ENTITY-RELATIONSHIP MODELING

By Hey Jude!

OBJECTIVES

- Define ER Modeling, its purpose, and the distinction between entity, entity set, and entity types.
- Explain attributes, keys, and the role of null values.
- Describe relationships and their types.
- Interpret and create basic ER Diagrams (ERDs).

ENTITY-RELATIONSHIP MODEL

Entity-Relationship Model (ERM) is a diagram that shows how data is related in a database using entities, attributes, and relationships.

WHY DO WE NEED ENTITY-RELATIONSHIP (ER) MODEL?

- Helps plan and organize the structure of a database.
- Shows how data is connected through entities and relationships.
- Prevents errors and reduces data redundancy.
- Makes the database easier to understand and design.
- Serves as a blueprint before creating the actual database.

WHAT IS AN ENTITY?

AN ENTITY IS ANY REAL-WORLD OBJECT, CONCEPT, OR THING THAT CAN BE CLEARLY IDENTIFIED AND STORED IN A DATABASE.

PERSON

PLACE

PRODUCT

WHAT IS AN ENTITY TYPE?

AN ENTITY TYPE DEFINES A COLLECTION OF ENTITIES THAT SHARE THE SAME PROPERTIES OR ATTRIBUTES.

EXAMPLES:

Entity type: Student

Attributes: StudentID, Name, Age

Entity type: Department

Attributes: DeptID, Dept_Name, Location

Entity type: Vehicle

Attributes: Plate_No, Model, Type

ENTITY SET

AN ENTITY SET IS THE COLLECTION OF ALL ENTITIES OF A PARTICULAR TYPE STORED IN THE DATABASE.

STUDENT

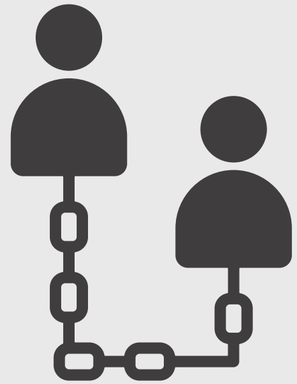
StudentID	Name	Age	Email
1001	Ruy Richards	20	Ruy@gmail.com
1002	Zach Marquee	82	Marquee@gmail.com
1003	Sai The Skye	19	Sai@gmail.com

KINDS OF
ENTITIES

ZACH LUTHER MARQUEZ

KINDS OF ENTITIES

INDEPENDENT

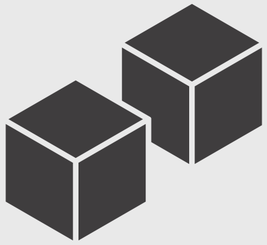


DEPENDENT

INDEPENDENT / STRONG / KERNEL

- **BACKBONE OF DATABASE**
- Can exist on their own and have a primary key that uniquely identifies each instance, independent of other entities.

INDEPENDENT / STRONG / KERNEL CHARACTERISTICS



BUILDING BLOCKS

PRIMARY KEY IS SIMPLE OR COMPOSITE

PRIMARY KEY IS NOT A FOREIGN KEY

RECORDS CAN EXIST ON THEIR OWN

IDENTIFYING THE INDEPENDENT ENTITY

PRODUCTS
(PROD_ID(PK), NAME, PRICE, DETAILS(FK))

PRODUCTS
(PROD_ID(PK), NAME, PRICE(FK), DETAILS)

PRODUCTS
(PROD_ID(FK), NAME, PRICE, DETAILS)

PRODUCTS
(ID(PK), NAME, PRICE, DETAILS)

IDENTIFYING THE INDEPENDENT ENTITY

PRODUCTS

(ID(PK), NAME, PRICE, DETAILS)

INDEPENDENT ENTITY

IDENTIFYING THE INDEPENDENT ENTITY

PRODUCTS
(PROD_ID(PK), NAME, PRICE, DETAILS(FK))

PRODUCTS
(PROD_ID(PK), NAME, PRICE(FK), DETAILS)

PRODUCTS
(PROD_ID(FK), NAME, PRICE, DETAILS)

PRODUCTS
(ID(PK), NAME, PRICE, DETAILS)

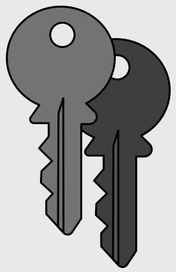
DEPENDENT / WEAK / DERIVED

Can’t exist without association with another, strong entity

KINDS OF ENTITIES

DEPENDENT ENTITY

DEPENDENT / WEAK / DERIVED CHARACTERISTICS



USED TO CONNECT STRONG ENTITIES

DEPENDENT ON OTHER ENTITIES

MAY CONTAIN OTHER ATTRIBUTES

FOREIGN KEY IDENTIFIES EACH ASSOCIATED TABLE

DEPENDENT ENTITY

DEPENDENT / WEAK / DERIVED CHARACTERISTICS

- CAN USE THESE OPTIONS AS ITS PRIMARY KEY.

CREATE A NEW SIMPLE KEY

COMPOSITE OF FOREIGN KEYS

COMPOSITE OF FOREIGN KEYS + QUALIFYING COLUMN

NOTE:

THE INDUSTRY-STANDARD BEST PRACTICE WOULD BE TO SIMPLY ADD A SINGLE, NON-MEANINGFUL SURROGATE KEY

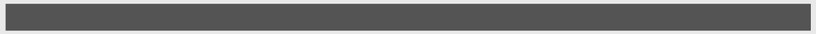
KINDS OF ENTITIES

DEPENDENT CHARACTERISTICS

REPRESENTATION

DASHED LINE

DEPENDENT / WEAK



SOLID LINE

INDEPENDENT / STRONG

TIP

CAN I ADD A ROW TO THIS TABLE WITHOUT THE NEED FOR ANOTHER TABLE?

CAN I ADD A ROW TO THIS TABLE WITHOUT THE NEED FOR ANOTHER TABLE?

YES

INDEPENDENT
STRONG
KERNEL

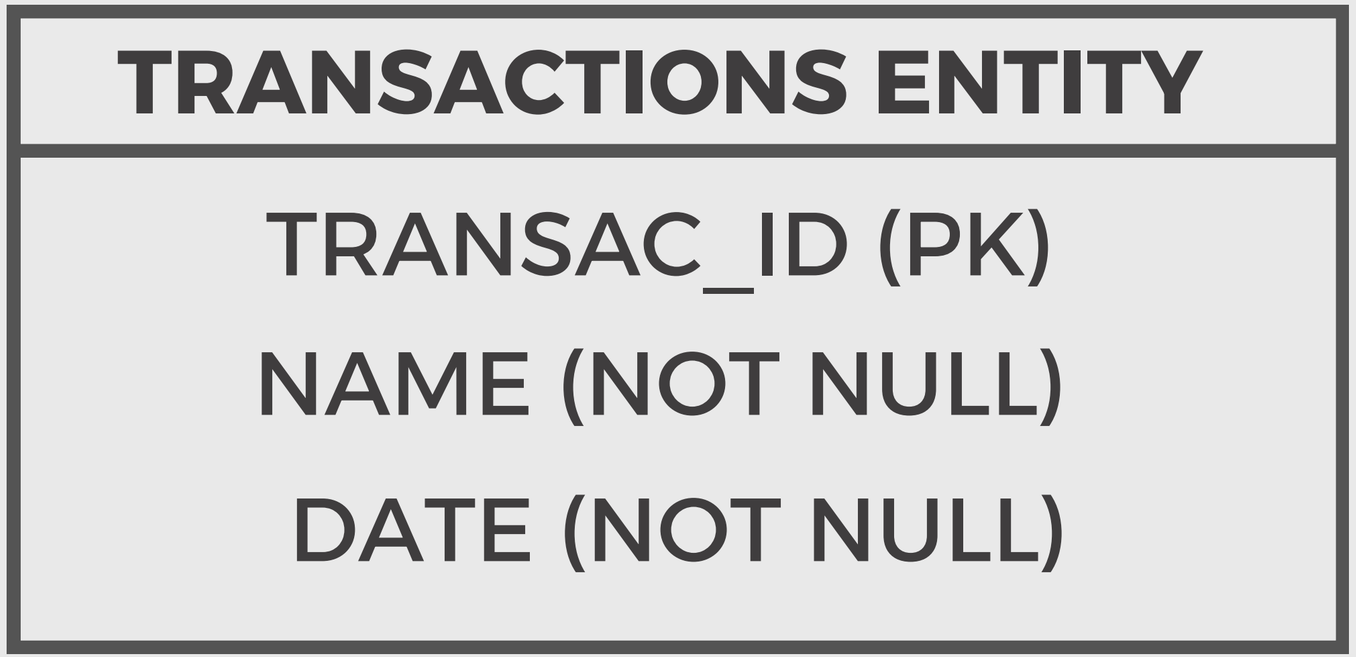
NO

DEPENDENT
WEAK
DERIVED

SHORT ACTIVITY

IDENTIFY IF **DEPENDENT** OR
INDEPENDENT ENTITY

DEPENDENT OR INDEPENDENT



INDEPENDENT ENTITY

DEPENDENT OR INDEPENDENT

TRANSACTIONS ENTITY
TRANSAC_ID (PK)
NAME (NOT NULL)
DATE (NOT NULL)
TAX (FK) (NOT NULL)

DEPENDENT ENTITY

DEPENDENT OR INDEPENDENT

EMPLOYEE ENTITY
EMPLOYEE_ID (PK)
NAME (NOT NULL)
HIRE_DATE (NOT NULL)
DEPARTMENT_ID (FK)

INDEPENDENT ENTITY

ATTRIBUTES

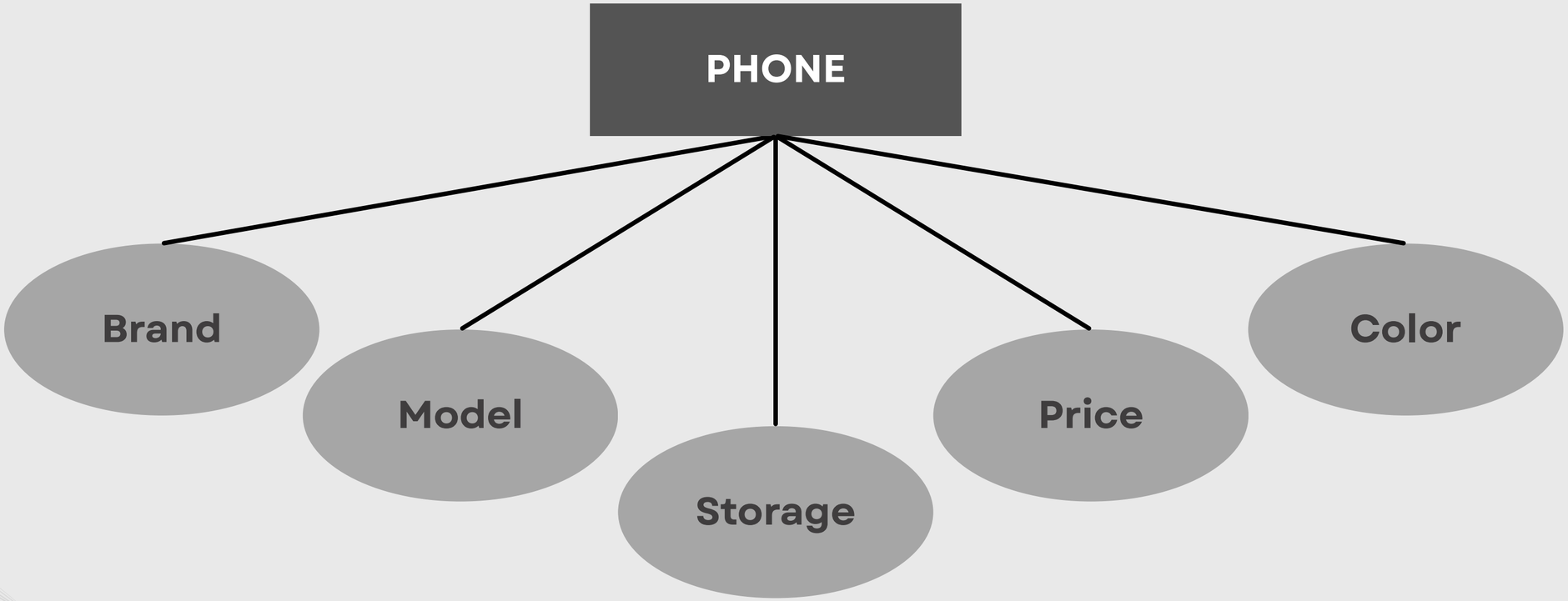
ANDREA SAI MALICDEM

ATTRIBUTES

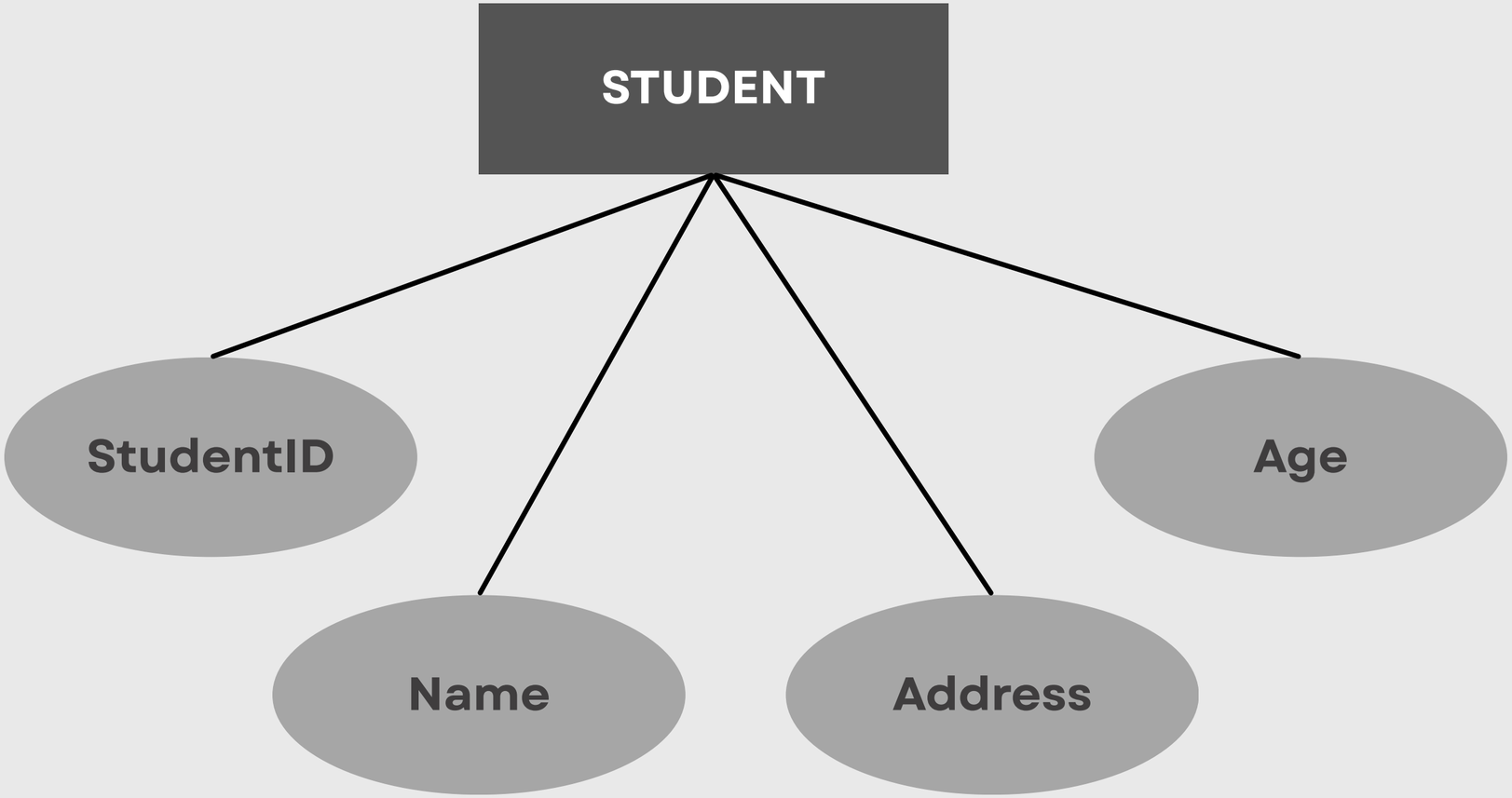
- A property or characteristic of an entity.
- Used to describe an entity in a database.
- Represents the individual pieces of data that are stored about each instance of an entity
- A column or field in a database table.

ATTRIBUTES

Entity: Phone
Attributes: Brand, Model, Storage, Color, Price



ATTRIBUTES



TYPES OF
ATTRIBUTES

TYPES OF ATTRIBUTES

1. SIMPLE ATTRIBUTE

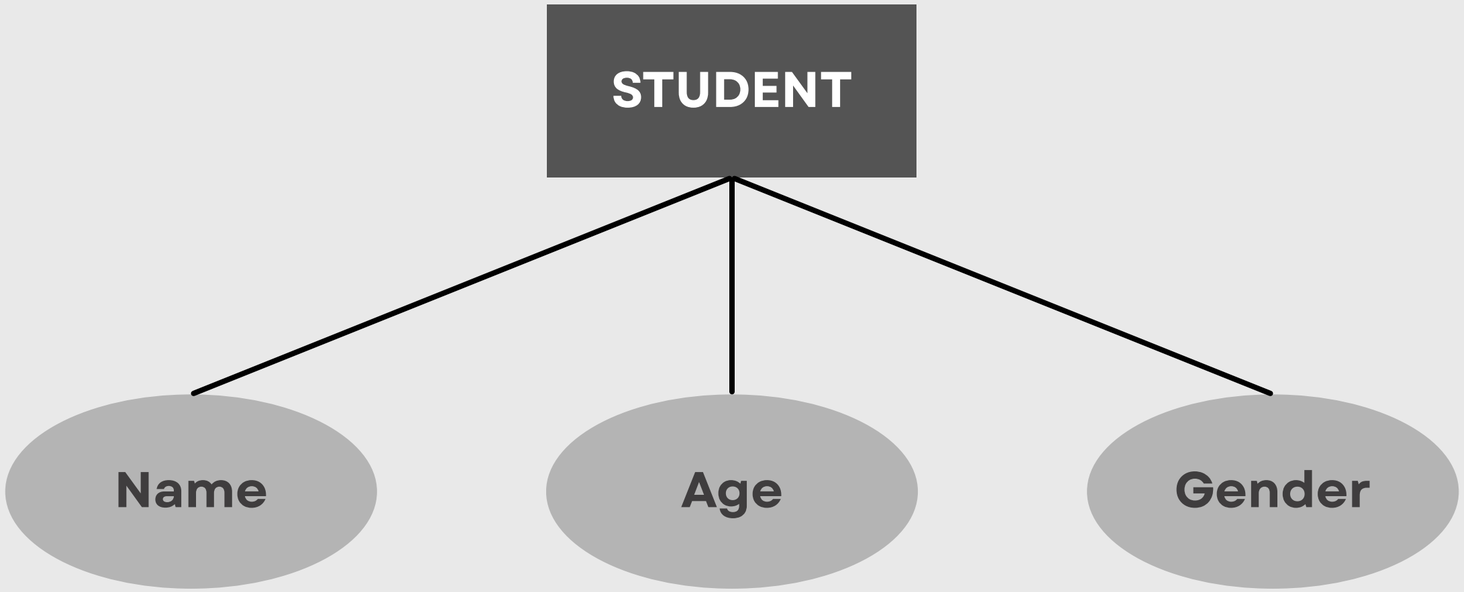
- Also called single-valued attributes.
- Drawn from atomic value domains.
- Represented by a single oval connected to an entity.

Atomic value - values that cannot be broken down into sub-attributes

TYPES OF ATTRIBUTES

SIMPLE ATTRIBUTE

Example:



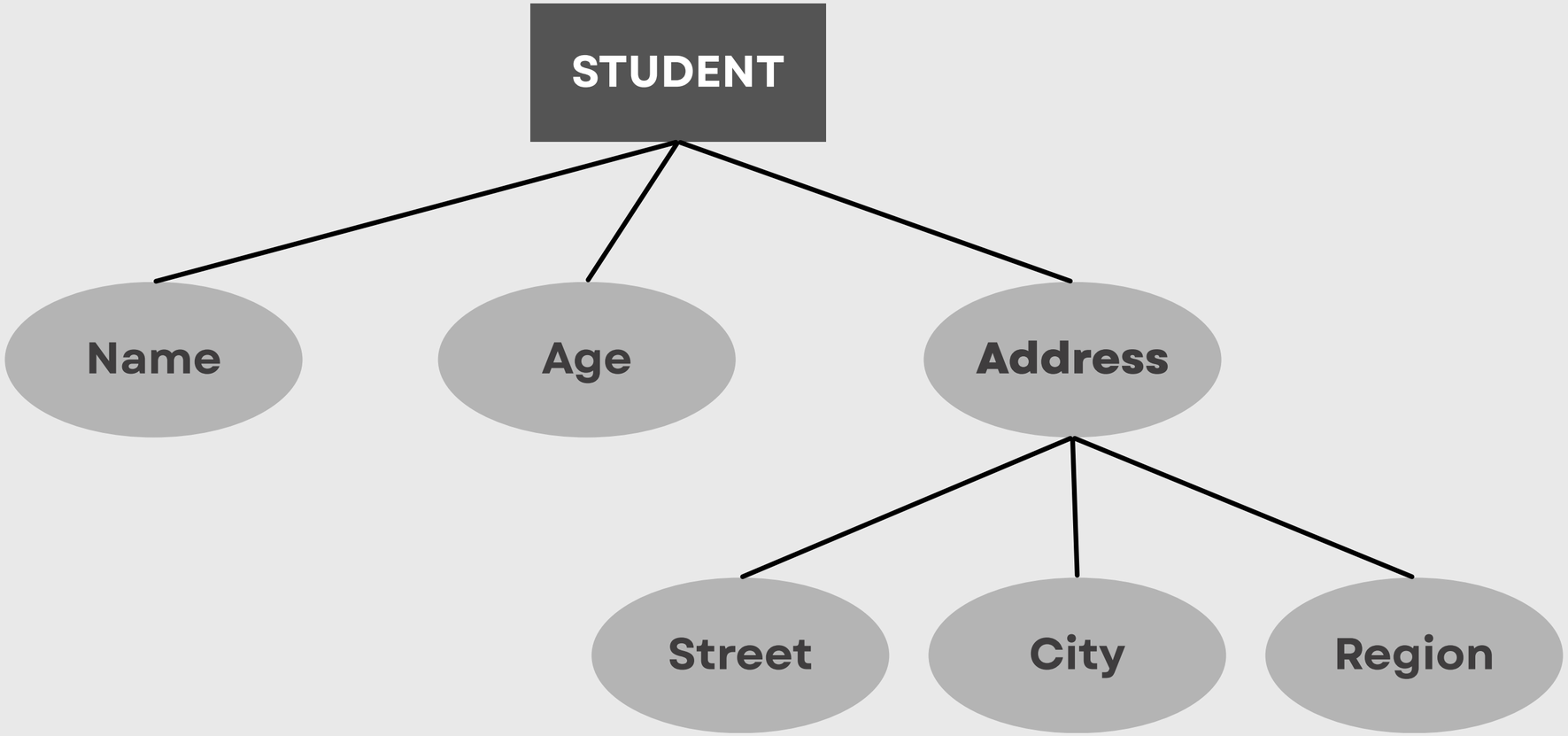
2. COMPOSITE ATTRIBUTE

- Consist of a hierarchy of attributes.
- Can be broken down into multiple, smaller and simpler attributes.
- Represented by an oval with smaller ovals branching out.

TYPES OF ATTRIBUTES

COMPOSITE ATTRIBUTE

Example:



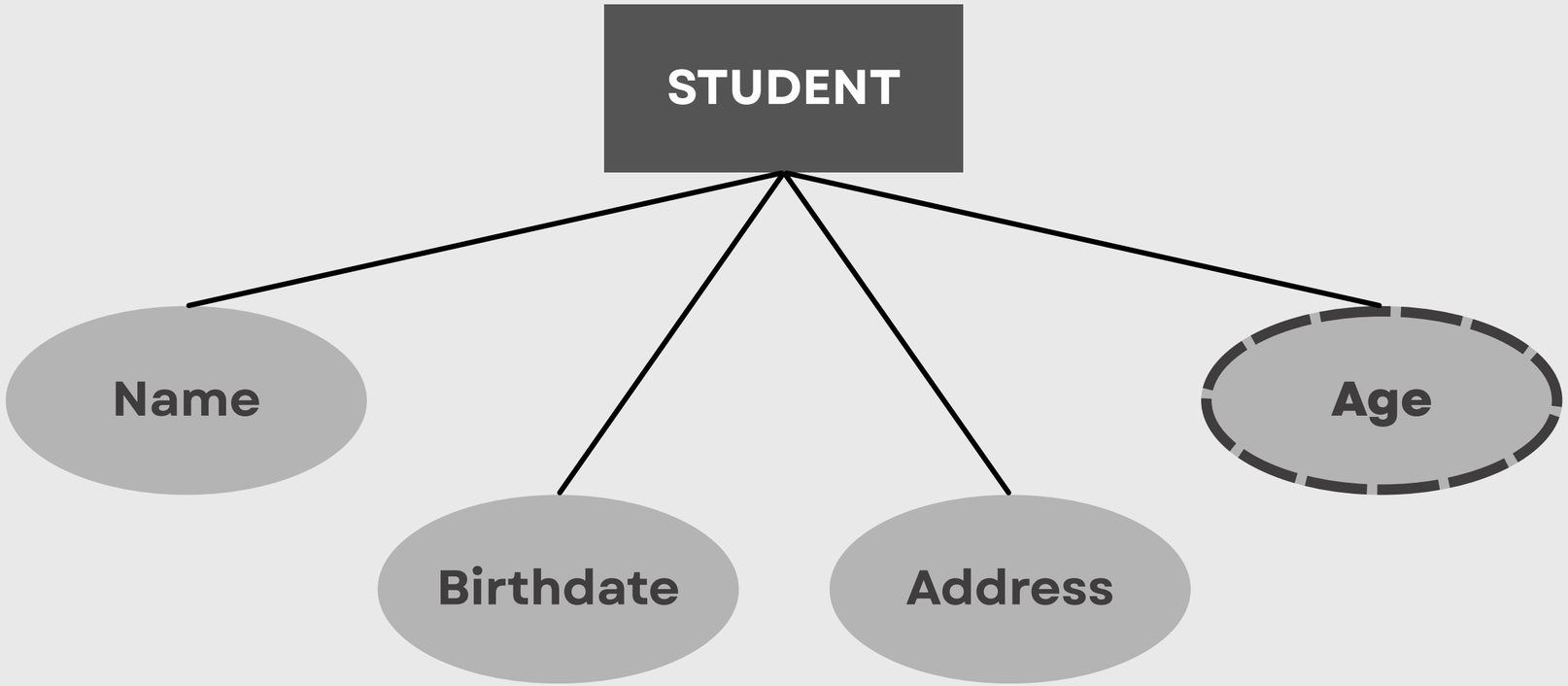
3. DERIVED ATTRIBUTE

- Attributes that contains values calculated from other attributes.
- They are not stored directly in the database but are obtained from stored values.
- Represented by a dashed oval.

TYPES OF ATTRIBUTES

DERIVED ATTRIBUTE

Example:



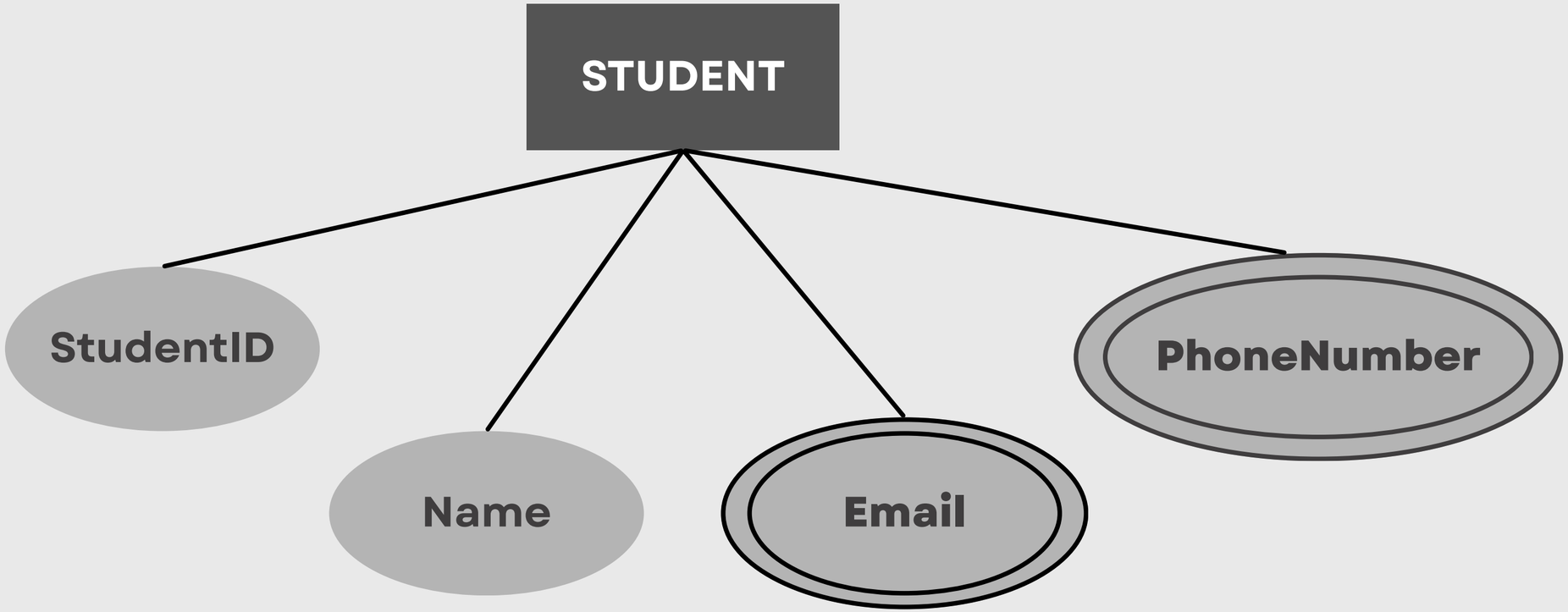
4. MULTIVALUED ATTRIBUTE

- Attributes that have a set of values for each entity.
- Allows more than one value to be stored for a single entity instance.
- Multivalued attributes become entities of their own.
- Represented by a double oval.

TYPES OF ATTRIBUTES

MULTIVALUED ATTRIBUTE

Example:



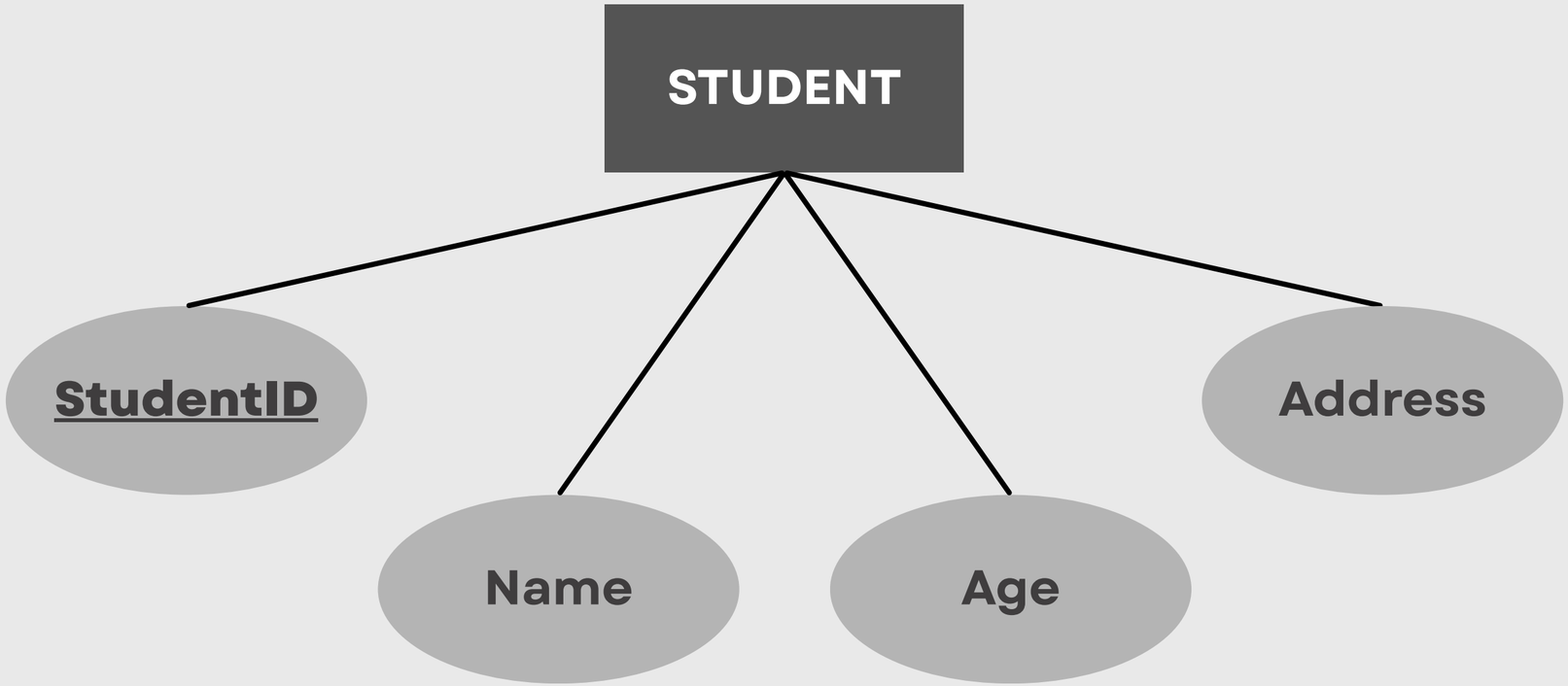
5. KEY ATTRIBUTE

- An attribute that uniquely identifies each entity instance.
- Ensures that no two entities can have the same value for this attribute.
- Represented by an oval with the attribute name underlined.

TYPES OF ATTRIBUTES

KEY ATTRIBUTE

Example:



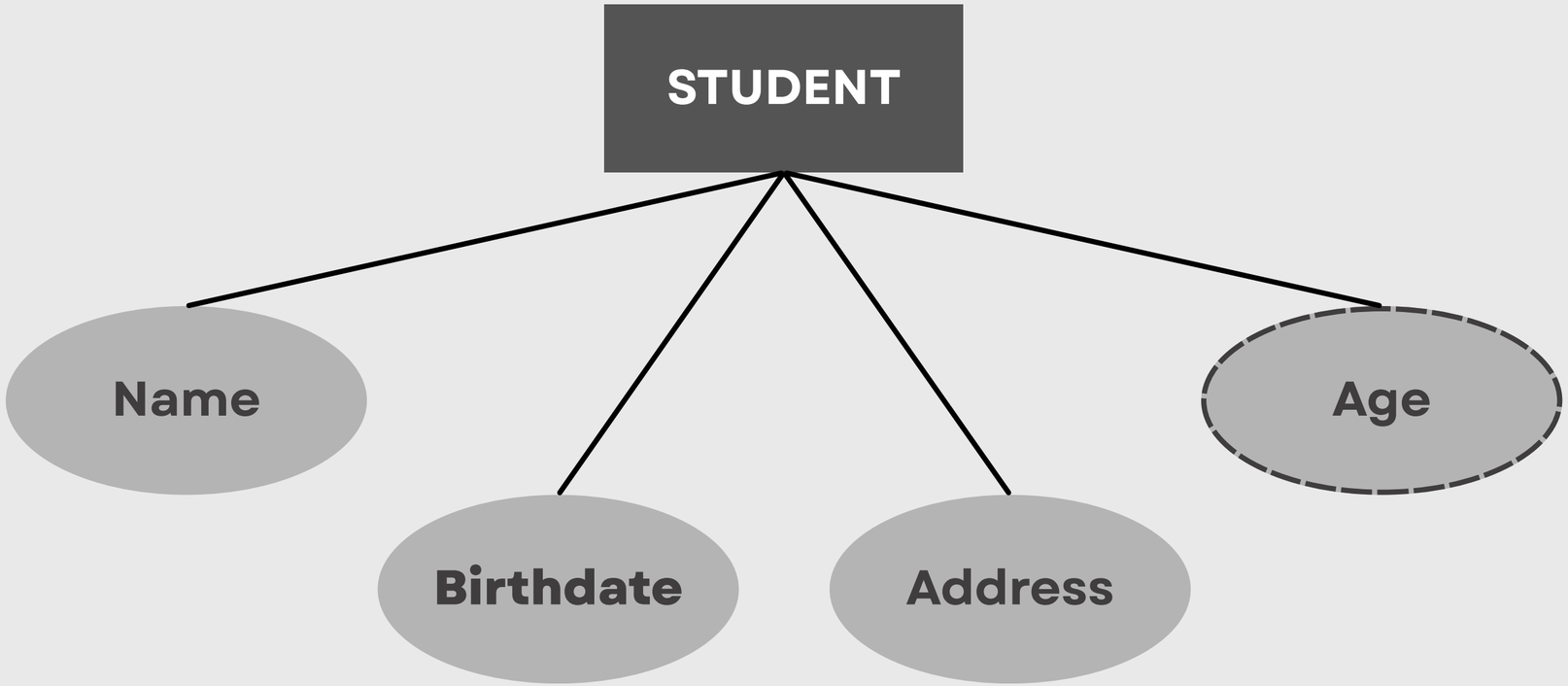
6. STORED ATTRIBUTE

- An attribute whose value is stored directly in the database.
- Doesn't require any type of update.
- Represented by a regular oval.

TYPES OF ATTRIBUTES

STORED ATTRIBUTE

Example:



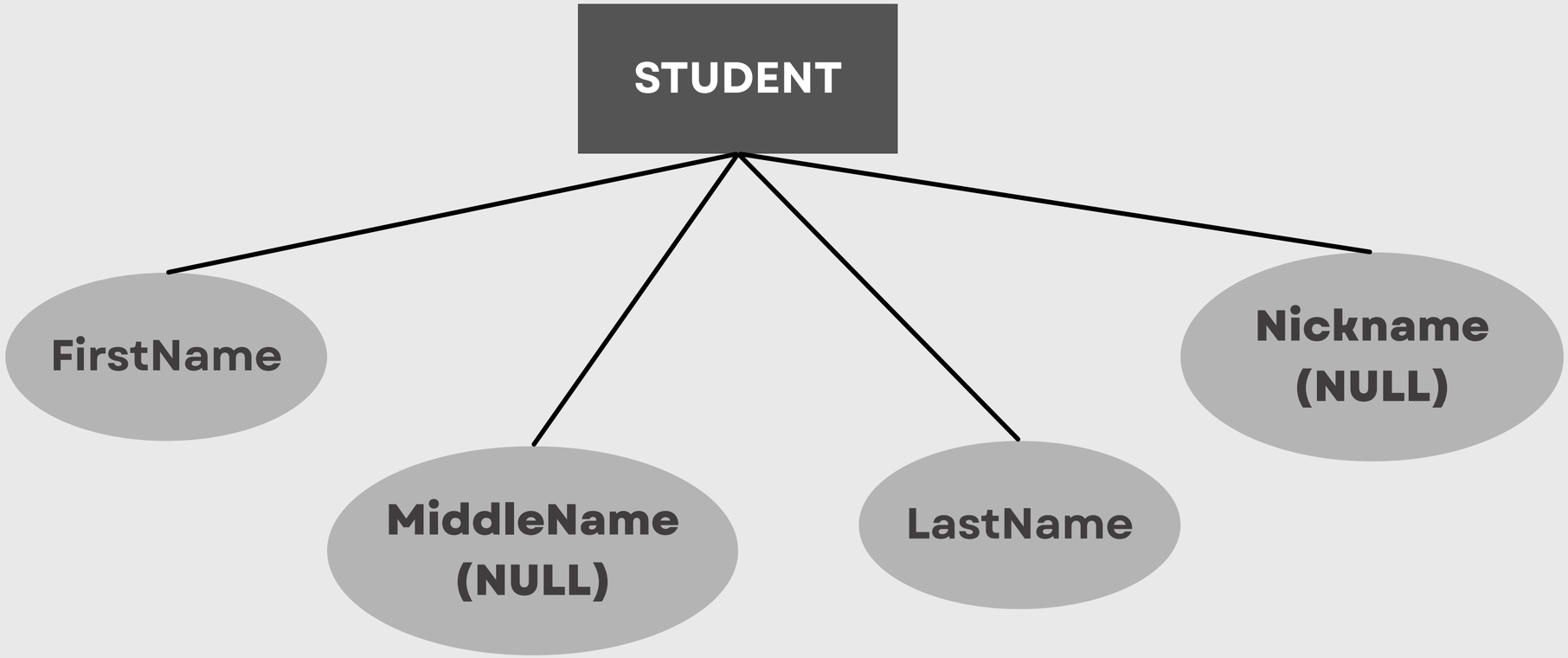
7. NULL ATTRIBUTE

- An attribute that may not always have a value.
- Can be empty or unknown.
- Used when data is optional or not applicable.
- Represented by a regular oval.

TYPES OF ATTRIBUTES

NULL ATTRIBUTE

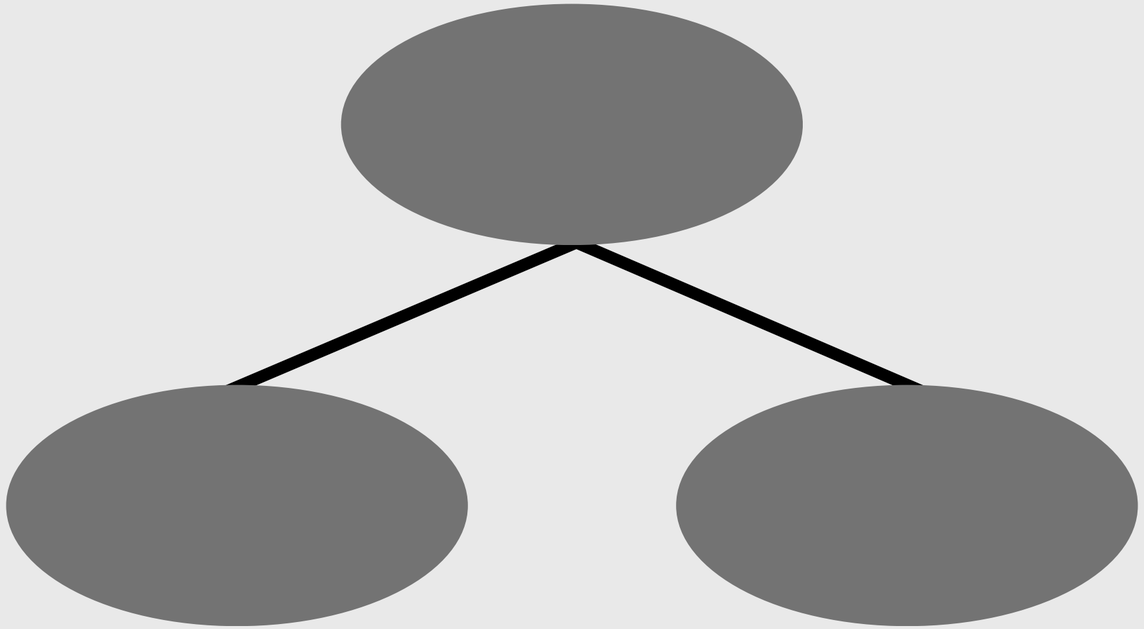
Example:



SHORT ACTIVITY

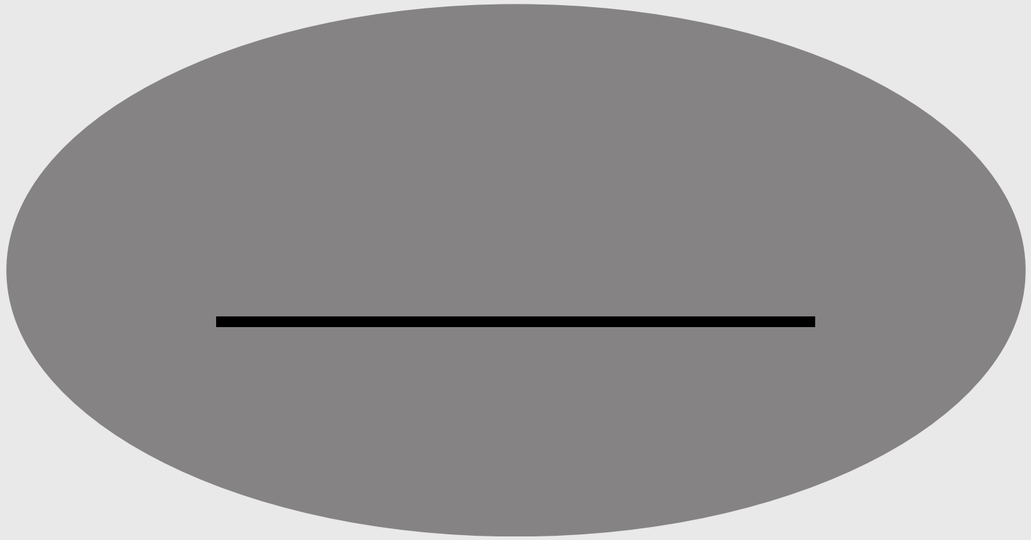
IDENTIFY THE **TYPE OF ATTRIBUTE** BY SHAPE

WHAT TYPE OF ATTRIBUTE IS THIS?



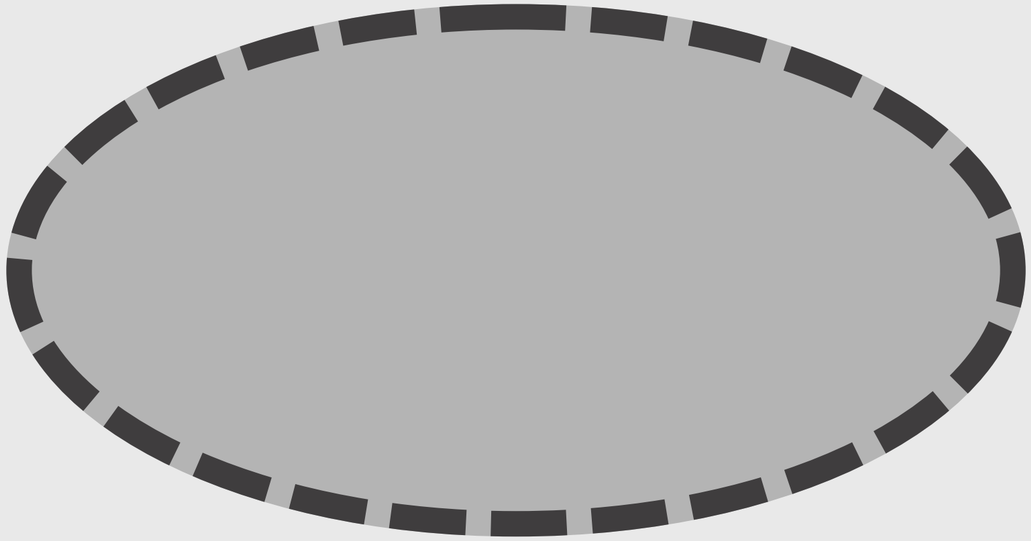
COMPOSITE ATTRIBUTE

WHAT TYPE OF ATTRIBUTE IS THIS?



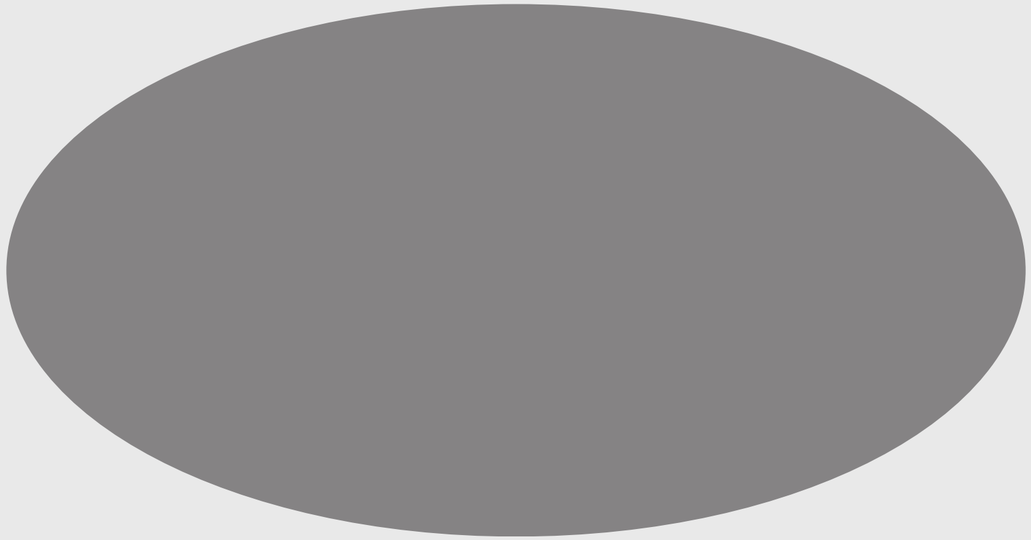
KEY ATTRIBUTE

WHAT TYPE OF ATTRIBUTE IS THIS?



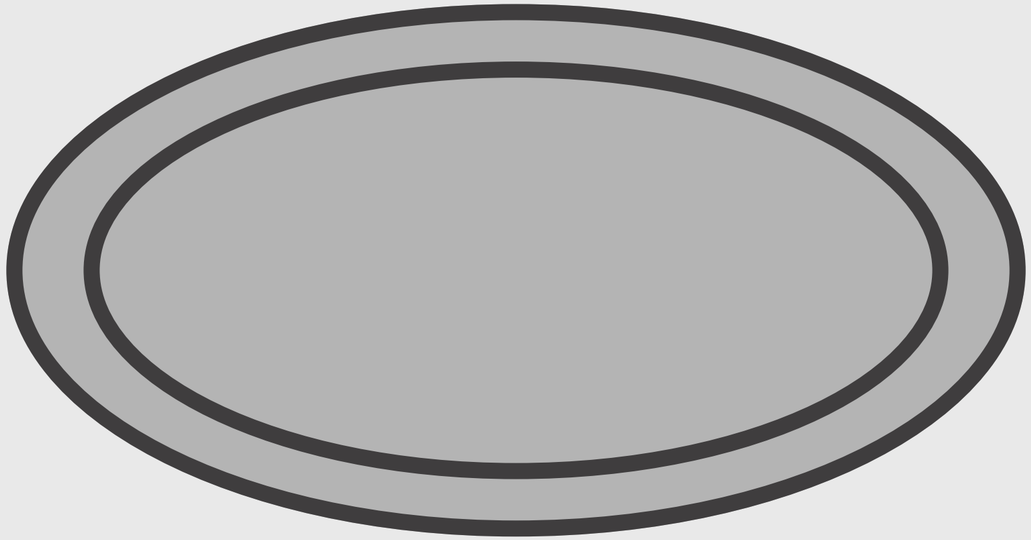
DERIVED ATTRIBUTE

WHAT TYPE OF ATTRIBUTE IS THIS?



SIMPLE ATTRIBUTE

WHAT TYPE OF ATTRIBUTE IS THIS?



MULTIVALUED ATTRIBUTE

WHAT IS NULL?

GIANN MIGUEL VILLAROSA

NULL

In a database, **NULL** represents a **missing**, **unknown**, or **inapplicable** value.
It does not mean zero, empty string or false. It simply means the value does **not exist** (yet or at all).

In short, it doesn't mean "nothing", it means "**we don't know**" (yet).
It represents **absence of data**, not absence of meaning.

WHAT IS NULL?

NULL

In a database, NULL represents a **missing**, **unknown**, or **inapplicable** value.

IMPORTANT NOTE

NULL DOES NOT MEAN

0

ZERO

""

EMPTY STRING

FALSE

FALSY VALUES

WHAT IS NULL?

In short, it **doesn't mean** "nothing", it means "**we don't know (yet).**" It represents absence of data, not absence of meaning.

TYPES OF
KEYS

KEYS

are special fields (or combinations of fields) used to identify records and establish relationships between tables.

In short, a key tells the database “which record is which” and “how tables are linked.

TYPE OF KEYS

1. PRIMARY KEY

- A unique identifier for each record in a table.
- Must be **UNIQUE**
- Cannot be **NULL**.

TYPE OF KEYS

Students

Student_ID	Name	Age
1980	Jude	45
2023	Faju	24

In this table, the **Student_ID** is the **Primary Key.** It should be **Unique** and it cannot be **NULL**

2. CANDIDATE KEY

- Any attribute (or combination of attributes) that can uniquely identify a record.
- There can be multiple candidate keys in a table.
- Cannot be **NULL**

TYPE OF KEYS

Students

Student_ID	Email	Age
1980	judegaling@gmail.com	45
2023	fajyou@gmail.com	24

In this table, the **Student_ID** is the can act as the **Primary Key** and the **Email** will act as the **Alternate Key.** Both should be **Unique** and it cannot be **NULL**

3. ALTERNATE KEY

- A candidate key that is not chosen as the primary key.
- Acts as a backup unique identifier.
- Cannot be **NULL**

TYPE OF KEYS

Students

Student_ID	Email	Age
1980	judegaling@gmail.com	45
2023	fajyou@gmail.com	24

Similar to the previous slide, the **Primary Key** here is the **Student_ID** while the **Alternate Key** is the **Email.**
It is unique, just not the main key.

4. FOREIGN KEY

- A column (or group of columns) in one table that references the Primary Key of another table.
- It establishes a relationship between two tables.
- Can be **NULL**

TYPE OF KEYS

Student

Student_ID	Name	Age
1980-131	Jude	45
2023-371	Faju	24

Orders

ORD_ID	Student_ID	Item
101	2023-371	Lanyard
102	1980-131	T-shirt

In the **Orders** table, **Student_ID** is a **Foreign Key** referencing **Student**(**Student_ID**). The uniqueness ensures that each order belongs to a customer.

5. COMPOSITE KEY (OR COMPOUND KEY)

- A Composite Key uses two or more columns together to uniquely identify a record.
- It can be used when no single column can uniquely identify a record.
- Cannot be **NULL**

TYPE OF KEYS

Grades

Student_ID	Course_ID	Grade
1980-131	**WRS**	1.5
2023-371	FR01	1.0
2023-371	**WRS**	1.5

The **Composite Key** here is the **Student_ID** and **Course_ID,** a student can take multiple courses, but the combination of both ID's uniquely identifies each record

6. SUPER KEY

- A Super Key is any set of attributes that can uniquely identify a record, possibly including extra attributes.
- Represents all possible unique identifiers, not necessarily minimal.
- Cannot be **NULL**

TYPE OF KEYS

Student

Student_ID	Name	Email
1980-131	Jude	judegaling@gmail.com
2023-371	Faju	fajyou@gmail.com

The **Super Keys** here are:

- Student_ID
- Email
- Student_ID, Name
- Student_ID, Email

Only the **Student_ID** and **Email** are minimal unique identifiers,
so they're **Candidate Keys.** The others are Super Keys with extra fields.

7. UNIQUE KEY

- A Unique Key ensures that all values in a column are unique, but it can contain NULL values (unlike Primary Key).
- To enforce uniqueness on optional fields.
- Can be **NULL**

TYPE OF KEYS

Students

Name	Email	Grade
Jude	judegaling@gmail.com	**NULL**
Faju	fajyou@gmail.com	1.0
Tagana	kamukhanialden@gmail.com	**NULL**

The **Unique Key** here is the **Email,** the **Grade** can be **NULL** and even repeated.

TYPES OF
RELATIONSHIPS

RUY INIGO FAJUTAGANA

RELATIONSHIPS

- Relationships are the glue that holds the tables together. They are used to connect related information between tables.

- When two (or more) entities connect, there are different “how many” rules. These are captured by cardinality (max number) and participation / minimum (0 or 1, etc.)

RELATIONSHIP

- A relationship shows how two (or more) entities are connected to each other in the real world.

- Student **enrolls** in a Class

!=

CARDINALITY

- Cardinality tells us how many entities can be connected in that relationship.

- A Teacher teaches **many** Students

CARDINALITY

- Maximum entities that can be connected to that relationship

- Might have **one or many** relations

!=

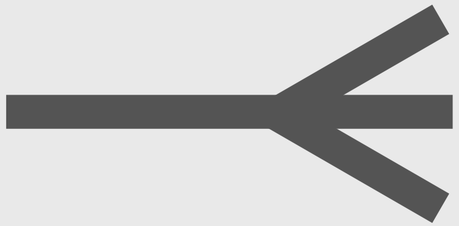
MODALITY

- Minimum entities that can be connected to that relationship

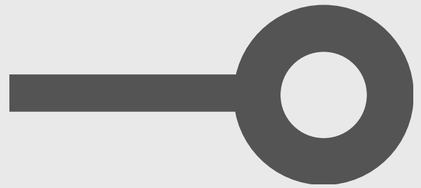
- Mandatory or optional

ONE

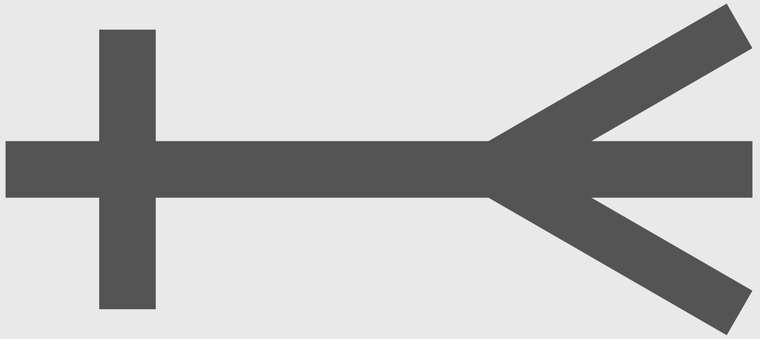
MANY



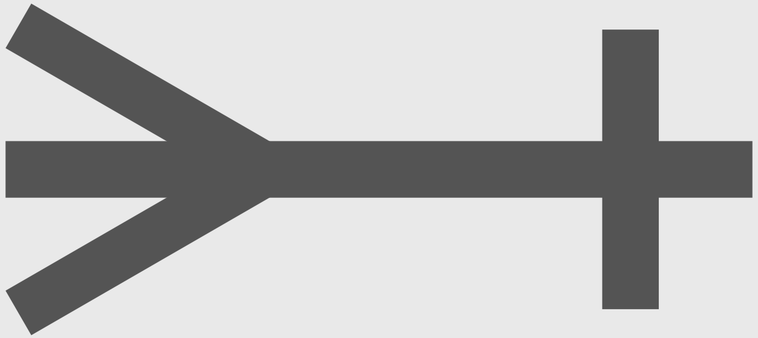
ZERO



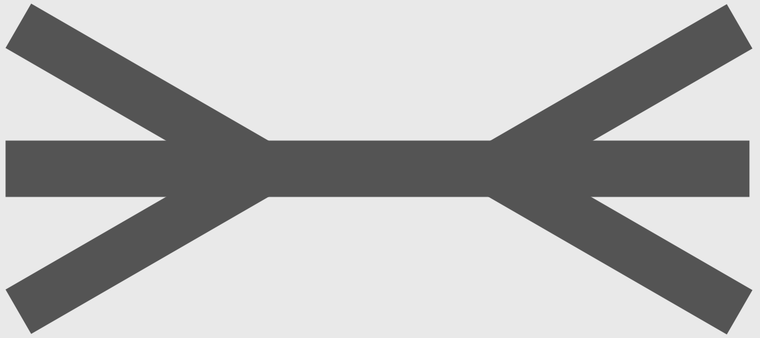
ONE TO ONE



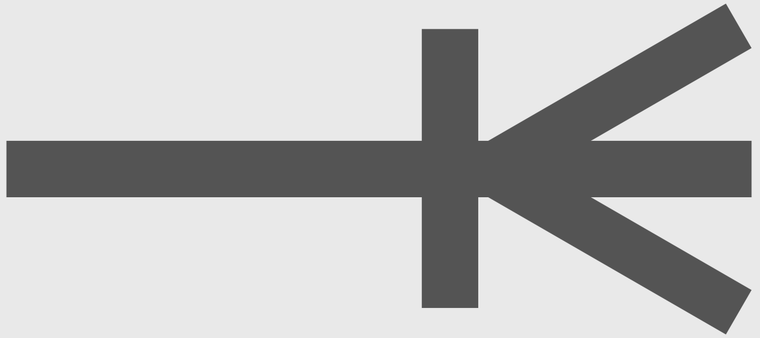
ONE TO MANY



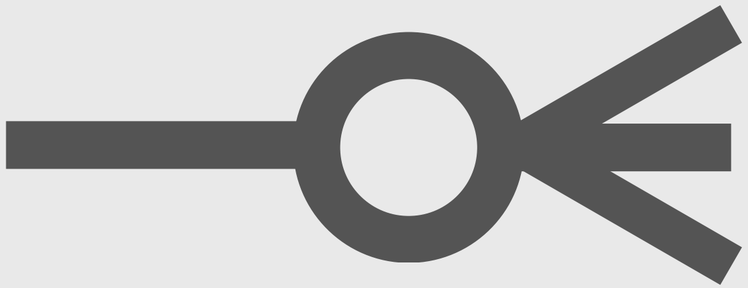
MANY TO ONE



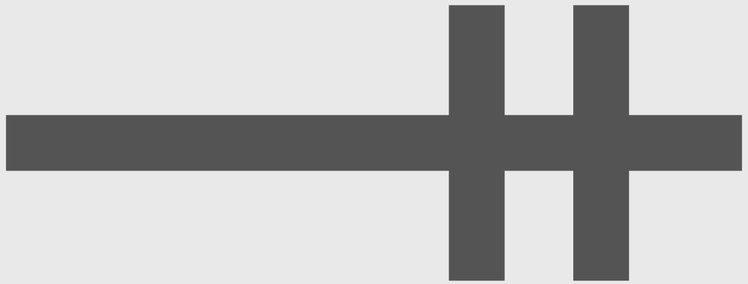
MANY TO MANY



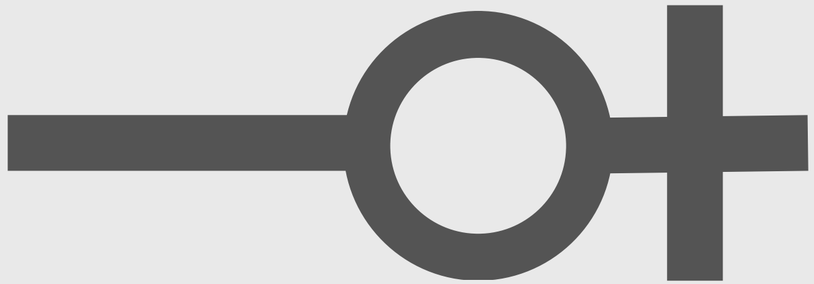
ONE OR MANY



ZERO OR MANY

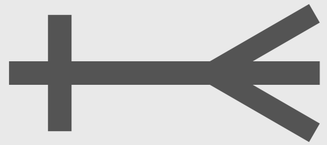


ONE AND ONLY ONE



ZERO OR ONE

ONE TO MANY 1:N



Indicates that a single instance of an entity can be linked to multiple instances of another entity.

Example: One customer can place many orders.

Example: One customer can place many orders.

Customer Table

Customer ID	Name	Address
280	Jude	Pasig , Somewhere
281	Giann	Pasig, Din
282	Luther	Tuguegarao

Primary Key (Unique)

Order Table

Order ID	Item	Price	Customer ID
A123	Musubi	55	280
B456	Jolly Hotdog	78	280
C789	Mouse Pad	350	280

Primary Key (Unique)

Foreign Key

ONE TO ONE 1:1

A one-to-one relationship exists when one instance of an entity is associated with exactly one instance of another entity.

Example: One StudentNumber is issued to only one student.

Example: One StudentNumber is issued to only one student.

Student Accounts

StudentID	Date Created
102345	10-6-2025
103491	10-6-2025
106741	10-6-2025

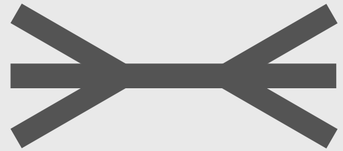
Primary Key (Unique)

Class List

StudentID	Name	Age
102345	Magnum	30
103491	Gichiji	21
106741	Voltage	20

Primary Key (Unique)

MANY TO MANY 1:1



A many-to-many relationship occurs when multiple instances of one entity can be related to multiple instances of another.

Example: Many students can develop many projects.

Example: Many students can develop many projects.

Student Table

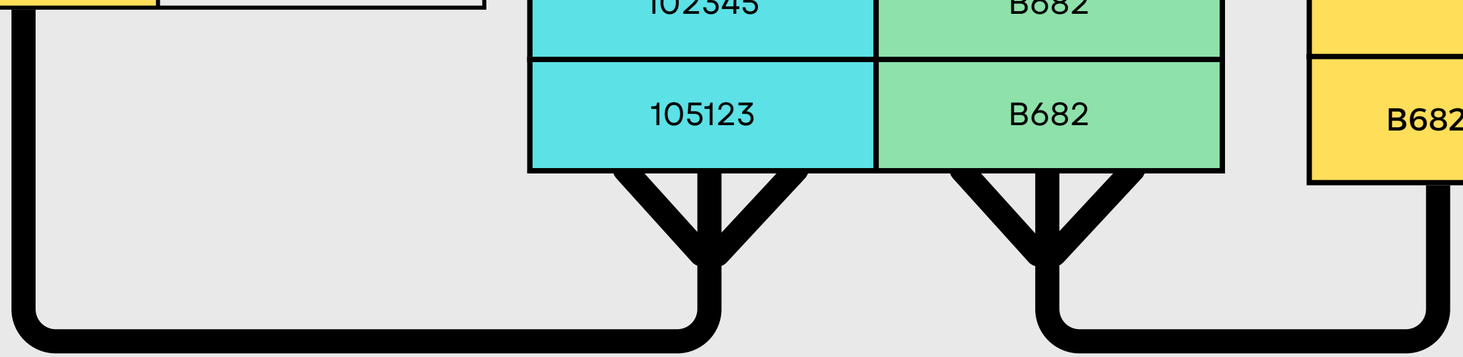
StudentID	Student Name
102345	Luther
105123	Cornelius
106741	Ruy
101291	Jude

Junction Table

Student ID	ProjectID
102345	K459
105123	L627
106741	L627
101291	F951
102345	B682
105123	B682

Project Table

ProjectID	Project Name	Description
F951	Compass	Travel Booking
L627	Campus Strike	FPS Game
K459	CSS Tutorial	Educational Website
B682	Komyut	Mobile App



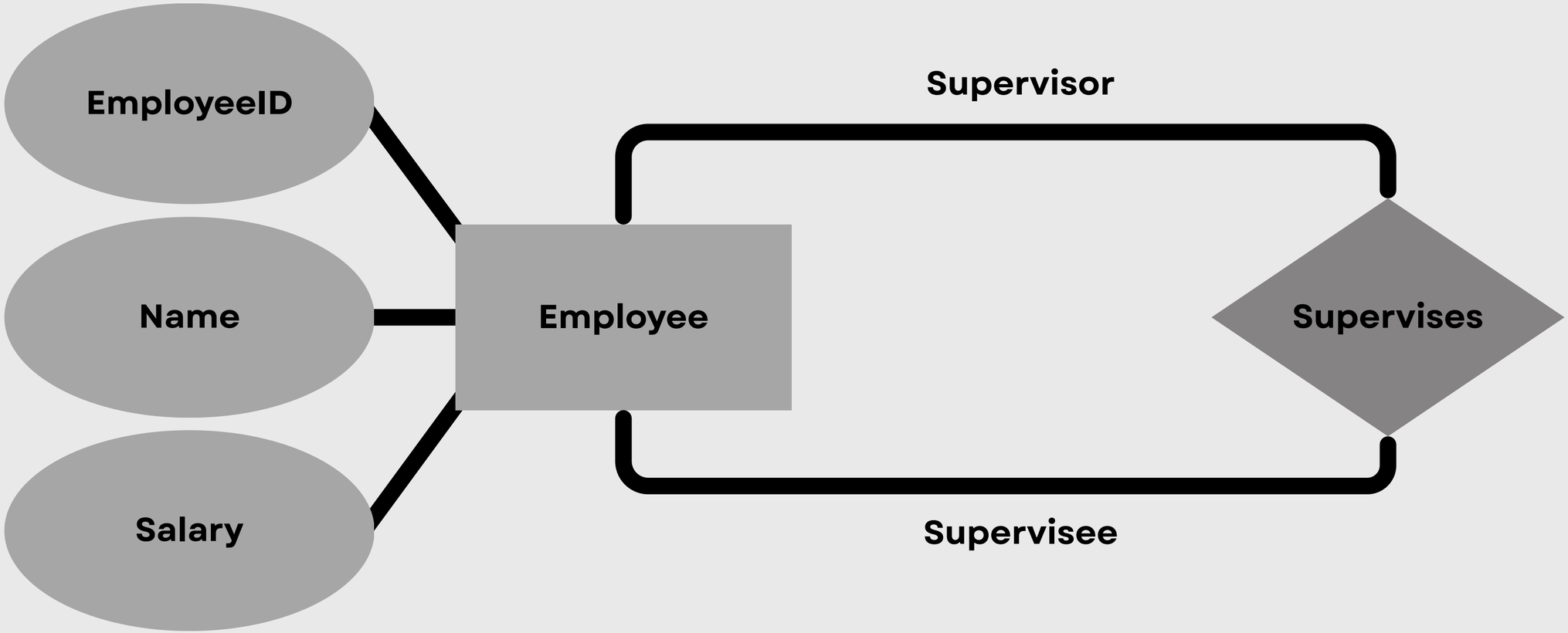
RECURSIVE / UNARY RELATIONSHIP

An entity relates to another record of the same entity type.

Example:

- A team plays against another team
- An organizational unit reports to another organizational unit

Example: An employee supervises another employee.



TERNARY / N-ARY RELATIONSHIPS

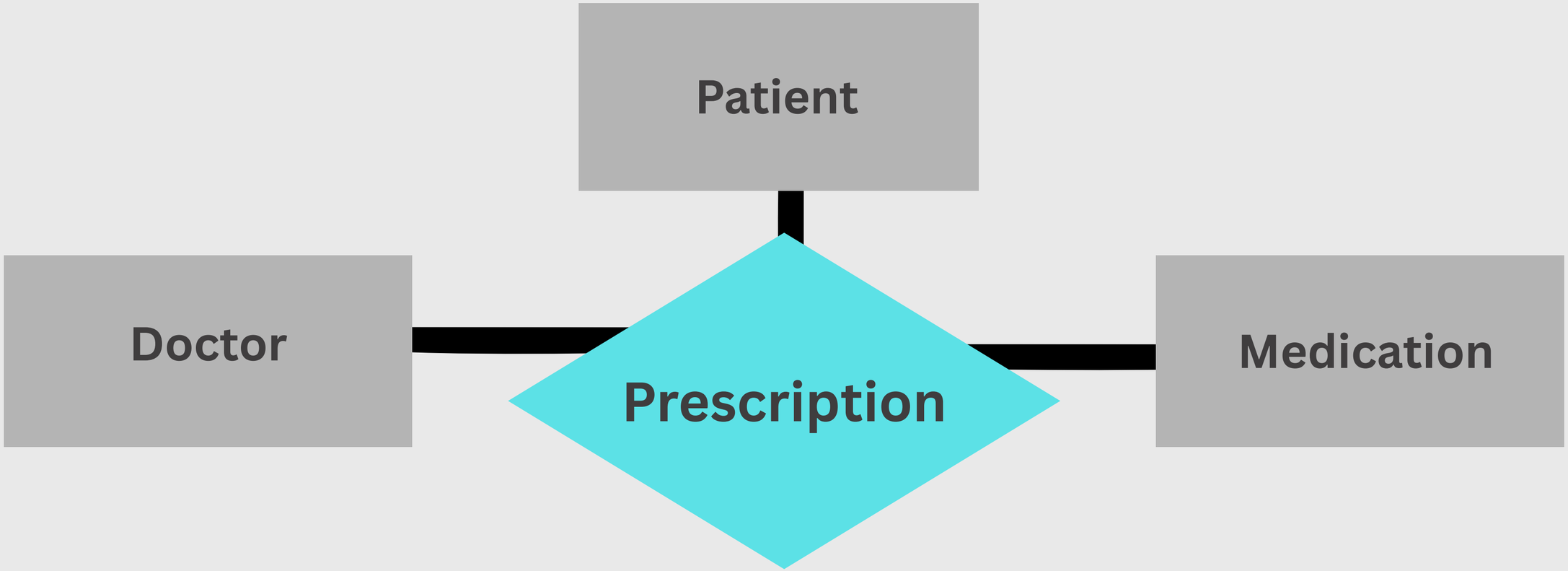
More than two entities involved in a single relationship.

Example:

- Doctor + Patient + Medication => Prescription
- The Prescription entity connects those three.

Example:

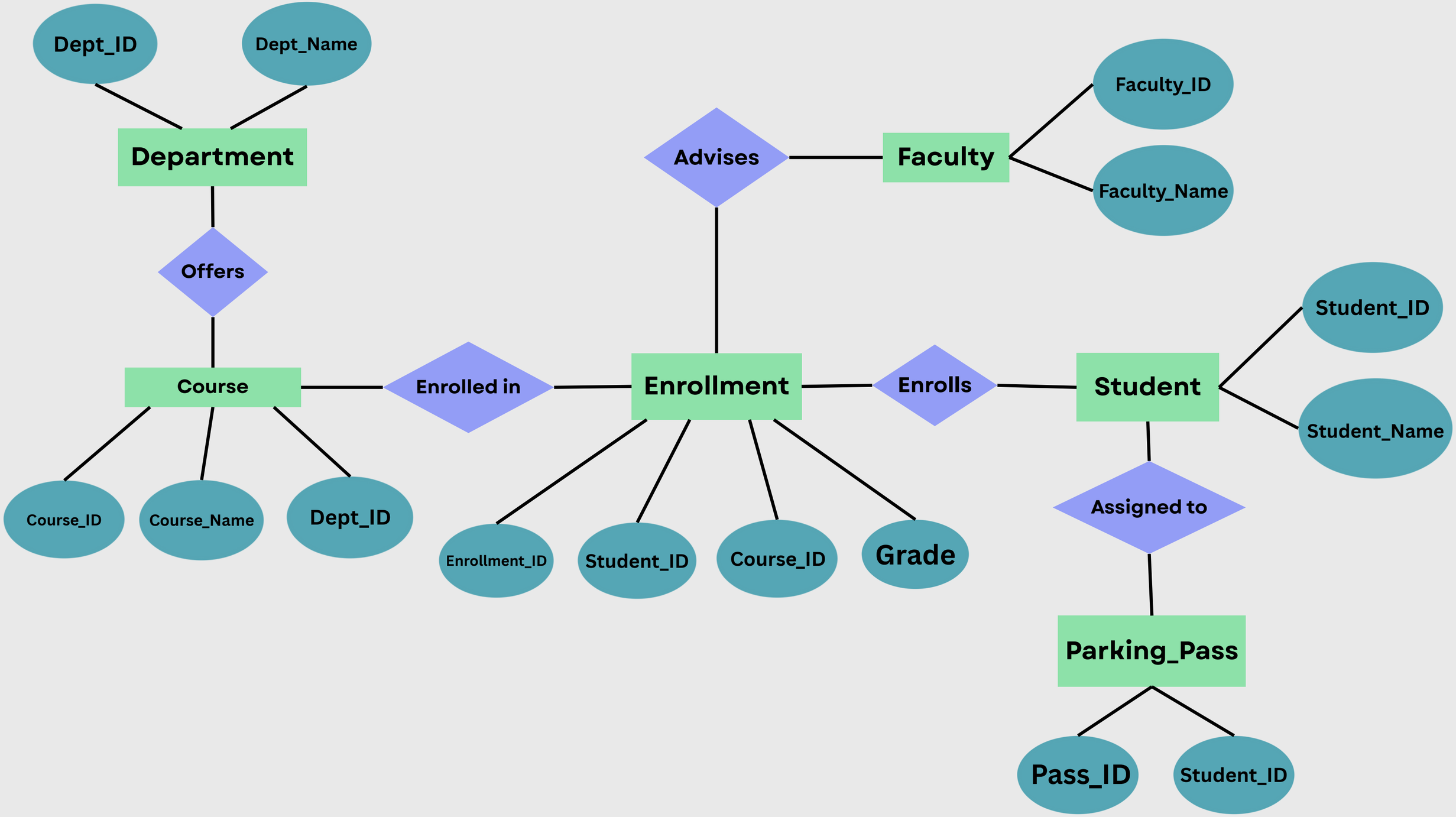
- Doctor + Patient + Medication => Prescription
- The Prescription entity connects those three.



DATABASE LOGICAL/ CONCEPTUAL REPRESENTATION

ENTITY-RELATIONSHIP MODEL

CORNELIUS JAMES LASALA


Dept_ID
Dept_Name
Department
Offers
Course
Course_ID
Course_Name
Dept_ID
Enrolled in
Advises
Faculty
Faculty_ID
Faculty_Name
Enrollment
Enrollment_ID
Student_ID
Course_ID
Grade
Enrolls
Student
Student_ID
Student_Name
Assigned to
Parking_Pass
Pass_ID
Student_ID

PHYSICAL REPRESENTATION

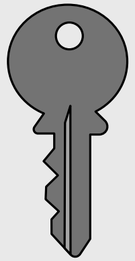
ENTITY-RELATIONSHIP DIAGRAM

MYSQL WORKBENCH

ENTITY-RELATIONSHIP DIAGRAM

A graphical tool used in database design and data modeling to visualize how different data entities are connected within a system

SYMBOLS AND REPRESENTATIONS



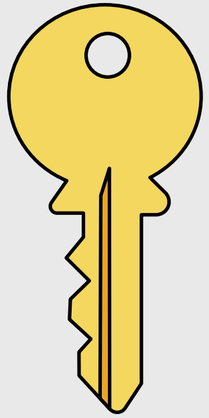
PRIMARY KEYS

ATTRIBUTES

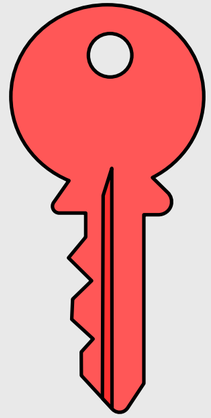
COLORS

SYMBOLS AND REPRESENTATIONS

PRIMARY KEYS



SIMPLE PRIMARY KEY



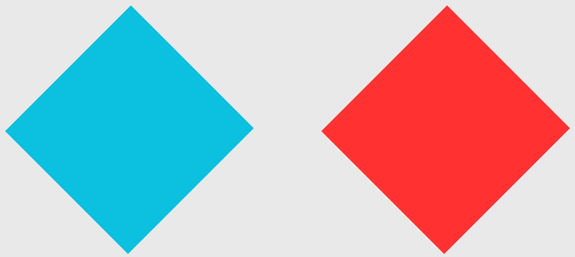
FOREIGN PRIMARY KEY

ENTITY-RELATIONSHIP DIAGRAM

MYSQL WORKBENCH

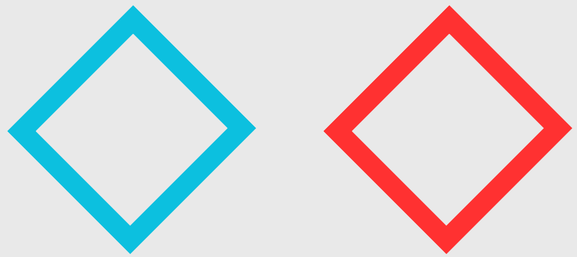
SYMBOLS AND REPRESENTATIONS

ATTRIBUTES



SOLID DIAMONDS

NOT NULL



NOT FILLED DIAMOND

NULL

ENTITY-RELATIONSHIP DIAGRAM

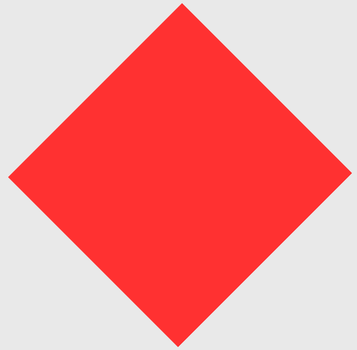
MYSQL WORKBENCH

SYMBOLS AND REPRESENTATIONS

COLORS

BLUE

SIMPLE KEY



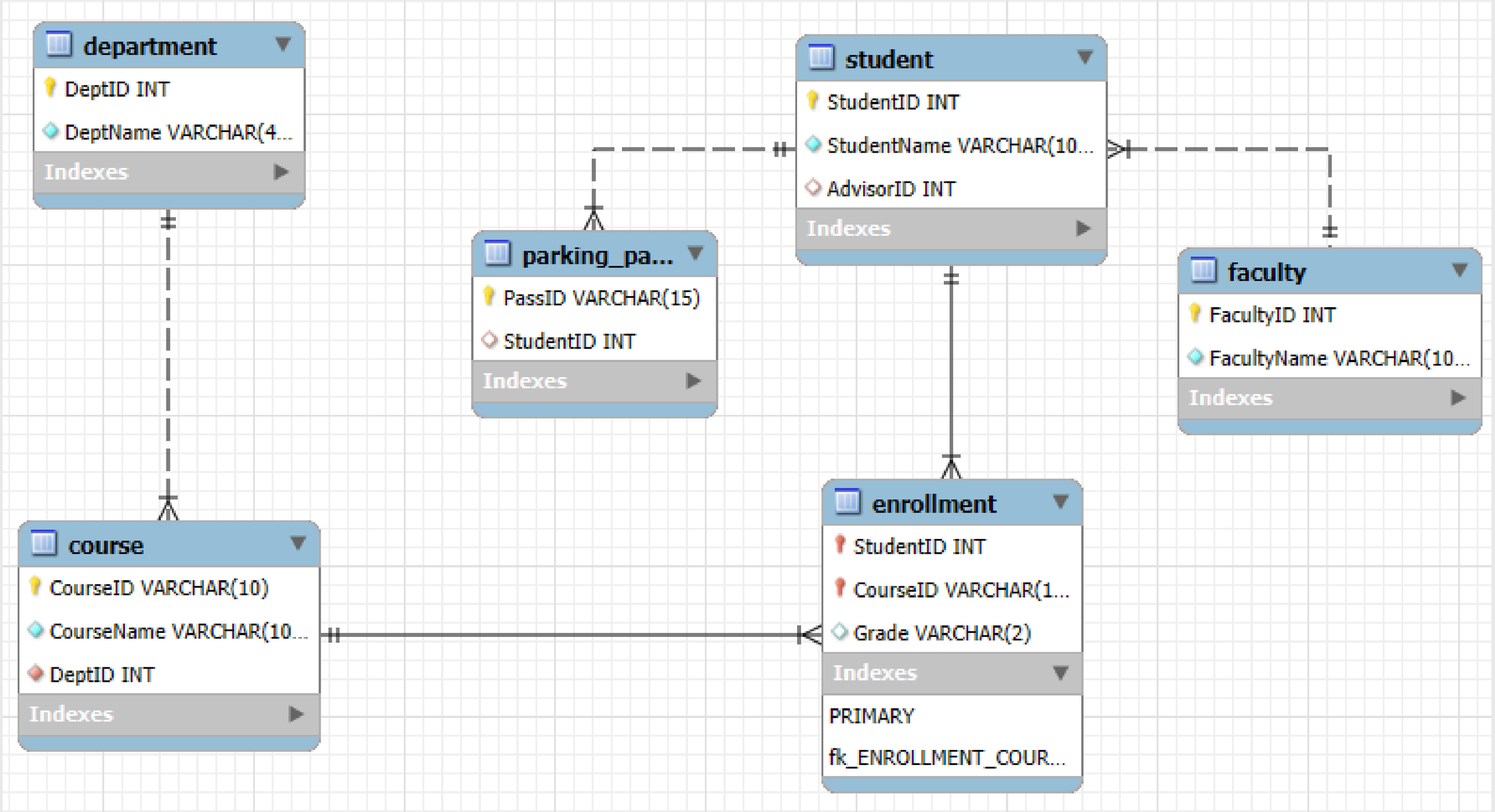
RED

FOREIGN KEY

ENTITY RELATIONSHIP DIAGRAM

LIVE DEMONSTRATION

MYSQL WORKBENCH


department
DeptID INT
DeptName VARCHAR(4...
Indexes
student
StudentID INT
StudentName VARCHAR(10...
AdvisorID INT
Indexes
parking_pa...
PassID VARCHAR(15)
StudentID INT
Indexes
faculty
FacultyID INT
FacultyName VARCHAR(10...
Indexes
course
CourseID VARCHAR(10)
CourseName VARCHAR(10...
DeptID INT
Indexes
enrollment
StudentID INT
CourseID VARCHAR(1...
Grade VARCHAR(2)
Indexes
PRIMARY
fk_ENROLLMENT_COUR...

THANK YOU PI

Hey Jude!

Fajutagana, Ruy Inigo

Lasala, Cornelius James

Malicdem, Andrea Sai

Marquez, Zach Luther

Villarosa, Giann Miguel