**ST. Xavier's College**

**Maitighar, Kathmandu**

****

**Lab Assignment on Database Management System #5**

# 

# SUBMITTED BY

Shaurabh Chapagain

O13BSCCSIT037

**SUBMITTED TO**

|  |  |
| --- | --- |
| Er. Sanjay Kumar Yadav  Lecturer  Department of Computer Science  St. Xavier’s College |  |

**Date of Submission: 27th August, 2015**

**Entity-Relationship Diagram**

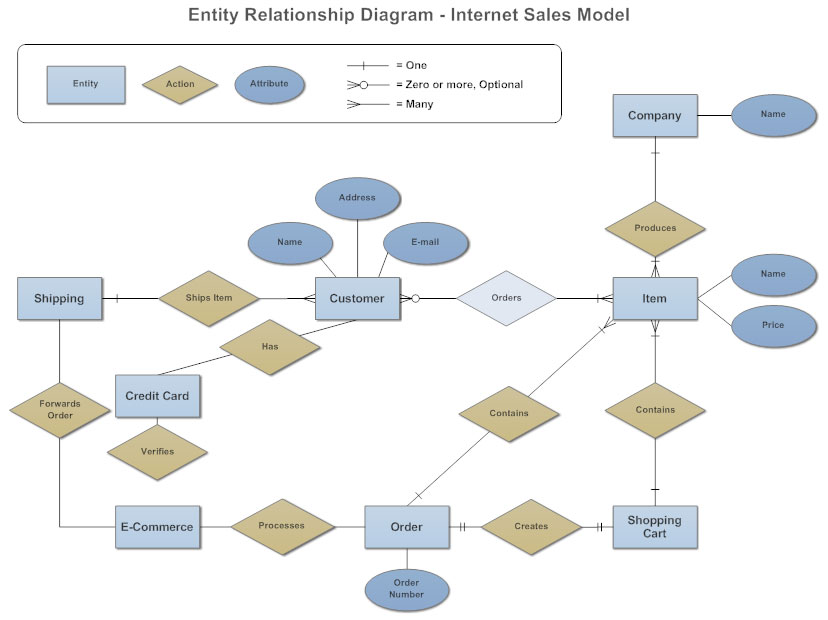
An entity-relationship diagram, or ERD, is a chart that visually represents the relationship between database entities. ERDs model an organization's data storage requirements with three main components: entities, attributes and relationships. Entity Relationship model allows us to sketch database designs. ERD is widely used in database design. It is the model that identifies the concepts or relationships between those entities.

The database analyst/designer gains a better understanding of the information to be contained in the database through the process of constructing the ERD. The ERD serves as a documentation tool. Finally, the ERD is used to communicate the logical structure of database to users. In particular, the ERD effectively communicates the logic of the database to users.

Components of an ERD:

An ERD typically consists of four different graphical components. They are as follow:

1. Entity
2. Relationship
3. Cardinality
4. Attribute



**Entity**

An entity is any singular, identifiable and separate object. It refers to individuals, organizations, systems, bits of data or even distinct system components that are considered significant in and of them-selves. It refers to individual things, including people, concepts or objects with data that is first stored in a database management system (DBMS) and has attributes and relationships to other entities.

For example, to develop a company's database for maintaining information on employees, the application should be able to store and provide data on employee such as when was the employee was hired; is the employee still with the company; if the employee has left the company when did he leave the company; which department does employee work for; who is his/her manager; what is his/her skill level etc.  In this example, the entities are company, department, employee, manager.

**Attributes**

An Attribute is a property that describes an entity. In the above example, the employee is the entity and employee’s name, age, address, salary and job etc are the attribute.

Some entities can have many attributes while others may only have a couple. As well, there are five categories that attributes are classified in. This simple table will be used to explain how each attribute can be a different type of attribute:

**Single and Composite attributes**

Attributes can be classified as having many parts to them or just a single unbreakable attribute. The composite attribute is an attribute that can be subdivided into other single attributes with meanings of their own. A single attribute is just an attribute that cannot be subdivided into parts.

Example: Imagine from the entity Student that instead of having the three attributes: stu\_LastName, stu\_MiddleName, stu\_FirstName it had one attribute called stu\_Name. The attribute stu\_Name would be considered a composite attribute since it can be subdivided into the other three attributes: stu\_LastName, stu\_MiddleName, stu\_FirstName. The rest of attributes would be considered single attributes since they can't be subdivided into parts.

**Single-valued and Multi-valued attributes**

Attributes can be classified as single or multi-value. The single-value attribute can only have one value, while the multi-valued attributes usually can store multiple data in them.

Example: In the entity Student, stu\_Address could be considered a multi-value attribute since a student could have multiple addresses where he lives at. An example of a single-value attribute would be stu\_LastNamesince a student usually has one last name that uniquely identifies him/her.

**Derived Attributes**

The last category that attributes can be defined is called a derived attribute, where one attribute is calculated from another attribute. The derived attribute may not be stored in the database but rather calculated using algorithm.

Example: In the entity Student, stu\_Age would be considered a derived attribute since it could be calculated using the student's date of birth with the current date to find their age.

Examples of derived attributes are: salary and age.

DOB is not an example of a derived attribute because it is inputted information and not calculated.