ENTITY RELATIONSHIP DIAGRAM (ERD)

Akhmad Bakhrun, S.Kom., M.T.

ERD

- ERD merupakan diagram konseptual tingkat tinggi
- ERD didasarkan pada gagasan entitas dunia nyata dan hubungan di antara entitas.
- ERD membantu untuk mengidentifikasi entitas yang ada dalam suatu sistem dan hubungan antara entitas tersebut.
- ERD adalah alat grafis yang mudah digunakan untuk memodelkan data
- ERD banyak digunakan dalam desain basis data dan dianggap sebagai praktek terbaik sebelum menerapkan basis data.

MENGAPA MENGGUNAKAN ERD?

- ✓ Membantu untuk menentukan istilah yang terkait dengan pemodelan data.
- ✓ Memberikan preview tentang bagaimana semua tabel harus terhubung, bidang apa yang akan ada di setiap tabel.
- ✓ Membantu untuk menggambarkan entitas, atribut, hubungan antar entitas.
- ✓ ERD diterjemahkan ke dalam tabel relasional yang memungkinkan untuk membangun basis data dengan cepat.
- ✓ ERD dapat digunakan oleh perancang basis data sebagai cetak biru untuk mengimplementasikan data dalam aplikasi perangkat lunak.
- ✓ Perancang basis data memperoleh pemahaman yang lebih baik tentang informasi yang akan disimpan di dalam basis data.
- ✓ ERD menjadi media komunikasi dengan pengguna terkait struktur logis dari basis data

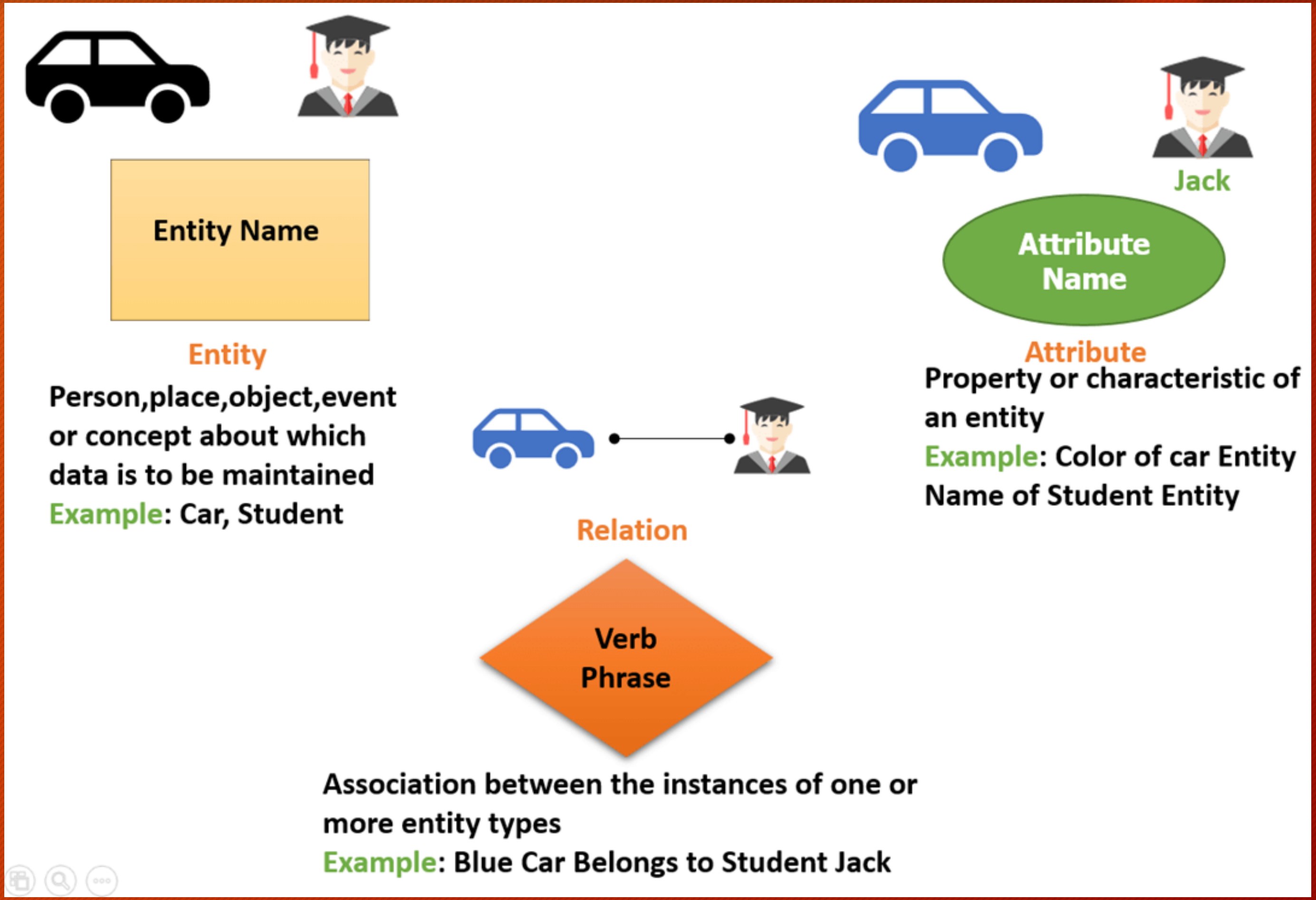
HISTORI MODEL DATA

- Diagram ER adalah alat visual yang membantu untuk merepresentasikan model ER.
- Diusulkan oleh Peter Chen pada tahun 1971 untuk membuat **konvensi** yang dapat digunakan untuk basis data relasional.
- Model ER digunakan sebagai pendekatan pemodelan konseptual.

ERD ELEMENTS

- ✓ Entity
- ✓ Attribute
- ✓ Relationship

Entity	Instances
STUDENT	Anugrah Agung Saputra
	Azka Jonas Wiratama
Student	Azy Ilham Sudibyo
	Herfiky Aji Praditya
	Husni Fadillah

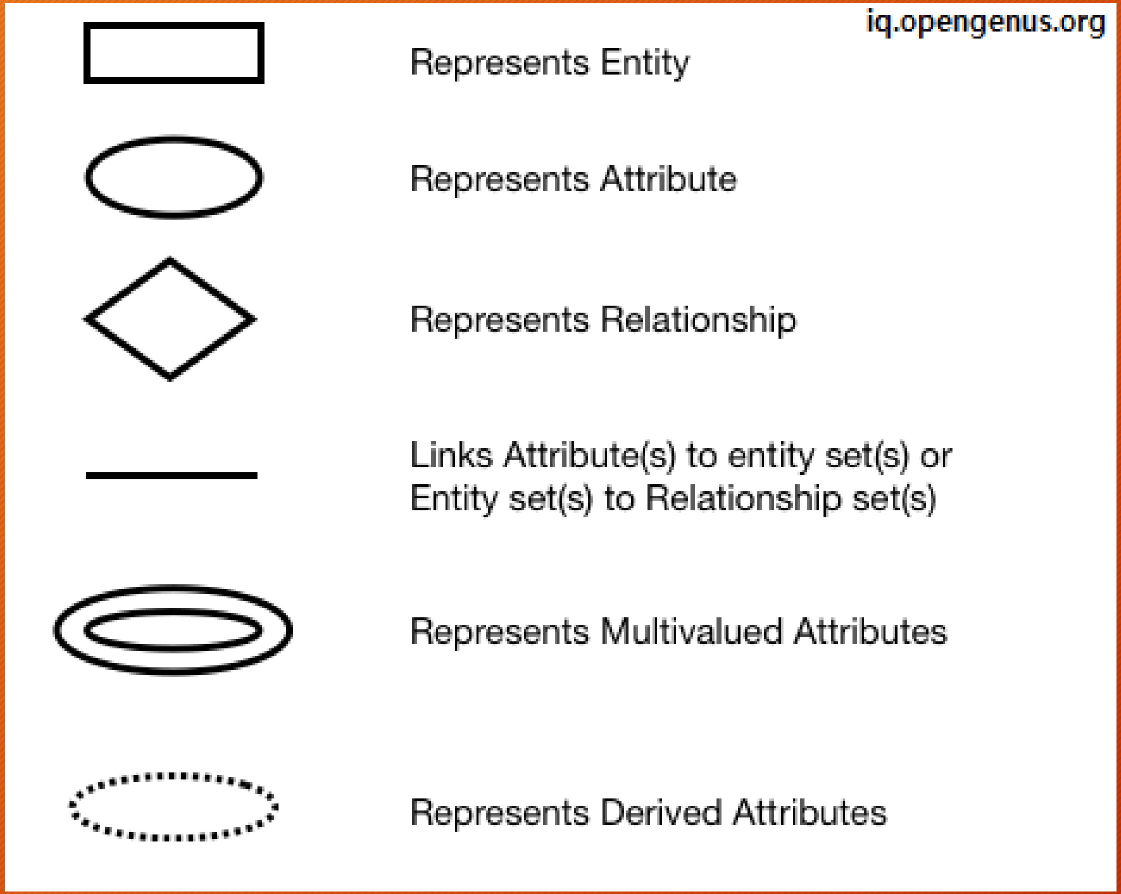


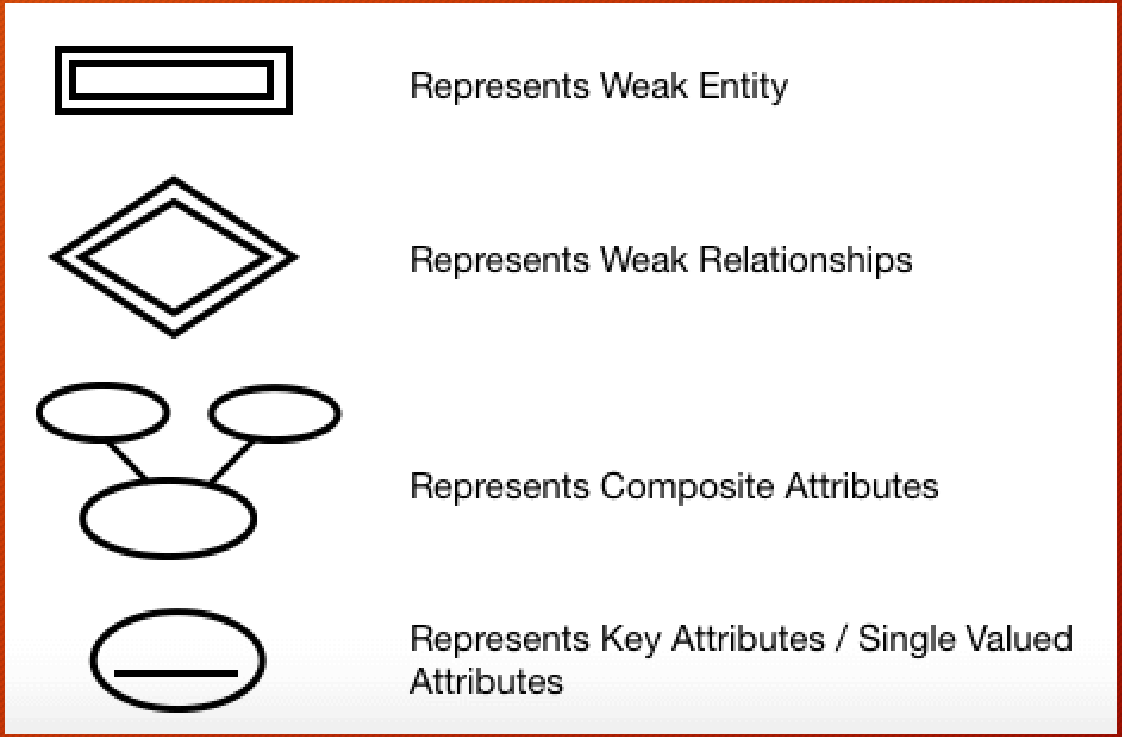
Entity
Person,place,object,event or concept about which data is to be maintained
Example: Car, Student

Attribute
Property or characteristic of an entity
Example: Color of car Entity
Name of Student Entity

Relation
Association between the instances of one or more entity types
Example: Blue Car Belongs to Student Jack

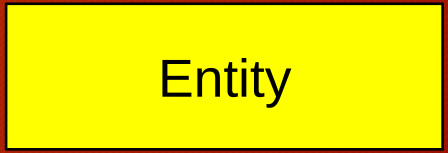
ERD SYMBOLS


iq.opengenus.org
Represents Entity
Represents Attribute
Represents Relationship
Links Attribute(s) to entity set(s) or Entity set(s) to Relationship set(s)
Represents Multivalued Attributes
Represents Derived Attributes


Represents Weak Entity
Represents Weak Relationships
Represents Composite Attributes
Represents Key Attributes / Single Valued Attributes

ENTITY

- ✓ The entity is the basic building block for a data model.
- ✓ Entity is a person, place, object, event, concept, or thing about which data is collected—for example, an employee, an order, or a product.
- ✓ An entity is depicted by a **rectangle**, and it is described by a **singular noun spelled** in capital letters (first letter capitalized). Written consistently.
- ✓ **All entities have a name**, a short description that **explains what they are**, and an identifier that is the way to locate information in the entity
- ✓ Entities represent something for which there exist **multiple instances**, or occurrences.
- ✓ For example, Anugrah Agung Saputra and Azka Jonas Wiratama could be instances of the student entit
- ✓ If there is **just one instance**, or occurrence, of a person, place, event, or thing, then **it should not be included as an entity in the data model**.

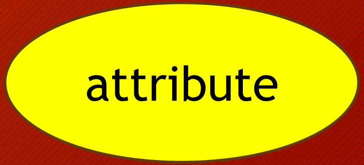


ENTITY EXAMPLE

No	Entity	Example
1.	Person	Student, Employee
2.	Place	Shop, Warehouse, Class
3.	Object	Book, Car, Product
4.	Event	Sale, Registration, Attendance
5.	Concept	Account, Subject

ATTRIBUTE

- ✓ An *attribute* is some **type of information that is captured** about an entity (characteristics of entity)
- ✓ For example: name, birth of date, gender, address, and e-mail are all attributes of a student.
- ✓ It is easy to come up with hundreds of attributes for an entity (e.g., a student has an eye color, a favorite hobby, a religious affiliation), **but only those that actually will be used by a business process should be included in the model**.
- ✓ Attributes are **nouns** that are **listed within an entity**.
- ✓ Usually, some form of the **entity name is appended to the beginning of each attribute** to make it clear as to what entity it belongs (e.g., student_id, student_name, student_gender, student_address, student_email).
- ✓ Without doing this, you can get confused by multiple entities that have the same attributes—for example, a student and a lecturer both can have an attribute called "name" student_name and lecturer_name are much clearer ways to name attributes on the data model.
- ✓ Each attribute has its **own domain** (value set)
- ✓ Domain is the **constraints of the values allowed for an attribute**.
- ✓ The attribute model is depicted with an **ellipse** symbol.



ATTRIBUTE IDENTIFIER

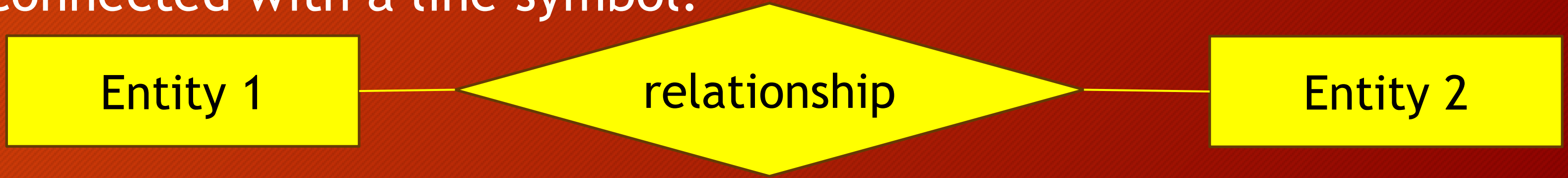
- ✓ One or more attributes can serve as **the identifier**
- ✓ Identifier attribute is the attribute(s) that **can uniquely identify one instance of an entity**—and the attributes that serve as the identifier are noted by an **underline** or asterisk next to the attribute name.
- ✓ If there are **no student with the same name**, then name can be used as the identifier of the student entity. **Otherwise**, the name cannot be used as an identifier for a student entity.
- ✓ There are three ways to define attribute as identifier:
  1) Use a **combination of multiple fields** to serve as the identifier (name and birth of date). This is called a *concatenated identifier* because several fields are combined, or concatenated, to uniquely identify an instance.
  2) Find a **field that is unique for each instance**, like the student student_id.
  3) Wait to **assign an identifier** (like a randomly generated number that the system will create). Many data modelers do not believe that randomly generated identifiers belong on this data model, because they **do not logically exist in the business process**.

ATTRIBUTE TYPE

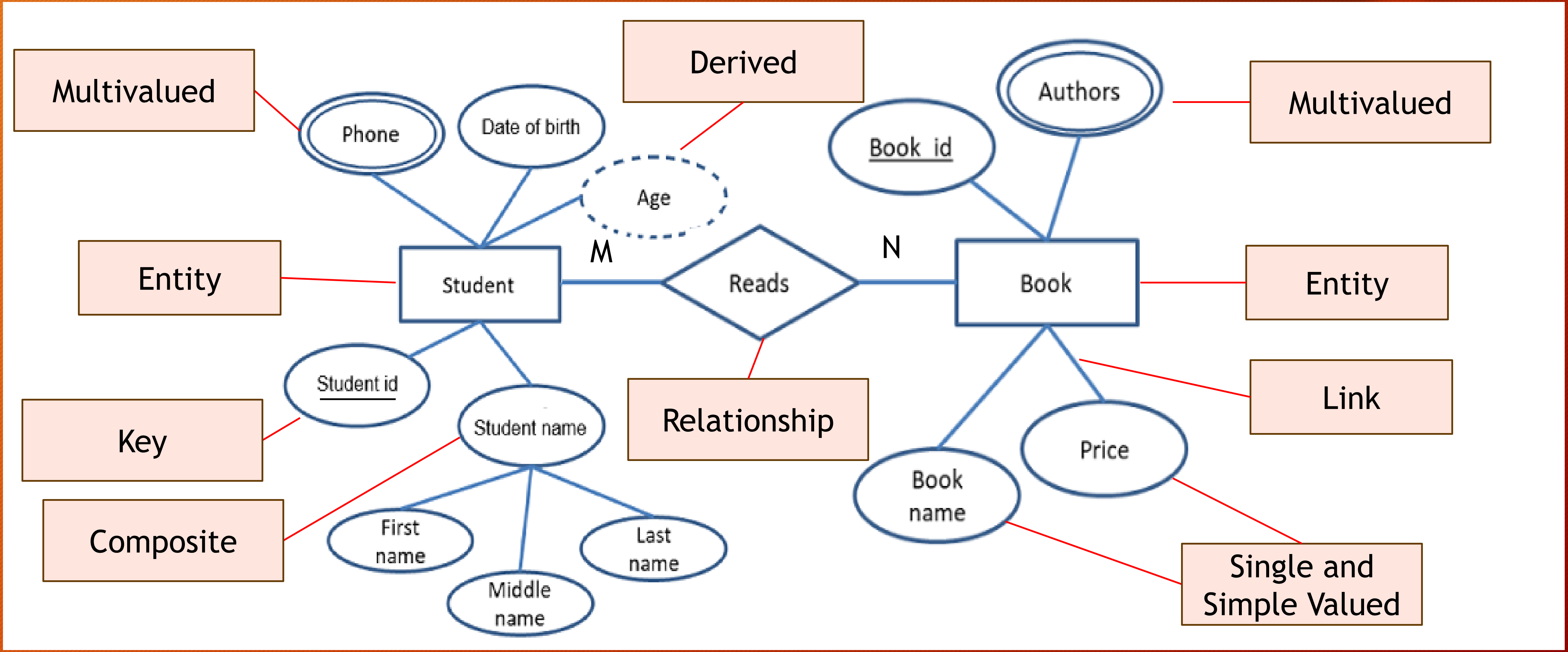
No	Type	Description	Example
1.	Key	Attributes used to uniquely identify an entity	student id, book id
2.	Simple	Can’t be broken down or atomic	book name, price
3.	Composite	Can be broken down	student name = first name + middle name + last name.
4.	Derived	Not present in the physical Database but derived from other attributes.	age, derived from the birth date attribute and the current system date.
5.	Single Value	Having only one value	book name, price
6.	Multivalued	Having more than one value	phone, one student has several phone numbers

RELATIONSHIP

- ✓ Relationships are **associations between entities**, and they are shown by lines that connect the entities together.
- ✓ Every relationship has **a parent entity** and a **child entity**, the parent being the **first entity** in the relationship, and the child being the second
- ✓ Relationships should be clearly labeled with **active verbs** so that the connections between entities can be understood.
- ✓ If one verb is given to each relationship, it is read in two directions.
- ✓ For example, we could write the verb **reads** alongside the relationship for the *Student* and *Book* entities, and this would be read as “a student reads a book” and “a book is read by a student.
- ✓ The relationship model is depicted with a diamond/rhombus symbol.
- ✓ The relationship and entity are connected with a line symbol.

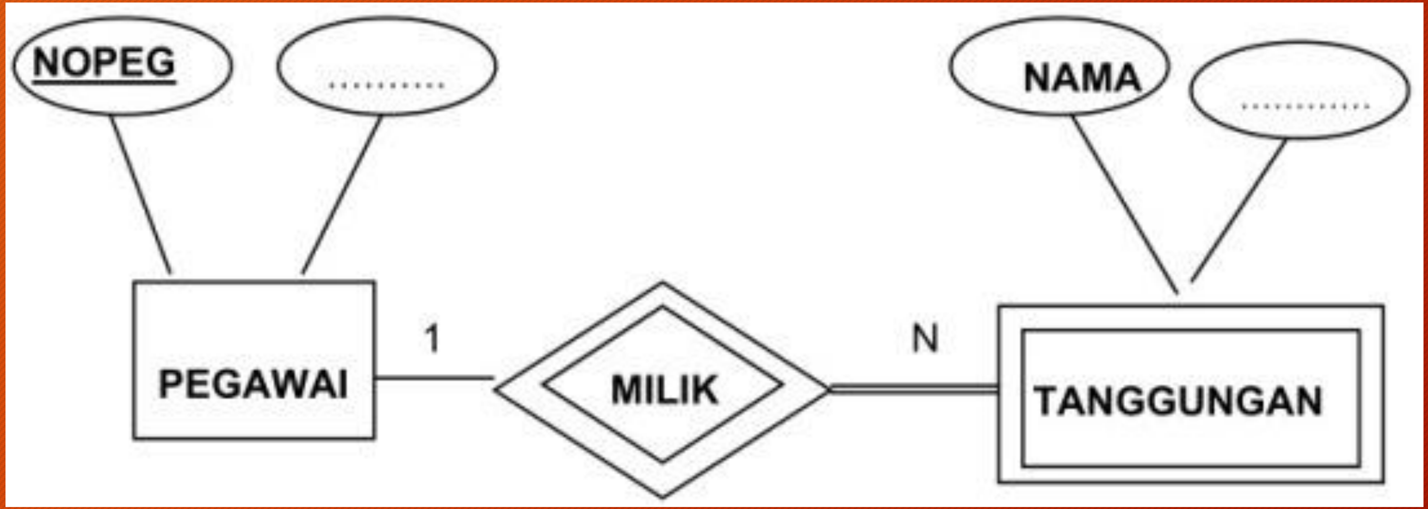


ERD EXAMPLE


Multivalued
Derived
Multivalued
Phone
Date of birth
Age
Authors
Book id
Entity
Student
M
Reads
N
Book
Entity
Student id
Student name
Relationship
Link
Key
Price
Book name
Composite
First name
Middle name
Last name
Single and Simple Valued

WEEK ENTITY

- ✓ Entitas lemah adalah jenis entitas yang tidak memiliki atribut kunci utama (*primary key*).
- ✓ Identifikasi unik pada *week entity* menggunakan kunci utama dari entitas lain.



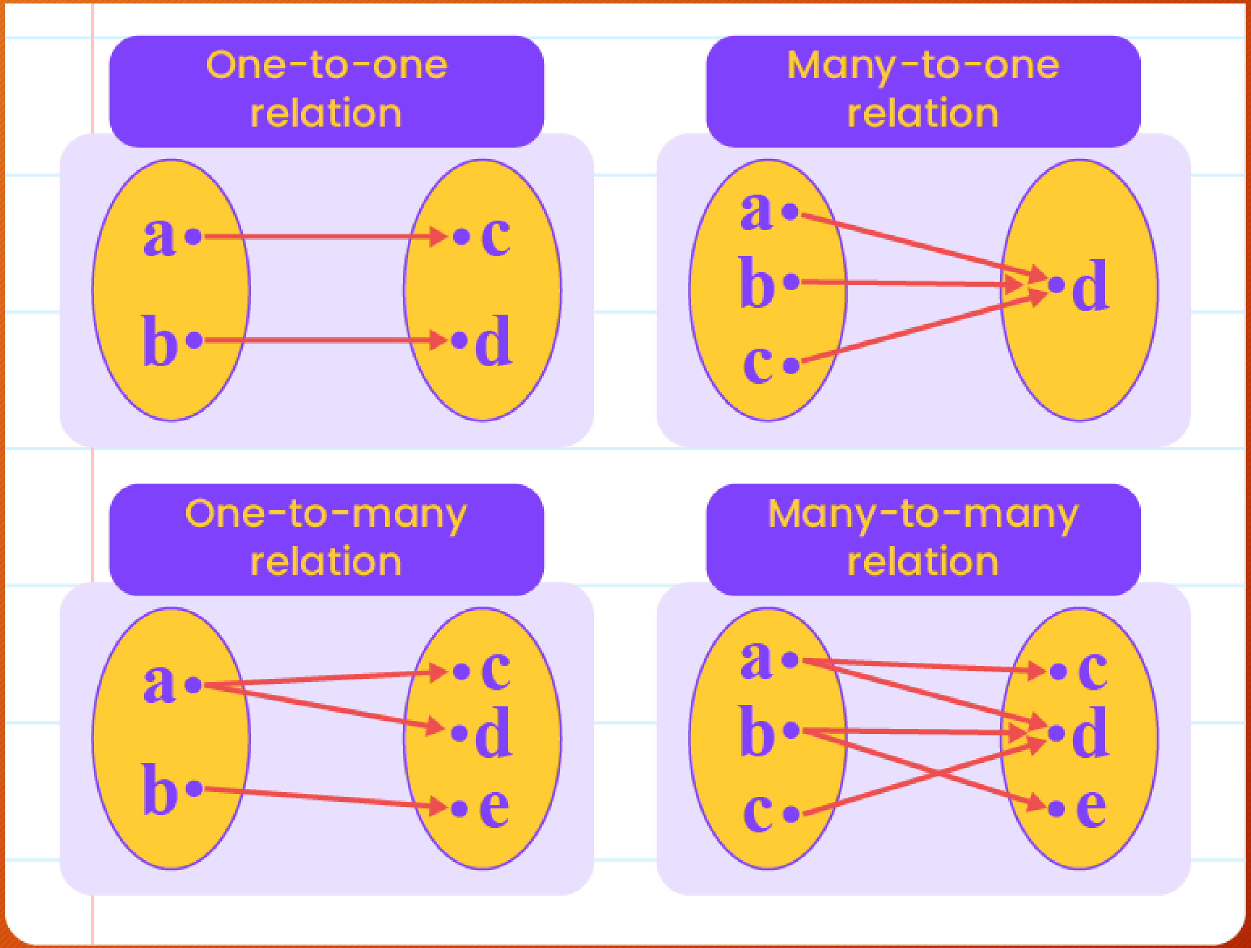
STRONG ENTITY vs WEEK ENTITY

No	Strong Entity	Week Entity
1.	Memiliki atribut untuk membangun primary key	Tidak memiliki atribut yang cukup untuk membangun primary key
2.	Menggunakan simbol persegi panjang tunggal	Menggunakan simbol persegi panjang ganda
3.	Primary key ditandai dengan garis bawah (underline)	Partial key ditandai dengan garis bawah putus-putus
4.	Anggota entitas disebut himpunan dominan	Anggota entitas disebut himpunan bawahan
5.	Primary key salah satu atribut untuk mengidentifikasi anggota himpunan	Kombinasi primary key dan partial key dari entitas himpunan yang kuat
6.	Reletionship menggunakan simbol berlian (*diamond*) atau belah ketupat tunggal	Reletionship menggunakan simbol berlian (*diamond*) atau belah ketupat ganda
7.	Garis penghubung antar entitas menggunakan relationship tunggal	Garis penghubung antar entity menggunakan relationship ganda
8.	Contoh: Pegawai	Contoh: Tanggungan_Pegawai

CARDINALITY

- ✓ Cardinality defines the possible **number of instances** in one entity (parent) which is associated with the number of instances in another (child).
- ✓ To determine the cardinality for a relationship, we ask ourselves: "**How many instances** of one entity are associated with an instance of the other?"
- ✓ Remember that an instance is one occurrence of an entity, such as Anugrah Agung Saputra atau Azka Jonas Wiratama.
- ✓ There are three types of cardinality:
  1) The 1:1 (read as **one to one**) relationship means that one instance of the parent entity is associated with one instance of the child entity.
  2) The 1:N (read as **one to many, more often**). In this kind of relationship, a single instance of a parent entity is associated with many instances of a child entity; however, the child entity instance is related to only one instance of the parent
  3) The M:N (read as **many to many**) relationship. In this case, many instances of a parent entity can relate to many instances of a child entity

MATHEMATICAL FUNCTION APPROACH



Contoh di dunia nyata:

One to One
Satu anak memiliki satu ibu

One to Many
Satu ibu memiliki banyak anak

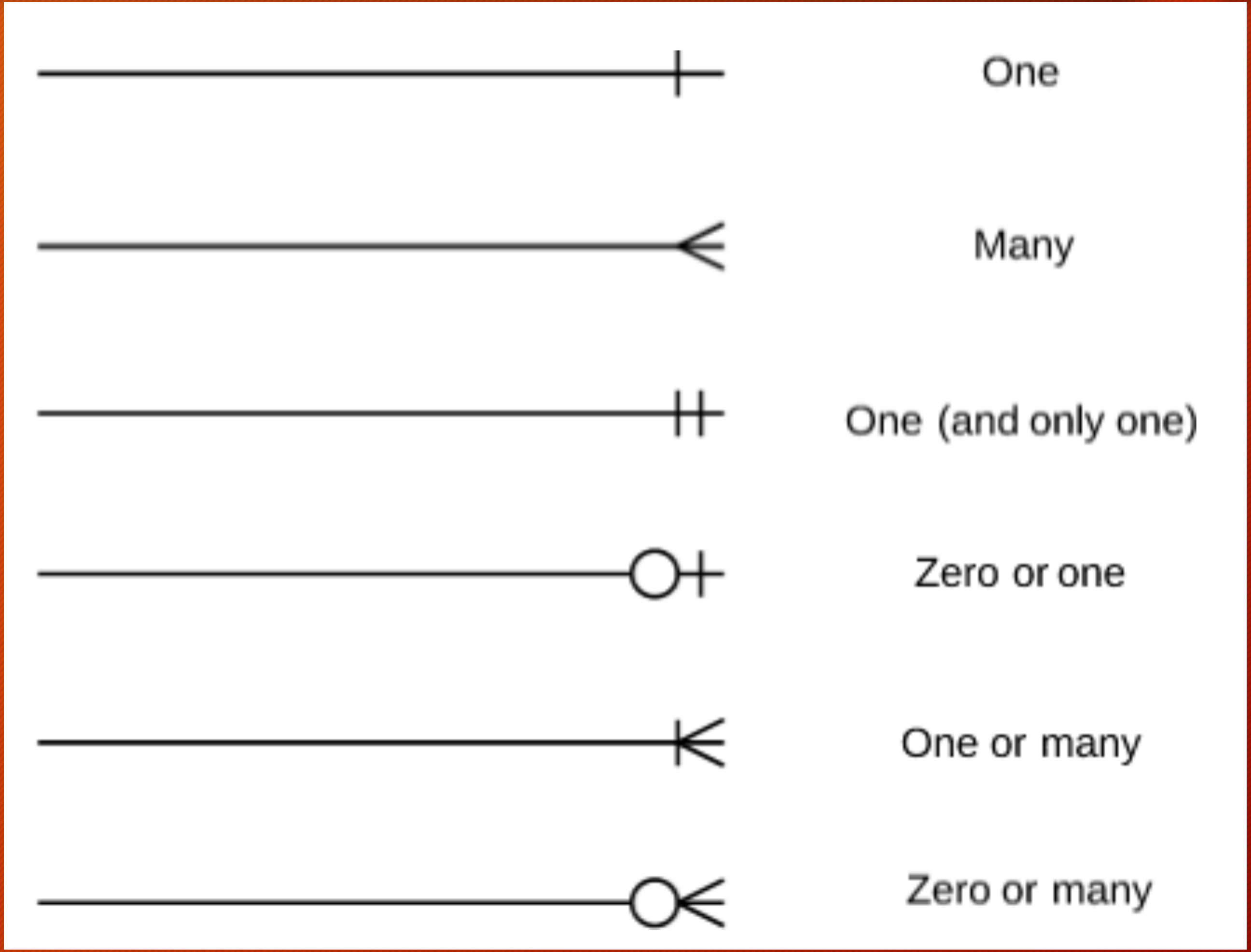
Many to One
Banyak mahasiswa melakukan perwalian dengan satu dosen

Many to Many
Banyak dosen mengajar banyak mahasiswa

MAPPING OF ENTITIES

Cardinality : Maximum number of times an instance in one entity can relate to instance of another entity.

Ordinality : Minimum number of times an instance in one entity can relate to instance of another entity.



KEYS

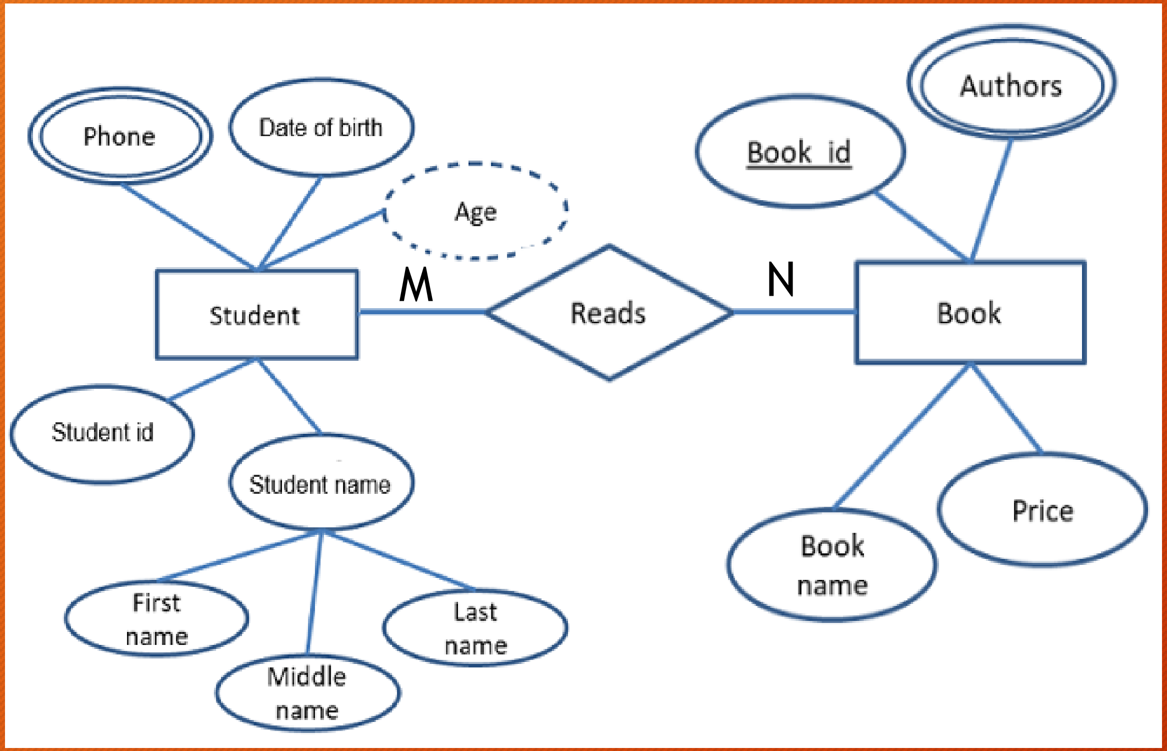
Keys are used to uniquely identify any entry of data in the database.

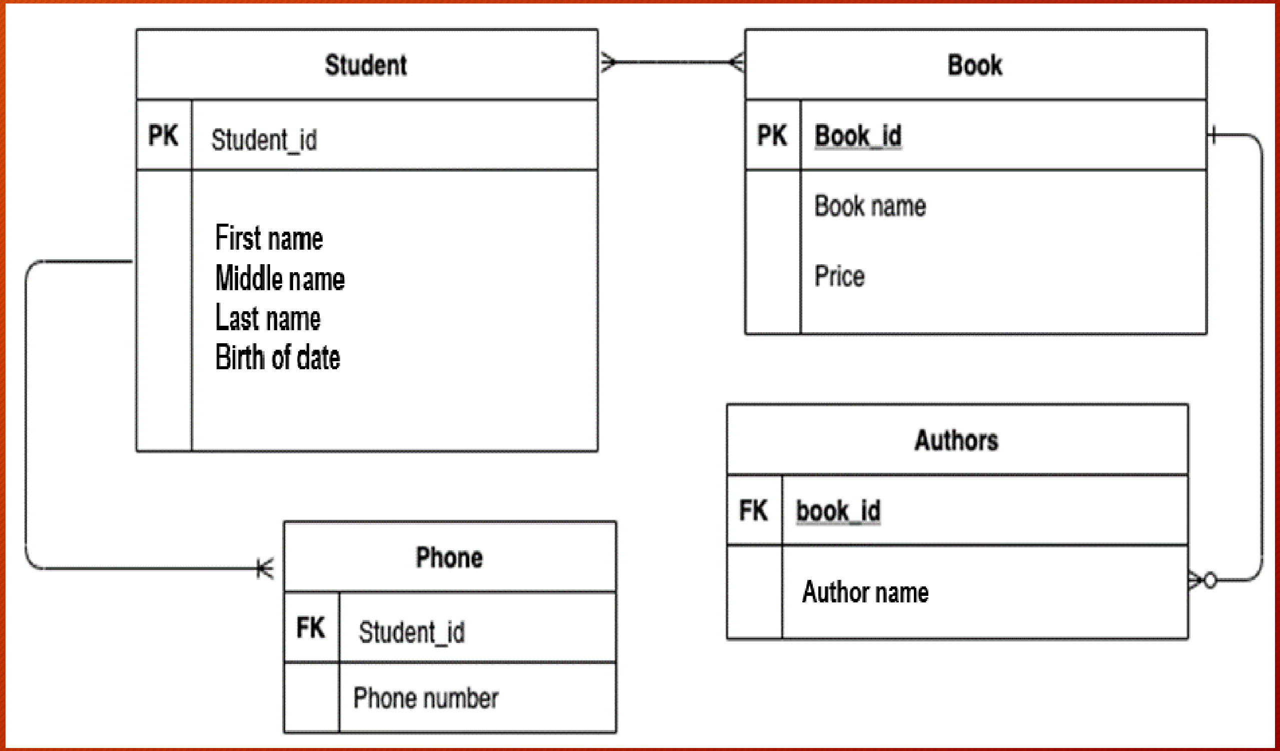
1. **Primary key:** It uniquely identifies only one entry in the database. For example, In the above example, *’Student id'* is the primary key as it is unique for every student entry.
2. **Candidate key:** It is all the attributes that are suitable to be a primary key. For example, *’Student id'* is a candidate key in the above example.
3. **Super key:** It is a set of attributes that are suitable to be a primary key.
4. **Foreign key:** When a primary key of one table is reflected another table to connect the tables, that key is called a foreign key in the receiver table and is called primary key in the host table. For example, *'Book id'* from Book table when relfected into Student table, it will be called 'Foreign key' in student table.

CONVERT ER MODEL TO RELATIONAL MODEL

No	ER Model	Relational Model
1	Entity	Table
2	Single valued attribute	Column
3	Primary key attribute	Underlined column
4	Multivalued attributed	Separate table
5	Composite attribute	Separate column
6	Derived attribute	Not considered or ignored

CONVERT ER MODEL TO RELATIONAL MODEL





GUIDE TO DESIGNING A GOOD ERD

1. Each entity must be interconnected with other entities through relationships.
2. Each entity must have attributes, including a primary key and descriptive attributes.
3. Relationships between entities are clearly depicted using diamond notation and appropriate verbs.
4. The degree of cardinality (one-to-one, one-to many, many-to-many) is clearly shown for each relationship.
5. The use of notations and symbols must be consistent, following applicable standards.