Course Contents

Unit-01: Database Concepts and Architecture	(4 Hrs.)
Unit-02: Data Modeling Using the Entity-Relational Model	(5 Hrs.)
Unit-03: The Relational Data Model and Relational Database Constraints	(5 Hrs.)
Unit-04: SQL	(10 Hrs.)
Unit-05: Relational Database Design	(7 Hrs.)
Unit-06: Transaction Processing and Concurrency Control, and Recovery	(8 Hrs.)
Unit-07: Database Recovery Techniques	(3 Hrs.)
Unit-08: NoSQL	(3 Hrs.)

Course Contents

Unit-02: Data Modeling Using the Entity-Relational Model (5 Hrs.)

Using High-Level Conceptual Data Models for Database Design; Entity Types, Entity Sets, Attributes, and Keys; Relationship Types, Relationship Sets, Roles, and Structural Constraints; Weak Entity Types; ER Diagrams, Naming Conventions, and Design Issues; Relationship Types of Degree Higher Than Two; Subclasses, Superclasses, and Inheritance; Specialization and Generalization; Constraints and Characteristics of Specialization and Generalization

Course Contents

Introduction of Conceptual Data Models for Database Design

Conceptual modeling is a very important phase in designing a successful database application. Database application refers to a particular database and the associated programs that implement the database queries and updates.

Major part of the database application will require the design, implementation, and testing of these application programs.

Traditionally, the design and testing of application programs has been considered to be part of software engineering rather than database design.

In many software design tools, the database design methodologies and software engineering methodologies are intertwined since these activities are strongly related.

Course Contents

Using High-Level Conceptual Data Models for Database Design

In this chapter, we discuss the approach of concentrating on the database structures and constraints during conceptual database design.

We present the modeling concepts of the entity–relationship (ER) model, which is a popular high-level conceptual data model. This model and its variations are frequently used for the conceptual design of database applications, and many database design tools employ its concepts.

We describe the basic data-structuring concepts and constraints of the ER model and discuss their use in the design of conceptual schemas for database applications. The diagrammatic notation associated with the ER model is known as ER diagrams.

Course Contents

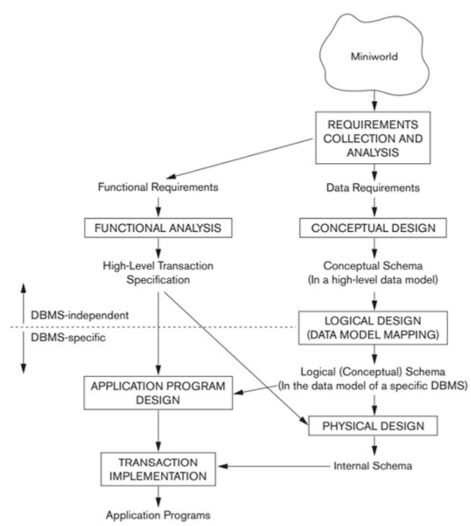
Using High-Level Conceptual Data Models for Database Design

Object modeling methodologies such as the Unified Modeling Language (UML) are becoming increasingly popular in both database and software design. These methodologies go beyond database design to specify detailed design of software modules and their interactions using various types of diagrams.

Course Contents

Using High-Level Conceptual Data Models for Database Design

A simplified diagram to illustrate the main phases of database design.



Course Contents

Using High-Level Conceptual Data Models for Database Design

Step-1: Create a conceptual schema using a high-level conceptual data model

The first step is requirements collection and analysis. During this step, the database designers interview prospective database users to understand and document their data requirements. The result of this step is a concisely written set of users' requirements. These requirements should be specified in as detailed and complete a form as possible. In parallel, designer also specifying the data requirements useful to specify the known functional requirements like user defined operations, retrievals and updates that will be applied to the database.

Once the requirements have been collected and analyzed, the next step is to create a conceptual schema for the database, using a high-level conceptual data model.

Course Contents

Using High-Level Conceptual Data Models for Database Design

The high-level conceptual schema can also be used as a reference to ensure that all users' data requirements are met. This approach enables database designers to concentrate on specifying the properties of the data, without being concerned with storage and implementation details and create a good conceptual database design.

During or after the conceptual schema design, the basic data model operations can be used to specify the high-level user queries and operations identified during functional analysis. This also serves to confirm that the conceptual schema meets all the identified functional requirements. Modifications to the conceptual schema can be introduced if some functional requirements cannot be specified using the initial schema.

Course Contents

Using High-Level Conceptual Data Models for Database Design

Step-2: Logical design or data model mapping

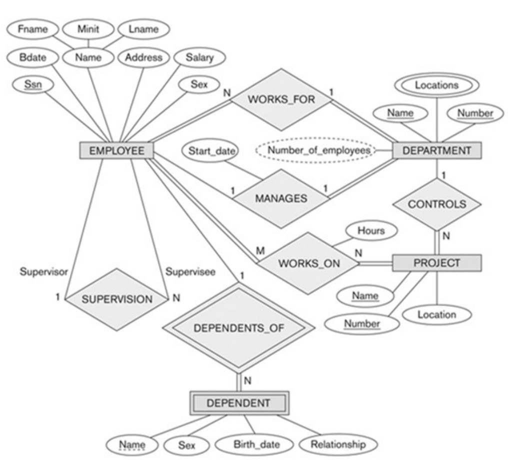
This is the actual implementation of the database, using a commercial DBMS such as the relational (SQL) model. So the conceptual schema is transformed from the high-level data model into the implementation data model. This step is called high-level data model; its result is a database schema in the implementation data model of the DBMS.

Step-3: Physical Design and Application Program

In this physical design phase, the internal storage structures, file organizations, indexes, access paths, and physical design parameters for the database files are specified. In parallel with these activities, application programs are designed and implemented as database transactions corresponding to the high-level transaction specifications.

Course Contents

Using High-Level Conceptual Data Models for Database Design



E-R diagram using High-level Data Model (Conceptual Design)

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Logical Design or data model mapping

Course Contents

Using High-Level Conceptual Data Models for Database Design

Database Design?

Database Design steps are:

1. Conceptual Design
2. Logical Design
3. Normalization
4. Physical Design

Course Contents

Using High-Level Conceptual Data Models for Database Design

1. Conceptual design

Data modelling is the first step in the process of database design and also called conceptual design. The aim of this phase is to describe:

- The data contained in the database (e.g., entities: students, lecturers, courses, subjects)
- The relationships between data items (e.g., students are supervised by lecturers; lecturers teach courses)
- The constraints on data (e.g., student number has exactly eight digits; a subject has four or six units of credit only)

Course Contents

Using High-Level Conceptual Data Models for Database Design

2. Logical Design

- The logical phase of database design is also called the data modeling mapping phase. This phase gives us a result of relation schemas. The basis for these schemas is the ER or the Class Diagram.
- To create the relation schemas is mainly mechanical operation. There are rules for transferring the ER model or class diagram to relation schemas.

3. Normalization

- Normalization in database design is a way to change the relation schema to reduce any superfluity. With every normalization phase, a new table is added to the database.

Course Contents

Using High-Level Conceptual Data Models for Database Design

4. Physical Design

- In the physical design phase, choose appropriate DBMS software to implement the database design.
- DBMS software (SQL clauses) are used in creating the database including the indexes and the integrity constraints (rules).
- Finally, the data is added and the database can finally be tested.

A Sample Database Application

A sample database application, called COMPANY, which serves to illustrate the basic ER model concepts and their use in schema design. We list the data requirements for the database and then create its conceptual schema step-by-step using the modeling concepts of the ER model.

The COMPANY database keeps track of a company's employees, departments, and projects. After the requirements collection and analysis phase, the database designers provide the following description of the miniworld—the part of the company that will be represented in the database.

- The company is organized into departments. Each department has a unique name, a unique number, and a particular employee who manages the department. We keep track of the start date when that employee began managing the department. A department may have several locations.
- A department controls a number of projects, each of which has a unique name, a unique number, and a single location.

A Sample Database Application

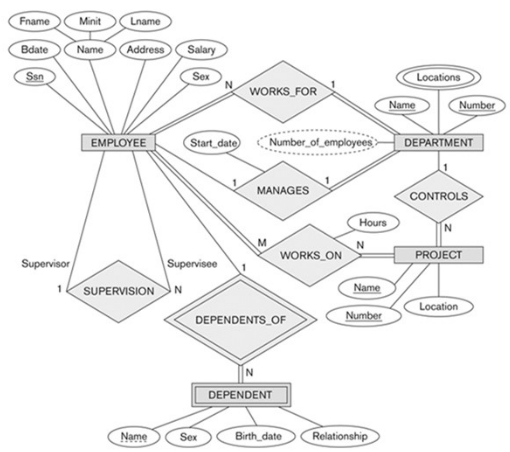
- The database will store each employee's name, Social Security number, address, salary, sex (gender), and birth date. An employee is assigned to one department, but may work on several projects, which are not necessarily controlled by the same department. It is required to keep track of the current number of hours per week that an employee works on each project, as well as the direct supervisor of each employee (who is another employee).
- The database will keep track of the dependents of each employee for insurance purposes, including each dependent's first name, sex, birth date, and relationship to the employee.

Figure below shows how the schema for this database application can be displayed by means of the graphical notation known as ER diagrams. We will learn how to prepare ER diagram using E-R model and the step-by-step process of deriving the schema from the stated requirements.

We will learn the ER diagrammatic notation—as we introduce the ER model concepts.

A Sample Database Application

Using High-Level Conceptual Data Models for Database Design



E-R diagram using High-level Data Model (Conceptual Design)

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Logical Design or data model mapping

Course Contents

Entity Types, Entity Sets, Attributes, and Keys:

The ER model describes data as entities, relationships, and attributes. We introduce the concepts of entities and their attributes and further define entity types and key attributes.

We draw the initial conceptual design of the entity types for the COMPANY database.

Entities and Attributes: The basic concept that the ER model represents is an entity. Entity is a thing or object in the real world with an independent existence. An entity may be an object with a physical existence (for example, a particular person, car, house, or employee) or it may be an object with a conceptual existence (for instance, a company, a job, or a university course).

Each entity has attributes-the particular properties that describe it. For example, an EMPLOYEE entity may be described by the employee's name, age, address, salary, and job.

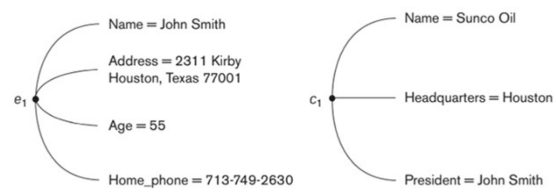
Course Contents

Entity Types, Entity Sets, Attributes, and Keys:

Entities and Attributes: A particular entity will have a value for each of its attributes. The attribute values that describe each entity become a major part of the data stored in the database.

The EMPLOYEE entity e1 has four attributes: Name, Address, Age, and Home_phone; their values are 'John Smith,' '2311 Kirby, Houston, Texas 77001', '55', and '713-749-2630', respectively.

The COMPANY entity c1 has three attributes: Name, Headquarters, and President; their values are 'Sunco Oil', 'Houston', and 'John Smith', respectively.



Course Contents

Entity Types, Entity Sets, Attributes, and Keys:

Entities and Attributes:

Several types of attributes occur in the ER model:

- Simple versus Composite,
- Single valued versus multivalued,
- Stored versus derived.
- NULL value for an attribute.

We define these attribute types and illustrate their use via examples.

Entity Types, Entity Sets, Attributes, and Keys:

Entities and Attributes:

Types of attributes:

- Simple versus Composite,
- Single valued versus multivalued,
- Stored versus derived.

Concept of NULL value for an attribute.

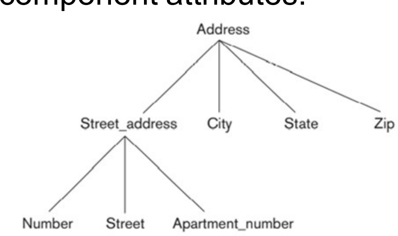
Entity Types, Entity Sets, Attributes, and Keys:

Entities and Attributes:

Composite versus Simple (Atomic) Attributes: Composite attributes can be divided into smaller subparts, which represent more basic attributes with independent meanings. For example, the Address attribute of the EMPLOYEE entity can be subdivided into Street_address, City, State, and Zip with the values ‘2311 Kirby’, ‘Houston’, ‘Texas’, and ‘77001’.

Attributes that are not divisible are called simple or atomic attributes.

Composite attributes can form a hierarchy; for example, Street_address can be further subdivided into three simple component attributes: Number, Street, and Apartment_number.



Entity Types, Entity Sets, Attributes, and Keys:

Single-Valued versus Multivalued Attributes: Most attributes have a single value for a particular entity; such attributes are called single-valued. For example,Age is a single-valued attribute of a person.

In some cases an attribute can have a set of values for the same entity- for instance, a Colors attribute for a car, or a College_degrees attribute for a person. Cars with one color have a single value, whereas two-tone cars have two color values. Similarly, one person may not have any college degrees, another person may have one, and a third person may have two or more degrees. Such attributes are called multivalued.

A multivalued attribute may have lower and upper bounds to constrain the number of values allowed for each individual entity. For example, the Colors attribute of a car may be restricted to have between one and two values, if we assume that a car can have two colors at most.

Entity Types, Entity Sets, Attributes, and Keys:

Stored versus Derived Attributes: In some cases, two (or more) attribute values are related- for example, the Age and Birth_date attributes of a person.

For a particular person entity, the value of Age can be determined from the current (today's) date and the value of that person's Birth_date. The Age attribute is hence called a derived attribute and is said to be derivable from the Birth_date attribute, which is called a stored attribute.

Some attribute values can be derived from related entities; for example, an attribute Number_of_employees of a DEPARTMENT entity can be derived by counting the number of employees related to (working for) that department.

Entity Types, Entity Sets, Attributes, and Keys:

NULL Values: In some cases, a particular entity may not have an applicable value for an attribute. For example, a College_degrees attribute applies only to entity people with college degrees. For such situations, a special value called NULL is created.

NULL can also be used if we do not know the value of an attribute for a particular entity- for example, if we do not know the home phone number of ‘John Smith’

Entity Types, Entity Sets, Attributes, and Keys:

Complex Attributes: Composite and multivalued attributes can be nested arbitrarily. We can represent arbitrary nesting by grouping components of a composite attribute between parentheses ( ) and separating the components with commas, and by displaying multivalued attributes between braces { }. Such attributes are called complex attributes. For example, if a person can have more than one residence and each residence can have a single address and multiple phones, an attribute Address_phone for a person can be specified as

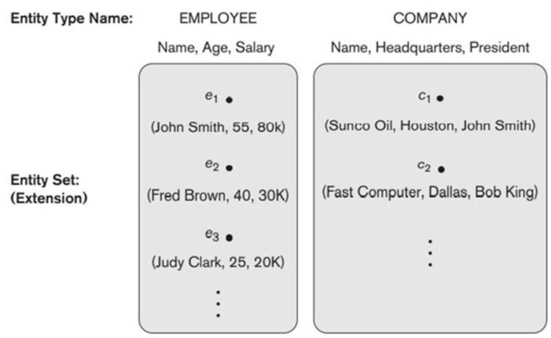
{Address_phone({Phone (Area_code,Phone_number) }, Address(Street_address(Number,Street,Apartment_number), City,State,Zip))}

Entity Types, Entity Sets, Attributes, and Keys:

Entity Types, Entity Sets, Keys, and Value Sets:

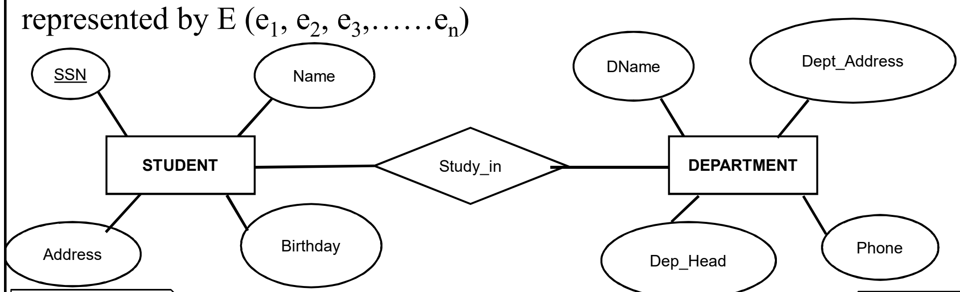
Entity Types and Entity Sets: An entity type defines a collection (or set) of entities that have the same attributes. Each entity type in the database is described by its name and attributes.

For example EMPLOYEE and COMPANY are Entity Types and a list of some of the attributes for each are shown beow.



Entity Types, Entity Sets, Attributes, and Keys:

Entity Set: An entity set is a set of entities of the same type. Entity set E is represented by E (e_1, e_2, e_3,……e_n)



Entity Set
E

Entity Type
EMPLOYEE &
DEPARTMENT

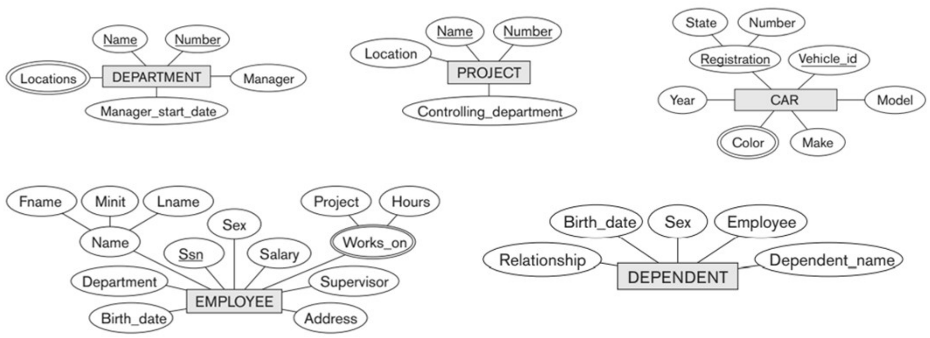
E	SSN	Name	Address	DName	Birthday
e_1	001	Shankar	Kathmandu	Computer Sc.	04/01/2031
e_2	002	Hari	Lalitpur	Computer Sc.	04/02/2032
e_3	003	Shyam	Lalitpur	Physics	04/03/2033
e_4	004	Ram	Kathmandu	Physics	05/12/2034
e_5	005	Deepak	Kirtipur	Chemistry	06/11/2035

Entity
(One individual instances, either e1, e2, e3, e4, e5)

Entity Types, Entity Sets, Attributes, and Keys:

Entity Types, Entity Sets, Keys, and Value Sets:

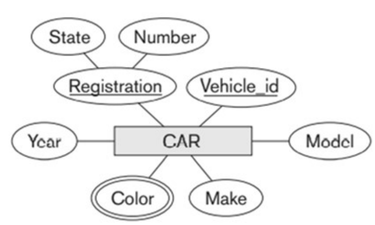
Entity Types and Entity Sets: A few individual entities of each type are also illustrated, along with the values of their attributes. The collection of all entities of a particular entity type in the database at any point in time is called an entity set or entity collection.



Entity Types, Entity Sets, Attributes, and Keys:

Entity Types, Entity Sets, Keys, and Value Sets:

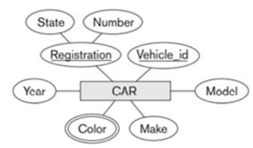
Entity Types and Entity Sets: An entity type is represented in ER diagrams as a rectangular box enclosing the entity type name. Attribute names are enclosed in ovals and are attached to their entity type by straight lines. Composite attributes are attached to their component attributes by straight lines. Multivalued attributes are displayed in double ovals. Figure below shows a CAR entity type in this notation. An entity type describes the schema or intension for a set of entities that share the same structure. The collection of entities of a particular entity type is grouped into an entity set, which is also called the extension of the entity type.



Entity Types, Entity Sets, Attributes, and Keys:

Entity Types, Entity Sets, Keys, and Value Sets:

Key Attributes of an Entity Type: An important constraint on the entities of an entity type is the key or uniqueness constraint on attributes. An entity type usually has one or more attributes whose values are distinct for each individual entity in the entity set. Such an attribute is called a key attribute, and its values can be used to identify each entity uniquely. For example, the Name attribute is a key of the COMPANY entity type in Figure 3.6 because no two companies are allowed to have the same name. For the PERSON entity type, a typical key attribute is Ssn (Social Security number). Sometimes several attributes together form a key, meaning that the combination of the attribute values must be distinct for each entity. In ER diagrammatic notation, each key attribute has its name underlined inside the oval.



Entity Types, Entity Sets, Attributes, and Keys:

Entity Types, Entity Sets, Keys, and Value Sets:

Value Sets (Domains) of Attributes: Each simple attribute of an entity type is associated with a value set (or domain of values), which specifies the set of values that may be assigned to that attribute for each individual entity. If the range of ages allowed for employees is between 16 and 70, we can specify the value set of the Age attribute of EMPLOYEE to be the set of integer numbers between 16 and 70.

Entity Types, Entity Sets, Attributes, and Keys:

Initial Conceptual Design of the COMPANY Database:

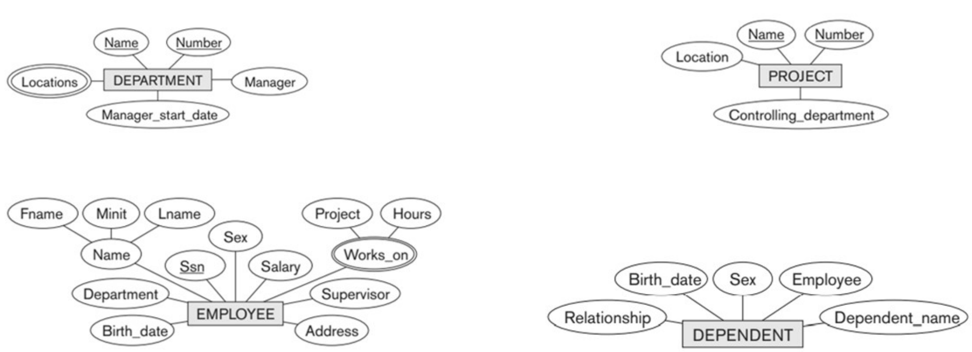
According to the requirements listed above, we can identify four entity types- one corresponding to each of the four items in the specification

1. An entity type DEPARTMENT with attributes Name, Number, Locations, Manager, and Manager_start_date. Locations is the only multivalued attribute. We can specify that both Name and Number are (separate) key attributes because each was specified to be unique.
2. An entity type PROJECT with attributes Name, Number, Location, and Controlling_department. Both Name and Number are (separate) key attributes.
3. An entity type EMPLOYEE with attributes Name, Ssn, Sex, Address, Salary, Birth_date, Department, and Supervisor. Both Name and Address may be composite attributes; however, this was not specified in the requirements. We must go back to the users to see if any of them will refer to the individual components of Name—First_name, Middle_initial, Last_name—or of Address. In our example, Name is modeled as a composite attribute, whereas Address is not, presumably after consultation with the users.
4. An entity type DEPENDENT with attributes Employee, Dependent_name, Sex, Birth_date, and Relationship (to the employee).

Entity Types, Entity Sets, Attributes, and Keys:

Initial Conceptual Design of the COMPANY Database:

Another requirement is that an employee can work on several projects, and the database has to store the number of hours per week an employee works on each project. This requirement can be represented by a multivalued composite attribute of EMPLOYEE called Works_on with the simple components (Project, Hours).



Course Contents

Relationship Types, Relationship Sets, Roles, and Structural Constraints:

Whenever an attribute of one entity type refers to another entity type, some relationship exists. For example, the attribute Manager of DEPARTMENT refers to an employee who manages the department; the attribute Controlling_department of PROJECT refers to the department that controls the project; the attribute Supervisor of EMPLOYEE refers to another employee (the one who supervises this employee); the attribute Department of EMPLOYEE refers to the department for which the employee works; and so on. In the ER model, these references should not be represented as attributes but as relationships. The initial COMPANY database schema will be refined in to represent relationships explicitly. In the initial design of entity types, relationships are typically captured in the form of attributes. As the design is refined, these attributes get converted into relationships between entity types.

Course Contents

Relationship Types, Relationship Sets, Roles, and Structural Constraints:

So we will learn the concepts of relationship types, relationship sets, and relationship instances, concepts of relationship degree, role names, and recursive relationships and then structural constraints on relationships - such as cardinality ratios and existence dependencies.

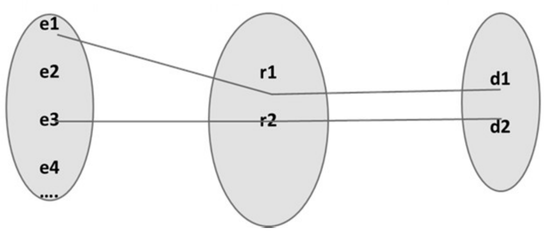
- Relationship types,
- Relationship sets
- Relationship instances
- Relationship degree
- Role names
- Recursive relationships
- Structural constraints
- Cardinality ratios
- Existence dependencies

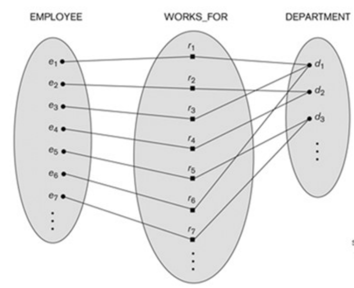
Course Contents

Relationship types, Relationship sets, and Relationship instances:

A relationship type R among n entity types E1, E2, . . . , En defines a set of associations-or a relationship set - among entities from these entity types.

A relationship is an association among entities.



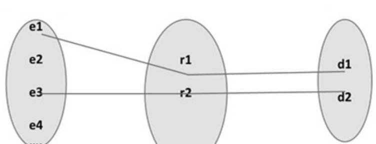


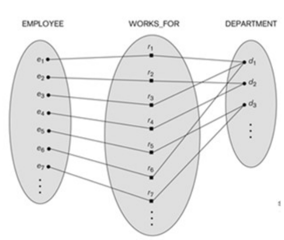
Course Contents

Relationship types, Relationship sets, and Relationship instances:

Relationship set: A relationship set is a set of relationships of the same type. R = { (e_1, e_2, …e_n) | $e_1 \in E_1$, $e_2 \in E_2$…$e_n \in E_n$}

A relationship type R among n entity types E1, E2, . . . , En defines a set of associations—or a relationship set—among entities from these entity types. Similar to the case of entity types and entity sets, a relationship type and its corresponding relationship set are customarily referred to by the same name, R.





Relationship set {r1,r2} where

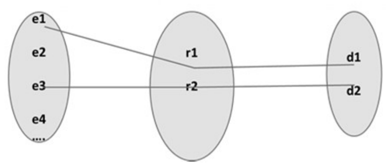
r1= {e1,d1} and r2={e3,d2}

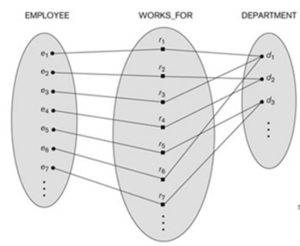
Course Contents

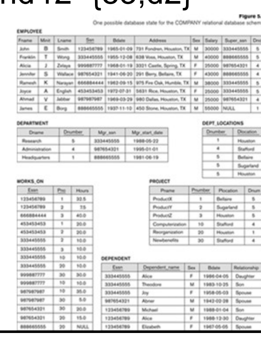
Relationship types, Relationship sets, and Relationship instances:

Relationship Instances: Mathematically, the relationship set R is a set of relationship instances r_i, where each r_i associates n individual entities $(e_1, e_2, \ldots, e_n)$, and each entity e_j in r_i is a member of entity set E_j, $1 \leq j \leq n$.

Instances of Relationship set {r1,r2} ? r1= {e1,d1} and r2={e3,d2}





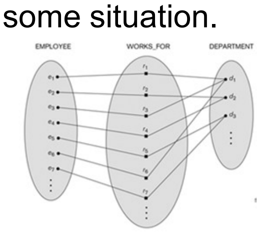


Course Contents

Relationship types, Relationship sets, and Relationship instances:

Relationship Instances: Mathematically, the relationship set R is a set of relationship instances r_i, where each r_i associates n individual entities $(e_1, e_2, \ldots, e_n)$, and each entity e_j in r_i is a member of entity set E_j, $1 \leq j \leq n$.

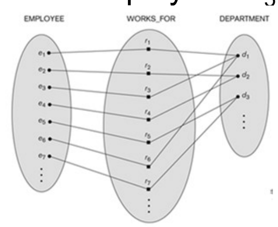
Each relationship instance r_i in R is an association of entities, where the association includes exactly one entity from each participating entity type. Each such relationship instance r_i represents the fact that the entities participating in r_i are related in some way in some situation.



Course Contents

Relationship types, Relationship sets, and Relationship instances:

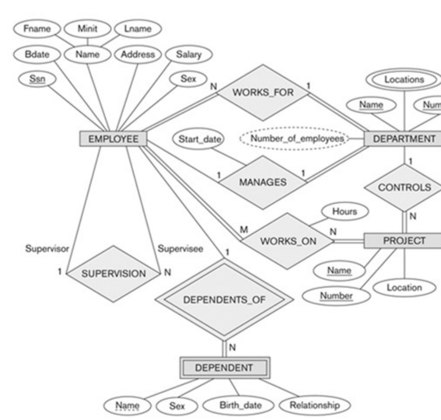
Relationship Instances: For example, consider a relationship type WORKS_FOR between the two entity types EMPLOYEE and DEPARTMENT, which associates each employee with the department for which the employee works. Each relationship instance in the relationship set WORKS_FOR associates one EMPLOYEE entity and one DEPARTMENT entity. Each relationship instance r_i is shown connected to the EMPLOYEE and DEPARTMENT entities that participate in r_i. The employees e_1, e_3, and e_6 work for department d_1; the employees e_2 and e_4 work for department d_2; and the employees e_5 and e_7 work for department d_3.



Course Contents

Relationship types, Relationship sets, and Relationship instances:

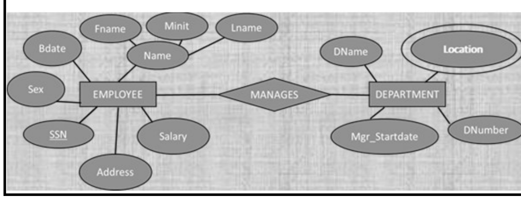
In ER diagrams, relationship types are displayed as diamond-shaped boxes, which are connected by straight lines to the rectangular boxes representing the participating entity types. The relationship name is displayed in the diamond-shaped box.

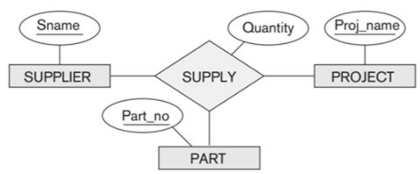


Course Contents

Degree of a Relationship Type: The degree of a relationship type is the number of participating entity types. Hence, the WORKS_FOR relationship is of degree two. A relationship type of degree two is called binary, and one of degree three is called ternary.

An example of a ternary relationship is SUPPLY, where each relationship instance ri associates three entities—a SUPPLIER s, a PART p, and a PROJECT j—whenever s supplies part p to project j. Relationships can generally be of any degree, but the ones most common are binary relationships. Higher-degree relationships are generally more complex than binary relationships.

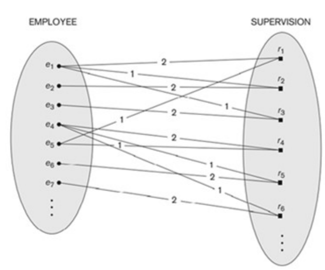




Course Contents

Role Names and Recursive Relationships.:

Role names are not technically necessary in relationship types where all the participating entity types are distinct, since each participating entity type name can be used as the role name. However, in some cases the same entity type participates more than once in a relationship type in different roles. In such cases the role name becomes essential for distinguishing the meaning of the role that each participating entity plays.



Course Contents

Role Names and Recursive Relationships.:

Role names are not technically necessary in relationship types where all the participating entity types are distinct, since each participating entity type name can be used as the role name. However, in some cases the same entity type participates more than once in a relationship type in different roles. In such cases the role name becomes essential for distinguishing the meaning of the role that each participating entity plays.

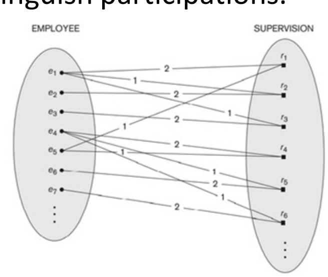
Such relationship types are called recursive relationships or self-referencing relationships. The SUPERVISION relationship type relates an employee to a supervisor, where both employee and supervisor entities are members of the same EMPLOYEE entity set. Hence, the EMPLOYEE entity type participates twice in SUPERVISION: once in the role of supervisor (or boss), and once in the role of supervisee (or subordinate).

Course Contents

Role Names and Recursive Relationships.:

In a recursive relationship type.

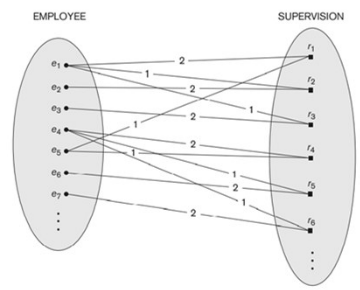
- Both participations are same entity type in different roles.
    - For example, SUPERVISION relationships between EMPLOYEE (in role of supervisor or boss) and (another) EMPLOYEE (in role of subordinate or worker).
- In figure, first role participation labelled with 1 and second role participation labelled with 2.
- In ER diagram, need to display role names to distinguish participations.



Course Contents

Role Names and Recursive Relationships.:

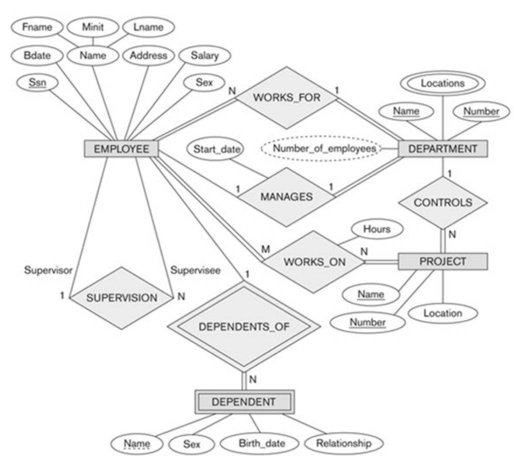
- An relationship type with the same participating entity type in distinct roles. Example - the SUPERVISION relationship
- EMPLOYEE participates twice in two distinct roles:
  - supervisor (or boss) role
  - supervisee (or subordinate) role
- Each relationship instance relates two distinct EMPLOYEE entities:
  - One employee in supervisor role
  - One employee in supervisee role



Course Contents

Constraints on Binary Relationship Types:

- Structural constraints
  - Cardinality ratios (Mapping Cardinality)
  - Participation (Full and Partial)



Course Contents

Constraints on Binary Relationship Types:

Relationship types usually have certain constraints that limit the possible combinations of entities that may participate in the corresponding relationship set. For example, if the company has a rule that each employee must work for exactly one department, then we would like to describe this constraint in the schema.

We can distinguish two main types of binary relationship constraints: cardinality ratio and participation.

Constraints on Binary Relationship Types

Mapping cardinality:
Mapping Cardinality express the number of entities to which binary entity can be associated via a relationship type.

Participation :
In relationship types, association of entities can be fully participated or partial participated.

In One to One relationship types, it is important to know the participation type to place the primary key while reducing E-R into Tables.

Constraints on Binary Relationship Types

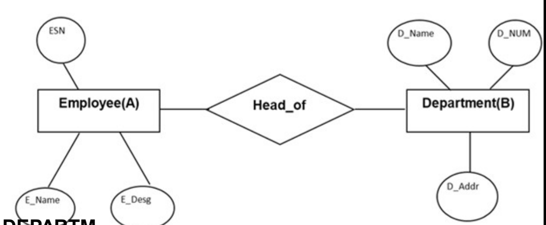
Mapping cardinality:
Mapping cardinalities express the number of entities to which another entity can be associated via a relationship. For binary relationship sets between entity sets A and B, the mapping cardinality must be one of:

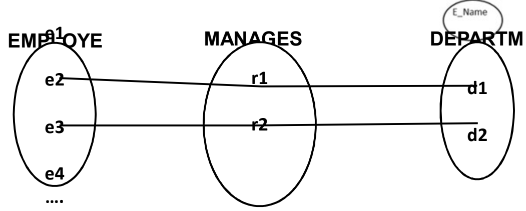
1. **One to One**
2. **One to many or Many to One**
3. **Many to Many**

Constraints on Binary Relationship Types

Mapping cardinality:

1. **One to One : An entity in A is associated with at most one entity in B, and an entity in B is associated with at most one entity in A.**

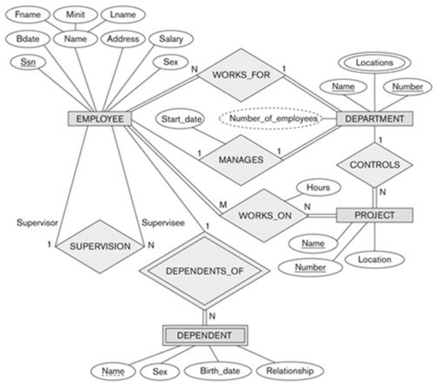


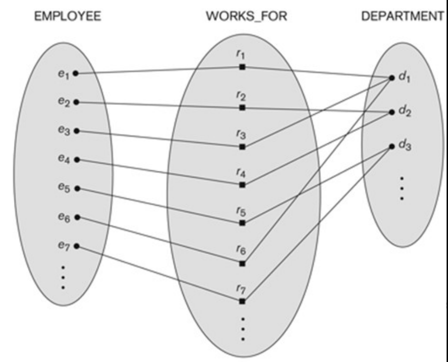


Constraints on Binary Relationship Types

Mapping cardinality:

1. **Many to One : An entity in A is associated with any number in B. An entity in B is associated with at most one entity in A.**

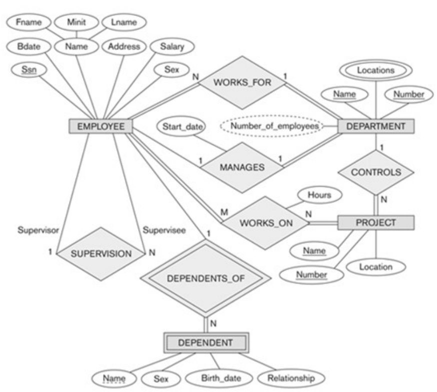


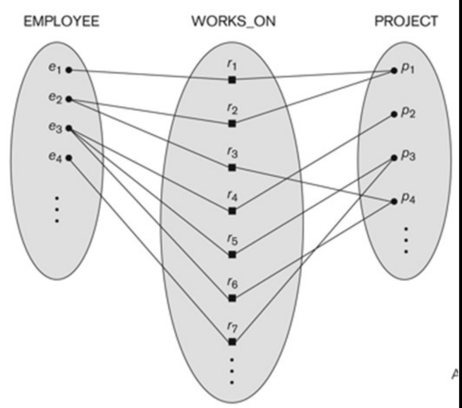


Constraints on Binary Relationship Types

Mapping cardinality:

1. **Many to Many : Entities in A and B are associated with any number from each other.**

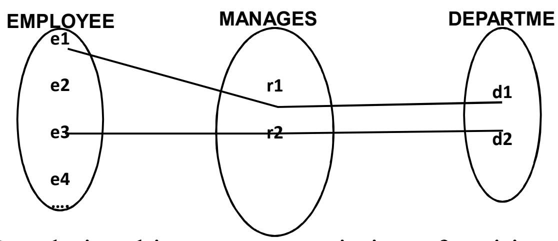


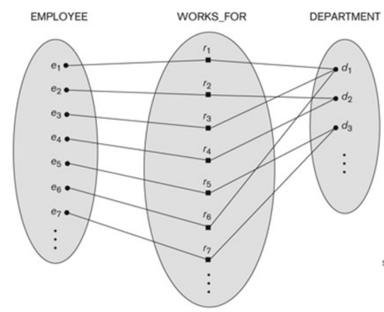


Participation

Mapping Constraints: Relationship types have certain constraints that limit the possible combinations of entities that may participates in the corresponding relationship set. We can distinguish two main types

1) Mapping Cardinality 2) Participation





In relationship types, association of entities can be fully participated or partial participated.

Keys

Differences between entity set (entities) must be expressed in terms of attributes.

Types of Keys

1. **Super Key**
2. **Candidate key**
3. **Primary key**
4. **Foreign Key**

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 of EMPLOYEE

Keys

Super key:

A superkey is a set of one or more attributes which, taken collectively, allow us to identify uniquely an entity in the entity set.

- **A super key (Sk) specifies a uniqueness constraints that no two distinct tuples (records) in entity set can have the same value of Sk.**
- **Every relation has at least one default superkey - the set of all its attributes.**
- **So a superkey can have redundant attributes.**

Keys

Super key:

- **For example, in the entity set EMPLOYEE, sin and EmpName is a superkey.**
- **Similarly ssn, EmpName and Age is also a super key**

So a more useful concept is that of a key which has no redundancies.

- **Note that EmpName alone is not, as two employees could have the same name.**

Keys

Candidate key:

- **A superkey may contain extraneous attributes, and we are often interested in the smallest superkey.**
- **A superkey for which no subset is a superkey is called a candidate key.**
- **In the example above, sin is a candidate key, as it is minimal, and uniquely identifies a EMPLOYEE entity.**
- **Entity CAR may have candidate key i.e. License No and Engine_SN**

Keys

Primary key:

- **A primary key is a candidate key (there may be more than one) or minimum number of superkey chosen by the DB designer to identify entities in an entity set.**
- **Each key is <u>underlined</u> in E-R diagram.**

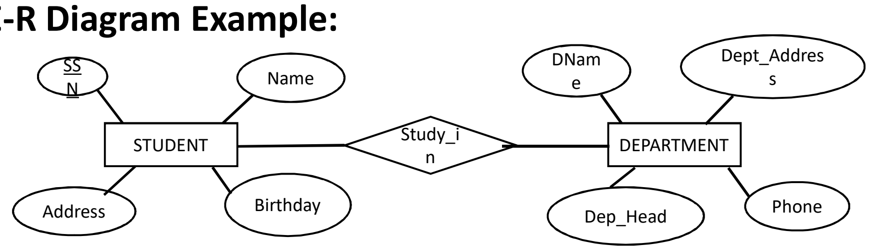
Keys

Foreign Key:

- **A primary key of entity placed in another entity to specify relationship and referential integrity constraints between two entities is called Foreign key.**

Keys

E-R Diagram Example:



STUDENT

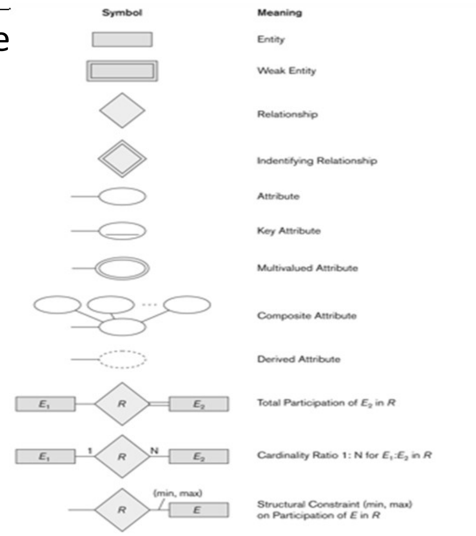
SSN	Name	Address	DName	Birthday
001	Shankar	Kathmandu	Computer Sc.	04/01/2031
002	Hari	Lalitpur	Computer Sc.	04/02/2032
003	Shyam	Lalitpur	Physics	04/03/2033
004	Ram	Kathmandu	Physics	05/12/2034
005	Deepak	Kirtipur	Chemistry	06/11/2035

DEPARTMENT

DName	Dept_Address	Dep_Head	Phone
Computer Sc.	Lalitpur	AAA	4444444
Physics	Lalitpur	BBB	5555555
Chemistry	Lalitpur	CCC	6666666

Symbols used in E-R diagram

Various Symbols used in E-R Mode
1. Entity
2. Weak Entity
3. Relationship
4. Identifying Relationship
5. Attribute
6. Key Attribute
7. Multivalued attribute
8. Composite attribute
9. Derived Attribute



Entity and Attributes in E-R Diagram

In ER diagrams,
- **Entity** is displayed in a rectangular box
- **Attributes** are displayed in ovals
- Each attribute is connected to its entity type
- Components of a composite attribute are connected to the oval representing the composite attribute
- Each key attribute is underlined
- Multivalued attributes displayed in double ovals

So Attributes are of 3 types- (simple, composite, multivalued)

Type of attributes

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). Composition may form a hierarchy where some components are themselves composite.

Multi-valued: An entity may have multiple values for that attribute. For example, Location of an DEPARTMENT.

Entity type

Entity is also of 2 types

1. Strong Entity
2. Weak Entity

Weak Entity: An entity set that does not possess sufficient attributes to form a primary key is called a weak entity set.
i.e. DEPENDENT

Strong Entity: An Entity that does have a primary key is called a strong entity set.
i.e. EMPLOYEE

Entity type - Weak Entity Types

- An entity that does not have a key attribute
- 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

Identifying Relationship

- An Identifying Relationship is a relationship between two entity types, where the key of one entity type is required to uniquely identify instances of a weak entity type.

- The weak entity type will have a partial key (a discriminator) attribute that, in conjunction with the key of the other entity type, uniquely identifies weak entity instances.

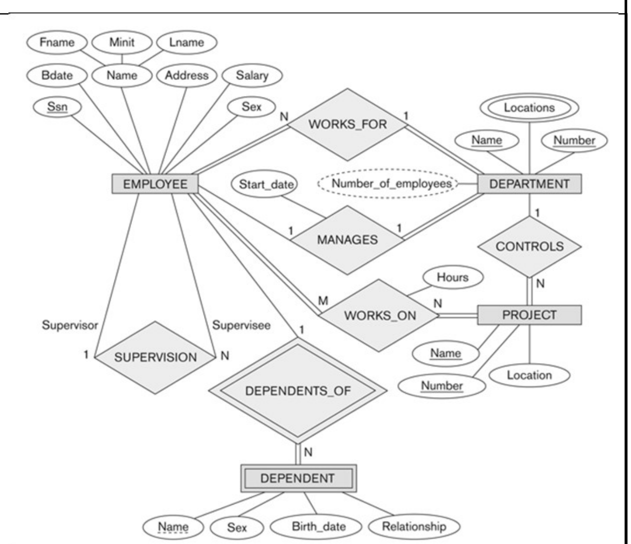
Identifying Relationship

Example:
- A DEPENDENT entity is identified by the dependent's first name, and the specific EMPLOYEE with whom the dependent is related
- Name of DEPENDENT is the partial key
- DEPENDENT is a weak entity type
- EMPLOYEE is its identifying entity type via the identifying relationship type DEPENDENT_OF

So a weak entity and it's identifying relationship in E-R diagram is represented with double line as shown in diagram.

Entity type - Weak Entity Types

Example:



Existence Dependencies

Existence Dependencies:

if the existence of entity X depends on the existence of entity Y, then X is said to be existence dependent on Y. (Or we say that Y is the **dominant entity** and X is the **subordinate entity**.)

i.e. DEPENDENT entity depend on EMPLOYEE Entity

The idea of strong and weak entity sets is related to the existence dependencies.

- Member of a strong entity set is a dominant entity.
- Member of a weak entity set is a subordinate entity.

A weak entity set does not have a primary key, but we need a means of distinguishing among the entities. The **discriminator** of a weak entity set is a set of attributes that allows this distinction to be made. The primary key of a weak entity set is formed by taking the primary key of the strong entity set on which its existence depends plus its discriminator.

ER Diagrams, Naming Conventions, and Design Issues:

ER Diagrams, Naming Conventions, and Design Issues:

1. Summary of Notation for ER Diagrams
2. Proper Naming of Schema Constructs
3. Design Choices for ER Conceptual Design
4. Alternative Notations for ER Diagrams

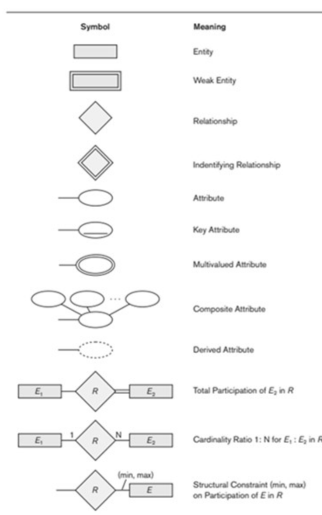
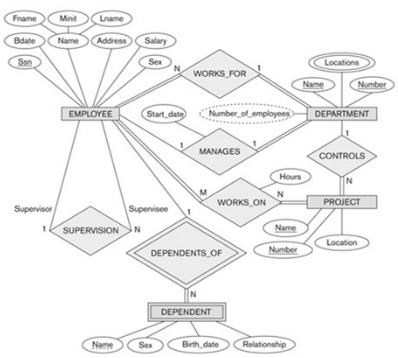


Figure 3.14 Summary of the notation for ER diagrams.

ER Diagrams, Naming Conventions, and Design Issues:

ER Diagrams, Naming Conventions, and Design Issues:

1. Summary of Notation for ER Diagrams
2. Proper Naming of Schema Constructs
3. Design Choices for ER Conceptual Design
4. Alternative Notations for ER Diagrams



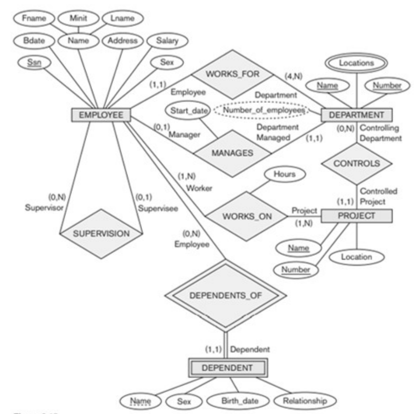


Figure 3.15
ER diagrams for the company schema, with structural constraints specified using (min, max) notation and role names.

ER Diagrams, Naming Conventions, and Design Issues:

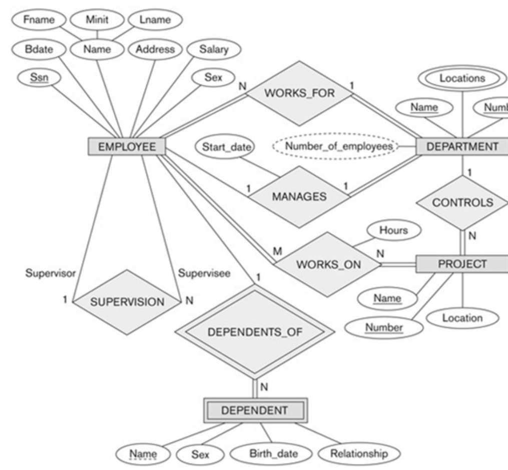
2. Proper Naming of Schema Constructs

- When designing a database schema, the choice of names for entity types, attributes, relationship types, and (particularly) roles is not always straightforward.
- Choose to use singular names for entity types, rather than plural ones, because the entity type name applies to each individual entity belonging to that entity type.
- Entity type and relationship type names are in uppercase letters, attribute names have their initial letter capitalized, and role names are in lowercase letters.
- Names that appear in the database's narrative description to give rise to entity-type names, and verbs tend to indicate relationship-type names. Attribute names generally arise from additional nouns that describe the nouns corresponding to entity types.

ER Diagrams, Naming Conventions, and Design Issues:

2. Proper Naming of Schema Constructs

- Another naming consideration involves choosing binary relationship names to make the ER diagram of the schema readable from left to right and from top to bottom.



ER Diagrams, Naming Conventions, and Design Issues:

3. Design Choices for ER Conceptual Design

- It is sometimes difficult to decide whether a particular concept should be modelled as a type of entity, attribute or relation. So there are brief guidelines on how you pick the concept in certain situations.
- Generally, the schema design process should be considered an iterative improvement process, in which an initial design is created and then iteratively refined until the most appropriate design is achieved.
- A concept may be first modeled as an attribute and then refined into a relationship because it is determined that the attribute is a reference to another entity type.
- Similarly, an attribute that exists in several entity types may be elevated or promoted to an independent entity type.

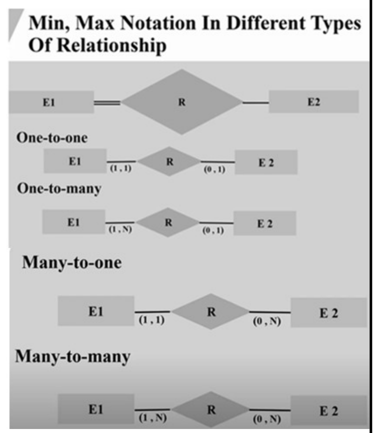
ER Diagrams, Naming Conventions, and Design Issues:

ER Diagrams, Naming Conventions, and Design Issues:

4. Alternative Notations for ER Diagrams

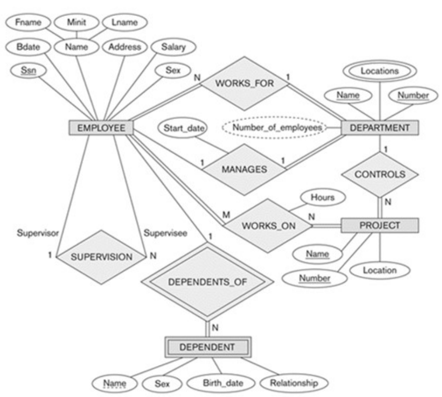
(min, max) notation:

- **min** represent Participation and **max** represent Cardinality. For full participation min=1 and for partial participation min=0. Similarly, max depend on type of cardinality which can be one-to-one, one-to-many, many-to-one and many-to-many. The max value for different types of cardinality are shown below.
- There is no double line for full participation.



ER Diagrams, Naming Conventions, and Design Issues:

4. Alternative Notations for ER Dia



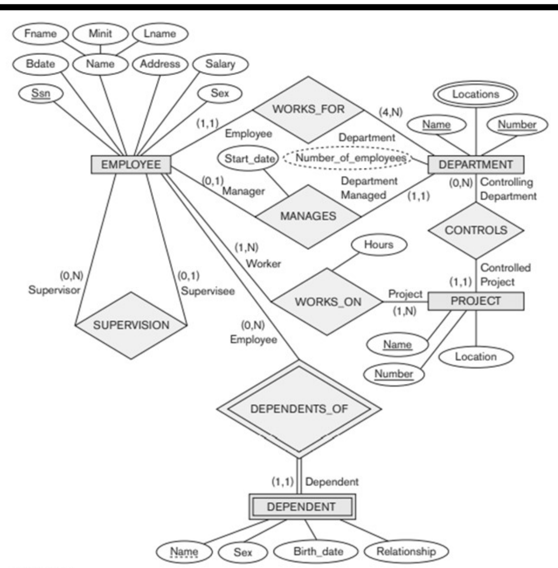


Figure 3.15
ER diagrams for the company schema, with structural constraints specified using (min, max) notation and role names.

Reducing E-R Diagrams to Tables

- **A database conforming to an E-R diagram can be represented by a collection of tables.**
- **For each entity set and relationship set, there is a unique table which is assigned the name of the corresponding set. Each table has a number of columns with unique names.**

1. Representation of Strong Entity Sets
2. Representation of Weak Entity Sets
3. Representation of Relationship Sets
    1) One to One Relation 2) One to Many and 3) Many to Many
4. Representation of Aggregation

Reducing E-R Diagrams to Tables

1. Representation of Strong Entity Sets

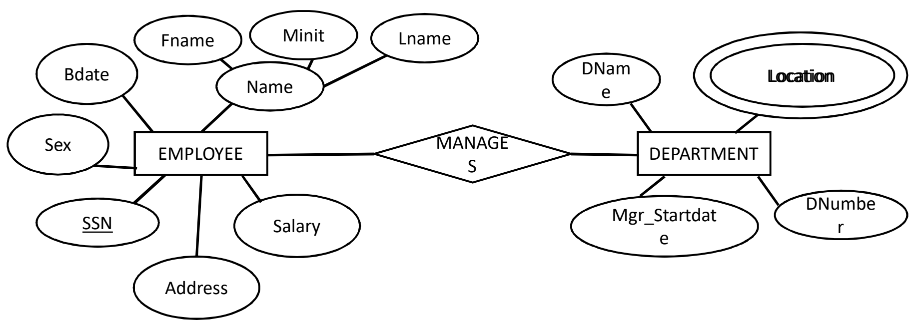
- We use a table with one column for each attribute of the set. Each row in the table corresponds to one entity of the entity set.

2. **Representation of Weak Entity Sets**

- For a weak entity set, we add columns to the table corresponding to the primary key of the strong entity set on which the weak set is dependent.

Reducing E-R Diagrams to Tables

1. Representation of Relationship Sets
    1) One to One Relation



Reducing E-R Diagrams to Tables

1. Representation of Relationship Sets
    1) One to One Relation

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

Reducing E-R Diagrams to Tables

1. Representation of Relationship Sets
    1) One to Many Relation

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

Reducing E-R Diagrams to Tables

1. Representation of Relationship Sets
    1) Many to Many Relation

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

PROJECT

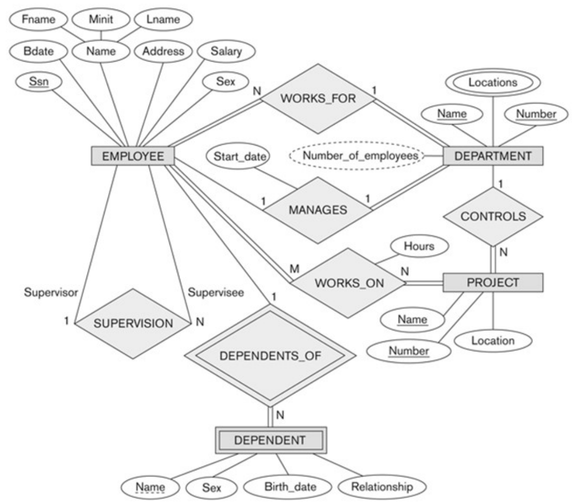
Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

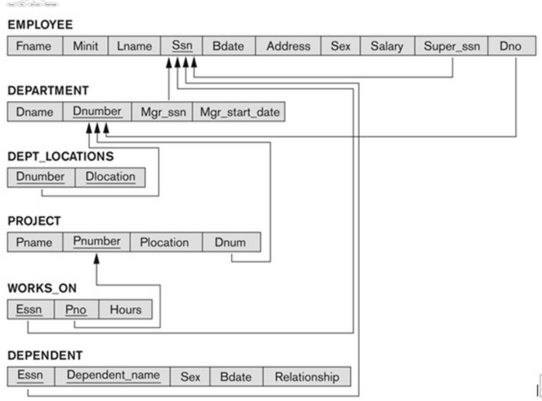
Database Design Example: COMPANY Database

FINAL E-R DIAGRAM:



Example: COMPANY Database

FINAL ER DIAGRAM:



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Example: COMPANY Database

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Course Contents

Relationship Types of Degree Higher than Two:

We defined the degree of a relationship type as the number of participating entity types and called a relationship type of degree two, binary and a relationship type of degree three, ternary.

In this section, we explain on the differences between binary and higher-degree relationships, when to choose higher degree versus binary relationships, and how to specify constraints on higher-degree relationships.

The degree of a relationship type is defined as the number of participating entity types. The relationship type of degree two is called binary and a relationship type of degree three is called ternary.

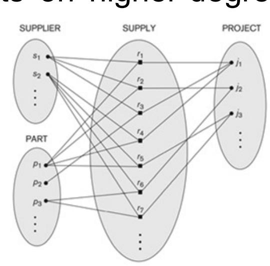
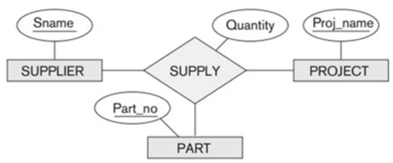


Figure: Some relationship instances in the SUPPLY ternary relationship set.

Course Contents

Relationship Types of Degree Higher than Two:

The ER diagram notation for a ternary relationship type is shown in Figure-1 below, which displays the schema for the SUPPLY relationship type that was displayed at the instance level.



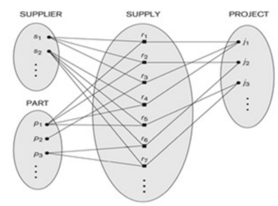


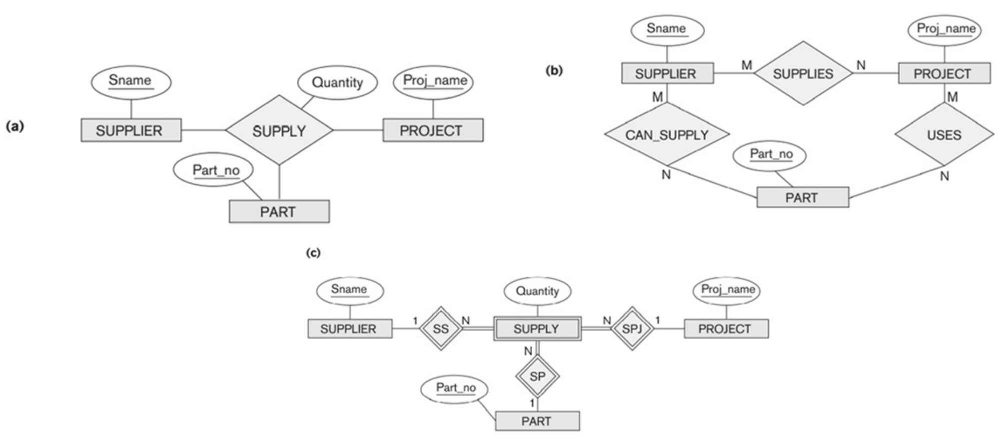
Figure-1: Some relationship instances in the SUPPLY ternary relationship set.

The relationship set of SUPPLY is a set of relationship instances (s, j, p), where the meaning is that s is a SUPPLIER who is currently supplying a PART p to a PROJECT j. In general, a relationship type R of degree n will have n edges in an ER diagram, one connecting R to each participating entity type. (Note: no uses has explained in this)

Course Contents

Relationship Types of Degree Higher than Two:

Figure-b shows an ER diagram for three binary relationship types CAN_SUPPLY, USES, and SUPPLIES.



Course Contents

Relationship Types of Degree Higher than Two:

An ER diagram for three binary relationship types CAN_SUPPLY, USES, and SUPPLIES in general, a ternary relationship type represents different information than do three binary relationship types.

- Consider the three binary relationship types CAN_SUPPLY, USES, and SUPPLIES.
- CAN_SUPPLY, between SUPPLIER and PART, includes an instance (s, p) whenever supplier s can supply part p (to any project);
- USES, between PROJECT and PART, includes an instance (j, p) whenever project j uses part p; and
- SUPPLIES, between SUPPLIER and PROJECT, includes an instance (s, j) whenever supplier s supplies some part to project j.
- The existence of three relationship instances (s, p), (j, p), and (s, j) in CAN_SUPPLY, USES, and SUPPLIES, respectively, does not necessarily imply that an instance (s, j, p) exists in the ternary relationship SUPPLY, because the meaning is different.

Course Contents

Relationship Types of Degree Higher than Two:

- It is often tricky to decide whether a particular relationship should be represented as a relationship type of degree n or should be broken down into several relationship types of smaller degrees. The designer must base this decision on the semantics or meaning of the particular situation being represented.

- Note: The solution is to include the ternary relationship plus one or more of the binary relationships, if they represent different meanings and if all are needed by the application.

Enhanced Entity–Relationship (EER) Model

Enhanced Entity–Relationship (EER):

Subclasses, Superclasses, and Inheritance;
Specialization and Generalization;
Constraints (Overlapping, Disjoint) and
Characteristics of Specialization and Generalization;

Enhanced Entity–Relationship (EER) Model

Enhanced Entity-Relationship (EER) diagrams are basically an expanded version of ER diagrams.

An EER diagram provides with all the elements of an ER diagram while adding:

1. Subclass and Superclass and Inheitance
2. Specialization and Generalization
3. Aggregation
4. Union or Category

These concepts are used in EER schema and the resulting schema diagrams called as EER Diagrams.

Enhanced Entity–Relationship (EER) Model

Features of EER Model:

- EER creates a design more accurate to database schemas.
- It reflects the data properties and constraints more precisely.
- It includes all modeling concepts of the ER model.
- Diagrammatic technique helps for displaying the EER schema.
- It includes the concept of specialization and generalization.
- It is used to represent a collection of objects that is union of objects of different entity types.

Enhanced Entity–Relationship (EER) Model

Subclasses, Superclasses, and Inheritance:

- Subclass and Superclass relationship leads the concept of Inheritance.
- The relationship between sub class and super class is denoted with (d) symbol.

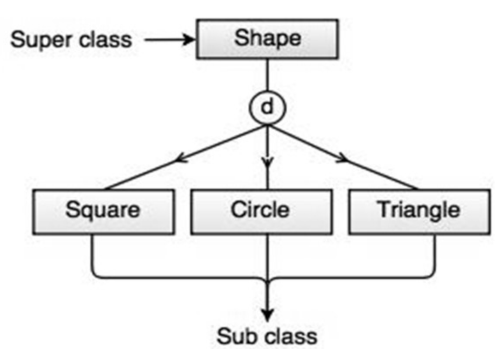


Fig. Super class/Sub class Relationship

Enhanced Entity–Relationship (EER) Model

Subclasses, Superclasses, and Inheritance:

- Many cases an entity type has numerous subgroupings or subtypes of its entities that are meaningful and need to be represented explicitly because of their significance to the database application.

For example, the entities that are members of the EMPLOYEE entity type may be distinguished further into SECRETARY, ENGINEER, MANAGER, TECHNICIAN, SALARIED_EMPLOYEE, HOURLY_EMPLOYEE, and so on.

The set or collection of entities in each of the latter groupings is a subset of the entities that belong to the EMPLOYEE entity set, meaning that every entity that is a member of one of these subgroupings is also an employee. We call each of these subgroupings a subclass or subtype of the EMPLOYEE entity type, and the EMPLOYEE entity type is called the superclass or supertype for each of these subclasses.

Enhanced Entity–Relationship (EER) Model

Subclasses, Superclasses, and Inheritance:

We call the relationship between a superclass and any one of its subclasses a superclass/subclass or supertype/subtype or simply class/subclass relationship. EMPLOYEE/SECRETARY and EMPLOYEE/TECHNICIAN are two class/subclass relationships.

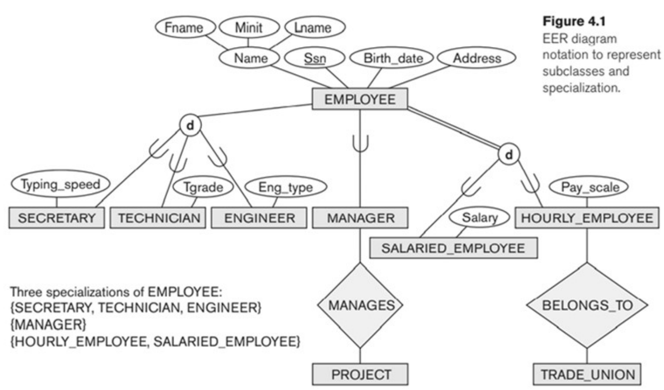


Figure 4.1
EER diagram notation to represent subclasses and specialization.

Enhanced Entity–Relationship (EER) Model

Subclasses, Superclasses, and Inheritance:

An important concept associated with subclasses (subtypes) is that of type inheritance.

- Recall that the entity type is defined by the attributes it possesses and the relationship types in which it participates.
- An entity in the subclass represents the same real-world entity from the superclass, it should possess values for its specific attributes as well as values of its attributes as a member of the superclass.
- An entity that is a member of a subclass inherits all the attributes of the entity as a member of the superclass. The entity also inherits all the relationships in which the superclass participates.

Notice that a subclass, with its own specific (or local) attributes and relationships together with all the attributes and relationships it inherits from the superclass, can be considered an entity type in its own right.

Enhanced Entity–Relationship (EER) Model

Specialization and Generalization:

- The process of making a superclass from a group of subclasses is called generalization. The process of making subclasses from a general concept is called specialization.
- Specialization: A means of identifying sub-groups within an entity set which have attributes that are not shared by all the entities (top-down).
- Generalization: Multiple entity sets are synthesized into a higher-level entity set, based on common features (bottom-up).

- Specialization is the process of defining a set of subclasses of an entity type; this entity type is called the superclass of the specialization.
- The set of subclasses that forms a specialization is defined on the basis of some distinguishing characteristic of the entities in the superclass

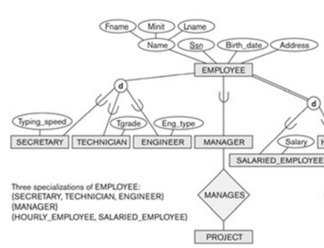


Figure 4.1
EER diagram notation to represent subclasses and specialization.

Enhanced Entity–Relationship (EER) Model

Specialization:

- For example, the set of subclasses {SECRETARY, ENGINEER, TECHNICIAN} is a **specialization** of the superclass EMPLOYEE that distinguishes among employee entities based on the job type of each employee.
- We may have several specializations of the same entity type based on different distinguishing characteristics. For example, another specialization of the EMPLOYEE entity type may yield the set of subclasses {SALARIED_EMPLOYEE, HOURLY_EMPLOYEE}; this specialization distinguishes among employees based on the method of pay.

Figure 4.1 shows how we represent a specialization diagrammatically in an EER diagram. The subclasses that define a specialization are attached by lines to a circle that represents the specialization, which is connected in turn to the superclass. The subset symbol on each line connecting a subclass to the circle indicates the direction of the superclass/subclass relationship. Attributes that apply only to entities of a particular subclass—such as TypingSpeed of SECRETARY—are attached to the rectangle representing that subclass. These are called specific (or local) attributes of the subclass.

Enhanced Entity–Relationship (EER) Model

Two main reasons for including class/subclass relationships and specializations are:

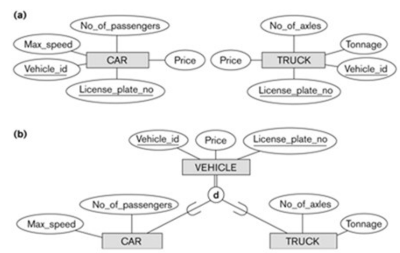
- The first is that certain attributes may apply to some but not all entities of the superclass entity type. A subclass is defined in order to group the entities to which these attributes apply. The members of the subclass may still share the majority of their attributes with the other members of the superclass. For example, in Figure, the SECRETARY subclass has the specific attribute Typing_speed, whereas the ENGINEER subclass has the specific attribute Eng_type, but SECRETARY and ENGINEER share their other inherited attributes from the EMPLOYEE entity type.
- The second reason for using subclasses is that some relationship types may be participated in only by entities that are members of the subclass. For example, if only HOURLY_EMPLOYEES can belong to a trade union, we can represent that fact by creating the subclass HOURLY_EMPLOYEE of EMPLOYEE and relating the subclass to an entity type TRADE_UNION via the BELONGS_TO relationship type.

Enhanced Entity–Relationship (EER) Model

Generalization:

We can think of a reverse process of abstraction in which we suppress the differences among several entity types, identify their common features, and generalize them into a single superclass of which the original entity types are special subclasses. For example, consider the entity types CAR and TRUCK

Because they have several common attributes, they can be generalized into the entity type VEHICLE, as shown in Figure. Both CAR and TRUCK are now subclasses of the generalized superclass VEHICLE. We use the term generalization to refer to the process of defining a generalized entity type from the given entity types.

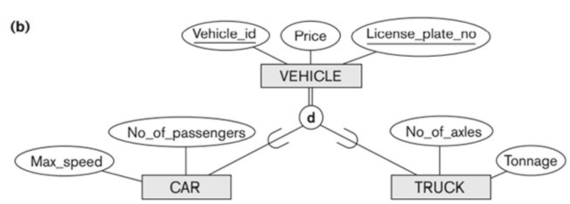


Enhanced Entity–Relationship (EER) Model

Generalization:

Notice that the generalization process can be viewed as being functionally the inverse of the specialization process; we can view {CAR, TRUCK} as a specialization of VEHICLE rather than viewing VEHICLE as a generalization of CAR and TRUCK.

An arrow pointing to the generalized superclass represents a generalization process, whereas arrows pointing to the specialized subclasses represent a specialization process.



Enhanced Entity–Relationship (EER) Model

Constraints and Characteristics of Specialization and Generalization Hierarchies:

- Constraints on Specialization and Generalization
- Specialization and Generalization Hierarchies and Lattices
- Utilizing Specialization and Generalization in Refining Conceptual Schemas

Enhanced Entity–Relationship (EER) Model

Constraints and Characteristics of Specialization and Generalization Hierarchies:

In we learn about:

- Constraints that apply to a single specialization or a single generalization (mainly to specialization)
- Differences between specialization/generalization lattices (multiple inheritance) and hierarchies (single inheritance).
- Differences between the specialization and generalization processes during conceptual database schema design

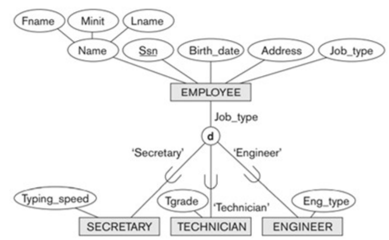
Enhanced Entity–Relationship (EER) Model

Constraints on Specialization and Generalization:

In general, several specializations can be defined on the same entity type (or superclass).

There are three constraints that may apply to a specialization/generalization:

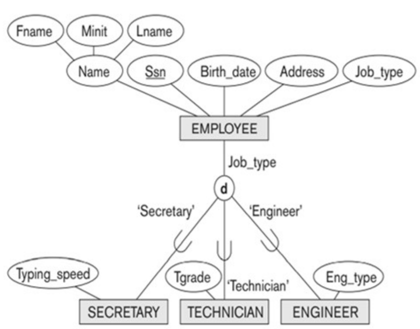
- Membership Constraints
  - Condition-defined (or Predicate-defined)
  - Attribute-defined specialization
  - User-defined
- Disjointness constraints
  - Disjoint
  - Overlapping
- Completeness (or totalness) constraint,
  - Total
  - Partial



EER diagram notation for an attribute-defined specialization on Job_type.

Enhanced Entity–Relationship (EER) Model

Constraints on Specialization and Generalization:



EER diagram notation for an attribute-defined specialization on Job_type.

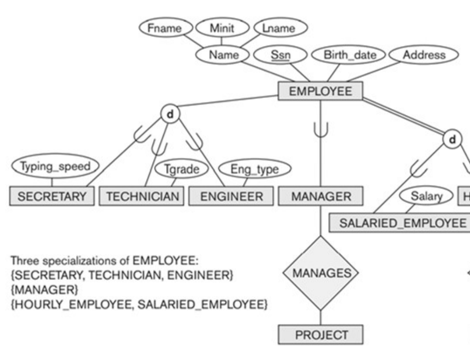


Figure 4.1
EER diagram notation to represent subclasses and specialization.

Enhanced Entity–Relationship (EER) Model

Constraints on Specialization and Generalization:

We may have several specializations defined on the same entity type (or superclass). In such a case, entities may belong to subclasses in each of the specializations. A specialization may also consist of a single subclass only, such as the {MANAGER} specialization in Figure; in such a case, we do not use the circle notation.

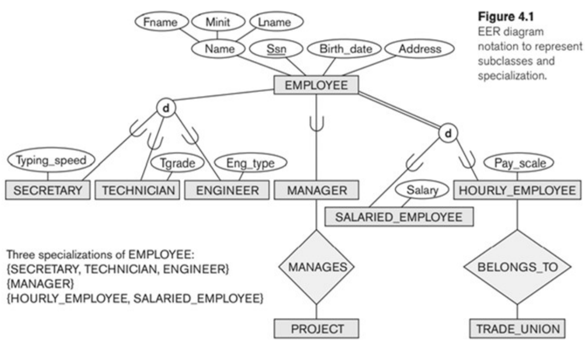
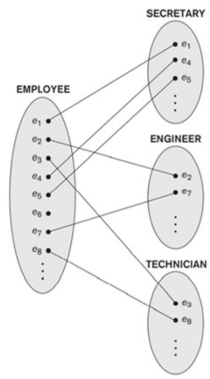


Figure 4.1
EER diagram notation to represent subclasses and specialization.



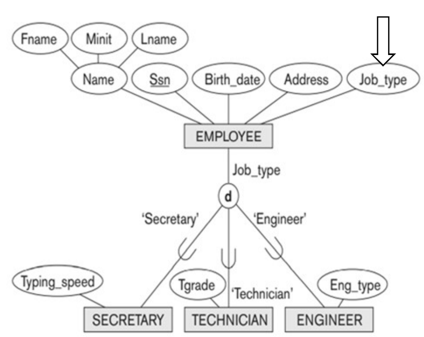
Course Contents

Constraints on Specialization and Generalization:

Membership Constraints: Membership of a specialization/generalization relationship can be defined as a condition in the requirements.

Condition-defined (or Predicate-defined): If we define predicate by placing a condition on the value of some attribute of the superclass. Such subclasses are called predicate-defined (or condition-defined) subclasses.

For example, if the EMPLOYEE entity type has an attribute Job_type, as shown in Figure , we can specify the condition of membership in the SECRETARY subclass by the condition (Job_type = 'Secretary'), which we call the defining predicate of the subclass. This condition is a constraint specifying that exactly those entities of the EMPLOYEE entity type whose attribute value for Job_type is 'Secretary' belong to the subclass. We display a predicate-defined subclass by writing the predicate condition next to the line that connects the subclass to the specialization circle.



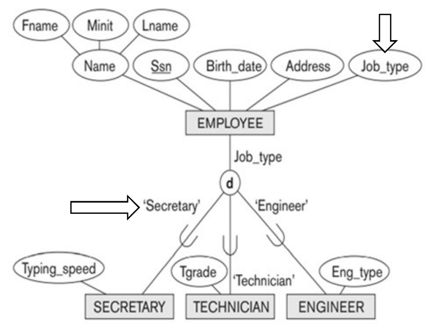
EER diagram notation for an attribute-defined specialization on Job_type.

Course Contents

Constraints on Specialization and Generalization:

Attribute-defined specialization: If all subclasses in a specialization have their membership condition on the same attribute of the superclass, the specialization itself is called an attribute-defined specialization, and the attribute is called the defining attribute of the specialization.

We display an attribute-defined specialization by placing the defining attribute name next to the arc from the circle to the superclass, as shown in Figure



EER diagram notation for an attribute-defined specialization on Job_type.

Course Contents

Constraints on Specialization and Generalization:

User-defined: When we do not have a condition for determining membership in a subclass, the subclass is called user-defined. Membership in such a subclass is determined by the database users when they apply the operation to add an entity to the subclass; hence, membership is specified individually for each entity by the user, not by any condition that may be evaluated automatically.

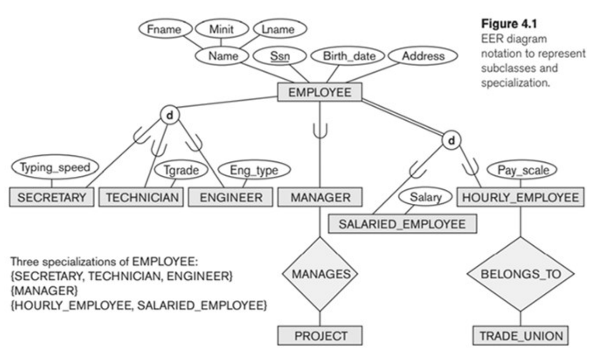
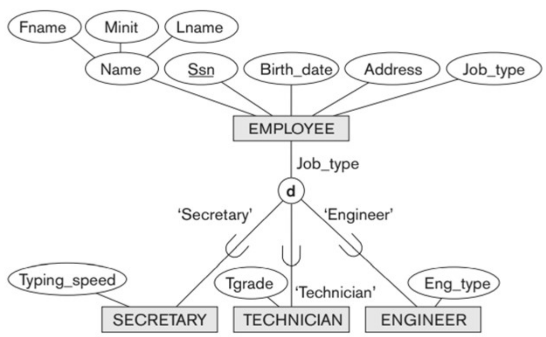


Figure 4.1
EER diagram notation to represent subclasses and specialization.

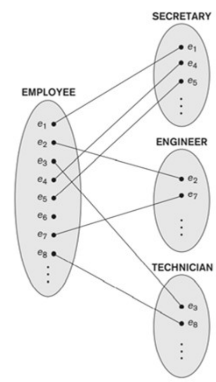
Course Contents

Constraints on Specialization and Generalization:

Disjointness constraints: The disjoint constraint only applies when a superclass has more than one subclass. If the subclasses are disjoint, then an entity occurrence can be a member of only one of the subclasses, e.g. SECRETARY, TECHNICIAN or ENGINEER.



EER diagram notation for an attribute-defined specialization on Job_type.

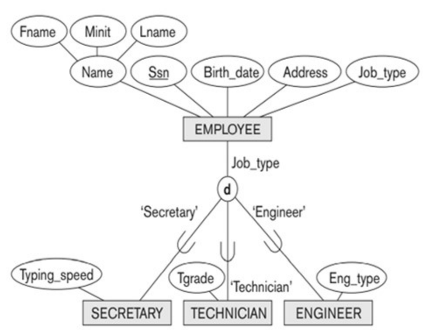


Course Contents

Constraints on Specialization and Generalization:

Disjoint: The disjointness constraint, which specifies that the subclasses of the specialization must be disjoint sets. This means that an entity can be a member of at most one of the subclasses of the specialization.

A specialization that is attribute-defined implies the disjointness constraint (if the attribute used to define the membership predicate is single valued). Figure illustrates this case, where the **d** in the circle stands for disjoint. The **d** notation also applies to user-defined subclasses of a specialization that must be disjoint

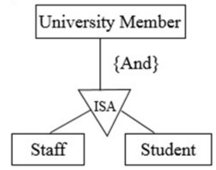


EER diagram notation for an attribute-defined specialization on Job_type.

Course Contents

Constraints on Specialization and Generalization:

Overlapping: This applies when an entity occurrence may be a member of more than one subclass, e.g. student and staff and some people are both.



If the subclasses are not constrained to be disjoint, their sets of entities may be overlapping; that is, the same (real-world) entity may be a member of more than one subclass of the specialization. This case, which is the default, is displayed by placing an **o** in the circle.

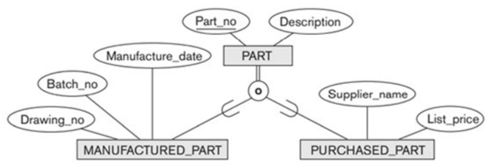


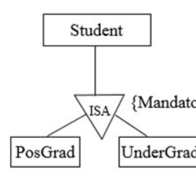
Figure 4.5
EER diagram notation for an overlapping (nondisjoint) specialization.

Course Contents

Constraints on Specialization and Generalization:

Completeness (or totalness) constraint: Constraint on specialization is called the completeness (or totalness) constraint, which may be total or partial.

Total: Each superclass (higher-level entity) must belong to subclasses (lower-level entity sets), e.g. a student must be postgrad or undergrad.



A total specialization constraint specifies that every entity in the superclass must be a member of at least one subclass in the specialization. For example, if every EMPLOYEE must be either an HOURLY_EMPLOYEE or a SALARIED_EMPLOYEE, then the specialization {HOURLY_EMPLOYEE, SALARIED_EMPLOYEE} in Figure is a total specialization of EMPLOYEE. This is shown in EER diagrams by using a double line to connect the superclass to the circle.

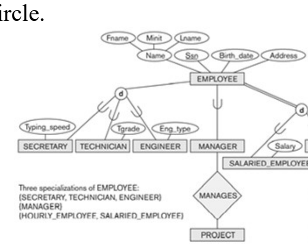


Figure 4.1
EER diagram notation to represent subclasses and specialization.

Course Contents

Constraints on Specialization and Generalization:

Completeness (or totalness) constraint: Constraint on specialization is called the completeness (or totalness) constraint, which may be total or partial.

Partial specialization: A single line is used to display a partial specialization, which allows an entity not to belong to any of the subclasses. For example, if some EMPLOYEE entities do not belong to any of the subclasses {SECRETARY, ENGINEER, TECHNICIAN} in Figures 4.1 then that specialization is partial.

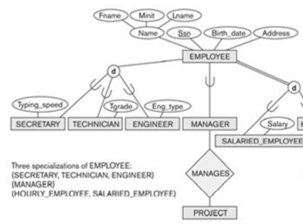
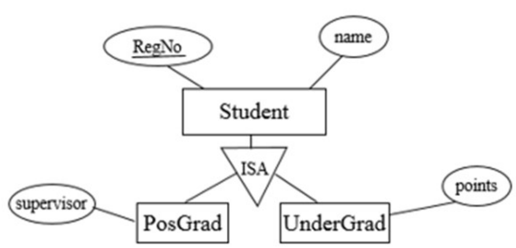


Figure 4.1 EER diagram notation to represent subclasses and specialization.

Mapping specialization/generalization to relational tables

Mapping specialization/generalization to relational tables:

Specialization/generalization relationship can be mapped to relational tables in three methods. To demonstrate the methods, we will take the student, postgraduate and undergraduate relationship. A student in the university has a registration number and a name. Only postgraduate students have supervisors. Undergraduates accumulates points through their coursework.



Mapping specialization/generalization to relational tables

Mapping specialization/generalization to relational tables:

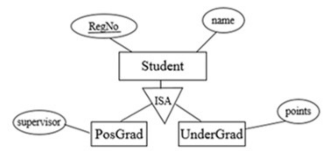
Method 1

All the entities in the relationship are mapped to individual tables.

Student (Regno, name)

PosGrad (Regno, supervisor)

UnderGrad (Regno, points)



Method 2

Only subclasses are mapped to tables. The attributes in the superclass are duplicated in all subclasses.

PosGrad (Regno, name, supervisor)

UnderGrad (Regno, name, points)

This method is most preferred when inheritance is disjoint and complete, e.g. every student is either PosGrad or UnderGrad and nobody is both.

Mapping specialization/generalization to relational tables

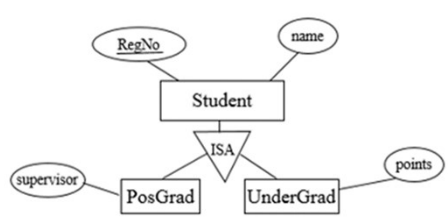
Mapping specialization/generalization to relational tables:

Method 3

Only the superclass is mapped to a table. The attributes in the subclasses are taken to the superclass.

Student (Regno, name, supervisor, points)

This method will introduce null values. When we insert an undergraduate record in the table, the supervisor column value will be null. In the same way, when we insert a postgraduate record in the table, the points value will be null.



Course Contents

Specialization and Generalization Hierarchies and Lattices:

A subclass itself may have further subclasses specified on it, forming a hierarchy or a lattice of specializations.

A specialization hierarchy has the constraint that every subclass participates as a subclass in only one class/subclass relationship; that is, each subclass has only one parent, which results in a tree structure or strict hierarchy.

In contrast, for a specialization lattice, a subclass can be a subclass in more than one class/subclass relationship. Hence, Figure 4.6 and 4.7 is a lattice.

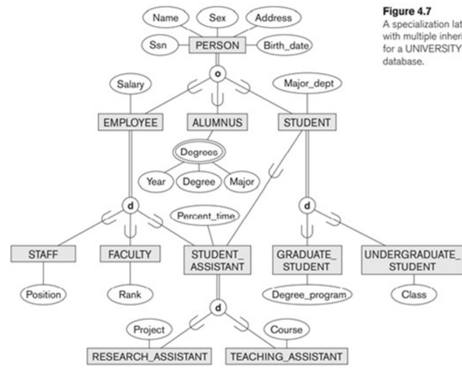
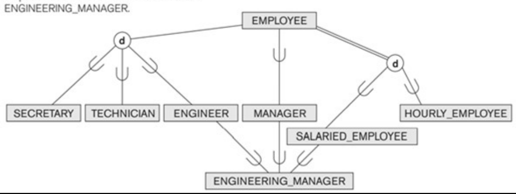


Figure 4.7
A specialization lattice with multiple inheritance for a UNIVERSITY database.

Figure 4.6
A specialization lattice with shared subclass ENGINEERING_MANAGER.



Course Contents

Specialization and Generalization Hierarchies and Lattices:

A subclass itself may have further subclasses specified on it, forming a hierarchy or a lattice of specializations.

A subclass with more than one superclass is called a shared subclass, such as ENGINEERING_MANAGER in Figure 4.6. This leads to the concept known as multiple inheritance, where the shared subclass ENGINEERING_MANAGER directly inherits attributes and relationships from multiple superclasses.

Notice that the existence of at least one shared subclass leads to a lattice (and hence to multiple inheritance); if no shared subclasses existed, we would have a hierarchy rather than a lattice and only single inheritance would exist.

Note: It is important to note here that some models and languages are limited to single inheritance and do not allow multiple inheritance (shared subclasses).

Course Contents

Specialization and Generalization Hierarchies and Lattices:

An important rule related to multiple inheritance can be illustrated by the example of the shared subclass STUDENT_ASSISTANT in Figure 4.7, which inherits attributes from both EMPLOYEE and STUDENT. Here, both EMPLOYEE and STUDENT inherit the same attributes from PERSON. The rule states that if an attribute (or relationship) originating in the same superclass (PERSON) is inherited more than once via different paths (EMPLOYEE and STUDENT) in the lattice, then it should be included only once in the shared subclass (STUDENT_ASSISTANT).

Hence, the attributes of PERSON are inherited only once in the STUDENT_ASSISTANT subclass in Figure 4.7.

Course Contents

Utilizing Specialization and Generalization in Refining Conceptual Schemas:

- The processes about how specialization and generalization are used to refine conceptual schemas during conceptual database design.
- In the specialization process, the database designers typically start with an entity type and then define subclasses of the entity type by successive specialization; that is, they repeatedly define more specific groupings of the entity type.

Generalization:

- It is possible to arrive at the same hierarchy or lattice from the other direction. In such a case, the process involves generalization rather than specialization and corresponds to a bottom-up conceptual synthesis.
- The final design of hierarchies or lattices resulting from either process may be identical; the only difference relates to the manner or order in which the schema superclasses and subclasses were created during the design process.
- Notice that the notion of representing data and knowledge by using superclass/subclass hierarchies and lattices is quite common in knowledge-based systems and expert systems, which combine database technology with artificial intelligence techniques.

Course Contents

Questions:

1. What are the phases of Database Design? Explain.
2. Exlain Entity types and Entity sets with example.
3. Explain Relationship types and Relationship sets with example.
4. What is Attributes? Explain different types of Attributes with example.
5. What is Primary key and Foreign Key? Why is this required in the BDMS?
6. What is E-R Diagram? Explain various symbols used in E-R diagram.
7. What are the differences between ER diagram and alternate representation of ER diagarm in (min, max) representation? Explain with example.
8. What is EER diagram? Explain the components of EER diagram with example.