CHAPTER 5: ENTITY RELATIONSHIP

Introduction and meaning of Entity Relationship

In software engineering, an entity–relationship model (ER model) is a data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as a relational database.

Connotations of Entity Relationship

An entity–relationship model is a systematic way of describing and defining a business process. The process is modeled as components (*entities*) that are linked with each other by *relationships* that express the dependencies and requirements between them, such as: *one building may be divided into zero or more apartments, but one apartment can only be located in one building.* Entities may have various properties (*attributes*) that characterize them. Diagrams created to represent these entities, attributes, and relationships graphically are called entity–relationship diagrams.

An ER model is typically implemented as a database. In the case of a relational database, which stores data in tables, every row of each table represents one instance of an entity. Some data fields in these tables point to indexes in other tables; such pointers represent the relationships.

www.knecnotes.co.ke

The three schema approach to software engineering uses three levels of ER models that may be developed.

Conceptual data model

This is the highest level ER model in that it contains the least granular detail but establishes the overall scope of what is to be included within the model set. The conceptual ER model normally defines master reference data entities that are commonly used by the organization. Developing an enterprise-wide conceptual ER model is useful to support documenting the data architecture for an organization.
A conceptual ER model may be used as the foundation for one or more *logical data models* (see below). The purpose of the conceptual ER model is then to establish structural metadata commonality for the master data entities between the set of logical ER models. The conceptual data model may be used to form commonality relationships between ER models as a basis for data model integration.

Logical data model

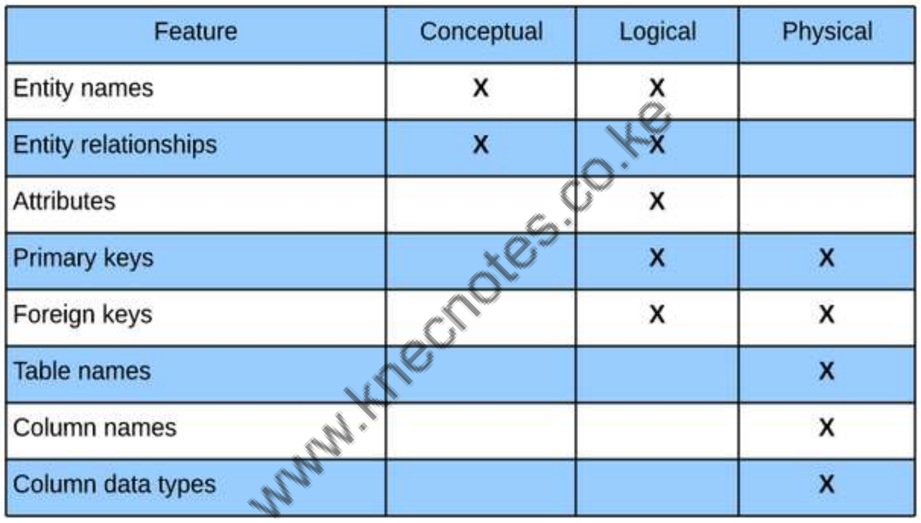
A logical ER model does not require a conceptual ER model, especially if the scope of the logical ER model includes only the development of a distinct information system. The logical ER model contains more detail than the conceptual ER model. In addition to master data entities, operational and transactional data entities are now defined. The details of each data entity are developed and the entity relationships between these data entities are established. The logical ER model is however developed independent of technology into which it is implemented.

Physical data model

One or more physical ER models may be developed from each logical ER model. The physical ER model is normally developed to be instantiated as a database. Therefore, each physical ER model must contain enough detail to produce a database and each physical ER model is technology dependent since each database management system is somewhat different. The physical model is normally instantiated in the structural metadata of a database management system as relational database objects such as database tables, database indexes such as unique key indexes, and database constraints such as a foreign key constraint or a commonality constraint. The ER model is also normally used to design modifications to the relational database objects and to maintain the structural metadata of the database.

Data Models

Before you look at specific symbols, it's important to understand the various levels of ERDs. There are several ways to model entity-relationship diagrams. The most high-level type is a conceptual data model; the next highest is the logical data model, and the lowest-level (and therefore most detailed) type is the physical data model. Consult the chart below to see which elements are covered in each data model.



Feature	Conceptual	Logical	Physical
Entity names	X	X	
Entity relationships	X	X	
Attributes		X	
Primary keys		X	X
Foreign keys		X	X
Table names			X
Column names			X
Column data types			X

www.knecnotes.co.ke

CONCEPTUAL DATA MODEL

This ER model establishes a broad view of what should be included in the model set. Conceptual data models:

- Include important entities and the relationship between them.
- Do not specify attributes.
- Do not specify primary keys.

Conceptual ERDs can be used as the foundation for logical data models. They may also be used to form commonality relationships between ER models as a basis for data model integration.

LOGICAL DATA MODEL

This model contains more detail than the conceptual ER model, without regard to how information will be physically implemented in the database. Logical data models:

- Include all entities and relationships between them.

- Specify attributes for each entity.
- Specify primary key for each entity.
- Specify foreign keys, which identify the relationship between different entities.
- Involve normalization, which is the process of removing redundancy in a table so that the table is easier to modify. Normalization typically occurs by dividing an entity table into two or more tables and defining relationships between the tables.

PHYSICAL DATA MODEL

The physical data model represents the process of adding information to the database. This model shows all table structures, including column name, column data type, column constraints, primary key, foreign key, and relationships between tables. Physical data models:

- Specify all tables and columns.
- Include foreign keys to identify relationships between tables.
- May include denormalization, depending on user requirements.
- May be significantly different from the logical data model.
- Will differ depending on which DBMS (database management system) is used.

Drawing ERD

www.knecnotes.co.ke

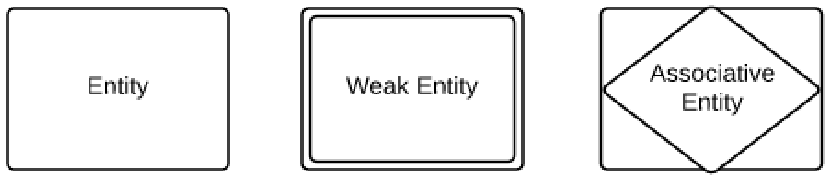
Conceptual ERD Symbols

These symbols are generally used for conceptual data models, although some aspects may spill over into logical data models. They can be found in the UML Entity Relationship and Entity Relationship shape libary of Lucidchart. If you don't see the shape you need, use an image file (Lucidchart supports .PNG, .JPG, or .SVG import) or create your own with our existing shapes and styling options.

ENTITIES

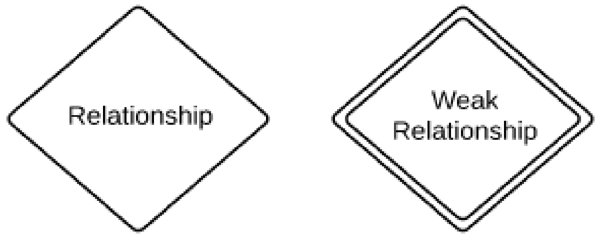
Entities are objects or concepts that represent important data. They are typically nouns, e.g. *customer*, *supervisor*, *location*, or *promotion*.

- Strong entities exist independently from other entity types. They always possess one or more attributes that uniquely distinguish each occurrence of the entity.
- Weak entities depend on some other entity type. They don't possess unique attributes (also known as a primary key) and have no meaning in the diagram without depending on another entity. This other entity is known as the owner.
- Associative entities are entities that associate the instances of one or more entity types. They also contain attributes that are unique to the relationship between those entity instances.



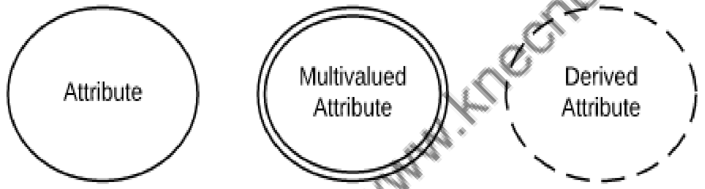
RELATIONSHIPS

- Relationships are meaningful associations between or among entities. They are usually verbs, e.g. *assign*, *associate*, or *track*. A relationship provides useful information that could not be discerned with just the entity types.
- Weak relationships, or identifying relationships, are connections that exist between a weak entity type and its owner.



ATTRIBUTES

- Attributes are characteristics of either an entity, a many-to-many relationship, or a one-to-one relationship.
- Multivalued attributes are those that are capable of taking on more than one value.
- Derived attributes are attributes whose value can be calculated from related attribute values.



www.knecnotes.co.ke

Physical ERD Symbols

The symbols below are used at the most granular level of ERDs: physical data models, although some elements are also used for logical data models.

- Tables are another way of representing entities.
- Fields represent attributes of the entity.
- Keys are one way to categorize attributes. A primary key is an attribute or combination of attributes that uniquely identifies one and only one instance of an entity. The primary key becomes a foreign key in any entity type to which it's related through a one-to-one or one-to-many relationship.
- Types may refer to the type of data associated with the corresponding field in a table. Types can also refer to entity types, which describe the structure of an entity; e.g., a book's entity types are author, title, and published date.