NAME- JYOTI
ENROLLMENTNO. - 05701032024
IT -1
SUBJECT- DATABASE MANAGEMENT SYSTEM

DBMS ASSIGNMENT

Topic: ER DIAGRAM INTO A RELATIONAL MODEL

1. Introduction

A **Database Management System (DBMS)** is a software system that enables the creation, management, and efficient manipulation of databases. One of the most important stages in database design is to first create a **conceptual model**, usually represented by an **Entity-Relationship (ER) Diagram**, and then translate it into a **Relational Model**.

The ER diagram provides a **high-level, visual representation** of the system, focusing on entities, their attributes, and relationships. However, computers cannot directly implement an ER diagram, so it must be translated into a **Relational Model** that can be executed in DBMS platforms such as MySQL, Oracle, or PostgreSQL.

This assignment discusses **how ER diagrams are converted into relational models**, explains rules, examples, and challenges, and highlights their importance in real-life applications.

2. What is an ER Diagram?

An **Entity-Relationship (ER) Diagram** is a **conceptual modeling tool** that describes:

- **Entities**: Real-world objects (e.g., Student, Teacher, Book).
- **Attributes**: Properties of entities (e.g., Student Name, Roll Number).
- **Relationships**: Associations between entities (e.g., Student borrows Book).

ER diagrams make database design **clear, structured, and easy to communicate** before moving to technical implementation.

Example of an ER Diagram

Consider a simple ER diagram with two entity sets: Customer and Loan, connected by the borrower relationship:

Customer Attributes: customer-id, customer-name, customer-street, customer-city
Loan Attributes: loan-number, amount

- Attributes that are part of the primary key are underlined.
- The relationship set borrower can be many-to-many, one-to-many, many-to-one, or one-to-one, which is indicated using directed or undirected lines.

Relationship Types:

- Directed line from borrower → loan → borrower is one-to-one or many-to-one from customer to loan.
- Undirected line → borrower is many-to-many or one-to-many.

Figures:

1. Many-to-many relationship: borrower (customer ↔ loan)
2. One-to-many: arrow points to “one” side
3. Many-to-one: arrow points to the “one” entity
4. One-to-one: arrows on both sides

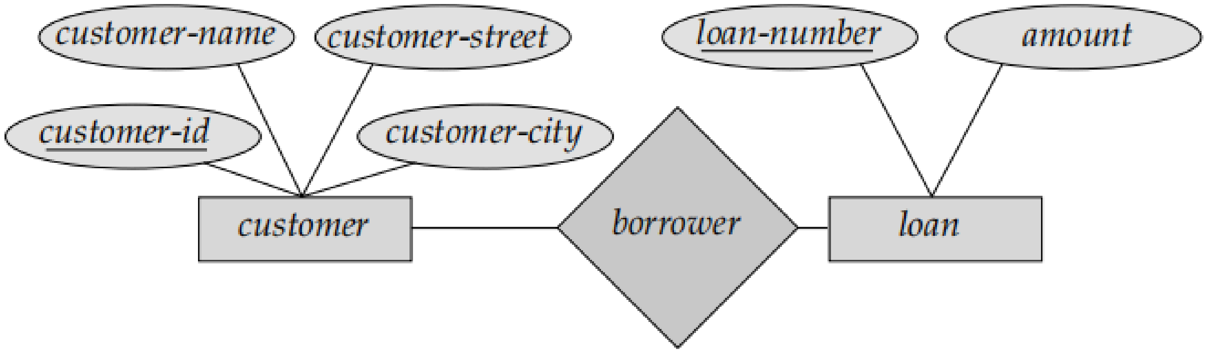

Figure 2.8 E-R diagram corresponding to customers and loans.

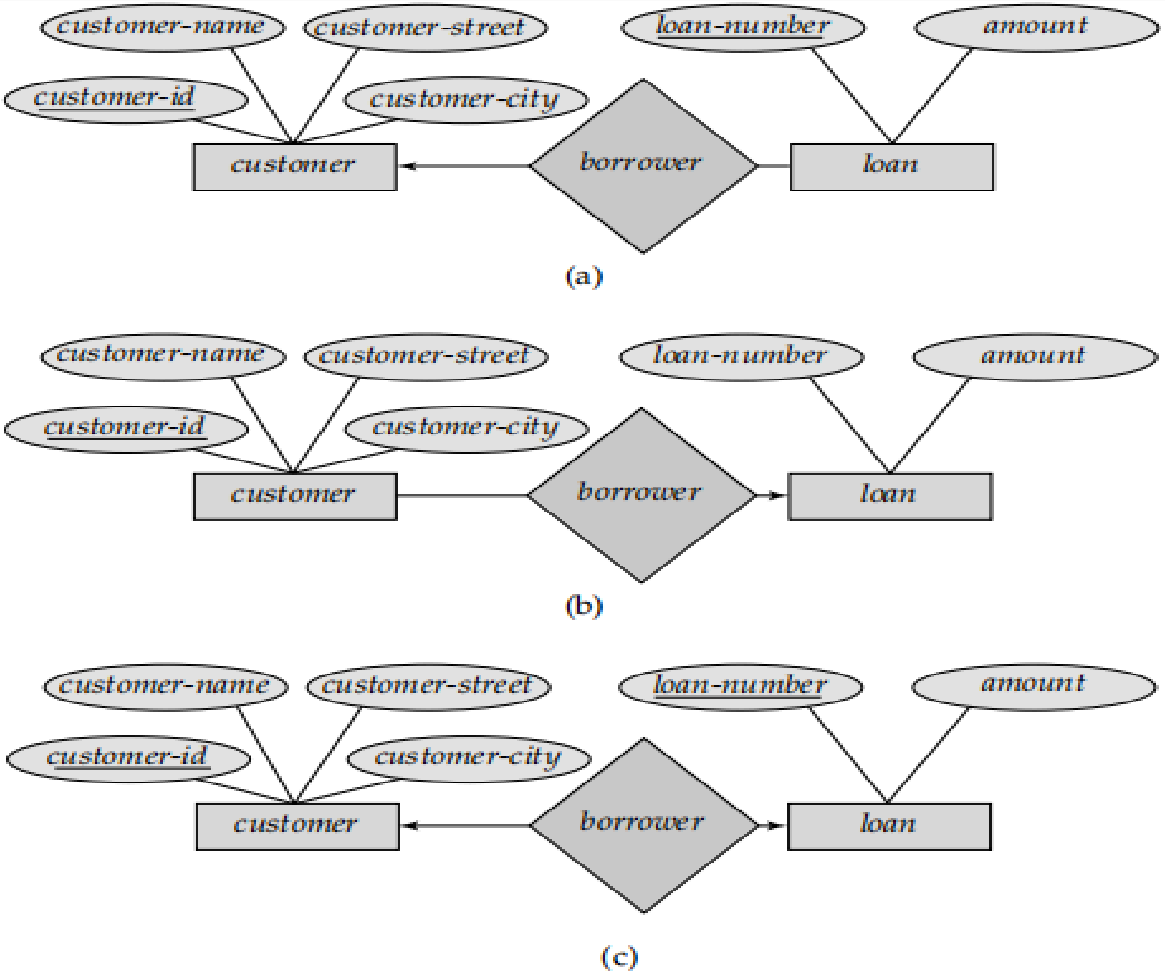

Figure 2.9 Relationships. (a) one to many. (b) many to one. (c) one-to-one.

3. What is a Relational Model?

The **Relational Model**, introduced by *E.F. Codd in 1970*, is the most widely used model for designing databases.

- Data is stored in **tables (relations)**.
- Each table consists of **rows (tuples)** and **columns (attributes)**.
- Relationships are represented using **primary keys and foreign keys**.

It provides a **mathematical and structured approach** to represent data in a way that computers can store and query efficiently.

4. Steps to Convert ER Diagram into Relational Model

Step 1: Map Regular Entities

- Each **entity** in the ER diagram becomes a **table** in the relational model.
- Attributes of the entity become columns of the table.
- The **primary key** uniquely identifies each row.

Example:
Entity: **STUDENT (RollNo, Name, Age, Course)**
Relational Table:

```
STUDENT(RollNo [PK], Name, Age, Course)
```

Step 2: Map Weak Entities

- Weak entities depend on strong entities.
- They include the **primary key of the strong entity** as a **foreign key**.

Example:
Weak Entity: **DEPENDENT (DepName, Age)** depends on **EMPLOYEE (EmpID)**.

```
EMPLOYEE(EmpID [PK], Name, Department)

DEPENDENT(EmpID [FK], DepName [PK], Age)
```

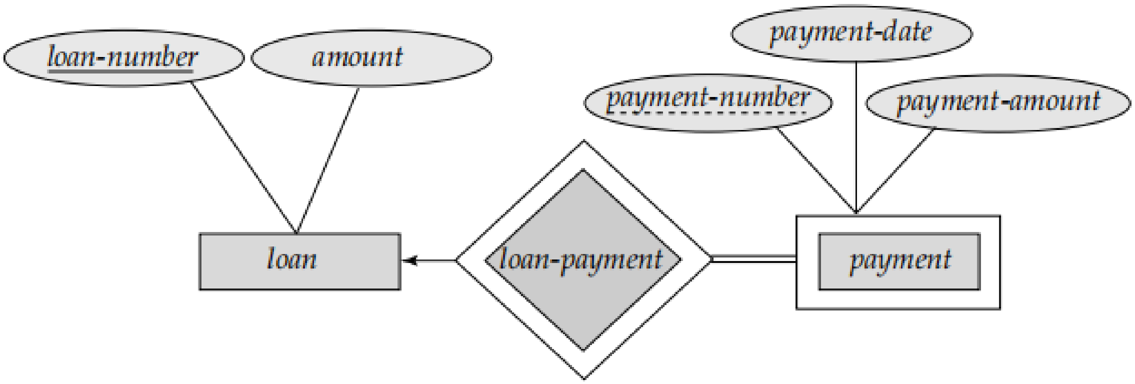

Figure 2.16 E-R diagram with a weak entity set.

Step 3: Map Relationships

- **1:1 Relationship** → Foreign key placed in either table.
- **1:N Relationship** → Foreign key placed in the "many" side.
- **M:N Relationship** → Create a **new table** with foreign keys of both entities.

Example (M:N):
STUDENT (RollNo, Name) and COURSE (CourseID, Title).
Relationship: STUDENT takes COURSE.

```
STUDENT(RollNo [PK], Name)

COURSE(CourseID [PK], Title)

ENROLLMENT(RollNo [FK], CourseID [FK])
```

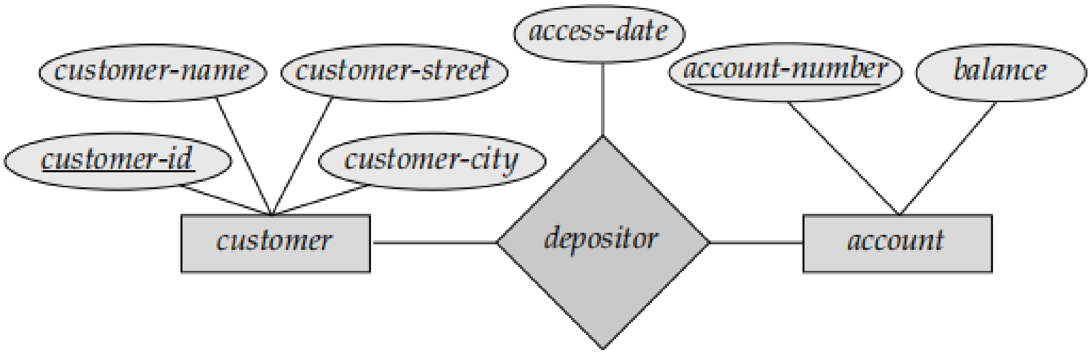

Figure 2.10 E-R diagram with an attribute attached to a relationship set.

Step 4: Map Multi-Valued Attributes

- Multi-valued attributes require a separate table.

Example:
If a Student has multiple phone numbers:

```
STUDENT(RollNo [PK], Name, Age)
```

```
PHONE(RollNo [FK], PhoneNo)
```

Step 5: Map Derived Attributes

- Derived attributes (like Age, which is derived from DateOfBirth) are usually **not stored directly** but calculated when required.

5. Example ER to Relational Model

Banking Enterprise

the **complete E-R diagram** for a banking enterprise represents a **conceptual model of a bank** using standard E-R concepts. This diagram includes **entity sets, attributes,**

relationship sets, roles, and cardinalities determined through the database design process.

Entities and Attributes

1. **Customer**
    - Attributes: customer-id (PK), customer-name, customer-street, customer-city
2. **Account**
    - Attributes: account-number (PK), balance
    - Subtypes: checking-account, savings-account
3. **Loan**
    - Attributes: loan-number (PK), amount, interest-rate, overdraft-amount
4. **Payment**
    - Attributes: payment-number (PK), payment-date, payment-amount, type
5. **Employee**
    - Attributes: employee-id (PK), employee-name, telephone-number, start-date, employment-length
6. **Branch**
    - Attributes: branch-name (PK), branch-city, assets
7. **Dependent**
    - Attribute: dependent-name

Relationships

1. **Borrower** → connects **Customer** and **Loan**

   - Represents customers who take loans
   - Attributes: loan-branch
   - Many-to-many relationship

2. **Depositor** → connects **Customer** and **Account**

   - Attributes: access-date
   - One-to-many relationship (one customer can have multiple accounts)

3. **Works-for** → connects **Employee** and **Branch**

   - Role indicators: manager, worker

4. **Loan-payment** → connects **Loan** and **Payment**

   - Shows payments made for loans

5. **Cust-banker** → connects **Customer** and **Employee**

   - Represents the banker responsible for a customer

Special Features

- **ISA hierarchy**: differentiates subtypes of accounts (checking-account, savings-account)
- **Total participation**: e.g., every loan must have at least one borrower
- **Composite and derived attributes**: included where needed
- **Cardinality constraints**: indicate one-to-many or many-to-many relationships

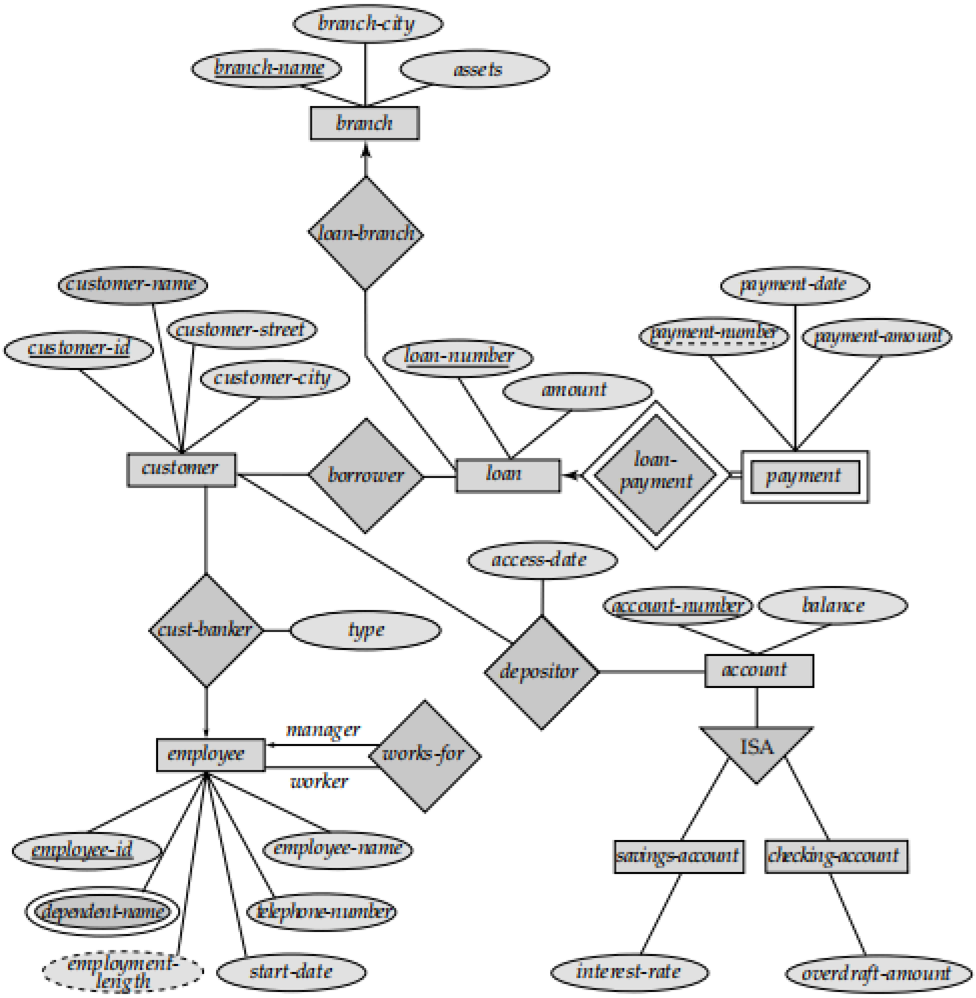

Figure 2.22 E-R diagram for a banking enterprise.

6. Rules and Guidelines

1. Every entity becomes a table.
2. Relationships are implemented using primary and foreign keys.
3. Multi-valued attributes → separate tables.
4. Weak entities → include foreign keys of strong entities.
5. Derived attributes are avoided in storage.

7. Importance of Translating ER to Relational Model

- Provides **logical structure** to data.
- Ensures **data integrity and consistency**.

- Enables implementation in real DBMS software.
- Helps in **query optimization** and **fast data retrieval**.
- Reduces redundancy and improves **normalization**.

8. Advantages of Relational Model

- **Simplicity**: Easy to design and understand.
- **Flexibility**: New tables and relations can be added easily.
- **Normalization**: Reduces duplication of data.
- **Data Security**: Access control and constraints can be applied.
- **Standardization**: Supported by most DBMS (MySQL, Oracle, SQL Server).

9. Applications in Real Life

- **Banking System**: Customer, Account, Transaction ER model → relational tables.
- **E-Commerce**: User, Product, Order ER model → implemented in MySQL.
- **Hospital System**: Patient, Doctor, Appointment ER model → relational DBMS.
- **University Database**: Students, Courses, Faculty relations → relational model.

10. Challenges in Translation

- Complex relationships (M:N with attributes).
- Handling recursive relationships (e.g., Employee supervises Employee).
- Large ER diagrams with hundreds of entities.
- Choosing correct primary keys.

11. conclusion

In this assignment, we explored how to **translate ER diagrams into relational models**. The process includes mapping entities, weak entities, relationships, multi-valued attributes, and derived attributes. Relational models provide the **logical framework** that DBMS uses to store and manipulate data efficiently.

The translation is essential for building reliable, consistent, and scalable databases that power applications across multiple industries such as **banking, education, healthcare, and e-commerce**.