**ENTITY RELATIONSHIP DIAGRAM (ERD)**

* ERD is a graphical tool for modeling data.
* ERD is widely used in data base design
* ERD is a graphical representation of the logical structure of a database
* ERD is a model that identifies the concepts or entities that exist in a system and the relationships between those entities.

Purposes of ERD:

An ERD serves several purposes:

* The database designer gains a better understanding of the information to be contained in the database through the process of constructing the ERD

**FUNCTIONAL DESIGN**

Functional Design is a paradigm used to simplify the design of hardware and software devices such as computer [software](https://en.wikipedia.org/wiki/Software) and increasingly, [3D models](https://en.wikipedia.org/wiki/3D_model). A [functional](https://en.wikipedia.org/wiki/Function_(engineering)) design assures that each modular part of a device has only one responsibility and performs that responsibility with the minimum of side effects on other parts. Functionally designed modules tend to have low [coupling](https://en.wikipedia.org/wiki/Coupling_(computer_science)). [1]

**Advantages:**

* The advantage for implementation is that if a software module has a single purpose, it will be simpler, and therefore easier and less expensive, to design and implement.
* Systems with functionally designed parts are easier to modify because each part does only what it claims to do.
* Since maintenance is more than 3/4 of a successful system's life, [1] this feature is a crucial advantage.
* It also makes the system easier to understand and document, which simplifies training. The result is that the practical lifetime of a functional system is longer.
* In a system of programs, a functional module will be easier to reuse because it is less likely to have side effects that appear in other parts of the system.

**DATABASE DESIGN**

Database design is the process of producing a detailed [data model](https://en.wikipedia.org/wiki/Data_model) of a [database](https://en.wikipedia.org/wiki/Database). This [logical data model](https://en.wikipedia.org/wiki/Logical_data_model) contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a [data definition language](https://en.wikipedia.org/wiki/Data_definition_language), which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity.

The term database design can be used to describe many different parts of the design of an overall [database system](https://en.wikipedia.org/wiki/Database_system). Principally, and most correctly, it can be thought of as the logical design of the base data structures used to store the data. In the [relational model](https://en.wikipedia.org/wiki/Relational_model) these are the [tables](https://en.wikipedia.org/wiki/Database_table) and [view](https://en.wikipedia.org/wiki/Database_view). In an [object database](https://en.wikipedia.org/wiki/Object_database) the entities and relationships map directly to object classes and named relationships. However, the term database design could also be used to apply to the overall process of designing, not just the base data structures, but also the forms and queries used as part of the overall database application within the [database management system](https://en.wikipedia.org/wiki/Database_management_system) (DBMS).[2]

The process of doing database design generally consists of a number of steps which will be carried out by the database designer. Usually, the designer must:

* Determine the relationships between the different data elements.
* Superimpose a logical structure upon the data on the basis of these relationships.[[2]](https://en.wikipedia.org/wiki/Database_design#cite_note-Teorey.2C_T.J._2009-2)

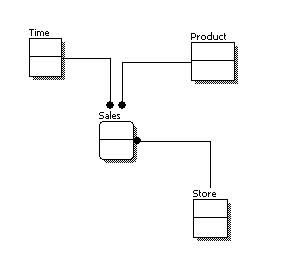
**TYPES OF DATABASE DESIGN**

1. **Conceptual database design**

A conceptual database design identifies the highest-level relationships between the different entities. Features of conceptual database design include:

* Includes the important entities and the relationships among them.
* No attribute is specified.
* No primary key is specified.

The figure below is an example of a conceptual data model.



From the figure above, we can see that the only information shown via the conceptual data model is the entities that describe the data and the relationships between those entities. No other information is shown through the conceptual data model.

1. **Logical Database design**

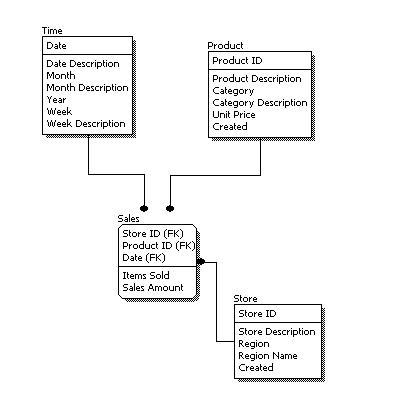
A logical database design describes the data in as much detail as possible, without regard to how they will be physical implemented in the database. Features of a logical data model include:

* Includes all entities and relationships among them.
* All attributes for each entity are specified.
* The primary key for each entity is specified.
* Foreign keys (keys identifying the relationship between different entities) are specified.
* Normalization occurs at this level.

The steps for designing the logical data model are as follows:

* Specify primary keys for all entities.
* Find the relationships between different entities.
* Find all attributes for each entity.
* Resolve many-to-many relationships.
* Normalization.

The figure below is an example of a logical data model.



1. **Physical database design**

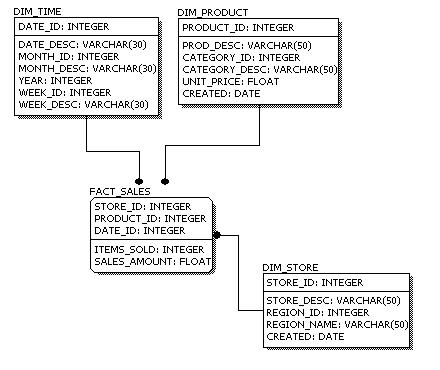
Physical database design represents how the model will be built in the database. A physical database model shows all table structures, including column name, column data type, column constraints, primary key, foreign key, and relationships between tables. Features of a physical data model include:

* Specification all tables and columns.
* Foreign keys are used to identify relationships between tables.
* De normalization may occur based on user requirements.
* Physical considerations may cause the physical data model to be quite different from the logical data model.
* Physical data model will be different for different RDBMS. For example, data type for a column may be different between My SQL and SQL Server.

The steps for physical data model design are as follows:

* Convert entities into tables.
* Convert relationships into foreign keys.
* Convert attributes into columns.
* Modify the physical data model based on physical constraints / requirements.

The figure below is an example of a physical data model.



**REFERENCES**

[1] Functional design:

Url: [https://en.wikipedia.org/wiki/Functional\_design[10/8/2015](https://en.wikipedia.org/wiki/Functional_design%5b10/8/2015)]

[2] Gehani, N. (2006). The Database Book: Principles and practice using MySQL. 1st ed., Summit, NJ.: Silicon Press

url: https://en.wikipedia.org/wiki/Database\_design#cite\_note-Teorey.2C\_T.J.\_2009-2

[3] Database design basics. (n.d.). Database design basics. Retrieved May 1, 2010, from <http://office.microsoft.com/en-us/access/HA012242471033.aspx>

url: <https://en.wikipedia.org/wiki/Database_design#cite_note-Teorey.2C_T.J._2009-2>