DATA MODELLING AND ENTITY RELATIONSHIP DIAGRAMS (ERDs)

DATABASE DESIGN AND PROGRAMMING — TOPIC 2

— Mutebi Bashir

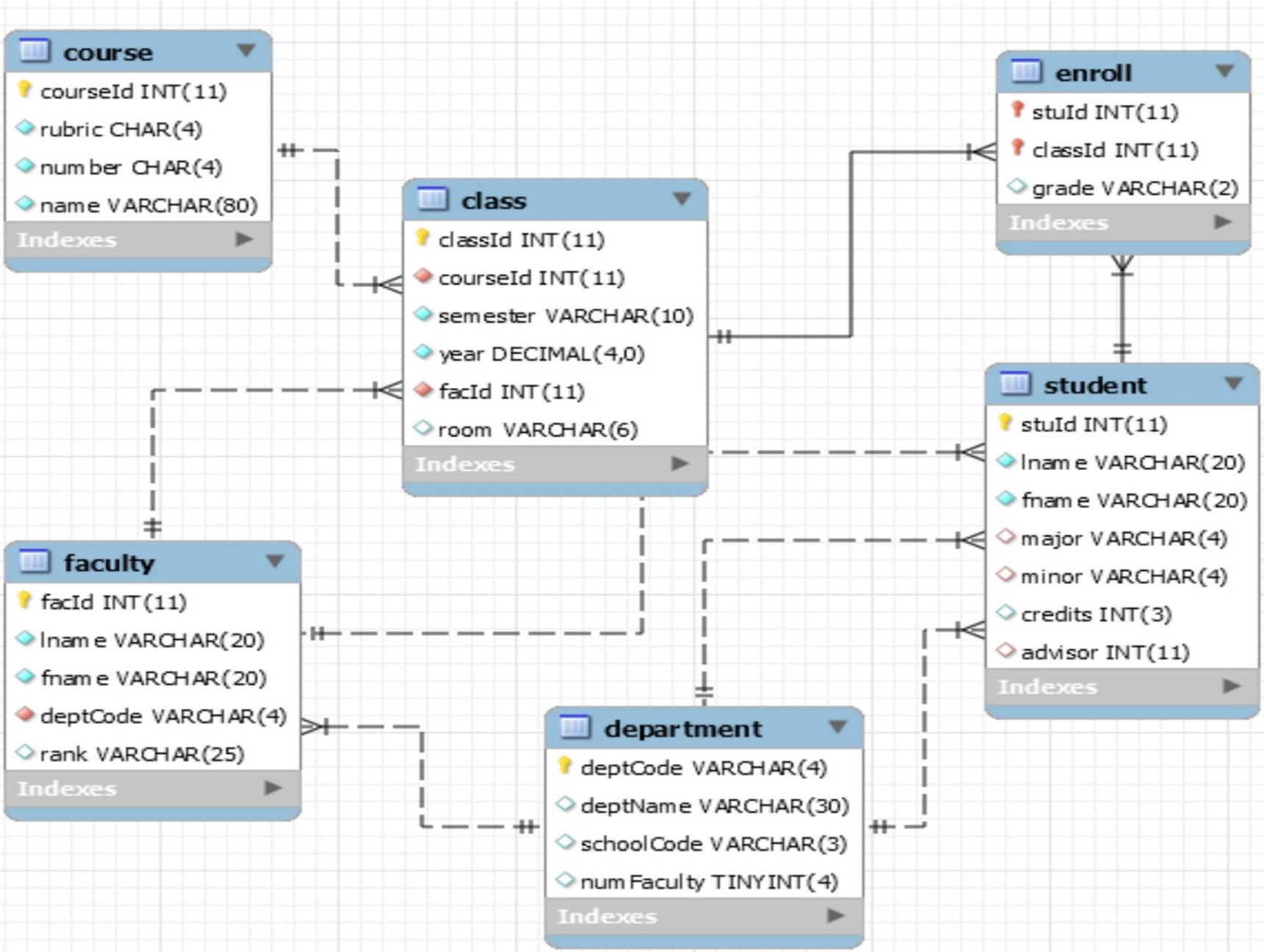
Objectives

- Understand the importance of data modeling in database design.
- Explore the levels of data modeling: conceptual, logical, and physical.
- Learn about Entity Relationship Diagrams (ERD) and their role in visualizing data.
- Master the steps of creating an ERD, from entity identification to diagram construction.
- Appreciate the advantages of using ERDs in database design and development.

Data Modelling & ERDs

- Data modelling refers to the process of developing visual representations of the data that will be kept in the database.
- The visual representations of data modeling are diagrams that show data elements, their relationships and their constraints i.e. ERDs.
- Data modeling and entity relationship diagrams (ERDs) are like the blueprints for organizing information in a structured way.
- Data modelling can be done at three levels; ***conceptual model, logical model*** and ***physical model.***

Data Modelling & ERDs ..



Why Data Modelling?

- Organizing Information: It helps to organize complex information into a structured format, making it easier to understand and manage.
- Improves communication because it t provides a common language for stakeholders to discuss and understand the data requirements of a system.
- Data models serve as blueprints for database design and application development thus guiding the development process.
- Enhances data quality by defining clear structures and relationships, data modeling improves the quality and consistency of data.
- Improves system performance due to properly modeled data structures that optimize data retrieval and manipulation processes.

Levels of Data Modelling - Conceptual

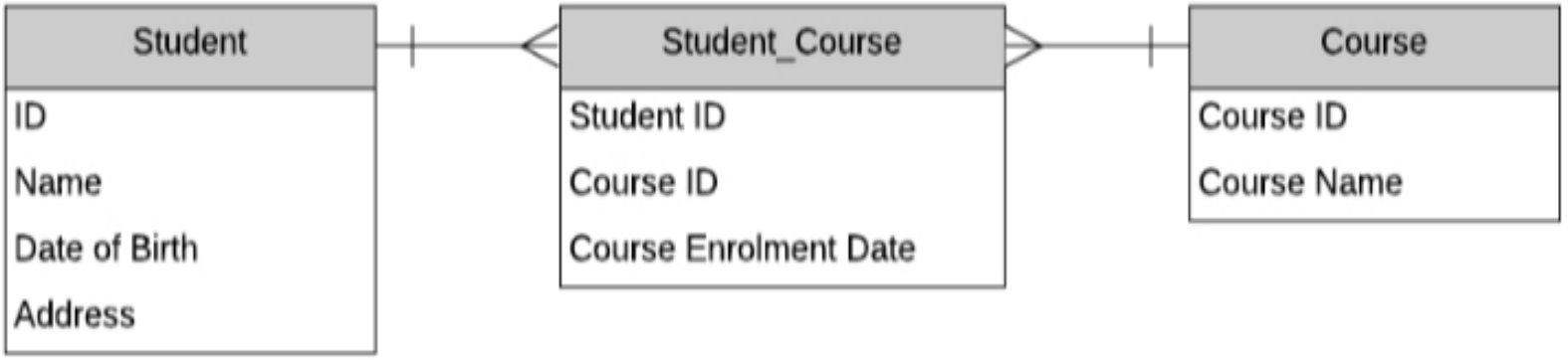
- The **conceptual data model** provides a high-level overview of the system's contents.
- It focuses on what the system contains rather than how it will be implemented and thus it answers the “what” of the system.
- For example, consider designing a system for a university. At the conceptual level, the model would identify entities such as “Student” and “Course”
- Relationships between these entities, like "Enroll" would also be defined.
- This level of abstraction allows stakeholders to understand the system's requirements without getting into technical details.

Levels of Data Modelling - Conceptual

Levels of Data Modelling - Logical

- The logical data model provides a more detailed view of the data structures and relationships.
- Data architects and analysts develop this model, focusing on creating a technical map of data structures and rules.
- For instance, in the university system, the logical model would specify attributes for entities such as “Student ID”, “Course Code”
- Primary keys, foreign keys, and relationships between entities would also be defined.
- This level ensures that the database design is robust and comprehensive, independent of the specific technology or DBMS used for implementation.

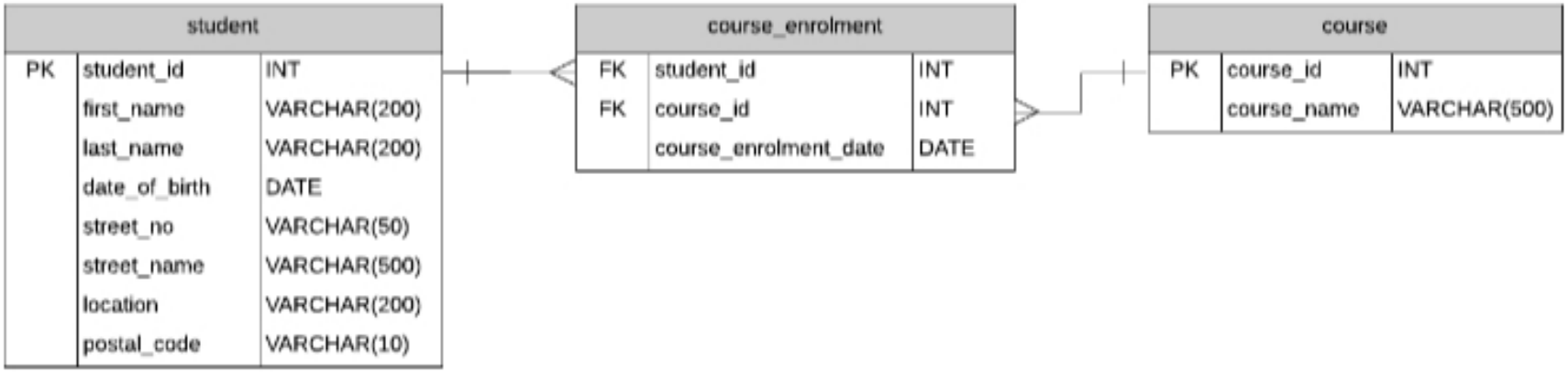
Levels of Data Modelling - Logical



Levels of Data Modelling - Physical

- At the physical data model level, the focus shifts to how the system will be implemented using a particular DBMS or technology.
- Database administrators and developers construct this model, specifying details such as table names, column names, and data types.
- Continuing with the university example, the physical model would include specifics like "Student" table with columns such as "Student_ID," "Name," and "Address."
- It also accounts for optimizations and constraints required for efficient database operations, considering the chosen DBMS's capabilities and requirements.

Levels of Data Modelling - Physical



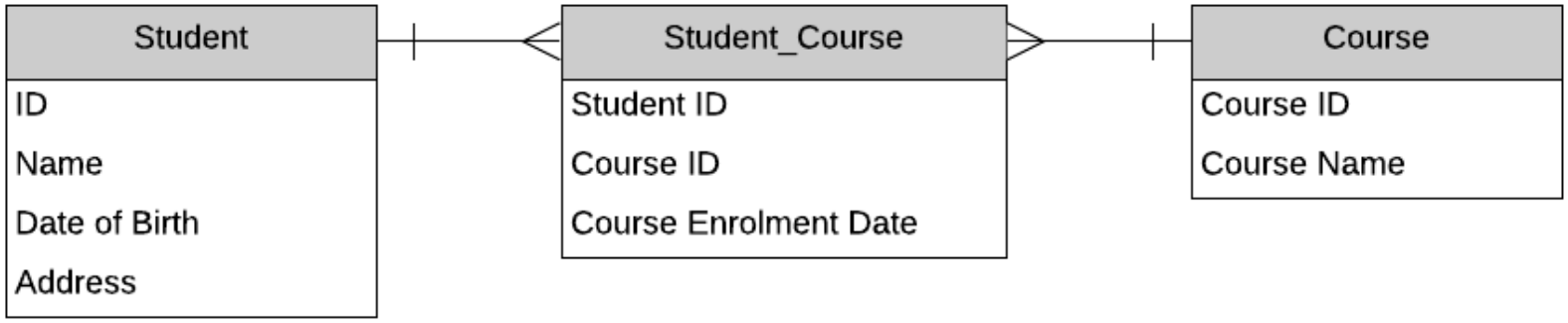
Key differences of the 3 models

FEATURE	CONCEPTUAL	LOGICAL	PHYSICAL
Level of Detail	High level overview	Detailed	Implementation
Audience	Non-technical	Designer, Developers	DBAs, Developers
Purpose	Business requirements	Design and planning	Actual database implementation
Entity Names	Yes	Yes	
Entity Relationships	Yes	Yes	
Attributes		Yes	Yes
Primary Keys		Yes	Yes
Foreign Keys		Yes	Yes
Table Names			Yes
Column Names			Yes
Column Data types			Yes

Entity Relationship Diagrams (ERDs)

- ERDs are graphical representations used to visualize relationships between or amongst entities (e.g. people, places, objects, etc.) in a database or a system.
- They help to visualize and communicate the structure and dependencies of data
- ERDs help to collect and understand the data requirements of a system
- ERDs provide a blueprint for the database schema, enabling the effective planning of entities, attributes, and relationships
- ERDs can be created at three levels: conceptual, logical, and physical
- ERDs use symbols and notations to represent entities, attributes, and relationships

Entity Relationship Diagrams (ERDs)



Source: *A Guide to the Entity Relationship Diagram (ERD)*

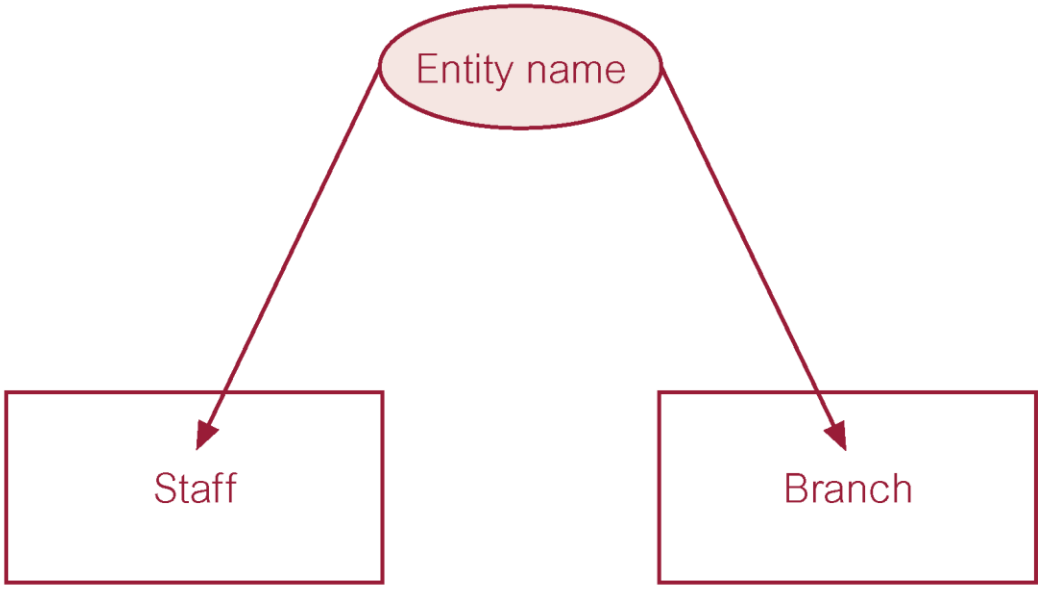
Component of ERDs

- **Entities**
- **Attributes**
- **Keys**
- **Relationship**

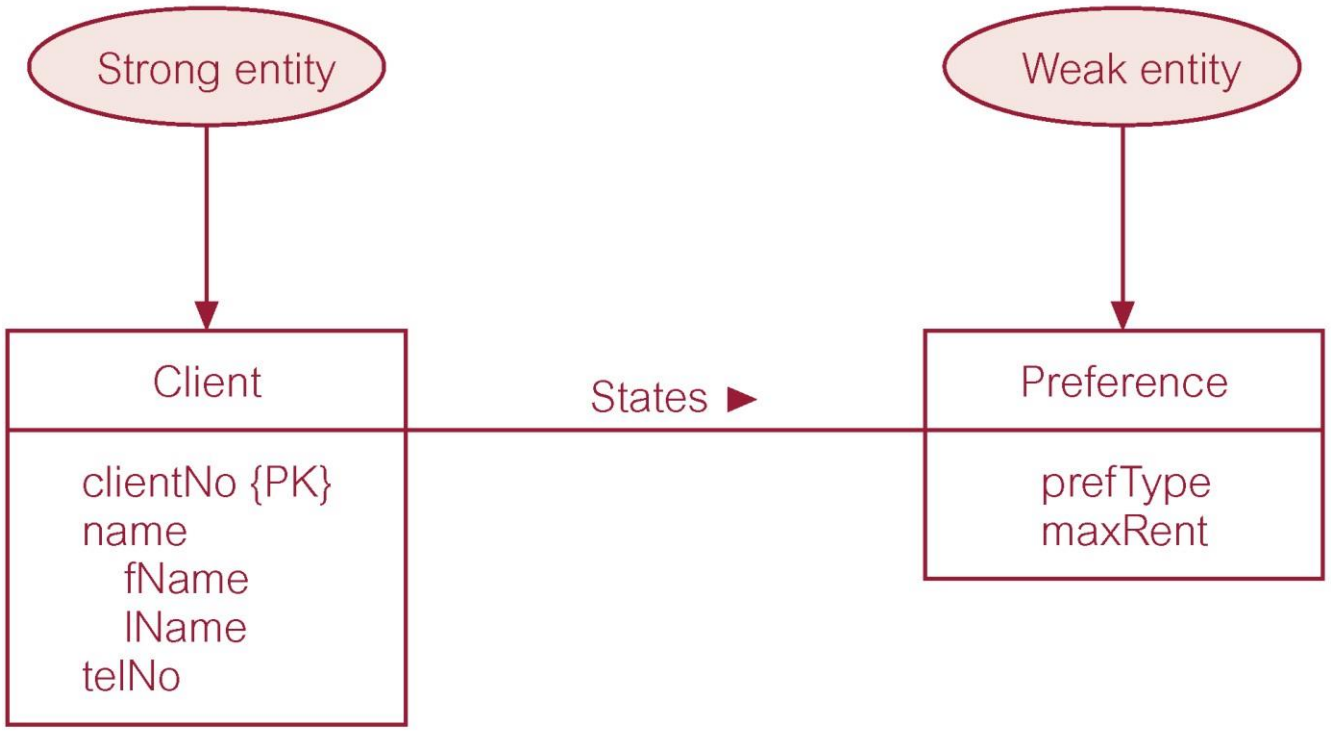
Components of ERDs - Entity

- An entity is anything that can have data stored about it in a database. It can be physical objects, concepts or events (e.g. order, payment, etc.)
- Entities represent a noun in natural language and they are usually represented by rectangles in ER diagrams with entity name inside.
- An entity can be of two types i.e. *strong entity or a weak entity.*
- A strong entity has its own unique ID (primary key) and doesn't rely on other entities. For instance, a student with a student ID is a strong entity as it exists independently.
- Weak entities do not have their own primary keys. Instead, they rely on another entity, known as the identifying or parent entity, for their existence.

Components of ERDs – Entity..



Components of ERDs – Entity..



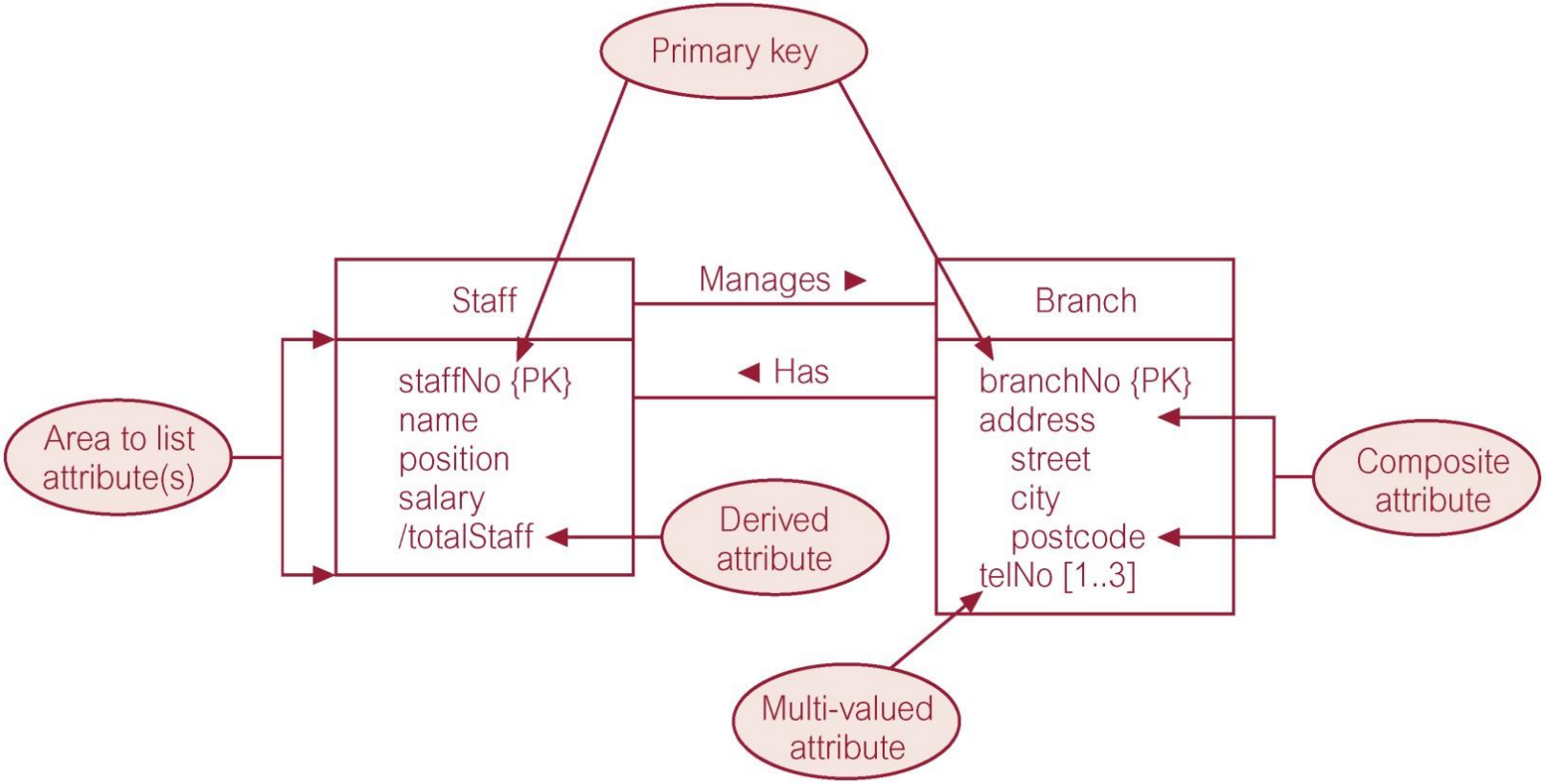
Components Of ERDs – Attributes

- Attribute is a characteristic or property of an entity, for example for a student entity, the possible attributes are registration number, student number, names, date of birth, etc.
- Attributes are represented by ovals or text in ERDs.
- Attribute Domain. Set of allowable values for one or more attributes. For example the types of values, whether numeric or alphabet, the number of digits, etc
- Attributes types include: simple attributes, compound attributes, derived attributes, single-valued attributes and multi-valued attributes.
- Simple attributes are those that can't be split into other attributes e.g. first name.

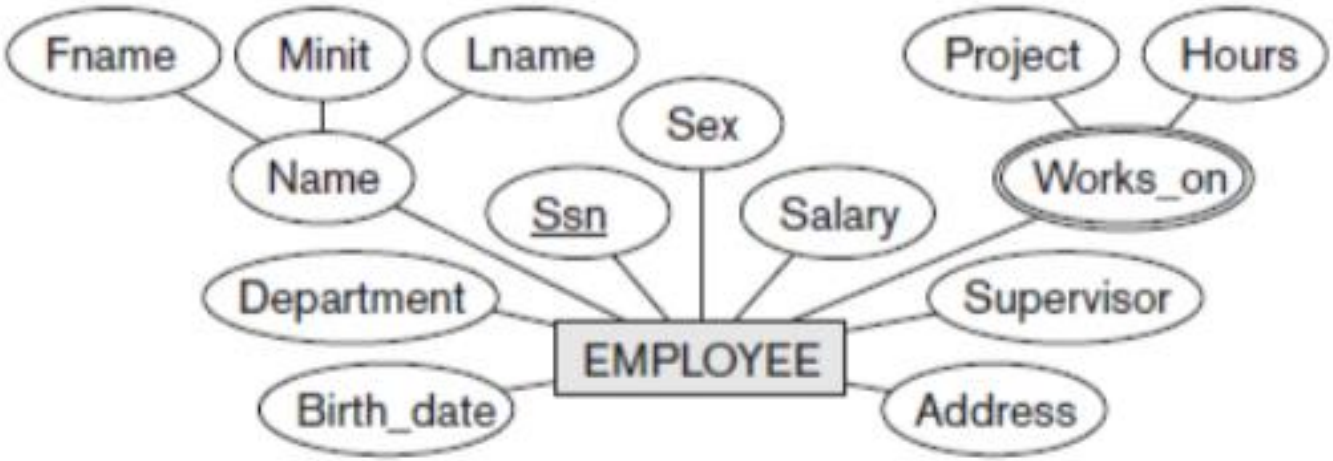
Components Of ERDs – Attributes

- Compound attributes can be split into other attributes e.g. location can be split into street address, city, country, etc.
- Single-valued attributes are those that take a single value e.g age.
- Multi-valued attributes are those that can take multi values for the same occurrence e.g. academic qualifications.
- Derived attributes are those whose value can be derived from other attributes.

Components Of ERDs – Attributes



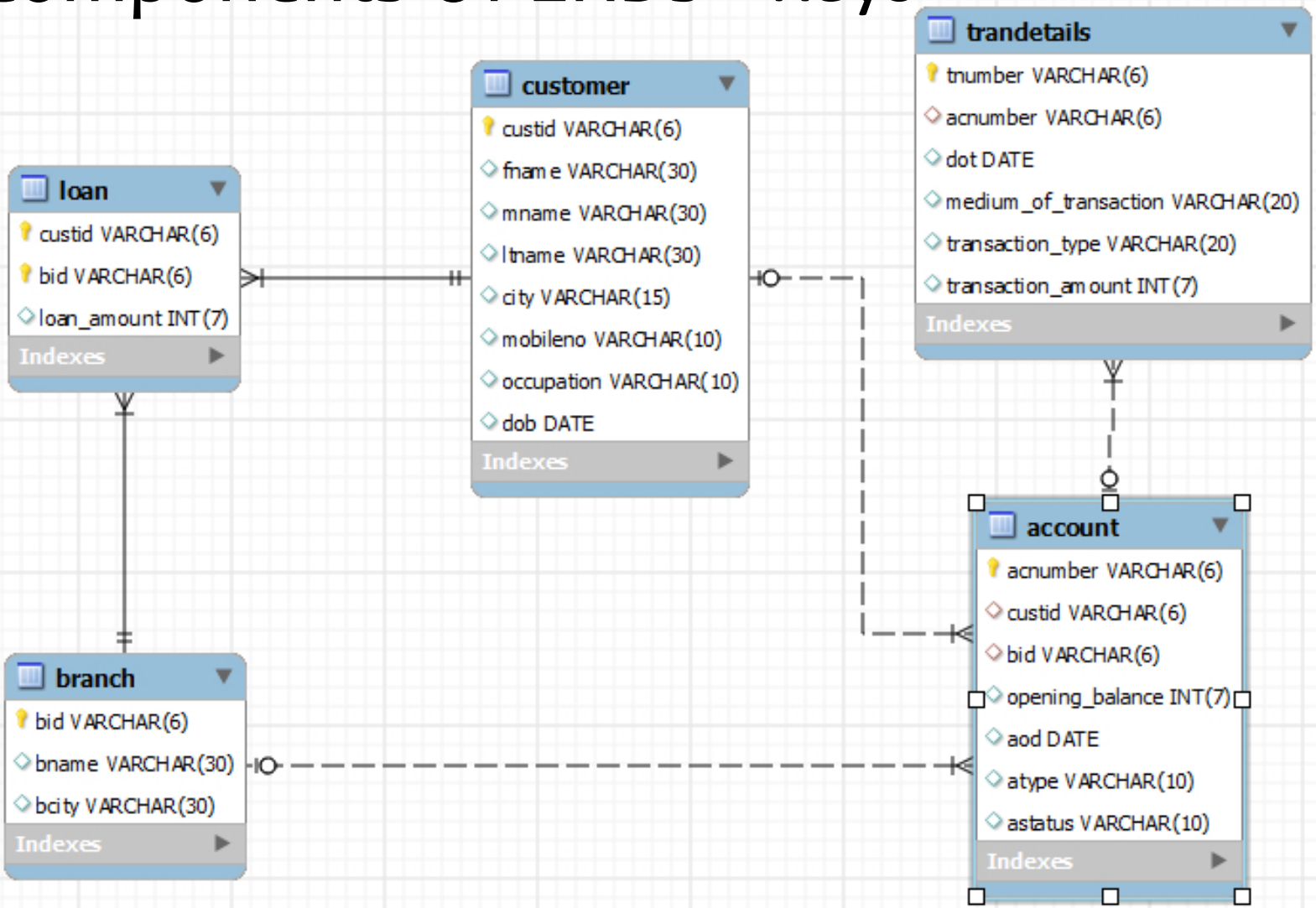
Components Of ERDs – Attributes



Components Of ERDs – Keys

- It is a special attribute or a set of attributes that uniquely identifies an entity or a relationship, such as a student ID, a course ID, or a combination of both.
- Keys are represented by underlining the attribute name in ER diagrams.
- Candidate key is a set of attributes that uniquely identify each record in the a table.
- Primary key is candidate key selected to uniquely identify each record in the table.
- Alternate key is a candidate key that is not selected to be primary key.
- Foreign key is an attribute in one table that refers to the primary key of another table. It creates a relationship between two tables, allowing us to link records in different tables.
- Composite Key is A candidate key that consists of two or more attributes.

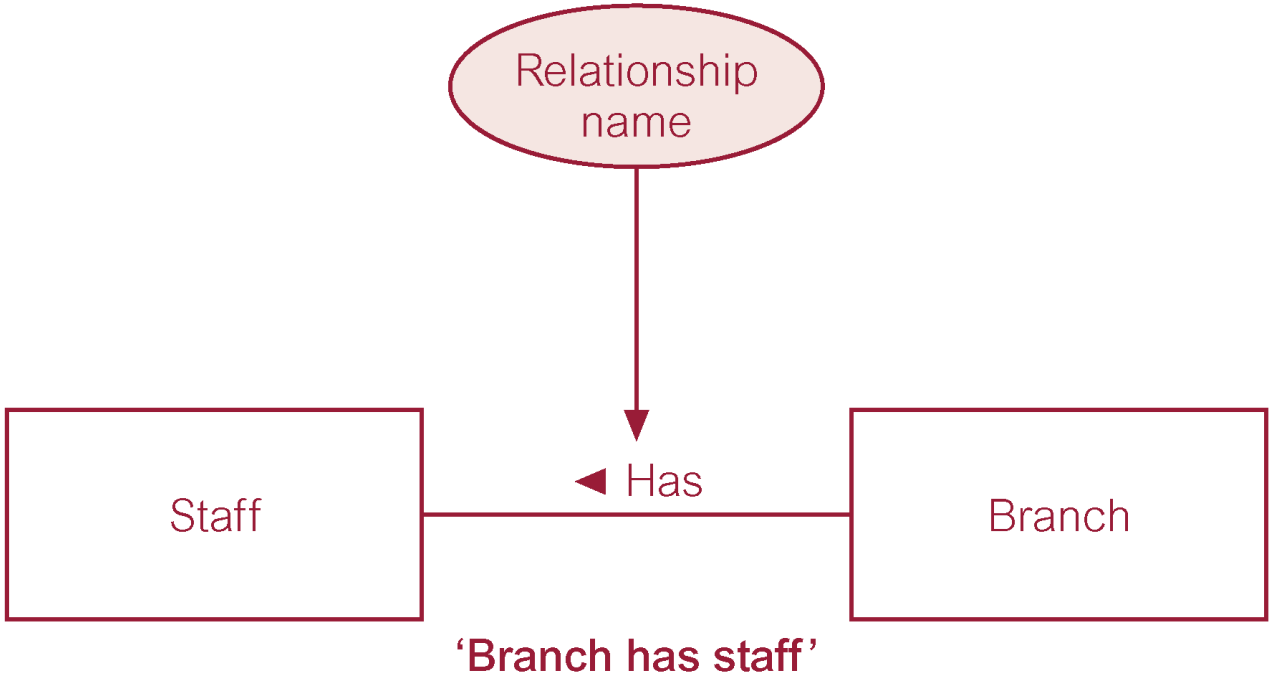
Components Of ERDs – Keys



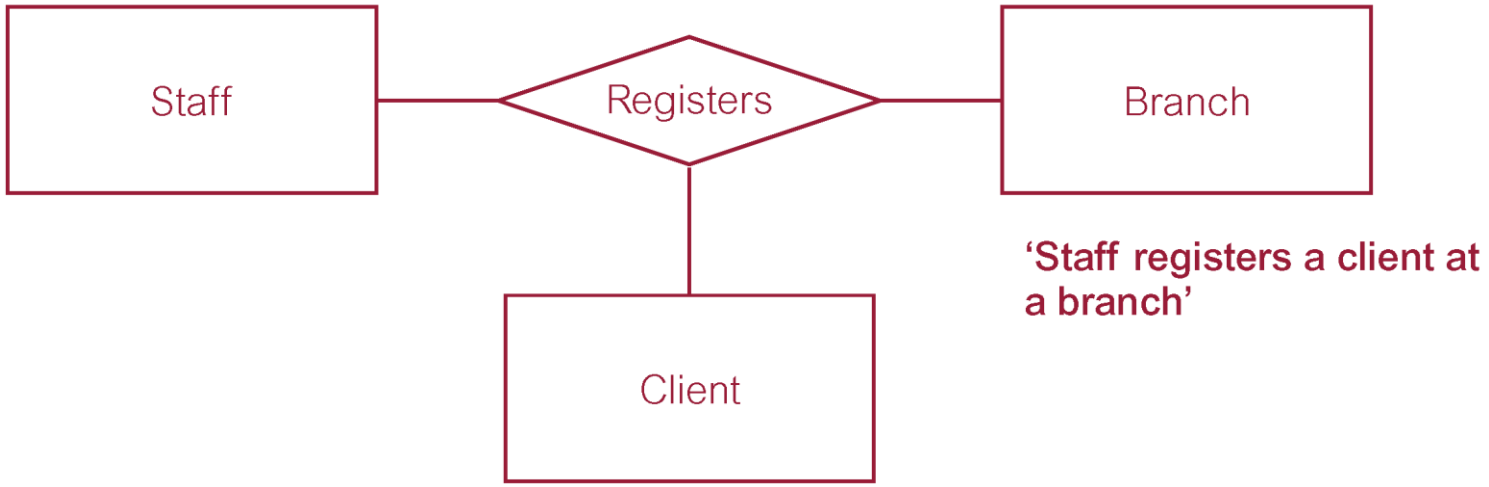
Components of ERDs - Relationship

- Derived from the fact that a database consists of related data meaning that the entities must be related according to the application being developed.
- A relationship in an ERD defines how two entities are related to each other. They can be derived from verbs when speaking about a database or a set of entities.
- Relationships in ERDs are represented as lines between two entities, and often have a label on the line to further describe the relationship or diamonds in ERDs.
- Relationship types include recursive relationship, identifying relationship, non-identifying relationship, mandatory relationship and optional relationship.

Components of ERDs - Relationship



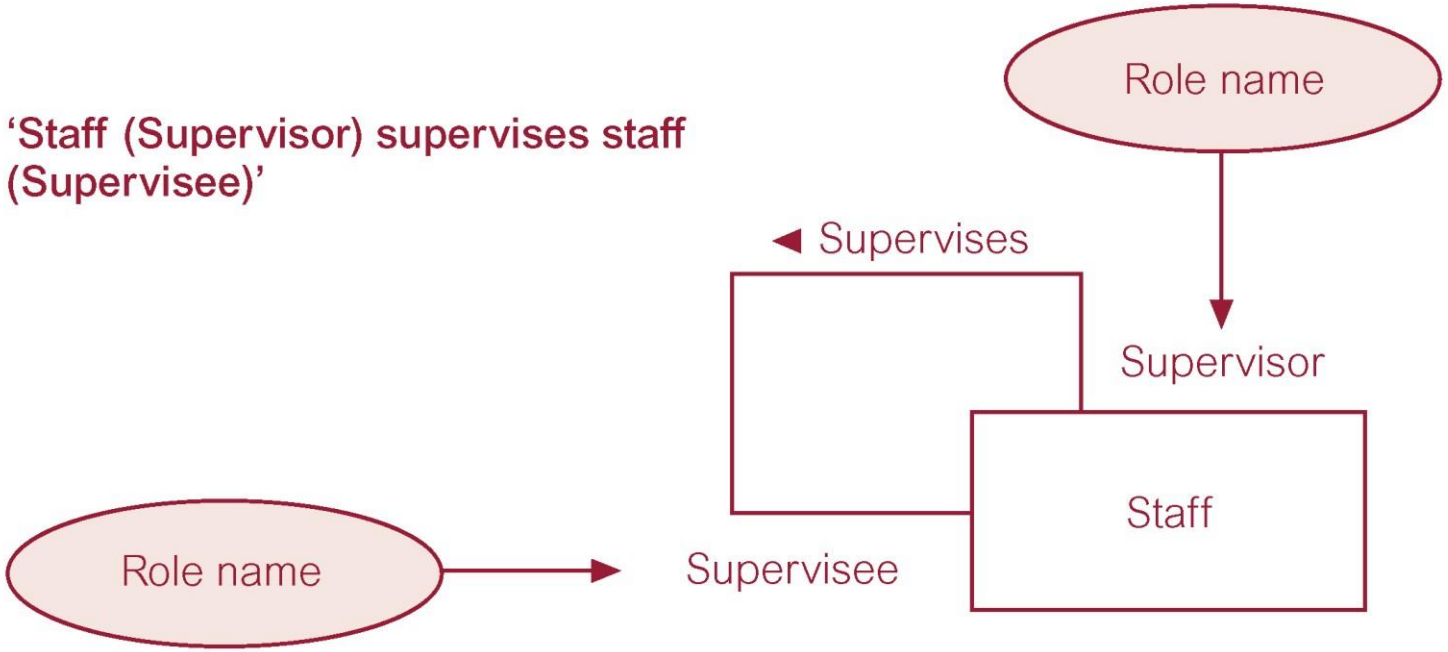
Components of ERDs - Relationship



'Staff registers a client at a branch'

Components of ERDs - Relationship Types

- ***Recursive relationship:*** A relationship where an entity is related to itself in some way. For example, an employee can be a supervisor of another employee.



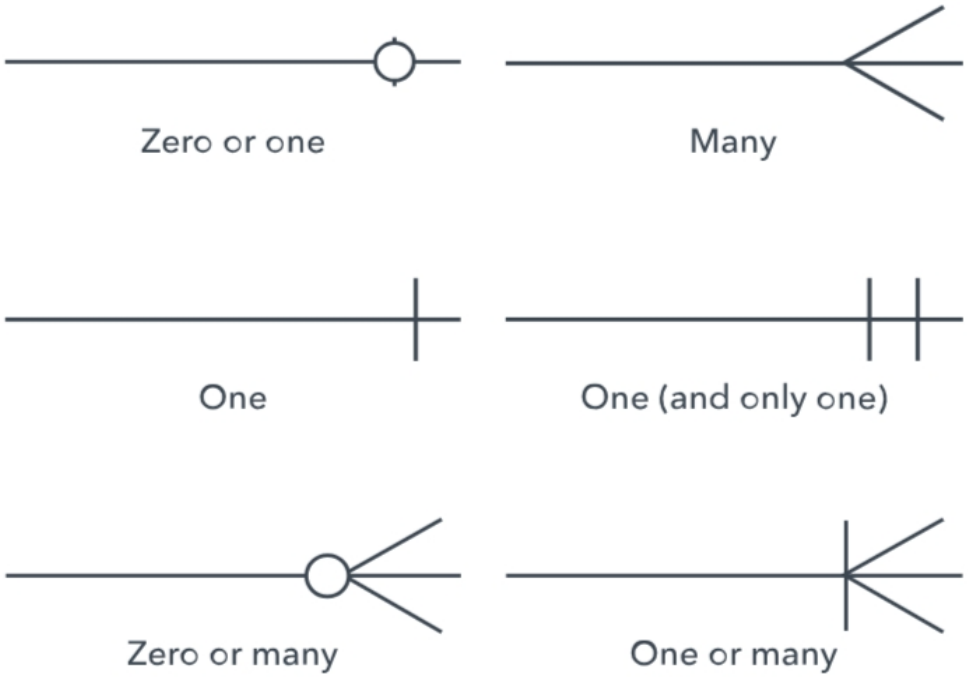
Components of ERDs - Relationship Types

- ***Identifying relationship:*** A relationship where a weak entity depends on a strong entity for its existence and identity. For example, an order line item depends on an order for its existence and identity.
- ***Non-identifying relationship:*** A relationship where an entity does not depend on another entity for its existence and identity. For example, an order does not depend on a customer for its existence and identity, even though it has a relationship with a customer.
- ***Mandatory relationship:*** A relationship where an entity must participate in the relationship with another entity. For example, a student must enrol in at least one course.
- ***Optional relationship:*** A relationship where an entity may or may not participate in the relationship with another entity. For example, a student may or may not have an address.

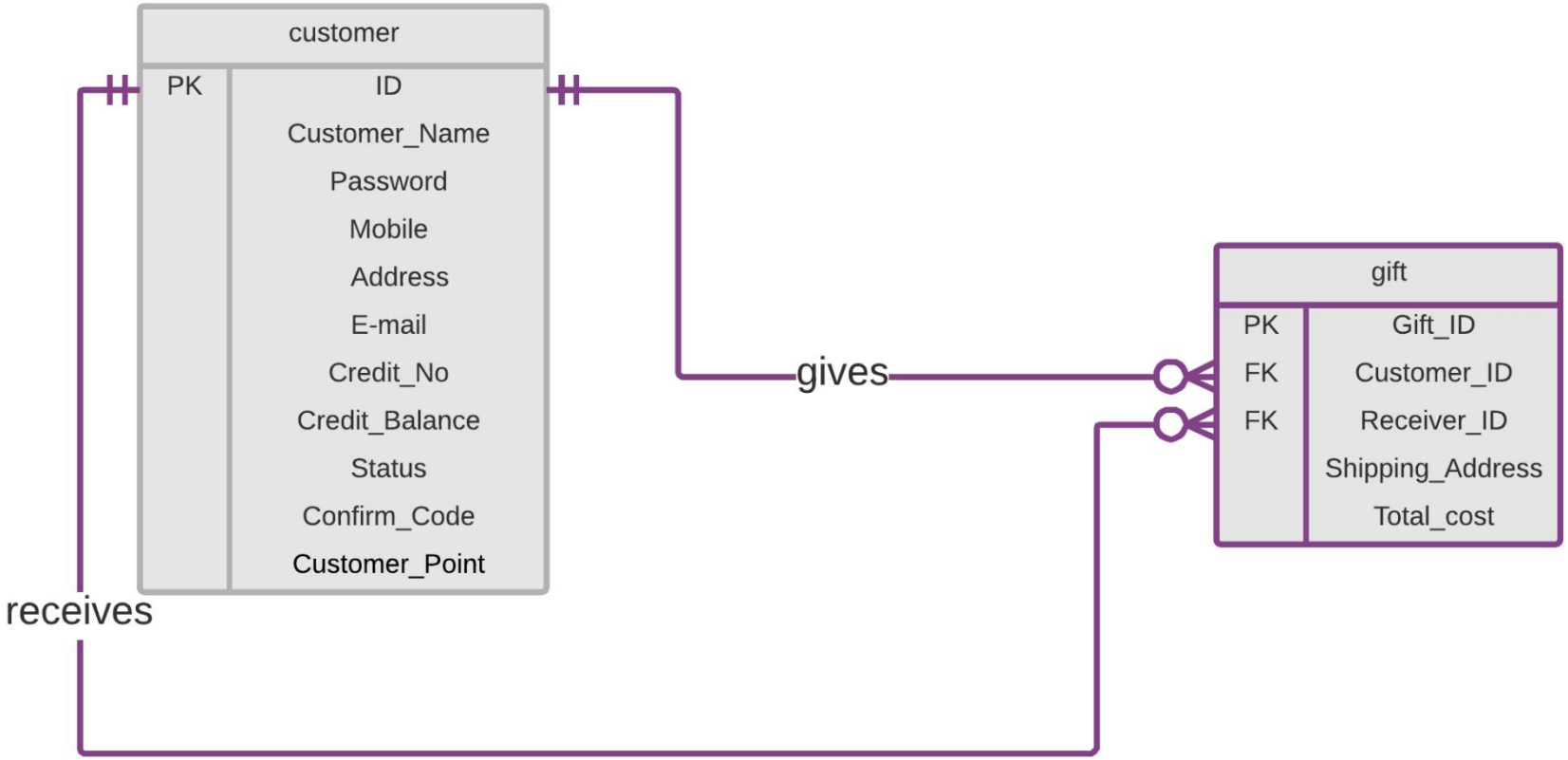
Relationships - Cardinality

- Cardinality represents the number of instances of an entity that exist in a relationship between two entities.
- ***One to one:*** One record of an entity is directly related to another record of an entity. A student has one address, and an address belongs to one student
- ***One to many:*** One record of an entity is related to one or more records of another entity. A professor teaches many courses, and a course is taught by one professor
- ***Many to many:*** Many records of one entity can be related to many records of another entity. A student enrolls in many courses, and a course has many students enrolled
- We often crow's foot notation or the Chen notation to represent the cardinality of a relationship in an ER diagram.

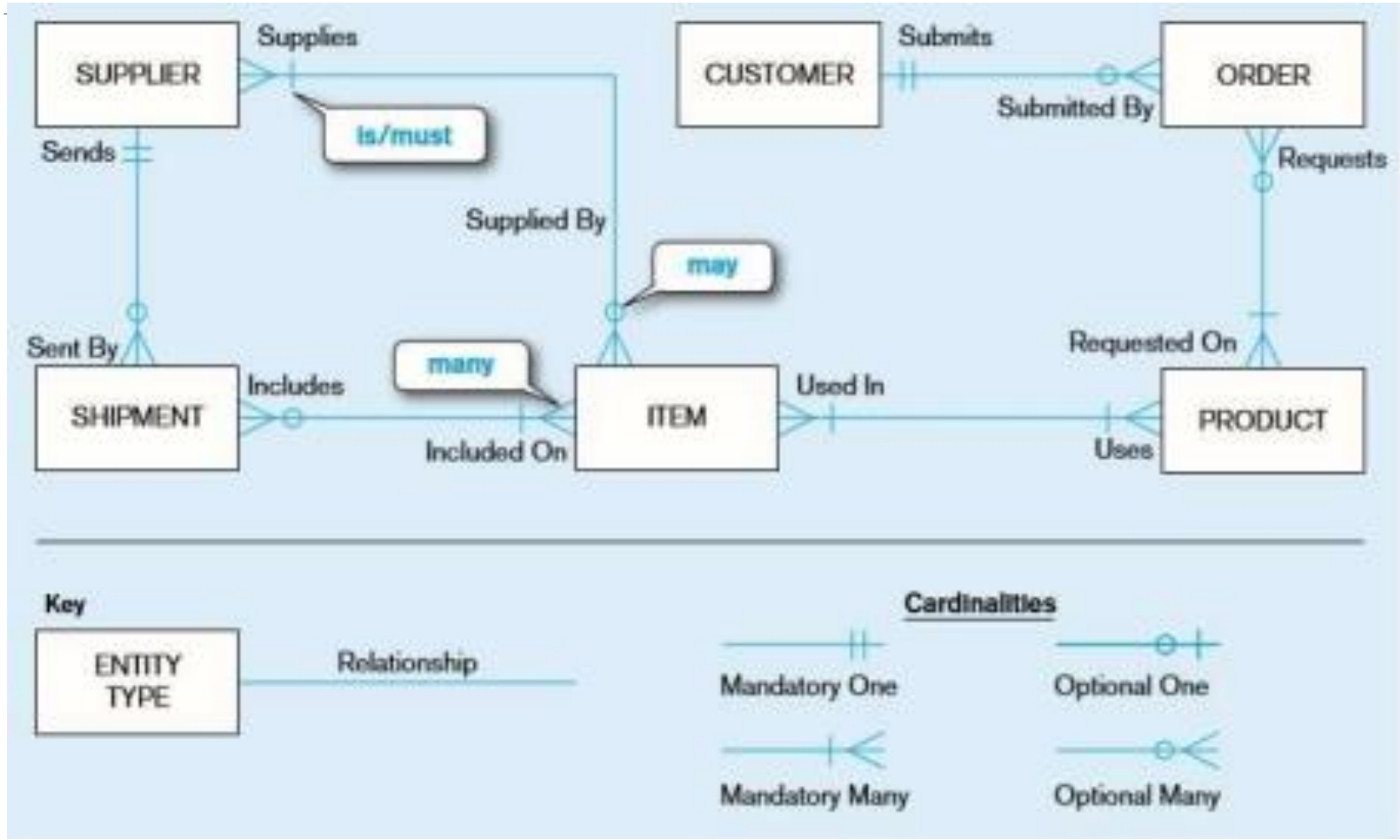
Relationship – Cardinality (Crows Foot)



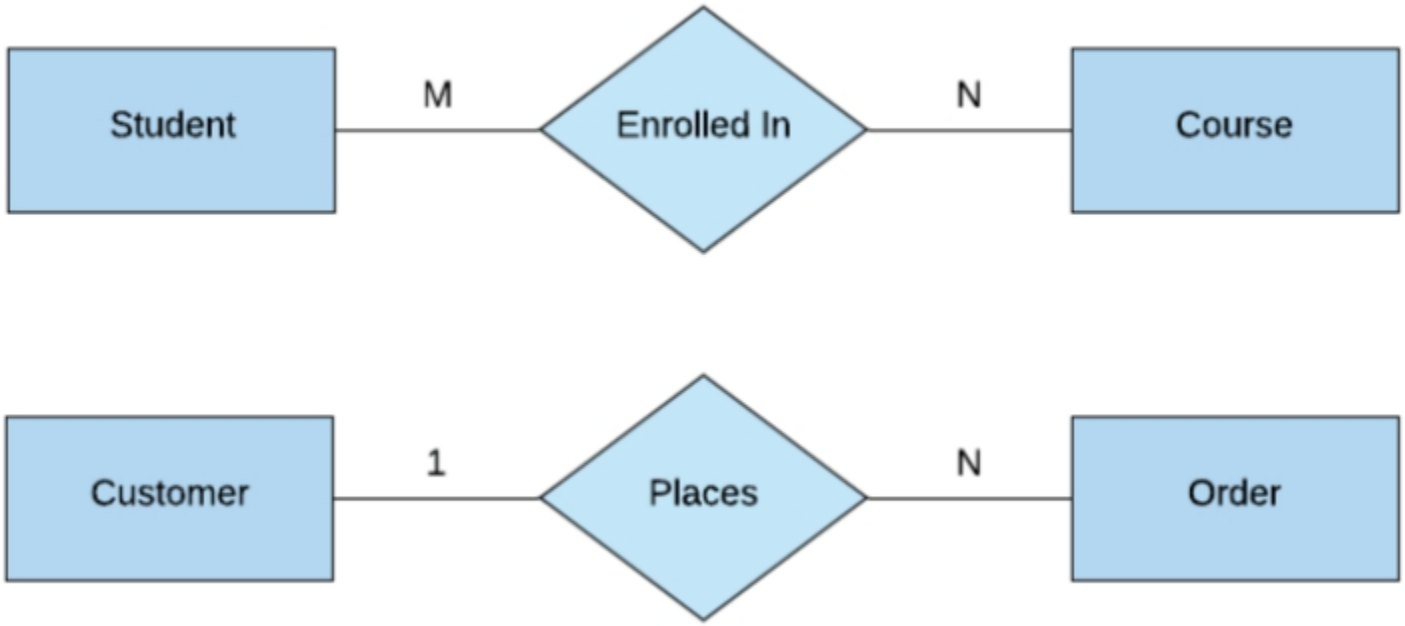
Relationship – Cardinality (Crows Foot)



Relationship – Cardinality (Crows Foot)

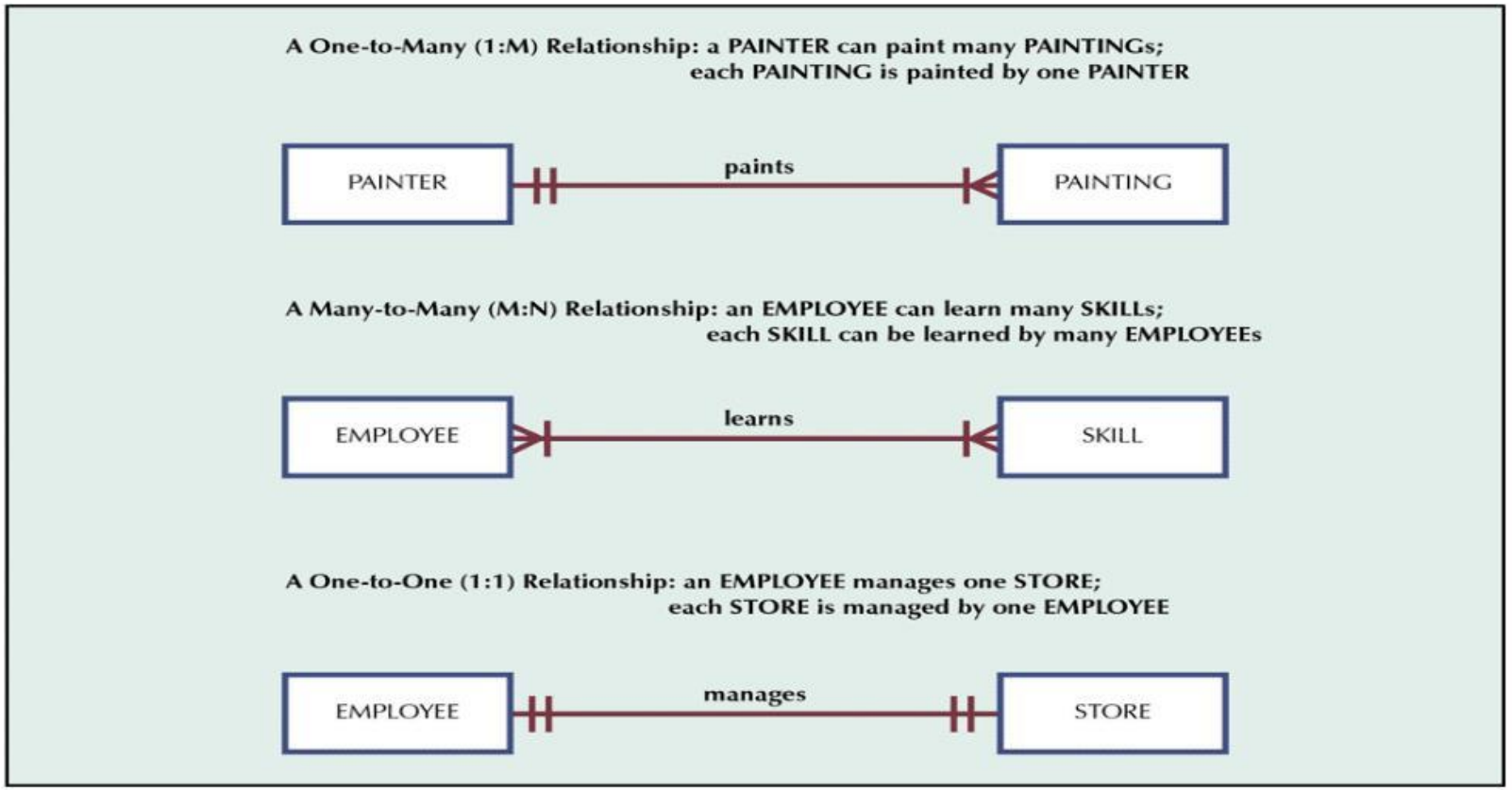


Relationship – Cardinality (Chen Notation)



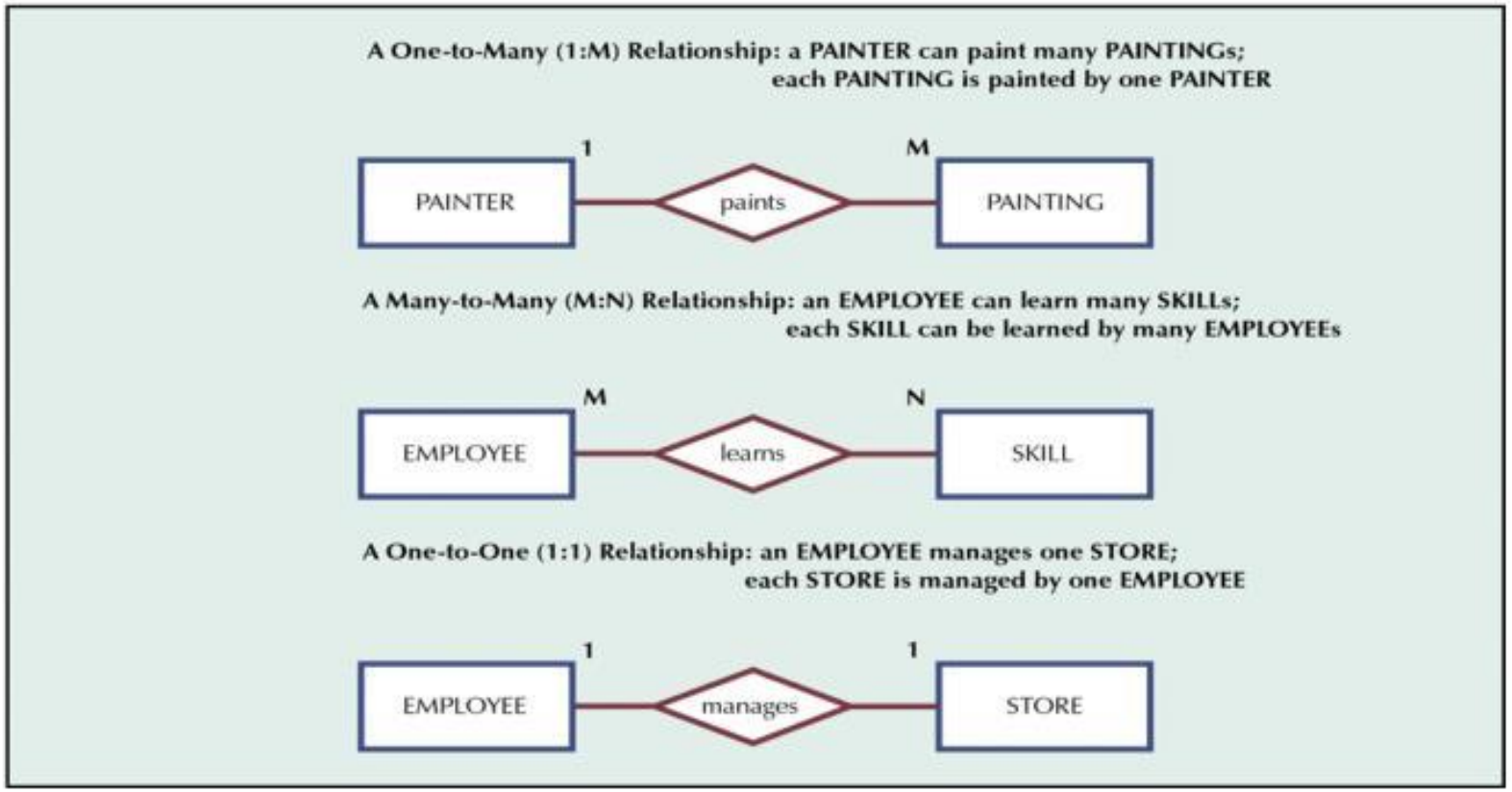
Relationships: The Basic Crow's Foot ERD

FIGURE 2.7 RELATIONSHIPS: THE BASIC CROW'S FOOT ERD



Relationships: The Basic Chen ERD

FIGURE 2.6 RELATIONSHIPS: THE BASIC CHEN ERD



Creating ERDs - Steps

- Identify Entities: Determine the main entities in the system being modeled. These are the key objects or concepts that need to be represented.
- Define Relationships: Establish the relationship between and their cardinality entities (e.g., one-to-one, one-to-many, many-to-many)
- Add Attributes: Define the characteristics or properties of each entity. These attributes provide additional details about the entities.
- Draw the Diagram: Use standard notation (such as crow's foot notation) to create the visual representation of the ERD. Draw entities as rectangles, relationships as lines connecting them, and attributes within the rectangles.
- You can use various tools to come up with an ERD.

Creating ERDs – Tools that can be Used

- Microsoft Visio
- Microsoft MySQL Workbench
- Oracle SQL Developer
- LucidChart
- Visual Paradigm
- Get Mind
- Gliffy
- Among others

Creating ERDs – Question

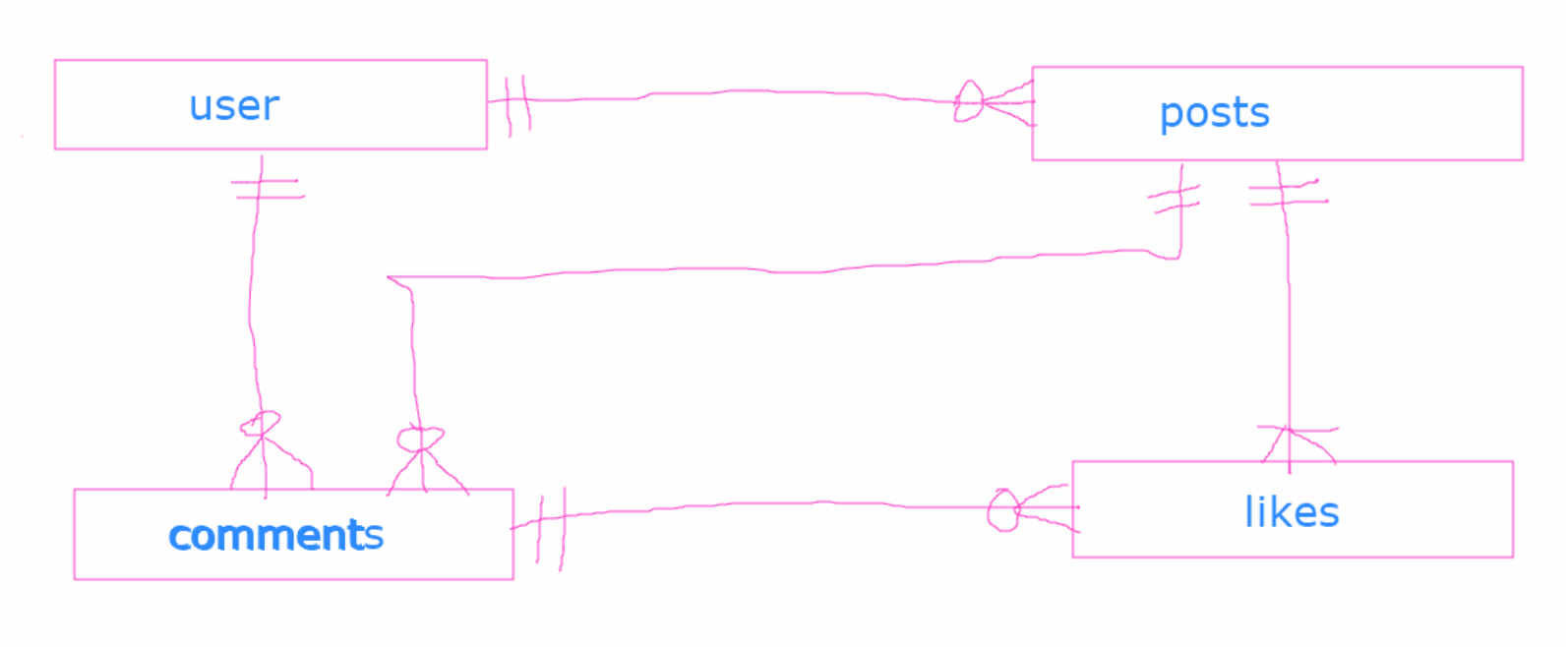
Imagine you're tasked with creating a database schema for a new social media platform. This platform allows users to interact with each other by creating posts, commenting on posts, and liking posts. Each user has a unique user ID, username, and email address. Posts are identified by a post ID and contain content and date information. Comments are associated with posts and have their own comment ID, content, and date. Likes are linked to posts and have a unique like ID along with the date.

Design the Entity-Relationship Diagram (ERD) for this social media platform, including all necessary entities, attributes, and relationships. Ensure that your ERD clearly represents how users interact with posts through comments and likes, capturing the essence of the platform's functionality.

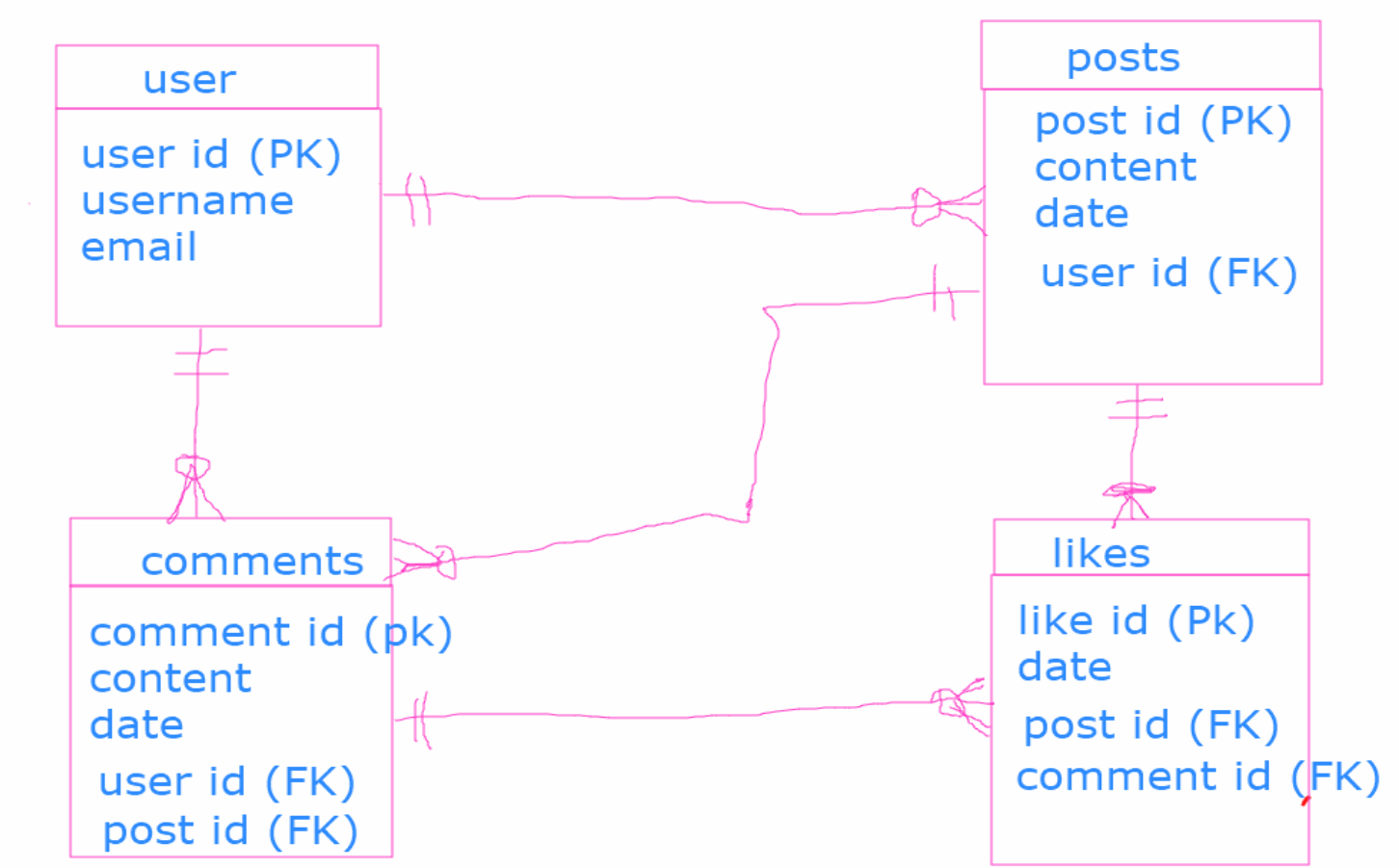
Creating ERDs – Question

- **Entities:** user, posts, comments, likes
- **Attributes:**
  - user: id, username, email
  - posts: post_id, date, content
  - comments: comment_id, date, content
  - likes: like_id, date)
- **Relationship:**
  - user creates posts - 1:M
  - post has comments - 1:M
  - post receives likes - 1:M
  - user comments on posts - 1:M
  - user likes posts - 1:M

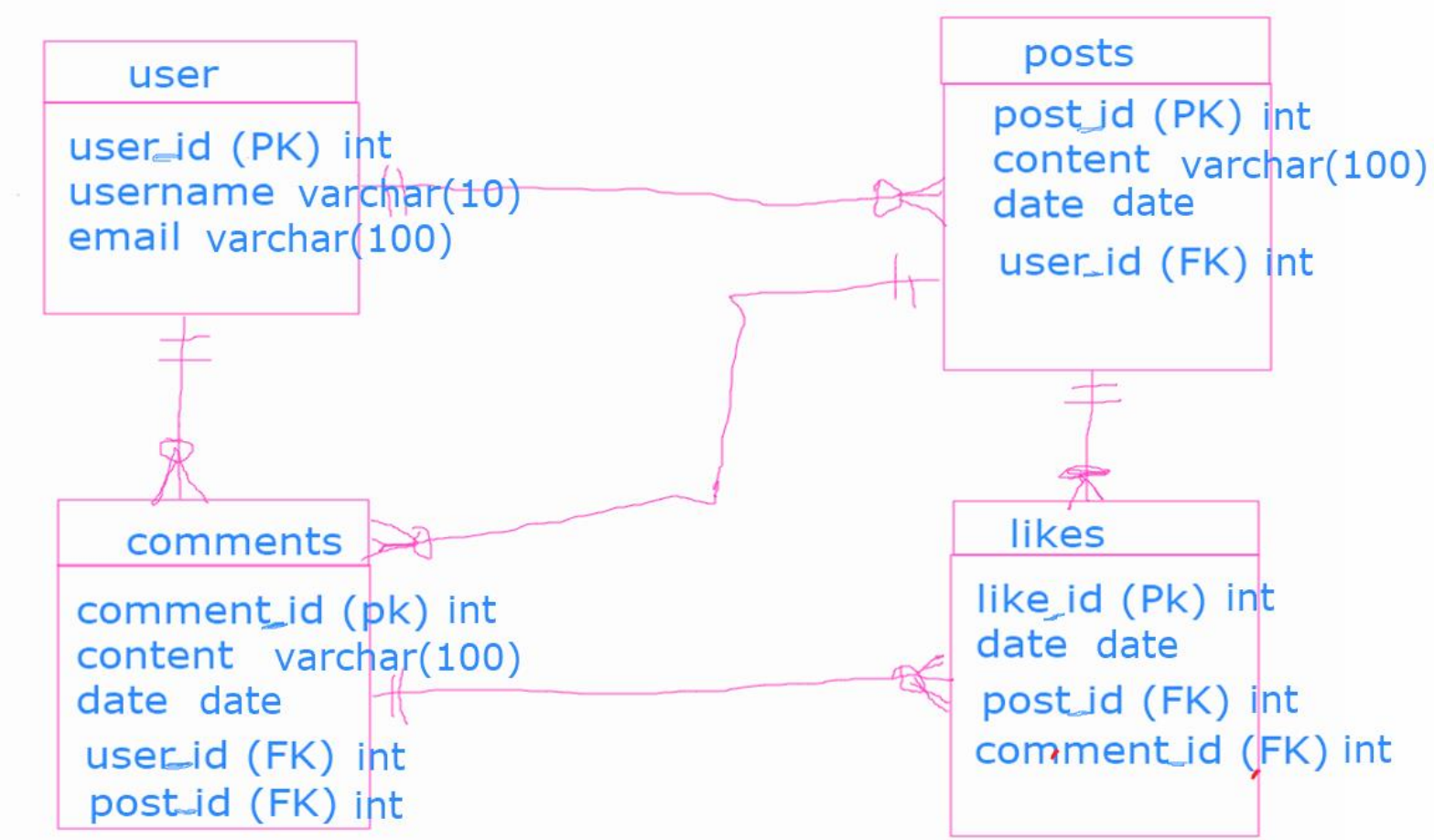
Creating ERDs – Question – Conceptual



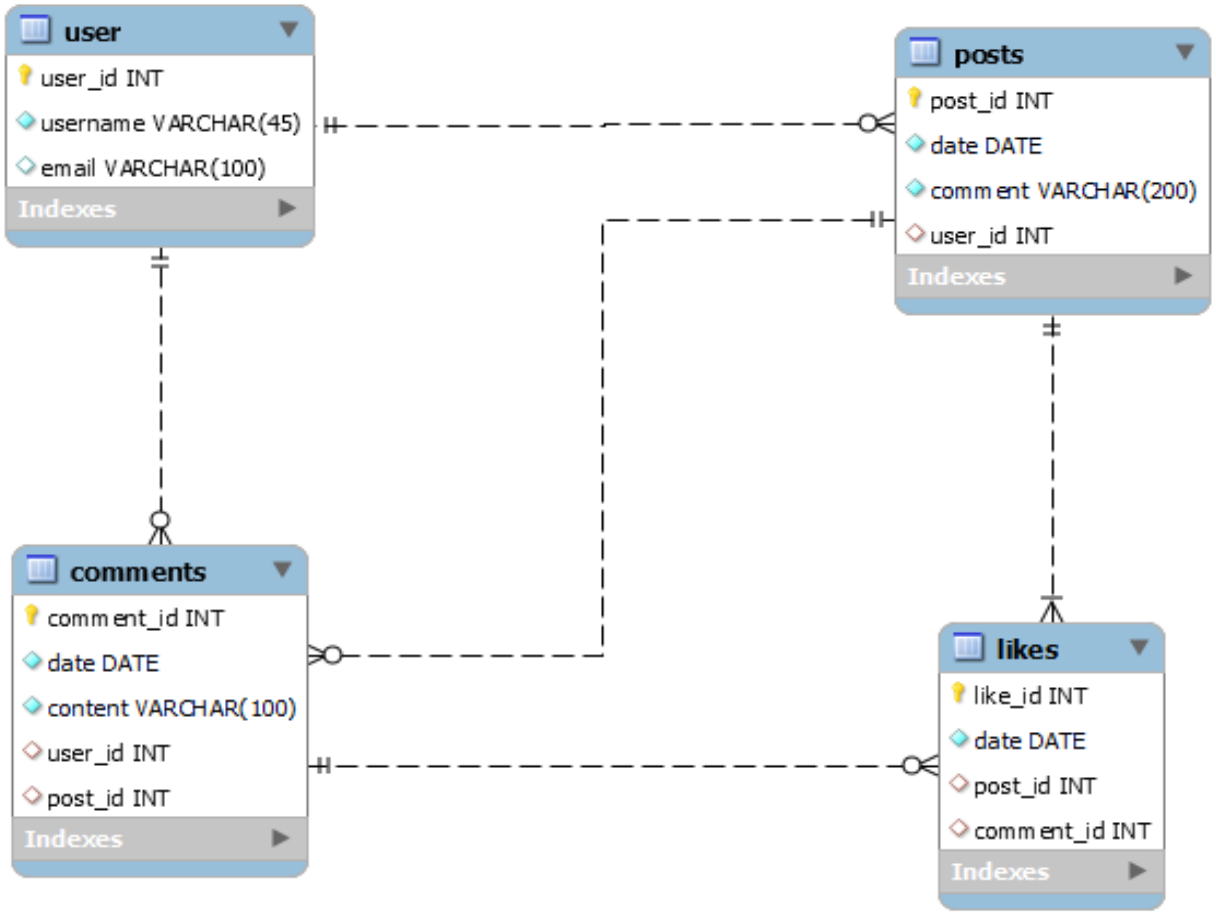
Creating ERDs – Question – Logical Model



Creating ERDs – Question – Physical Model



Creating ERDs – Question – Using a tool



Thank you!

Image was generated by Copilot AI to depict Data Modelling and ERDs