : student_id



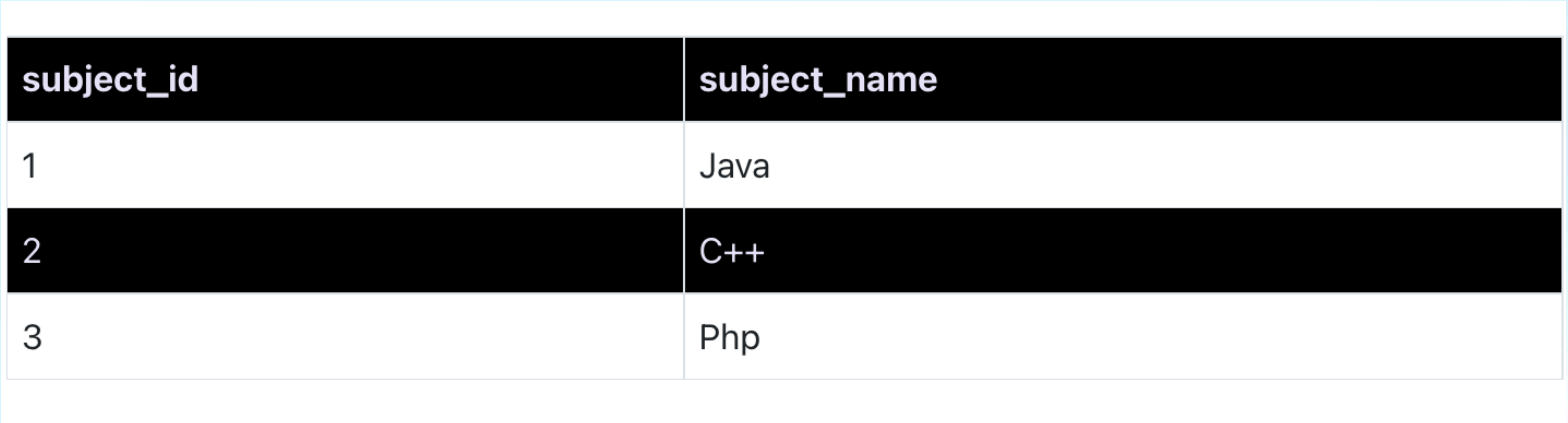
student_id	name	reg_no	branch	address
10	Akon	07-WY	CSE	Kerala
11	Akon	08-WY	IT	Gujarat

student_id —> name
student_id —> name, branch
student_id —> name,branch , address

Every Column is dependent on student_id, hence it is known as Dependency or mainly **Functional Dependency(FD)**

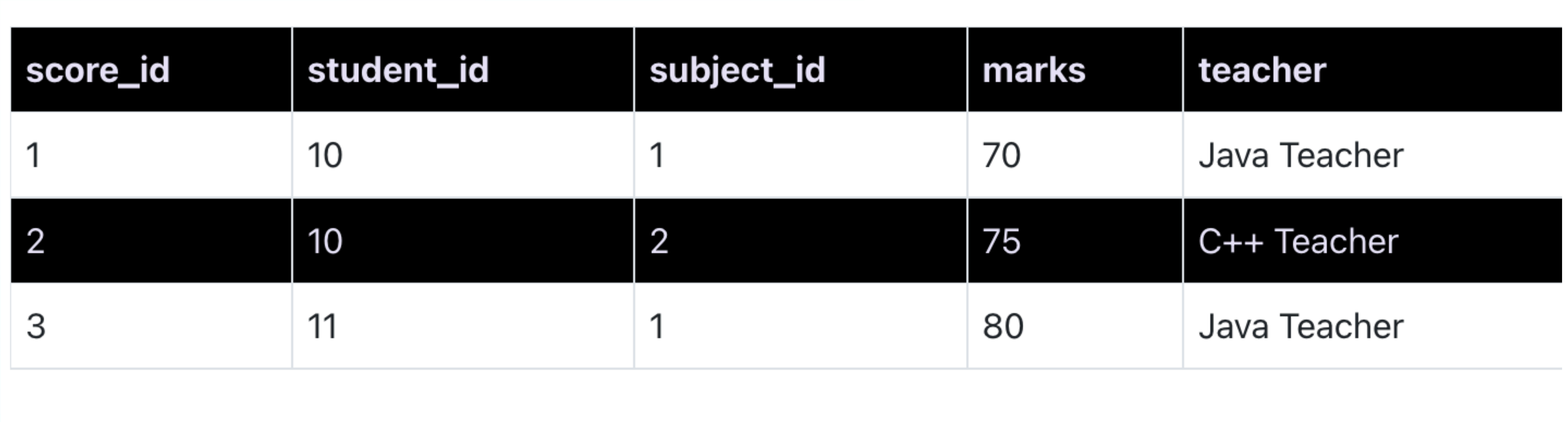
Partial Dependency

Subject



subject_id	subject_name
1	Java
2	C++
3	Php

Marks/Score



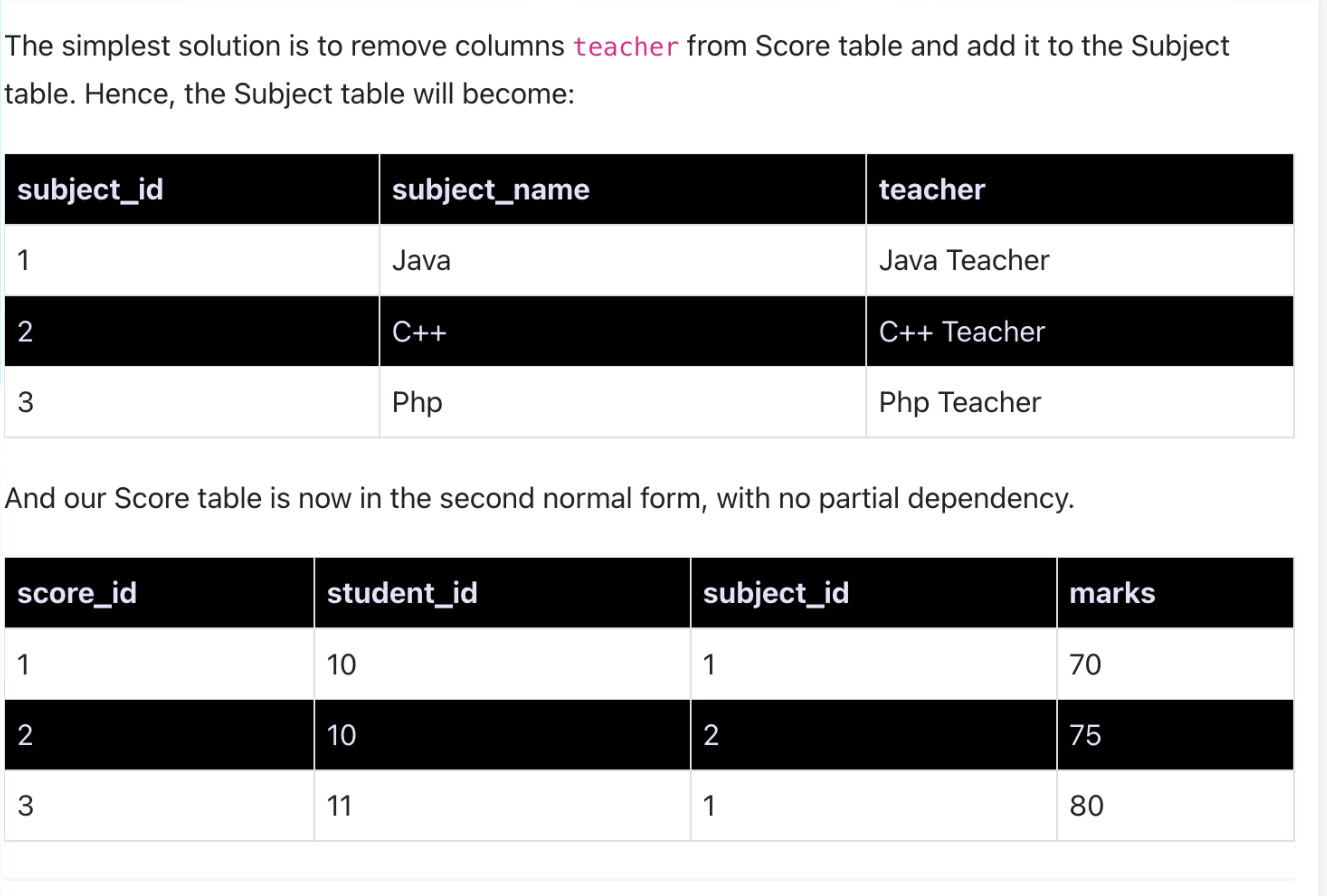
score_id	student_id	subject_id	marks	teacher
1	10	1	70	Java Teacher
2	10	2	75	C++ Teacher
3	11	1	80	Java Teacher

- student_id + subject_id —> marks (Candidate key or Primary key)
- subject_id + student_id —> teacher
- subject_id —> teacher

*This is **Partial Dependency**, where an attribute in a table*

depends on only a part of the primary key and not on the whole key.

Remove Partial Dependency



The simplest solution is to remove columns `teacher` from Score table and add it to the Subject table. Hence, the Subject table will become:

subject_id	subject_name	teacher
1	Java	Java Teacher
2	C++	C++ Teacher
3	Php	Php Teacher

And our Score table is now in the second normal form, with no partial dependency.

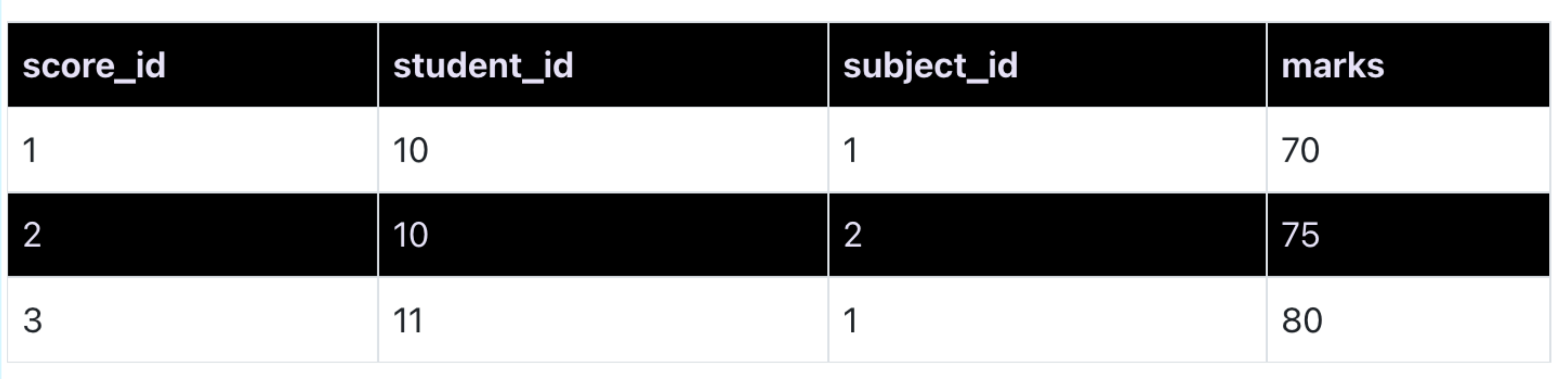
score_id	student_id	subject_id	marks
1	10	1	70
2	10	2	75
3	11	1	80

Third Normal Form (3NF)

For a table to be in the Third Normal Form:

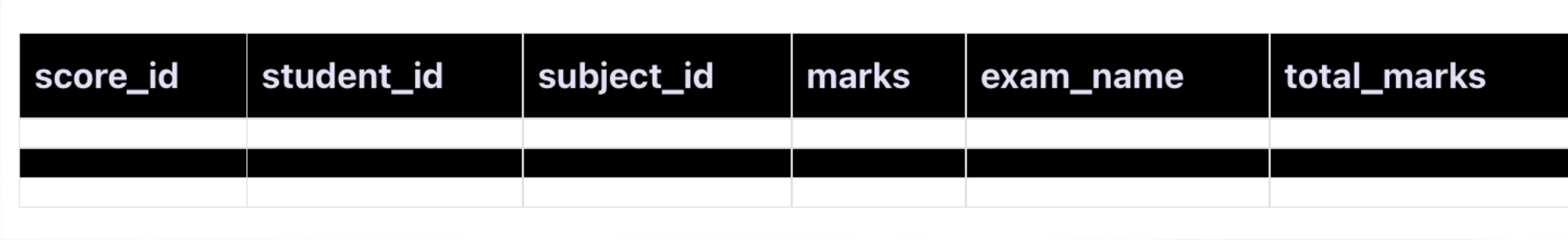
1. It is in the Second Normal form.
2. No Transitive Dependency.

Transitive Dependency



score_id	student_id	subject_id	marks
1	10	1	70
2	10	2	75
3	11	1	80

Add two columns →



score_id	student_id	subject_id	marks	exam_name	total_marks

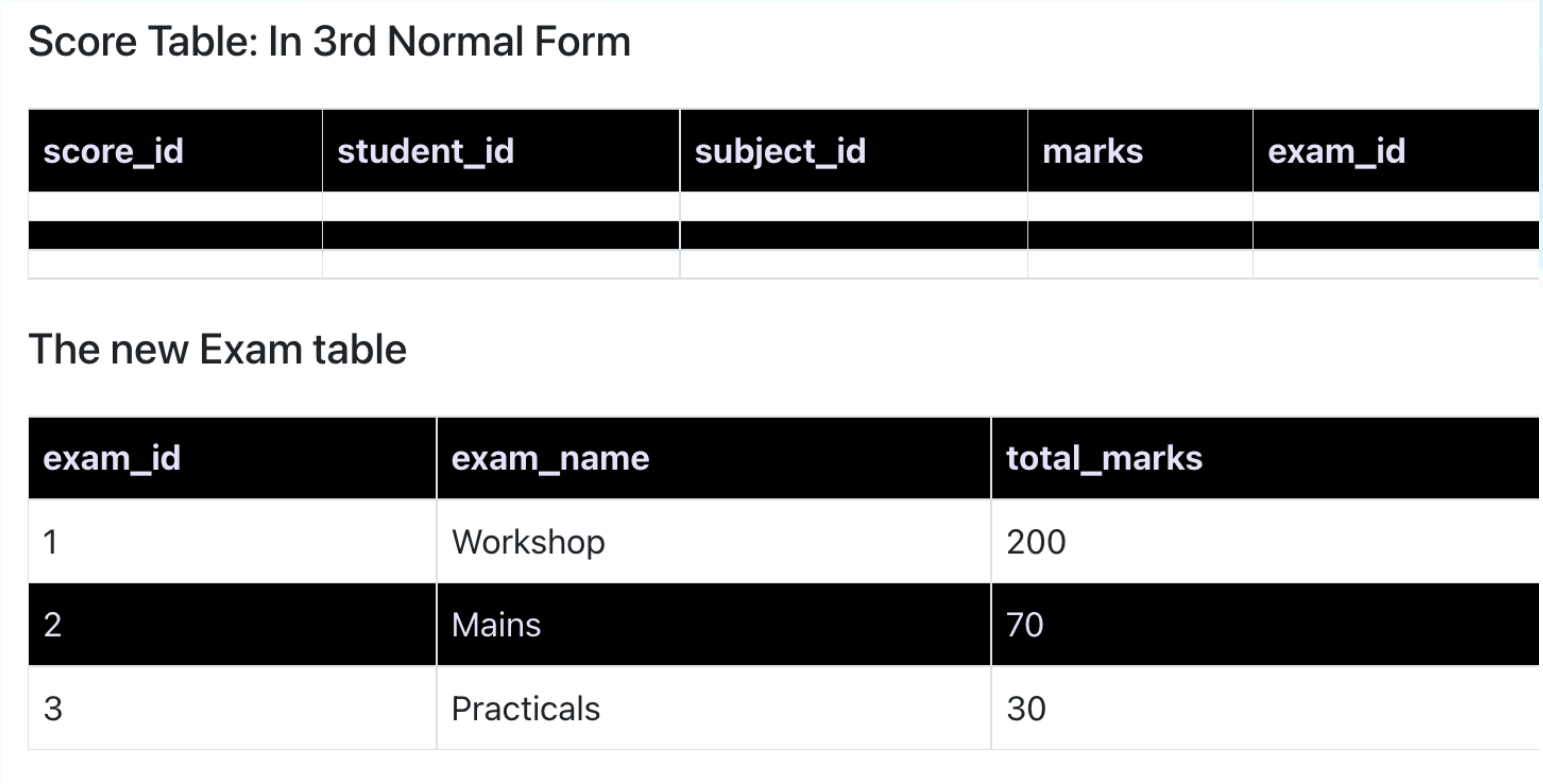
exam_name depends on student_id + subject_id

(Non prime key attribute depends on Primary Key)

total_marks depends on exam_marks

(Non-prime key attribute depends on other Non-Primary Key) —> Transitive Dependency

Remove Transitive Dependency



Score Table: In 3rd Normal Form

score_id	student_id	subject_id	marks	exam_id

The new Exam table

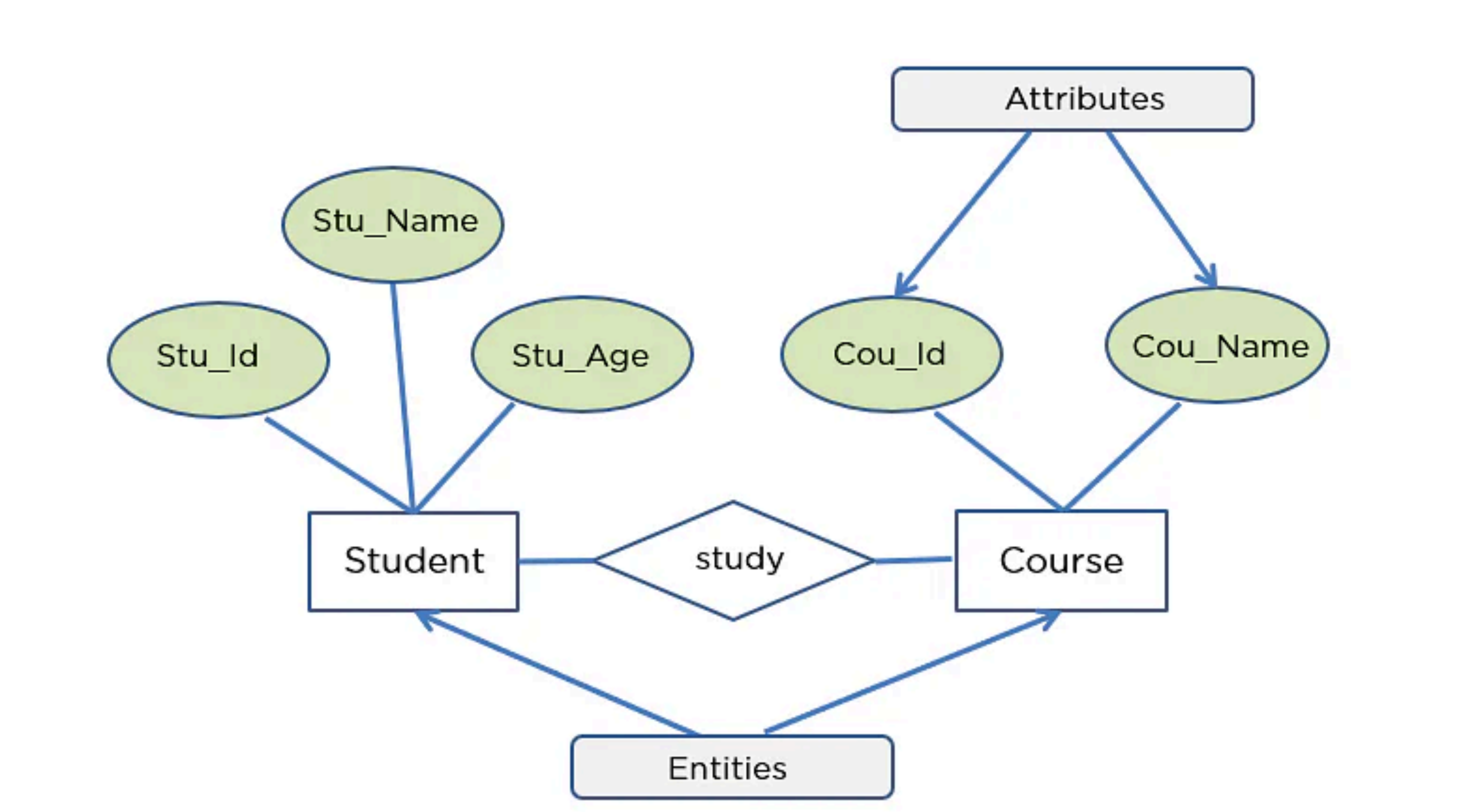
exam_id	exam_name	total_marks
1	Workshop	200
2	Mains	70
3	Practicals	30

ER Model

ER Model

- The ER model defines the conceptual view of a database.
- It works around real-world entities and the associations among them.
- At view level, the ER model is considered a good option for designing databases.

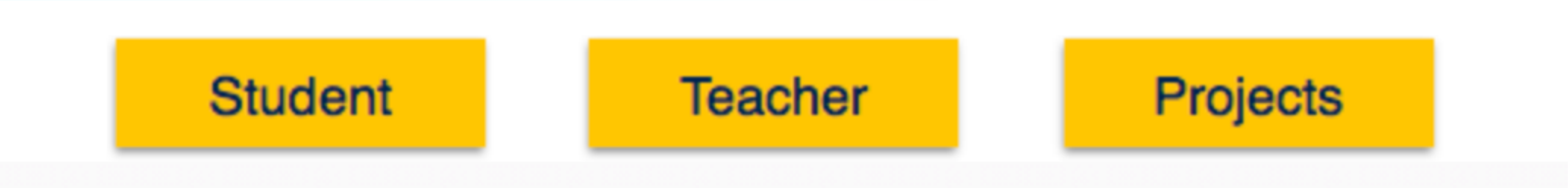
ER Model


Attributes
Stu_Name
Stu_Id
Stu_Age
Cou_Id
Cou_Name
Student
study
Course
Entities

ER Diagram Representation

Entity

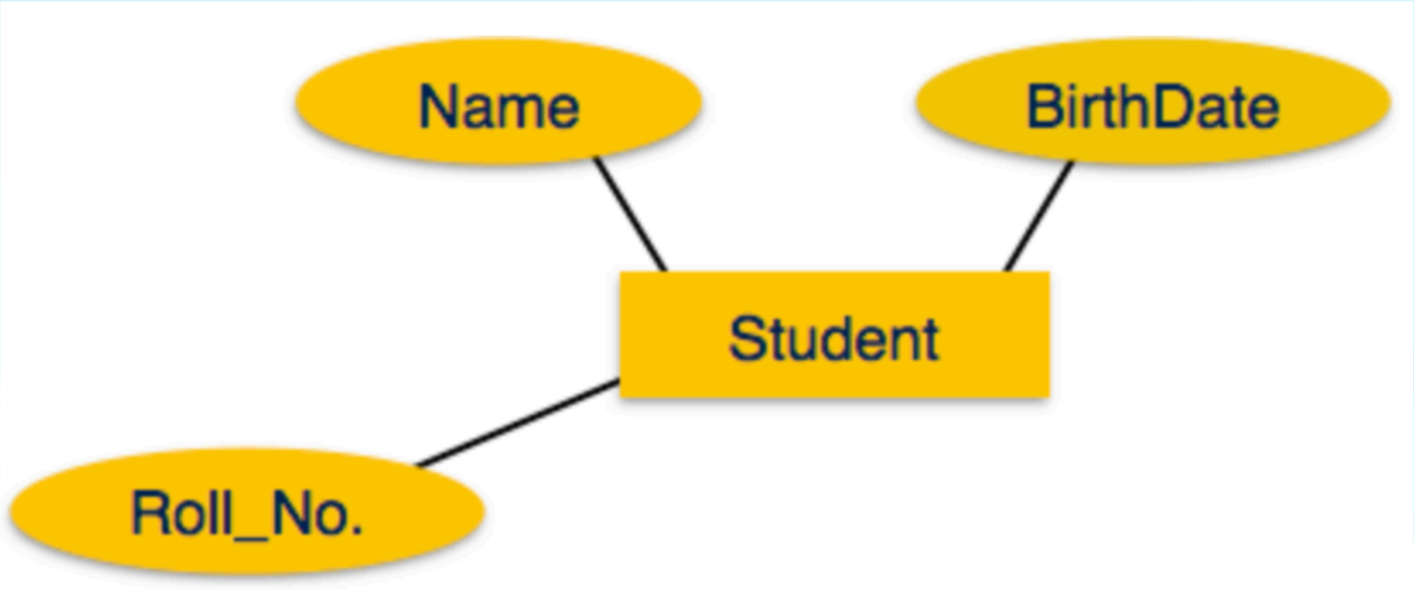
Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.

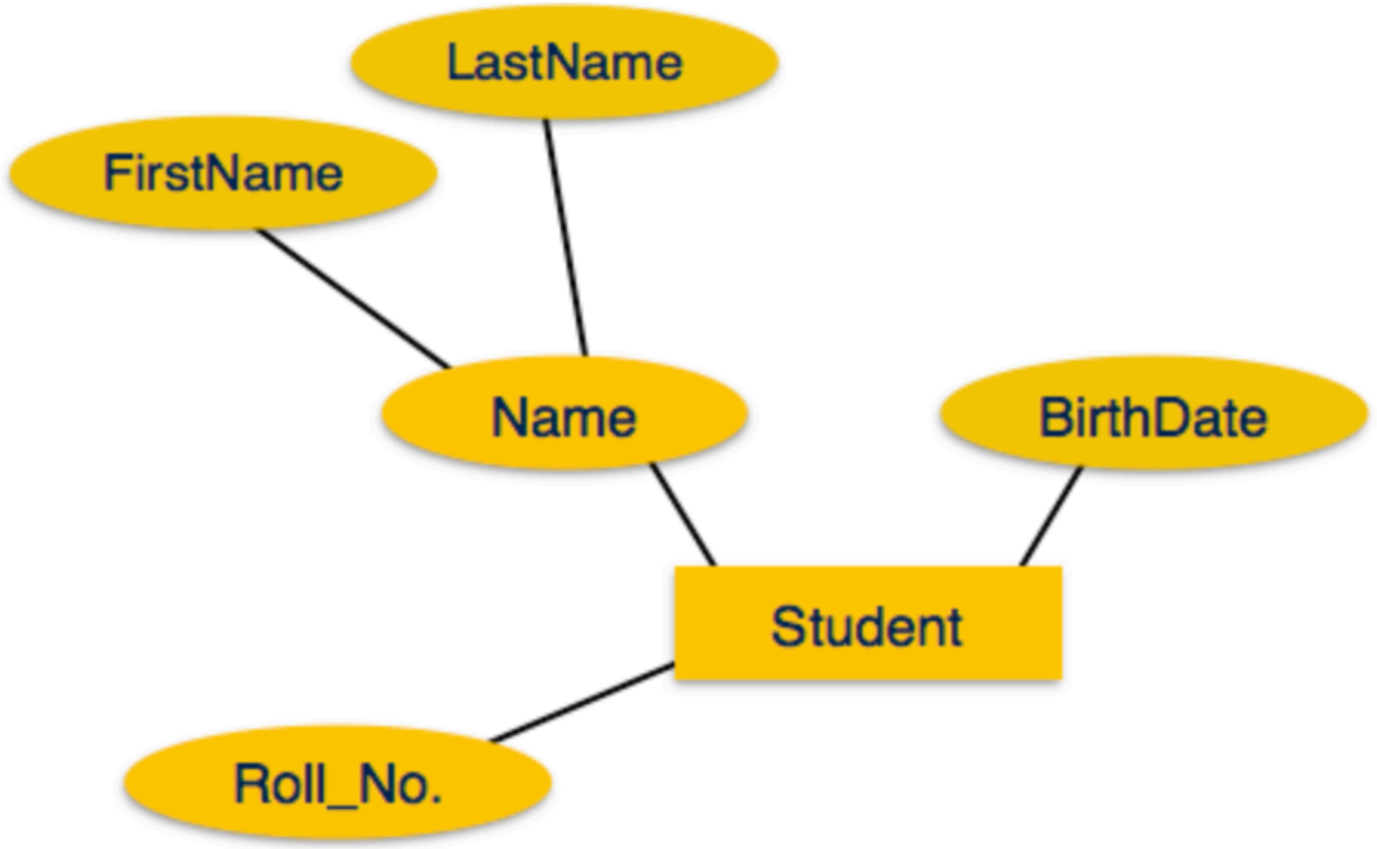


ER Diagram Representation

Attributes

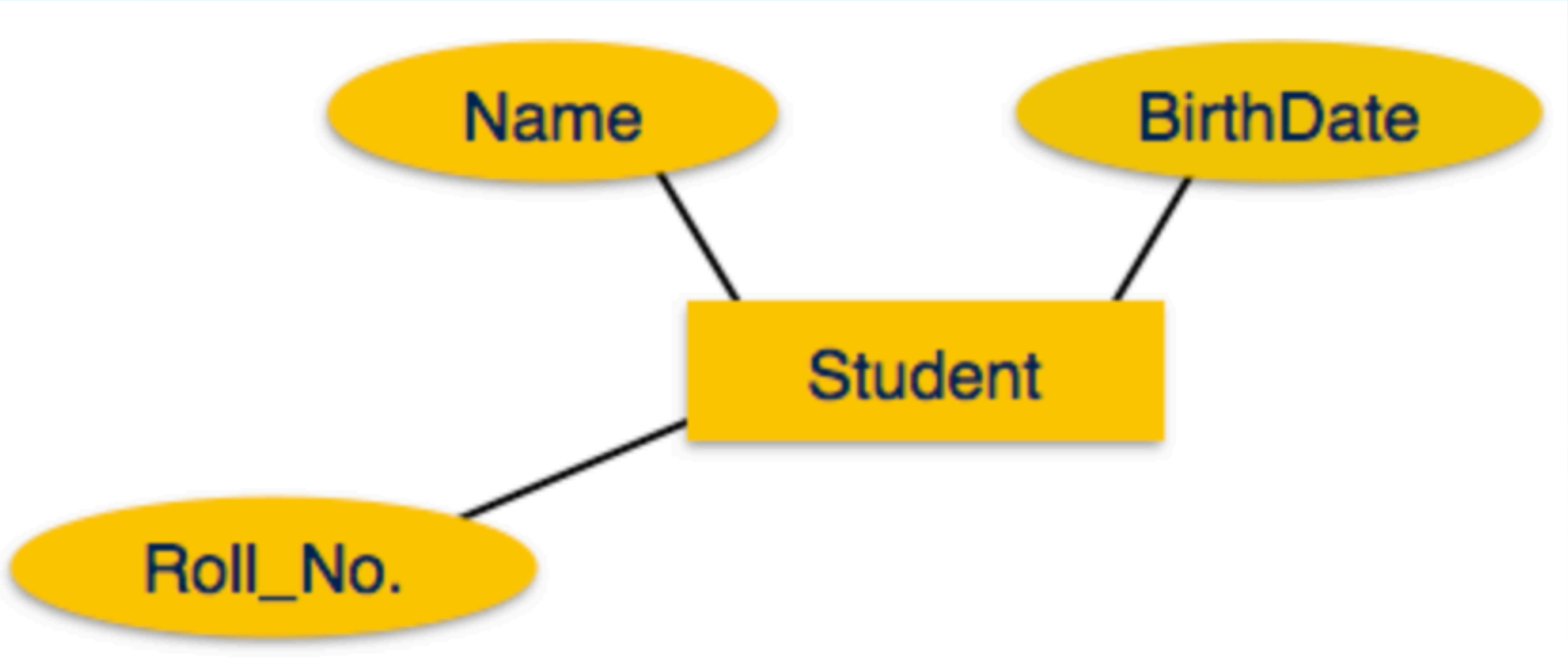
- Attributes are the properties of entities. Attributes are represented by means of ellipses. Every ellipse represents one attribute and is directly connected to its entity (rectangle).
- If the attributes are **composite**, they are further divided in a tree like structure.

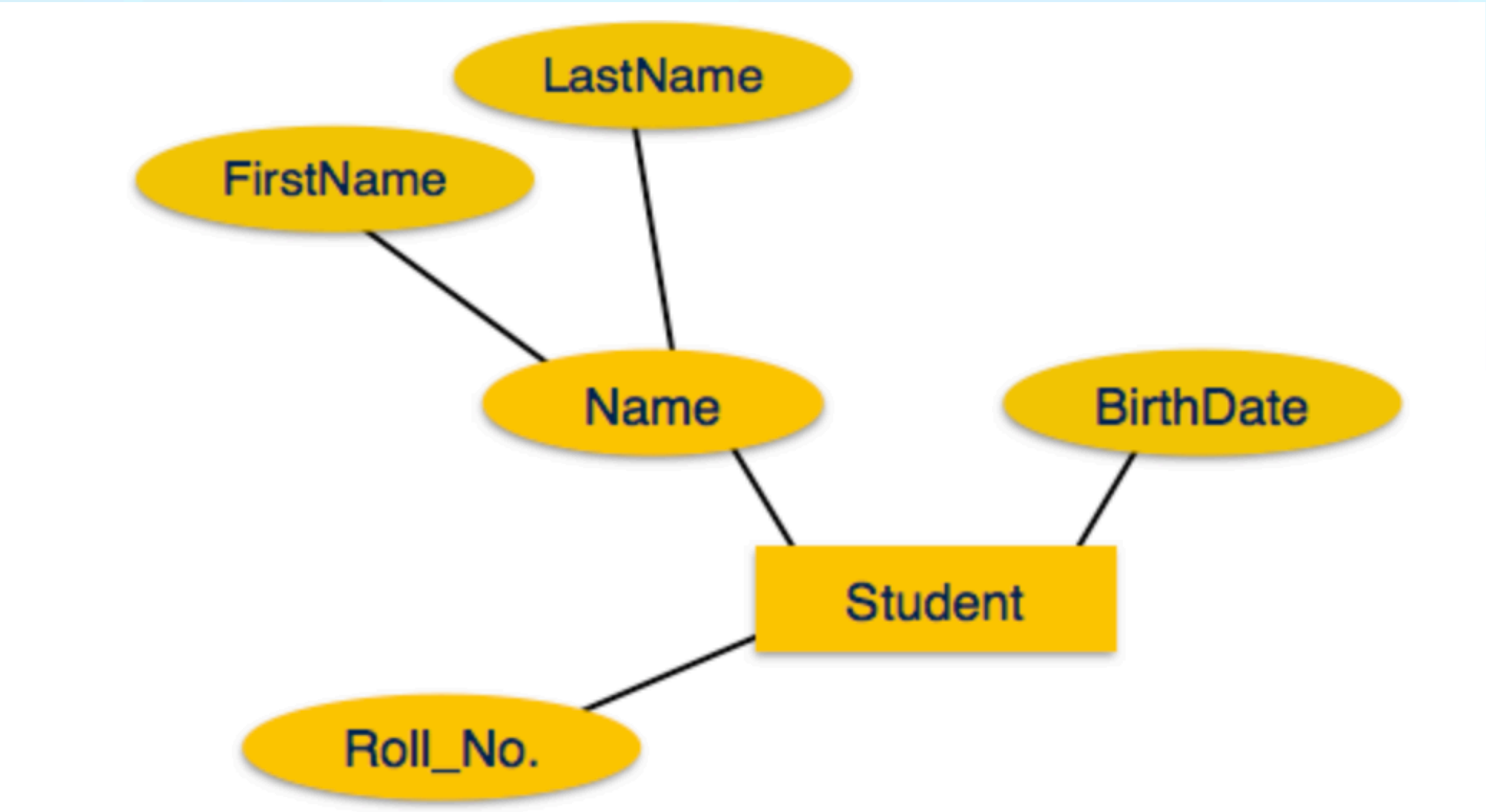


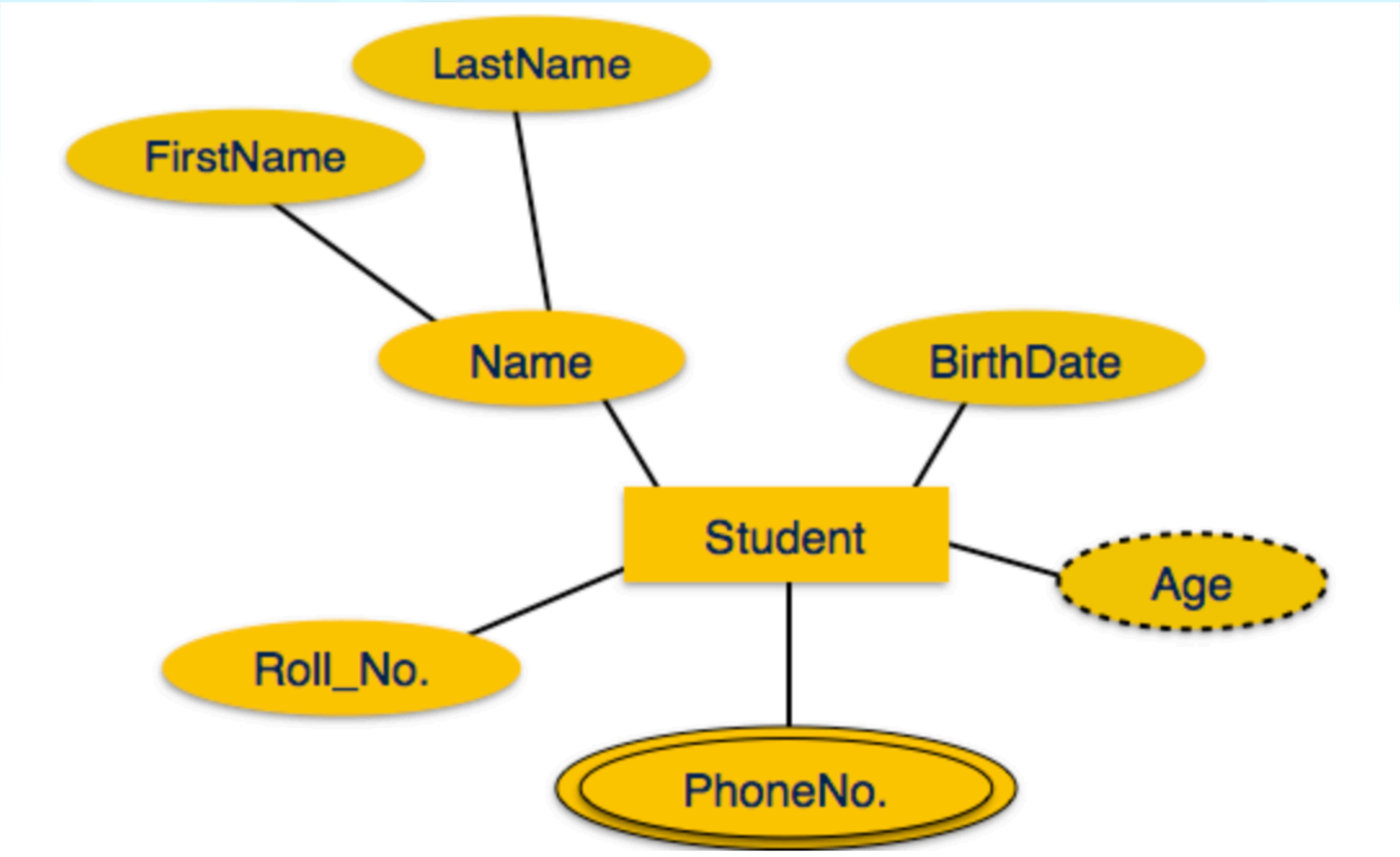


ER Diagram Representation

Attributes *(Simple , Composite , Multi-Valued , Derived)*







ER Diagram Representation

Relationships

- Relationships are represented by diamond-shaped box. Name of the relationship is written inside the diamond-box. All the entities (rectangles) participating in a relationship, are connected to it by a line.

ER Diagram Representation

Relationships and Cardinality

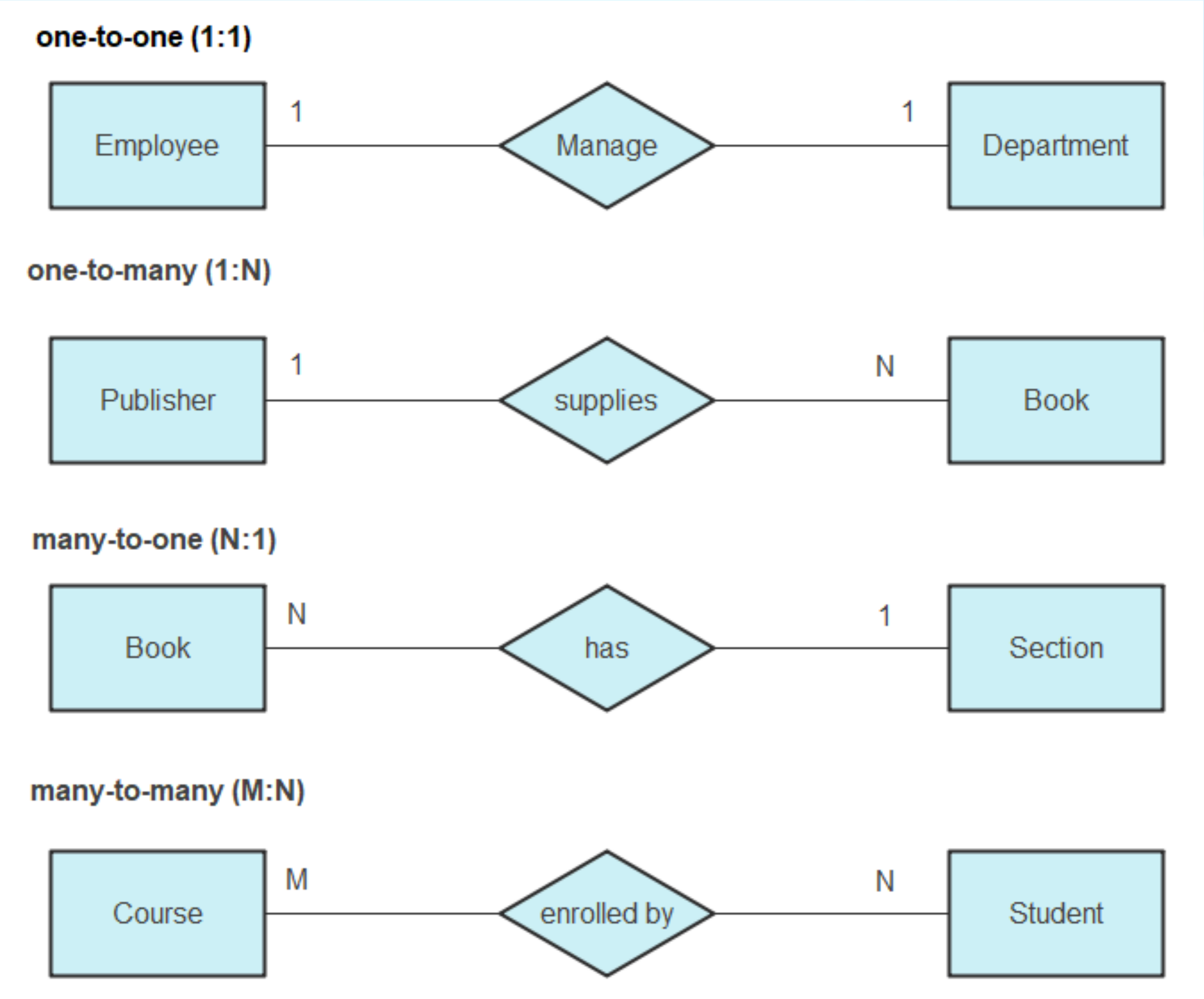
Cardinality represents the number of times an entity of an entity set participates in a relationship set or we can say that the cardinality of a relationship is the number of tuples (rows) in a relationship.

Types of cardinality in between tables are:

- one-to-one
- one-to-many
- many-to-one
- many-to-many

ER Diagram Representation

Relationships and Cardinality



Any Questions?