BACHELOR OF COMPUTER APPLICATIONS

SEMESTER 3

DCA2102

DATABASE MANAGEMENT SYSTEM

Unit 3

Database Models and Implementation

Table of Contents

1. INTRODUCTION

A data model is an integrated collection of concepts for describing data, relationships between data, and constraints on the data. A data modelallows us to treat a database as an abstract machine. In other words, wecan concentrate on the principles of design divorced from an immediate concern with implementation. Data models constitute formal languages for defining data structures declaring integrity and for manipulating data. A datamodel is a mechanism for specifying the schema of some database. Data models establish the principles underlying DBMS. Every database, andindeed every DBMS, must adhere to the principles of some data model. However, the term data model is somewhat ambiguous. In the database literature, the term is used in a number of different senses, two of which are the most important: that of architecture for data, and that of an integrated set of data requirements.

1.1 Objectives:

After completing Unit 3, you should be able to:

- *Define the concepts of the Relational Model and its implementations*
- *Understand the concepts of other Data Models like Hierarchical Model, Network Data Model, Object/Relational Model, Object-Oriented Model, and the Associative Database Model.*
- *Understand the Entity-Relationship Model and its applications*
- *Differentiate between data models*

2. DATA MODEL AND TYPES OF DATA MODELS

The term data model is used to refer to a set of general principles for handling data. Here, people talk of the relational data model, the hierarchical data model, or the object-oriented data model.

This set of principles that define a data model may be divided into three major parts:

- Data definition – a set of principles concerned with how data isstructured.
- Data manipulation – a set of principles concerned with how data isoperated upon.
- Data integrity – a set of principles concerned with determining whichstates are valid for a database.

Data definition involves defining an organization for data, a set of templates for the organization of data. Data manipulation concerns the process of howthe data is accessed, and how it is changed in the database. Data integrityis very much linked with the idea of data manipulation, in the sense that integrity concerns the idea of what are valid changes and invalid changes todata.

We may make a distinction between three generations of architectural data model:

- **Primitive data models:** In this approach, objects are represented by record structures grouped in file structures. The main operations available are read and write operations over records
- **Classic data models:** These are the hierarchical, network, and relational data models. The hierarchical data model is an extension of the primitive data model discussed above. The network is an extension of the hierarchical approach. The relational data model is a fundamental departure from the hierarchical and network approaches.
- **Semantic data models:** The main problem with classic data models, such as the relational data model, is that they maintain a fundamental record orientation.

2.1 Relational Data Model

The relational model today, is the primary data model for commercial data-processing applications. It has attained its primary position because of its simplicity, which eases the job of the programmer, as compared to earlier data models such as the network model or the hierarchical model.

A database based on the relational model developed by E.F. Codd allows the definition of data structures, storage and retrieval operations, and integrity constraints. In such a database the data and relations between them are organized in tables. A table is a collection of records and each record in a table contains the same fields. Properties of Relational Tables:

- Values Are Atomic
- Each Row is Unique
- Column Values Are of the Same Kind
- The Sequence of Columns is Insignificant
- The Sequence of Rows is Insignificant
- Each Column has a unique Name

In the relational data model, the database is represented as a group of related tables. The relational data model was introduced in 1970. It iscurrently the most popular model. The mathematical simplicity, and ease of visualization of the relational data model, have contributed to its success. The relational data model is based on the mathematics of set theory, whosebasic components are the following.

Relation: A two-dimensional table. A relation is a collection of tuples, each of which contains values for a fixed number of attributes. Relations are sometimes referred to as at files, because of their resemblance to an unstructured sequence of records. Each tuple in a relation must be unique; that is, there can be no duplicates.

A Relation may be defined in multiple ways. The **Relation Schema** R, denoted by *R* (A1, A2, An), is made up of a relation name R and is a list of **attributes** A1, A2, An.

For Example - CUSTOMER (Cust-id, Cust-name, Address, Phone#). Here, CUSTOMER is a relation defined over the four attributes Cust-id, Cust- name, Address, Phone#, each of which has a **domain** or a set of valid values.

A **tuple** is an ordered set of values. Each value is derived from an appropriate domain. Each row in the CUSTOMER table may be referred to as a tuple in the table and would consist of four values.

<1759, "Rama Krishna", "101 Main 3rd Cross Manipal", "0820-2653487">is a tuple belonging to the CUSTOMER relation.

A relation may be regarded as a ***set of tuples*** (rows). Columns in a table are also called attributes of the relation.

A domain has a logical definition, for example, “India_Phone_numbers” is the set of 10 digit mobile numbers. A domain may have a data type or a format defined for it. The India_Phone_numbers may have a format: (ddd)- ddddddd where each d is a decimal digit. E.g., Dates have various formats such as a month, name, date, year or yyyy-mm-dd, or dd,mm,yyyy, etc.

An attribute designates the **role** played by the domain. E.g., the domainDate may be used to define attributes “Invoice-date” and “Payment-date”. The relation is formed over the cartesian product of the sets; each set has values from a domain; that domain is used in a specific role which isconveyed by the attribute name. For example, attribute Cust-name is defined over the domain of strings of 25 characters. The role these strings play in the CUSTOMER relation is that of the name of the customers.

Given R(A1, A2, , An)

r(R) ⊂ dom (A1) X dom (A2) X X dom(An)

Where R is a schema of the relation and r(R) is a specific "value" orpopulation of R. Here R is also called the **intension** of a relation and r is also called the **extension** of a relation.

Let S1 and S2 be domains, S1 = {0,1} and S2 = {a,b,c}. The Relation R canbe written as

R ⊂ S1 X S2

r(R) = {<0,a> , <0,b> , <1,c> }

is one possible “state” or “population” or “extension” r(R), defined over domains S1 and S2. It has three tuples.

Characteristics of Relations

- **Ordering of tuples in a relation r(R):** The tuples are not considered to be ordered, even though they appear to be in the tabular form.

- **Ordering of attributes in a relation schema R (and of values within each tuple):** We will consider the attributes in R(A1, A2, ..., An) and the values in t=<v1, v2, ..., vn> to be ordered.
- **Values in a tuple:** All values are considered *atomic* (indivisible). A special null value is used to represent values that are unknown orinapplicable to certain tuples.

Relational Integrity Constraints

Constraints are *conditions* that must hold on to *all* valid relation instances.There are three main types of constraints:

- Key constraints
- Entity integrity constraints
- Referential integrity constraints

Key Constraints

Superkey of R: A set of attributes SK of R such that no two tuples *in any valid relation instance r(R)* will have the same value for SK. That is, for any distinct tuples t1 and t2 in r(R), t1[SK] ≠ t2[SK].

Key of R: A "minimal" superkey; that is, a superkey K such that removal of any attribute from K results in a set of attributes that is not a superkey.

For Example: Consider a CAR relation schema:

CAR(State, Reg#, SerialNo, Make, Model, Year)

It has two keys Key1 = {State, Reg#}, Key2 = {SerialNo}, which are also superkeys. {SerialNo, Make} is a superkey but *not* a key.

If a relation has *several* candidate keys, one is chosen arbitrarily to be theprimary key. The primary key attributes are *underlined*.

Entity Integrity

Relational Database Schema: A set S of relation schemas that belong tothe same database. S is the *name* of the database.

S = {R1, R2, ..., Rn}

Entity Integrity: The *primary key attributes* PK of each relation schemaR in S cannot have null values in any tuple of r(R). This is because primary key values are used to *identify* the individual tuples.

t[PK] ≠ null for any tuple t in r(R)

Referential Integrity

A constraint involving *two* relations (the previous constraints involve a *single*relation). Used to specify a *relationship* among tuples in two relations, the referencing relation, and the referenced relation.

Tuples in the *referencing relation* R1 have attributes FK (called foreign key attributes) that reference the primary key attributes PK of the *referenced relation* R2. A tuple t1 in R1 is said to reference a tuple t2 in R2 if t1[FK] = t2[PK].

A referential integrity constraint can be displayed in a relational database schema as a directed arc from R1.FK to R2.

Referential Integrity Constraints

The value in the foreign key column (or columns) FK of the referencing relation R1 can be either a value of an existing primary key value of the corresponding primary key PK in the referenced relation R2 or a null.

Attribute: A table column. Other commonly used terms for attribute are property and field. The set of permissible values for each attribute is called the domain for that attribute.

Tuple: A table row. A tuple is an instance of an entity or relationship or whatever is represented by the relation.

Key: A single attribute or combination of attributes whose values uniquely identify the tuples of the relation. That is, each row has a different value for the key attribute(s). The relational model requires that every relation have a key and that:

- No two tuples may have the same key value and
- Every tuple must have a value for the key attribute (the key fields have non-null values).

There are two restrictions on the relational model that are sometimes circumvented in practice:

- Duplicate tuples are not permitted. If two tuples are entered with the same value for each and every attribute, they are considered to be the same tuple.
- No ordering of tuples within a relation is assumed. In practice, however, one method or another of ordering tuples is often used.

One of the main advantages of the relational model is that it is conceptually simple and more importantly based on the mathematical theory of relation. It also frees the users from details of the storage structure and access methods. The relational model like all other models consists of three basic components:

- a set of domains
- a set of relations operation on relations
- integrity rules

In this unit, we first provide the formal definition of a relational data model. The basic operations of relational algebra are discussed in Unit 4.

In general, we say that a relation defined over n domains has a degree n or is n-ary. The elements of this set are n-tuples. We shall distinguish between the definition of a relation and the relation itself. We shall say that the definition of a relation gives a name to the relation and specifies the components over which it is defined. These components are referred to as relation attributes or attributes for short. An attribute has a domain associated with it from which it takes on values. The relation itself, on the other hand, is the set of tuples which constitute it at a given instance of time.

For example, a statement which says that a relation Supplier is built over attributes S#, P#, SCITY having domains integer, character string respectively is the definition of the relation **Supplier**. The relation itself is shown below. It must be noted at the time the definition of a relation is just given, a relation with no tuples in it, i.e. a null relation is just given, a relation with no tuples in it, i.e. a null relation, is created.

Supplier

S#	P#	SCITY
10	1	BANGALORE
10	2	BANGALORE
10	3	BANGALORE
11	1	BOMBAY
11	2	BOMBAY

2.2 Hierarchical Model

The hierarchical data model organizes data in a tree structure. There is a hierarchy of parent and child data segments. This structure implies that a record can have repeating information, generally in the child data segments.Data is in a series of records, which has a set of field values attached to it. Itcollects all the instances of a specific record together as a record type. These record types are the equivalent of tables in the relational model, with the individual records being the equivalent of rows. To create links between these record types, the hierarchical model uses Parent-Child Relationships. These are a 1:N mapping between record types. For example, an organization might store information about an employee, such as name, employee number, department, and salary. The organization might also store information about an employee's children, such as name and date of birth. The employee and children data form a hierarchy, where the employee data represents the parent segment and the children data represents the child segment. If an employee has three children, then there would be three child segments associated with one employee segment. In a hierarchical database, the parent-child relationship is one to many. This restricts a child segment to having only one parent segment. Hierarchical DBMSs were popular from the late 1960s, with the introduction of IBM's InformationManagement System (IMS) DBMS, through the 1970s.

2.3 Network Data Model

The popularity of the network data model coincided with the popularity of thehierarchical data model. Some data were more naturally modeled with morethan one parent per child. So, the network model permitted the modeling of many-to-many relationships in data. In 1971, the Conference on Data Systems Languages (CODASYL) formally defined the network model. The basic data modeling constructs in the network model is the set construct. A set consists of an owner record type, a set name, and a member record type. A member record type can have that role in more than one set; hence the multi-parent concept is supported. An owner record type can also be a member or owner in another set. The data model is a simple network, and link and intersection record types may exist, as well as sets between them. Thus, the complete network of relationships is represented by several pair-wise sets; in each set, some (one) record type is the owner (at the tail of the network arrow) and one or more record types are members (at the head of the relationship arrow). Usually, a set defines a 1:M relationship, although 1:1 is permitted. The CODASYL network model is based on mathematical set theory.

2.4 Object/Relational Model

Object/relational database management systems (ORDBMSs) add new object storage capabilities to the relational systems at the core of modern information systems. These new facilities integrate the management of traditional fielded data, complex objects such as time-series and geospatial data, and diverse binary media such as audio, video, images, and applets.By encapsulating methods with data structures, and ORDBMS server can execute complex analytical and data manipulation operations to search and transform multimedia and other complex objects.

As an evolutionary technology, the object/relational (OR) approach hasinherited the robust transaction and performance-management features of its relational ancestor and the flexibility of its object-oriented cousin.

Database designers can work with familiar tabular structures and data definition languages (DDLs) while assimilating new object-management possibilities. Query and procedural languages and call interfaces in ORDBMSs are familiar: SQL3, vendor procedural languages, and ODBC, JDBC, and proprietary call interfaces are all extensions of RDBMS languages and

interfaces. And the leading vendors are, of course, quite wellknown: IBM, Inform ix, and Oracle.

2.5 Object-Oriented Model

Object DBMSs add database functionality to object programming languages. They bring much more than persistent storage of programming language objects. Object DBMSs extend the semantics of the C++, Smalltalk, and Java object programming languages to provide full-featured database programming capability, while retaining native language compatibility. A major benefit of this approach is the unification of the application and database development into a seamless data model and language environment. As a result, applications require less code, use more natural data modeling, and code bases are easier to maintain. Object developers can write complete database applications with a modest amountof additional effort.

The object-oriented database (OODB) paradigm is the combination of object-oriented programming language (OOPL) systems and persistent systems. The power of the OODB comes from the seamless treatment of both persistent data, as found in databases, and transient data, as found in executing programs.

In contrast to a relational DBMS where a complex data structure must be flattened out to fit into tables or joined together from those tables to form thein-memory structure, object DBMSs have no performance overhead to storeor retrieve a web or hierarchy of interrelated objects. This one-to-one mapping of object programming language objects to database objects has two benefits over other storage approaches: it provides higher performance management of objects, and it enables better management of the complex interrelationships between objects. This makes object DBMSs better suited to support applications such as financial portfolio risk analysis systems, telecommunications service applications, WWW (World Wide Web) document structures, design and manufacturing systems, and hospital patient record systems, which have complex relationships between data.

Self-Assessment Questions – 1

1. _____is a set of principles concerned with determining whichstates are valid for a database.
2. In Primitive data models approach, objects are represented by _____structures grouped in file-structures.
3. In the_____data model the database is represented as agroup of related tables.
4. Each tuple in a relation must be_____; that is, there can be no duplicates.
5. A tuple is an_____set of values.
6. In relational integrity constraints, there are main types ofconstraints.
7. If a relation has several candidate keys, one is chosen arbitrarily to bethe _____key.
8. The set of permissible values for each attribute is called the _____for that attribute.
9. A single attribute or combination of attributes whose values uniquely identify the_____of the relation.
10. The hierarchical data model organizes data in a _____ structure.
11. In a hierarchical database the parent-child relationship is _____ .
12. The popularity of the network data model coincided with the popularityof the data model.
13. Object DBMSs add database functionality to_____programming languages.
14. A major benefit of Object- Oriented data model approach is the _____of the application.

3. ENTITY-RELATIONSHIP MODEL

There are three basic notions that the E-R data model employs: entity sets, relationship sets, and attributes. Consider an example COMPANY Database,the COMPANY is organized into DEPARTMENTs. Each department has a name, number, and employee who *manages* the department. We keep track of the start date of the department manager. Each department *controls*a number of PROJECTs. Each project has a name, number and is located at a single location.

We store each EMPLOYEE's social security number, address, salary, sex, and birth date. Each employee *works for* one department but may *work on* several projects.

We keep track of the number of hours per week that an employee currently works on each project. We also keep track of the *direct supervisor* of each employee.

Each employee may *have* a number of DEPENDENTs. For eachdependent, we keep track of their name, sex, birth date, and relationship to the employee.

An **entity** is a "thing" or object in the real world that is distinguishable fromall other objects. For example, each employee in an enterprise or companyis an entity. An entity has a set of properties, and the values for some set of properties may uniquely identify an entity.

An **entity set** is a set of entities of the same type that share the same properties or attributes. **Attributes** are properties used to describe an entity.For example, an EMPLOYEE entity may have a Name, SSN, Address, Sex, BirthDate. A specific entity will have a value for each of its attributes. For example a specific employee entity may have Name='John Smith', SSN='123456789', Address ='731, Fondren, Houston, TX', Sex='M', BirthDate='09-JAN-55'. Each attribute has a *value set* (or data type) associated with it – e.g. integer, string, subrange, enumerated type.

The attribute used in the E-R model can be characterized by the following attribute types.

- Simple
  - Each entity has a single atomic value for the attribute. For example, SSN or Sex.
- Composite
  - The attribute may be composed of several components. For example, Address (Apt#, House#, Street, City, State, ZipCode, Country) or Name (FirstName,

MiddleName, LastName). The composition may form a hierarchy where some components are themselves composite.

- Single valued
  - An entity has a single value for that attribute. For example, the loan- number in the loan entity.
- Multi-valued
  - An entity may have multiple values for that attribute. For example, the Color of a CAR or the Previous Degrees of a STUDENT. Denoted as {Color} or {Previous Degrees}.
- Null attribute
  - A null value is used when an attribute does not have a value or valueexists but not known at present. For an instance, an employee has no dependent. In another example where the task completed status is not known at present, the value for this attribute can be kept as null.
- Derived attribute
  - The value for this type of attribute can be derived from the values of other related attributes or entities. For example - value of the attribute "age" can be derived if we know the value of attribute "dob" (date of birth) as age= current_date – dob. Another example – value of the attribute "loe" (length of employment) can be derived if know the value of attribute "doj" (date of joining) as loe= current_date –doj.
  - For instance, account number and balance. Here balance attribute isderived from the account number attribute.

An attribute of an entity type for which each entity must have a unique value, is called a **key attribute** of the entity type. For example, SSN is the key attribute of the EMPLOYEE relation schema. A key attribute may be composite. For example, Vehicle Tag Number is a key of the CAR entity type with components (Number, State).

3.1 Modeling using E-R Diagrams

A model is an abstract form of any system or process that hides the unnecessary details while highlighting those details important to the application. We have noticed the model of huge campuses or buildings which help to visualize the structure before they are built. On similar lines, we can also model our software applications before they are developed. Modeling the databases using E-R diagrams is called as E-R modeling. Thistechnique is also called as Top-Down approach because one need not identify all the attributes to model the system using this technique.

Steps in E-R Modeling

Usually, the following six steps are followed to generate E-R Models.

- **Identify the entities:** Look for general nouns in the requirement specification document which are of business interest to business users.
- **Find relationships:** Identify the natural relationship and their cardinalities between the entities.
- **Identify the key attributes for every entity:** Identify the attribute or set of attributes which can identify the instance of the entity uniquely.
- **Identify other relevant attributes:** Identify other attributes which are of interest to business users and which they want to store the informationin the database.
- **Complete E-R diagram:** Draw a complete E-R diagram with all attributes, including the primary key.
- **Review your results with your business users:** Look at the list of attributes associated with each entity to see if anything has been omitted.

3.2 Notation used in E-R Model

An entity-relationship model is a tool for analyzing the semantic features of an application that are independent of events. This approach includes a graphical notation, which depicts entity classes as rectangles, relationships as diamonds, and attributes as circles or ovals. For complex situations, a partial entity-relationship diagram may be used to present a summary of the entities and relationships that do not include the details of the attributes.

The entity-relationship diagram provides a convenient method for visualizing the interrelationships among entities in a given application. This tool has proven to be useful in

making the transition from an information application description to a formal database schema. The entity-relationship model is used for describing the conceptual schema of an enterprise without attention to the efficiency of the physical database design. The entity-relationship diagrams are then turned into a logical schema in which the database is actually implemented. Figure 3.0 shown below shows the diagram used for entity-relationship.

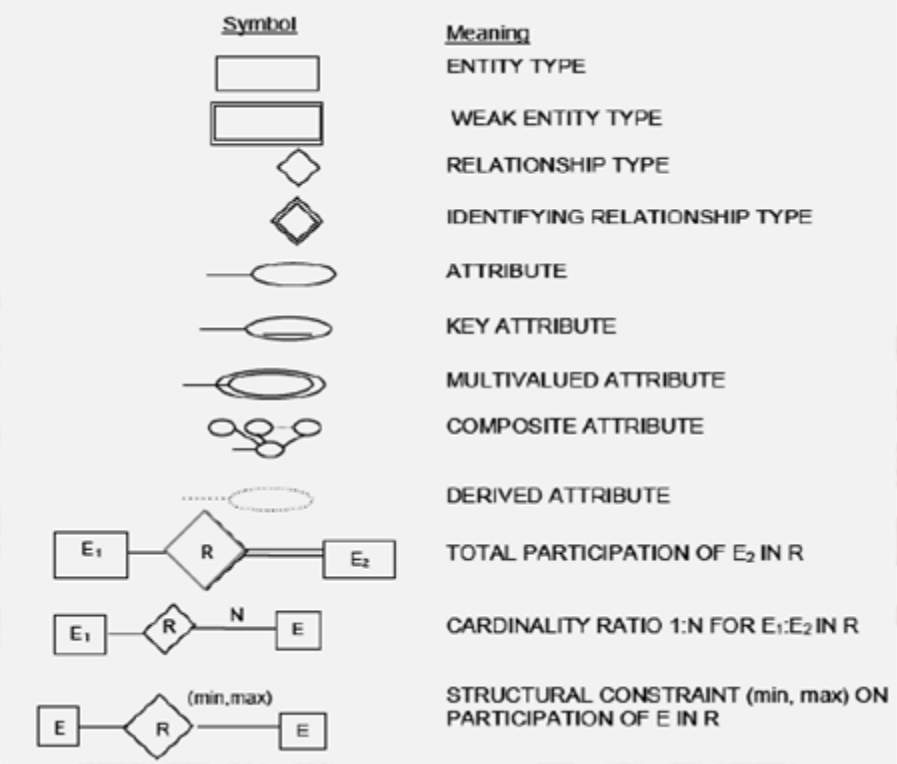


Fig 3.0: Diagram used in entity-relationship.

3.3 Relationships and Relationship Types

A ***relationship*** is an association among several entities. A ***relationship set*** is a set of the same type. Formally, it is a mathematical relation on n ≥ 2 entity sets. If E_1, E_2,……. E_n are entity sets, then a relationship set R is a subset of

$\{ (e_1, e_2, \ldots\ldots e_n) \mid e_1 \varepsilon E_1, e_2 \varepsilon E_2 , \ldots..e_n \varepsilon E_n \}$

where $(e_1, e_2, \ldots\ldots e_n)$ is a relationship.

A relationship relates to two or more distinct entities with a specific meaning. For example, EMPLOYEE John Smith works on the Product X PROJECT orEMPLOYEE Franklin manages the Research DEPARTMENT. Relationships of the same type are grouped or typed into a relationship type. For example,the WORKS_ON relationship type in which EMPLOYEEs and

PROJECTs participate, or the MANAGES relationship type in which EMPLOYEEs and DEPARTMENTs participate. The degree of a relationship type is the number of participating entity types. Both MANAGES and WORKS_ON are binary relationships. More than one relationship type can exist with the same participating entity types. For example, MANAGES and WORKS_FOR are distinct relationships between EMPLOYEE and DEPARTMENT, but with different meanings and different relationship instances.

Figure 3.1 shows an example relationship of the WORKS_FOR relationship between EMPLOYEE and DEPARTMENT

EMPLOYEE WORKS_FOR DEPARTMENT

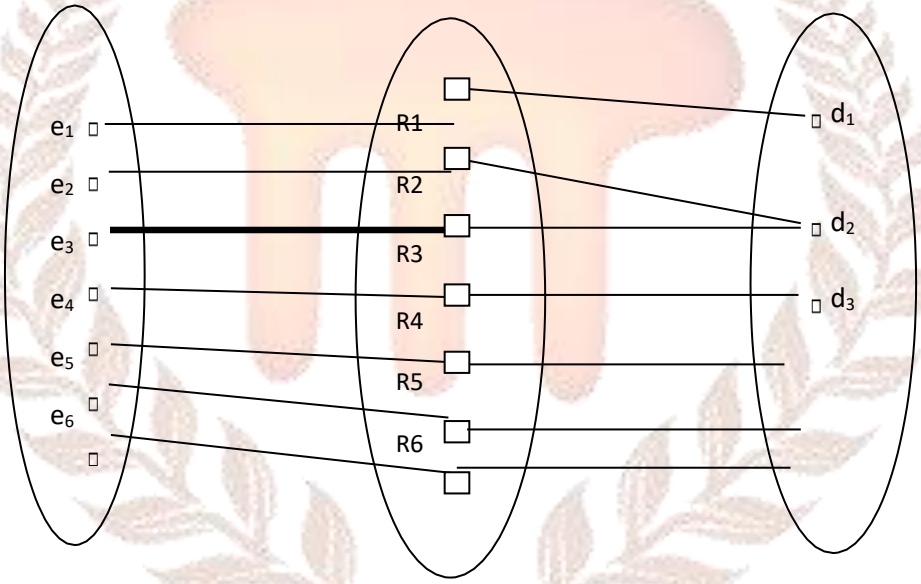


Fig 3.1: Relationship of WORKS_FOR relationship

Weak Entity Types

An entity that does not have a key attribute is called weak entity types. A weak entity must participate in an identifying relationship type with an owner, or identifying entity type. Entities are identified by the combination of:

- A partial key of the weak entity type
- The particular entity they are related to in the identifying entity type

Example:

Suppose that a DEPENDENT entity is identified by the dependent's first name and birthdate, *and* the specific EMPLOYEE that the dependent is related to. DEPENDENT is a weak entity type with EMPLOYEE as its identifying entity type via the identifying relationship type DEPENDENT_OF

Relationships Types

An E-R enterprise schema may define certain constraints to which the contents of a database must conform. Mapping cardinalities, or cardinality ratios, express the number of entities to which another entity can be associated via a relationship set.

For a binary relationship set R between entity sets A and B, the mapping cardinality must be one of the following:

- **One-to-one (1:1):** An entity in A is associated with at most one entity in B, and an entity B is associated with at most one entity in A. See Figure 3.2 (a).
- **One-to-many (1:N):** An entity in A is associated with any number of entities in B. An entity in B, however, can be associated with at most one entity in A. See Figure 3.2 (b).
- **Many-to-one (N:1):** An entity in A is associated with at most one entityin B. An entity in B, however, can be associated with any number of entities in A. See Figure 3.2 (c).
- **Many-to-many (M:N):** An entity in A is associated with any number of entities in B. An entity in B is associated with any number of entities in A. See Figure 3.2 (d).

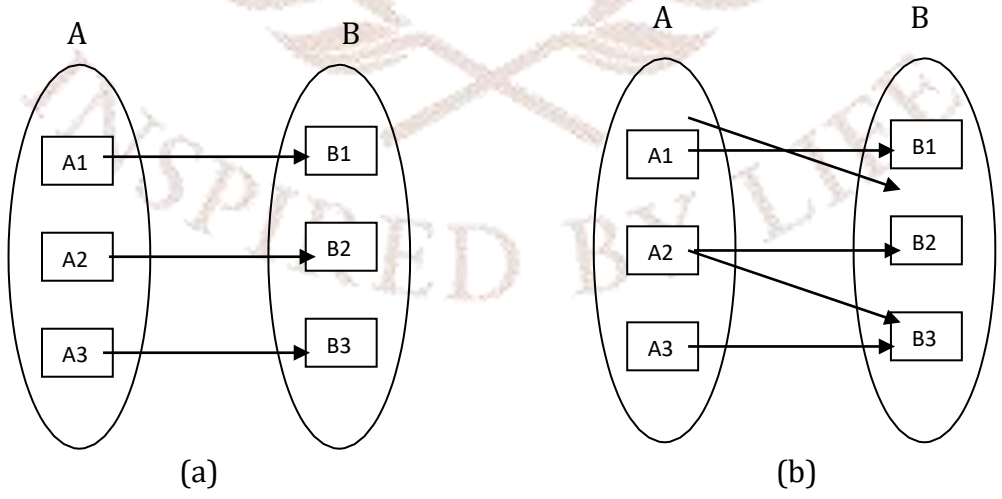
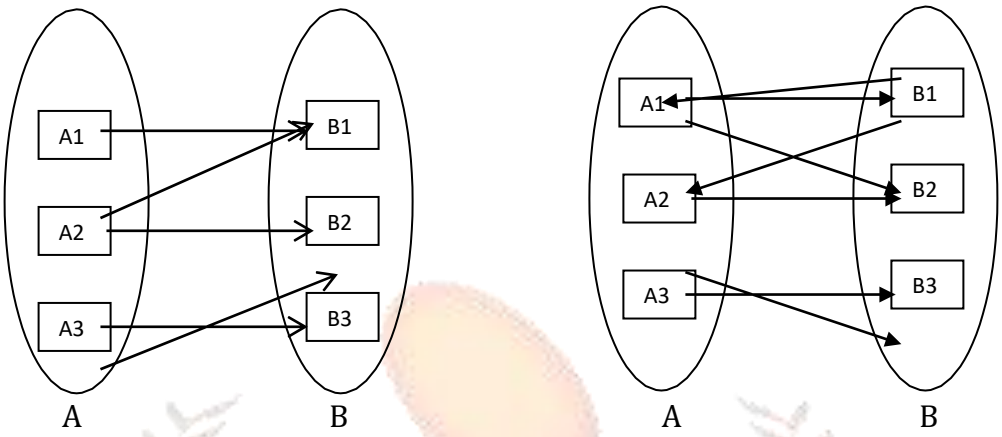


Fig 3.2: Relationships types a) One-to-One b) One-to-many

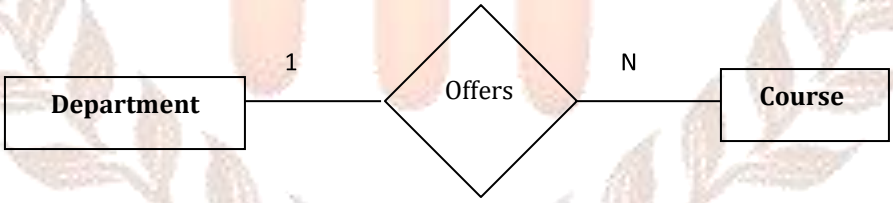


(c) (d)

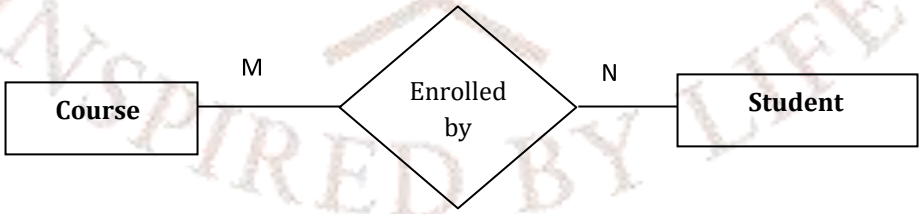
Fig 3.3: Relationships types c) Many-to-One d) Many-to-many

Examples for Relationships

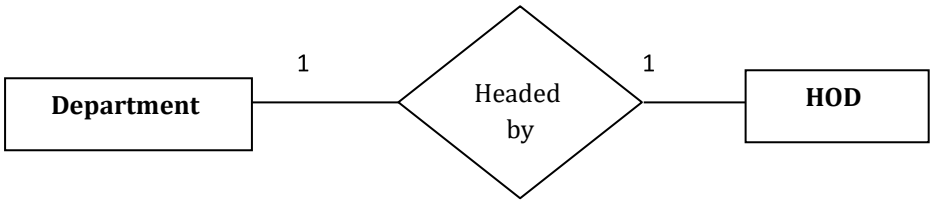
1. The department offers multiple courses and each course belongs to only one department, hence cardinality between department and course is **one to many**.



2. One course is taken up by multiple students and one student enrolls for multiple courses, hence the relationship is **many-to-many**.



3. Each department has one "Head of Department" (HOD), hence relationis **one-to-one**.



Self-Assessment Questions – 2

15. There are_____basic notions that E-R data model employs.
16. An_____is a "thing" or object in the real world that is distinguishable from all other objects.
17. An entity set is a set of entities of the same type that share the same properties or_____.
18. Attributes are properties used to describe an_____.
19. Each entity has a single atomic value for the attribute is called _____ attribute.
20. Address is an example for_____attribute.
21. The value for_____type of attribute can be derived from thevalues of other related attributes or entities.
22. An attribute of an entity type for which each entity must have a unique value is called a_____of the entity type.
23. One of the following is a demerit of the ER modeling.
    a. Gives a higher level abstraction of the system.
    b. Can be generalized and specialized based on needs.
    c. Physical design derived from E-R Model may have some amount of ambiguities or inconsistency.
    d. Intuitive and helps in physical database creation
24. How many basics are there in E-R data model?
    a. Three
    b. Four
    c. Five
    d. Six
25. An_____is a "thing" or object in the real world that is distinguishable from all other objects.
    a. Relation
    b. Entity
    c. Attribute
    d. Simple attribute

26. Pick out the composite attribute from the list of attributes
    a. Sex
    b. Address
    c. SSN
    d. Department number
27. 'Color of the car and degrees of students' are examples for the_____.
    a. Null attribute
    b. Derived attribute
    c. Single valued
    d. Multi valued
28. Identifying the natural relationship and their cardinalities between the entities is a step of_____.
    a. Identify the entities
    b. Find relationships
    c. Identify the key attributes for every entity
    d. Identify other relevant attributes
29. ER diagram includes a graphical notation, which depicts entity classesas _____.
    a. Rectangles
    b. Ovals
    c. Diamonds
    d. Circles
30. A_is an association among several entities.
    a. Relationship
    b. Key
    c. Partial key
    d. Entity
31. An entity that does not have a key attribute is called
    a. Weak entity types
    b. Entity Types
    c. Null attribute
    d. Derived attribute

4. ASSOCIATIVE DATABASE MODEL

The Associative model was devised by Simon Williams of Lazy Software and is said to be built upon current research with some unique additions. If you are familiar with applying XML to data you will feel comfortable with most of the underlying concepts. Both XML and Associative databases havea common route in Semantic Databases and Topic Maps. Most of the terms and concepts used will be found in references to Binary Relational Databases.

An Associative database has two fundamental data structures. There is aset of "Items" and a set of "Links" that connect them together. In the "Item" structure entries have a unique identifier, a name, and a type. Each entry in the "links" structure also has a unique identifier together with the identifiers for the relevant "source", "verb" and "target" (the subject, verb, and object from our sentences). This can be illustrated with the following two diagrams.For clarity, the question of item type has been ignored in this illustration.

Items

Identifier	Name
12	Red
41	Is a
76	Colour
14	Mary
81	Vegetarian
43	Eats
82	Plants
15	Ski Lessons
39	Start at
83	08:00
42	On
85	Sunday

Links

Identifier	Source	Verb	Target
101	12	41	76
103	14	41	81
124	81	43	82
105	15	39	83
107	105	42	85

The last entry in the Links structure (107) shows how another entry (identifier 105) has become the Source for that entry. The two entries show how you could store the "ski lesson" sentence within the Links structure. Readers with some familiarity with Microsoft Access will have seen the "AutoNumber" facility, that can be used to create a unique numeric identifierfor any row in a given table. Here we see an identifier being assigned on a database-wide basis. The number itself has no significance, it is simply required to be unique.

The Associative model structure is economical with storage space as there is no need to hold available "spaces" for data that is not available, even if itis a normal part of a given data set. This contrasts with relational databases.A relational database stores a minimum of a single "null" byte for missing data items in any given row. Some relational databases reserve the maximum space for a given column in every row. The Associative database makes the storage of "custom" data for different users, or for other varying needs, straightforward and "inexpensive" in terms of maintenance or network resources. If there is a need to store different data about, say, different customers or customer groups in different countries, then an Associative database can manage this more efficiently than a relational database.

The Associative model differentiates between what it calls Entities andAssociations. An entity is defined as being discrete and having an independent existence. An association is something that depends upon one or more other things. An example may help with this. A person or companyis an entity while a supplier or a customer or an employee is association – their existence depends upon the role being played at any one time. Indeed it is possible for an Entity to have multiple business roles simultaneously, each being recorded as an association. If circumstances change, one or more of the associations may die away but the entity would endure. The difference may seem a little moot at first but is designed to simplify rather than complicate the data model.

An Associative database is comprised of a number of "chapters" and a user's view of the content of a database is controlled by his or her "Profile".A Profile is a list of Chapters. The database designer consigns the various elements to specific Chapters, and the user Profile restricts access to the relevant Chapters for a given user. If some links exist between items in chapters inside and outside of a particular user Profile, then those links are not visible to that user. The combination of Chapters and Profiles cansimplify the tailoring of the database

to particular users or subject groups. Data that is relevant to one user group could be invisible to another, and indeed may be replaced by an alternate data set.

The concept of a record is missing from the Associative model. To assemble all of the current information on something as complex as (say) a sales order, the data storage will need to be re-visited many times. This is a potential disadvantage, although it should be recognized that a well a normalized relational database would probably also require a number of data store reads to establish a similar data set. Some rough calculations based upon a small personal sample would suggest that the Associative database would require more than four times as many data reads as a relational database. If the process of reading a sequence of links can be optimized, then this may go some way to minimizing the difference as experienced by the user.

Self-Assessment Questions – 3

32. An Associative database has _____ fundamental data structures.
33. In the "Item" structure entries have a _____ identifier, a name and a type.
34. A relational database stores a minimum of a single_____byte formissing data items in any given row.
35. A_____is a list of Chapters.
36. The combination of Chapters and Profiles can simplify the_____ofthe database to particular users.
37. The concept of a_____is missing from the Associative model.

5. SUMMARY

In this unit, we reviewed three major traditional data models used in current DBMSs. These three models are primitive, classic and semantic data models. In the Primitive data, models approach objects are represented by record structures grouped in file structures. The main operations available are read and write operations over records. Classic data models are the hierarchical, network and relational data models.

The hierarchical data model is an extension of the primitive data model discussed above. The network is an extension of the hierarchical approach. The relational data model is a fundamental departure from the hierarchical and network approaches. The main problem with classic data models such as the relational data model is that they maintain a fundamental record- orientation.

The hierarchical model evolved from the file-based system. It uses tree-type data structure to represent the relationship among records. The hierarchical data model restricts each record type to only the parent record type. Eachparent record type can have any number of children record types.

In a network model, one child record may have more than one parent node. A network can be converted into one or more trees by introducing redundant nodes.

The relational model is based on a collection of tables. A table is also called a relation. A tree or network structure can be converted into a relational structure by separating each node in the data structure into a relation.

The entity-relationship diagrams are useful in representing the relationship among entities. They help in logical database design. We have also presented implementation schemes of each of the traditional database models.

6. TERMINAL QUESTIONS

1. Distinguish between three major types of the architectural data model.
2. Describe the differences in meaning between the terms *relation* and *relation schema*.
3. Explain the distinctions among the terms primary key, candidate key, and superkey.
4. Construct an E-R diagram for a car-insurance company whose customers own one or more cars each. Each car has associated with it zero to any number of recorded accidents.
5. Construct an E-R diagram for a hospital with a set of patients and a set of medical doctors. Associate with each patient a log of the various tests and examinations conducted.
6. Explain the difference between a weak and a strong entity set.
7. We can convert any weak entity set to a strong entity set by simply adding appropriate attributes. Why, then, do we have weak entity sets? Self Assessment Questions

7. ANSWERS

Self-Assessment Questions

1. Data integrity
2. Record
3. Relational
4. Unique
5. Ordered
6. Three
7. Primary
8. Domain
9. Tuples
10. Tree
11. one to many
12. hierarchical
13. object
14. unification
15. Three
16. Entity
17. Attributes
18. Entity
19. Simple
20. Composite
21. Derived
22. key attribute
23. Physical design derived from E-R Model may have some amount ofambiguities or inconsistencies.
24. Three
25. Entity
26. Address
27. Multi-valued
28. Find relationships
29. Rectangles

30. Relationship
31. Week-entity types
32. Two
33. Unique
34. "null"
35. Profile
36. Tailoring
37. Record

Terminal Questions

1. In terms of three generations, the architectural data models are:
   (i) Primitive data models (ii) Classic data models (iii) Semantic data models. These three data models can be distinguished in terms ofobject representation, hierarchical network formation, and record-orientation. (Refer section 2)
2. A relation is a collection of tuples, each of which contains values for a fixed number of attributes whereas the Relation Schema R, denoted by*R* (A1, A2, ... An), is made up of a relation name R and is a list of attributes A1, A2, An. (Refer section 2.1)
3. A different set of attributes which are able to identify any row in thedatabase is known as super key. And minimal super key is termed as candidate key i.e. among set of super keys one with minimum number ofattributes. If a relation has *several* candidate keys, one is chosen arbitrarily to be the primary key. (Refer section 2.1)
4. To construct E-R diagram, identify car as entities, find naturalrelationships between car and customer, identify the key attributes for every entity, identify other relevant attributes related to car, customer, car-insurance to complete E-R diagram. Also review results. (Refer section 3.1 for detail)
5. Follow the same steps as shown in Q. No. 4 and refer section 3.1 for detail)
6. A strong entity is independent of other entities and can exist on its own. A weak entity is dependent on one or more other entities in order for it toexist. (Refer section 3.3 for detail)
7. A weak entity is required because it is dependent on one or more other entities in order for it to exist and it is often prevalent in database. (Refersection 3.3 for detail)