**St. Xavier’s College**

**(Affiliated to Tribvhuwan University)**

Maitighar, Kathmandu



**Database Management System**

**LAB REPORT ON BOOK STORE**

**Submitted By:**

**Narayan Panthi**

013BSCCSIT024

Submitted To:

**Er. Sanjay Kumar Yadav**

Lecturer

Department of Computer Science

Date of Submission: August 10, 2015

# STATEMENT:

## ER DIAGRAM WITH ONE CASE STUDY

## DESIGN

## FUNCTIONAL DESIGN

## DATABASE DESIGN

## CONCEPTUAL DATABASE DESIGN

## LOGICAL DATABASE DESIGN

## PHYSICAL DATABASE DESIGN

## CHARACTERISTICS OF RELATION

## ER TO RELATIONAL MAPPING ALGORITHM

## MAPPING OF N-ARRAY RELATIONSHIP TYPE

**BEST- BOOK STORE**

* **Introduction**  
    
  The purpose of this case study is to present a data-intensive application for which a database can be designed. The methodology to be followed uses conceptual design in the Entity Relationship Model, followed by a mapping of the ER schema into the relational schema. The application is broken into a number of tasks and the inputs and outputs are related to the tasks by means of an Information Flow Diagram. Students should use ‘Role of Information Systems’ and ‘Database Design and Implementation Process’ to understand the overall process of database design used for design of large databases. Best-BookStore is an information management system that supports some of the services involved in an online bookstore.
* **Welcome screen.**  
    
  The rest of this document is organized as follows. We describe the operations and screens that a user will go through when one of the above options are selected.
* **Search Only**

Let’s say that the user only wants to search the bookstore for a specific book. She clicks “Search only” and the screen in Figure 2 pops up. Internally, the system will create a temporary shopping cart for these new user main functionalities of the system.

* **Search Result screen.**  
    
  In the Search Result screen, the user can perform the following activities:
  + Add to Cart one or more books listed. By clicking on the “Add to Cart” button next to a book, the system inserts this book into the user’s shopping cart, disables the respective “Add to Cart” button, and updates the total number of items in the cart. If the user later decides that she does not want this book in her shopping, she can remove it by clicking on “Manage Shopping Cart” button.
  + Review one or more books listed. By clicking on the “Review” button next to a book, the system brings up the Book Review screen (Figure 4), which displays a scrollable window that lists all the reviews for the selected book. By clicking the “Done” button on Book Review screen, the system brings back the Search Result screen.
  + Manage Shopping Cart, which allows the user to view the list of books in her shopping cart and modify this list accordingly.
  + Proceed to checkout, which allows the user to place an order for the books in her shopping cart and make the payment. If she is a new user, then she needs to first register into the system. For that, the system will pop the screen from Figure 6. Otherwise, the screen from Figure 8 is displayed. We will describe the checkout process shortly.  
      
    If the user requests a New Search, the system will bring back the Search screen
* Book Reviews screen.  
    
  Let say that the user has selected both books listed in Figure 3, and she wants to manage her shopping cart. Upon clicking the “Manage Shopping Cart” button, the screen in will pop up.
* Manage Shopping Cart screen.
* Customer Registration screen
* Message informing the user why she needs to register to the system.
* Confirm Order screen
* Update Customer Profile screen
* Proof of Purchase screen.
* Register New Customer
* User Login screen.
* List of Suggested Books screen.
* Administrator
* Administrator Tasks screen.
* Manage Bookstore Catalog
* Manage Bookstore Catalog screen.
* Inserting a New Book
* Book Insertion screen.
* Catalog screen
* Modifying/Deleting an Existing Book
* Customer Profile

## ER DIAGRAM WITH ONE CASE STUDY

# Functional design

In conventional database systems, procedures, data structures and actual content are usually separated. Thus, a conventional database management systems (DBMS) provides users with a possibility to store, modify or retrieve data that structured in accordance with a current database schema

# Conceptual Design

Once all the requirements have been collected and analyzed, the next step is to create a conceptual shema for the database, using a high level conceptual data model. This phase is called conceptual design.  
  
The result of this phase is an Entity-Relationship (ER) diagram or UML class diagram. It is a high-level data model of the specific application area. It describes how different entities (objects, items) are related to each other. It also describes what attributes (features) each entity has. It includes the definitions of all the concepts (entities, attributes) of the application area.

# Logical Design

The result of the logical design phase (or data model mapping phase) is a set of relation shcemas. The ER diagram or class diagram is the basis for these relation schemas.  
  
The relation schemas are the basis for table definitions. In this phase (if not done in previous phase) the primary keys and foreign keys are defined.

# Normalization

Normalization is the last part of the logical design. The goal of normalization is to eliminate redundancy and potential update anomalies.  
  
Normalization is a technique by which one can modify the relation schema to reduce the redundancy. Each normalization phase adds more relations (tables) into the database.

# Physical Design

The goal of the last phase of database design, physical design, is to implement the database. At this phase one must know which database management system (DBMS) is used. For example, different DBMS's have different names for datatypes and have different datatypes.

# Characteristics of relation

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.  
It may be noted that many of the properties of relations follow the fact that the body of a relation is a mathematical set.

## ER to Relational Model

* ***ER-to-Relational Mapping Algorithm***
* STEP 1: For each regular (strong) entity type E in the ER schema, create a relation R that includes all the simple attributes of E. Include only the simple component attributes of a composite attribute. Choose one of the key attributes of E as primary key for R. If the chosen key of E is composite, the set of simple attributes that form it will together form the primary key of R.  
    
  **NOTE:** The foreign key and relationship attributes, if any, are not included yet at this step.
* STEP 2: For each weak entity type W in the ER schema with owner entity type E, create a relation R, and include all simple attributes (or simple components of composite attributes) of W as attributes of R. In addition, include as foreign key attributes of R the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s); this takes care of the identifying relationship type of W. The primary key of R is the combination of the primary key(s) of the owner(s) and the partial key of the weak entity type W, if any.

Example: The DEPENDENT relation  
  
**NOTE:** It is common to choose the *propagate* (CASCADE) option for the referential triggered action (see Section 8.1) on the foreign key in the relation corresponding to the weak entity type.

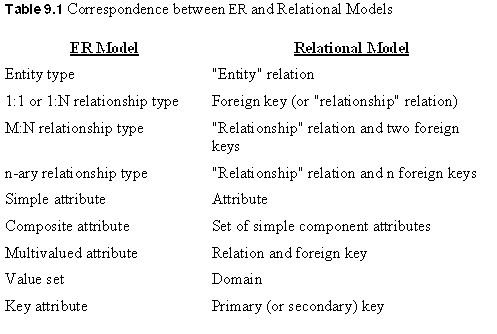
* STEP 3: For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R. Choose one of the relations—S, say—and include as foreign key in S the primary key of T.  Include all the simple attributes (or simple components of composite attributes) of the 1:1 relationship type R as attributes of S.    
    
  Example: The MANAGE relationship
* STEP 4: For each regular binary 1:N relationship type R, identify the relation S that represents the participating entity type at the N-side of the relationship type. Include as foreign key in S the primary key of the relation T that represents the other entity type participating in R.

Examples: WORKS\_FOR, CONTROLS, and SUPERVISION

* STEP 5: For each binary M:N relationship type R, create a new relation S to represent R. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types; their combination will form the primary key of S.   
    
  Example: WORKS\_ON
* STEP 6: For each multivalued attribute A , create a new relation R. This relation R will include an attribute corresponding to A, plus the primary key attribute K—as a foreign key in R—of the relation that represents the entity type or relationship type that has A as an attribute. The primary key of R is the combination of A and K. If the multivalued attribute is composite, we include its simple components.

Example: a new relation DEPT\_LOCATIONS

* STEP 7: For each n-ary relationship type R, where n > 2, create a new relation S to represent R. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types.
* **Summary of the Mapping**



## RELATIONAL SCHEMA DESIGN

#### REFERENCES & RESOURCES

 Paul Litwin, "Fundamentals of Relational Database Design", available at <http://www.deeptraining.com/litwin/dbdesign/FundamentalsOfRelationalDatabaseDesign.aspx>.

 Codd E. F., "A Relational Model of Data for Large Shared Data Banks", Communications of the ACM, vol. 13, issue 6, pp. 377–387, June 1970.