CHAPTER II

The Extended Entity Relationship Model.

The Entity_Relationship (ER) approach to Database design was originally proposed by Chen in 1976 [30], and has become more and more popular. Several conferences have been organized on the application of the ER model to database design and to software design in general. In 1988 ANSI chose the ER model as the standard model for Information Resource Dictionary System(IRDS). We consider below a version of the extended ER_Model which includes the additional data abstraction capabilities, such as generalization and full aggregation. We sketch below first the definitions of the basic ER_Constructs and then the additional concepts in subsequent sections.

2.1. basic Elements of the ER Model.

The basic concepts provided by the ER model are Entities, Relationships, and Attributes. We use in this work the terms entities and relationships to denote classes of objects.

Entities. Entities represent classes of real word objects. For example, in a personnel database system, PERSON, PROJECT, EMPLOYEE, CITY, and COMPANY are examples of entities. Entities are graphically represented by means of rectangles, as shown in Figure 2.1.

Relationships. Relationships represents associations between two or more entities. An example of a binary relationship in the personnel database is WORK_IN, which relates PERSON and the COMPANY in which they work. N_ary relationships connects more then two entities; for example the relationship USE in Fig. 2.2 is a ternary relationship that connects EMPLOYEE, SKILL, and PROJECT entities. Unary relationships are binary relationships connecting an entity to itself. For example, the relationship SPOUSE_OF in the Fig. 2.3 connects the Entity PERSON to itself. In order to distinguish between the two roles the entity has in the relationship, we associates two labels with the entity; in Fig.2.3 the two labels are MALE and FEMALE. Each relationship has a significant meaning; hence we need to select significant names for relationships. Relationships are represented by means of diamonds, as shown in Fig. 2.2, 2.3 and 2.4. Relationships are characterized in terms of mapping ratios known as cardinalities. Values for the cardinalities are either <u>one</u> or <u>many</u>. For a relationship among entities $E_1, E_2, \ldots, E_n$ a cardinality of "one" for entity E_i means that given all entities except E_i, there is at most one related entity of E_i. Fig.2.4 shows an example of a binary relationship with cardinalities.

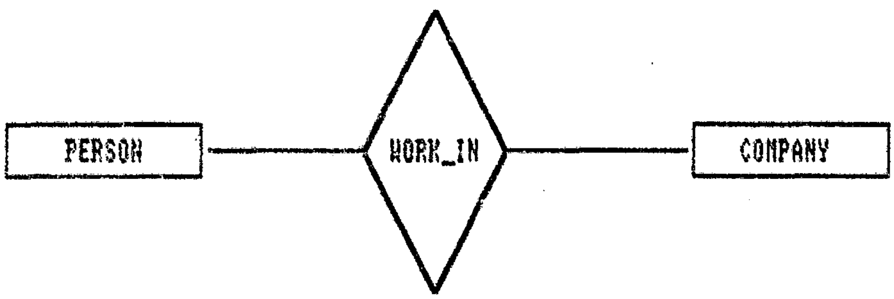

FIGURE 2.1

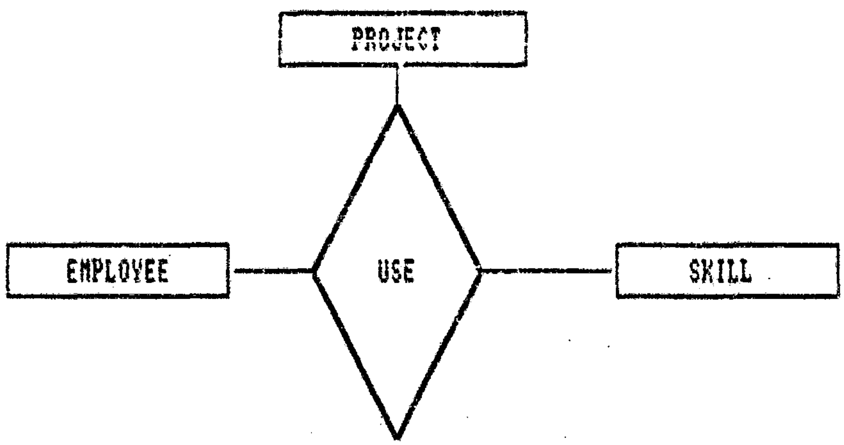

FIGURE 2.2 N-ARY RELATIONSHIPS

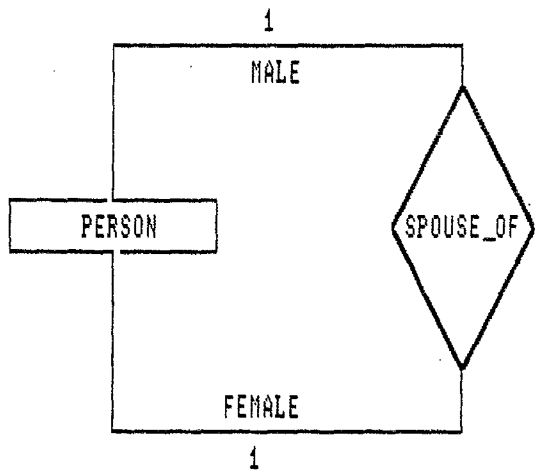

FIGURE 2.3 UNARY RELATIONSHIP

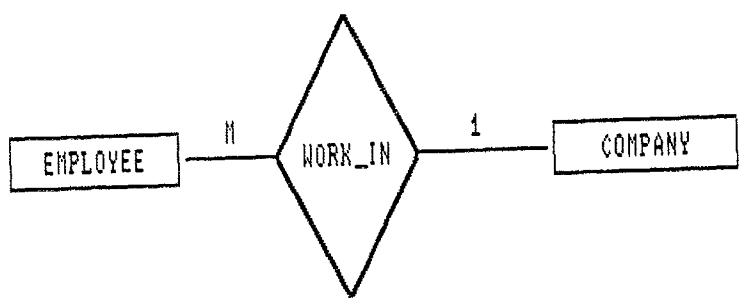

FIGURE 2.4

Attributes. Attributes represent elementary properties or characteristics of entities and relationships, which are used to describe and distinguish them from other objects in the same class. All the extensional information are carried out by attributes. For example, attributes of the entity PERSON are: Name, Social_security_Number, Profession, Date_of_birth; Attributes of CITY are Name and Population. Each attribute is associated to a particular Domain, that is the set of legal values for that attribute, and the Domain of values is known as the Value set. Domain declaration are similar to type declaration in programming languages. Like relationships Attributes are characterized by the types of mappings from the objects (Entities or relationships) to the value set of the attribute. The mapping can be one_to_one, one_to_many, many_to_one and many_to_many. If the mapping is many_to_many or one_to_many then the corresponding attribute is known as Multivalued Attribute. Attributes are generally represented by means of oval or circles. Without any loss of any generality, a second kind of representation can also be used to represent an Entity or a relationship with its set of attributes as shown in Fig. 2.5. This representation is mostly used in this work.

[EMP_ID,name,address,age,salary,....]

key = [EMP_ID]

FIGURE 2.5

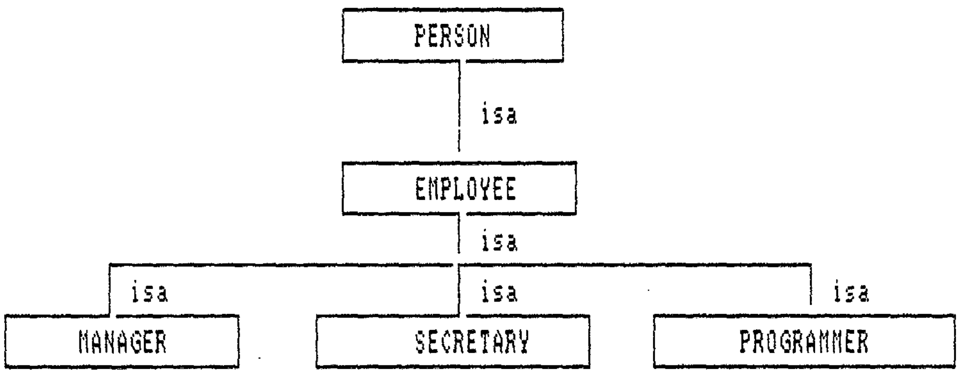

FIGURE 2.6 Generalization Hierarchy for entity Person

2.2 Other elements of ER Model.

Other elements of ER model include the generalization hierarchies, subsets, identifiers, and weak entities.

Generalization hierarchies An entity E is a generalization of a group of entities E_1, E_2,....,E_n if each object of classes E_1, E_2,....,E_n is also an objects of class E. The entity E is known as the generic entity and the entities E_i (i=1,2,...,n) are known as specialized (or subset) entities. The diagrammatic representation of generalization hierarchies is shown in Fig. 2.6. In this work we restrict the generalization hierarchies to specialized entities having a single generic source. Fig. 2.6 presents an example of a generalization hierarchies for the entity PERSON.

In a generalization hierarchy, all the properties of the generic entity are inherited by all of its specialized entities. in terms of the ER model, this means that every attribute and relationship defined for the generic entity are automatically inherited by all subset entities in the generalization. This property is an important one, because it allows building structured generalization hierarchies.

Consider Figure 2.6. The inherited property states that the attributes Name and addresses of PERSON are also attributes of Employee, Manager, Secretary, and Programmer; thus they can be eliminated from the subset entities, and similarly the attribute salary can also be eliminated from the entities Manager, Programmer and Secretary; thus simplifying the schema.

Subset. A subset is a particular case of the generalization hierarchy having just one specialized entity.

Identifiers. An identifier of an entity E is a collection of attributes having the property of uniquely determining all the instances of E. This is also defined as a minimal set of attributes K of an entity E which defines a one_to_one mapping from E to the cartèsian product of the value sets of the attributes of K. From a terminological view point, identifiers are sometimes referred to as keys or candidate keys. There may exist more then one candidate key for an entity E and in such cases, we designate one of the keys as the Primary key of the entity E. Digramatically, the key attributes are represented by capital letters in the list of attributes for entities as shown in Figure 2.5.

Weak Entities. When an entity can not have its own attributes to uniquely identify each instance of it, then the entity depends for its identification (ID_Dependent) on other entities, and is called a weak entity. Digramatically weak entities are represented by double rectangles. For the sake of simplicity, we allow only weak entities having a single external key, i.e. each weak entity dependent on a single strong entity for its identification.

2.3 Extended concepts for EER model

The EER model is extended with the following concepts

Abstraction hierarchies : The first type of hierarchy is the isa hierarchy (or subset hierarchy) [120]. The isa_hierarchy " A isa B " implies that something that is true for the generic entity B must also be true for the Specific entity A. For each occurrence of A, there is one and only once corresponding occurrence of B, for each occurrence of B there may or may not be one corresponding occurrence of A. Any attribute that are common to a set of entities in an isa hierarchy should be stored at the highest level generic entity and deleted from the specific entities below it (unless they are a part of a key). The specific entities can simply inherit these attributes from the generic entities inorder to minimize the amount of redundancy that appears in a design.

The second type of hierarchies are known as Generalization hierarchies. When the isa relationship involves only two entities, it is called as a subtype (or specialization) hierarchy. A generalization hierarchy occurs when there are two or more specific entity types that partition the generic entity type [120]. For example, if an Employee must be either a manager, Secretary or Programmer , then the three relationships "manager isa Employee", "Secretary isa Employee", and "Programmer isa Employee" form a generalization hierarchy.

Aggregation : An aggregation is a set of attributes that can be referred to as a single property [120]. In the basic ER model, the aggregation constructs take three forms_(i) Aggregation of collection of attributes into an entity, (ii) Aggregation of collection of attributes and keys of several existing entities into a weak entity, and (iii) Aggregation of two or more entities into a relationship. The ER model falls short of the capability of aggregation by disallowing relationship to be aggregated further. What is needed in order to provide this capability is to simply allow relationships to associate both entities and relationships, rather then only entities. Fig. 2.7 shows an example of a binary aggregation.

2.4 EER DIAGRAMS

EER schema are expressible in a diagramatic form called an EER_diagram, in which entities, relationships and

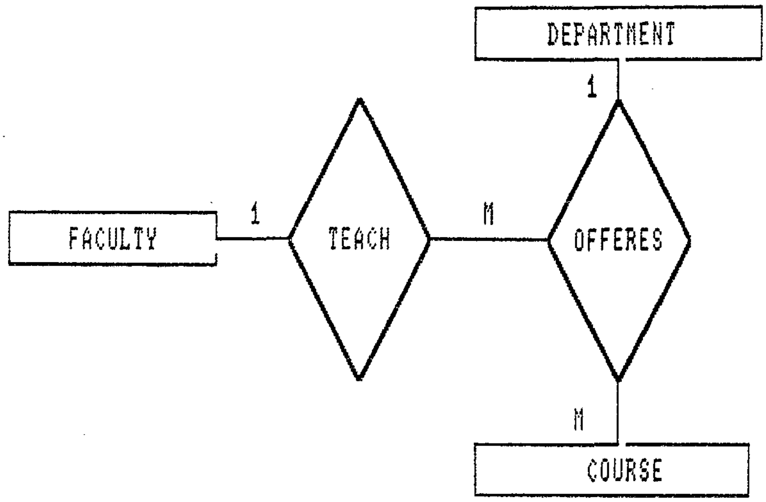

FIGURE 2.7 BINARY AGGREGATION

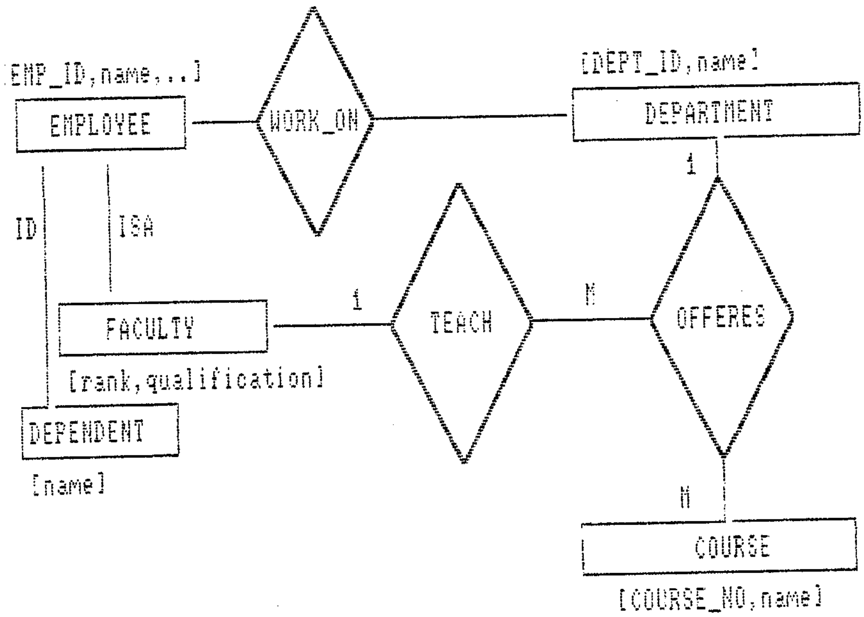

FIGURE 2.8 EERD FOR DEPARTMENT VIEW

attributes are represented respectively by rectangle, diamond and ellipse shaped vertices, and every vertex is labelled by the name of the concept it represents. Formally, we will define here an ERD as a certain type of labelled, directed graphs EG = (N,E), where the nodes in N are objects, and the edges in E are arrows from boxes to boxes, or from diamonds to boxes or from diamonds to diamonds. The directonality of edges in such a diagram allows the explicit representatin not only of the interaction of EER elements but also of their mutual existence dependencies. Thus in the EER diagram, there are directed edges: (i) from relationship to object they associate, labelled by the corresponding cardinalities (1 on M), and possibly by Roles; ii) From weak entity to the entity on which they depend for identification, labelled by ID; iii) From sepcialised entity to their direct generic source entity labelled by ISA.

Example: The EERD represented by figure 2.8 has five entities two of which are "Weak entities" FACULTY and DEPENDENT, and three relationships shown as diamonds plus the "ISA" and "ID" relationships. DEPENDENT can only be fully identified with EMP_ID and NAME in its key; FACULTY is an ISA (Subset) entity of Employee and uses EMP_ID as its key. WORK_ON is a many_to_one (M:1) relationship between entities employee and DEPARTMENT: TEACH is a 1:M relationship between the entity

FACULTY and the relationship OFFER; OFFER is a 1:M relationship between the entities DEPARTMENT and COURSE. The relationship OFFER is an aggregations of two entities DEPARTMENT and COURSE.

2.5 EERD Representation in PROLOG

To develop the expert system VMITS, we choose the Prolog language. In our system an EERD is represented in the Prolog-database in the form of logical assertions of predicates acting on objects as follows:-

1. Predicates associated to entities

    Ent ("View", "Entity")

    Key ("View", "Entity", "Key-attribute-list")

    alt ("View", "Entity", "Alternative-key-attribute-list")

    att ("View", "Entity" "Non-key-attribute-list")

The predicate ent (V,E) indicates that E is an entity in the view V.

The predicate key (V,E,K) indicates that K is the key of the entity E in view V.

The predicate att (V,E,K) indicates that K is the set of non-key attributes of entity E in view V.

The predicate alt (V,E,K) indicates that K is an alternative key of the entity E in view V.

2. Predicates associated to relationships

rel ("View", "relationship")

att ("View", ""relationship", relationship-attributes")

edg ("View", "relationship", "Entity", "Role", "Cardinality")

The predicate rel (V,R) indicates that R is a relationship in view V.

The predicate att (V,R,A) indicates that A is the set of the relationship attributes of the relationship R in view V.

The predicate edge (V,R,O,RL,C) indicates that the object O (Relationship or entity) plays the role RL in the relationship R in view V and has cardinality C (1 or N).

3. Predicates associated to generalization and specialization hierarchies

Edge (V,E1, E2, "ISA","")

Which indicates that the entity E1 is a subset of the entity E2 in view V.

4. Predicates associated to weak entities (ID-relationships)

Edge (V, E1, E2, "ID", "")

which indicates that the entity E1 is ID-dependent on the entity E2 in view V.

```
ent ("Dept-view", "EMPLOYEE")
ent ("Dept-view", "DEPENDENT")
ent ("Dept-view", "FACULTY")
ent ("Dept-view", "DEPARTMENT")
ent ("Dept-view", "COURSE")
rel ("Dept-view", "WORK_ON")
rel ("Dept-view", "TEACH")
key ("Dept-view", "EMPLOYEE", ["EMP_ID"])
Key ("Dept-view", "COURSE", ["COURSE_NO"])
key ("Dept-view", "DEPARTMENT",["DEPT_ID"])
att ("Dept-view", "EMPLOYEE", ["name", "address", "salary"])
att ("Dept-view", "FACULTY", ["rank", "qualification"])
att ("Dept-view", "DEPARTMENT", ["name"])
att ("Dept-view", "COURSE", ["name"])
edge("Dept-view", "DEPENDENT", "EMPLOYEE", "ID","")
edge("Dept-view,  "FACULTY", "EMPLOYEE", "ISA","")
edge("Dept-view,  "WORK_ON, "DEPARTMENT","","")
edge("Dept-view,  "WORK_ON", "DEPARTMENT","","")
edge("Dept-view,  "OFFER", "COURSE", "","M")
edge("Dept-view", "TEACH", "OFFER", "","m")
edge("Dept-view", "TEACH", "FACULTY", ", " 1")
```

Fig. 2.9 Prolog representation of the EERD for DEPARTMENT view.

5. Predicates associated to aggregations

Edge (V, R1, R2, RL, C)

which indicates that the relationship R2 is an aggregation in view v.

Example: The EERD of the figure 2.8 can be represented in PROLOG using these predicates as shown in Fig 2.9 and can be stored in a file with '.dba' extension, which can be loaded to the prolog's system database for further use.