Lab assignment #4

1.ER diagram with 1 case study.

2. Design

i. Functional design.

ii. Database design.

-conceptual database design

-logical database design

-physical database design

3. Characteristics of relation

4. E-R to relational mapping algorithm

i. Mapping of regular entity types.

ii. Mapping of weak entity type

iii. Mapping of binary 1:1 relational type.

iv. Mapping of 1:N relational type

v. Mapping of M:N relationship type

vi. Mapping of multivalued attributes.

vii. Mapping of N-array relation type.

ST. XAVIER’S COLLEGE

**(Affiliated to Tribhuvan University)**

Maitighar, Kathmandu



**Database Management System**

**Lab aassignment #4**

**Submitted by:**

Arjun Neupane

013BSCCSIT009

**Submitted to:**

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

Date of submission: 10th August 2015

**1.E-R diagram :**

An entity-relationship diagram (ERD) is a graphical representation of an information system that shows the relationship between people, objects, places, concepts or events within that system. An ERD is a data modeling technique that can help define business processes and can be used as the foundation for a relational database.

While useful for organizing data that can be represented by a relational structure, an entity-relationship diagram can't sufficiently represent semi-structured or unstructured data, and an ERD is unlikely to be helpful on its own in integrating data into a pre-existing information system.

Three main components of an ERD are the entities, which are objects or concepts that can have data stored about them, the relationship between those entities, and the cardinality, which defines that relationship in terms of numbers.

For example, an ER diagram represents the relationship between two entities set user and database designer. They do have a strong relationship.

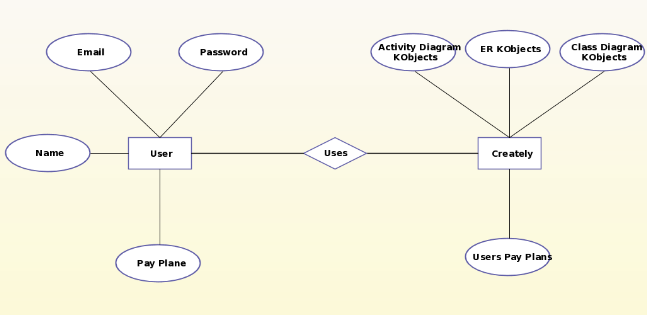


Fig 1: E-R diagram

**2. Design:**

Determine the purpose of the database which helps prepare for the remaining steps.

Find and organize the information required - Gather all of the types of information to record in the database, such as product name and order number.

Divide the information into tables - Divide information items into major entities or subjects, such as Products or Orders. Each subject then becomes a table.

Turn information items into columns - Decide what information needs to be stored in each table. Each item becomes a field, and is displayed as a column in the table. For example, an Employees table might include fields such as Last Name and Hire Date.

Specify primary keys - Choose each table’s primary key. The primary key is a column, or a set of columns, that is used to uniquely identify each row. An example might be Product ID or Order ID.

Set up the table relationships - Look at each table and decide how the data in one table is related to the data in other tables. Add fields to tables or create new tables to clarify the relationships, as necessary.

Refine the design - Analyze the design for errors. Create tables and add a few records of sample data. Check if results come from the tables as expected. Make adjustments to the design, as needed.

Apply the [normalization rules](https://en.wikipedia.org/wiki/Database_normalization) - Apply the data normalization rules to see if tables are structured correctly; make adjustments to the tables.

**Types of design:**

**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.

Types :

i.Conceptual database design:

The first phase should be the Conceptual Design Review (CDR).  The purpose of this review is to validate the application concept.  This involves a presentation of the statement of purpose as well as a general overview of the desired functionality that will be provided by the application.  A CDR should be conducted as early as possible to determine the overall feasibility of a project.  Failure to conduct a CDR can result in:

* projects which provide duplicate or inadequate functionality.
* projects which are cancelled due to lack of funds, staffing, planning, user participation, and/or management interest
* over-budget projects

ii. Logical database design:

The second phase is the Logical Design Review (LDR).  This should be conducted when the first cut of the logical data model has been completed.  A thorough review of all data elements, descriptions, and relationships should occur during the LDR.  The LDR should scrutinize the following areas:

* Has the logical model been thoroughly examined to ensure that all of the required business functionality can be achieved based solely upon the model?
* Is the model in (at least) third normal form?
* Have all of the data elements (entities and attributes) required for this application been identified?
* Have the data elements which have been identified been documented accurately?
* Have all of the relationships been defined properly?

Failure to hold an LDR can result in a poorly designed database, a failure to identify all required pieces of data, and a lack of documentation causing an application to be developed which is difficult to maintain.  If further data modeling occurs after the logical design review is held, further LDRs can be scheduled as the project progresses.

iii. Physical Database design:

The Physical Design Review is the third design review phase.  It is this component which most DB2 developers associate with the design review process.  This is where the database is reviewed in detail to ensure that all of the proper design choices were made.  In addition, the DA and DBA should ensure that a proper translation from logical to physical was made including conformance to standards, mapping to appropriate database objects, and performance tweaking for physical DBMS issues. Additionally, all de-normalization decisions are to be documented for posterity.

The overall operating environment for the application should be described and verified at this stage.  The choice of transaction processing (CICS, IMS/DC, Tuxedo, web, etc.) and a complete description of the on-line environment should be provided.  This should include the number of transactions and the anticipated workload.  Correspondingly, a complete description of any batch processes should also be provided.

At this stage, all of the SQL that will be used for this application may not be available.  General descriptions of the processes required, however, should be available.  From the process descriptions, a first-cut de-normalization effort (if required at all) should be either attempted or verified.

As the PDR phase requires a lot of in-depth attention, it can be broken into discrete pieces if so desired.  The PDR, or pieces thereof, can also be done more than once prior to implementation if significant changes occur to the physical design of the database or application.

Participants should include:  AA, AD, DA, DBA, IC, OLS, OS, TS.

Reference:

<https://datatechnologytoday.wordpress.com/2013/12/12/the-7-types-of-database-design-reviews/>