**ST. XAVIER’S COLLEGE**

**(Affiliated to Tribhuvan University)**

Maitighar, Kathmandu



**Database Management System**

**Lab Assignment #4**

**Submitted by:**

Prashraya Hada  
013BSCCSIT027

**Submitted to:**

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

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

1. ER diagram with case study
2. Design
   1. Functional design
   2. Database design
      1. Conceptual database design
      2. Logical database design
      3. Physical database design
3. Characteristics of relation
4. ER to relational mapping algorithm

Step 1: Mapping of regular entity types

Step 2: Mapping of weak entity types

Step 3: Mapping of binary 1:1 relational types

Step 4: Mapping of binary 1: N relational types

Step 5: Mapping of binary M: N relational types

Step 6: Mapping of Multivalued attributes

Step 7: Mapping of N-ary relational types

1. **ER diagram with case study**

An entity relationship model, also called an entity-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 within database or information systems. An entity is a piece of data-an object or concept about which data is stored.

An ERD is a conceptual and representational model of data used to represent the entity framework infrastructure.

The elements of an ERD are:

* Entities
* Relationships
* Attributes

Steps involved in creating an ERD include:

1. Identifying and defining the entities
2. Determining all interactions between the entities
3. Analyzing the nature of interactions/determining the cardinality of the relationships
4. Creating the ERD
5. **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. **Functional Design:**

Functional design is the design method in which the system is seen from the functional viewpoint. The design concentrates on isolating high-level functions that can then be decomposed into and synthesized from lower-level functions. The standard way to assure functional design is to review the description of a module. The responsibilities need to be divided into several modules in order to achieve a functional design.

* 1. **Database Design:**
  2. **Conceptual Data Model:** This is the highest level ER model in that it contains the least granular detail but establishes the overall scope of what is to be included within the model set. The conceptual ER model normally defines master reference data entities that are commonly used by the organization. Developing an enterprise-wide conceptual ER model is useful to support documenting the data architecture for an organization.

A conceptual ER model may be used as the foundation for one or more *logical data models*. The purpose of the conceptual ER model is then to establish structural metadata commonality for the master data entities between the set of logical ER models. The conceptual data model may be used to form commonality relationships between ER models as a basis for data model integration.

* 1. **Logical Data Model:** A logical ER model does not require a conceptual ER model, especially if the scope of the logical ER model includes only the development of a distinct information system. The logical ER model contains more detail than the conceptual ER model. In addition to master data entities, operational and transactional data entities are now defined. The details of each data entity are developed and the relationships between these data entities are established. The logical ER model is however developed independent of technology into which it can be implemented.
  2. **Physical Data Model:** One or more physical ER models may be developed from each logical ER model. The physical ER model is normally developed to be instantiated as a database. Therefore, each physical ER model must contain enough detail to produce a database and each physical ER model is technology dependent since each database management system is somewhat different.

The physical model is normally instantiated in the structural metadata of a database management system as relational database objects such as database table, database index such as unique key indexes, and database constraints such as a key constraints or a commonality constraint. The ER model is also normally used to design modifications to the relational database objects and to maintain the structural metadata of the database.

1. **Relational Database Characteristics**

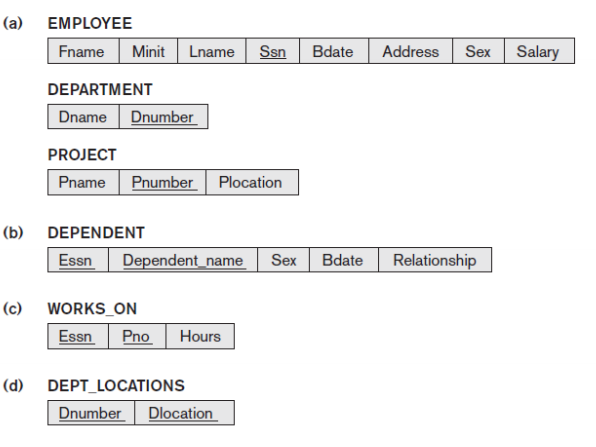
* + - * Data in the relational database must be represented in tables, with values in columns within rows.
      * Data within a column must be accessible by specifying the table name, the column name, and the value of the primary key of the row.
      * The DBMS must support missing and inapplicable information in a systematic way, distinct from regular values and independent of data type.
      * The DBMS must support an active on-line catalogue.
      * The DBMS must support at least one language that can be used independently and from within programs, and supports data definition operations, data manipulation, constraints, and transaction management.
      * Views must be updatable by the system.
      * The DBMS must support insert, update, and delete operations on sets.
      * The DBMS must support logical data independence.
      * The DBMS must support physical data independence.
      * Integrity constraints must be stored within the catalogue, separate from the application.
      * The DBMS must support distribution independence.  The existing application should run when the existing data is redistributed or when the DBMS is redistributed.
      * If the DBMS provides a low level interface (row at a time), that interface cannot bypass the integrity constraints.

1. **ER to relational mapping algorithm**

Step 1: Mapping of Regular Entity Types:

For each regular entity type, create a relation R that includes all the simple attributes of E Called entity relations. Each tuple represents an entity instance.

Step 2: Mapping of Weak Entity Types:

 For each weak entity type, create a relation R and include all simple attributes of the entity type as attributes of R. Includes primary key attribute of owner as foreign key attributes of R.

Step 3: Mapping of binary 1:1 relational types

For each binary 1:1 relationship type, identify relations that correspond to entity types participating in R. Possible approaches:

• Foreign key approach

• Merged relationship approach

• Cross reference or relationship relation approach

Step 4: Mapping of binary 1: N relational types

Step 5: Mapping of binary M: N relational types

Step 6: Mapping of Multivalued attributes

Step 7: Mapping of N-ary relational types