ST. XAVIER’S COLLEGE

**(Affiliated to Tribhuvan University)**

Maitighar, Kathmandu



**Database Management System Assignment #4**

**Submitted by:**

Lokendra Puri  
013BSCCSIT023

**Submitted to:**

|  |  |
| --- | --- |
| Er. Sanjay Kumar Yadav Lecturer, St. Xavier’s College |  |

**Date of Submission: August 10th, 2015**

1. **ER DIAGRAM WITH ONE CASE STUDY**

Database designs also include ER ([entity-relationship model](https://en.wikipedia.org/wiki/Entity-relationship_model)) diagrams. An ER diagram is a diagram that helps to design databases in an efficient way.

Attributes in ER diagrams are usually modeled as an oval with the name of the attribute, linked to the entity or relationship that contains the attribute.

Within the [relational model](https://en.wikipedia.org/wiki/Relational_model) the final step can generally be broken down into two further steps that of determining the grouping of information within the system, generally determining what are the basic objects about which information is being stored, and then determining the relationships between these groups of information, or objects. This step is not necessary with an [Object database](https://en.wikipedia.org/wiki/Object_database).

1. **DESIGN**
   1. **FUNCTIONAL DESIGN**

Functional Design is a paradigm used to simplify the design of hardware and software devices such as computer software and increasingly, 3D models. A functional 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.

* 1. **DATABASE DESIGN**

**Database design** is the process of producing a detailed data model of a **database**. This 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, which can then be used to create a **database**.

* + 1. **CONCEPTUAL DATABASE DESIGN**

Conceptual database design involves modeling the collected information at a high-level of abstraction without using a particular data model or DBMS.

**REASONS FOR CONCEPTUAL MODELING**

• Independent of DBMS.

• Allows for easy communication between end-users and developers.

• Has a clear method to convert from high-level model to relational model.

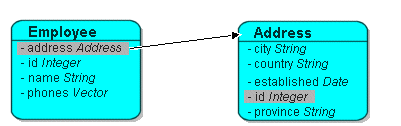
• Conceptual schema is a permanent description of the database requirements.

* + 1. **LOGICAL DATABASE DESIGN**
    2. **PHYSICAL DATABASE DESIGN**

The physical design of the database specifies the physical configuration of the database on the storage media. This includes detailed specification of [data elements](https://en.wikipedia.org/wiki/Data_element), [data types](https://en.wikipedia.org/wiki/Data_type), [indexing](https://en.wikipedia.org/wiki/Index_%28database%29) options and other parameters residing in the DBMS [data dictionary](https://en.wikipedia.org/wiki/Data_dictionary). It is the detailed design of a system that includes modules & the database's hardware & software specifications of the system.

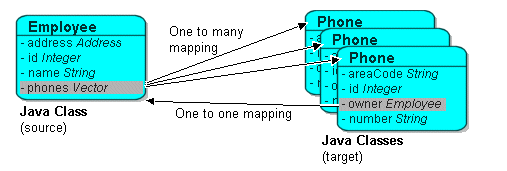
1. **CHARACTERISTICS OF RELATION**
2. Data in the relational database must be represented in tables, with values in columns within rows.
3. Data within a column must be accessible by specifying the table name, the column name, and the value of the primary key of the row.
4. The DBMS must support missing and inapplicable information in a systematic way, distinct from regular values and independent of data type.
5. The DBMS must support an active on-line catalogue.
6. The DBMS must support insert, update, and delete operations on sets.
7. The DBMS must support logical data independence.
8. The DBMS must support physical data independence.
9. Integrity constraints must be stored within the catalogue, separate from the application.
10. The DBMS must support distribution independence. The existing application should run when the existing data is redistributed or when the DBMS is redistributed.
11. **ER TO RELATIONAL MAPPING ALGORITHM**
    1. **MAPPING OF REGULAR ENTITY TYPE**
    2. **MAPPING OF WEAK ENTITY TYPE**
    3. **MAPPING OF BINARY 1:1 TYPE**

One-to-one mappings represent simple pointer references between two Java objects. In Java, a single pointer stored in an attribute represents the mapping between the source and target objects. Relational database tables implement these mappings using foreign keys.



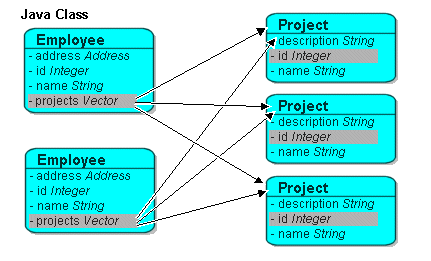
* 1. **MAPPING OF 1:N RELATION TYPE**

One-to-many mappings are used to represent the relationship between a single source object and a collection of target objects. They are a good example of something that is simple to implement in Java using a Collection (or other collection types) of target objects, but difficult to implement using relational databases.

****

* 1. **MAPPING OF M:M RELATION TYPE**

Many-to-many mappings represent the relationships between a collection of source objects and a collection of target objects. They require the creation of an intermediate table for managing the associations between the source and target records.

****

* 1. **MAPPING OF MULTIVALUED ATTRIBUTES**
  2. **MAPPING OF N-ARY RELATIONSHIP TYPES**