ST. XAVIER’S COLLEGE

**(Affiliated to Tribhuvan University)**

Maitighar, Kathmandu



**Database Management System Assignment #5**

**Submitted by:**

Bikash Paneru  
013BSCCSIT012

**Submitted to:**

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

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

**1. What do you mean by Entity- Relationship Diagram? Explain**

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 databases or information systems. ERDs model an organization’s data storage requirements with three main components: entities, attributes, and relationships. An entity is a piece of data-an object or concept about which data is stored.

As noted above, there are 3 ingredients in a standard entity-relationship diagram:

* **Entities**, which represent people, places, items, events, or concepts.
* **Attributes**, which represent properties or descriptive qualities of an entity. These are also known as data elements.
* **Relationships**, which represent the link between different entities.

Entities, attributes, and relationships can be represented in one of three ways: with a **conceptual model**, **logical model**, or **physical model**. These models increase in complexity as we move from conceptual to logical to physical. It's usually best to start with a conceptual ERD model, so we can understand—at the highest level—the entities in our data and how they relate to each other. As we transform a conceptual ERD to a physical model, we'll learn exactly how to implement modeled information into the database of our choice.

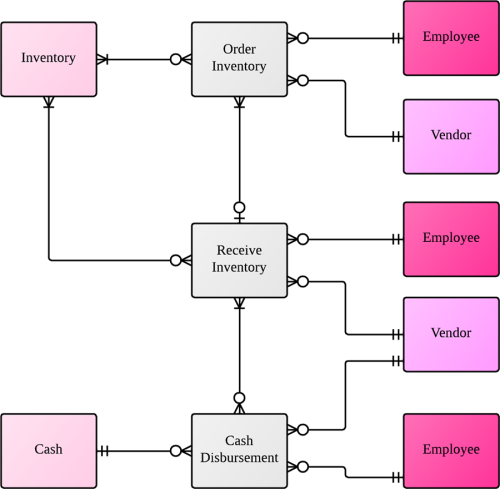


Figure Entity Relationship Diagram

**2. Define entity and give an example.**

A database contains one or more related tables. Each table holds all the information about an object, person or thing i.e each table is about an object, person or thing. These are entities.

Thus, an entity is any real world object which has a set of attributes that define it.

Example: Person, Customer, Student, Payment, Product. Etc.

**3. Explain the difference between an entity class and an entity instance.**

|  |  |  |
| --- | --- | --- |
| **Entity Class** | **S.N** | **Entity Instance** |
| An Entity Class defines where in a relationship an instance of this class can play a role. | 1 | An Entity Instance plays specific roles in a relationship, defined by its class. |
| A class can have zero or more instances. | 2 | An instance can have only one class |
| They define the nature of characteristics that an instance may have. | 3 | They have the nature of characteristics that their class defines. |
| Example: | 4 | Example: |
| http://www.openresource.com/on_ontiki/CIER_1.png |  | http://www.openresource.com/on_ontiki/CIER_2.png |

**4. Define attribute and its types.**

An attribute may describe a component of the database, such as a table or a field, or may be used itself as another term for a field. The types of attributes are:

* **Simple and Composite Attributes:** 
  + **Simple attributes** are those attributes which cannot be broken down further. Eg: first\_name, last\_name.
  + **Composite attributes** are those attributes which can be broken down. Eg: person\_name can be broken down into first\_name and last\_name
* **Stored and Derived Attributes:**
  + **Stored attributes** are those attributes whose values are simply stored and do not depend on other attributes. Eg: id, class
  + **Derived attributes** are those attributes whose values depend on other attributes. Eg: age can be derived from date\_of\_birth
* **Single valued and Multi valued Attributes:**
  + **Single valued attributes** are those attributes which can only have one value. Eg: fname, age
  + **Multi valued attributes** are those attributes which can have more than one value. Eg: phone\_no

**5. What is derived attributes?**

**Derived attributes** are those attributes whose values depend on other attributes. Derived attributes are usually calculated from other attributes, such as multiplying an employee’s monthly salary by twelve or deriving a person’s full name from first name and last name attributes. Derived attributes are effectively read-only since there is no place to write them back to.

**Example:** age can be derived from date of birth. interest can be calculated from principle, interest and time.

**6. Define relationship and give an example.**

A relationship, in the context of databases, is a situation that exists between two relational database tables when one table has a foreign key that references the primary key of the other table. Relationships allow relational databases to split and store data in different tables, while linking disparate data items.

A relationship is an association between two entities.

Example: In a bank database a CUSTOMER\_MASTER table stores customer data with a primary key column named CUSTOMER\_ID; it also stores customer data in an ACCOUNTS\_MASTER table, which holds information about various bank accounts and associated customers. To link these two tables and determine customer and bank account information, a corresponding CUSTOMER\_ID column must be inserted in the ACCOUNTS\_MASTER table, referencing existing customer IDs from the CUSTOMER\_MASTER table. In this case, the ACCOUNTS\_MASTER table’s CUSTOMER\_ID column is a foreign key that references a column with the same name in the CUSTOMER\_MASTER table.

**7. Explain the difference between a relationship class and a relationship instance.**

|  |  |  |
| --- | --- | --- |
| **Relationship Class** | **S.N** | **Relationship Instance** |
| A Relationship Class defines the nature of characteristics that instances may have. | 1 | A Relationship Instance defines the specific details of these characteristics. |
| A class can have zero or more instances. | 2 | An instance can have only one class |
| They have descriptive names. | 3 | They have mechanically generated IDs. |
| Example: | 4 | Example: |
| http://www.openresource.com/on_ontiki/CIER_1.png |  | http://www.openresource.com/on_ontiki/CIER_2.png |

**8. Define degree of relationship.**

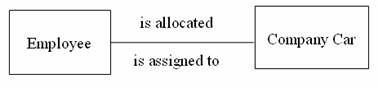
The degree of relationship is defined by the number of participating entities in a relationships.

* Binary: If two entities participate in a relationship, it is a binary relationship.
* Ternary: If three entities participate in a relationship, it is a ternary relationship.
* N-ary: If more than three entities participate in a relationship, it is a N-ary relationship.

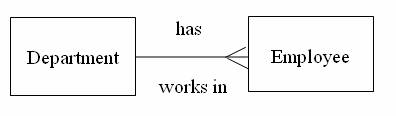
**9. List and give an example of the three types of binary relationships. Draw an E-R diagram for each.**

The three types of binary relationships are:

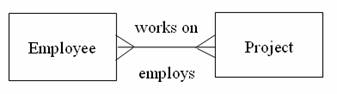
1. **one-to-one (1:1)**
   * This is where one occurrence of an entity relates to only one occurrence in another entity.
   * A one-to-one relationship rarely exists in practice, but it can. However, you may consider combining them into one entity.
   * For example, an employee is allocated a company car, which can only be driven by that employee.
   * Therefore, there is a one-to-one relationship between employee and company car.



1. **one-to-many (1:M)**
   * Is where one occurrence in an entity relates to many occurrences in another entity.
   * For example, taking the employee and department entities shown on the previous page, an employee works in one department but a department has many employees.
   * Therefore, there is a one-to-many relationship between department and employee.



1. **many-to-many (M:N)**
   * This is where many occurrences in an entity relate to many occurrences in another entity.
   * The normalisation process discussed earlier would prevent any such relationships but the definition is included here for completeness.
   * As with one-to-one relationships, many-to-many relationships rarely exist. Normally they occur because an entity has been missed.
   * For example, an employee may work on several projects at the same time and a project has a team of many employees.
   * Therefore, there is a many-to-many relationship between employee and project.



**10. Define the terms maximum cardinality and minimum cardinality.**

The **minimum number of instances** of one entity that may associated with each instance of another entity is known as **minimum cardinality.**

The **maximum number of instances** of one entity that may associated with each instance of another entity is known as **maximum cardinality.**

****

**11. Explain the distinctions among the terms primary key, candidate key and super key.**

Key is an attribute or collection of attributes that uniquely identifies an entity among entity set. For example, the roll\_number of a student makes him/her identifiable among students.

* **Super Key** − A set of attributes (one or more) that collectively identifies an entity in an entity set.
* **Candidate Key** − A minimal super key is called a candidate key. An entity set may have more than one candidate key.
* **Primary Key** − A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

**12. What are the main building modules of the entity relationship model? Discuss each one.**

The main building modules of the entity relationship model are:

* **Entities**, which represent people, places, items, events, or concepts. They are real life objects which are defined by a set of attributes.
* **Attributes**, which represent properties or descriptive qualities of an entity. These are also known as data elements.
* **Relationships**, which represent the link between different entities. They define the association of one entity with another entity.

**13. What is composite attributes, when it is used?**

**Composite attributes** are those attributes which can be broken down further into other simple or composite attributes. Eg: person\_name can be broken down into first\_name and last\_name.

They are used when:

- User needs to refer to an entire attribute on an occasion and only to a component of the attribute in some others.

Composite attributes are the ones that can be divided into subparts (that is, other attributes). For example, an attribute name could be structured as a composite attribute consisting of first name, middle initial, and last name. Using composite attributes in a design schema is a good choice if a user will wish to refer to an entire attribute on some occasions and to only a component of the attribute on other occasions. Suppose we were to add an address to the student entity-set. The address can be defined as the composite attribute address with the Attributes Street, city, state, and zip code. Composite attributes help us to group together related attributes, making the modeling cleaner.

**14. Explain the difference between single-value attributes and simple attributes.**

**15. Discuss the difference between a composite key and a composite attribute. How would each indicated in an E-R diagram?**

**16. What two courses of action are available to a designer when a multivalued attribute is encountered ?**

**17. Explain the various terms of an E-R model and how are they represented in an E-R model?**

**18. Explain the concept of dependent entities? Give example.**

**19. What is the difference total and partial participation? Explain.**

**20. What do you mean by mapping cardinalities ? explain various type of cardinalities.**

**21. What is the difference between single-value and multivalued attributes? Explain**

**22. Explain the concept of participation constraints.**

**23. Difference the binary relationship with ternary relationship with example.**

**24. Explain the difference between weak and strong entity set.**

**25. Define the components of extended E-R features.**

**26. Define the concept of aggregation. Give two examples of where this concept is useful.**

**27. Explain the distinction between disjoint and overlapping constraints.**

**28. Explain the distinction between total and partial constraints.**

**29. Write short notes on:**

**· Specialization**

**· Generalization**

**· Aggregation**