DBMS & RDBMS

Welcome

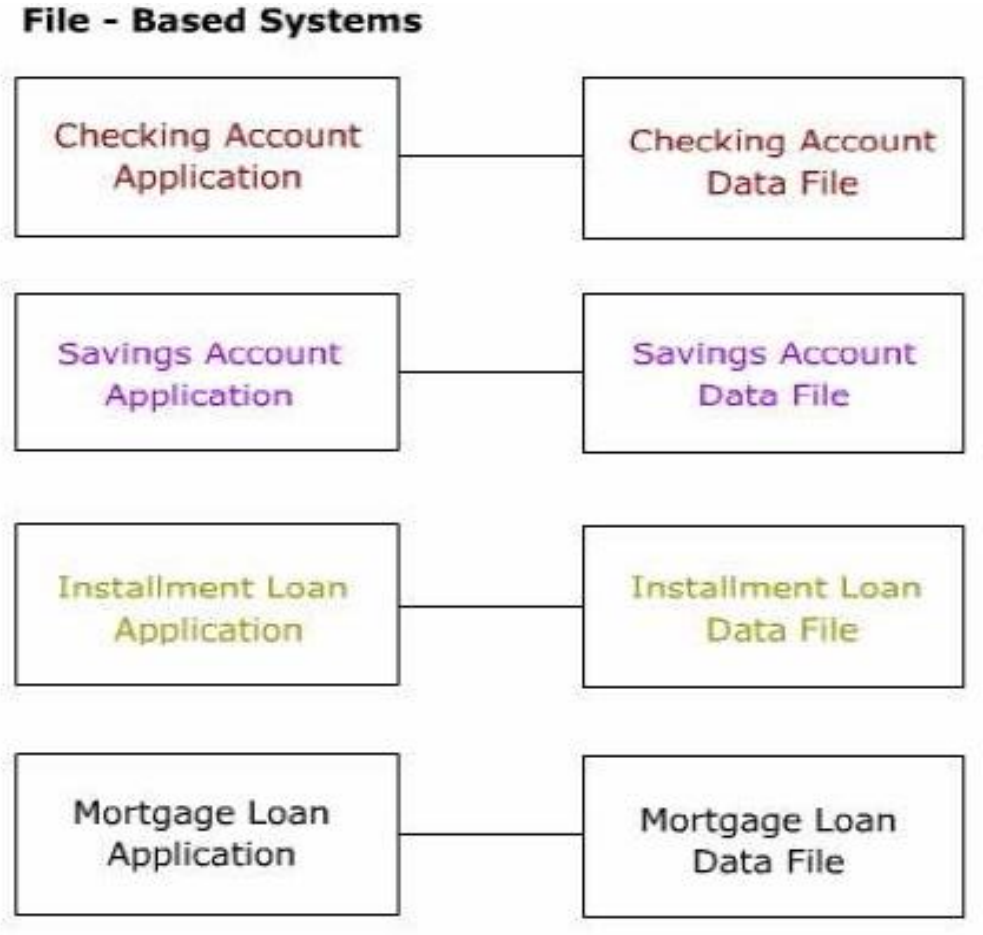
Introduction to DBMS & RDBMS

- Overview of Data Management System
- DBMS & RDBMS Overview
- CODD's Relational Rules apply on RDBMS
- Normalization Vs De-Normalization
- Entity-Relationship Model

Data Management System

- File-Based Systems
- Database Systems (DBMS)

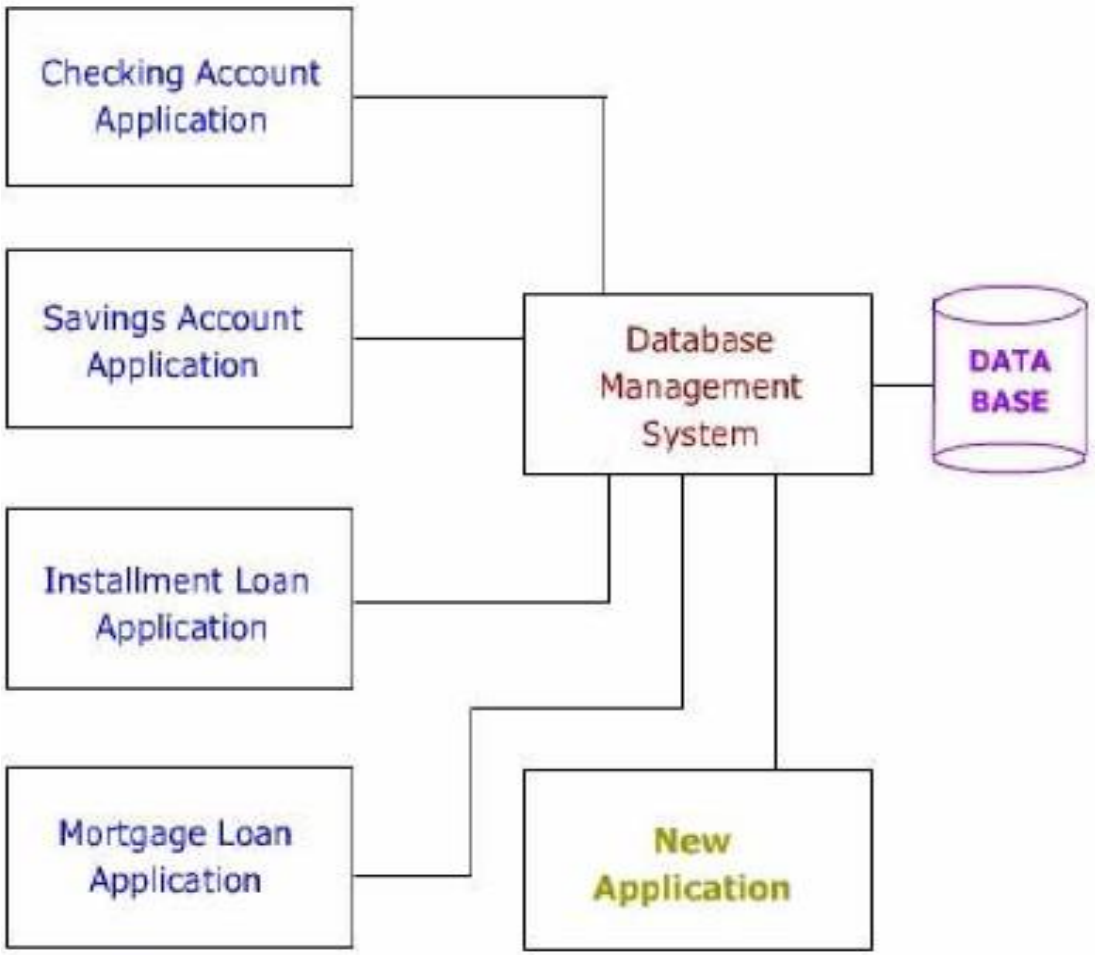
File Based Systems



Drawbacks of File Based System

- Data Redundancy and Inconsistency
- Unanticipated Queries
- Data Isolation
- Concurrent Access Anomalies
- Security Problems
- Integrity Problems

Database System



Advantages of Database System

- Minimal Data Redundancy
- Data Consistency
- Data Integration
- Data Sharing
- Enforcement of Standards
- Application Development Ease
- Better Controls
- Data Independence
- Reduced Maintenance

Functions of DBMS

- Data Definition
- Data Manipulation
    - Planned
    - Unplanned
- Data Security & Integrity
- Data Recovery & Concurrency
- Data Dictionary Maintenance
- Performance

Role of the Database Administrator

- Defining the Schema
- Liaising with Users
- Defining Security & Integrity Checks
- Defining Backup / Recovery Procedures
- Monitoring Performance

RDBMS

Overview

Objective of Module - RDBMS

At the end of this module, you will be able to:

- Explain RDBMS
- List Terminologies in RDBMS
- Explain Dr. Codd's rule
- Describe properties of Relations
- Explain Normalization and Normal Forms

RDBMS

- Relational Database is a collection of programs that enables user to create and maintain a database based on formal mathematical concepts and definitions.
- It has to maintain the data integrity and data consistency.
- In relational database, the data's are arranged in row and column wise and there is only one value for each column.

Relational Concepts - Terminology

Relation	A table or File
Tuple	Row contains an entry for each attribute
Attributes	Columns or the characteristics that define the entity
Domain	A range of values (or Pool)
Entity	Some object about which we wish to store information
Null	Represents an unknown/empty value
Atomic value	Smallest unit of data; the individual data value
Candidate key	Some attribute (or a set of attributes) that may uniquely identify each row (tuple) in the relation (table)
Primary key	The candidate key is chosen for primary attributes to uniquely identify each row
Alternate key	The remaining candidate keys that were not chosen as primary key
Foreign key	An attribute of one relation that might be a primary key of another relation

Codd Rules for RDBMS

- Codd proposed thirteen rules (numbered zero to twelve) and said that if a Database Management System meets these rules, it can be called as a Relational Database Management System. These rules are called as Codd's12 rules. Hardly any commercial product follows all.
- 12 Codd Rules:
- **Rule zero**

  This rule states that for a system to qualify as an RDBMS, it must be able to manage database entirely through the relational capabilities.
- **Rule 1 : Information rule**

  All information(including metadeta) is to be represented as stored data in cells of tables. The rows and columns have to be strictly unordered.

Codd Rules for RDBMS

- **Rule 2 : Guaranteed Access**

  Each unique piece of data(atomic value) should be accesible by : Table Name + primary key(Row) + Attribute(column).

  NOTE : Ability to directly access via POINTER is a violation of this rule.

- **Rule 3 : Systematic treatment of NULL**

  Null has several meanings, it can mean missing data, not applicable or no value. It should be handled consistently. Primary key must not be null. Expression on NULL must give null.

- **Rule 4 : Active Online Catalog**

  Database dictionary(catalog) must have description of Database. Catalog to be governed by same rule as rest of the database. The same query language to be used on catalog as on application database.

Codd Rules for RDBMS

- **Rule 5 : Powerful language**

  One well defined language must be there to provide all manners of access to data. Example: SQL. If a file supporting table can be accessed by any manner except SQL interface, then its a violation to this rule.

- **Rule 6 : View Updating rule**

  All view that are theoretically updatable should be updatable by the system.

- **Rule 7 : Relational Level Operation**

  There must be Insert, Delete, Update operations at each level of relations. Set operation like Union, Intersection and minus should also be supported.

- **Rule 8 : Physical Data Independence**

  The physical storage of data should not matter to the system. If say, some file supporting table were renamed or moved from one disk to another, it should not effect the application.

Codd Rules for RDBMS

- **Rule 9 : Logical Data Independence**

  If there is change in the logical structure(table structures) of the database the user view of data should not change. Say, if a table is split into two tables, a new view should give result as the join of the two tables. This rule is most difficult to satisfy.

- **Rule 10 : Integrity Independence**

  The database should be able to conforce its own integrity rather than using other programs. Key and Check constraints, trigger etc should be stored in Data Dictionary. This also make RDBMS independent of front-end.

- **Rule 11 : Distribution Independence**

  A database should work properly regardless of its distribution across a network. This lays foundation of distributed database.

- **Rule 12 : Non-subversion rule**

  If low level access is allowed to a system it should not be able to subvert or bypass integrity rule to change data. This can be achieved by some sort of looking or encryption

Relational Database Terms

- **Entity**: an object about which you want to store data
- **Attributes**: Columns or the characteristics that define the entity
- **Relationships**: links that show how different records are related
- **Key Fields**: establish relationships among records in different tables

ER diagram

- It is a collection of objects.
- An entity is an object that is distinguishable from other objects by a set of attributes.
- This is the basic object of E-R Model, which is a 'thing' in the real world with an independent existence.
- An entity may be an 'object' with a physical existence.
- Entities can be represented by 'Ellipses'.
- An entity type is represented in ER diagrams as a rectangular box.
- Example:
  - Customer, account etc.

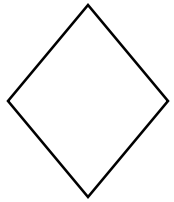
Introduction to (E-R) Modeling

- Entity-Relationship (E-R) Diagram
  - A detailed, logical representation of the entities, associations and data elements for an organization or business
- Notation uses three main constructs
  - Data entities
  - Relationships
  - Attributes

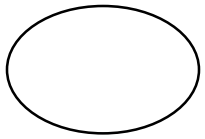
Entity-Relationship (E-R) Modeling

- Entity and Entity set
  - An entity is anything that has characteristic features. A person, place, object, event or concept in the user environment about which the organization wishes to maintain data
  - An entity set is a set of entities of the same type that share the same properties or attributes
  - An Entity set is represented by a rectangle in E-R diagrams

- Relationship and Relationship set
  - A relationship is an association among entities.
  - A relationship set is a set of relationships of the same type
  - A relationship set is represented by diamonds in E-R diagrams.



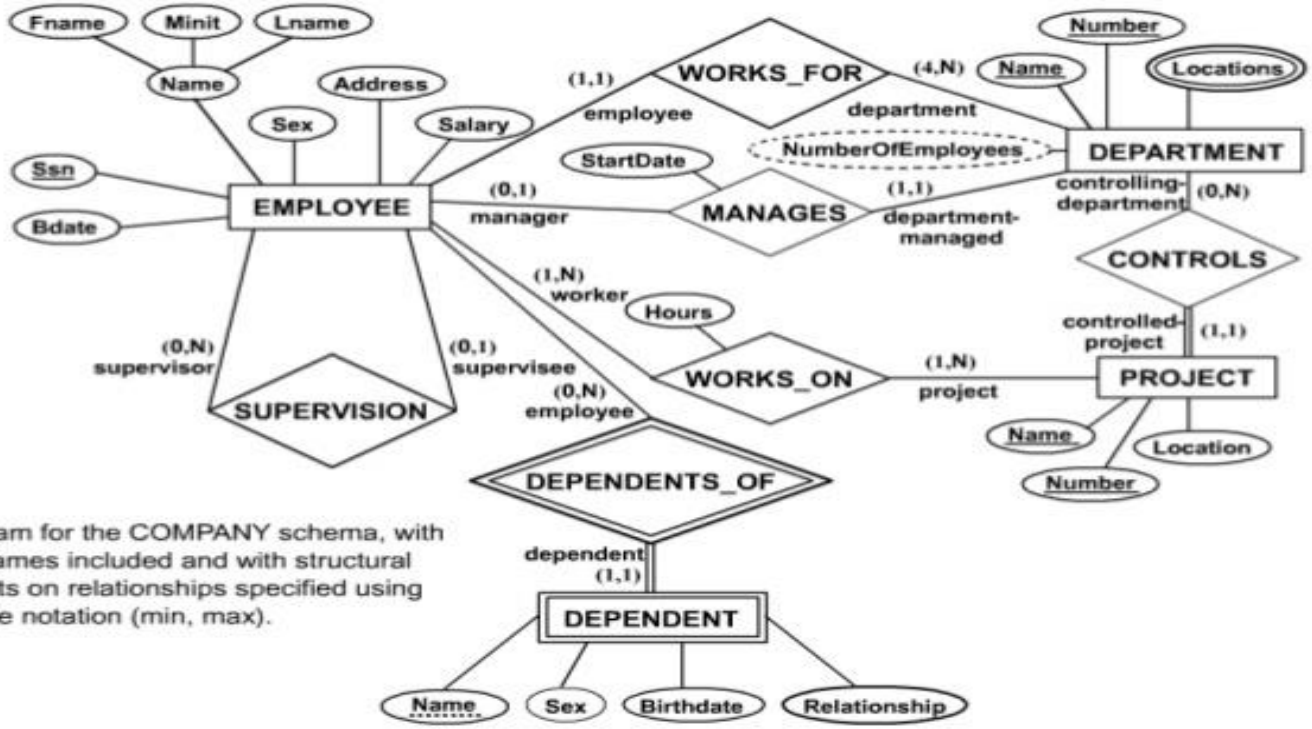
- Attribute
  - A named property or characteristic of an entity that is of interest to an organization.
  - Represented by ovals in E-R diagrams.



ER-Model presentation

COMPANY ER Schema Diagram using (min, max) notation

Alternative ER Notations



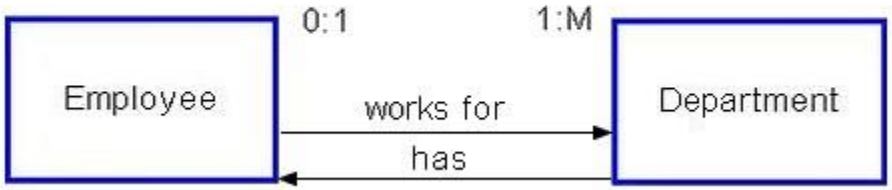
ER diagram for the COMPANY schema, with all role names included and with structural constraints on relationships specified using alternative notation (min, max).

Tools for Designing E–R Diagrams

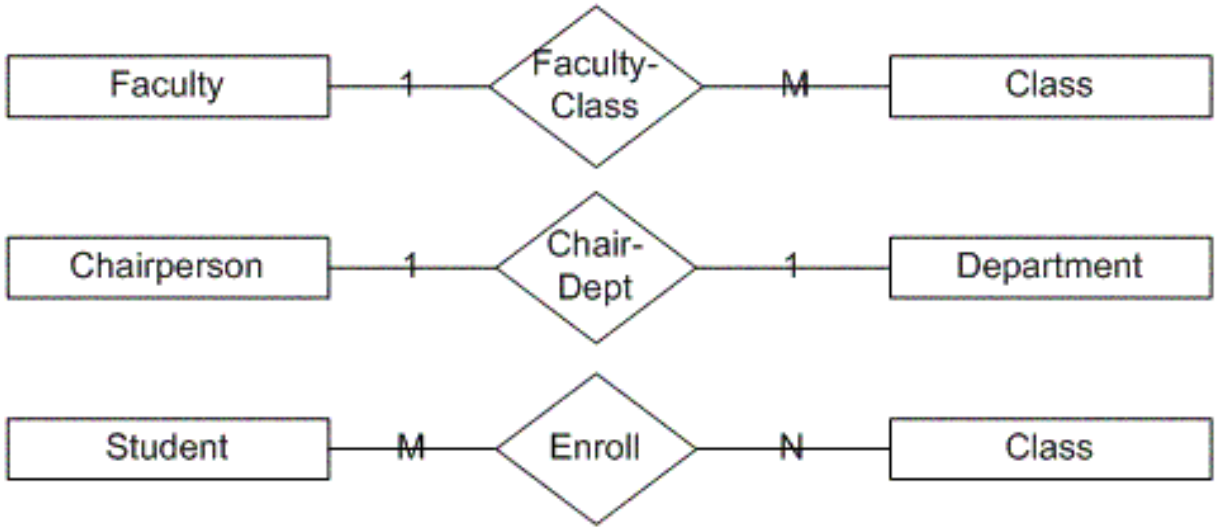
- ERWin
- Oracle Designer
- Embarcadero ER/Studio
- Sybase PowerDesigner
- Datanamic DeZign

Applying Cardinality Rules (Relationships)

EXAMPLE OF CARDINALITY



Example 2



Method 1: One: 1
Many M,N

Applying Normalization Rules

- By Definition:
- **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).

Normalization rules are divided into following normal form:

First Normal Form
Second Normal Form
Third Normal Form
BCNF

Check-Out this, for Details:

Microsoft Office
erPoint 97-2003 P

ERD Development Process

- Identify the entities
- Determine the attributes for each entity
- Select the primary key for each entity
- Establish the relationships between the entities
- Draw an entity model
- Test the relationships and the keys

Identification of the entities

Types of entities:

- Regular Entities:
  - any physical object, event, or abstract concept that we can record facts about.
- Weak Entities:
  - any entity that depends on another entity for its existence.

Determining the Attributes

- Every Entity has attributes.
- The characteristics of the entities are represented by attributes.
- e.g., entity Employee has the attributes:
  - Emp_id
  - Emp_name
  - Salary
  - Date_of_birth

Key Attributes

- Certain attributes identify particular facts within an entity, these are known as KEY attributes.

The different types of KEY attribute are:

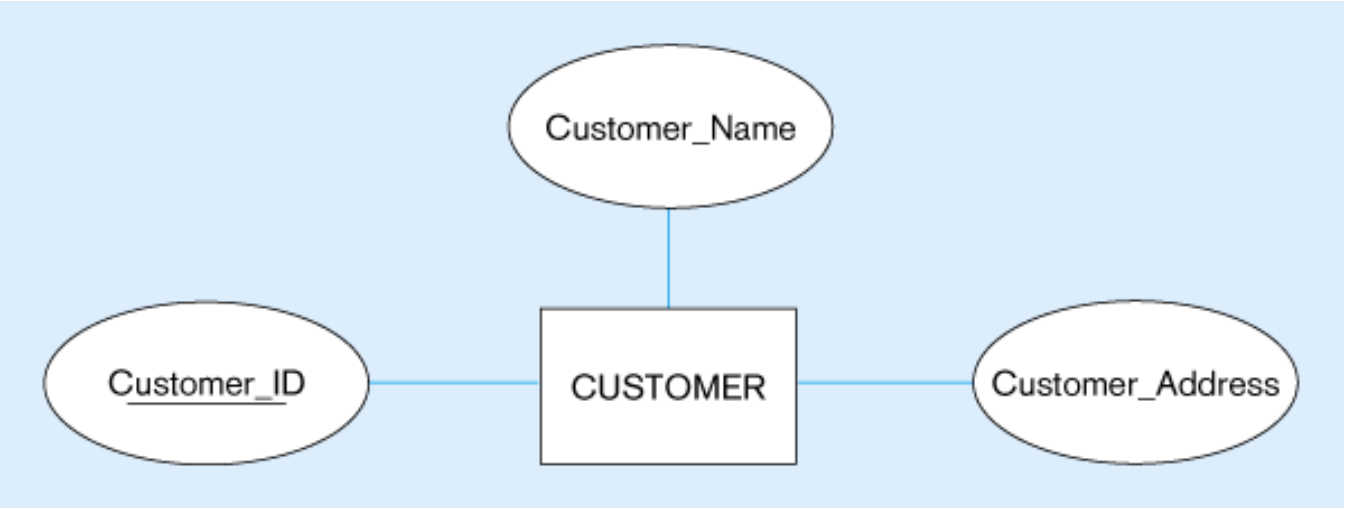
- Primary Key:
  - An attribute whose value can uniquely identify a complete record (one row of data) within an entity.
- Composite Primary Key
  - A primary key that consists of two or more attribute within an entity.
- Foreign Key
  - A copy of a primary key that exists in another entity (may exist in the same entity too) for the purpose of forming a relationship between the entities involved.

ER Diagram Components

Rectangles->*they represent entities set*
Ellipses->*they represent attributes.*
Diamonds->*they represent relationship set*
Lines ->*they link attributes to entity and entity to relationship.*
Double ellipses ->*they represent multi-valued attributes.*
Dashed ellipses->*they represent derived attributes.*
Double Lines ->*they represent total participation of an entity in a relationship*.
Double Rectangles->*they represent weak entities.*

ATTRIBUTES

CUSTOMER entity with simple attributes



CUSTOMER relation

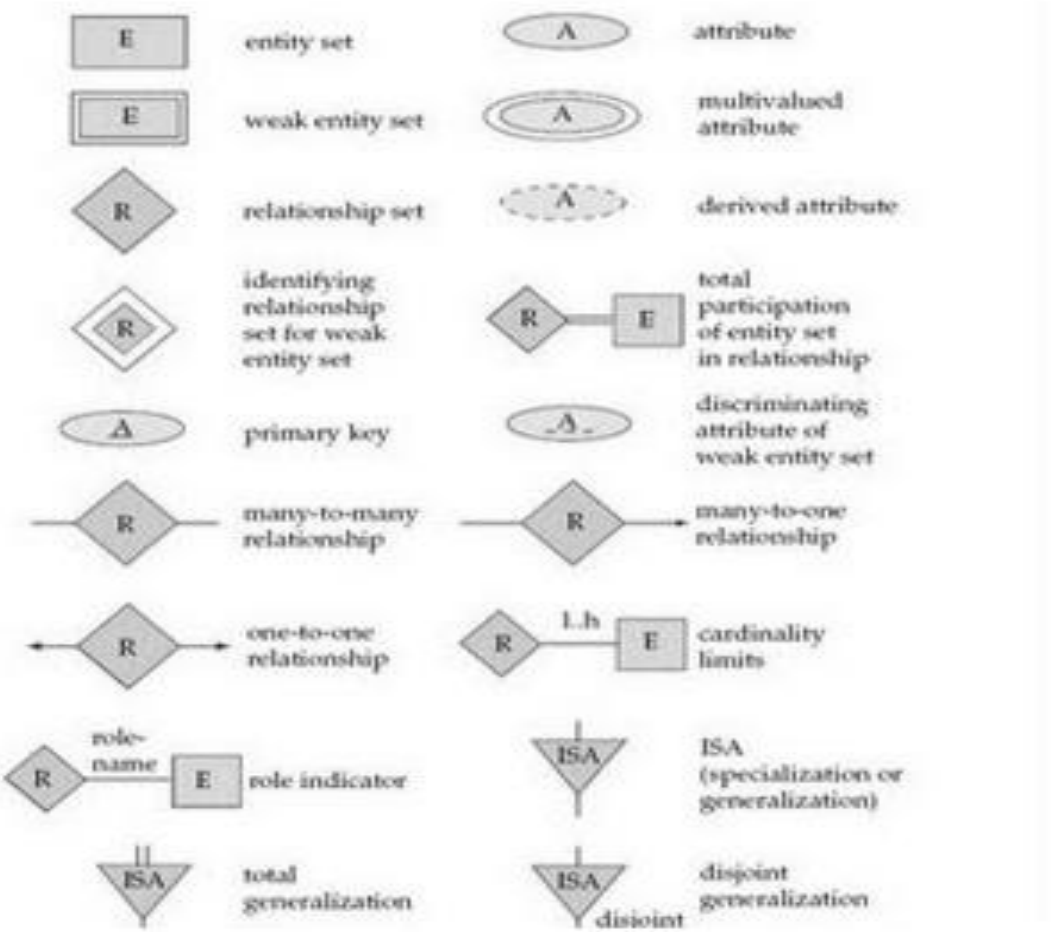
CUSTOMER

Customer_ID	Customer_Name	Customer_Address

Conversion Rules for ER Diagrams

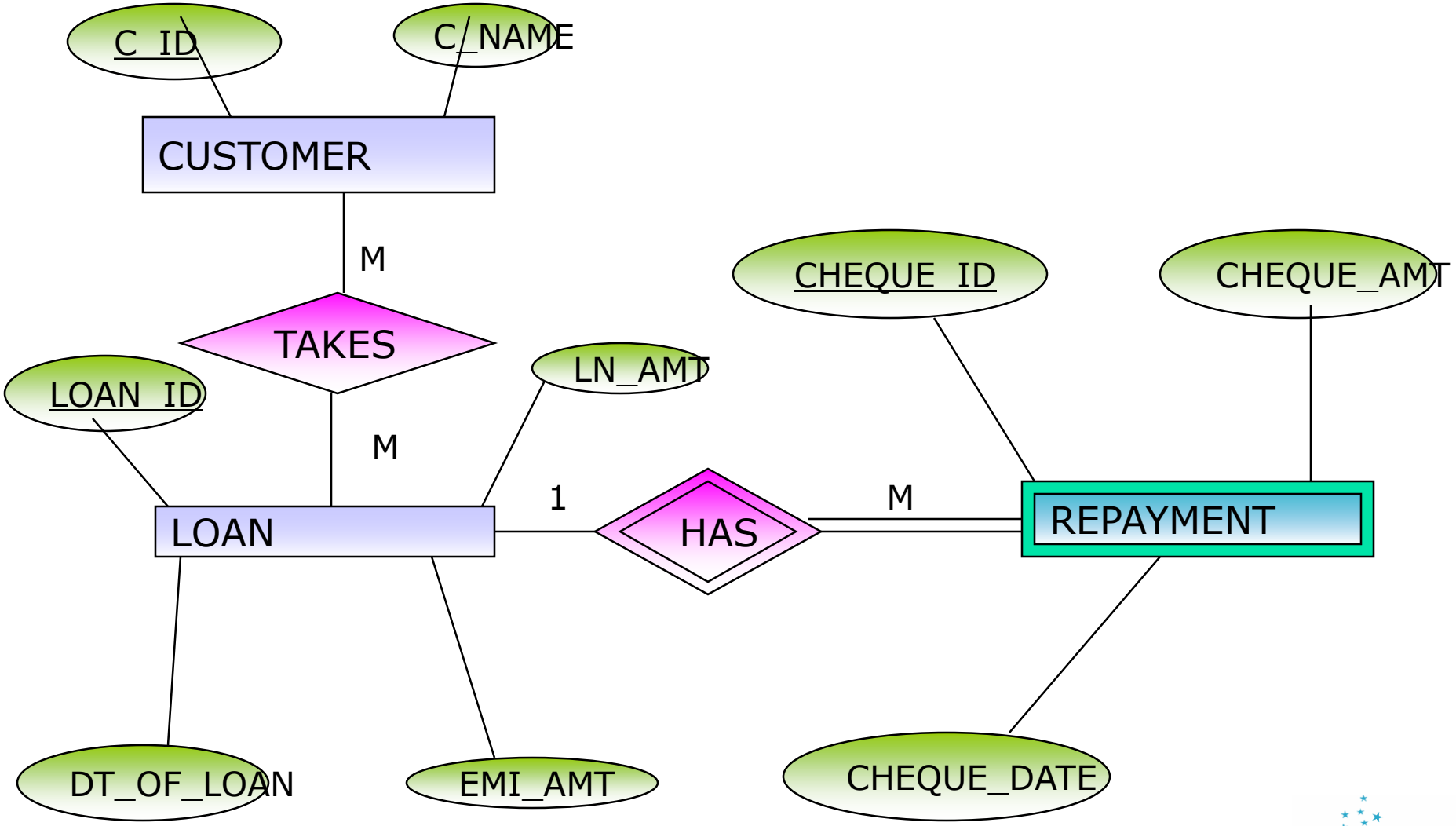
- The entities are mapped to relations.
- Their attributes become the columns in the relation.
- There is a particular attribute which becomes the primary key of the relation.
- In case of ***one-many relationship*** the primary key of the entity on the one side goes as a foreign key in the entity on the many side
- In case of a ***many to many relationship*** a separate relation has to be created with attributes from concerned entities participating in the many to many relationship. So we will have a composite key comprising of the primary keys of the participating relations.
- In case of ***a weak entity*** we have a partial key, which on itself cannot participate in a relationship. Hence a composite key is required where the primary key from the strong entity is clubbed with this partial key to form this composite key.

Symbols in ER Diagram



Entity Reletionship Diagram (ER Diagram) Symbols

Sample E-R Model



Data Modeling Practice

- Case Study1: CellDial, a Mobile Phone Manuf. Company

- Understand the information presented and Create a **Data** model and later on building the required **Dimensional** model as well, based on a given set of business Requirements:

- Define and document the complete process of building the Data-Model from a given set of sources and populating the final Model by Identifying...:
- **Entities** & **Attributes**
- Establishing **Relationship** btw Entities
- Applying **Normalization** Rules in building Model
- Follow the Data Model Development Life Cycle process, etc

Data Modeling Practice

- Case Study2: **LIBRARY MANAGEMENT SYSTEM**

- This will be a Scenario based Case Study to be built by the participants, and later this same Schema will be used to work-out on SQL practice examples

Case Study - Library System

- Define and document the complete process of building the Data-Model from a given source file and populating the final Model by Identifying...:
- **Entities** & **Attributes**
- Establishing **Relationship** btw Entities
- Applying **Normalization** Rules in building Model
- Follow the Data Model Development Life Cycle process, etc