ST. XAVIER’S COLLEGE

**Maitighar, Kathmandu**

**(Affiliated to Tribhuvan University)**



**Database Management System**

**Theory Assignment #4**

**Submitted By**

Linus Dhakal

B.Sc. CSIT

Year II/IV Semester

013BSCIT022

**Submitted To**

Er. Sanjay Yadav

Lecturer

Department of Computer Science

St. Xavier’s College

Maitighar, Kathmandu

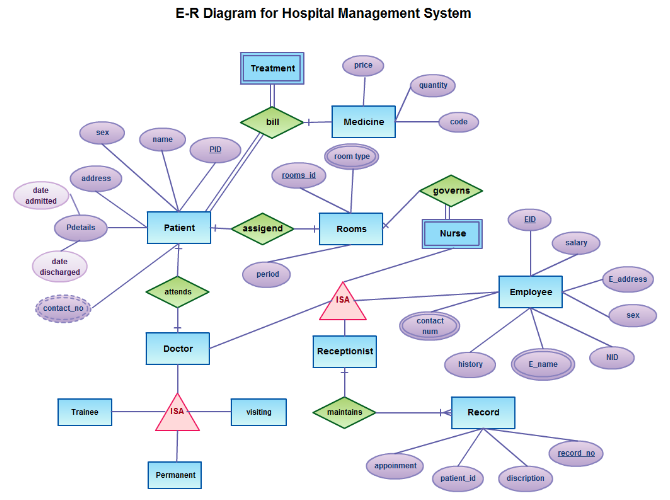
**Submitted On**

10th August, 2015

# ER Diagram with one case study

An entity-relationship diagram (ERD) is a type of data modeling that shows a graphical representation of objects or concepts within an information system or organization and their relationship to one another. Three main components of an ERD are the [entities](http://whatis.techtarget.com/definition/entity), which are objects or concepts that can have data stored about them, the relationship between those entities, and the [cardinality](http://whatis.techtarget.com/definition/cardinality), which defines that relationship in terms of numbers. Anentity relationship model*,* also called anentity-relationship (ER) diagram*,* is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of [data](http://www.webopedia.com/TERM/D/data.html) within [databases](http://www.webopedia.com/TERM/D/database.html) or information systems. An entity is a piece of data-an [object](http://www.webopedia.com/TERM/O/object.html)or concept about which data is stored.

For example, an ER diagram representing the information system for a company's sales department might start with graphical representations of entities such as the sales representative, the customer, the customer's address, the customer's order, the product and the warehouse. (See diagram) Then lines or other symbols can be used to represent the relationship between entities, and text can be used to label the relationships.



# Design

## 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)). 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, 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.

The essential guides for functional design are as follows:

1. Consider the product’s goal
2. Consider who will be using it
3. Consider what the audience wants to do with it
4. Define the transparency
5. Define how the user will know it is working
6. Decide if it able to engage users
7. Define its error handling procedure

## 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). 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.

The different types of database design are as follows:

## Conceptual Database Design

It is a process of constructing a data model for each view of the real world problem which is independent of physical considerations.

This step involves:

* Constructing the ER Model
* Check the model for redundancy
* Validating the model against user transactions to ensure all the scenarios are supported

## Logical Database Design

It is a process of constructing a model of information, which can then be mapped into storage objects supported by the Database Management System.

This step involves:

* Table Generation From ER Model
* Normalization of Tables

## Physical Database Design

The physical design of the database specifies the physical configuration of the database on the storage media.

This step involves describing the base relations, file  organizations, and indexes design used to achieve efficient access to the data, and any associated integrity constraints and security measures.

# Characteristics of Relation

The following are the major characteristics of relations in database:

• No Duplicate Tuples – A relation cannot contain two or more tuples which have the same values for all the attributes. i.e., in any relation, every row is unique.

• Tuples are unordered – The order of rows in a relation is immaterial.

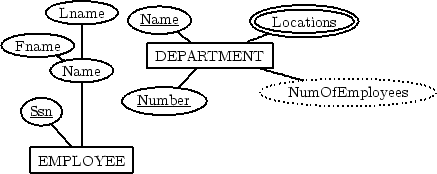
• Attributes are unordered – The order of columns in a relation is immaterial.

• Attribute Values are Atomic – Each tuple contains exactly one value for each attribute

# ER to Relational Mapping Algorithm

## Mapping of regular entity type

* Include the simple attributes
* Include the simple components of the composite attributes
* Identify the primary keys
* Don’t include: non-simple components of composite attributes, foreign keys, derived attributes, relational attributes



|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | |  |  |  Mapping of weak entity type  * For each weak entity type W with owner type E create a new relation RW that includes all the simple attributes of W as attributes of RW. * In addition include a foreign key reference to the key of the translation RE of E. * The key of RW will be the key of foreign key together with the mapped partial key from W. |
| Mapping of binary 1:1 relation type  * Include as foreign keys, in the relation of one entity type, the primary keys of the other entity type * Include also the simple attributes of the relationship type * If possible, the first entity type should have total participation in the relationship (to save memory)   -|||||||||        -|Lname||-   ----------------|  |||||||||   -||||||||-|       |DEPARTMENT     ---|Number-|--  ||Fname|-|-||||    -------||-------|||||||||||| -Ssn -  |Name||--          ||1     |Name||--  |||---  |||            --------  |-------|----|    1 ----       ----  |EMPLOYEE------------MANAGING  ----                        ----- --||||||||||                            --||StartDate||-                                 |||||||| Mapping of 1: M relationship type  * Add as foreign keys, to the relation of the entity type at the N side, the primary keys of the entity type at the 1 side (don’t duplicate records!) * Include also the simple attributes of the relationship type   -|||||||||                  -|Lname||-             -||||||||-|        -|||||Fname|-|-|||||        |Ssn--     |Name||--          -------------|          -EMPLOYEE----|             |||  |||            || ---- ||          ||----  ----|| employee||--        --|employer      N |-SUPERVISION ---|1           ----    ----               ---- Mapping M: N relationship type  * No want of duplicate records * Set as foreign keys the primary keys of the participating entity types * Include the simple attributes of the relationship type   ------      |||||||||-     ----    ----   |||||Lname||| N |-WORKS  -FOR --- |Fname||--|      ||----      -----  |||||--||||||-|||    ---||---       |Name||--||        |M   -||-||  |  |||  ---------------|   -Ssn--  | |||   |DEPARTMENT    |       --- ||||    ----------|||---    --------||---|--||||-|- -|||||||||-    -EMPLOYEE----|-|Name||- ||Number||- Mapping of multivalve attributes  * Include the given attribute * Include as foreign keys the primary attributes of the entity/relationship type owning the multivalve attribute * Keys not designated within primary keys are to be mentioned as such in side comments   -|||||||       ||||||||||||||   -|Name|--       |||Locations||-      -||-------------|||-|||||      -DEPARTMENT-----| -|||||||--|    |||||-||||||||||||||||- ||Number||-     |||NumOfEmployees||||| Mapping of N-ary relationship  * For each n-ary relationship R with n >2, create a new relation RR to represent R. * Include as foreign key attributes in RR the primary keys of the relations that correspond to the participating entities with cardinality constraints other than 1. * The primary key of RR is the combination of these foreign keys. |