ST. XAVIER’S COLLEGE

**(Affiliated to Tribhuvan University)**

**Maitighar, Kathmandu**

****

**Database Management Systems**

**Assignment #4**

**SUBMITTED BY:**

Urendra khakurel

013BSCCSIT045

**SUBMITTED TO**

|  |  |
| --- | --- |
| **Mr. Sanjay Kr. Yadav**  **( Lecturer )** |  |
| **Department of Computer Science** | |

1) ER diagram with case study :

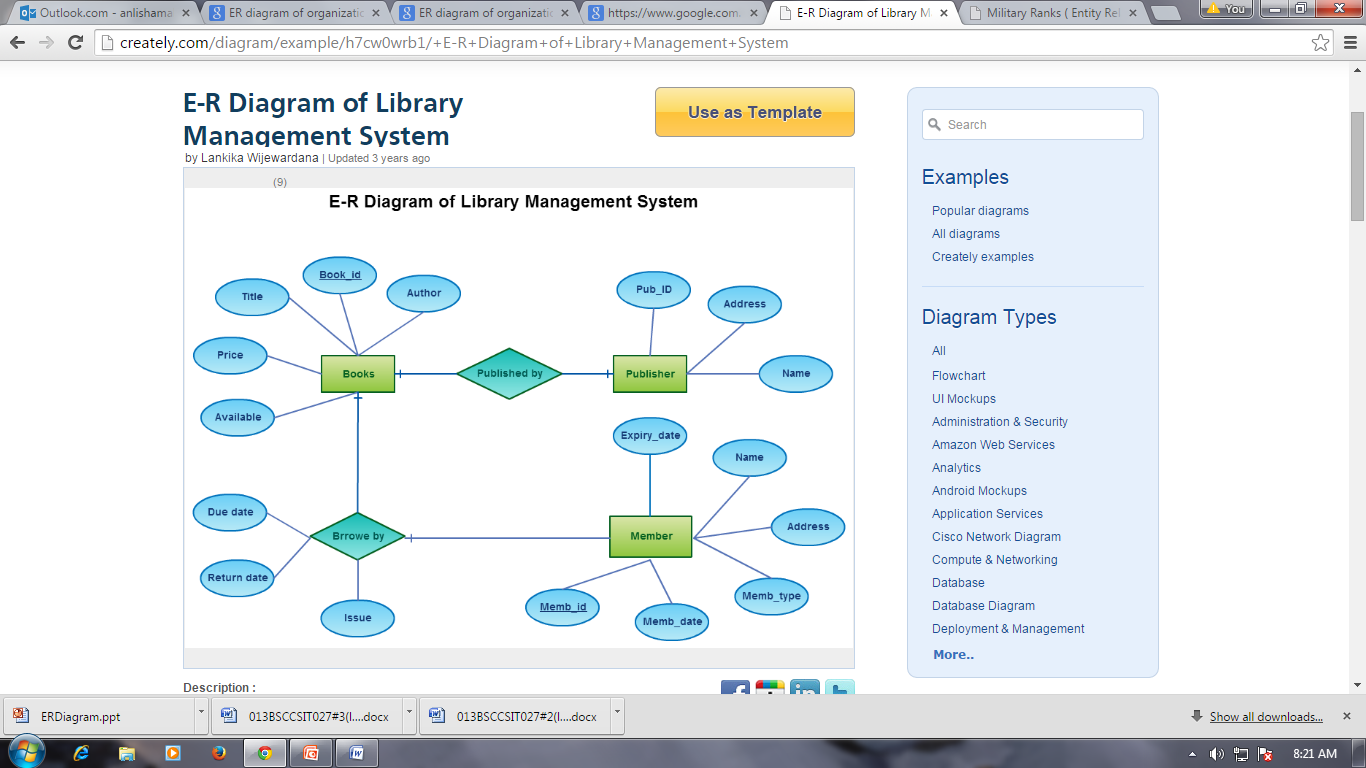
2) Design

1. Functional design
2. Database design
3. Conceptual database design
4. Logical database design
5. Physical database design

3) Characterstics of relation

4) ER to relationship mapping algorithm

1) ER diagram with case study :



2) 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.

**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.

**Conceptual Design**

Once all the requirements have been collected and analyzed, the next step is to create a conceptual schema 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.  
  
 During or after the conceptual schema design, the basic data model operations can be used to specify the high-level user operations identified during the functional analysis. This also serves to confirm that the conceptual schema meets all the indentified functional requirements.

**Logical Database Design:**

Once all the requirements have been collected and analyzed, the next step is to create a conceptual schema 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.  
  
During or after the conceptual schema design, the basic data model operations can be used to specify the high-level user operations identified during the functional analysis. This also serves to confirm that the conceptual schema meets all the indentified functional requirements.

**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 data types and have different data types.  
  
The SQL clauses to create the database are written. The indexes, the integrity constraints (rules) and the users' access rights are defined.

Finally the data to test the database is added in.  
  
In parallel with these activities, application programs are designed. The implementation of the programs can start when the database is created and data has been added in. Source: Elmasri, Navathe. 2004. Fundamentals of Database Systems. Addison-Wesley. Chapter 3. Data Modelling Using the Entity-Relationship Model.

1. **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.
* 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.

* 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.
* 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.  (WHY?)
* 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.   
    
    
  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.
* 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.