Microsoft® Official Course

RDBMS

Microsoft®

Topics (Module 1)

- Database Fundamentals and RDBMS Concepts
  - Database & DBMS
  - Relational Databases and associated concepts
  - Normalization and different Normal forms

(Module 2)

- Introduction to SQL

  To explain the basic DDL ,DML commands, the concept of sub queries (independent and correlated), views, set operations (union, intersect, minus)

(Module 3)

- Advanced SQL queries
  - Multi-Table Database Architecture
  - SQL Joins and Multi-table Operations
  - Subqueries

Database Fundamentals and RDBMS Concepts

What is RDBMS?

RELATIONAL DATABASE MANAGEMENT SYSTEM

Introduction

- All software is divided into two general categories: **data** and **programs**.
- Programs are collections of instructions for manipulating data.
- Data can exist in a variety of forms (number , text ,symbols, images etc..)

EmployeeName	Age	Image	Total Salary
Shiv	40		$25.000

Database management System

What is Database ?

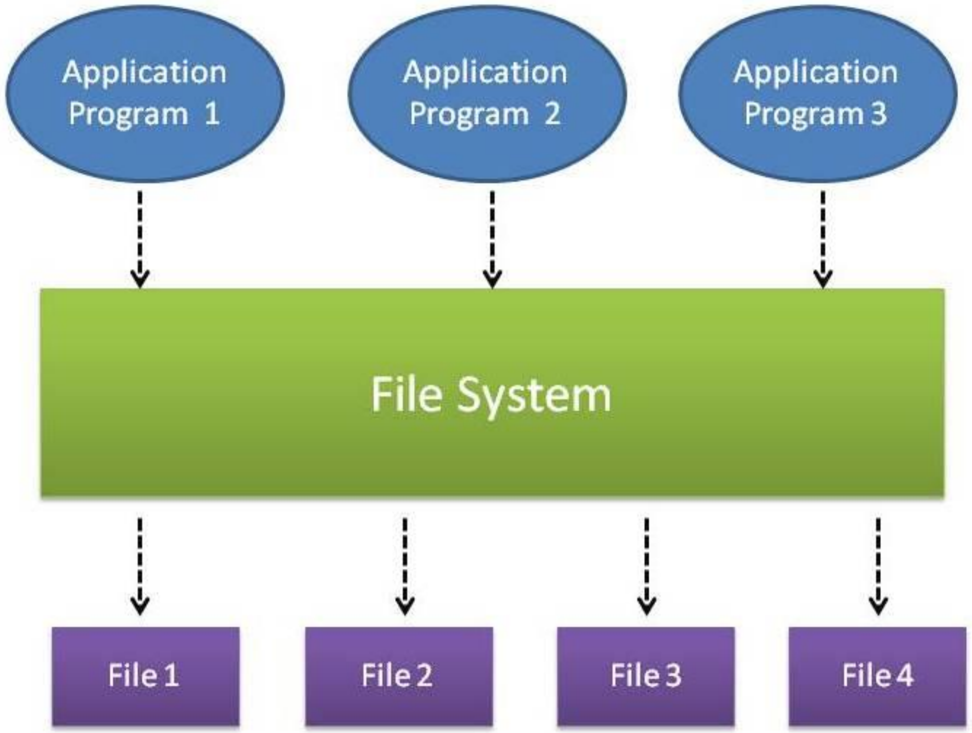
A Database is a collection of related data

What is DBMS?

Application where the following operations can be performed on data:

- Storing
- Modifying
- Retrieving

Traditionally the data was stored in form of **File System.**

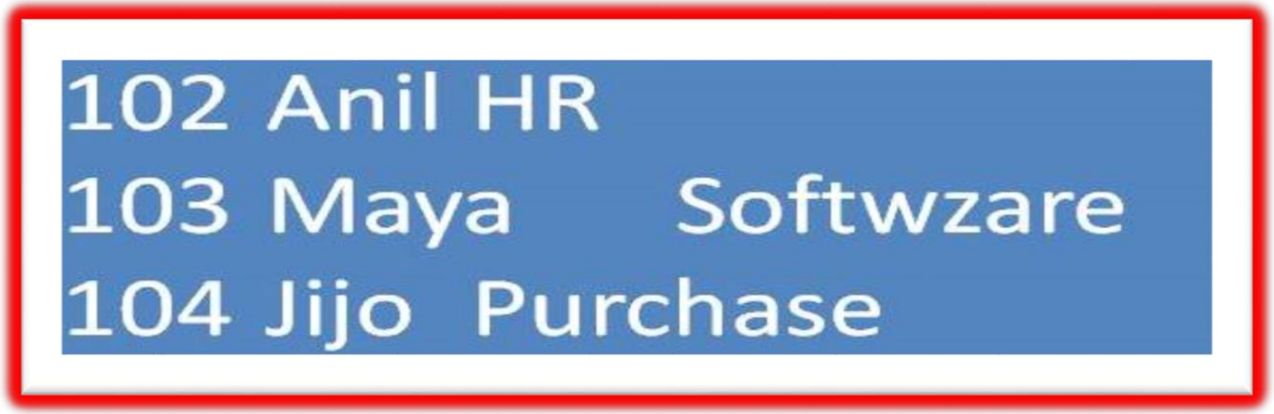


- In traditional approach, information is stored in flat files which are maintained by the file system under the operating system’s control.
- Application programs go through the file system in order to access these flat files

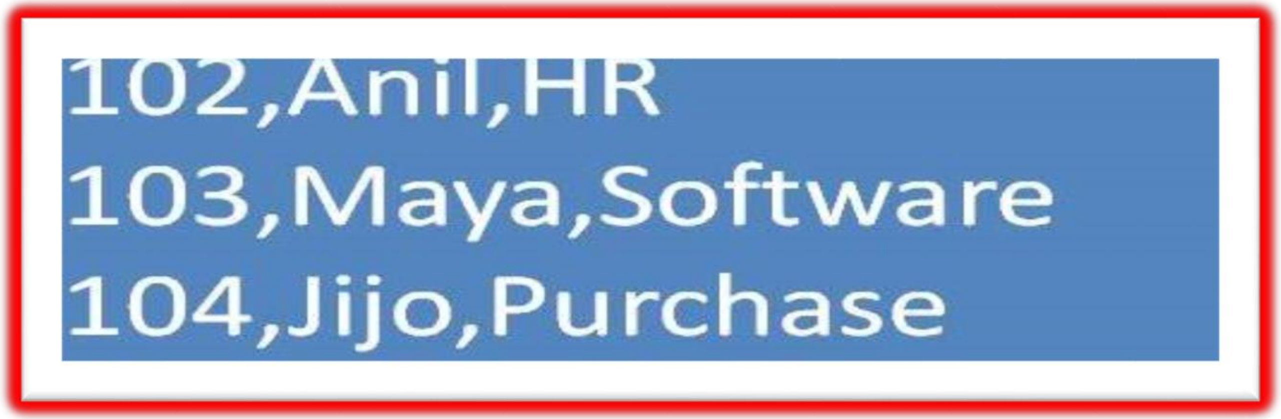
How data is stored in flat files

- Data is stored in flat files as records.
- Records consist of various fields which are delimited by a space, comma, pipe, any special character etc.
- End of records and end of files will be marked using any predetermined character set or special characters in order to identify them

Storing employee data in flat files



```
102 Anil HR
103 Maya       Softwzare
104 Jijo  Purchase
```

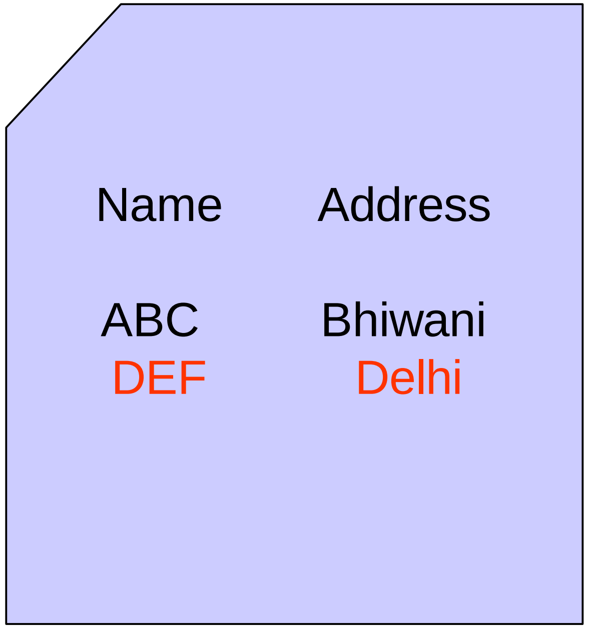


```
102,Anil,HR
103,Maya,Software
104,Jijo,Purchase
```

Problems with traditional approach for storing data

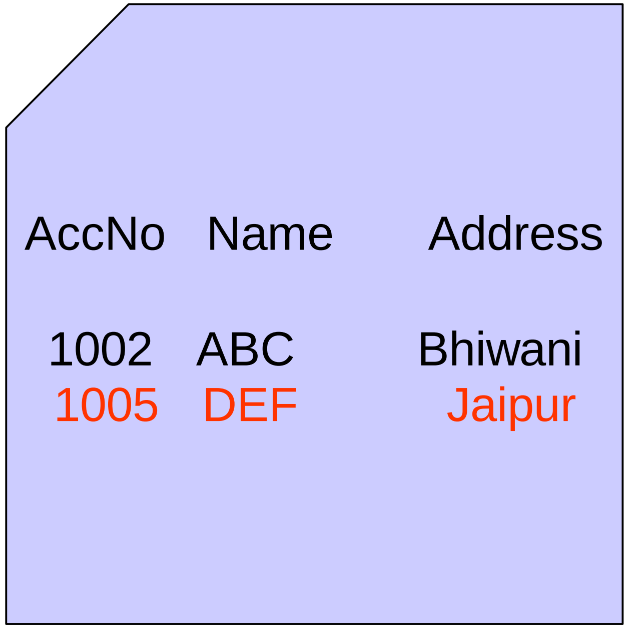
- Data Redundancy
- Difficulty Accessing data
- Data Isolation
- Atomicity problem
- Program/Data Dependence
- Concurrent Access Anomalies
- Data Security

Data Redundancy and Inconsistency



Name	Address
ABC	Bhiwani
DEF	Delhi

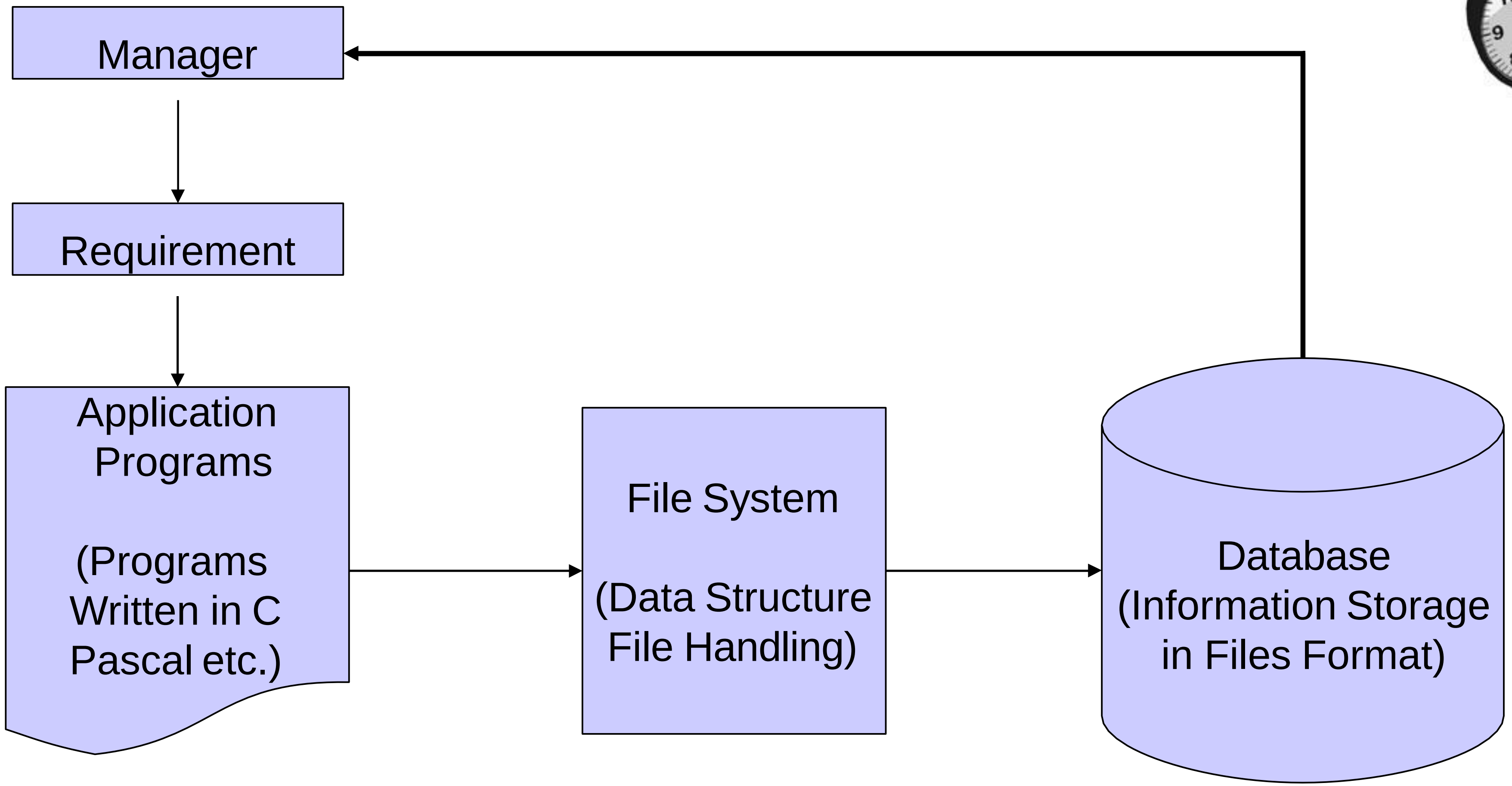
Customer Information



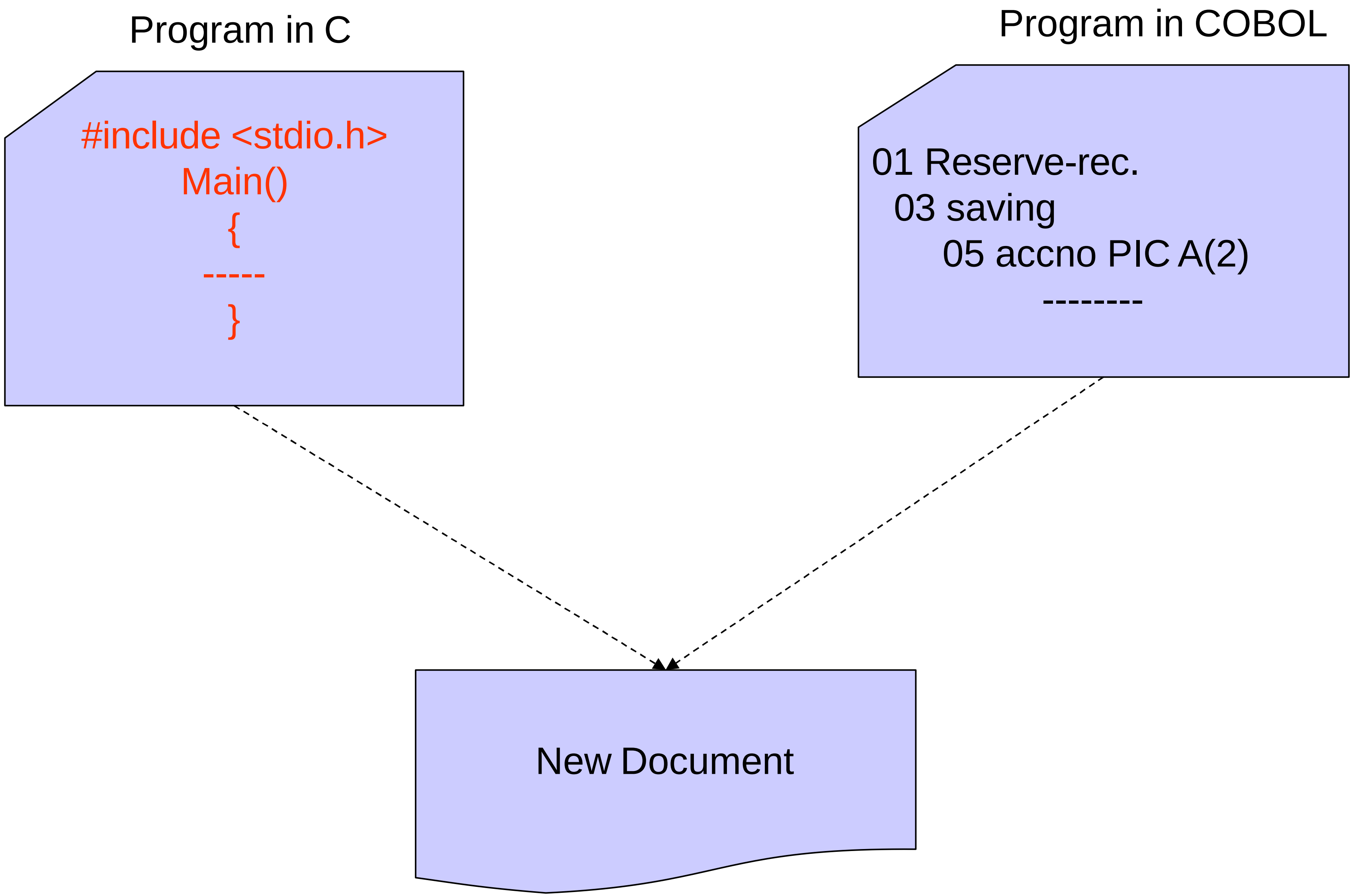
AccNo	Name	Address
1002	ABC	Bhiwani
1005	DEF	Jaipur

Saving Account

Difficulty in accessing data


Manager
Requirement
Application Programs
(Programs Written in C Pascal etc.)
File System
(Data Structure File Handling)
Database
(Information Storage in Files Format)

Data Isolation and Integrity Problems



Program in C

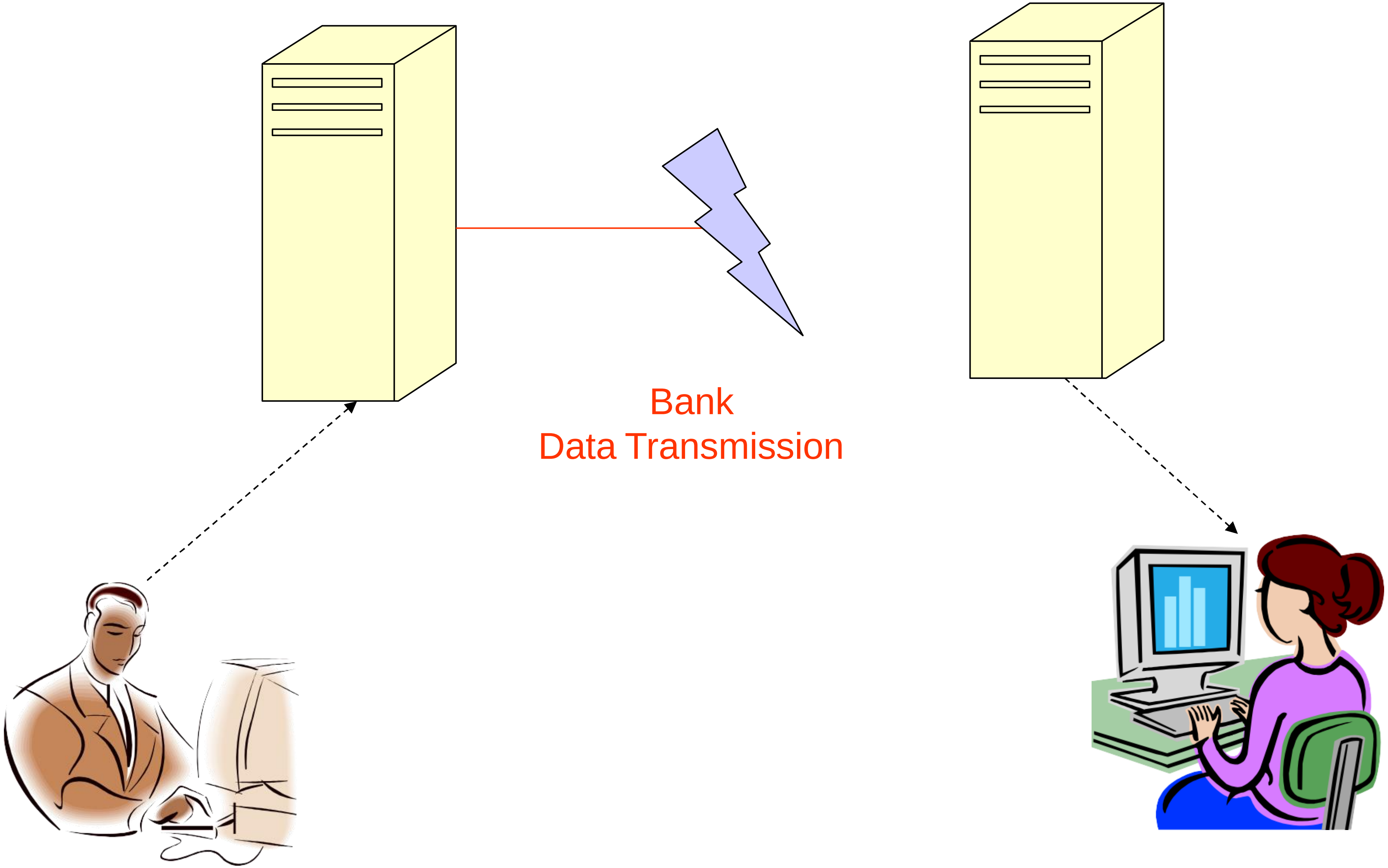
```
#include <stdio.h>
Main()
{
-----
}
```

Program in COBOL

```
01 Reserve-rec.
  03 saving
      05 accno PIC A(2)
            --------
```

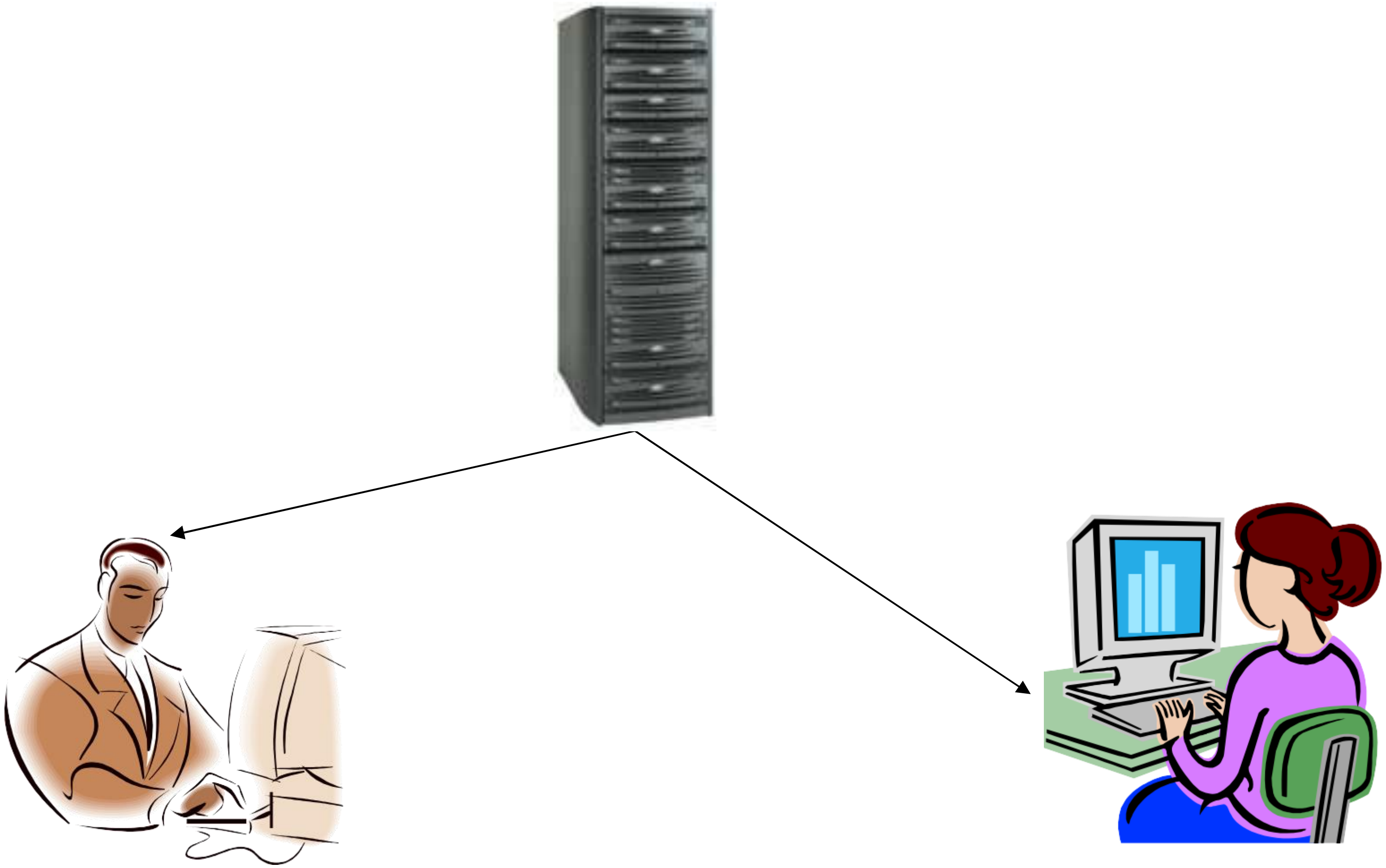
New Document

Atomicity Problems

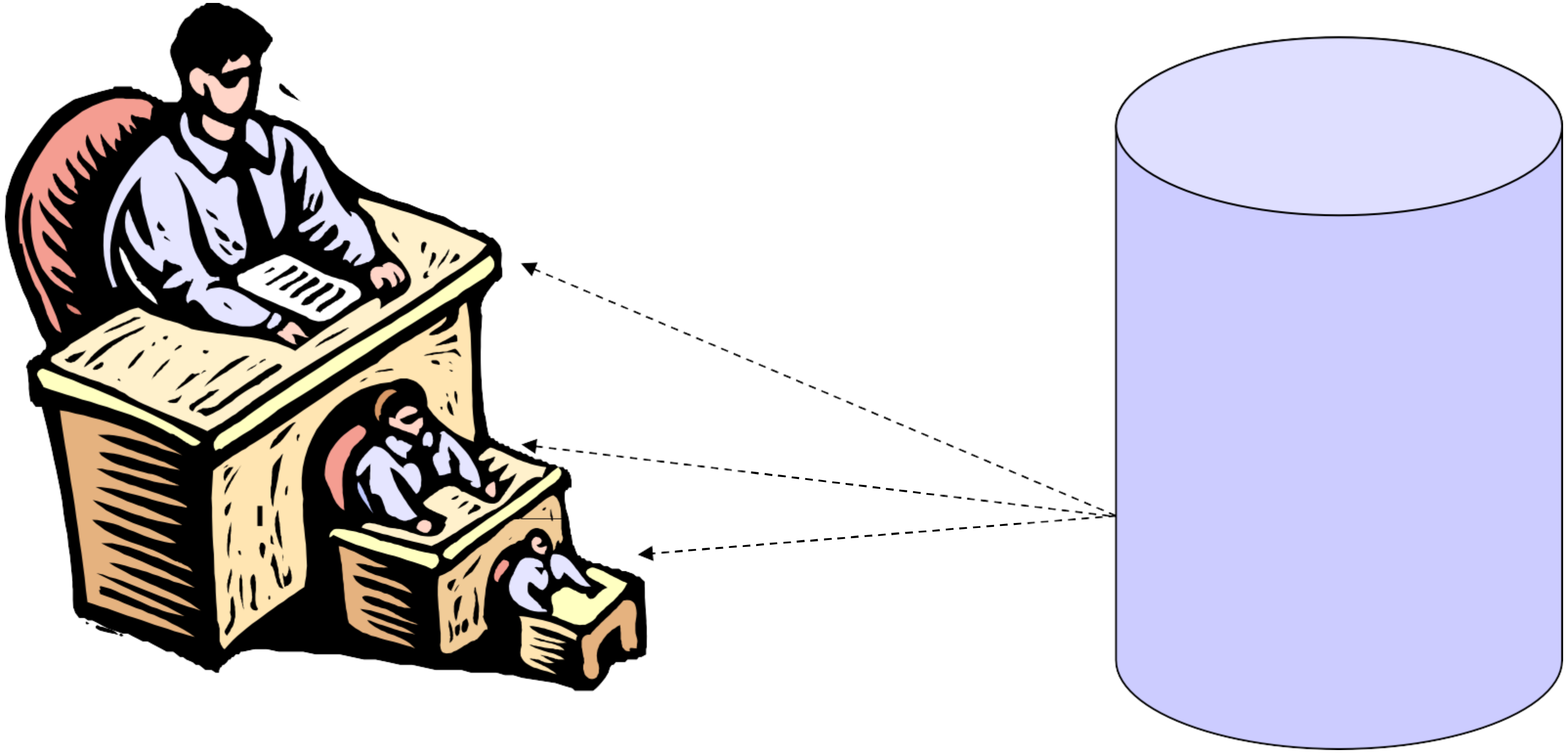

Bank

Data Transmission

Concurrent-access anomalies



Security Problems



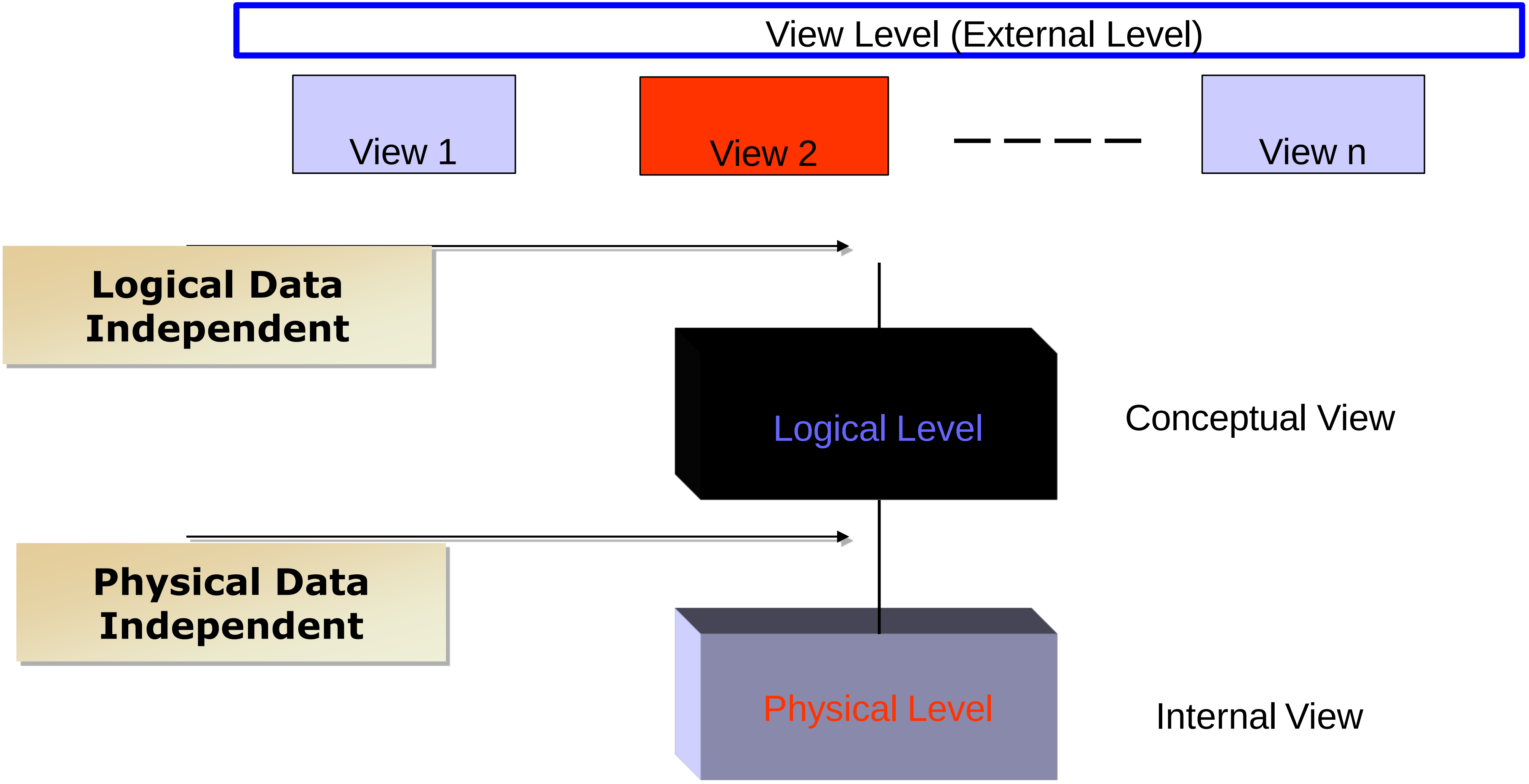
Difference between DBMS and RDBMS

DBMS	RDBMS
DBMS applications store data as file.	RDBMS applications store data in a tabular form.
DBMS does not apply any security with regards to data manipulation	RDBMS **defines the integrity constraint** for the purpose of ACID (Atomicity, Consistency, Isolation and Durability) property.
DBMS **does not support distributed database**.	RDBMS supports distributed database.
Normalization is not present in DBMS.	Normalization is present in RDBMS
Data redundancy is common in DBMS.	Keys and indexes are used to avoid redundancy.

Difference between DBMS and RDBMS

DBMS	RDBMS
DBMS is meant to be for small organization and deal with small data. it supports single user	RDBMS is designed to handle large amount of data. it supports multiple users
In dbms relationship between two tables or files are maintained programmatically	In Rdbms relationship between two tables or files can be specified at the time of table creation.
Transaction management is not possible ,insecure.	Efficient Transaction management and secure.
Examples of DBMS are file systems, xml etc.	Example of RDBMS are mysql, postgre, sql server, oracle etc.

View of DATA


View Level (External Level)
View 1
View 2
View n
Logical Data Independent
Logical Level
Conceptual View
Physical Data Independent
Physical Level
Internal View

Tier Architecture of DBMS (Database Architecture)

1. External Schema   (View Level)
2. Conceptual schema (logical Level)
3. Internal schema (Storage Level)

External Schema

- Provides a separate view for different groups of users, where each view represents only a part of database
- It allows user to access data in customized manner based on their requirement
- It allows use a security mechanism by hiding part of data from a group of users

Conceptual schema

- Provides a logical structure of entire database
- However it does not define how data is stored in database
- It defines what type of data can be stored by specifying the type and size
- What type of data cannot be stored , by specifying constraints
- And what is the relationship among the data by specifying referential integrity

Internal schema

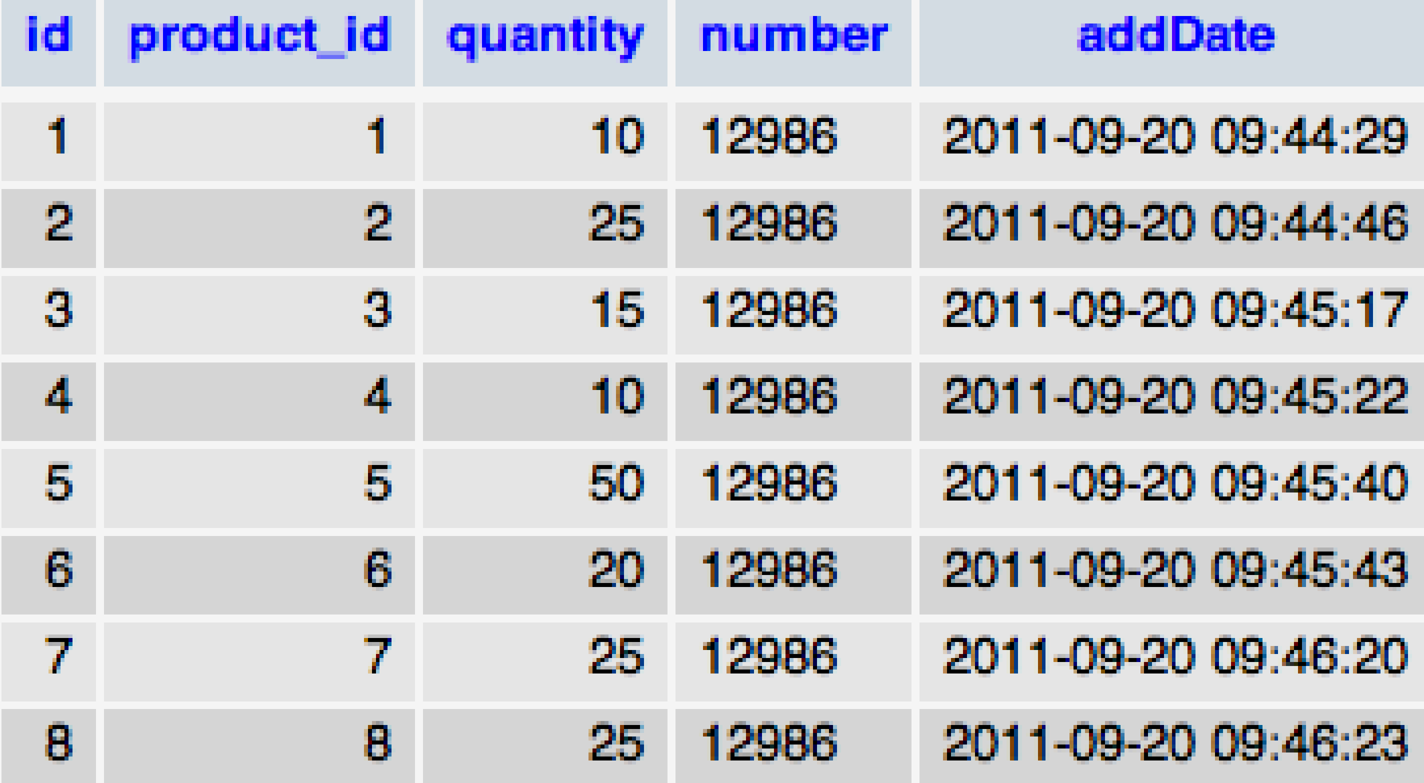
Physical storage (the number bytes required , and the data storage location)

OVERVIEW OF CODD’s RULE

- A **relational database management system (RDBMS)** is a database management system (DBMS) that is based on the relational model as introduced by *E. F. Codd*.
- A short definition of an RDBMS may be a DBMS in which data is stored ***in the form of tables and the relationship among the data is also stored in the form of tables.***
- *E.F. Codd,* the famous mathematician has introduced 12 rules (0-12)for the relational model for databases commonly known as **Codd's rules**. The rules mainly define what is required for a DBMS for it to be considered *relational*, i.e., an RDBMS.

Rule 1: INFORMATION RULE

- All information in the database is to be represented in one and only one way.
- All information in an RDBMS is **represented as values in the tables.**
- This is achieved by values in column and rows of tables.
- All information including table names, column names and column data types should be available in same table within the database.
- The basic requirement of the relational model.

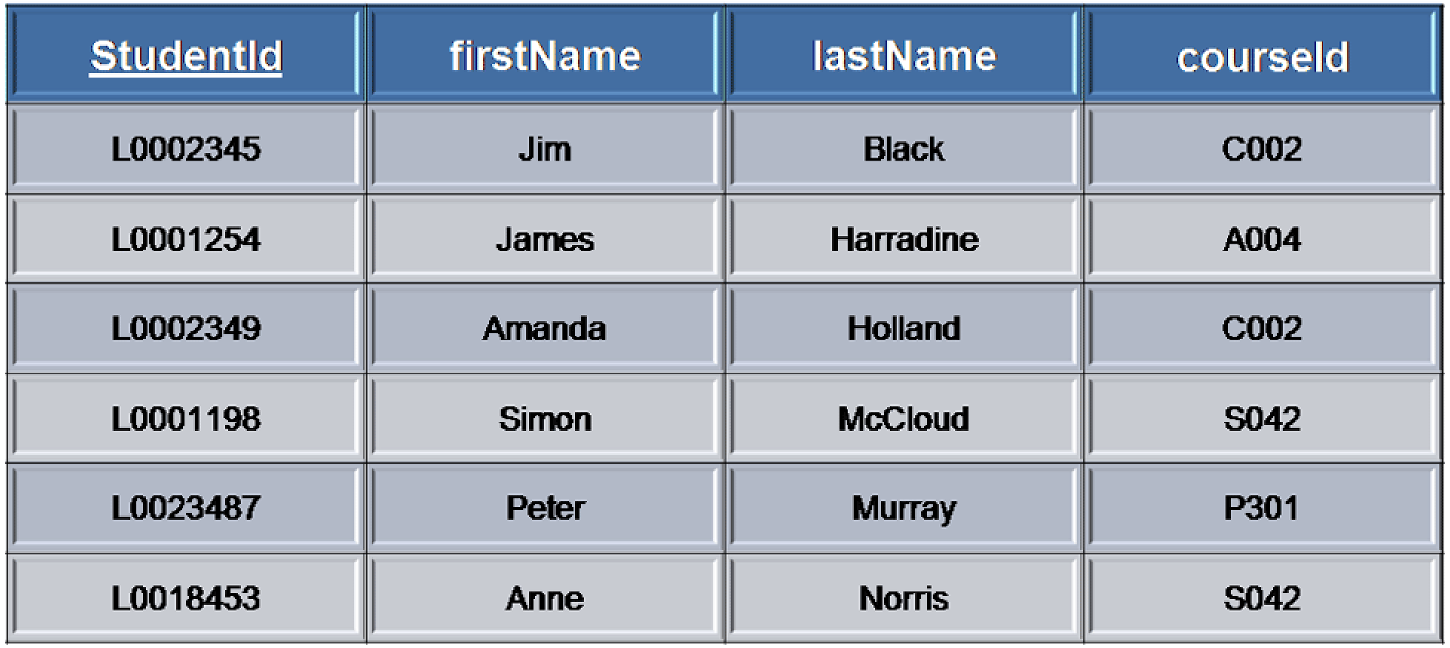


id	product_id	quantity	number	addDate
1	1	10	12986	2011-09-20 09:44:29
2	2	25	12986	2011-09-20 09:44:46
3	3	15	12986	2011-09-20 09:45:17
4	4	10	12986	2011-09-20 09:45:22
5	5	50	12986	2011-09-20 09:45:40
6	6	20	12986	2011-09-20 09:45:43
7	7	25	12986	2011-09-20 09:46:20
8	8	25	12986	2011-09-20 09:46:23

Rule 2: GUARANTEED ACCESS RULE (No Ambiguity)

- Each unique piece of data should be accessible by:table name+primary key(row) + attribute(column).
- All data are uniquely identified and accessible via this identity.
- Most RDBMS do not make the definition of the primary key mandatory and are deficient to that extent .

Primary Keys



StudentId	firstName	lastName	courseId
L0002345	Jim	Black	C002
L0001254	James	Harradine	A004
L0002349	Amanda	Holland	C002
L0001198	Simon	McCloud	S042
L0023487	Peter	Murray	P301
L0018453	Anne	Norris	S042

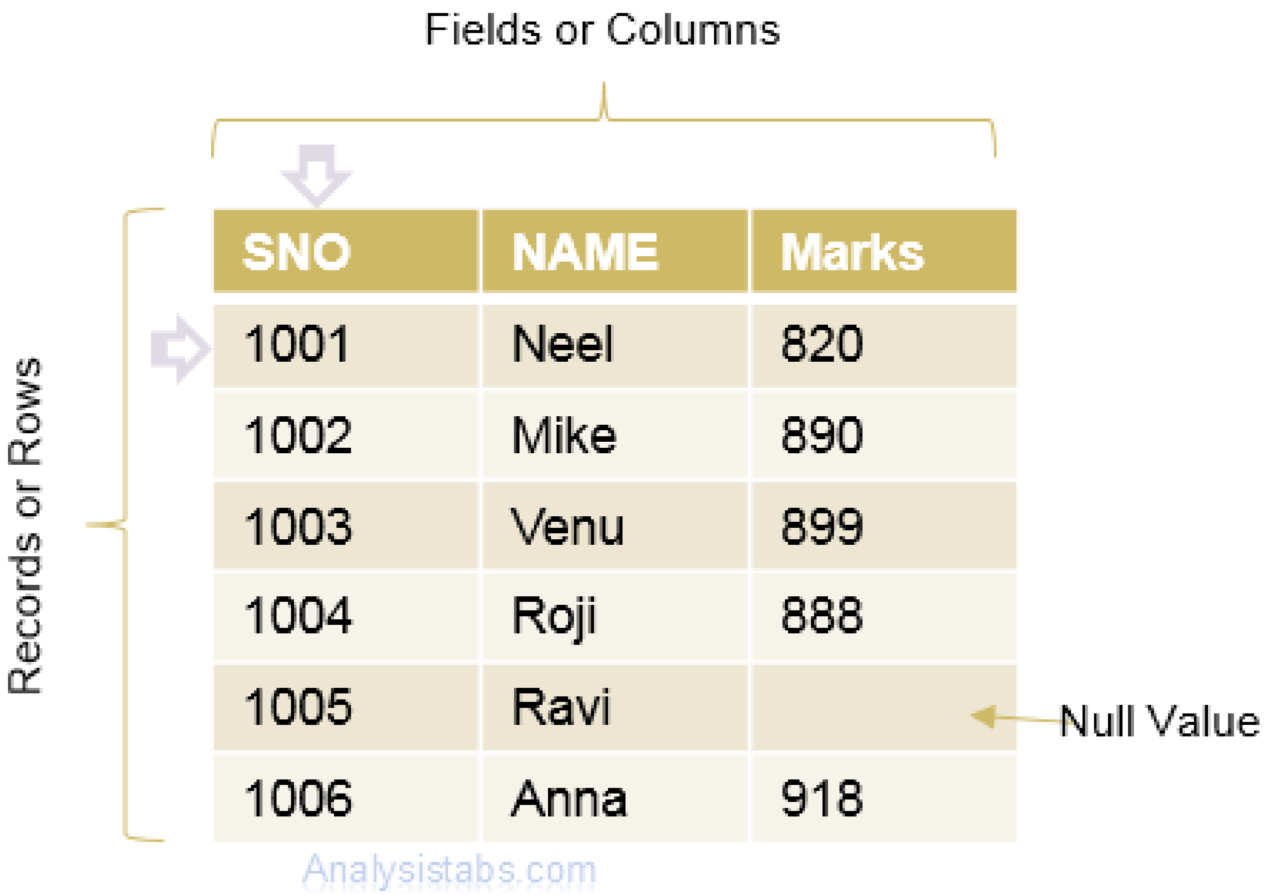
Rule 3 : Systematic treatment of null values

- "Null values (distinct from the empty character string or a string of blank characters and distinct from zero or any other number) are supported in fully relational DBMS for **representing missing information and inapplicable information** in a systematic way, independent of data type."

Null

- NULLs may mean: Missing data, Not applicable
- Should be handled consistently - Not Zero or Blank
- Primary keys — Not NULL
- This is distinct to zero or empty strings

Example Table: Student Marks



SNO	NAME	Marks
1001	Neel	820
1002	Mike	890
1003	Venu	899
1004	Roji	888
1005	Ravi	
1006	Anna	918

Rule 4:DATABASE DESCRIPTION RULE

- The data base description is represented at the logical level in the same way as-ordinary data, so that authorized users can apply the same relational language to its interrogation as they apply to the regular data.
- The authorized users can access the database structure by using common language i.e. SQL

Rule 5: COMPREHENSIVE DATA SUBLANGUAGE

A relational system may support several languages and various modes of terminal use .However, **there must be at least one language whose statements are expressible,** per some well-defined syntax, as character strings and that is comprehensive in supporting all the following items :

- *Every RDBMS should provide a language to allow the user to query the contents of the RDBMS and also manipulate the contents of the RDBMS*

- Data Definition (create,insert,update)
- View Definition
- Data Manipulation (alter,delete,truncate)
- Integrity Constraints (primary key,foreign key,null values)
- Authorization (GRANT , REVOKE)
- Transaction boundaries (begin,commit,rollbacketc)

.

Rule 6: VIEW UPDATING RULE

- View = ”Virtual table”, temporarily derived from base tables.

Example: If a view is formed as join of 3 tables, changes to view should be reflected in base tables.

BELOW IS A TABLE NAMED ‘STUDENT’.

RDBMS GIVES US THE FACILITY TO VIEW ONLY SOME PARTICULAR FIELDS ACCORDING TO OUR NEED WHICH ARE DIRECTLY ACCESSED FROM BASE TABLES WHEN REQUIRED.

SONALI	BCA-2	95	17231
TAMANNA	BCA-2	90	17236
RAJWINDER	BCA-2	90	17267
SAKSHI	BCA-2	86	17893
SADHANA	BCA-2	82	17453

TO VIEW ONLY THE NAME AND MARKS OF THE STUDENT TABLE WE CAN WRITE THE FOLLOWING SYNTAX:

```
CREATE VIEW RECORD
AS SELECT NAME,MARKS FROM STUDENT;
```

VIEW IS CREATED.

```
SELECT * FROM RECORD;
```

```
SONALI        95
TAMANNA       90
RAJWINDER     90
SAKSHI        86
SADHANA       82
```

RULE 7 : HIGH-LEVEL INSERT , UPDATE AND DELETE

This rule states that insert, update, and delete operations should be supported for any retrievable set rather than just for a single row in a single table.

It also perform the operation on multiple row simultaneously .

There must be delete, updating and insertion at the each level of operation. Set operation like union, all union , insertion and minus should also supported.

EXAMPLE:

Suppose if we need to change ID then it will reflect everywhere automatically.

EXAMPLE

- Create table:

```
SQL>CREATE TABLE STUDENT_DATA
{
NAME VARCHAR 2(20),
ROLL_NO VARCHAR 2(10
),
CLASS VARCHAR 2(20);
};
```

INSERT ION:

```
SQL>INSERT  INTO STUDENT_DATA('&NAME',&ROLL_NO,'&CLASS');
SQL>ENTER VALUE FOR NAME:KIRAN
 SQL>ENTER VALUE FOR ROLL_NO:4556
SQL>ENTER VALUE FOR CLASS:BCA
SQL>/
```

TABLE CREATED

NAME	ROLL_NO	CLASS
KIRAN	4566	BCA
RAHUL	3455	BCA

RULE 8 :LOGICAL DATA INDEPENDENCE RULE

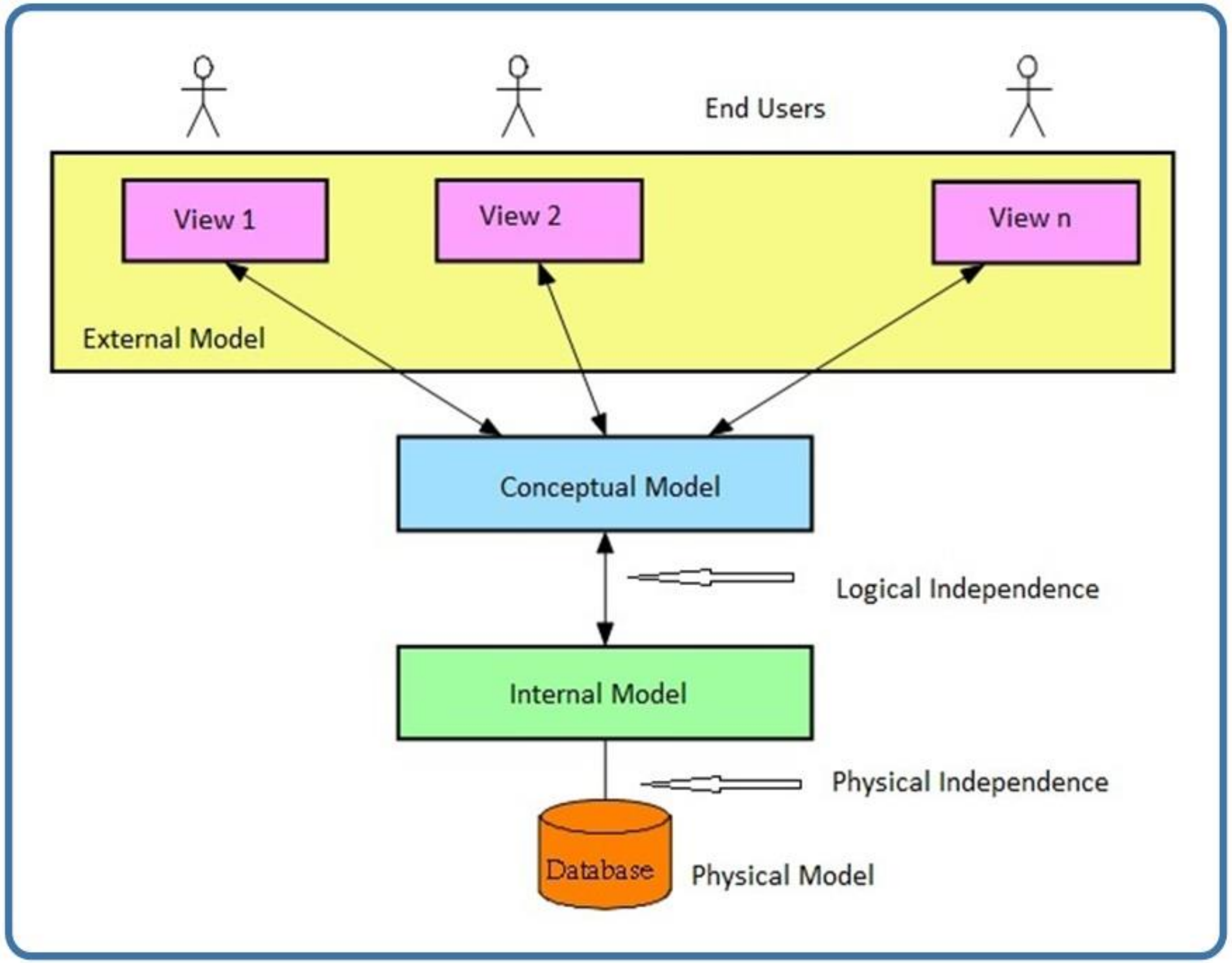
- What is independence?

The ability to modify schema definition in on level without affecting schema definition in the next higher level is called data independence

- The ability to change the logical (conceptual) schema without changing the External schema (User View) is called logical data independence.

- EXAMPLE:

The addition or removal of new entities, attributes, or relationships to the conceptual schema should be possible without having to change existing external schemas or having to rewrite existing application programs.


End Users
View 1
View 2
View n
External Model
Conceptual Model
Logical Independence
Internal Model
Physical Independence
Database
Physical Model

RULE 9:PHYSICAL DATA INDEPENDENCE

- The ability to change the physical schema without changing the logical schema is called physical data independence.

- This is saying that users shouldn't be concerned about how the data is stored or how it's accessed. In fact, users of the data need only be able to get the basic definition of the data they need.

- EXAMPLE:

  A change to the *internal schema*, such as using different file organization or storage structures, storage devices, or indexing strategy, should be possible without having to change the conceptual or external schemas.

- Data integrity refers to maintaining assuring the accuracy and consistency of data over its entire life cycle.

1. First insure that correct data type is used.
2. Check constraints: these allow column value to be checked agenized other column before insertion is allowed.

RULE 11 : DISTRIBUTION INDEPENDECE RULE

- “THE RELATION DATA BASE MANAGEMENT HAS DISTRIBUTION INDEPENDENCE”

- Distribution independence implies that user should not have to be aware of whether a database is distributed at different sites or not.
- Application program and adhoc request are not affected by the change in distribution of physical data. Application program will work even if the programs and data are moved on different site

- The RDBMS may spread across the more one system or several networks

RULE 12 : NON-SUBVERSION RULE

- There should be no way to modify to database structure other then through the multiple row data base language(SQL).

Example:

A relational system has a low-level (single-record-at-a-time) language, that low level cannot be used to subvert or bypass the integrity Rules and constraints expressed in the higher level relational language (multiple-records-at-a-time)."

Question 1

What is true regarding NULL value requirements confirming to Codd rule?

a. Null value should be zero.
b. Null value should be space.
c. Null value should represent missing information.
d. Either a or b.

Answer :

- c. Null value should represent missing information. According to Codd rule, Null value should not be any regular data like zero, spaces etc, but should represent that data is not available.

Question 2

According to Codd rule, how one should be able to access information about data structures like databases, tables etc.

- a. Should be directly accessible via all programming languages.
  b. Should be directly accessible via same query language used for data manipulation (e.g. select, update statements etc)
  c. Should be directly accessible via XML
  d. All of the above

Answer :

- b. Should be directly accessible via same query language used for data manipulation. For example, SQL can be used to retrieve information about the tables, the column types etc.

Question 3

Which is true regarding multi row update?

- a. Multiple row updates should be prohibited.
  b. Multiple row updates should be allowed only on tables without null values.
  c. Multiple row updates should be possible.
  d. Multiple row updates should be allowed only on integer data types

Answer :

- c. Multiple row updates should be possible.

Normalization

Normalization

- Normalization is a Process of **Organizing Data to minimize data redundancy (Duplicate data)**

There are 2 Major Goals of normalization

1. Eliminating redundant data (for example, storing the same data in more than one table) ensuring data dependencies make sense (only storing related data in a table)

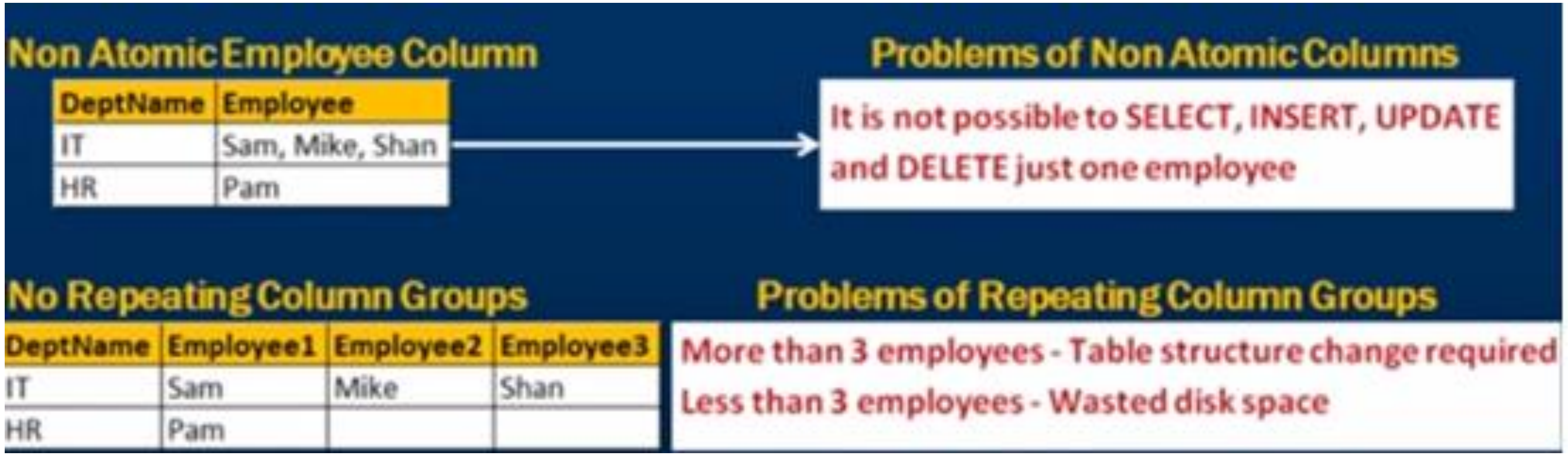
2. Normalization reduce the amount of space a database consumes

Types of normalization

- 1NF (First Normalization Form)
- 2NF (Second Normalization Form)
- 3NF (Third Normalization Form)

1NF (First Normalization Form)

- The table is said to first normal form if the data in each column is automic (no mutiple value or comma seperated)
- No repeating column groups



Non Atomic Employee Column

DeptName	Employee
IT	Sam, Mike, Shan
HR	Pam

Problems of Non Atomic Columns

It is not possible to SELECT, INSERT, UPDATE and DELETE just one employee

No Repeating Column Groups

DeptName	Employee1	Employee2	Employee3
IT	Sam	Mike	Shan
HR	Pam		

Problems of Repeating Column Groups

More than 3 employees - Table structure change required
Less than 3 employees - Wasted disk space

unnormalized and normalized

Course	**Student**
Database	Ajay, Raj, Vijay
Math	Kumar, Kiran

Course	**Student**
Database	Ajay
Database	Raj
Database	Vijay
Math	Kumar
Math	Kiran

2NF (Second Normalization Form)

The table is said to be second normal form :-

1. it should be in 1NF
2. no redundant data
3. create a relationship using foreign key

1NF

Customerid	custname	age	address	phone	ordered	orderdate	productid	product	price	qty	total
C001	Ajay	12	bangalore	11	1	1-1-2000	p1	books	400	4	1600
C001	Ajay	12	bangalore	11	1	3-4-2000	p2	Toys	400	2	800

2NF (Second Normalization Form)

Customer

Customerid custname age address phone

Orders

Customerid orderid orderdate product price qty total

- Customer-> Customerid (primary key)
- Orders-> Customerid (foriegn key)

3NF (Third Normalization Form)

The table is said to be third normal form if:-

1. Meet all conditions of 1NF and 2NF

2. does not contain any column which is not fully depend on primary key.

3NF

Customer

Customerid custname age address phone

Orders

Customerid orderid orderdate product price qty total

1. Total is depends upon price and qty
2. Age is depends upon (DOB)
3. Remove price it depends on product(instead replace with pid)

DATABASE KEYS

- A Key is a data item that exclusively identifies a record. In other words, key is a set of columns that is used to uniquely identify the record in a table. It is used to fetch or retrieve records or data-rows from data table according to the condition.
- The different database keys are:

1)Primary key

2)Candidate key

3)Foreign key

4)Alternate key

5)Secondary/Composite key

6)Super key

Primary key:

- A primary key is an attribute which has its own unique identity. A primary key never repeat itself. The primary key should be chosen such that its attributes are never or rarely changed.

Eg: Branch_id

- Candidate key:
- The keys which are eligible to be a primary key those set of keys are called as candidate keys. It can be a attribute or set of attributes that uniquely identifies a record. So a table can have multiple candidate key but each table can have maximum one primary key.
- Eg: Branch_code

Foreign key:

- Foreign key is used to generate the relationship between the tables. Foreign key is a field in database table that is primary key in another table.

- Eg: Branch_Id is a Foreign Key in Student_Information table that primary key exist in Branch_Info(Branch_Id) table.

- ## Alternate key:

- Alternate keys are candidate keys that are not selected as primary key. Alternate key can also work as a primary key. Alternate key is also called "**Secondary Key**".

- Eg: Branch_code

Composite key:

- Composite key is a combination of more than one attributes that can be used to uniquely identity each record. It is also known as “Compound” key. A composite key may be a candidate or primary key.

- Eg: Branch_name & Branch_code together acts as Primary key.
- Super key:
- Super key is a set of on e or more than one keys that can be used to uniquely identify the record in table. A super key is a combine form of Primary Key, Alternate key and Unique key and Primary Key, Unique Key and Alternate Key are subset of super key.
- Eg: { Branch_Name , Branch_Code }

ENTITY RELATION MODEL AND ATTRIBUTE TYPES

ER DIAGRAM

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation of an information system that depicts the relationships among people, objects, places, concepts or events within that system.

An ERD is a data modeling technique that can help define business processes and be used as the foundation for a relational database.

COMPONENTS OF ER DIAGRAM

- Entity
- Weak entity
- Weak entity relationship
- Attributes
- Relationships

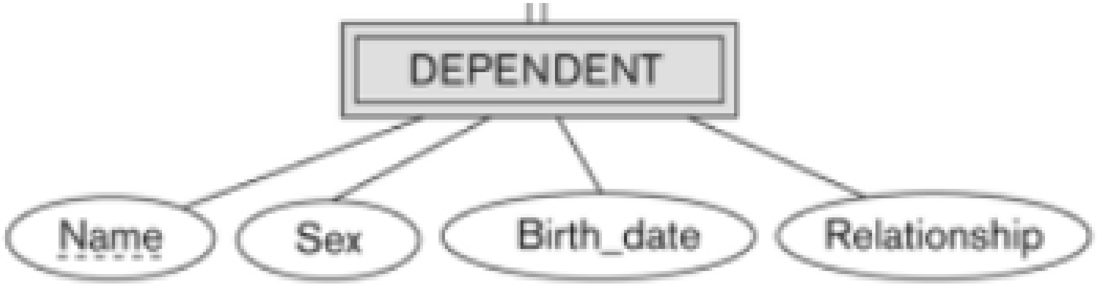
ENTITY

Entities are represented by means of rectangles. Rectangles are named with entity set that they represent.

Student
Subject

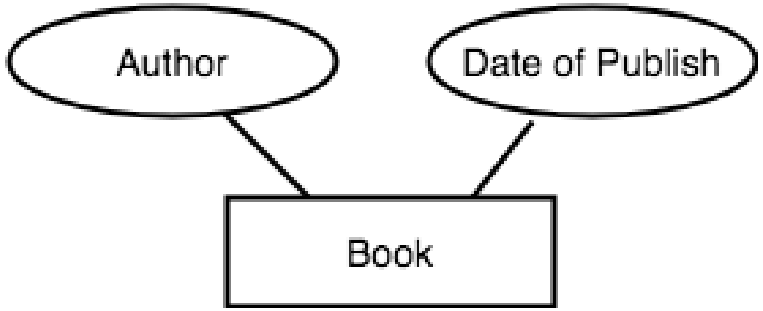
WEAK ENTITY

- A weak entity is simply an entity where it's existence depends on another entity.
- We can't logically have dependent(son, daughter, ..etc.) with the absence of the employee table.
- It is sketched same as a normal entity but wit double lines.



ATTRIBUTE

- Attributes are the properties of entities.
- Attributes are represented by means of ellipses.
- Every ellipse represents one attribute and is directly connected to its entity (rectangle).

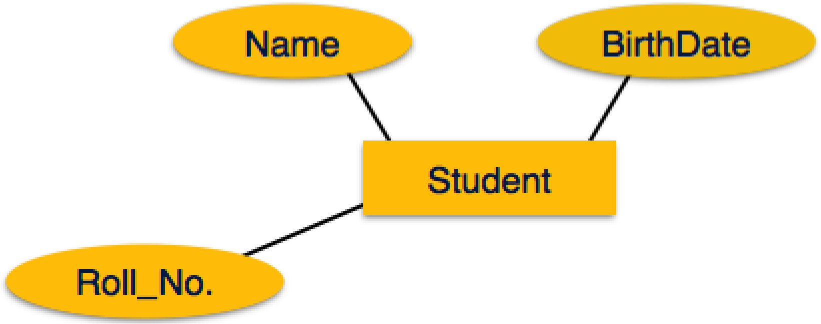


Types of Attributes

- Simple Attribute
- Compound Attribute
- Derived Attribute
- Multi-valued Attribute

Simple Attribute

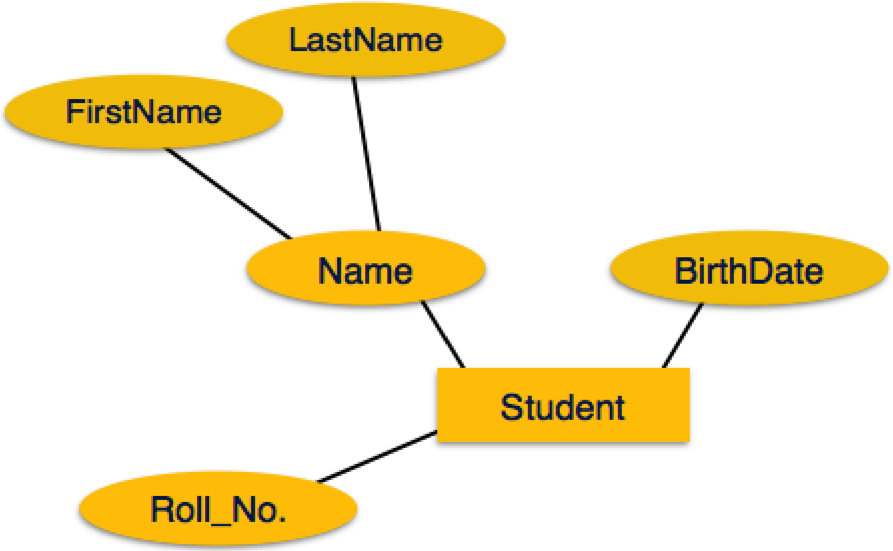
Simple attributes are atomic values, which cannot be divided further. For example, a student's phone number is an atomic value of 10 digits.



Compound Attribute

Composite attributes are made of more than one simple attribute.

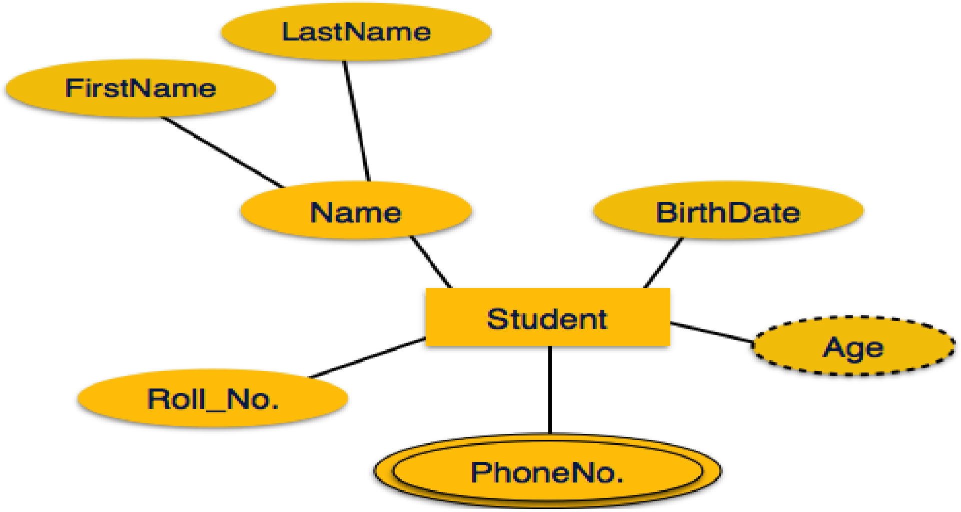
For example, a student's complete name may have first_name and last_name.



Derived Attribute

Derived attributes are the attributes that do not exist in the physical database, but their values are derived from other attributes present in the database.

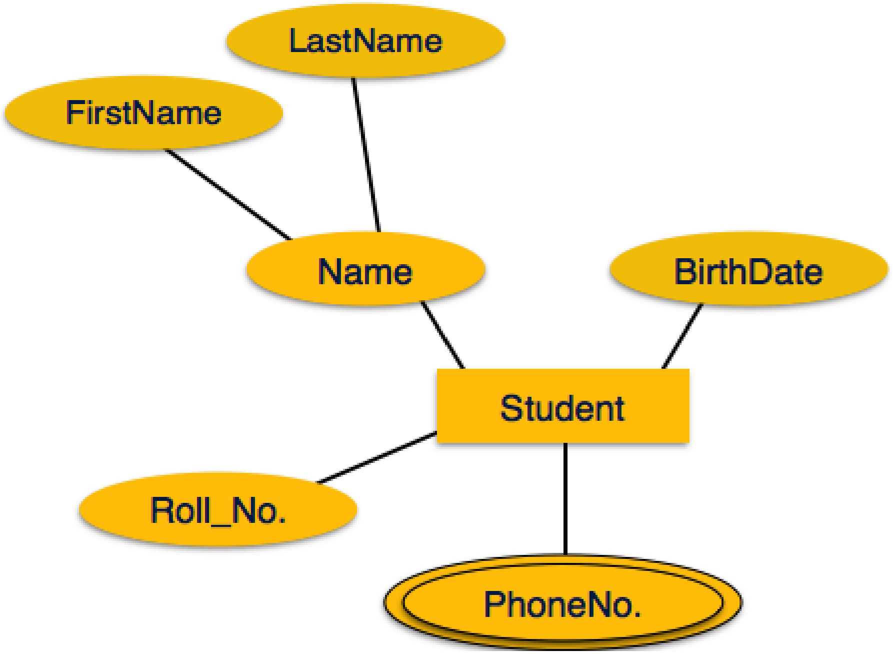
For example, age can be derived from data_of_birth.



Multi-valued Attribute

Multi-value attributes may contain more than one values.

For example, a person can have more than one phone number, email_address, etc.



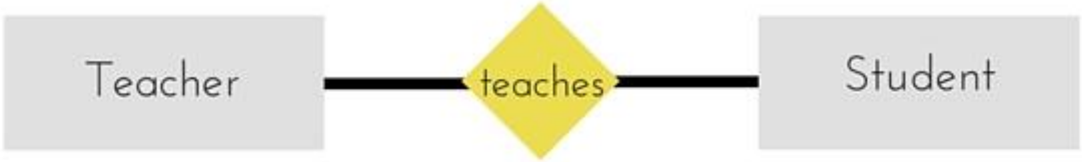
RELATIONSHIP

A Relationship describes relation between entities.

Relationships are represented by diamond-shaped box.

Name of the relationship is written inside the diamond-box.

All the entities (rectangles) participating in a relationship, are connected to it by a line.

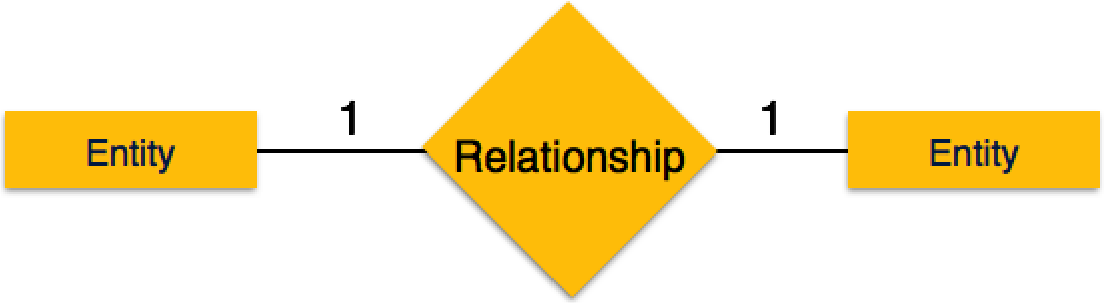


One to One Relationship

When only one instance of an entity is associated with the relationship, it is marked as '1:1'.

The following image reflects that only one instance of each entity should be associated with the relationship.

It depicts one-to-one relationship.

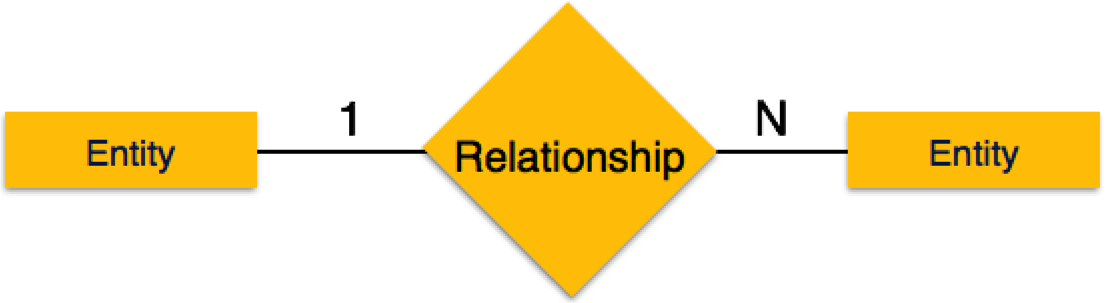


ONE TO MANY RELATIONSHIP

When more than one instance of an entity is associated with a relationship, it is marked as '1:N'.

The following image reflects that only one instance of entity on the left and more than one instance of an entity on the right can be associated with the relationship.

It depicts one-to-many relationship

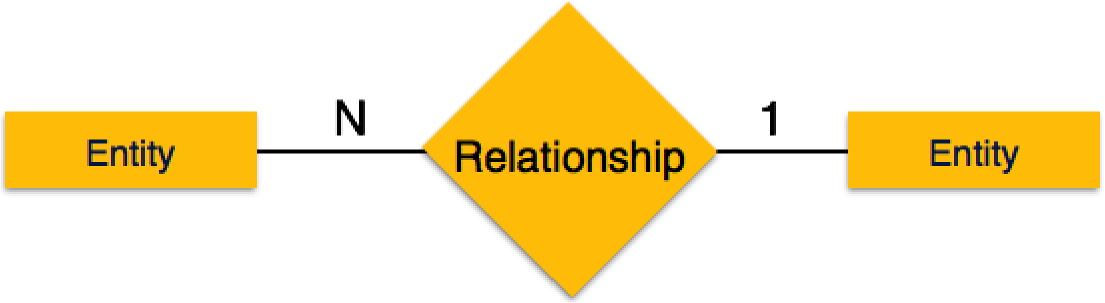


MANY TO ONE RELATIONSHIP

When more than one instance of entity is associated with the relationship, it is marked as 'N:1'.

The following image reflects that more than one instance of an entity on the left and only one instance of an entity on the right can be associated with the relationship.

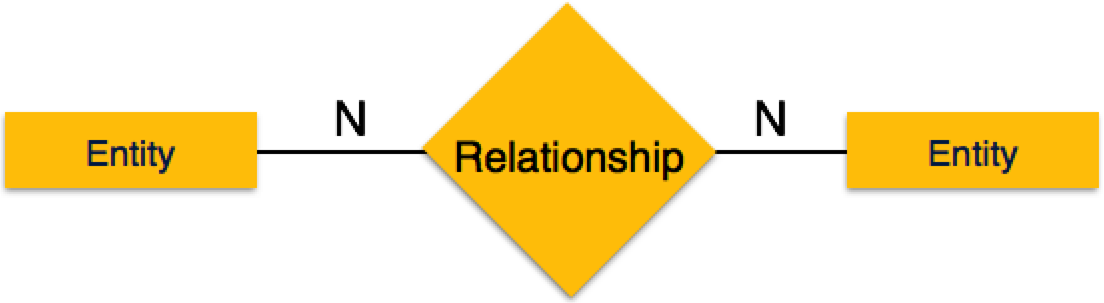
It depicts many-to-one relationship.



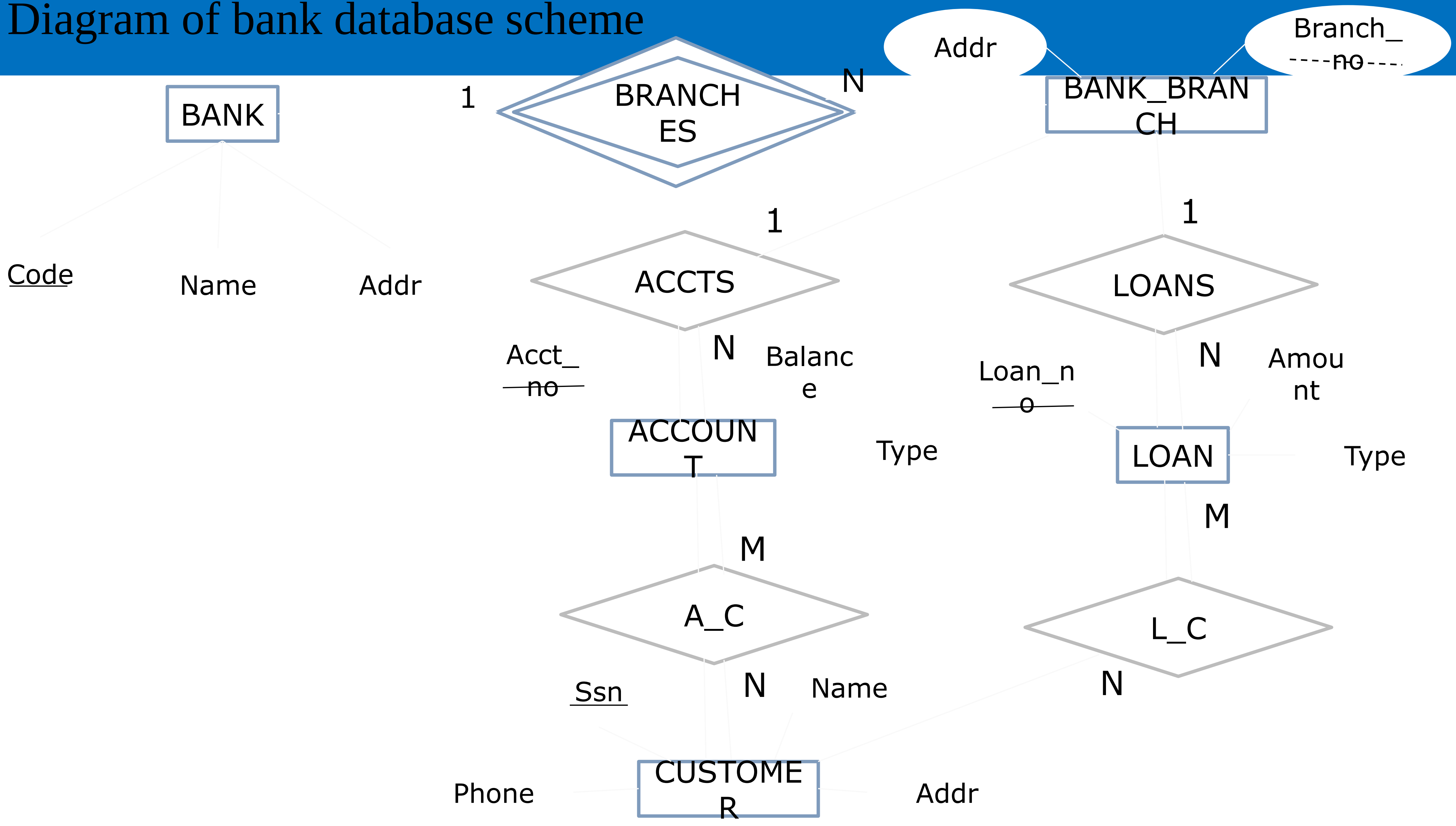
MANY TO MANY RELATIONSHIP

The following image reflects that more than one instance of an entity on the left and more than one instance of an entity on the right can be associated with the relationship.

It depicts many-to-many relationship.



ER Diagram of bank database scheme



Why DBMS?

- Database systems are basically developed for dealing with large amount of data.

- There are two main things in DBMS: Storing and Retrieval of Data
- Storage:
- According to the principles of database systems, the data is stored in such a way that it acquires lot less space as the redundant data has been removed before storage.
- Retrieval:
- Along with storing the data in an optimized and systematic manner, it is also important that we retrieve the data quickly when needed. Database systems ensure that the data is retrieved as quickly as possible.

DBMS Database Models

- A Database model defines the logical design and structure of a database and defines how data will be stored, accessed and updated in a database management system. While the Relational Model is the most widely used database model, there are other models too:

- Hierarchical Model
- Network Model
- Entity-relationship Model
- Relational Model

Hierarchical Model

- this type of structure is also called as inverted tree with the
- top referred as ROOT
- In this model , data is organized like an organization chart
- each node in a chart represents an entity and its subordinate
- entity describes the next level of hierarchical tree