ST. XAVIER’S COLLEGE

**(Affiliated to Tribhuvan University)**

**Maitighar, Kathmandu**

****

**Database Management System**

**Theory Lab Assignment #5**

**SUBMITTED BY:**

**Siddhant Rimal**

**013BSCCSIT039**

**SUBMITTED TO**

|  |  |
| --- | --- |
| **Er. Sanjay Kr. Yadav**  **( Lecturer )** |  |
| **Department of Computer Science** | |

Submission Date: August 31st 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.

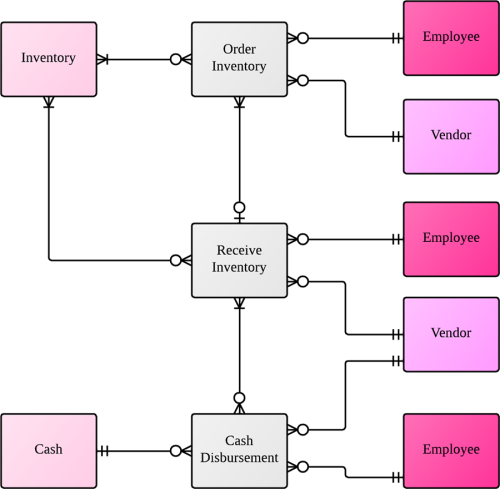


Figure 1 Entity Relationship Diagram

**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.**

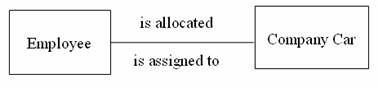
The degree of relationship is defined by the number of participating entities in a relationships.

* 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.

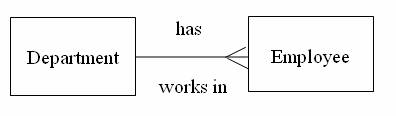
**9. List and give an example of the three types of binary relationships. Draw an E-R diagram for each.**

The three types of binary relationships are:

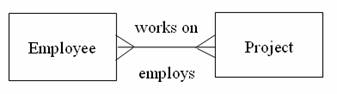
1. **one-to-one (1:1)**
   * This is where one occurrence of an entity relates to only one occurrence in another entity.
   * A one-to-one relationship rarely exists in practice, but it can. However, you may consider combining them into one entity.
   * For example, an employee is allocated a company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

- The user needs to find related information like number of clients living in the same locality during data modelling.

**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 ?**

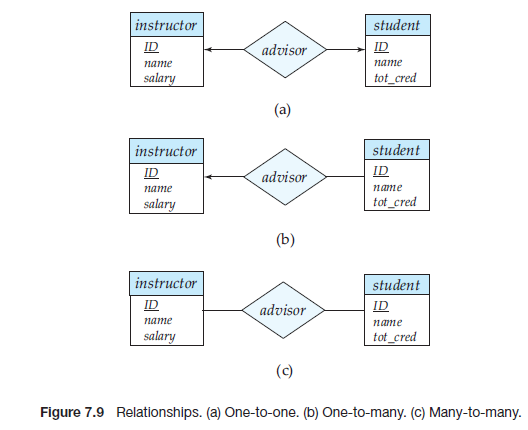
**17. Explain the various terms of an E-R model and how are they represented in an E-R model?**

**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.**

Mapping Cardinalities describes no. of entities to which another entity can be associated via relationship set. Mapping cardinalities are most useful in describing binary relationship sets but it can also describe relationship sets that involve more than two entity sets. For binary relationship set between entity set A and B mapping cardinality must one of the following.

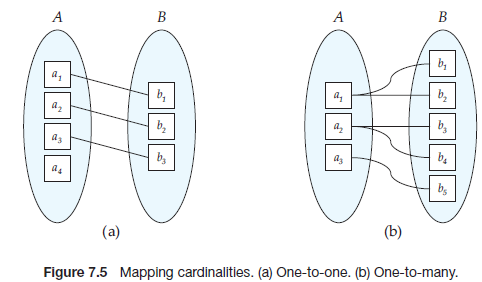
* One-to-one: We draw a directed line from the relationship set advisor to both entity sets instructor and student (see Figure 7.9a). This indicates that an instructor may advise at most one student, and a student may have at most one advisor.
* One-to-many: We draw a directed line from the relationship set advisor to the entity set instructor and an undirected line to the entity set student (see Figure 7.9b). This indicates that an instructor may advise many students, but a student may have at most one advisor.
* Many-to-one: We draw an undirected line from the relationship set advisor to the entity set instructor and a directed line to the entity set student. This indicates that an instructor may advise at most one student, but a student may have many advisors.
* Many-to-many: We draw an undirected line from the relationship set advisor to both entity sets instructor and student (see Figure 7.9c). This indicates that an instructor may advise many students, and a student may have many advisors. 

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

The attributes that have a single value for a particular entity is called single valued attribute. For example, the student ID attribute for a specific student entity refers to only one student ID.

There may be instances where an attribute has a set of values for a specific entity. Suppose we add to the instructor entity set a phone number attribute. An instructor may have zero, one, or several phone numbers, and different instructors may have different numbers of phones. This type of attribute is said to be multivalued. As another example, we could add to the instructor entity set an attribute dependent name listing all the dependents. This attribute would be multivalued, since any particular instructor may have zero, one, or more dependents.

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



The participation of an entity set E in a relationship set R is said to be total if every entity in E participates in at least one relationship in R. If only some entities in E participate in relationships in R, the participation of entity set E in relationship R is said to be partial. In Figure 7.5a, the participation of B in the relationship set is total while the participation of A in the relationship set is partial. In Figure 7.5b, the participation of both A and B in the relationship set are total. For example, we expect every student entity to be related to at least one instructor through the advisor relationship. Therefore the participation of student in the relationship set advisor is total. In contrast, an instructor need not advise any students. Hence, it is possible that only some of the instructor entities are related to the student entity set through the advisor relationship, and the participation of instructor in the advisor relationship set is therefore partial.

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

Relationship set that involves only two entity sets known as binary-relationship set. For example: depositor relationship set is a binary relationship set where relationship set involves only two entity set “customer” and “account”. Most relationship sets in database system are binary. However relationship set may involves in more than two entity sets.

Relationship set that involves three entity sets known as ternary relationship. For example: the relationship set “work-on” among employee, branch and job is example of ternary relationship. The no. of entity sets that participate in relationship set refers degree of relationship set. Here degree of ternary relationship is 3.

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

An entity set that does not have sufficient attributes to form a primary key is termed a weak entity set. An entity set that has a primary key is termed a 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.**

* Disjoint. A disjointness constraint is the one that requires that an entity belong to no more than one lower-level entity set. For example, student entity can satisfy only one condition for the student type attribute; an entity can be either a graduate student or an undergraduate student, but cannot be both.
* Overlapping. In overlapping constraints, the same entity may belong to more than one lower-level entity set within a single generalization. For an illustration, consider the employee work-team example, and assume that certain employees participate in more than one work team. A given employee may therefore appear in more than one of the team entity sets that are lower level entity sets of employee.

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

**29. Write short notes on:**

**· Specialization**

The process of designating subgroupings within an entity set is called specialization. Specialization follows top down design approach. Entity sets are subgroups in distinct entity sets. For example entity set person with attributes name, street and city can further subgroup into two entities sets customer and employee. Each of these person types can describes by set of attributes that includes all the attributes of entity set person plus all possible attributes of itself. For example, customer entity set can further described by set of attributes: customer\_id, enroll\_date etc. Similarly entity attributes can further describes by set of attributes: emplouee\_id, salary etc. The process of sub groupings within an entity set is called specialization. We can apply specialization repeatedly to refine a design schema. For instance bank employees may be further classified into officer, teller or secretary. In E-R diagram, specialization can be represented by a triangle component labeled ISA. The label ISA stands for “is a “. For example customer is a person, officer is an employee etc. The ISA relationship also called super class-subclass relationship.

Specialization stems from a single entity set; it emphasizes differences among entities within the set by creating distinct lower-level entity sets. These lower level entity sets may have attributes, or may participate in relationships, that do not apply to all the entities in the higher-level entity set. Indeed, the reason a designer applies specialization is to represent such distinctive features. If student and employee have exactly the same attributes as person entities, and participate in exactly the same relationships as person entities, there would be no need to specialize the person entity set.

**· Generalization**

Generalization is a containment relationship that exists between a higher-level entity set and one or more lower-level entity sets. Higher- and lower-level entity sets also may be designated by the terms superclass and subclass, respectively. For example, The person entity set is the superclass of the employee and student subclasses.

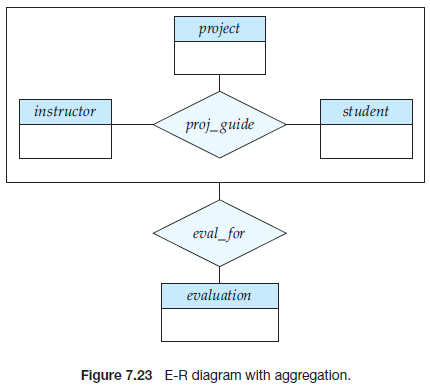
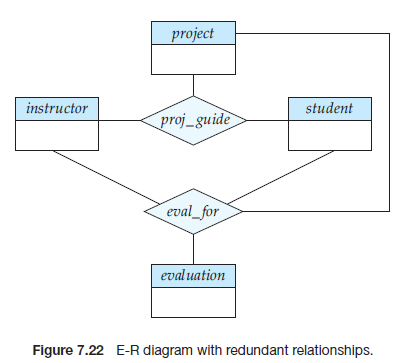
Generalization follows bottom-up approach in which multiple entity sets are synthesized into higher-level entity set on the basis of common features. For example, the database designer may have first identified a customer entity set with the attributes: name,street, city and customer\_id and employee entity set with the attributes name, street, city, employee\_id and salary. In both entities some attributes are common. These similarities between these two entities can be expressed by generalization. During the course of database design or E-R schema for enterprise database designer may use both specialization and generalization process. Specialization and generalization in E-R diagram represent by a same way. The terms specialization and generalization are used interchangeably.

Generalization proceeds from the recognition that a number of entity sets share some common features (namely, they are described by the same attributes and participate in the same relationship sets). On the basis of their commonalities, generalization synthesizes these entity sets into a single, higher-level entity set. Generalization is used to emphasize the similarities among lower-level entity sets and to hide the differences; it also permits an economy of representation in that shared attributes are not repeated.

**· Aggregation**

Aggregation is an abstraction through which relationships are treated as higher-level entities.

E-R model cannot express relationship among relationship. To illustrate this, let us consider quaternary relationship manages among employee, branch, job and manager. Its main job is to record managers who manages particular job/task perform by particular employee at particular branch. This quaternary relationship is required since binary relationship between manager and employee can not represent required information. This E-R diagram is able to represent the required information but information are redundant since every employee, branch and job exist both relationship set “work-on” and “manages”. Here aggregation is better to represent such information. Aggregation is in fact an abstraction it treats relationships as higher level entities. In our example, it treats relationship set work-on (including entity set employee, branch and job) as entity set. So now we can create binary relationship set “manages” between work-on and manager. This removes redundant information.



**REFERENCE**

1. “E-R Modelling”, Internet url: <http://www.guru99.com/er-modeling.html> 2015 [10/08/2015]
2. KL University, Lecture notes, Internet url: <http://www.kluniversity.in/elearn/materials/tdbqucsbsb77152TDBQUCSBSB.pdf> 2015 [10/08/2015]
3. Elmasri, Ramez, & Navathe, Shamkant B., Fundamentals of Database Systems, Second Edition, Addison-Wesley Publishing Company, 1994, ISBN 0-8053-1748-1, pages 450-452.