ST. XAVIER’S COLLEGE

**(Affiliated to Tribhuvan University)**

**Maitighar, Kathmandu**

****

**DATABASE MANAGEMENT SYSTEM**

**THEORY ASSIGNMENT#4**

**Submitted by:**

**Rojesh Tamrakar**

013BSCCSIT032

**Submitted to:**

|  |  |
| --- | --- |
| **Er. Sanjay Kumar Yadav** |  |

Lecturer

Department of Computer Science

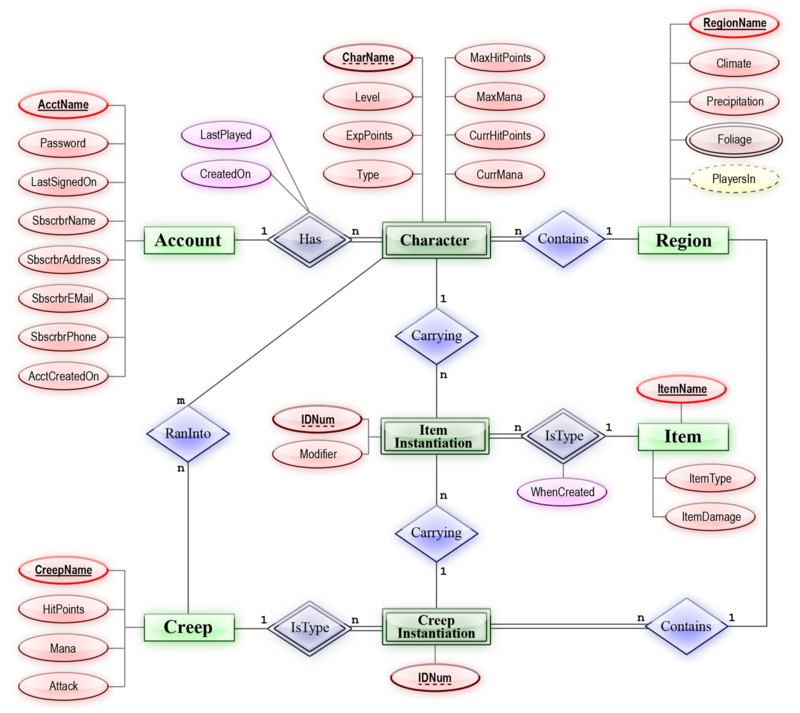
Date of submission: 10th August, 2015

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**
2. **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. Functionally designed modules tend to have low coupling. The advantage for implementation is that if a software module has a single purpose; it will be simpler, and therefore easier and less expensive, to design and implement. Systems with functionally designed parts are easier to modify because each part does only what it claims to do.

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. A fully attributed data model contains detailed attributes for each entity. The term database design can be used to describe many different parts of the design of an overall database system. Principally, and most correctly, it can be thought of as the logical design of the base data structures used to store the data. In the relational model these are the tables and view. In an object database the entities and relationships map directly to object classes and named relationships. However, the term database design could also be used to apply to the overall process of designing, not just the base data structures, but also the forms and queries used as part of the overall database application within the database management system (DBMS).

1. **Conceptual Database Design**

Conceptual Design is an umbrella term given to all forms of non-aesthetic design management disciplines. It is the design of interactions, experiences, processes and strategies and is the point at which people, knowledge, products, services, processes, and profitability meet vision and endless possibilities, each acting as a distinct colour on the canvass of the designer. Conceptual designers tend to be better geared to abstract reasoning in any given environment, quickly able to understand the underlying motivations of key players, root-causes of failure, as well as feeling and connecting with the human elements that any system, experience or interaction has on its users.

1. **Logical Database Design**

Once the relationships and dependencies amongst the various pieces of information have been determined, it is possible to arrange the data into a logical structure which can then be mapped into the storage objects supported by the [database management system](https://en.wikipedia.org/wiki/Database_management_system). In the case of [relational databases](https://en.wikipedia.org/wiki/Relational_databases) the storage objects are [tables](https://en.wikipedia.org/wiki/Database_table) which store data in rows and columns.

Each table may represent an implementation of either a logical object or a relationship joining one or more instances of one or more logical objects. Relationships between tables may then be stored as links connecting child tables with parents. Since complex logical relationships are themselves tables they will probably have links to more than one parent.

1. **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, data types, indexing options and other parameters residing in the DBMS 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 Relations**
   * + - 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.
2. **ER to relational mapping algorithm**
3. **Mapping of regular entity types**

* Create table for a relationship.
* Add the primary keys of all participating Entities as fields of table with their respective data types.
* If relationship has any attribute, add each attribute as field of table.
* Declare a primary key composing all the primary keys of participating entities.
* Declare all foreign key constraints.

1. **Mapping of weak entity types**

* Create table for weak entity set.
* Add all its attributes to table as field.
* Add the primary key of identifying entity set.
* Declare all foreign key constraints

1. **Mapping of binary (1:1) relationship type**
2. **Mapping of binary (1:n) relationship type**
3. **Mapping of binary (m:n) relationship type**
4. **Mapping of multivalved attribute**
5. **Mapping of n-array relationship types**