

2

Database Design

Syllabus

Entity-Relationship model - E-R Diagrams - Enhanced-ER Model - ER-to-Relational Mapping - Functional Dependencies - Non-loss Decomposition - First, Second, Third Normal Forms, Dependency Preservation - Boyce/Codd Normal Form - Multi-valued Dependencies and Fourth Normal Form - Join Dependencies and Fifth Normal Form.

Contents

Introduction to Entity Relationship Model

ER Diagrams

Enhanced ER Model

Examples based on ER Diagram

ER to Relational MappingMay-17, Marks 13

Concept of Relational Database Design

Functional Dependencies

Concept of Redundancy and Anomalies

DecompositionDec.-17, Marks 7

Normal FormsDec.-14, 15, May-18 Marks 16

Boyce / Codd Normal Form (BCNF)

Multivalued Dependencies and Fourth Normal Form May-14, Dec.-16..... Marks 16

Join Dependencies and Fifth Normal Form

Part I Entity Relationship Model

2.1 Introduction to Entity Relationship Model

Entity Relational model is a model for identifying entities to be represented in the database and representation of how those entities are related.

Let us first understand the design process of database design.

2.1.1 Design Phases

Following are the six steps of database design process. The ER model is most relevant to first three steps

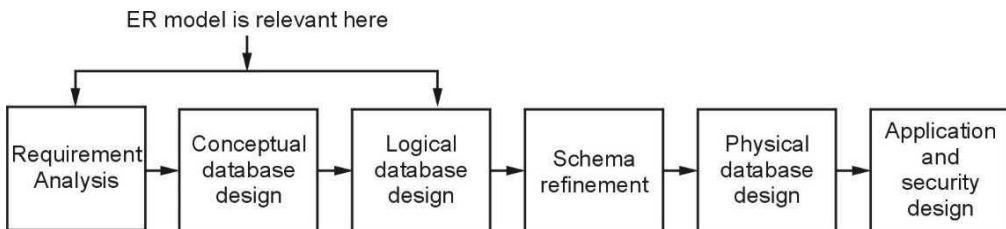


Fig. 2.1.1 : Database design process

Step 1 : Requirement analysis :

- In this step, it is necessary to understand what data need to be stored in the database, what applications must be built, what are all those operations that are frequently used by the system.
- The requirement analysis is an **informal** process and it requires proper **communication** with user groups.
- There are several methods for **organizing and presenting information** gathered in this step.
- Some **automated tools** can also be used for this purpose.

Step 2 : Conceptual database design :

- This is a steps in which **E-R Model** i.e. Entity Relationship model is built.
- E-R model is a **high level data model** used in database design.
- The **goal** of this design is to create a simple description of data that matches with the requirements of users.

Step 3 : Logical database design :

- This is a step in which ER model in **converted to relational database schema**, sometimes called as the logical schema in the relational data model.

Step 4 : Schema refinement :

- In this step, **relational database schema is analyzed** to identify the potential problems and to refine it.
- The schema refinement can be done with the help of **normalizing and restructuring the relations**.

Step 5 : Physical database design :

- In this step, the design of database is **refined further**.
- The tasks that are performed in this step are - building **indexes** on tables and **clustering** tables, redesigning some parts of schema obtained from earlier design steps.

Step 6 : Application and security design :

- Using design methodologies like UML(Unified Modeling Language) the design of the database can be accomplished.
- The **role of each entity** in every process must be reflected in the application task.
- For each role, there must be the provision for **accessing** the some part of database and **prohibition of access** to some other part of database.
- Thus some **access rules** must be enforced on the application(which is accessing the database) to protect the **security features**.

2.2 ER Model

The ER data model specifies enterprise schema that represents the overall logical structure of a database.

The E-R model is very useful in mapping the meanings and interactions of real-world entities onto a conceptual schema.

The ER model consists of three basic concepts –

1) Entity Sets

- **Entity** : An entity is an object that exists and is distinguishable from other objects. For example - Student named “Poonam” is an entity and can be identified by her name. The entity can be concrete or abstract. The concrete entity can be - Person, Book, Bank. The abstract entity can be like - holiday, concept entity is represented as a box.

Student

Employee

Department

- **Entity set** : The entity set is a set of entities of the same types. For example - All students studying in class X of the School. The entity set need not be disjoint. Each entity in entity set have the same set of attributes and the set of attributes will

distinguish it from other entity sets. No other entity set will have exactly the same set of attributes.

2) Relationship Sets

Relationship is an association among two or more entities.

The **relationship set** is a collection of similar relationships. For example - Following Fig. 2.1.2 shows the relationship **works_for** for the two entities **Employee** and **Departments**.

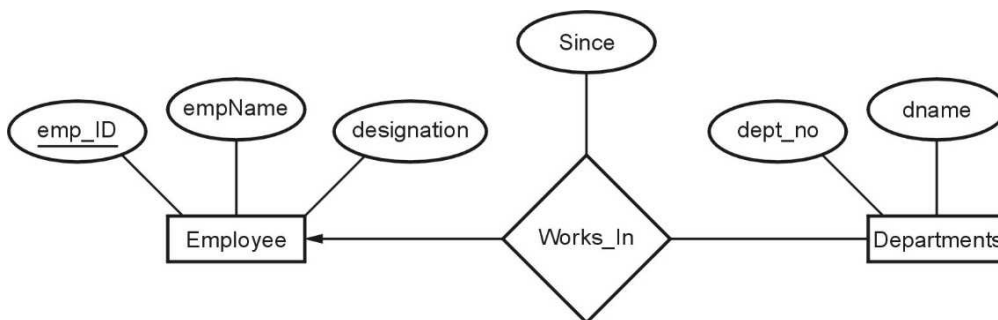


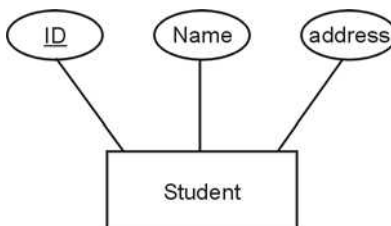
Fig. 2.1.2 : Relation set

The association between entity sets is called as **participation**. that is, the entity sets E_1, E_2, \dots, E_n participate in relationship set R .

The function that an entity plays in a relationship is called that entity's **role**.

3) Attributes

Attributes define the properties of a data object of entity. For example if student is an entity, his ID, name, address, date of birth, class are its attributes. The attributes help in determining the unique entity. Refer Fig. 2.1.3 for Student entity set with attributes - ID, name, address. Note that entity is shown by rectangular box and attributes are shown in oval. The primary key is underlined.



Types of Attributes

Fig. 2.1.3 : Student entity set with attributes

1) Simple and Composite Attributes :

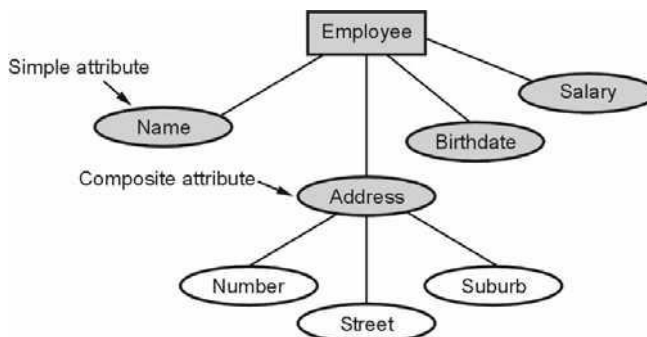
1) Simple attributes are attributes that are drawn from the atomic value domains

For example - Name = {Parth} ; Age = {23}

2) Composite attributes: Attributes that consist of a hierarchy of attributes

For example - Address may consists of "Number", "Street" and "Suburb"

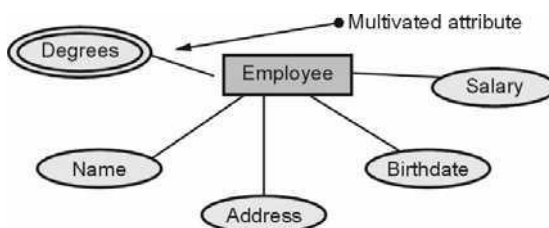
→ Address = {59 + 'JM Road' + 'ShivajiNagar'}



2) Single valued and multivalued :

- There are some attributes that can be represented using a single value. For example - StudentID attribute for a Student is specific only one studentID.
- Multivalued attributes : Attributes that have a set of values for each entity. It is represented by concentric ovals

For example - Degrees of a person: ' BSc' , 'MTech' , 'PhD'



3) Derived attribute :

Derived attributes are the attributes that contain values that are calculated from other attributes. To represent derived attribute there is dotted ellipse inside the solid ellipse. For example -Age can be derived from attribute DateOfBirth. In this situation, DateOfBirth might be called Stored Attribute.

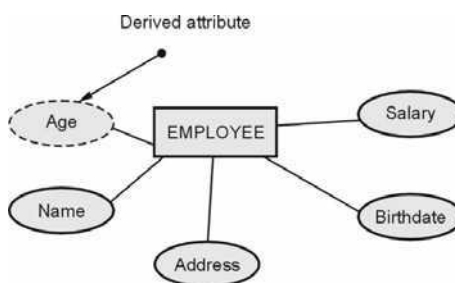


Fig. 2.1.4



2.3 ER Diagrams

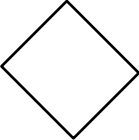
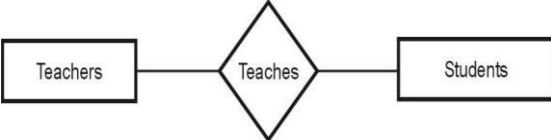
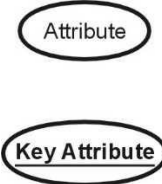
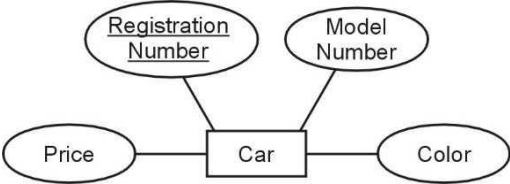

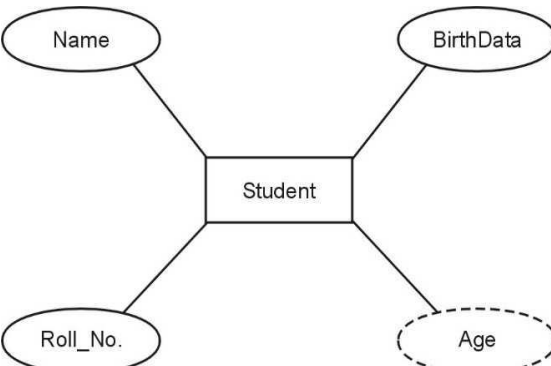

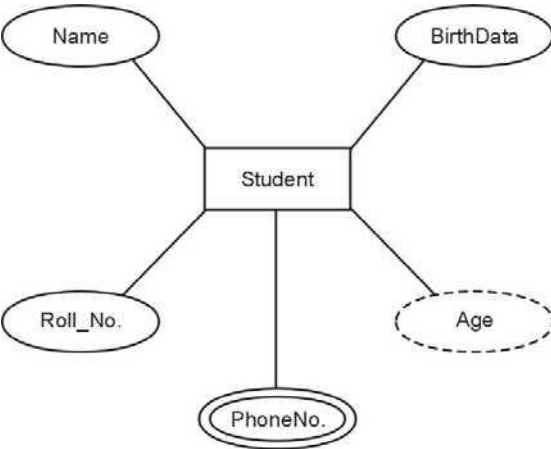
An E-R diagram can express the overall logical structure of a database graphically. E-R diagrams are used to model real-world objects like a person, a car, a company and the relation between these real-world objects.

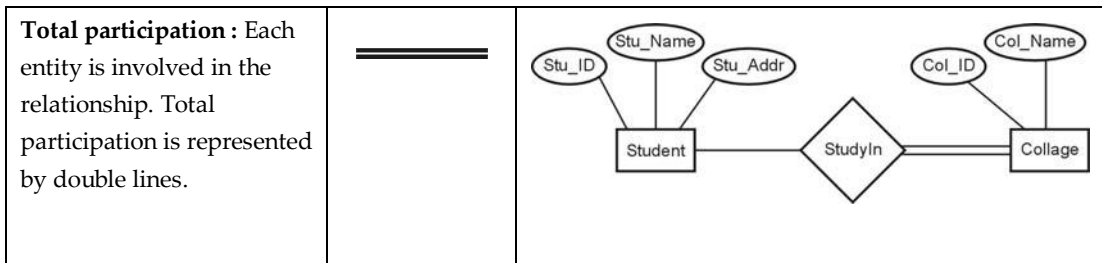
Features of ER model

- i) E-R diagrams are used to represent E-R model in a database, which makes them easy to be converted into relations (tables).
- ii) E-R diagrams provide the purpose of real-world modeling of objects which makes them intently useful.
- iii) E-R diagrams require no technical knowledge and no hardware support.
- iv) These diagrams are very easy to understand and easy to create even by a naive user.
- v) It gives a standard solution of visualizing the data logically.

Various Components used in ER Model are -

Component	Symbol	Example
Entity: Any real-world object can be represented as an entity about which data can be stored in a database. All the real world objects like a book, an organization, a product, a car, a person are the examples of an entity.		

<p>Relationship : Rhombus is used to setup relationships between two or more entities.</p>		
<p>Attribute : Each entity has a set of properties. These properties of each entity are termed as attributes. For example, a car entity would be described by attributes such as price, registration number, model number, color etc</p>		
<p>Derived attribute : Derived attributes are those which are derived based on other attributes, for example, age can be derived from date of birth.</p> <p>To represent a derived attribute, another dotted ellipse is created inside the main ellipse</p>		
<p>Multivalued attribute : An attribute that can hold multiple values is known as multivalued attribute. We represent it with double ellipses in an E-R Diagram. E.g. A person can have more than one phone numbers so the phone number attribute is multivalued.</p>		

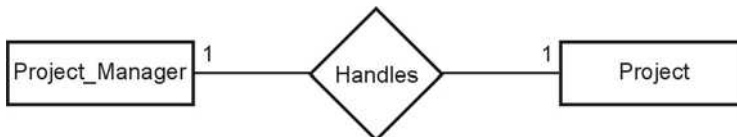


2.3.1 Mapping Cardinality Representation using ER Diagram (Binary Relationship)

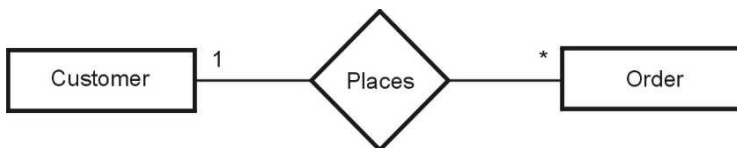
Binary Relationship means relation between two Entities.

There are four types of relationships that are considered for key constraints.

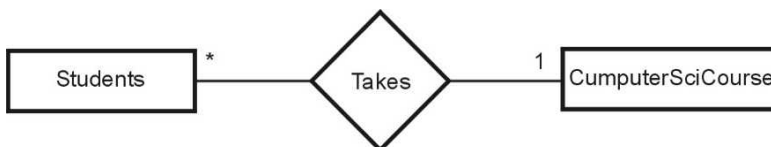
- i) **One to one relation :** When entity A is associated with at the most one entity B then it shares one to one relation. For example - There is one project manager who manages only one project.



- ii) **One to many :** When entity A is associated with more than one entities at a time then there is one to many relation. For example - One customer places order at a time.



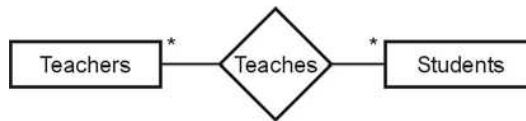
- ii) **Many to one :** When more than one entities are associated with only one entity then there is many to one relation. For example - Many student take a ComputerSciCourse.



Alternate representation can be



iii) **Many to many** : When more than one entities are associated with more than one entities. For example -Many teachers can teach many students.



Alternate representation can be



2.3.2 Ternary Relationship

The relationship in which three entities are involved is called ternary relationship. For example -

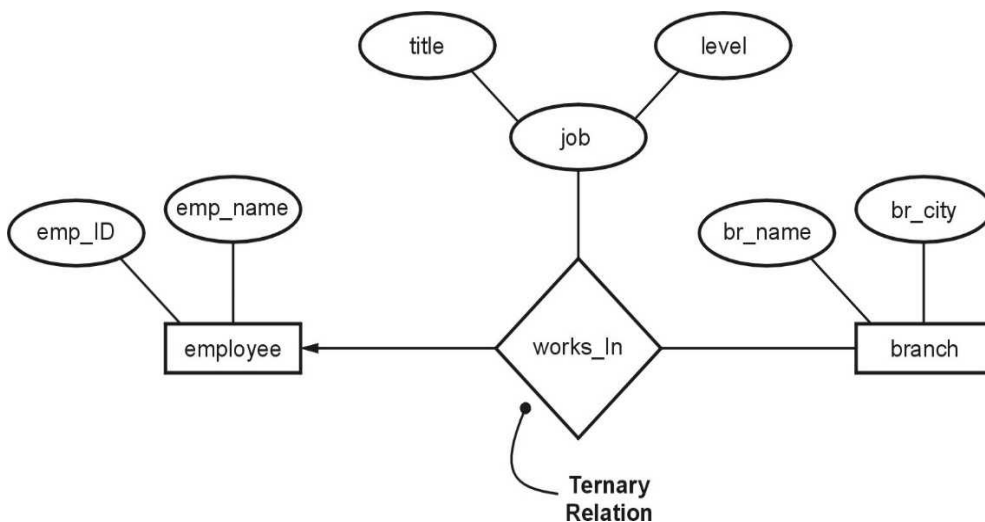


Fig. 2.3.3 : Ternary relation

Difference between Strong and Weak Entity Set

Sr. No.	Strong entity set	Weak entity set
1	It has its own primary key.	It does not have sufficient attribute to form a primary key on its own.
2.	It is represented by rectangle	It is represented by double rectangle.

Sr. No.	Strong entity set	Weak entity set
3.	It represents the primary key which is underlined.	It represents the partial key or discriminator which is represented by dashed underline.
4.	The member of strong entity set is called as dominant entity set	The member of weak entity set is called subordinate entity set.
5.	The relationship between two strong entity sets is represented by diamond symbol.	The relationship between strong entity set and weak entity set is represented by double diamond symbol.
6.	The primary key is one of the attributes which uniquely identifies its member.	The primary key of weak entity set is a combination of partial key and primary key of the strong entity set.

2.4 THE ENHANCED ER MODEL

As the complexity of data increased in the late 1980s, it became more and more difficult to use the traditional ER Model for database modelling. Hence some improvements or enhancements were made to the existing ER Model to make it able to handle the complex applications better.

Hence, as part of the Enhanced ER Model, along with other improvements, three new concepts were added to the existing ER Model, they were:

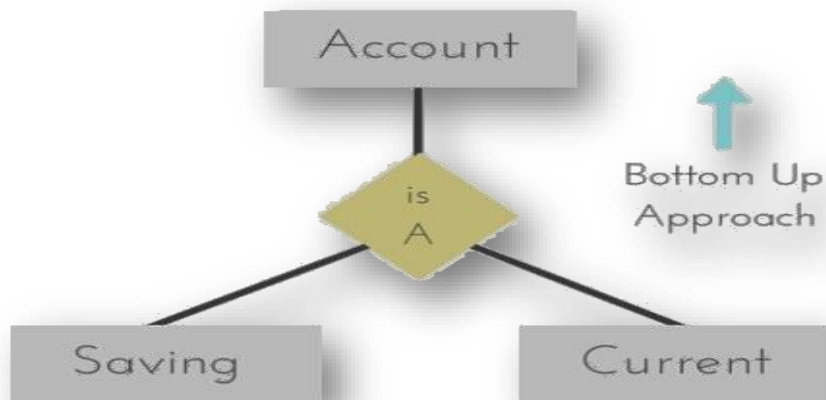
1. Generalization
2. Specialization
3. Aggregation

Generalization

Generalization is a bottom-up approach in which two lower level entities combine to form a higher level entity. In generalization, the higher level entity can also combine with other lower level entities to make further higher level entity.

It's more like Superclass and Subclass system, but the only difference is the approach, which is bottom-up. Hence, entities are combined to form a more generalised entity, in other words, sub-classes are combined to form a super- class.

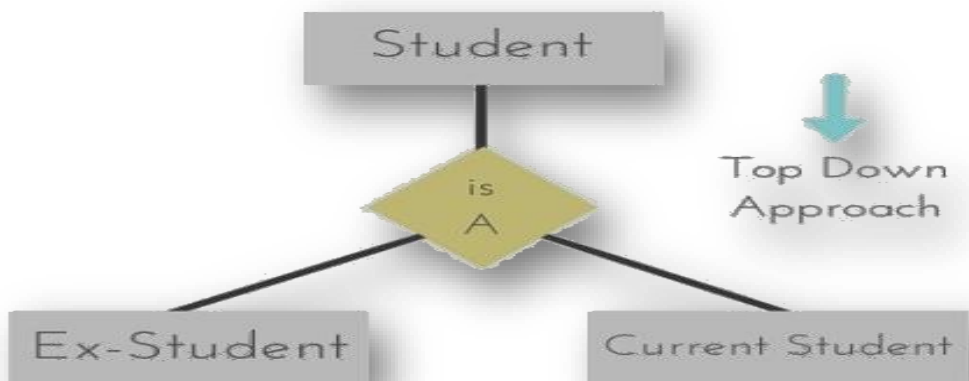
Generalization



For example, Saving and Current account types entities can be generalised and an entity with name Account can be created, which covers both.

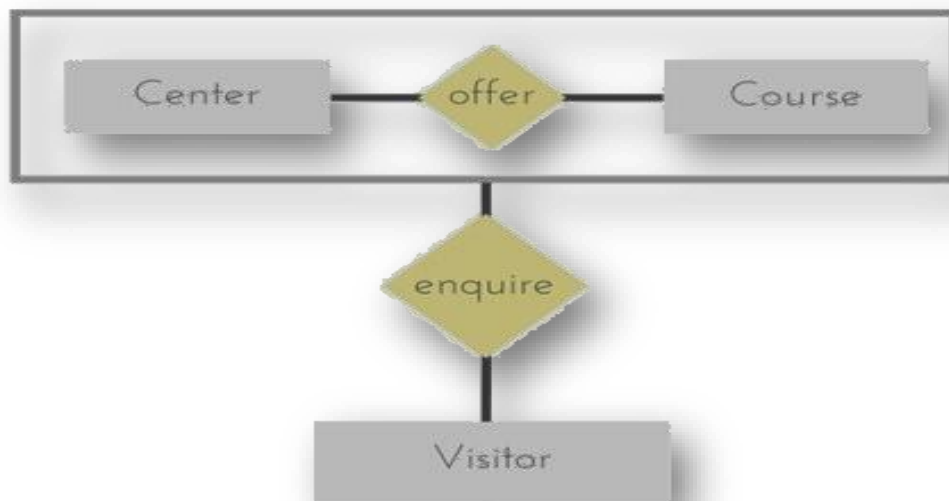
Specialization

Specialization is opposite to Generalization. It is a top-down approach in which one higher level entity can be broken down into two lower level entity. In specialization, a higher level entity may not have any lower-level entity sets, it's possible.



Aggregation

Aggregation is a process when relation between two entities is treated as a single entity.



In the diagram above, the relationship between Center and Course together, is acting as an Entity, which is in relationship with another entity Visitor. Now in real world, if a Visitor or a Student visits a Coaching Center, he/she will never enquire about the center only or just about the course, rather he/she will ask enquire about both.

2.5 Examples based on ER Diagram

Example 2.5.1 Draw the ER diagram for banking systems (home loan applications).

AU : Dec.-17, Marks 8

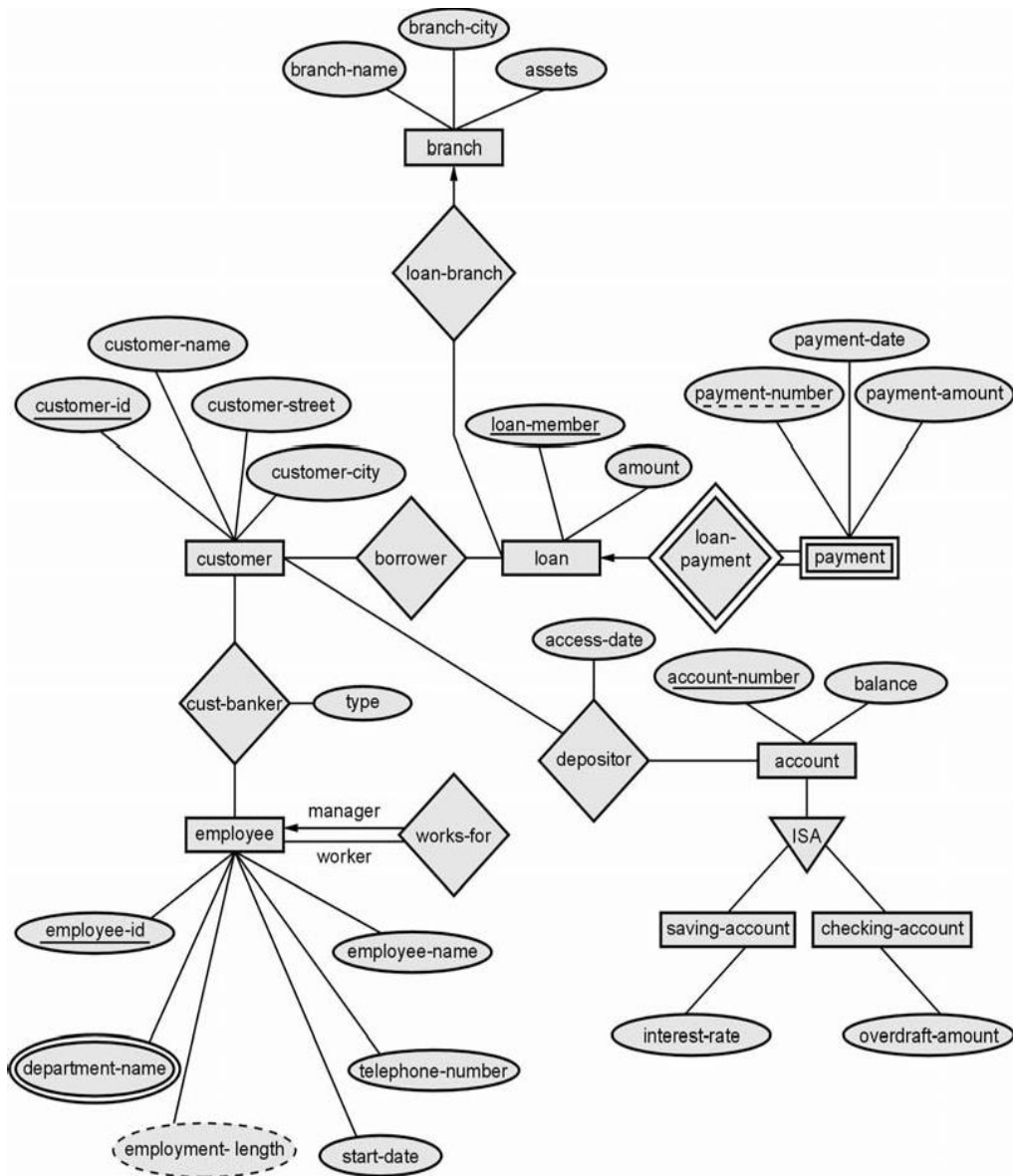
OR Draw an ER diagram corresponding to customers and loans.

AU : May.-14, Marks 8

OR Write short notes on : E-R diagram for banking system .

AU : Dec.-14, Marks 8

Solution :



Example 2.5.2 Consider the relation schema given in Figure. Design and draw an ER diagram that capture the information of this schema.

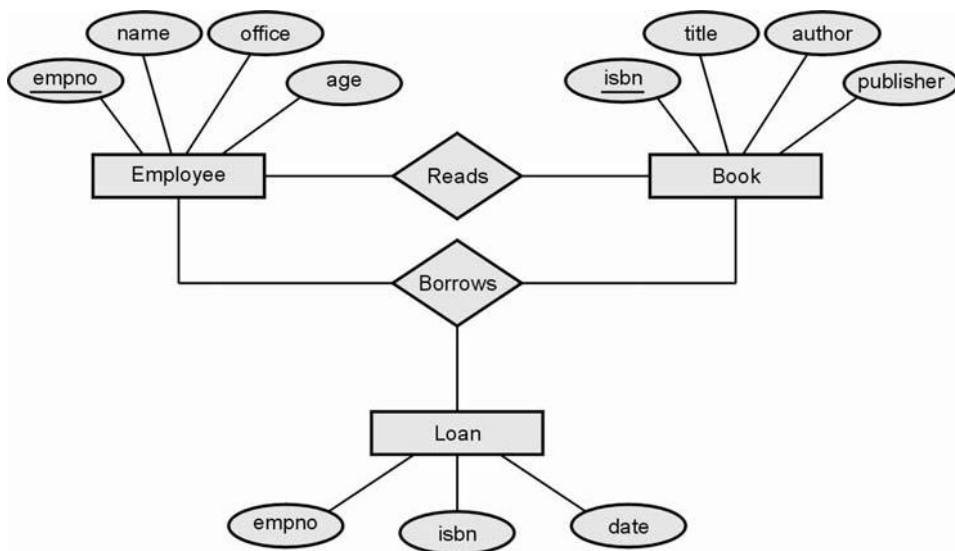
AU : May-17, Marks 5

Employee(empno,name,office,age)

Books(isbn,title,authors,publisher)

Loan(empno,isbn,date)

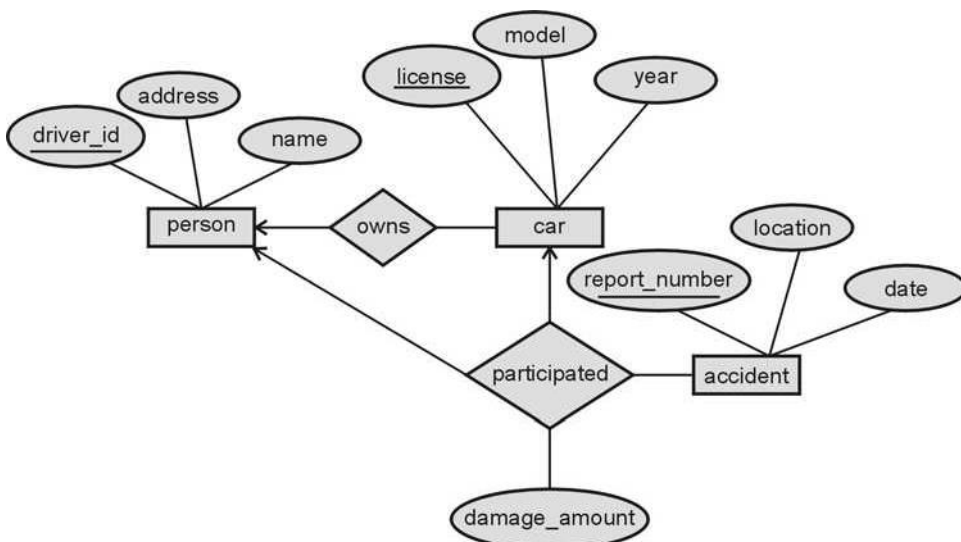
Solution :



Example 2.5.3 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. Each insurance policy covers one or more cars and has one or more premium payments associated with it. Each payment is for particular period of time and has an associated due date and date when the payment was received.

AU : Dec.-16, Marks 7

Solution :



Example 2.5.4 A car rental company maintains a database for all vehicles in its current fleet.

For all vehicles, it includes the vehicle identification number license number, manufacturer, model, date of purchase and color. Special data are included for certain types of vehicles.

Trucks : Cargo capacity

Sports cars : horsepower, renter age

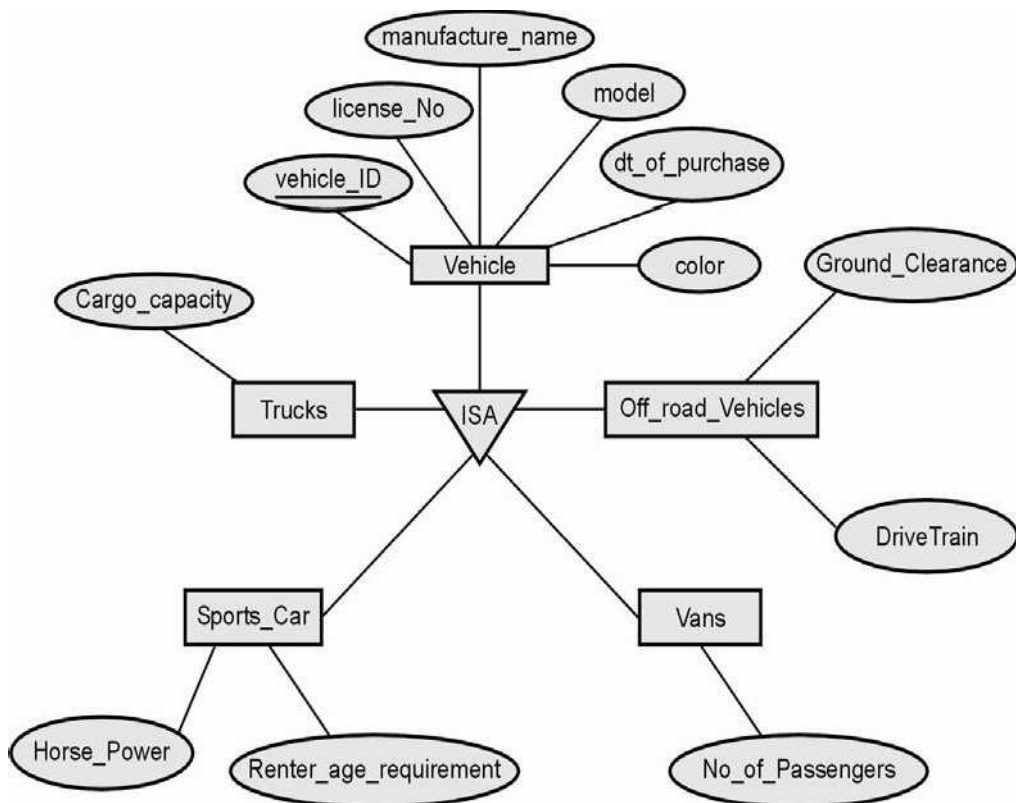
requirement Vans : number of passengers

Off-road vehicles : ground clearance, drivetrain (four-or two-

Construct an ER model for the car rental company

AU : Dec.-15, Marks 16

Solution :

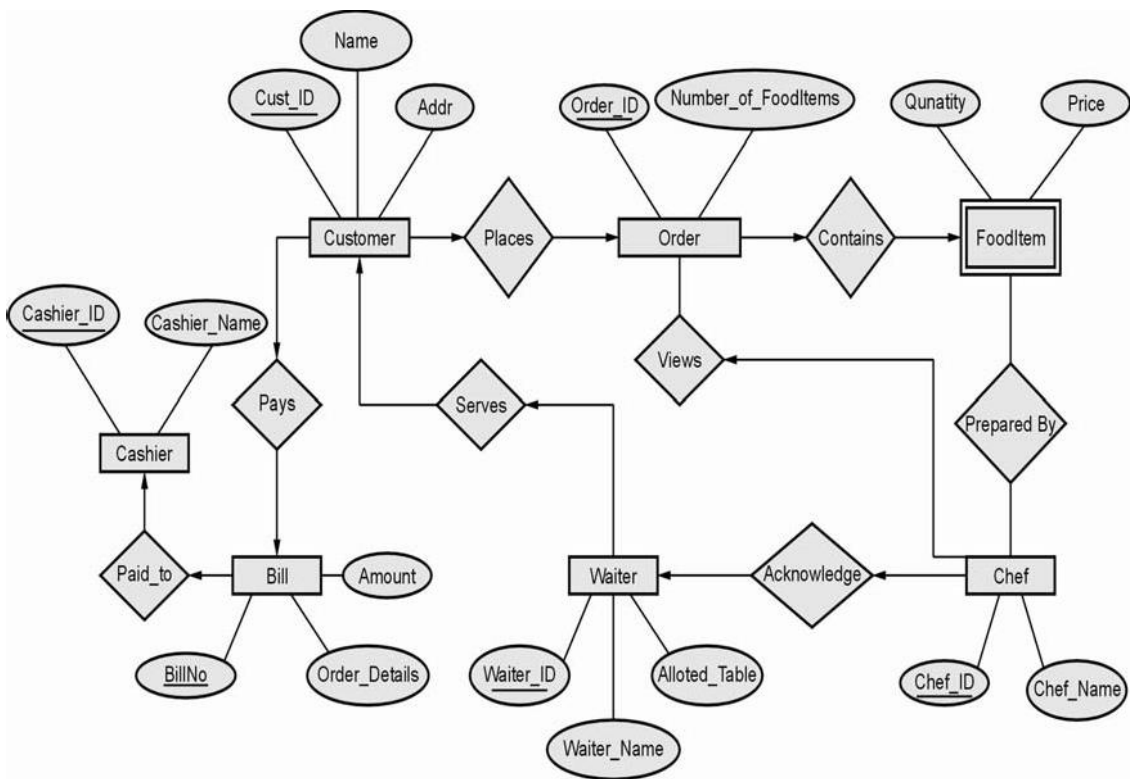


Example 2.5.5 Draw E-R diagram for the "Restaurant Menu Ordering System", which will facilitate the food items ordering and services within a restaurant. The entire restaurant scenario is detailed as follows. The customer is able to view the food items menu, call the waiter, place orders and obtain the final bill through the computer kept in their table. The Waiters through their wireless tablet PC are able to initialize a table for customers, control the table functions to assist customers, orders, send orders to food preparation staff (chef) and finalize the customer's bill. The Food preparation staffs (chefs), with their touch-display interfaces to the system, are able to view orders sent to the kitchen by waiters. During preparation they are able to let the waiter know the status of each item, and can send notifications when items are completed. The system should have full accountability and logging facilities, and should support supervisor actions to account for exceptional

circumstances, such as a meal being refunded or walked out on.

AU : May-15, Marks 16

Solution :



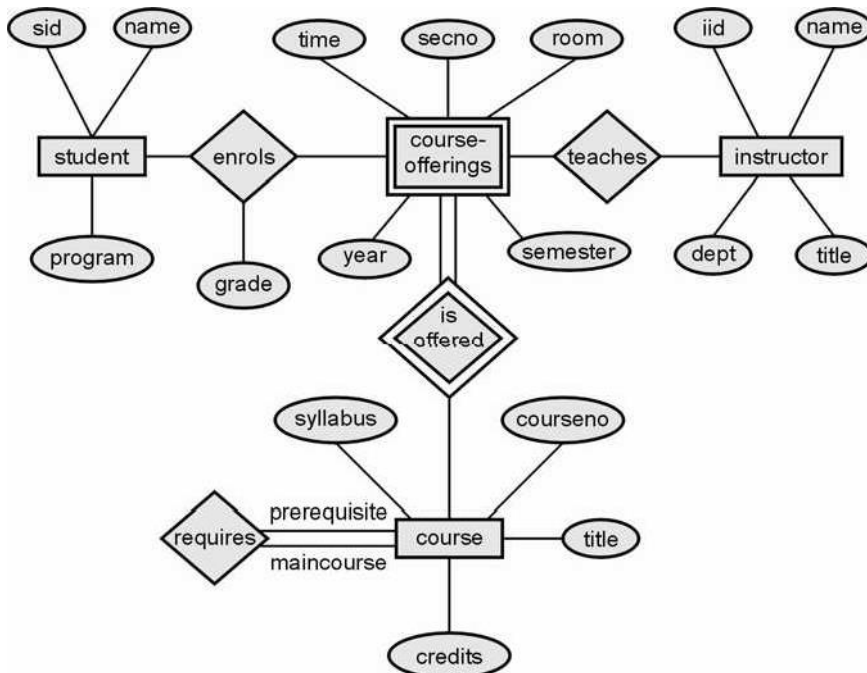
Example 2.5.6 A university registrar's office maintains data about the following entities :

- (1) courses, including number, title, credits, syllabus, and prerequisites;
- (2) course offerings, including course number, year, semester, section number, instructor(s), timings, and classroom;
- (3) students, including student-id, name, and program; and
- (4) instructors, including identification number, name, department, and title.

Further, the enrollment of students in courses and grades awarded to students in each course they are enrolled for must be appropriately modeled. Construct an E-R diagram for the registrar's office. Document all assumptions that you make about the mapping constraints.

AU : Dec.-13, Marks 10

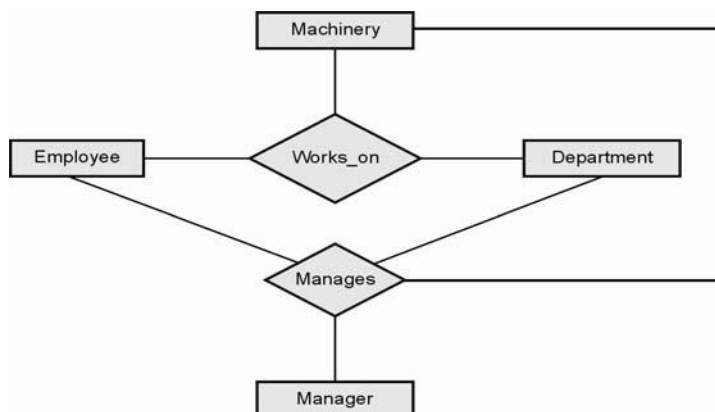
Solution :



Example 2.5.7 What is aggregation in ER model ? Develop an ER diagram using aggregation that captures following information : Employees work for projects. An employee working for particular project uses various machinery. Assume necessary attributes. State any assumptions you make. Also discuss about the ER diagram you have designed. **AU : Dec.-11, Marks 8**

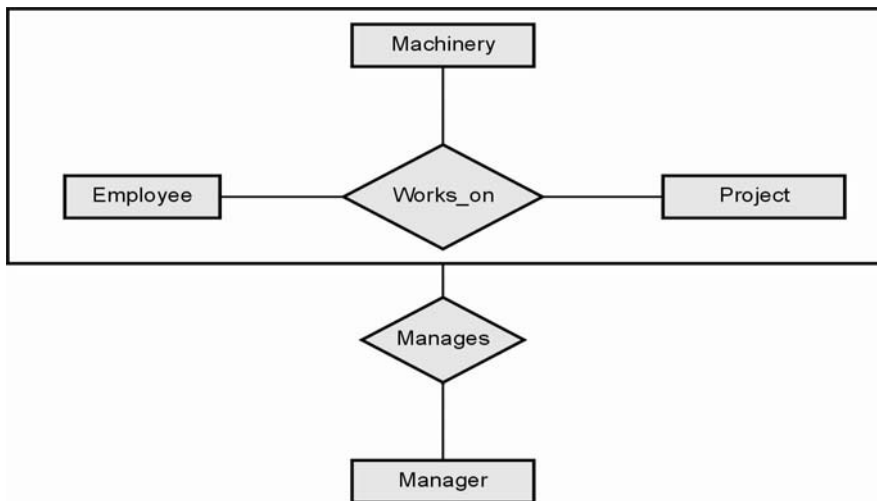
Solution : Aggregation : Refer section 2.4.3.

ER Diagram : The ER diagram for above described scenario can be drawn as follows -



The above ER model contains the redundant information, because every Employee, Project, Machinery combination in **works_on** relationship is also considered in **manages**

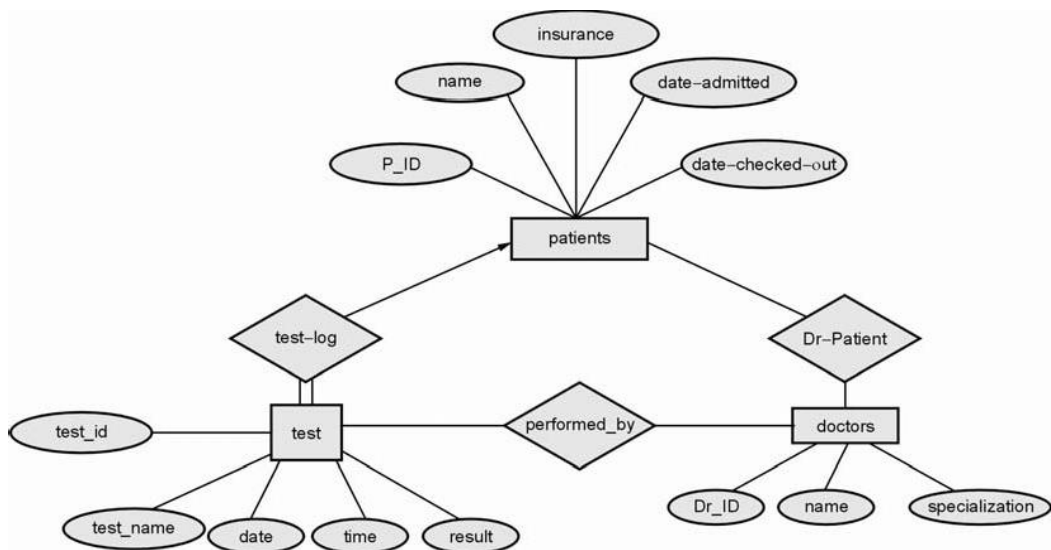
relationship. To avoid this redundancy problem we can make use of aggregation relationship in ER diagram as follows -



We can then create a binary relationship **manages** for between **Manager** and (**Employee, Project, Machinery**).

Example 2.5.8 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. **AU : Dec.-07, Marks 8**

Solution :



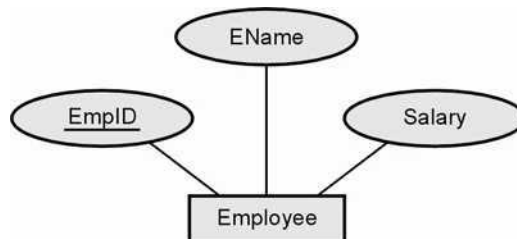
2.6 ER to Relational Mapping

AU : May-17, Marks 13

In this section we will discuss how to map various ER model constructs to Relational Model construct.

2.6.1 Mapping of Entity to Table

- An entity set is mapped to a relation in a straightforward way.
- Each attribute of entity set becomes an attribute of the table.
- The primary key attribute of entity set becomes an entity of the table.
- For example - Consider following ER diagram.



The converted employee table is as follows -

EmpID	EName	Salary
201	Poonam	30000
202	Ashwini	35000
203	Sharda	40000

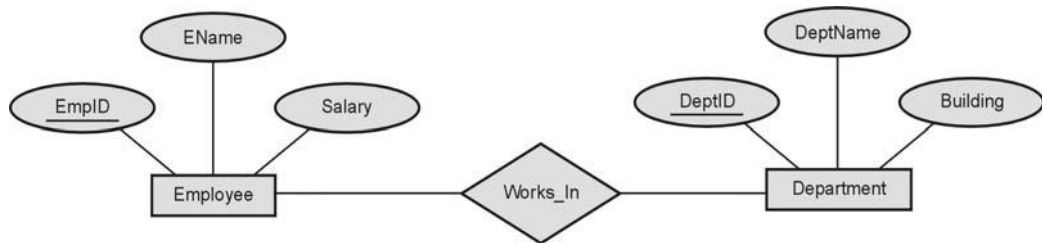
The SQL statement captures the information for above ER diagram as follows -

```
CREATE TABLE Employee( EmpID CHAR(11),
    EName CHAR(30),
    Salary INTEGER,
    PRIMARY KEY(EmpID))
```

2.6.2 Mapping Relationship Sets to Tables

- Create a table for the relationship set.
- Add all primary keys of the participating entity sets as fields of the table.
- Add a field for each attribute of the relationship.
- Declare a primary key using all key fields from the entity sets.

- Declare foreign key constraints for all these fields from the entity sets.
- For example - Consider following ER model



The SQL statement captures the information for relationship present in above ER diagram as follows -

```

CREATE TABLE Works_In (EmpID CHAR(11),
                        DeptID CHAR(11),
                        EName CHAR(30),
                        Salary INTEGER,
                        DeptName CHAR(20),
                        Building CHAR(10),
                        PRIMARY KEY(EmpID,DeptID),
                        FOREIGN KEY (EmpID) REFERENCES Employee,
                        FOREIGN KEY (DeptID) REFERENCES Department
                        )
  
```

University Question

1. Discuss the correspondence between the ER model construct and the relational model constructs. Show how each ER model construct can be mapped to the relational model. Discuss the option for mapping EER construct.

AU : May-17, Marks 13

Part II Relational Database Design

2.7 Concept of Relational Database Design

- There are two primary goals of relational database design – i) to generate a set of relation schemas that allows us to store information without unnecessary redundancy, and ii) to allows us to retrieve information easily.
- For achieving these goals, the database design need to be **normalized**. That means we have to check whether the schema is it normal form or not.
- For checking the normal form of the schema, it is necessary to check the functional dependencies and other data dependencies that exists within the schema.

Hence before letting us know what the normalization means, it is necessary to understand the concept of functional dependencies.

2.8 Functional Dependency

The functional dependency is a relationship that exists between two attributes. It typically exists between the primary key and non-key attribute within a table.

$X \rightarrow Y$

The left side of FD is known as a determinant, the right side of the production is known as a dependent.

For example:

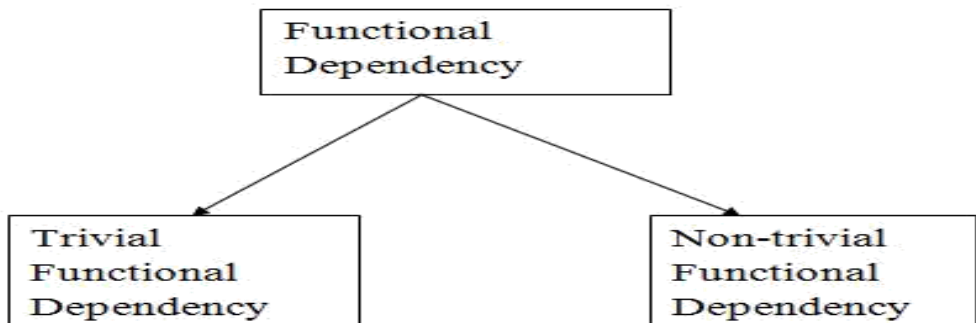
Assume we have an employee table with attributes: Emp_Id, Emp_Name, Emp_Address.

Here Emp_Id attribute can uniquely identify the Emp_Name attribute of employee table because if we know the Emp_Id, we can tell that employee name associated with it.

Functional dependency can be written as:

$$\text{Emp_Id} \rightarrow \text{Emp_Name}$$

Types of Functional dependency



Trivial functional dependency

- $A \rightarrow B$ has trivial functional dependency if B is a subset of A .
- The following dependencies are also trivial like: $A \rightarrow A$, $B \rightarrow B$

Example:

Consider a table with two columns Employee_Id and Employee_Name.

$\{\text{Employee_id}, \text{Employee_Name}\} \rightarrow \text{Employee_Id}$ is a trivial functional dependency as Employee_Id is a subset of $\{\text{Employee_Id}, \text{Employee_Name}\}$.

Also, $\text{Employee_Id} \rightarrow \text{Employee_Id}$ and $\text{Employee_Name} \rightarrow \text{Employee_Name}$ are trivial dependencies too.

Non-trivial functional dependency

$A \rightarrow B$ has a non-trivial functional dependency if B is not a subset of A .

When $A \cap B$ is NULL, then $A \rightarrow B$ is called as complete non-trivial.

Example:

$$\begin{array}{lcl} \text{ID} & \rightarrow & \text{Name,} \\ \text{Name} & \rightarrow & \text{DOB} \end{array}$$

2.9 Concept of Redundancy and Anomalies

Definition : Redundancy is a condition created in database in which same piece of data is held at two different places.

Redundancy is at the root of several problems associated with relational schemas.

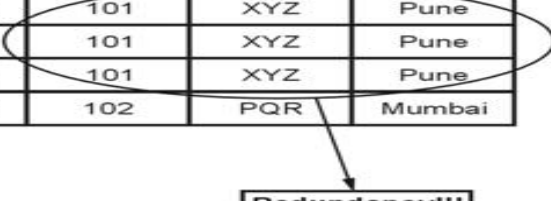
Problems caused by redundancy : Following problems can be caused by redundancy-

- i) **Redundant storage :** Some information is stored repeatedly.
- ii) **Update anomalies :** If one copy of such repeated data is updated then inconsistency is created unless all other copies are similarly updated.
- iii) **Insertion anomalies :** Due to insertion of new record repeated information get added to the relation schema.
- iv) **Deletion anomalies :** Due to deletion of particular record some other important information associated with the deleted record get deleted and thus we may lose some other important information from the schema.

Example : Following example illustrates the above discussed anomalies or redundancy problems

Consider following Schema in which all possible information about Employee is stored.

EmpID	ENAME	Salary	DeptID	DeptName	DeptLoc
1	AAA	10000	101	XYZ	Pune
2	BBB	20000	101	XYZ	Pune
3	CCC	30000	101	XYZ	Pune
4	DDD	40000	102	PQR	Mumbai



Redundancy!!!

- 1) **Redundant storage :** Note that the information about **DeptID**, **DeptName** and **DeptLoc** is repeated.
- 2) **Update anomalies :** In above table if we change **DeptLoc** of Pune to Chennai, then it will result **inconsistency** as for **DeptID 101** the **DeptLoc** is Pune. Or otherwise, we need to **update multiple copies** of **DeptLoc** from Pune to Chennai. Hence this is an update anomaly.
- 3) **Insertion anomalies :** For above table if we want to add new tuple say (5, EEE,50000) for **DeptID 101** then it will cause repeated information of (101, XYZ,Pune) will occur.
- 4) **Deletion anomalies :** For above table, if we delete a record for **EmpID 4**, then automatically information about the **DeptID 102, DeptName PQR** and **DeptLoc Mumbai** will get deleted and one may not be aware about **DeptID 102**. This causes deletion anomaly.

2.10 Decomposition

AU : Dec.-17, Marks 7

- Decomposition is the process of breaking down one table into multiple tables.
- **Formal definition of decomposition is -**
- A decomposition of relation Schema R consists of replacing the relation Schema by two relation schema that each contain a subset of attributes of R and together include all attributes of R by storing projections of the instance.
- For example - Consider the following table

Employee_Department table as follows -

Eid	Ename	Age	City	Salary	Deptid	DeptName
E001	ABC	29	Pune	20000	D001	Finance
E002	PQR	30	Pune	30000	D002	Production
E003	LMN	25	Mumbai	5000	D003	Sales
E004	XYZ	24	Mumbai	4000	D004	Marketing
E005	STU	32	Hyderabad	25000	D005	Human Resource

We can decompose the above relation Schema into two relation schemas as **Employee** (Eid, Ename, Age, City, Salary) and **Department** (Deptid, Eid, DeptName). as follows -

Employee Table

Eid	Ename	Age	City	Salary
E001	ABC	29	Pune	20000
E002	PQR	30	Pune	30000
E003	LMN	25	Mumbai	5000
E004	XYZ	24	Mumbai	4000
E005	STU	32	Hyderabad	25000

Department Table

Deptid	Eid	DeptName
D001	E001	Finance
D002	E002	Production
D003	E003	Sales
D004	E004	Marketing
D005	E005	Human Resource

- The decomposition is used for eliminating redundancy.

- **For example :** Consider following relation **Schema R** in which we assume that the grade determines the salary, the redundancy is caused

Schema R

Name	eid	deptname	Grade	Salary
AAA	121	Accounts	2	8000
AAA	132	Sales	3	7000
BBB	101	Marketing	4	7000
CCC	106	Purchase	2	8000

Redundancy!!!

- Hence, the above table can be decomposed into two Schema S and T as follows :

Name	eid	deptname	Grade
AAA	121	Accounts	2
AAA	132	Sales	3
BBB	101	Marketing	4
CCC	106	Purchase	2

Grade	Salary
2	8000
3	7000
4	7000
2	8000

Problems Related to Decomposition :

Following are the potential problems to consider :

- 1) Some queries become more **expensive**.
- 2) Given instances of the decomposed relations, we may not be able to reconstruct the corresponding instance of the original relation!
- 3) Checking some dependencies may require joining the instances of the decomposed relations.
- 4) There may be loss of information during decomposition.

Properties Associated With Decomposition

There are two properties associated with decomposition and those are –

- 1) **Loss-less Join or non Loss Decomposition :** When all information found in the original database is preserved after decomposition, we call it as loss less or non loss decomposition.
- 2) **Dependency Preservation :** This is a property in which the constraints on the original table can be maintained by simply enforcing some constraints on each of the smaller relations.

University Question

1. Differentiate between lossless join decomposition and dependency preserving decomposition.

AU : Dec.-17, Marks 7

- Normalization is the process of reorganizing data in a database so that it meets **two basic requirements**:
 - 1) There is **no redundancy** of data (all data is stored in only one place), and
 - 2) **data dependencies** are logical (all related data items are stored together)
- The normalization is important because it allows database to take up **less disk space**.
- It also help in increasing the **performance**.

2.11.1 First Normal Form

The table is said to be in 1NF if it follows following rules -

- i) It should only have single (atomic) valued attributes/columns.
- ii) Values stored in a column should be of the same domain
- iii) All the columns in a table should have unique names.
- iv) And the order in which data is stored, does not matter.

Consider following Student table

Student

sid	sname	Phone
1	AAA	11111 22222
2	BBB	33333
3	CCC	44444 55555

As there are multiple values of phone number for sid 1 and 3, the above table is not in 1NF. We can make it in 1NF. The conversion is as follows -

sid	sname	Phone
1	AAA	11111
1	AAA	22222
2	BBB	33333
3	CCC	44444
3	CCC	55555

2.11.2 Second Normal Form

Before understanding the second normal form let us first discuss the concept of partial functional dependency and prime and non prime attributes.

Concept of Partial Functional Dependency

Partial dependency means that a nonprime attribute is functionally dependent on part of a candidate key.

For example : Consider a relation $R(A,B,C,D)$ with functional dependency $\{AB \rightarrow CD, A \rightarrow C\}$

Here (AB) is a candidate key because

$$(AB)^+ = \{ABCD\} = R$$

Hence $\{A,B\}$ are prime attributes and $\{C,D\}$ are non prime attribute. In $A \rightarrow C$, the non prime attribute C is dependent upon A which is actually a part of candidate key AB . Hence due to $A \rightarrow C$ we get partial functional dependency.

Prime and Non Prime Attributes

- **Prime attribute** : An attribute, which is a part of the candidate-key, is known as a prime attribute.
- **Non-prime attribute** : An attribute, which is not a part of the prime-key, is said to be a non-prime attribute.
- **Example : Consider a Relation** $R=\{A,B,C,D\}$ and candidate key as AB , the Prime attributes : A, B

Non Prime attributes : C, D

The Second Normal Form

For a table to be in the Second Normal Form, following conditions must be followed

- i) It should be in the First Normal form.
- ii) It should not have partial functional dependency.

For example : Consider following table in which every information about a the Student is maintained in a table such as student id(sid), student name(sname), course id(cid) and course name(cname).

Student_Course

sid	sname	cid	cname
1	AAA	101	C
2	BBB	102	C++
3	CCC	101	C
4	DDD	103	Java

This table is not in 2NF. For converting above table to 2NF we must follow the following steps -

Step 1 : The above table is in 1NF.

Step 2 : Here **sname** and **sid** are associated similarly **cid** and **cname** are associated with each other. Now if we delete a record with **sid=2**, then automatically the course C++ will also get deleted.

Thus,

sid->sname or **cid->cname** is a partial functional dependency, because {**sid,cid**} should be essentially a candidate key for above table. Hence to bring the above table to 2NF we must decompose it as follows :

Student

sid	sname	cid
1	AAA	101
2	BBB	102
3	CCC	101
4	DDD	103

Here candidate key is
(sid,cid)
and
(sid,cid)->sname

Course

cid	cname
101	C
102	C++
101	C
103	Java

Here candidate key is
cid
Here cid->cname

Thus now table is in 2NF as there is no partial functional dependency

2.11.3 Third Normal Form

Before understanding the third normal form let us first discuss the concept of transitive dependency, super key and candidate key

Concept of Transitive Dependency

A functional dependency is said to be transitive if it is indirectly formed by two functional dependencies. For example -

$X \rightarrow Z$ is a transitive dependency if the following functional dependencies hold true :

$X \rightarrow Y$

$Y \rightarrow Z$

Concept of Super key and Candidate Key

Superkey : A super key is a set or one of more columns (attributes) to uniquely identify rows in a table.

Candidate key : The minimal set of attribute which can uniquely identify a tuple is known as candidate key.

For example consider following table

RegID	RollNo	Sname
101	1	AAA
102	2	BBB
103	3	CCC
104	4	DDD

Superkeys

- {RegID}
- {RegID, RollNo}
- {RegID, Sname}
- {RollNo, Sname}
- {RegID, RollNo, Sname}

Candidate Keys

- {RegID}
- {RollNo}

Third Normal Form

A table is said to be in the Third Normal Form when,

- i) It is in the Second Normal form.(i.e. it does not have partial functional dependency)
- ii) **It doesn't have transitive dependency.**

In other words 3NF can be defined as : A table is in 3NF if it is in 2NF and for each functional dependency

$X \rightarrow Y$

at least one of the following conditions hold :

- i) X is a super key of table
- ii) Y is a prime attribute of table

For example : Consider following table **Student_details** as follows -

sid	sname	zipcode	cityname	state
1	AAA	11111	Pune	Maharashtra
2	BBB	22222	Surat	Gujarat
3	CCC	33333	Chennai	Tamilnadu
4	DDD	44444	Jaipur	Rajasthan
5	EEE	55555	Mumbai	Maharashtra

Here

Super keys : {sid},{sid,sname},{sid,sname,zipcode}, {sid,zipcode,cityname}... and so on.

Candidate keys : {sid}

Non-Prime attributes : {sname,zipcode,cityname,state}

The dependencies can be denoted as

sid->sname

sid->zipcode

zipcode->cityname

cityname->state

The above denotes the transitive dependency. Hence above table is not in 3NF.

We can convert it into 3NF as follows :

Student

sid	sname	zipcode
1	AAA	11111
2	BBB	22222
3	CCC	33333
4	DDD	44444
5	EEE	55555

Zip

zipcode	cityname	state
11111	Pune	Maharashtra
22222	Surat	Gujarat
33333	Chennai	Tamilnadu
44444	Jaipur	Rajasthan
55555	Mumbai	Maharashtra

1. What is database normalization ? Explain the first normal form, second normal form and third normal form.

AU : May-18, Marks 13; Dec.-15, Marks 16

2. What are normal forms. Explain the types of normal form with an example.

AU : Dec.-14, Marks 16

2.12 Boyce / Codd Normal Form (BCNF)

Boyce and Codd Normal Form is a **higher version** of the Third Normal form. This form deals with certain type of anomaly that is not handled by 3NF.

A 3NF table which **does not have multiple overlapping** candidate keys is said to be in BCNF.

Or in other words,

For a table to be in BCNF, following conditions must be satisfied :

- R must be in 3rd Normal Form
- For each functional dependency ($X \rightarrow Y$), X should be a super Key. In simple words if Y is a prime attribute then X can not be non prime attribute.

For example - Consider following table that represents that a Student enrollment for the course -

Enrollment Table

sid	course	Teacher
1	C	Ankita
1	Java	Poonam
2	C	Ankita
3	C++	Supriya
4	C	Archana

From above table following observations can be made :

- One student can enroll for multiple courses. For example student with sid=1 can enroll for C as well as Java.
- For each course, a teacher is assigned to the student.
- There can be multiple teachers teaching one course for example course C can be taught by both the teachers namely - Ankita and Archana.
- The candidate key for above table can be (sid,course), because using these two columns we can find
- The above table holds following dependencies
 - (sid,course)->Teacher
 - Teacher->course
- The above table is not in BCNF because of the dependency **teacher->course**. Note that the teacher is not a superkey or in other words, **teacher** is a non prime attribute and **course** is a prime attribute and non-prime attribute derives the prime attribute.
- To convert the above table to BCNF we must decompose above table into Student and Course tables

Student

sid	Teacher
1	Ankita
1	Poonam
2	Ankita
3	Supriya
4	Archana

Course

Teacher	course
Ankita	C
Poonam	Java
Ankita	C
Supriya	C++
Archana	C

Now the table is in BCNF

2.13 Multivalued Dependencies and Fourth Normal Form

AU : May-14, Dec.-16, Marks 16

Concept of Multivalued Dependencies

- A table is said to have multi-valued dependency, if the following conditions are true,
 - 1) For a dependency $A \twoheadrightarrow B$, if for a single value of A, **multiple values** of B exists, then the table may have multi-values dependency.
 - 2) Also, a table should have **at-least 3 columns** for it to have a multi-valued dependency.

- 3) And, for a relation R(A,B,C), if there is a multi-valued **dependency between**, A and B, then B and C should be independent of each other.

~~If all these conditions are true for any relation(table), it is said to have multi-valued dependency.~~

- In simple terms, if there are two columns A and B - and for column A if there are multiple values of column B then we say that MVD exists between A and B
- The multivalued dependency is denoted by \twoheadrightarrow
- If there exists a multivalued dependency then the table is **not in 4th normal form**.
- **For example :** Consider following table for information about student

Student

sid	Course	Skill
1	C C++	English German
2	Java	English French

Here sid =1 leads to multiple values for courses and skill. Following table shows this

sid	Course	Skill
1	C	English
1	C++	German
1	C	German
1	C++	English
2	Java	English
2	Java	French

Here **sid** and **course** are dependent but the **Course** and **Skill** are independent. The multivalued dependency is denoted as :

sid \twoheadrightarrow Course

sid \twoheadrightarrow Skill

Fourth Normal Form

Definition : For a table to satisfy the Fourth Normal Form, it should satisfy the following two conditions :

- 1) It should be in the Boyce-Codd Normal Form(BCNF).
- 2) And, the table should not have any multi-valued dependency.

For example : Consider following student relation which is not in 4NF as it contains multivalued dependency.

Student Table

sid	Course	Skill
1	C	English
1	C++	German
1	C	German
1	C++	English
2	Java	English
2	Java	French

Now to convert the above table to 4NF we must decompose the table into following two tables.

Student_Course TableKey : (sid, Course)

sid	Course
1	C
1	C++
2	Java

Student_Skill Table

Key : (sid, Skill)

sid	Skill
1	English
1	German
2	English
2	French

Thus the tables are now in 4NF.

University Questions

1. Explain first normal form, second normal form, third normal form and BCNF with example.

AU : Dec.-16, Marks 13

2. Explain Boyce Codd Normal form and fourth normal form with suitable example.

AU : May-14, Marks 16

2.14 Join Dependencies and Fifth Normal Form

Concept of Join Dependencies

- Join decomposition is a further generalization of Multivalued dependencies.
- If the join of R1 and R2 over C is equal to relation R, then we can say that a Join Dependency (JD) exists.
- Where R1 and R2 are the decompositions R1(A, B, C) and R2(C, D) of a given relations R (A, B, C, D).
- Alternatively, R1 and R2 are a lossless decomposition of R.

- A JD $\bowtie \{R_1, R_2, \dots, R_n\}$ is said to hold over a relation R if R_1, R_2, \dots, R_n is a lossless-join decomposition.
- The $*(A, B, C, D), (C, D)$ will be a JD of R if the join of join's attribute is equal to the relation R.
- Here, $*(R_1, R_2, R_3)$ is used to indicate that relation R_1, R_2, R_3 and so on are a JD of R.

Concept of Fifth Normal Form

The database is said to be in 5NF if -

- It is in 4th Normal Form
- If we can decompose table further to eliminate redundancy and anomalies and when we rejoin the table we should not be losing the original data or get a new record (**join Dependency Principle**)

The fifth normal form is also called as **project join normal form**

For example - Consider following table

Seller	Company	Product
Rupali	Godrej	Cinthol
Sharda	Dabur	Honey
Sharda	Dabur	HairOil
Sharda	Dabur	Rosewater
Sunil	Amul	Icecream
Sunil	Britania	Biscuits

Here we assume the keys as {Seller, Company, Product}

The above table has multivalued dependency as

$\text{Seller} \twoheadrightarrow \{\text{Company, Product}\}$. Hence table is not in 4th Normal Form. To make the above table in 4th normal form we decompose above table into two tables as

Seller_Company

Seller	Company
Rupali	Godrej
Sharda	Dabur
Sunil	Amul
Sunil	Britania

Seller_Product

Seller	Product
Rupali	Cinthol
Sharda	Honey
Sharda	HairOil
Sharda	RoseWater
Sunil	Icecream
Sunil	Biscuits

The above table is in 4th Normal Form as there is no multivalued dependency. But it is not in 5th normal form because if we join the above two table we may get

Seller	Company	Product
Rupali	Godrej	Cinthol
Sharda	Dabur	Honey
Sharda	Dabur	HairOil
Sharda	Dabur	Rosewater
Sunil	Amul	Icecream
Sunil	Amul	Biscuits
Sunil	Britania	Icecream
Sunil	Britania	Biscuits

Newly added records
which are not present in
original table

To avoid the above problem we can decompose the tables into three tables as
Seller_Company, Seller_Product, and Company Product table

Seller_Company		Seller_Product		Company_Product	
Seller	Company	Seller	Product	Company	Product
Rupali	Godrej	Rupali	Cinthol	Godrej	Cinthol
Sharda	Dabur	Sharda	Honey	Dabur	Honey
Sunil	Amul	Sharda	HairOil	Dabur	HairOil
Sunil	Britania	Sharda	RoseWater	Dabur	RoseWater
		Sunil	Icecream	Amul	Icecream
		Sunil	Biscuit	Britania	Biscuit

Thus the table in in 5th normal form.