

The background features abstract, overlapping green geometric shapes, primarily triangles and polygons, in various shades of green, creating a modern and dynamic visual effect.

# **Data, Data Management and Data Modelling**

# Contents

- ▶ What is Data?
- ▶ How is Data managed?
- ▶ Database Management System
- ▶ Purpose of Database Management System
- ▶ Structured Data Management and Schema
- ▶ Logical and Physical Schema
- ▶ Important Terminologies of Database Schema
- ▶ Elements of Database Schema

# What is Data?

Any abstract value (“numeric”, “alphabetic” and/or, “alpha-numeric”) stored/recorded about a person, an object or an event can be called “Data”.

Once the “Data” is collected and stored formally, it can be utilized for various computational operations like data retrieving, analyzing and report generation.

Those computational operations generate meaningful “Information” from the stored data.

# How is Data managed?

A collection of data can be stored in a standardized format that can be accessed, shared and analyzed by multiple users.

- ▶ DBMS (Database Management System) software is one of the ways to record and/or, retrieve data by its “users” in formal manner in secured way
- ▶ Some examples of DBMS software: *MySQL*, Oracle, MS SQL Server, SQL Lite, Microsoft Access etc.

# DBMS (Database Management System)

## Database Management System consists of:

- ▶ Collection of interrelated Data
- ▶ Set of Programs to access the stored Data
- ▶ An environment that is convenient and efficient for Data/Information retrieval

## Some Application Sectors of DBMS

- Banking: all transactions, customers, payments and accounts
- Airlines: reservations, schedules
- Universities: registration, grades, courses, results
- Sales: customers, products, sale
- Online retailers: order tracking, customized recommendations
- Manufacturing: production, inventory, orders, supply chain
- Human resources: employee records, salaries, tax deductions

# Purpose of Data Management

- ▶ Reducing data redundancy and inconsistency
- ▶ Uniformity of 'methods of access' to data from a database
- ▶ Maintaining consistency of data recording formats
- ▶ Facilitates concurrent usage of the same system
- ▶ Provide Data Security
- ▶ Offers Data Integrity

# Structured Data Management: RDBMS

Database Schema is a structure that contains how the data is organized and how the relations are associated among them in a RDBMS (Relational Database Management System)

- ▶ Logical Schema -
  - ▶ Conveys the logical constraints and associations of the database
  - ▶ *Example : ERD (Entity Relationship Diagram)*
- ▶ Physical Schema -
  - ▶ Conveys the layouts of actual way of what data will be stored where exactly in the database
  - ▶ *Example : Physical Data Model*

# Structured Data Management: RDBMS

## Logical Schema:

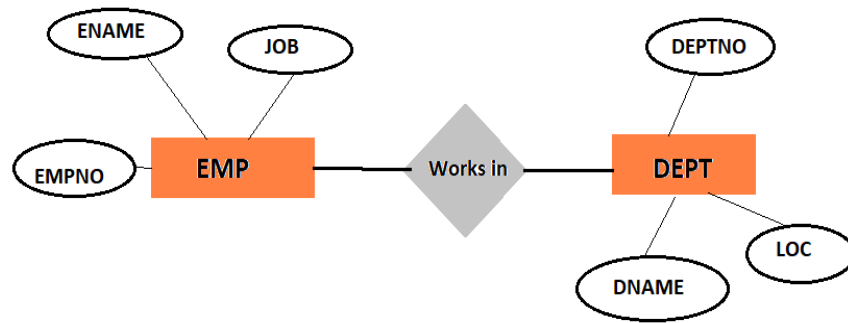
- Deciding on the logical database schema
- Business decision:** What attributes should we record in the database?
- Computer Science decision:** What schema should we have and how should the attributes be distributed in the relation schema?

## Physical Schema:

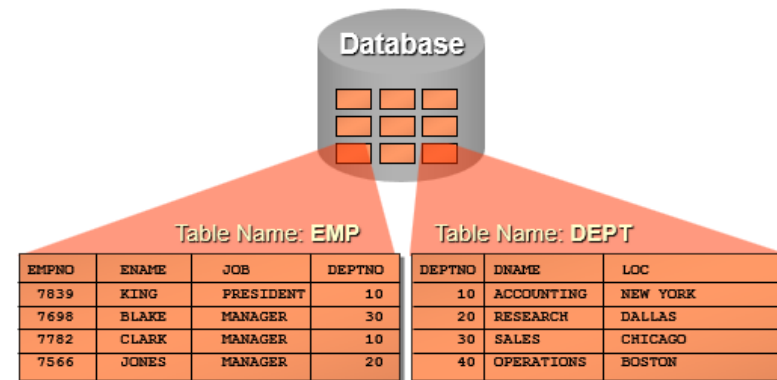
- Deciding the physical layout of the database



# Logical Schema VS. Physical Schema



Logical Database Schema (ERD)



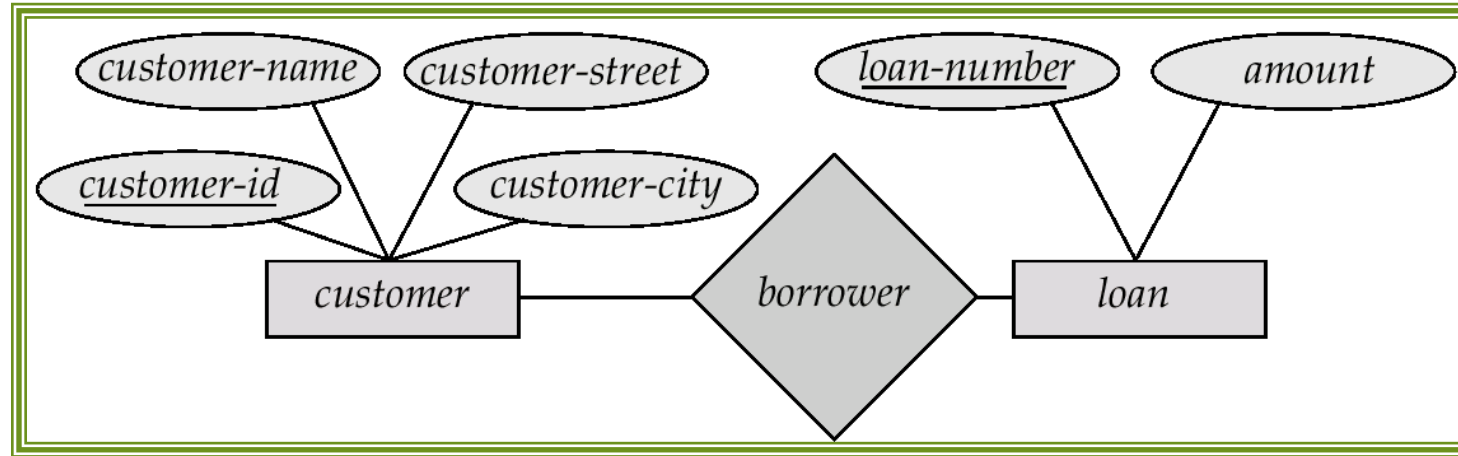
Physical Database Schema  
(Data Model)

# Logical Schema Design: ERD (Entity Relationship Diagram)

## Elements and Terminologies of ERD:

- ▶ Entity, Entity Set
- ▶ Attribute
- ▶ Entity Relationship, Entity Relationship Set
- ▶ Constraint

# Logical Schema: ERD (Entity Relationship Diagram)



- ▶ Rectangles represent entity sets
- ▶ Diamonds represent relationship sets
- ▶ Lines link attributes to entity sets and entity sets to relationship sets
- ▶ Ellipses represent attributes

# Entity and Entity Sets

A person, organization, object type, or concept about which information is stored.

For Example :

321-12-3123	Jones	Main	Harrison
019-28-3746	Smith	North	Rye
677-89-9011	Hayes	Main	Harrison
555-55-5555	Jackson	Dupont	Woodside
244-66-8800	Curry	North	Rye
963-96-3963	Williams	Nassau	Princeton
335-57-7991	Adams	Spring	Pittsfield

*customer*

L-17	1000
L-23	2000
L-15	1500
L-14	1500
L-19	500
L-11	900
L-16	1300

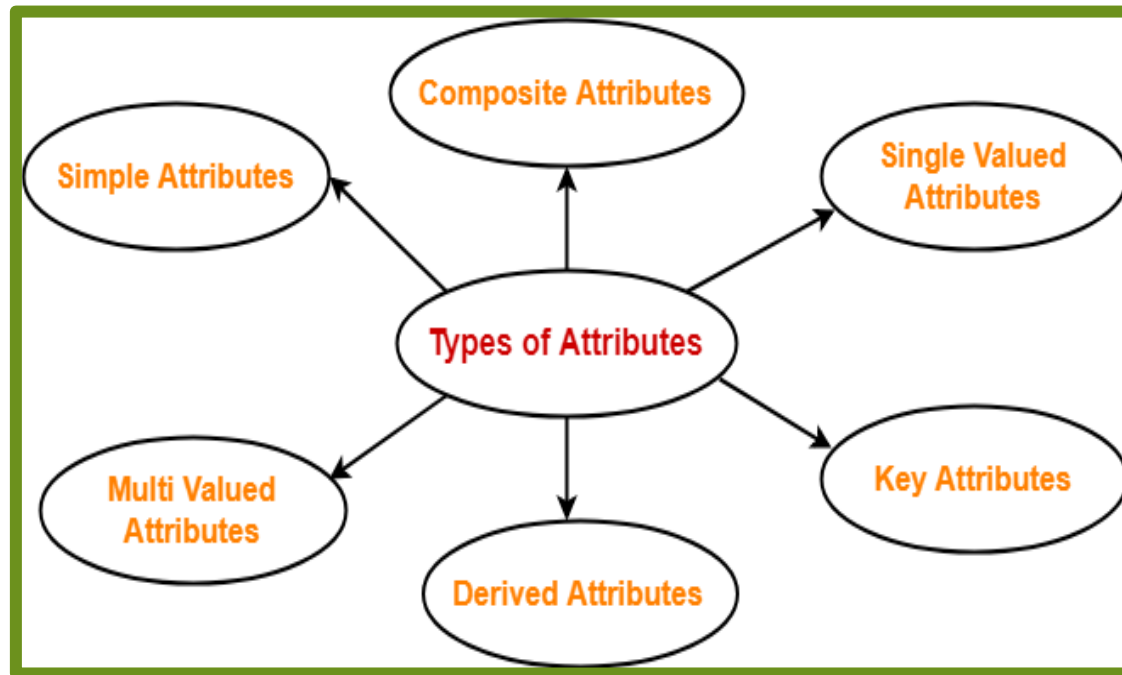
*loan*

# Attributes

An entity is represented by a set of attributes, which in turn are the descriptive properties possessed by all members of an entity set.

In other words, Attributes are the properties of Entity.

For Example, in “Customers” Entity, customer’s id, name street city, phone number and date of birth are the attributes that describes every customer.



# Types of Attributes

- ▶ **Keyed Attribute:** Key attribute is a distinguishable attribute which holds a characteristic of the entity uniquely.

For Example, Loan number is unique for each of the loans, and customer id is unique for each customer in the bank.

Generally, the “primary key” attribute is considered as the “keyed” attribute during the database design.

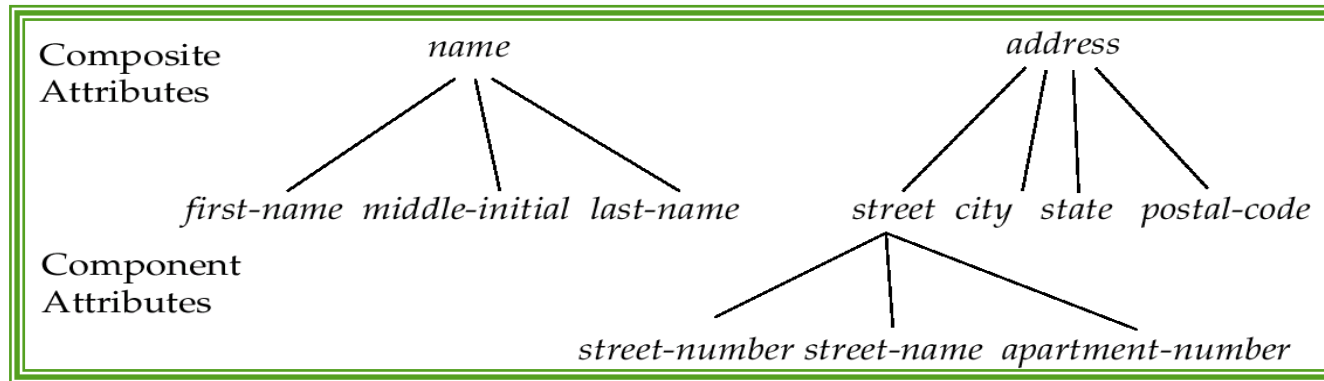
- ▶ **Simple Attribute:** Simple Attributes are atomic values which cannot be divided in any further components.

For Example: the Loan amount. The amount is an atomic value cannot be fragmented in any further meaningful manner.

# Types of Attributes

- **Composite Attribute** is composed of two or, many component attributes

For Example,



Here, in the example above, the “name” is composed of first, middle and last name. Similarly, the address attribute is composed of street, city, state and the post code.

# Types of Attributes

- ▶ **Multivalued Attribute:** Multivalued Attribute can have more than one value associated with the entity.




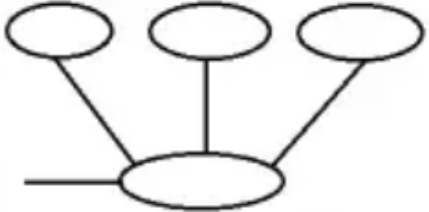

For Example, A customer can have multiple phone numbers. In that case, the “phone no” attribute will become a multivalued attribute

- ▶ **Derived Attribute:** If the value of an attribute can be derived from other attributes in the entity, then it's called derived attribute.

For Example, if there is any necessity to know customer's age then that can be derived from the “date of birth” of the customer, and this is how “age” will become a derived attribute.



# Notations of Attributes in ERD

	Attribute
	Keyed Attribute
	Multivalued Attribute
	Composed/ Composite Attribute
	Derived Attribute

# Entity Relationships

Entity Relationships in ERD are the connections among the “Entities” that illustrates how “Entities” such as, “People”, “Objects” and/or, concepts are related to each other within that ERD.

# Entity Relationships: Mapping Cardinalities

Mapping Cardinality is an expression of the number of entities to which another entity can be associated via a relationship set.

For a binary relationship set the mapping cardinality must be one of the following types:

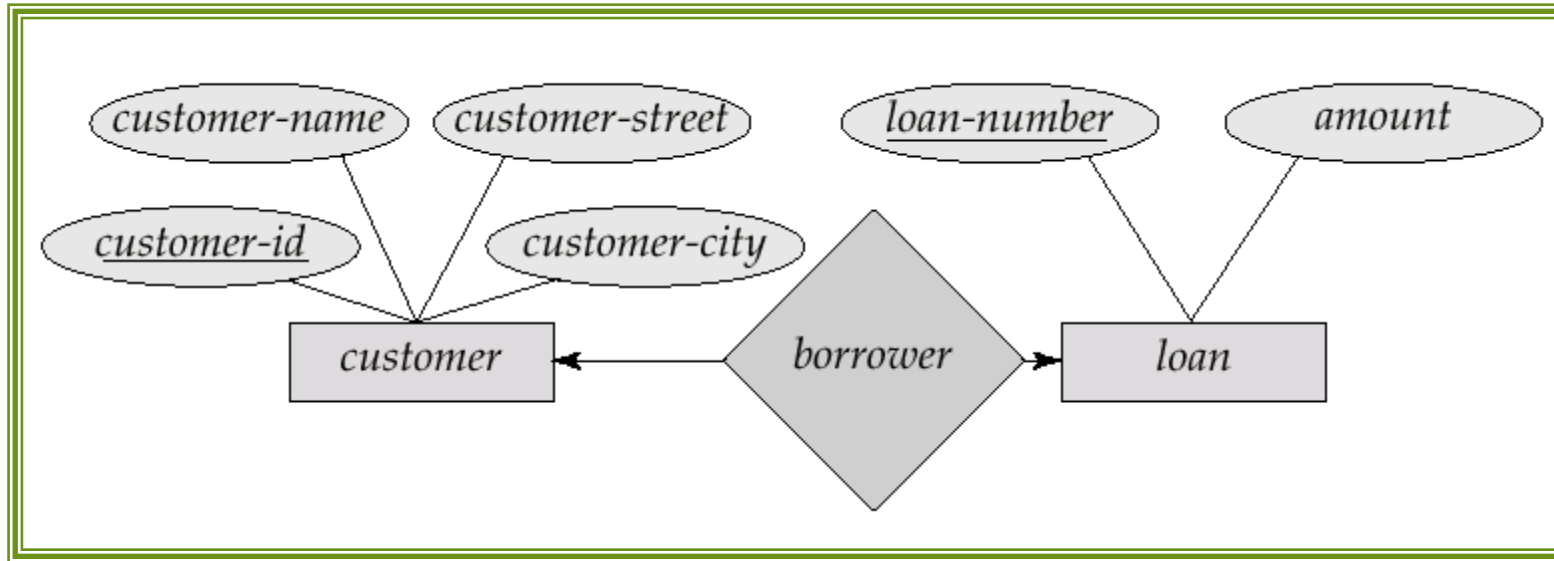
- ▶ One to one
- ▶ One to many
- ▶ Many to one
- ▶ Many to many

# Entity Relationships: Mapping Cardinalities

Cardinality constraints can be expressed by drawing either a directed line ( $\rightarrow$ ), signifying “one,” or an undirected line ( $-$ ), signifying “many,” between the relationship set and the entity set.

**Let's see the next Slides!!**

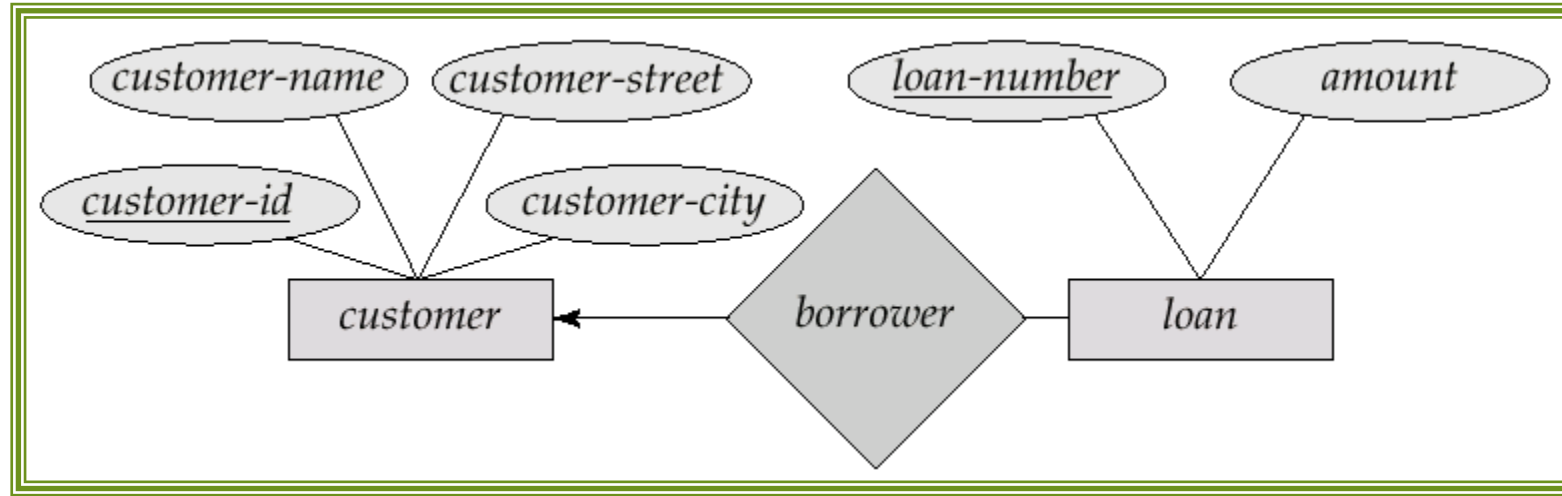
# Cardinality Constraints: One to One



A customer is associated with at most one loan via the relationship borrower

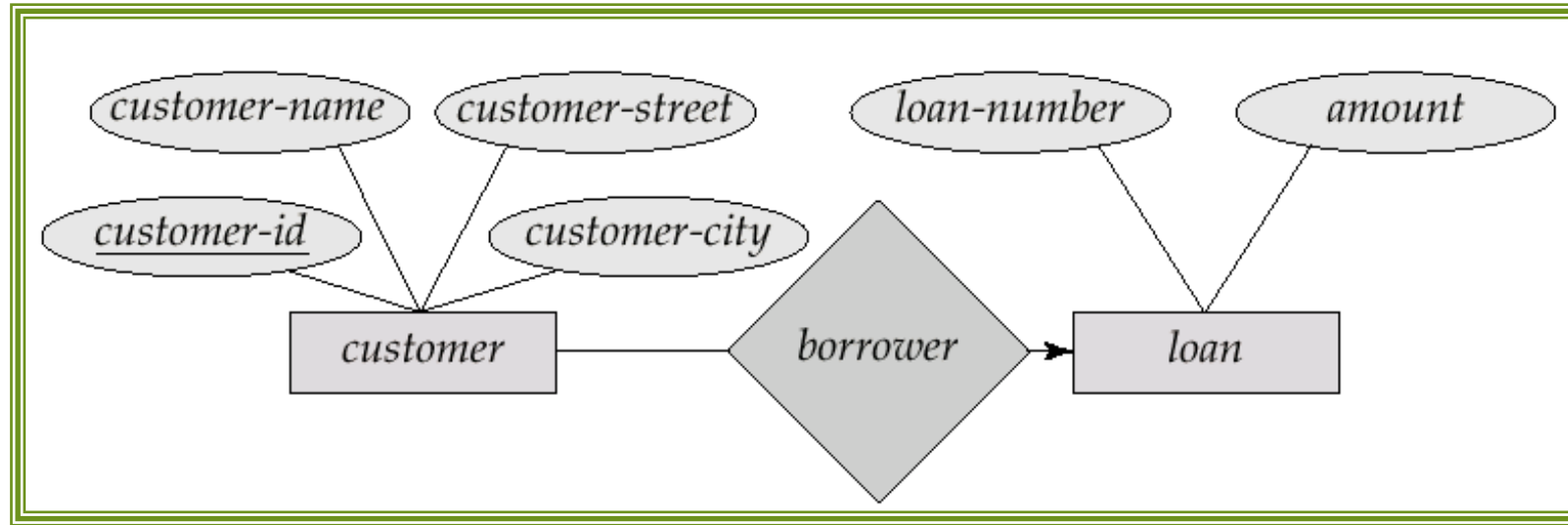
A loan is associated with at most one customer via borrower

# Cardinality Constraints: One to Many



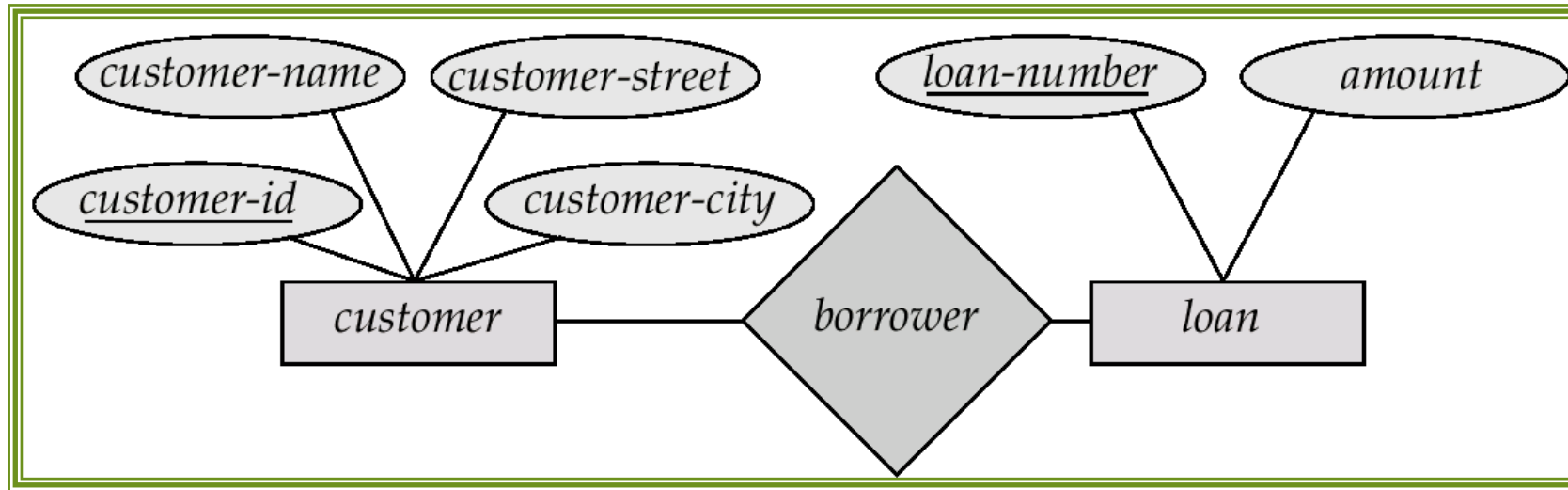
In the one-to-many relationship a loan is associated with at most one customer via borrower, a customer is associated with several (including 0) loans via borrower

# Cardinality Constraints: Many to One



In a many-to-one relationship a loan is associated with several (including 0) customers via *borrower*, a customer is associated with at most one loan via *borrower*

# Cardinality Constraints: Many to Many



**A customer is associated with several (possibly 0) loans via borrower**

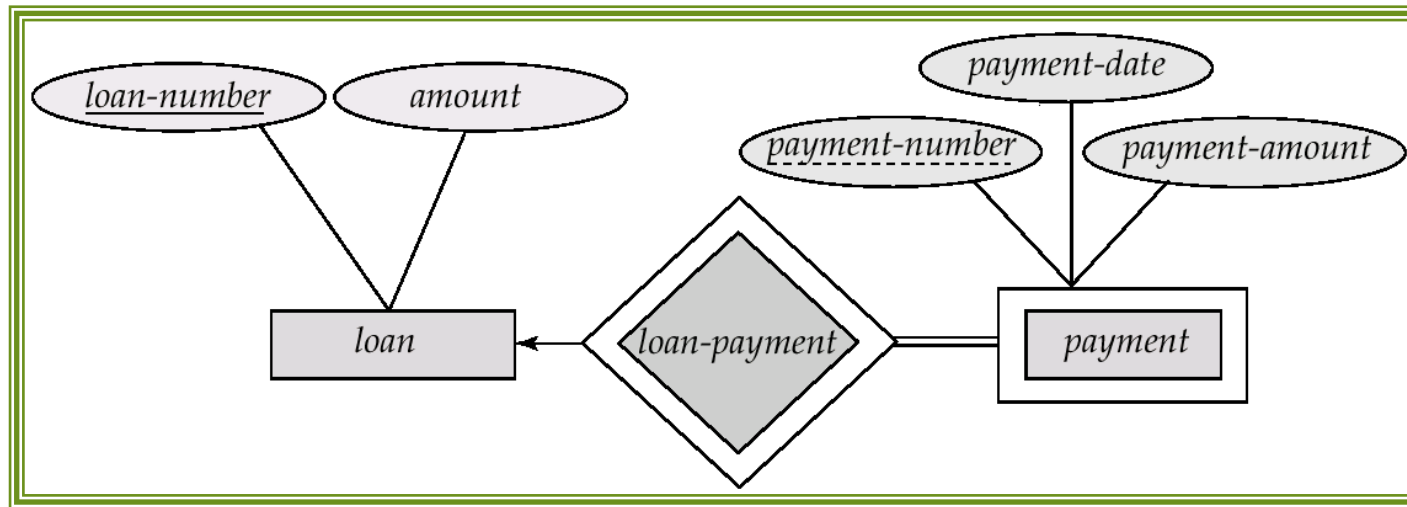
**A loan is associated with several (possibly 0) customers via borrower**



# Weak Entity

An entity set that does not have a primary key is referred to as a *weak entity set*.

- ▶ It must relate to the identifying entity set via a total; one-to-many relationship set from the identifying to the weak entity set
- ▶ Identifying relationship depicted using a double diamond
- ▶ Identifying Entity depicted using double rectangle



# Primary Key in Structured Databases

A Primary Key is a special key in Relational Database, that provides the facility to identify each row of the database table uniquely.

**\*\*\*There can be ONLY one Primary Key for an Entity**

For Example, Student's ID in the student table can be a primary key, as each value under "StudentID" column will find one specific student and that ID is designated for one student only in the entire database table.

*Primary Key is a special case of unique keys but there can be other "Unique Key" attributes in the Entity.* For example, email address of the students are also unique

# Foreign Key in Structured Databases

- ▶ A Foreign Key is a column or combination of columns that is used to establish and enforce a link between the data in two tables.
- ▶ A Foreign Key in one table must be the Primary Key of another table.
- ▶ This is done to control the data that can be stored in the table containing the Foreign key.

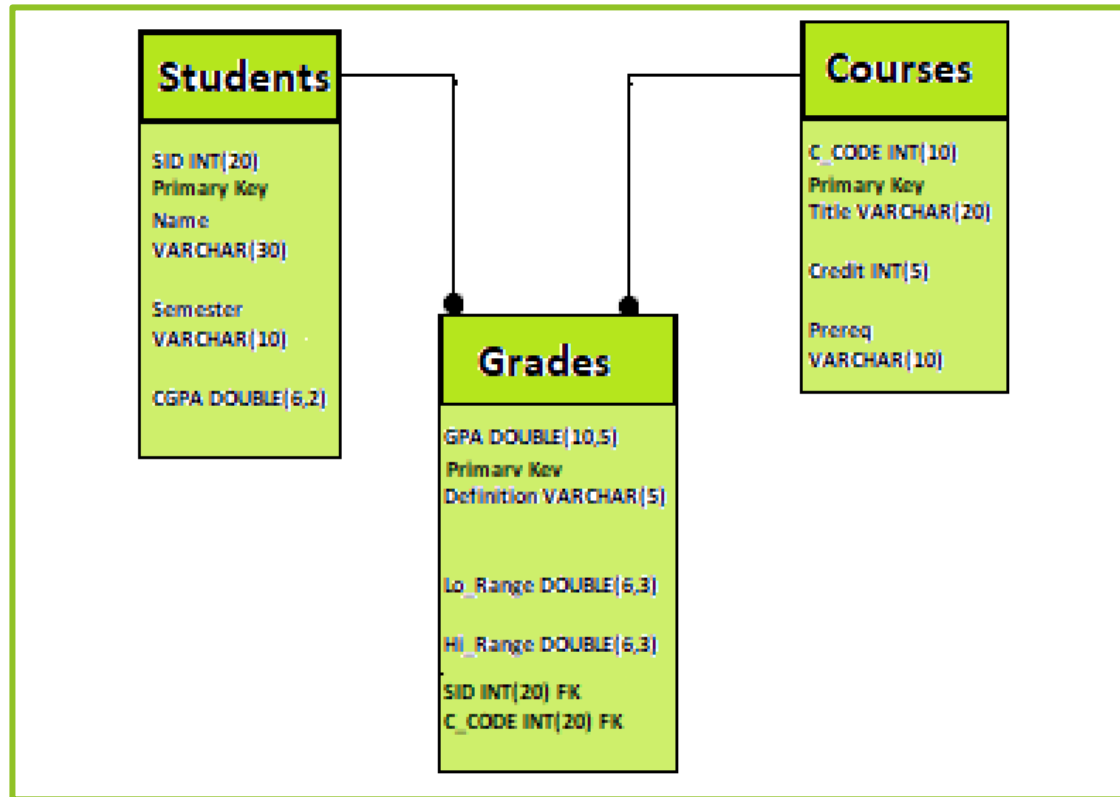
# Data Modelling

A Data Model is a “picture” of the whole architecture of a database which represents the concepts present in the database, data objects and the rules for storing the data in that particular database.

Data Model is a more detailed version of Logical Schema (ERD)

\*\*\* It's way easier to understand the organization of the tables and their relationships through diagram than keeping them listed in a random manner!

# Data Model



- ▶ Data Model is more detailed than ERD
- ▶ The “Entities” are referred as “Tables” and the “Attributes are referred as “Columns” of the tables
- ▶ The table names as well the attribute names are compatible names in this stage instead of conventional “user-friendly” names, e.g, there is no “spacing” allowed in between the names of the columns or, the tables while database implementation
- ▶ Data types are specified along with the attributes

# Purpose of using Data Modelling

- ▶ Tables in a Data Model will exactly represent the exact Database Tables that are used to store the data
- ▶ The Data Model is a detailed picture of the whole application or, business process for which the physical database is being built.
- ▶ The information in the Data Model can be used for defining the Relationship between Tables(Entities) using relevant Constraints
- ▶ Data Model helps all types of stakeholders to understand the businesses processes the database is being developed to support
- ▶ Data model helps to understand data mappings in ETL process

