



E-ISSN: 2664-8784  
P-ISSN: 2664-8776  
IJRE 2025; SP-7(2): 32-36  
© 2025 IJRE  
[www.engineeringpaper.net](http://www.engineeringpaper.net)  
Received: 15-04-2025  
Accepted: 16-05-2025

**Veena Beniwal**  
Assistance Professor, M. Tech.,  
Department of CSE, DPG  
Degree College, Gurugram,  
Haryana, India

**Renuka Chauhan**  
Assistance Professor, M. Tech.,  
Department of CSE, DPG  
Degree College, Gurugram,  
Haryana, India

**Two-Days National Conference on Multidisciplinary Approaches for  
Innovation and Sustainability: Global solution for contemporary Challenges-  
NCMIS (DPG Degree College: 17<sup>th</sup>-18<sup>th</sup> 2025)**

## **Relational database design based on the entity- relationship model**

**Veena Beniwal and Renuka Chauhan**

**DOI:** <https://www.doi.org/10.33545/26648776.2025.v7.i2a.88>

### **Abstract**

The Entity-Relationship (E-R) model is often endorsed for use in the database design process because its concepts seem to be both natural and easy to use. This paper describes a methodology for the design of a relational database based on the E-R model. The procedure starts with the identification of the basic E-R constructs that occur in an application resulting in a preliminary database design. Next, ways of arresting certain semantics of an application through data abstraction and events for identifying probable design problems are discussed. It is commonly known as an ER Diagram. An ER Diagram in DBMS plays a crucial role in designing the database. Today's business world previews all the requirements demanded by the users in the form of an ER Diagram. Later, it's forwarded to the database administrators to design the database. The Entity Relational Model is a model for classifying entities to be represented in the database and representation of how those entities are interrelated. The ER data model requires enterprise schema that signifies the overall logical structure of a database graphically. In particular, the approach enables transformations of the imaginative ER model by Chen as well as prevalent extensions. We have proven the applicability of this approach in two ways: First, we have created a graphical editor capable of compliantly modeling ER diagrams and automatically transforming them to relational models. Second, we have conducted a practical study with our methodology and the created editor. Our findings show, that the transformation approach works correctly and the executed editor makes it accessible to users.

### **ER modelling is based on two perceptions.**

- Entities, defined as tables that hold explicit information (data).
- Relationships, defined as the relations or interactions between entities.

**Keywords:** Database design, entity-relationship model, relational model, database; editor

### **Introduction**

The Entity-Relationship (E-R) Model is often used as a tool for communication between a designer and an end-user of a record because of its ease of use and its suitability in representation. According to Brodie, the popularity of the E-R model for high level design is due to its economy of concepts and belief in entities and relations as natural showing concepts. This paper presents a organization for relational database design based on the E-R model. It provides both a step-by-step process and various guidelines for a good design. The procedure consists of first recognizing the main constructs of the E-R model - entities and relationships- and their associated attributes. Rules are provided for selecting primary keys from candidate keys and for capturing some of the semantics of the application through data abstractions. This results in an Entity-Relationships model of the application. Most of these approaches have two major weaknesses: First, they often impose certain constraints on the ER models and second, they remain rather theoretic. In many cases, after giving their approach, the authors recommended them to be used by practitioners or in automatic tools. Despite having a clear formalism, a practical implementation was not ever achieved, often due to the lack of operational semantics. On the other hand, there is a high numeral of editor tools for ER models, contained in drawing tools, in client software of databases, low-code platforms, or in modeling implements like enterprise architect.

**Correspondence**  
**Veena Beniwal**  
Assistance Professor, M. Tech.,  
Department of CSE, DPG  
Degree College, Gurugram,  
Haryana, India

### History of ER models

Peter Chen (a.k.a. Peter Pin-Shan Chen), presently a faculty member at Carnegie-Mellon University in Pittsburgh, is credited with developing ER modeling for database design in the 1970s. While serving as an assistant professor at MIT's Sloan School of Organization, he published a seminal paper in 1976 titled "The Entity-Relationship Model: Near a Unified View of Data."

In a broader sense, the depiction of the interconnectedness of belongings dates back to least ancient Greece, with the works of Aristotle, Socrates and Plato. It's seen more recently in the 19th and 20th Century works of philosopher-logicians like Charles Sanders Peirce and Gottlob Frege.

By the 1960s and 1970s, Charles Bachman (above) and A.P.G. Brown were working with close predecessors of Chen's approach. Bachman established a type of Data Structure Diagram, named after him as the Bachman Diagram. Brown published works on real-world systems modeling. James Martin added ERD enhancements. The work of Chen, Bachman, Brown, Martin and others also contributed to the development of Integrated Modeling Language (UML), widely used in software design.

### Database Design

Database design can be defined as the process of capturing relevant information and processing desires of an enterprise and mapping them onto an underlying database management system. The database design process can be divided into four phases.

1. Requirements specification.
2. Conceptual design.
3. Logical design; and.
4. Physical design.

#### 1.1. Requirements specification

The first phase of the database process is the requirements specification phase through which an analysis is made of the information needs within an organization resulting preliminary specification of the information requirements of several user groups.

#### 1.2. Conceptual design

The conceptual design phase models and represents the users' and applications' views of information and, possibly, a description of the processing or use of the information. The Objective of this phase is to produce a high-level representation of the requirements independent of the Database Management System which will be used. This high-level representation is called a conceptual schema or conceptual design. It is regularly expressed as an Entity-Relationship (E-R) model or a semantic data model.

### 1.3. Logical design

During the logical design phase, a logical design (or schema) that corresponds to the data model of the selected DBMS is produced; for example, a relational data model. This step implementation design because it represents the revolution of the conceptual schema into the logical schema of the DBMS.

### 1.4. Physical design

Physical database design transforms the logical design into a form that is suitable for the given hardware and database management system. It plans the logical schema into a suitable stored representation, and determines the physical parameters necessary to optimize the database performance against a set of required transactions.

### 1.5. Scope

This paper provides a step-by-step methodology for the development of a conceptual model, communicated as an Entity-Relationship model, and its transformation into the logical design of a relational database management system.

### Why Use ER Diagrams In DBMS




- ER diagrams represent the E-R model in a database, creating them easy to convert into relations (tables).
- ER diagrams serve the purpose of real-world modeling of objects which makes them attentively useful.
- ER diagrams require no technical knowledge of the fundamental DBMS used.
- It gives a standard solution for visualizing the data logically.



### Symbols Used in ER Model

ER Model is used to model the logical view of the system from a data perspective which consists of these symbols.

- **Rectangles:** Rectangles represent entities in the ER Model.
- **Ellipses:** Ellipses represent attributes in the ER Model.
- **Diamond:** Diamonds represent relationships among Entities.
- **Lines:** Lines represent attributes to entities and entity sets with other relationship types.
- **Double Ellipse:** Double ellipses represent multi-valued Attributes.
- **Double Rectangle:** Double rectangle represents a weak entity.

**Table 1:** Symbols used in ER Model

Figure	Symbol	Represents
Rectangle		Entities in E-R Models
Ellipse		Attribute in E-R Models
Diamond		Relationship among Entities

Line	.	Attribute to Entities set and entities and Relationship
Double Ellipse		Multi-Valued Attribute
Double Rectangle		Weak-Entity

### Entity, Entity Set and Entity Type

An entity is a piece in the real world with an independent existence that can be differentiated from other objects. An entity might be

- An object with physical existence (e.g., a lecturer, a student, a car).
- An object with theoretical existence (e.g., a course, a job, a position).

Entities can be classified based on their strength. An entity is considered weak if its tables are existence dependent.

- That is, it cannot occur without a relationship with another entity.
- Its primary key is derived from the primary key of the parent entity.
- The Spouse table, in the COMPANY database, is a

weak entity because its primary key is needy on the Employee table. Without a corresponding employee record, the spouse record would not exist.

An entity is considered strong if it can exist apart from all of its related entities.

- Kernels are strong entities.
- A table without a foreign key or a table that contains a foreign key that can contain nulls is a strong entity.

Another term to know is entity type which defines a collection of similar entities.

An entity set is a collection of entities of an entity type at a particular point of time. In an entity relationship diagram (ERD), an entity type is represented by a name in a box. For example, in Figure 8.1, the entity type is employee.

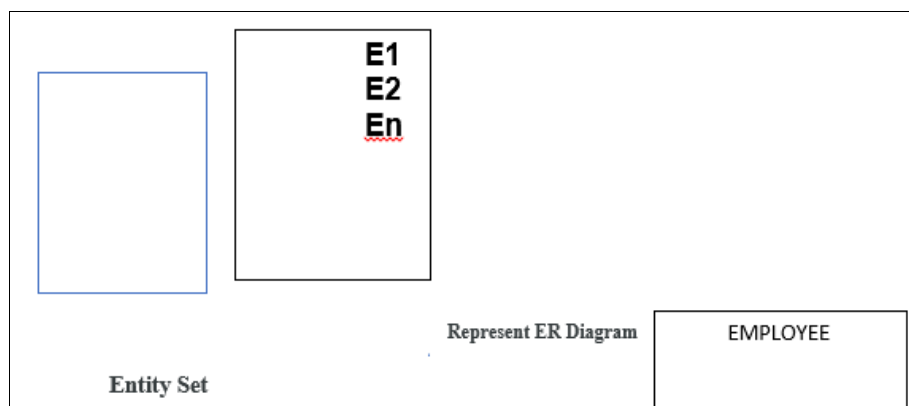


Fig 1: ERD with entity type Employee

### Existence dependency

An entity's existence is dependent on the existence of the related entity. It is existence-dependent if it has a mandatory foreign key (i.e., a foreign key attribute that cannot be null). For example, in the COMPANY file, a Spouse entity is existence -dependent on the Employee entity.

### Kinds of Entities

You should also be aware with different kinds of entities including independent entities, dependent entities and characteristic entities. These are described below.

#### Independent entities

Independent entities, also referred to as kernels, are the strength of the database. They are what other tables are based on. Kernels have the following characteristics.

- They are the structure blocks of a database.
- The primary key may be simple or composite.
- The primary key is not a foreign key.
- They do not depend on another entity for their

existence.

If we refer back to our COMPANY database, examples of an independent entity include the Customer table, Employee table or Product table.

### Dependent entities

Dependent entities, also mentioned to as derived entities, depend on other tables for their meaning. These entities have the following characteristics.

- Dependent entities are used to connect two kernels organized.
  - They are said to be existence dependent on two or more tables.
  - Many too many relationships become associative tables with at smallest two foreign keys.
  - They may contain other attributes.
  - The foreign key identifies each associated table.
  - There are three options for the primary key.
1. Use a composite of foreign keys of associated tables if

- unique.
- 2. Use a composite of foreign keys and a qualifying column.
- 3. Create a new simple primary key.

### Characteristic entities

Characteristic entities provide more evidence about another table. These entities have the following characteristics:

- They represent multivalued attributes.
- They describe other entities.
- They typically have a one to many relationship.
- The foreign key is used to add identify the characterized table.
- Options for primary key are as follows.
- 1. Use a composite of foreign key plus a succeeding column
- 2. Create a new simple primary key. In the COMPANY database, these might include.
- Employee (EID, Name, Address, Age, Salary) - EID is the simple primary key.
- EmployeePhone (EID, Phone) - EID is part of a composite primary key. Here, EID is also a foreign key.

### Attributes in the ER Model

In the ER model, an attribute is a characteristic or property of an entity that describes some phase of the entity. For example, in a database of employees, an attribute of the "Employee" entity might be "name," "email," or "salary." There are numerous types of attributes, including?

- **Simple attribute?** An attribute that has a single value for a given entity or relationship. For sample, a person entity might have a simple attribute called "name".
- **Composite attribute?** An attribute that is made up of multiple simple attributes. For example, a person entity might have a composite attribute called "address" that is complete up of simple attributes such as "street", "city", "state", and "zip code".
- **Single-valued attribute?** An attribute that can only have one value. For example, a person entity might have a single-valued attribute called "gender" that can only have the values "male" or "female".
- **Multi-valued attribute?** An attribute that can have multiple values. For example, a person entity might have a multi-valued attribute called "hobbies" that can have multiple standards such as "reading", "running", and "cooking".
- **Derived attribute?** An attribute that is derived from other attributes or entities. For example, a person entity might have a derived attribute called "age" that is calculated from the person's date of birth.
- **Null attribute?** An attribute that has no value. This can occur when an attribute is elective and not all entities have a value for that attribute. For example, a person entity might have a null attribute called "middle name" if not all people have a middle name.

### Relationships in the ER Model

In the ER model, a relationship is a connection between two or more entities. For example, in a database of employees, there might be a relationship between the "Employee" entity and the "Department" entity, representing the statistic that each employee belongs to a department.

There are three types of relationships in the ER model: one-to-one, one-to-many, and many-to-many.

- A one-to-one relationship is a relationship between two entities where each entity can be associated to at most one instance of the other entity. For example, in a database of employees, there might be a one-to-one relationship between the "Employee" entity and the "Employee Contact Information" entity, as each employee can have only one set of connection information.
- A one-to-many relationship is a relationship between two entities where an instance of the first entity can be related to multiple instances of the second entity, but an instance of the second object can be related to only one instance of the first entity. For example, in a database of employees, there is a one-to-many relationship between the "Employee" entity and the "Project" entity, as an employee can work on multiple projects, but a project can have only one lead employee.
- A many-to-many relationship is a relationship between two entities where an instance of the first entity can be related to multiple instances of the second entity, and vice versa. For example, in a database of employees, there might be a many-to-many relationship between the "Employee" entity and the "Skill" entity, as an employee can have multiple skills, and a skill can be controlled by multiple employees.

### How to create an Entity - Relationship Diagram

ERDs are generally represented in one or more of these models.

- A conceptual data model, which lacks specific detail but provides an indication of the scope of the project and how data sets relate to one another.
- A logical data model, which is more detailed than a conceptual data model, showing specific attributes and relationships among data points. While a conceptual data model does not need to be designed before a logical data model, a physical data model is based on a logical data model.
- A physical data model, which offers the blueprint for a physical manifestation -- such as a relational database -- of the logical data model. One or more physical data models can be established based on a logical data model.

There are five elementary components of an entity relationship diagram. Similar components will be designated by the same shape. For example, all entity types potency be enclosed in a rectangle, while all attributes are enclosed in a diamond. The components include the following:

1. Entities, which are objects or conceptions that can have data stored about them. Entities refer to tables used in databases.
2. Attributes, which are properties or characteristics of entities. An ERD attribute can be represented as a primary key, which identifies a unique attribute, or a foreign key, which can be assigned to several attributes.
3. The relationships between and among those entities.
4. Actions, which define how entities share information in the database.
5. Connecting lines.

### Limitations of ER diagrams and models

- Only for relational data: Understand that the persistence



is to show relationships. ER diagrams show only that relational structure.

- Not for unstructured data: Unless the data is cleanly delineated into different fields, rows or columns, ER diagrams are possibly of limited use. The same is true of semi-structured data, because only some of the data will be useful.
- Difficulty assimilating with an existing database: Using ER Models to integrate with an existing database can be a challenge because of the dissimilar architectures.

### Conclusion

To sum up, attributes and relationships are key components of Entity-Relationship (ER) modelling, which is used to design and characterize the data structures of a database. Attributes are characteristics or properties of an entity, relationship, or another attribute, and can be simple or merged, single-valued or multi-valued, derived or null. Relationships are connections or associations between entities and can be one-to-one, one-to-many, or many-to-many.

Together, attributes and relationships form a thorough and accurate representation of the data in a system, making ER modeling a useful tool for database design and management.

### References

1. Batlm C, Lenzen M, Navathe SB. A comparative analysis of methodologies for database schema integration. *ACM Comput Surv.* 1986;18(2):323-364.
2. Blaha MR, Premerlani WJ, Rumbaugh JE. Relational database design using an object-oriented methodology. *Commun ACM.* 1988;31(4):414-427.
3. Potter WD, Kerschberg L. A unified approach to modeling knowledge and data. In: Meersman RA, Sernadas AC, editors. *Data and Knowledge (DS-2)*. Amsterdam: North-Holland; 1988. p. 265-291.
4. Shaw M. The impact of modeling and abstraction concerns on modern programming languages. In: Brodie ML, Mylopoulos J, Schmidt JW, editors. *On Conceptual Modeling*. Berlin: Springer; 1984. p. 19-47.
5. Smith JM, Smith CP. Database abstractions: aggregation and generalization. *ACM Trans Database Syst.* 1977;2(2):105-133.
6. Storey VC, Goldstein RC. Design and development of an expert database design system. *Int J Man Mach Stud.*
7. Castellanos M. Semantic enrichment of interoperable databases. In: *Proc IEEE RIDE-IMS*. 1993.
8. Chen P. The Entity-Relationship model—towards a unified view of data. *ACM Trans Database Syst.* 1976;1(1):9-36.
9. Connolly T, Begg C, Strachan A. *Database systems*. 1st ed. Boston: Addison-Wesley; 1999.
10. Davis F. *Requirements specification: objects, functions, and states*. Englewood Cliffs (NJ): Prentice-Hall; 1993.
11. Dos Santos CS, Neuhold EJ, Furtado AL. A data type approach to the Entity-Relationship model. In: *Proc 1st Int Conf on Entity-Relationship Approach to Software Engineering*. 1980.
12. Kim W, Choi I, Gala S, Scheevel M. On resolving schematic heterogeneity in multidatabase systems. *Distrib Parallel Databases*. 1993.
13. Larson J, Navathe S, Elmasri N. A theory of attribute equivalence in databases with application to schema integration. *IEEE Trans Softw Eng.* 1989;15(4):449-463.
14. Miller R. Using schematically heterogeneous structures. In: *Proc ACM SIGMOD Int Conf*. 1998.
15. Melton J, Simon AR. *SQL:1999—Understanding relational language components*. San Francisco: Morgan Kaufmann; 2002.
16. Goh C, Bressan S, Madnick S, Siegel M. Context mediation: new features and formalisms for the intelligent integration of information. *Sloan Working Paper* 3941. 1997.
17. Pressman R. *Software engineering*. New York: McGraw-Hill; 1997.
18. Rumbaugh J, Jacobson I, Booch G. *The unified modeling language reference manual*. Boston: Addison-Wesley; 1999.
19. Song WW, Johannesson P, Bubenko JA. Semantic similarity relations in schema integration. In: *Proc 11th Int Conf on the Entity-Relationship Approach*. 1992.
20. Li WS, Clifton C. Semantic integration in heterogeneous databases using neural networks. In: *Proc 20th VLDB Conf*. 1994.
21. Sciore E, Siegel M, Rosenthal A. Using semantic values to facilitate interoperability among heterogeneous information systems. *ACM Trans Database Syst.* 1994;19(2):254-290.
22. Thalheim B. *Entity-relationship modeling: foundations of database technology*. Berlin: Springer; 2000.
23. Pressman R. *Software engineering*. New York: McGraw-Hill; 1997.
24. Rumbaugh J, Jacobson I, Booch G. *The unified modeling language reference manual*. Boston: Addison-Wesley; 1999.
25. Song WW, Johannesson P, Bubenko JA. Semantic similarity relations in schema integration. In: *Proc 11th Int Conf on the Entity-Relationship Approach*. 1992.
26. Li WS, Clifton C. Semantic integration in heterogeneous databases using neural networks. In: *Proc 20th VLDB Conf*. 1994.