Introduction to Database Management Systems (DBMS)

By:

BN SHANKAR GOWDA **BE, MIT(AUS)**

Associate professor

Dept. of Computer Science & Engineering

Bangalore Institute of Technology

Bangalore – 560 004

bnsgowda@gmail.com

Syllabus

- Introduction to Databases
- Database system- Concepts and Architecture
- Data Modeling using ER-model
- Relational Model
- Relational Algebra
- Relational Db Design ER_Relational Mapping
- Normalization
- SQL
- Transaction Processing and Concurrency
- Datawarehousing
- DDBMS, ORDBMS

Objectives

- An Overview of Database Management
- Database
- DBMS
- Database Systems
- Why Use Database
- Database Architecture
- An Example of the Three Levels
- Schema
- Data Independence
- Types Of Database Models
- Database Design Phases

The Database Technology

- Database plays a critical role in every aspect of day-to-day life where computers are put to use including business, engineering , medicine, education, law, e-commerce…….etc that we may or maynot be aware of using one.
- Eg:
  - ATM
  - Purchase from a supermarket
  - Purchase using a credit card
  - Reserving a ticket for a train/bus/ flight

The Database Technology

Data:

is representation of raw facts that are recorded

- that may have an implicit meaning and is
- Generally voluminous

- Eg: shankar 2345678

Database:

- Place where data resides
- Computer based record-keeping system
- Collection of interrelated (persistent) data
- Records & maintains data

The Database Technology

- **Information :**

is the organized data/processed data, wherein the data is made meaningful with the use of known symbols in a particular context

  - Used for decision making
  - Needed for conduct of business
  - Normally stored in a file for later use, auditing & evaluation

Information at one place may be data at other place, therefore data and information are interchangingly used.

- **Knowledge :**

is the awareness and understanding of a set of information. It is derived from information.

Database Management System ?

- A Complex software or collection of programs that enables users to create, manipulate, control and maintain data in the database.
- It is a layer of abstraction/Interface between the Application Programs and the Database.
- It is a system software that manages and controls access to database.
- Eg: oracle, SQL server, Dbase, Ingres

Traditional ways of storing data in files

- Data is stored in the form of records in the files.
- Records consist of various fields which are delimited by a space , comma , tab etc.
  - Shankar 123456 male
  - Shankar,123456,male
- There used to be special characters to mark end of records and end of files.

4176	Aniruddha Sarkar	SBU1
4181	Manoj Saha	SBU1
4183	Moushumi Dharchoudhury	SBU1
4203	Suryanarayana D.V.S.S.	SBU1
4204	Vivek Rai	SBU1

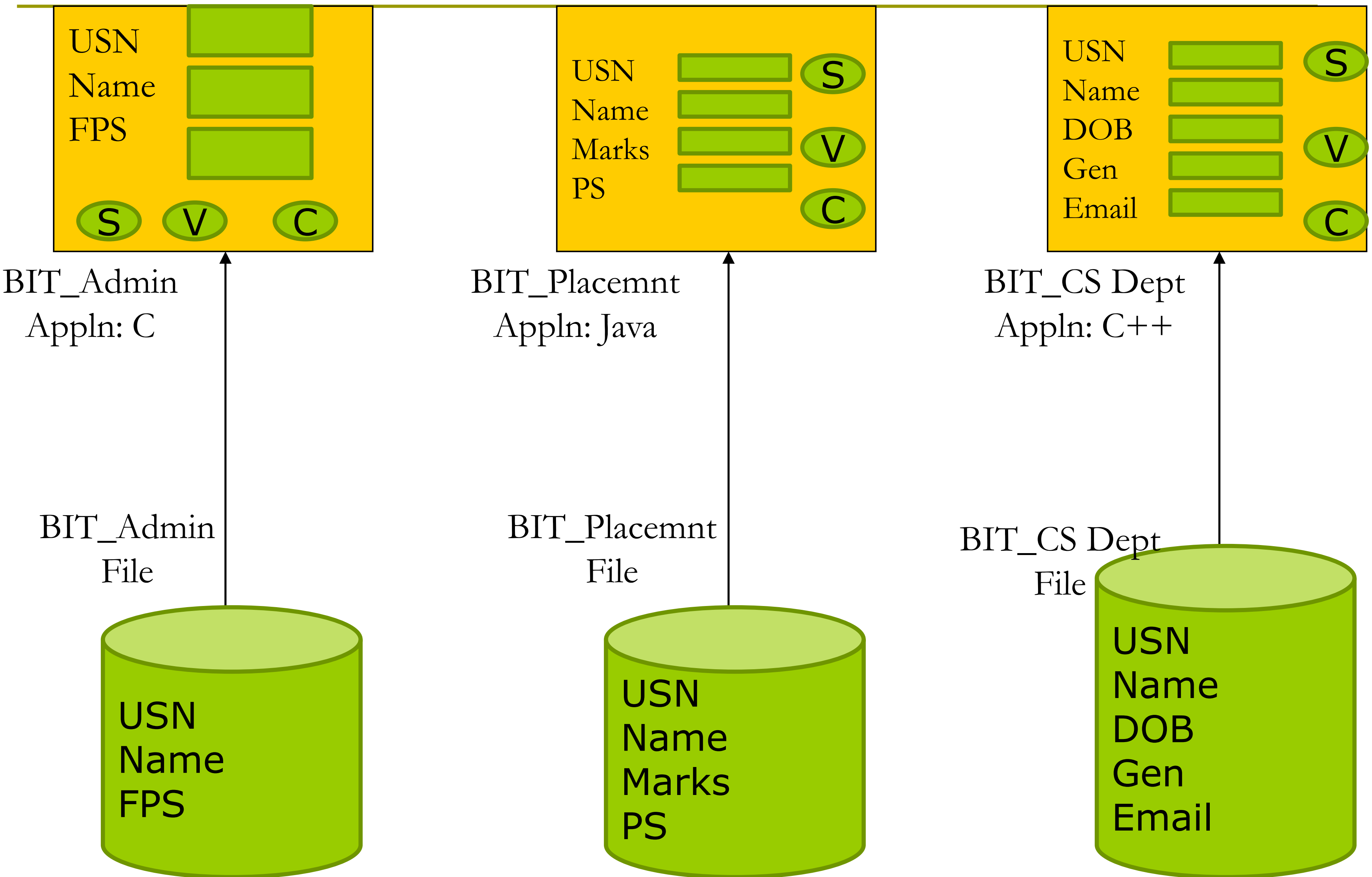
Traditional ways of storing data in files

Lets observe the Retail Application Table

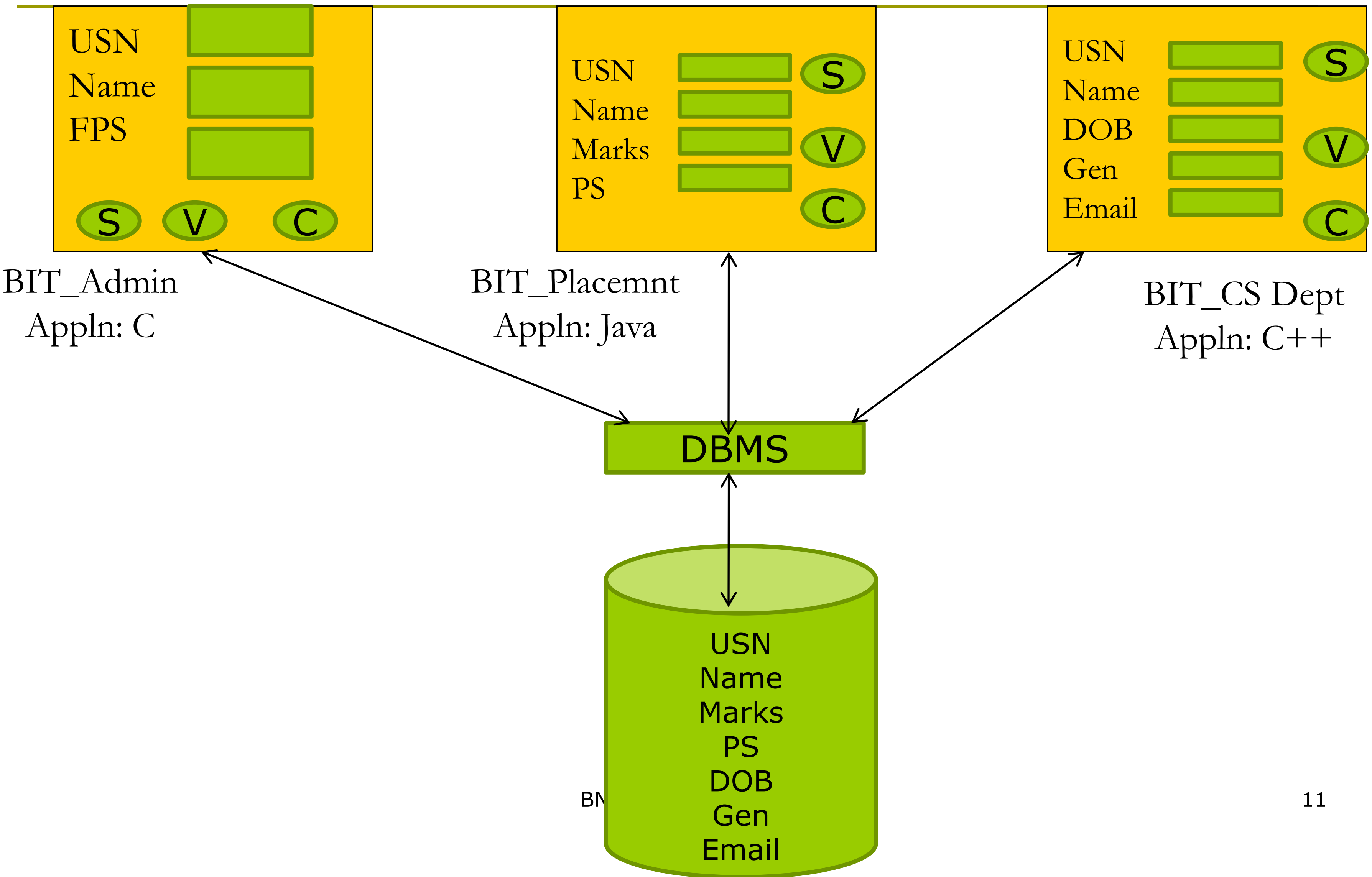
CustomerDetails	ItemDetails	PurchaseDetails
1001 John 1500012351	STN001 Pen 10 A	5 50
1002 Tom 1200354611	BAK003 Bread 10 A	1 10
1003 Maria 2134724532	GRO001 Potato 20 B	1 20

Each row of the table Represents the information of a customer who has purchased an item .

Traditional File Structure

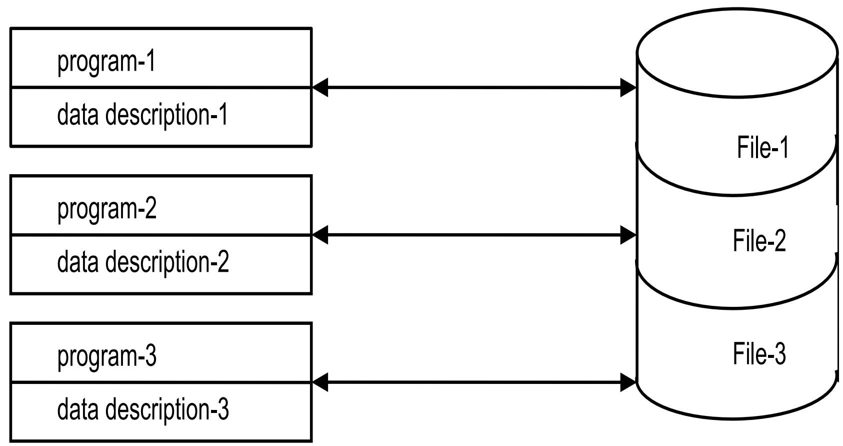


Database Mgmt. System

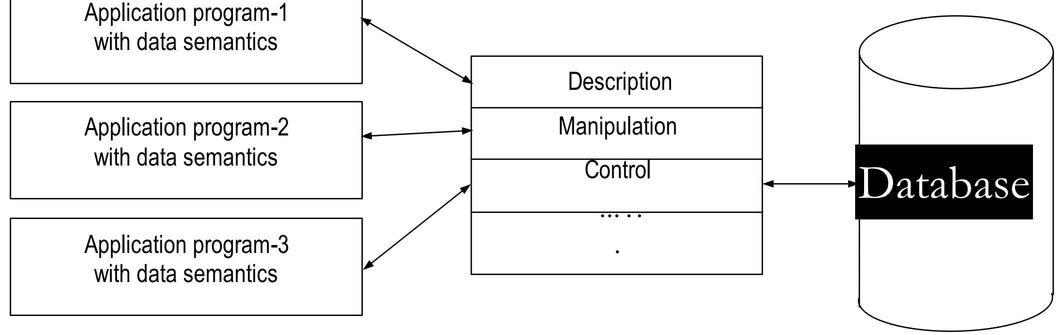


Traditional v/s DBMS

File System approach



DBMS approach

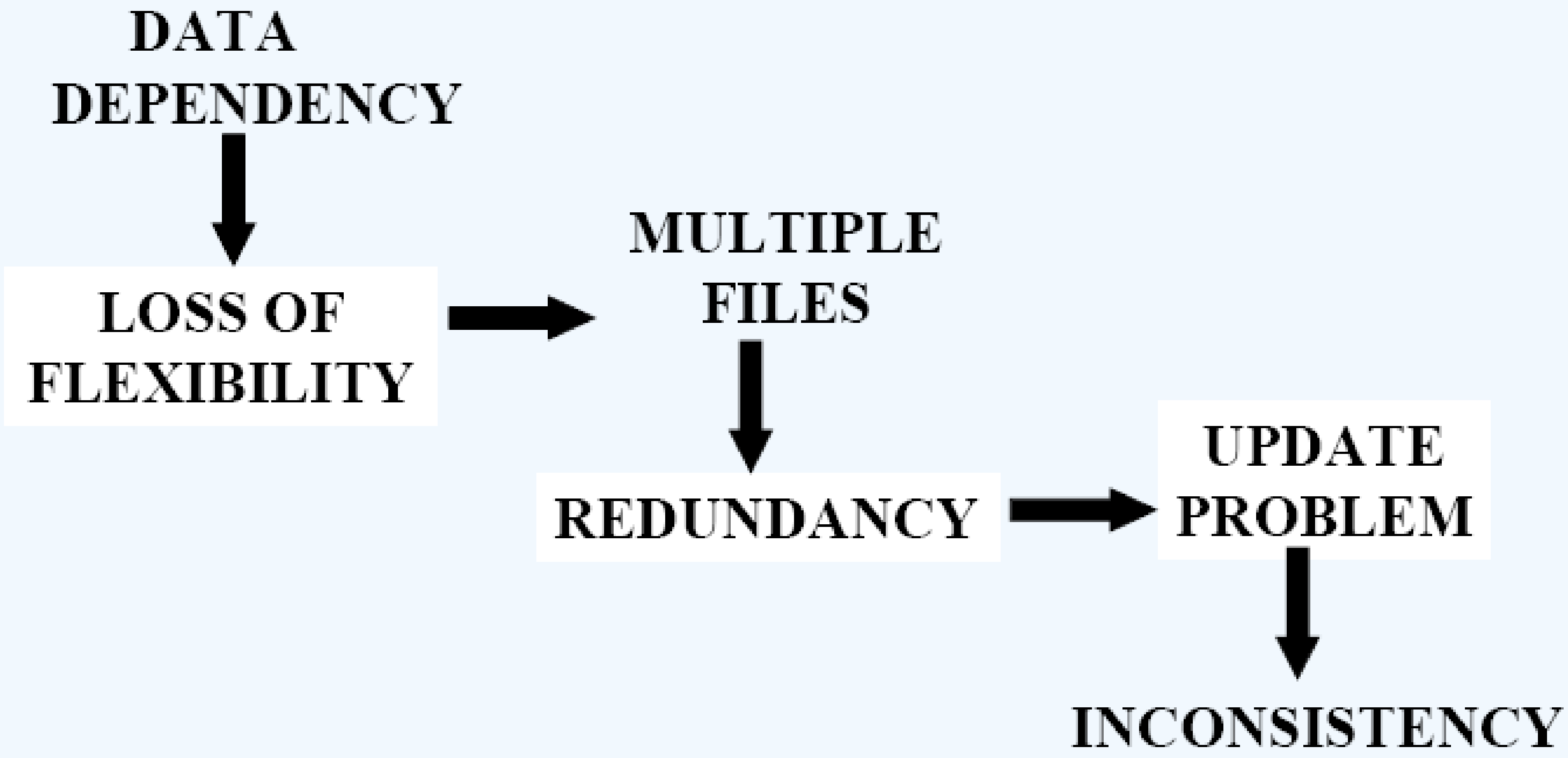


Data definition in file systems is part of application programs

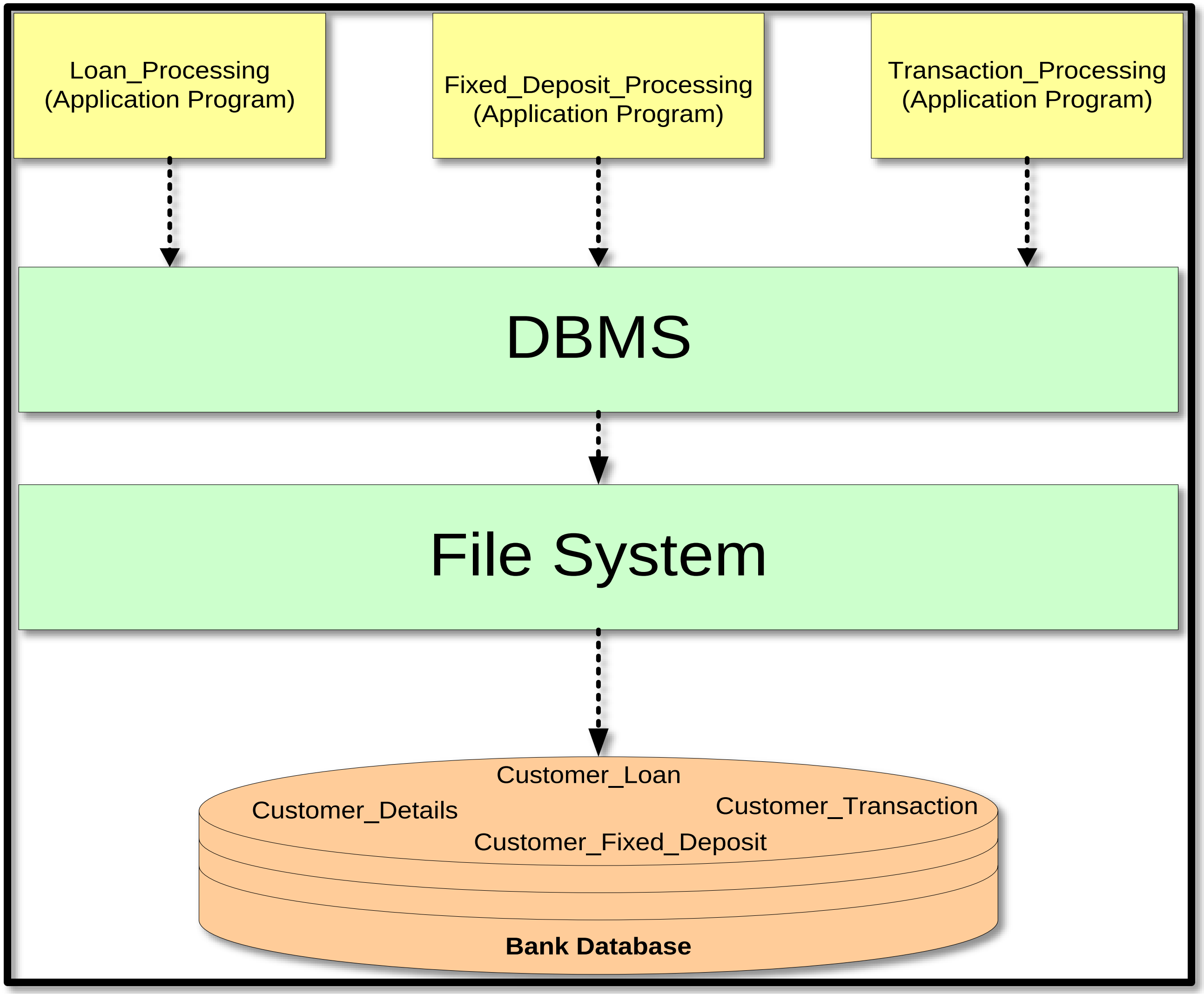
Problems: Traditional approach

- Data spread across multiple files where dependent on each other . This led to loss of flexibility. Since data was spread across multiple files and there was no formal way of maintaining relationships between these files, the same information was repeated in multiple files. **This led to redundancy**.
- When a particular data had to be updated , say for example, an employee's information to be deleted, it has to be done in all the files where the employee data occurs. If the deletion is missed out on even one of the files, it would leave **the data inconsistent**.

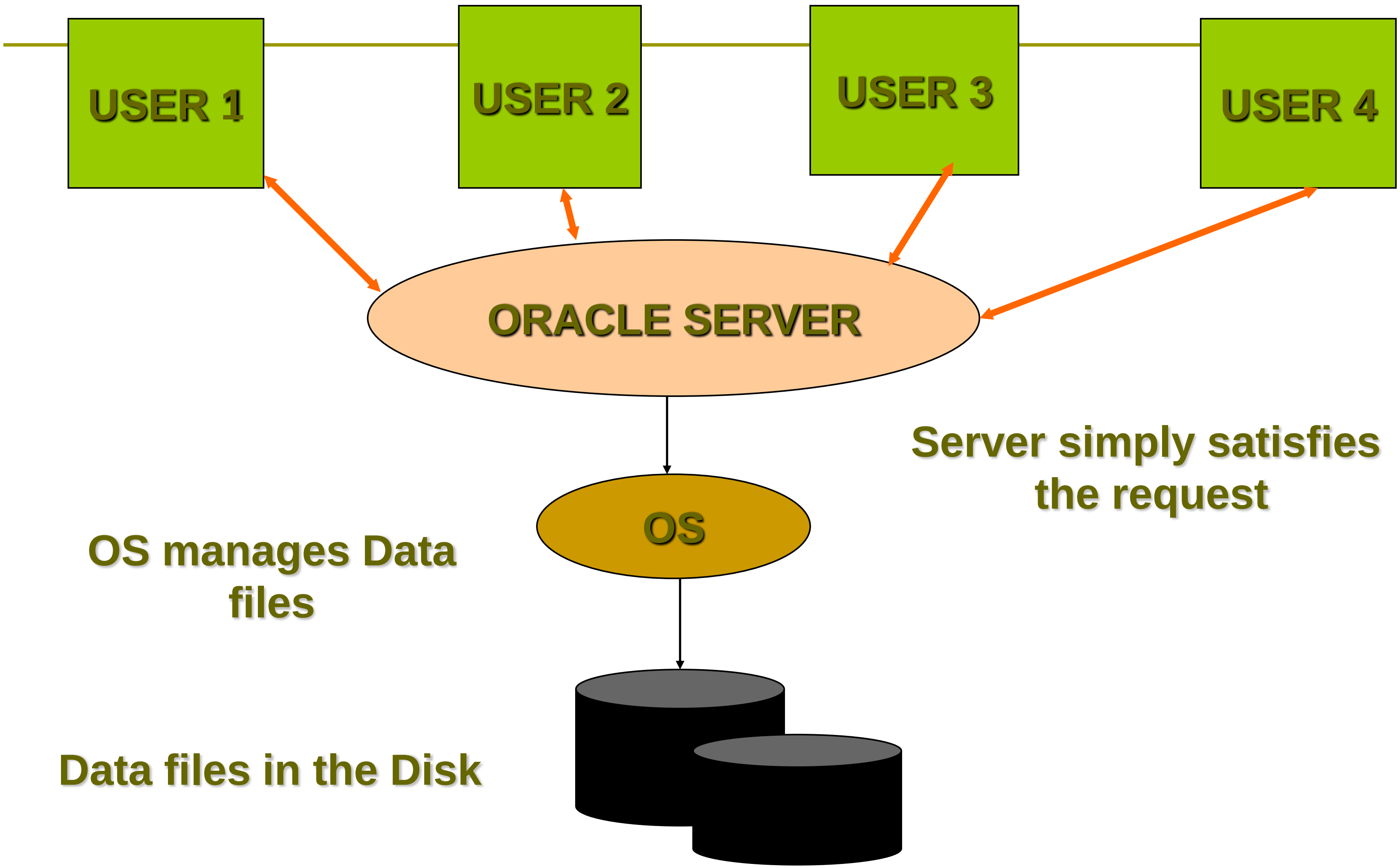
Problems:Traditional approach



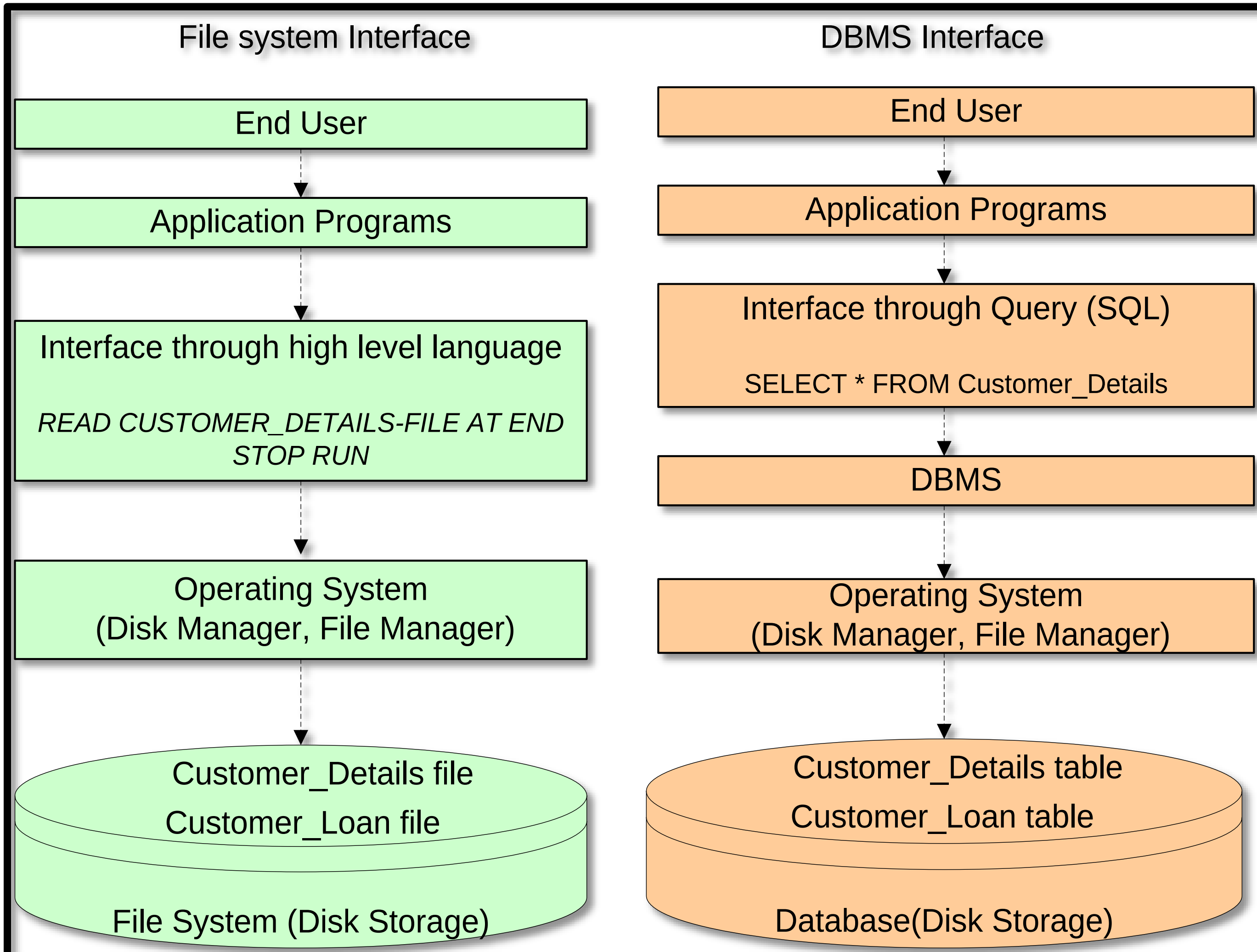
Where does the DBMS fit in?



Platform Independency



Difference Between File and DBMS Operations

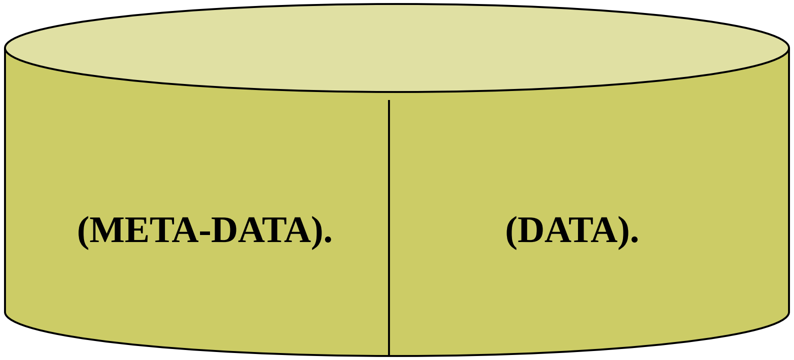


Characteristics of Database approach verses file-processing approach

- Self-describing nature of database system
- Insulation between data and programs, and data abstraction
- Support for multiple views of data
- Sharing of data and multi-user transaction processing

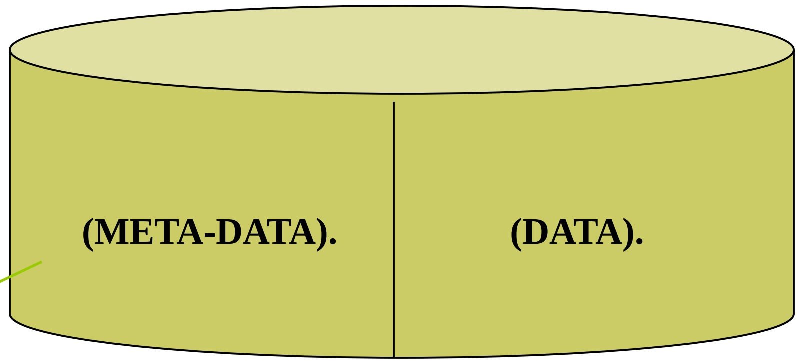
Self-describing nature of database system

- The characteristic of database approach is that it not only contains the data, but also the complete definition or description of the data structure, constraint, storage location……



Schema

- Schema: The description of a database in terms of a data model is called as database schema.
- A schema is specified during the database design and is not expected to change frequently.



description of a database

Logical Structure

TABLE SPACE

SEGMENTS

EXTENTS

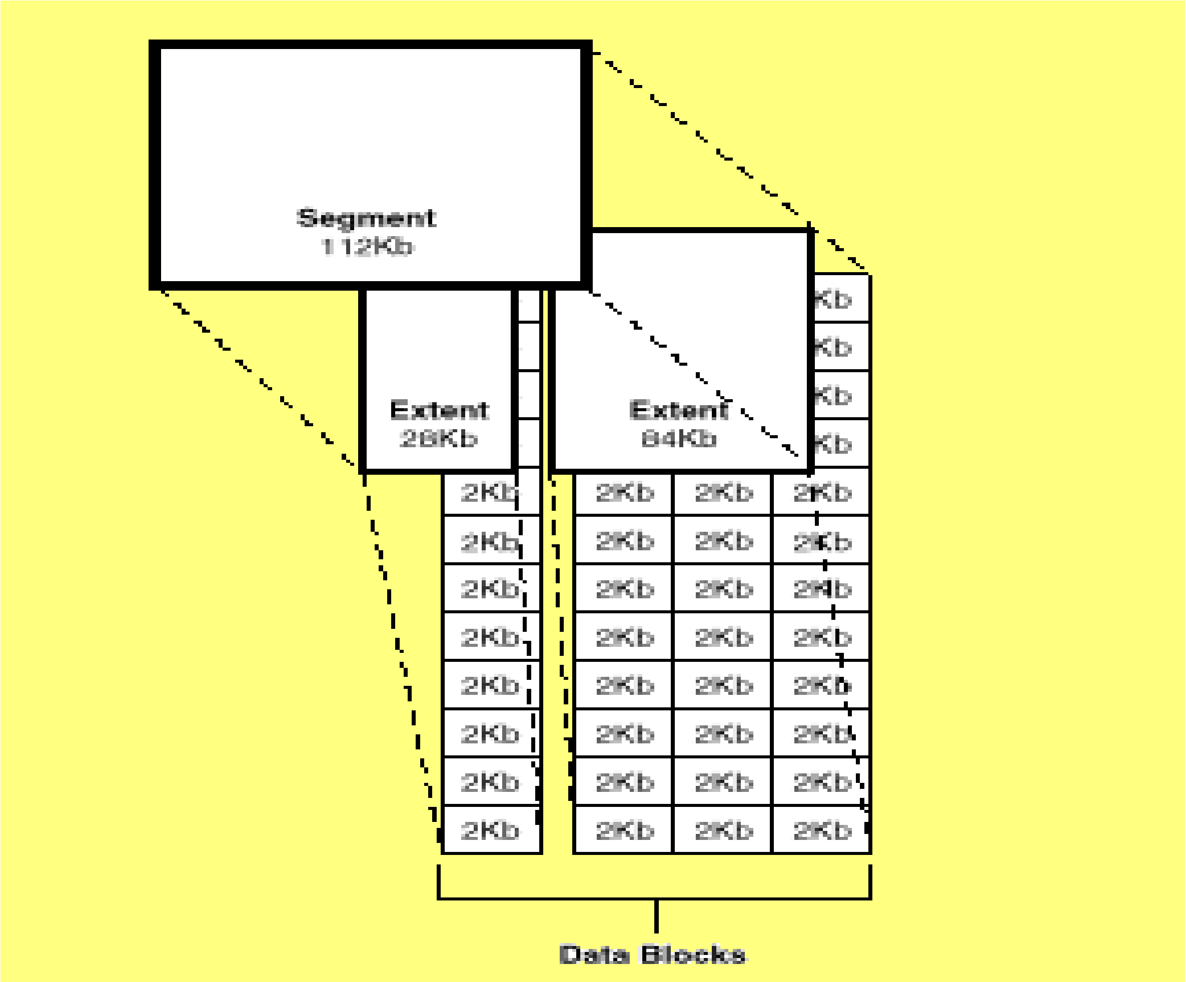
BLOCKS/PAGES

PAGE1	PAGE2	PAGE3	PAGE4
PAGE5	PAGE6	…………….	PAGEN

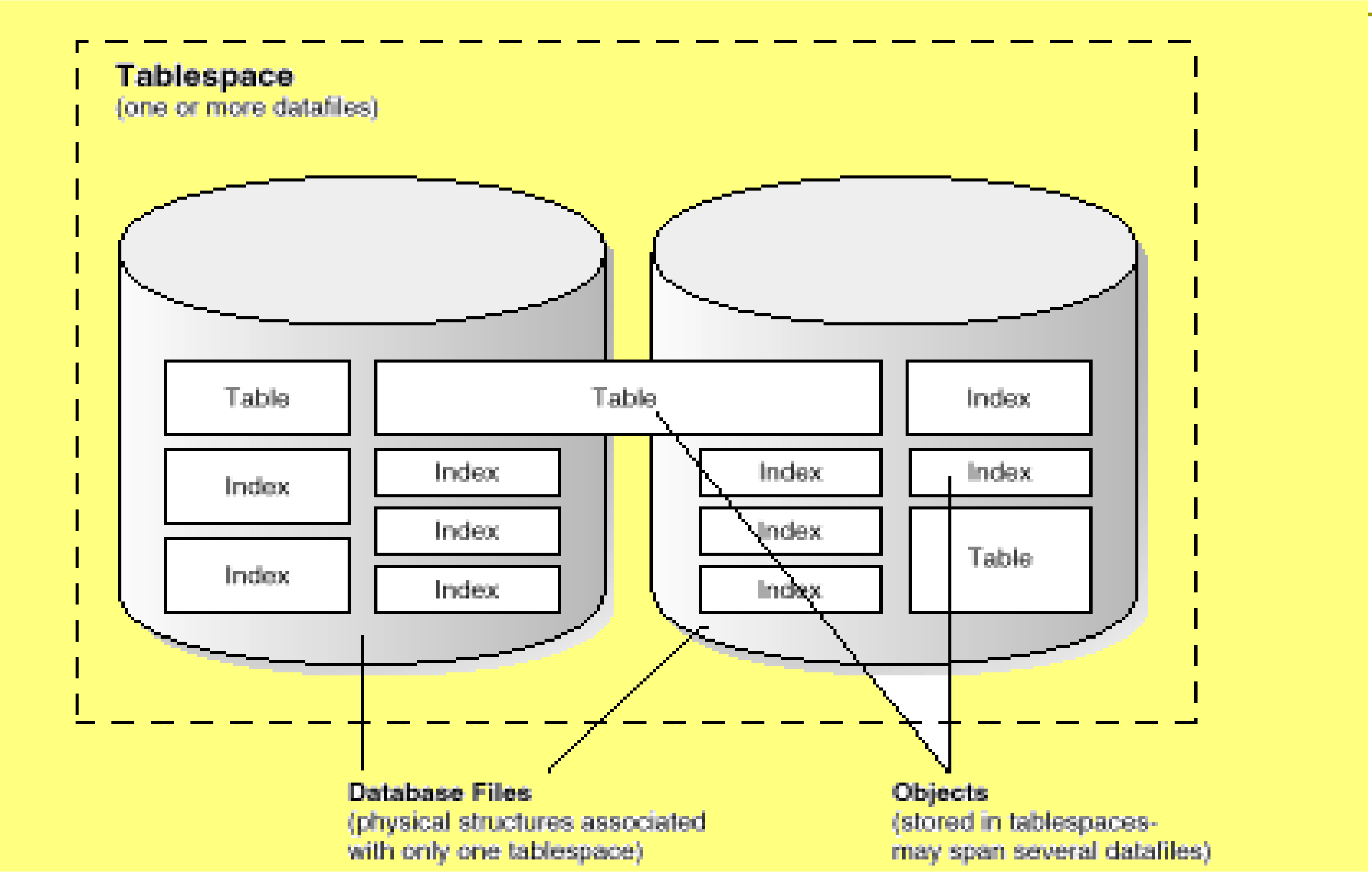
Data on External Storage

- Data is stored on Secondary storage – DISKS.
- The TRACK is divided into equal-sized DISK BLOCK or PAGES.
- When needed, data is fetched from DISK to MAIN MEMORY in units of Blocks or *pages.*
- Typical Block range from 512 Bytes to 4096 Bytes.
- The block size is fixed during initialization and cannot be changed dynamically.
- A Block/Page holds one or more RECORDS.
- FILES: is a sequence of RECORDS.

Blocks, Extents , Segments

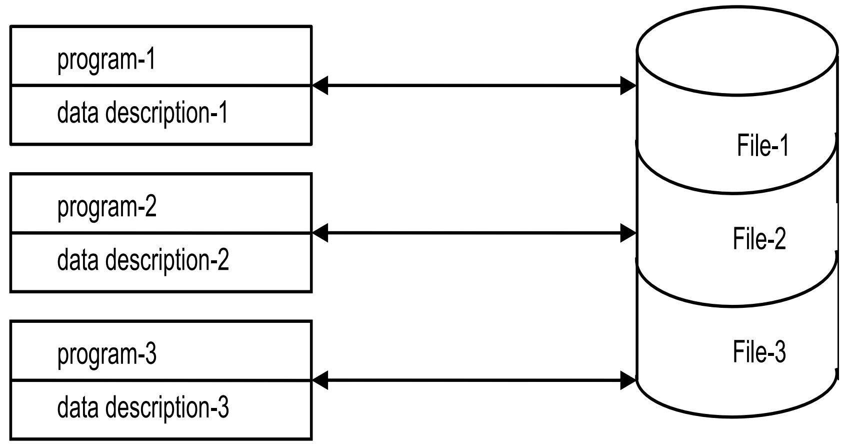


Data Files and Tablespaces

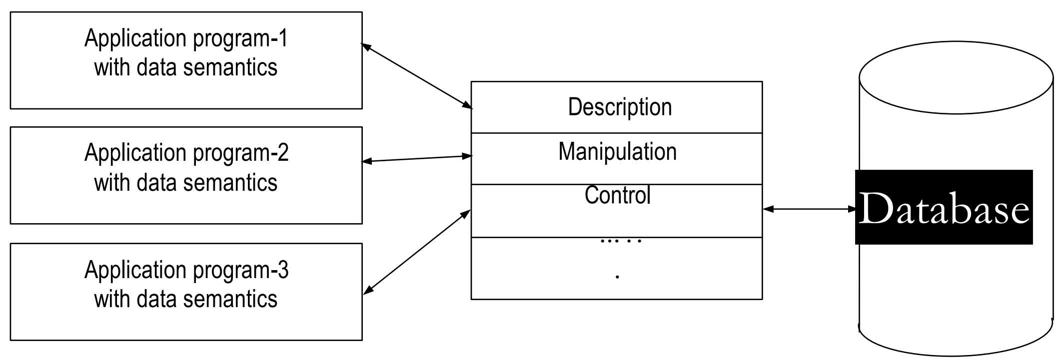


Insulation between data and programs, and data abstraction

File System approach



DBMS approach

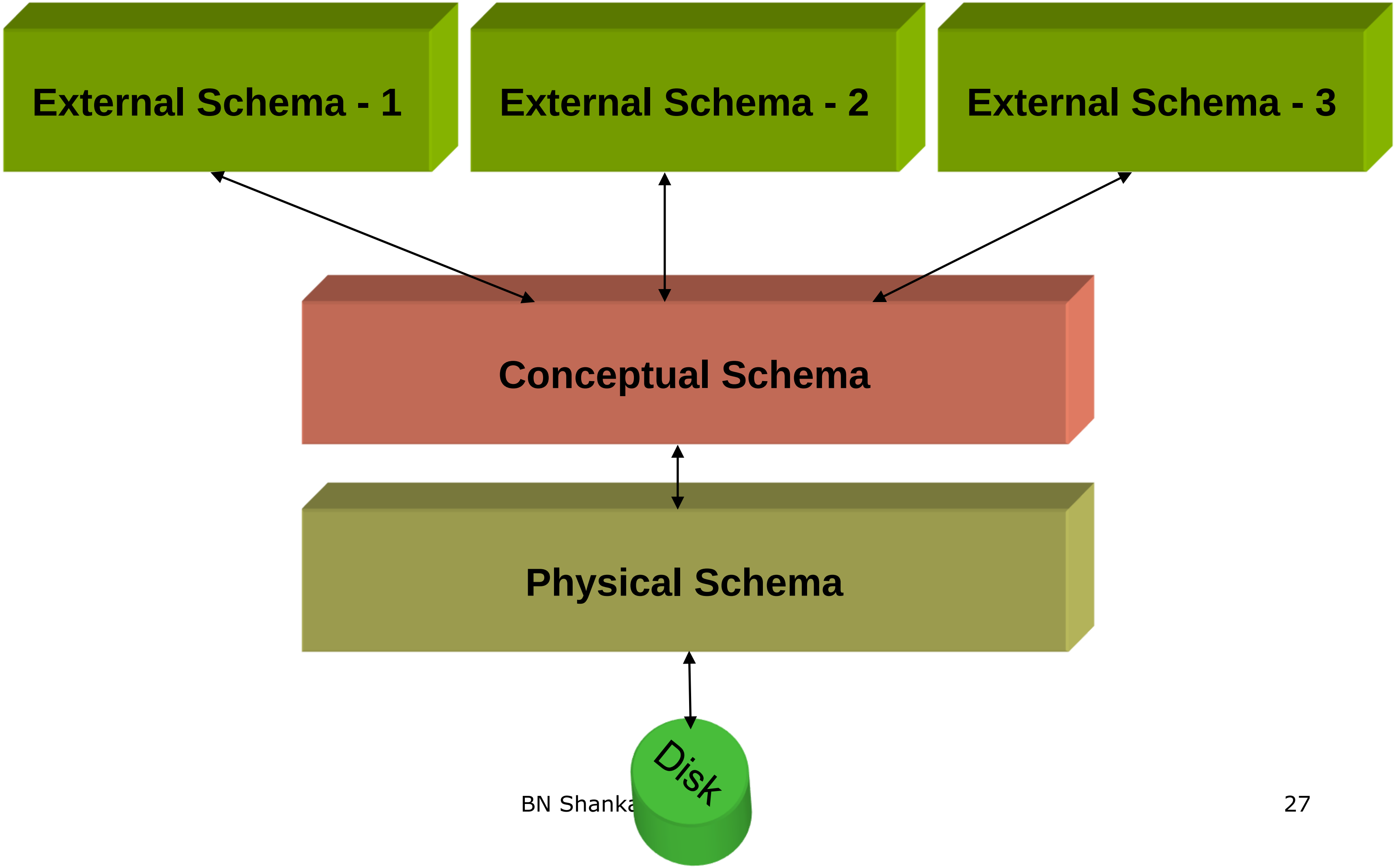


Data definition in file systems is part of application programs

DATABASE ARCHITECTURE

- ANSI/SPARC 3-Level DB Architecture
- Metadata - What is it? Why is it important?
- ISO Information Resource Dictionary System (ISO-IRDS)

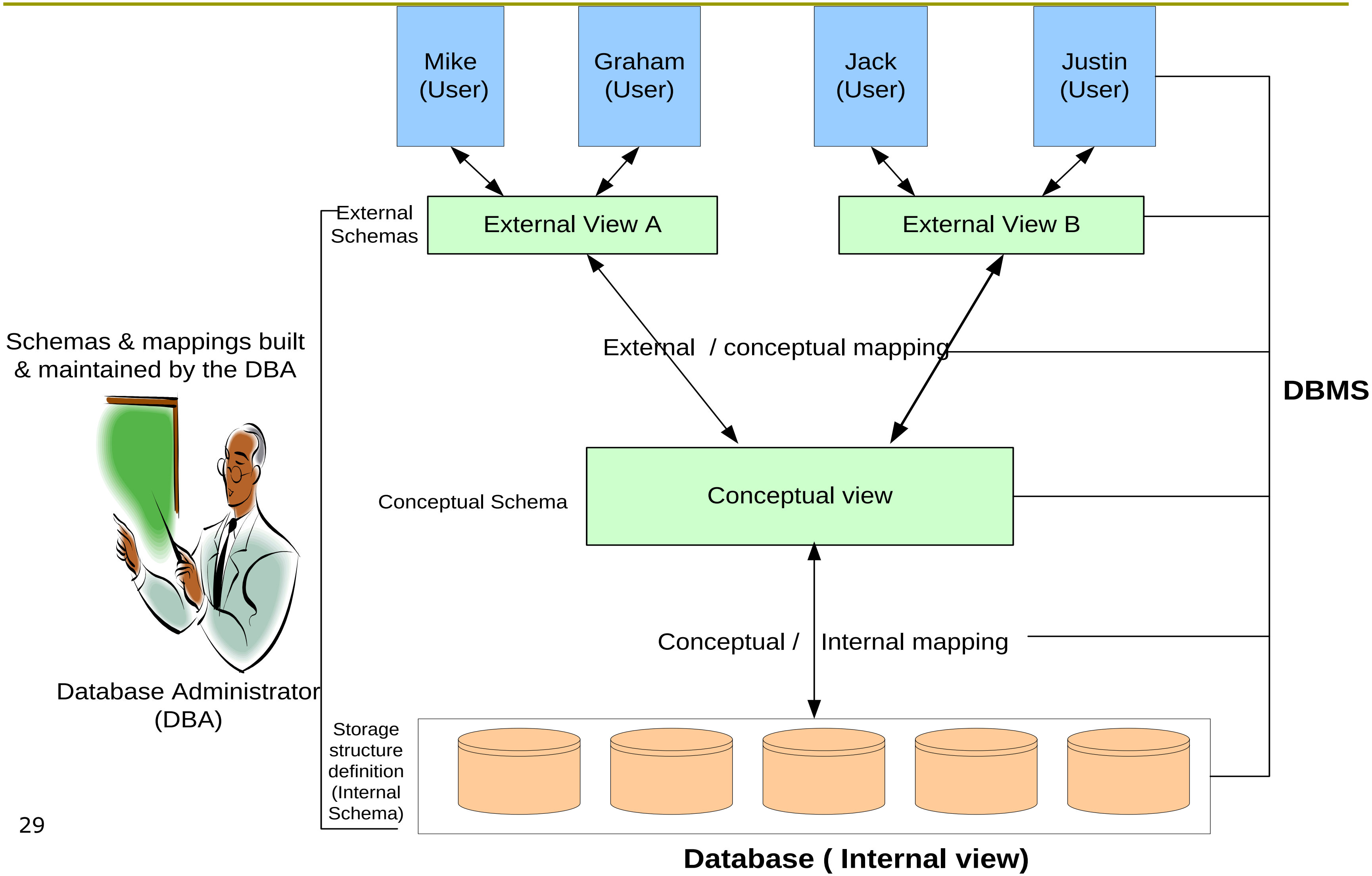
Three-Layer Abstraction



Three-Schema Abstraction

- **EXTERNAL SCHEMA:**
- **USE OF DATA:** **Describes several VIEWS of the database based on the database model.**
- **CONCEPTUAL SCHEMA:**
- **MEANING OF DATA:** **Describes the STORED DATA in terms of the data model of the DBMS**
- **INTERNAL SCHEMA:**
- **STORAGE OF DATA:** **Describes the ACTUAL STORAGE details of the relations described in conceptual schema.**

Detailed System Architecture



Physical Schema

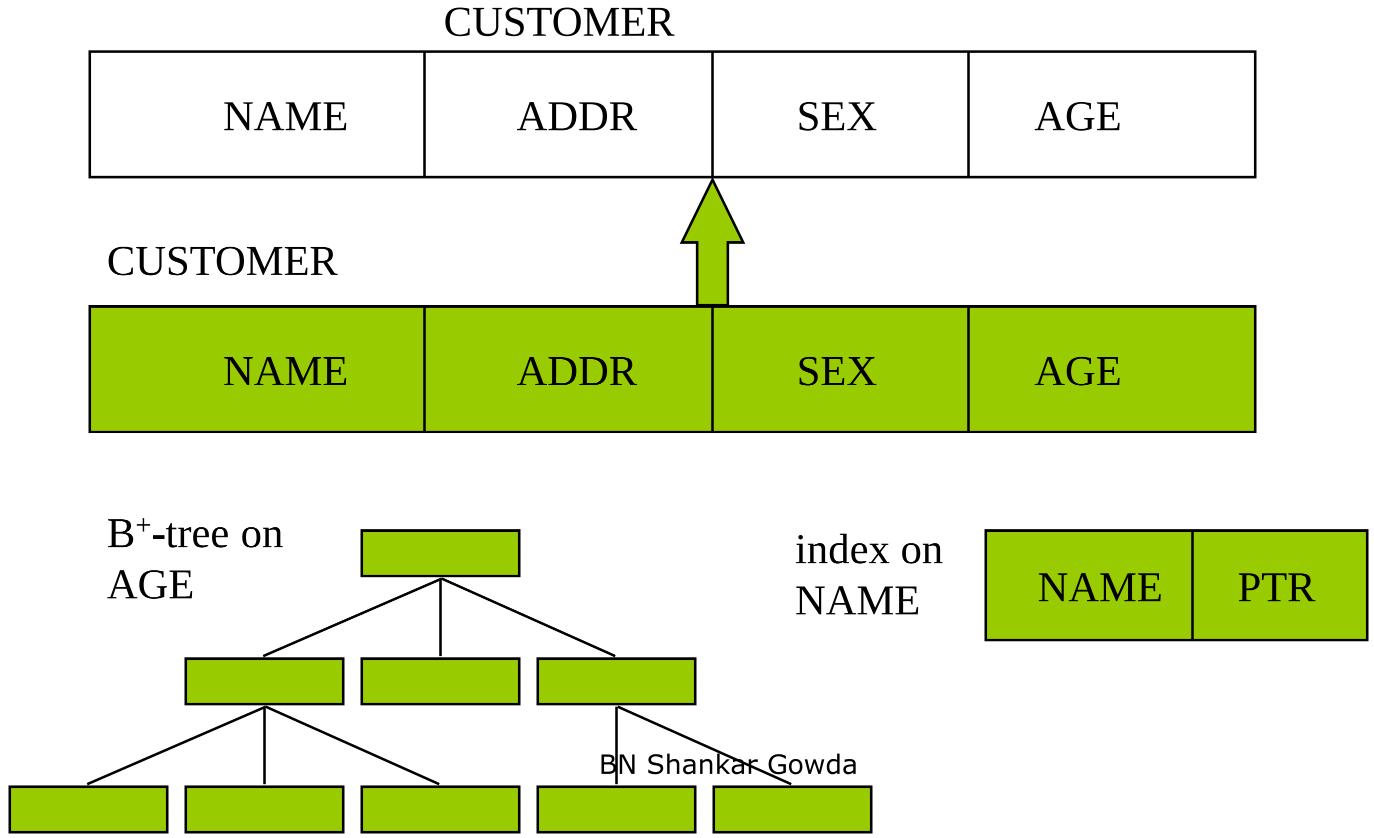
- **Describes the ACTUAL STORAGE details of the relations described in conceptual schema.**
- Primary indexing, sequential, binary, secondary indexing, etc.
- This leads to the physical database design.
- Eg: Select * from USER_CONSTRAINTS where TABLE_NAME = ' EMP ';

EMP: Index on Emp_ID owner Scott Date created 01/Jan/09……..

Emp_ID: string length 25 offset 0 primary key
SSN Integer size 10 offset 25 unique
Ename: Varchar size 20 offset 25 unique
Salary number 9,2 offset 100 not null default 1000
DOJ date dd-mm-yy offset 125 check

Physical / Internal Schema

- Describes how the information described in the conceptual schema is physically represented to provide the overall best performance



Conceptual Schema

- **Describes the STORED DATA in terms of the data model of the DBMS. This leads to <u>conceptual database design</u>.**
- Eg: DESC EMP;

EMP

Emp_ID String
Ename Varchar
SSN Integer
Salary number
DOJ date

OR

- Example:

Student(RegNo:Integer, Name:String,Sem:Integer, Branch:String)

Faculty(Fid:Integer, FName:String, Salary:Float)

Course(CourseNo:Integer, CName:String,Credit:Integere, Dept:String)

Conceptual Schema

- Describes all conceptually relevant, general, time-invariant structural aspects of the universe of discourse
- Excludes aspects of data representation and physical organization, and access
- DESC customer;

CUSTOMER

NAME	ADDR	SEX	AGE

- An object-oriented conceptual schema would also describe all process aspects

External Schema

- **Describes several VIEWS of the database based on the database model.**
- Several external schemas are possible for a single database.
- Each view is based upon the user requirements.

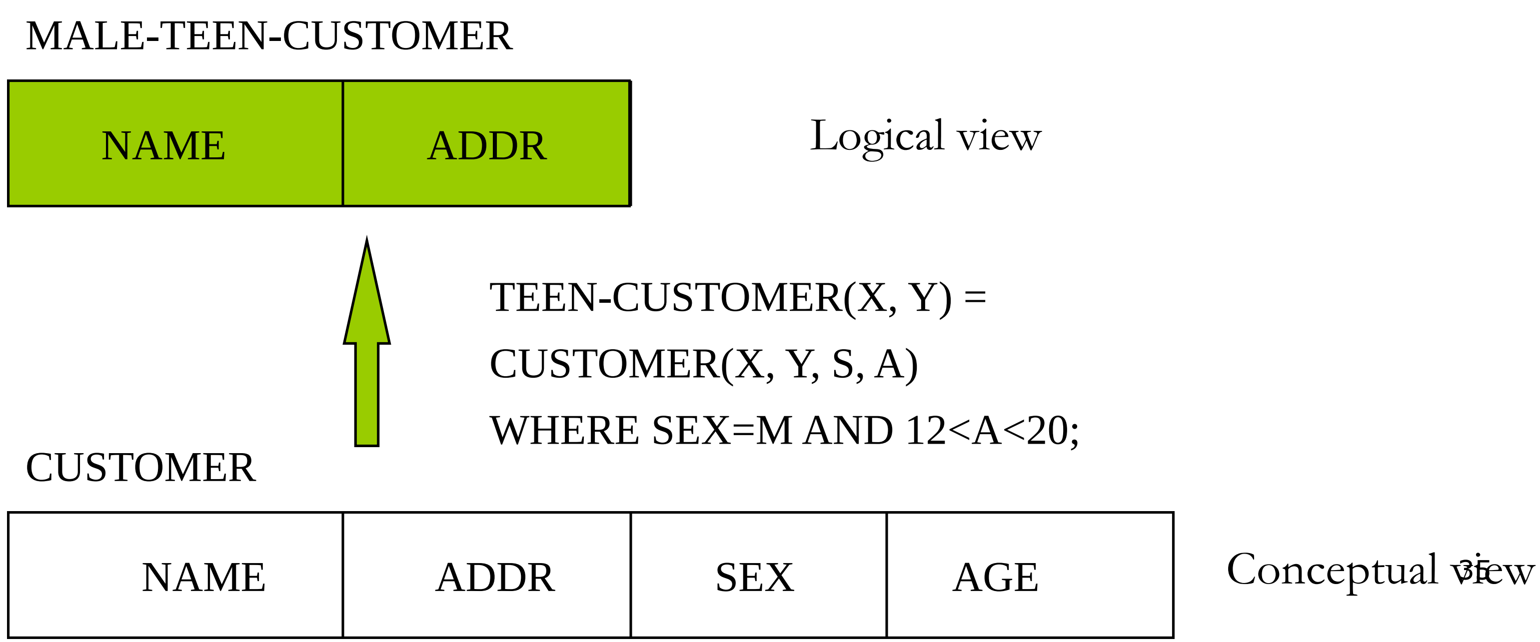
- Example:
- SELECT SSN, ENAME FROM EMP;

SSN	Name
111	john

- View-1

External Schema

- Describes parts of the information in the conceptual schema in a form convenient to a particular user group's view
- Is derived from the conceptual schema.
- SELECT NAME, ADDR from customer;



An example of the three levels

Customer_Loan

Cust_ID	: 101
Loan_No	: 1011
Amount_in_Dollars	: 8755.00

External

CREATE TABLE Customer_Loan (

Cust_ID	NUMBER(4)
Loan_No	NUMBER(4)
Amount_in_Dollars	NUMBER(7,2))

Conceptual

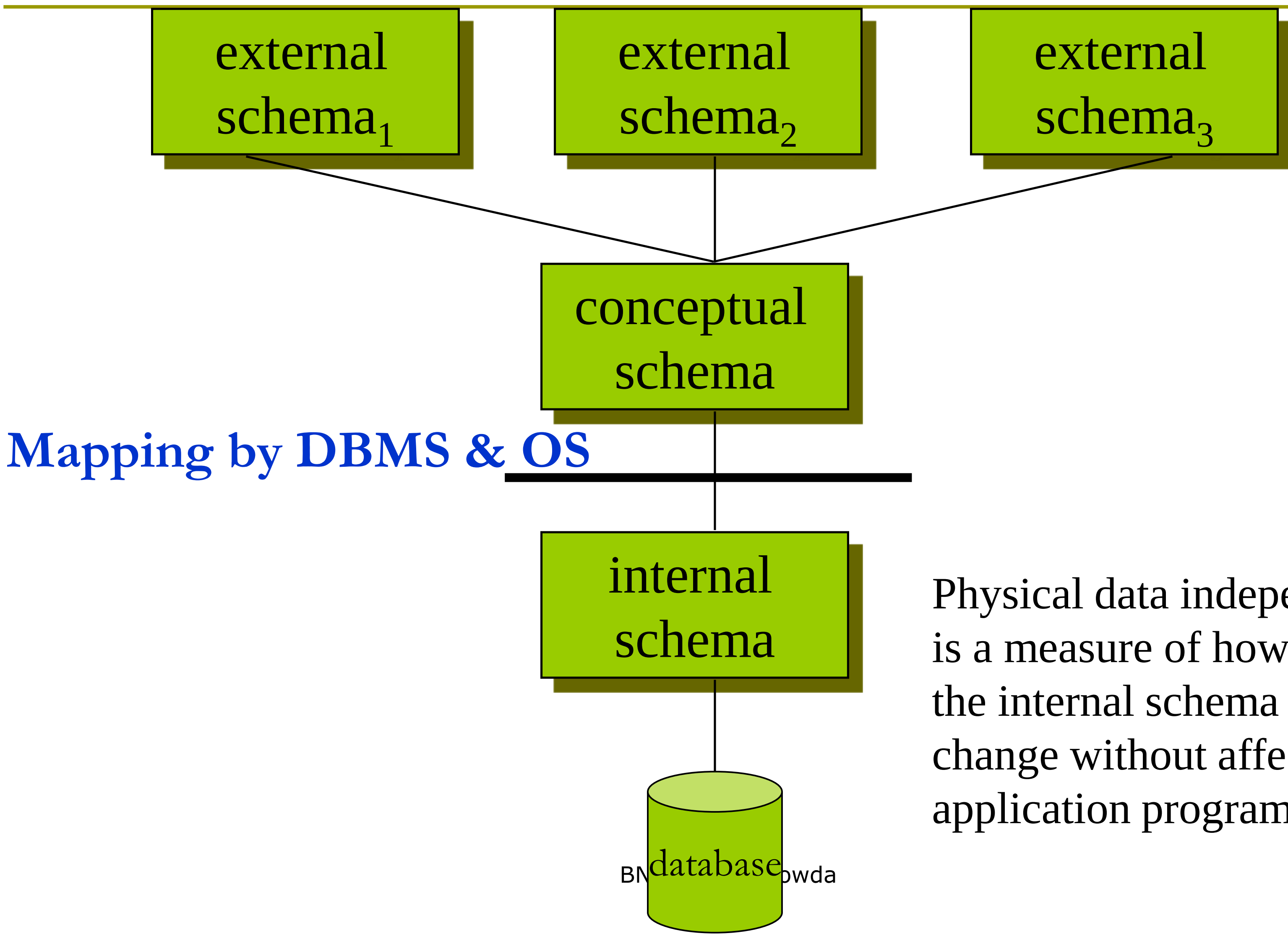
Cust_ID	TYPE = BYTE (4), OFFSET = 0
Loan_No	TYPE = BYTE (4), OFFSET = 4
Amount_in_Dollars	TYPE = BYTE (7), OFFSET = 8

Internal

Data Independence

- Change the schema at one level of a database system without a need to change the schema at the Next Higher Level
- Types of Data Independence
  - ***Logical Data Independence***: Refers to the immunity of the external schemas to changes in the conceptual schema e.g., add new record or field
  - ***Physical Data Independence***: Refers to the immunity of the conceptual schema to changes in the internal schema e.g., adding new index should not void existing ones

Physical Data Independence



Physical data independence is a measure of how much the internal schema can change without affecting the application programs

Physical data independence

There are occasions for changing the internal structures or storage structure for improved performance of the retrieval of data.

Any change introduced to the internal schema or physical schema will not affect the other higher level schemas.

Eg: Changing OS from windows to unix….

- Splitting of Db of 1GB to 500 MB of two

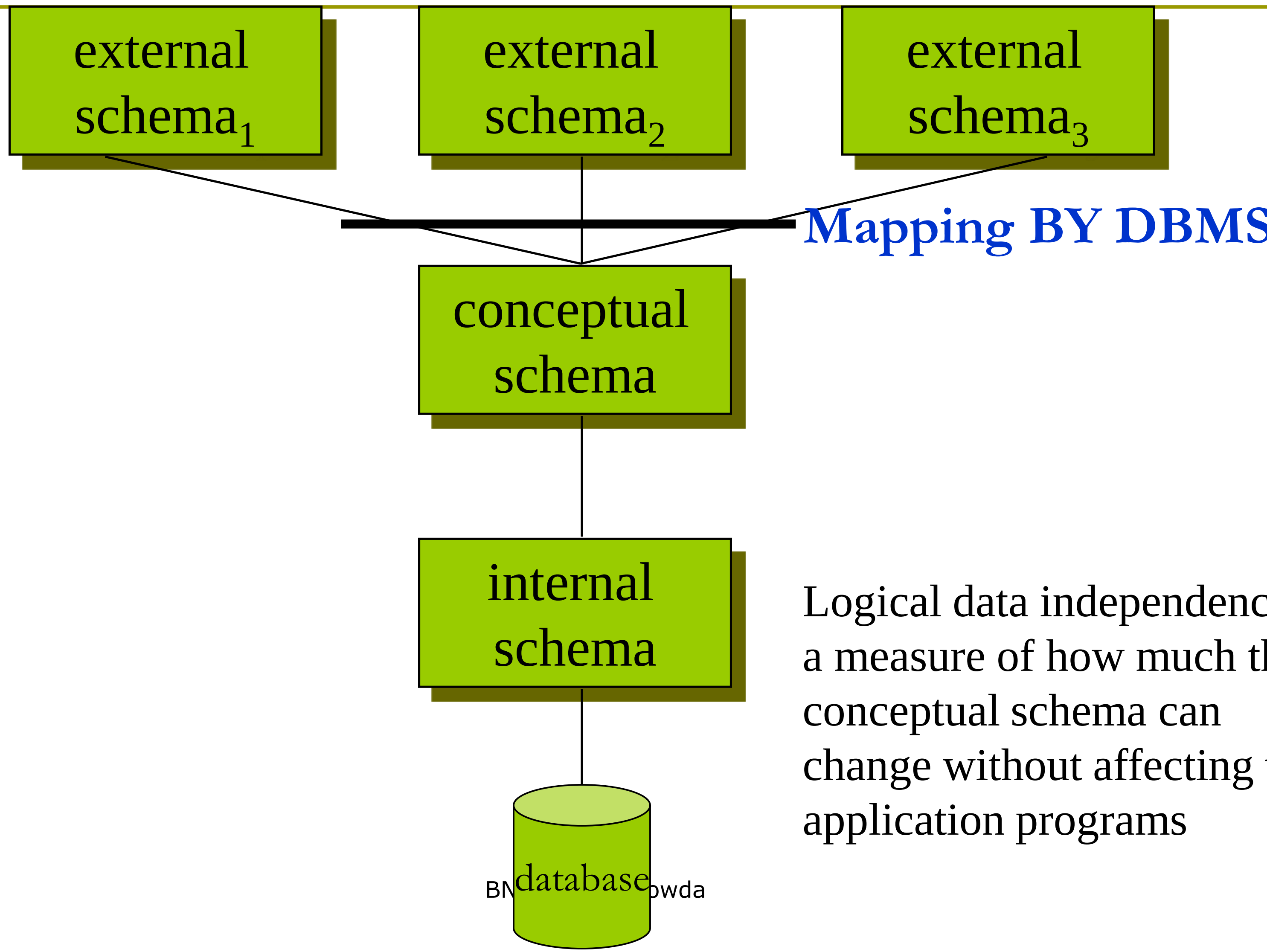
Representation of numeric values may be different; i,e

Physical : BINARY   Conceptual : Number

Representation of characters may be different

i,e Physical : EBCDIC   Conceptual : ASCII

Logical Data Independence



Logical data independence is a measure of how much the conceptual schema can change without affecting the application programs

Logical data independence

Suppose the Faculty relation is modified as:

- Faculty_Public(Fid:Int, FName:String, Office:Int)
- Faculty_Private(Fid:Integer, Salary:Float)

Any view designed before this modification can still retrieve the data with little modification (relation name) and obtain the same answer.

Eg:

1. Names of fields may be different;
2. Two/more fields may join & represent one field

Benefits of database approach

- Redundancy can be reduced but not eliminated
- Inconsistency can be avoided
- Data can be shared
- Standards can be enforced
- Security restrictions / No unauthorized access
- Integrity can be maintained
- Data independence can be provided
- Provides efficient Backup and recovery mechanisms

Benefits of database approach

- Supports Data abstraction
- Efficient Data access
- Simultaneous access by multiple users and applications
- Access to data without application programs (via a query language)
- Managing organizational data with uniform access and content controls
- Views
- Transaction processing (OLTP)

DBMS (Contd.)

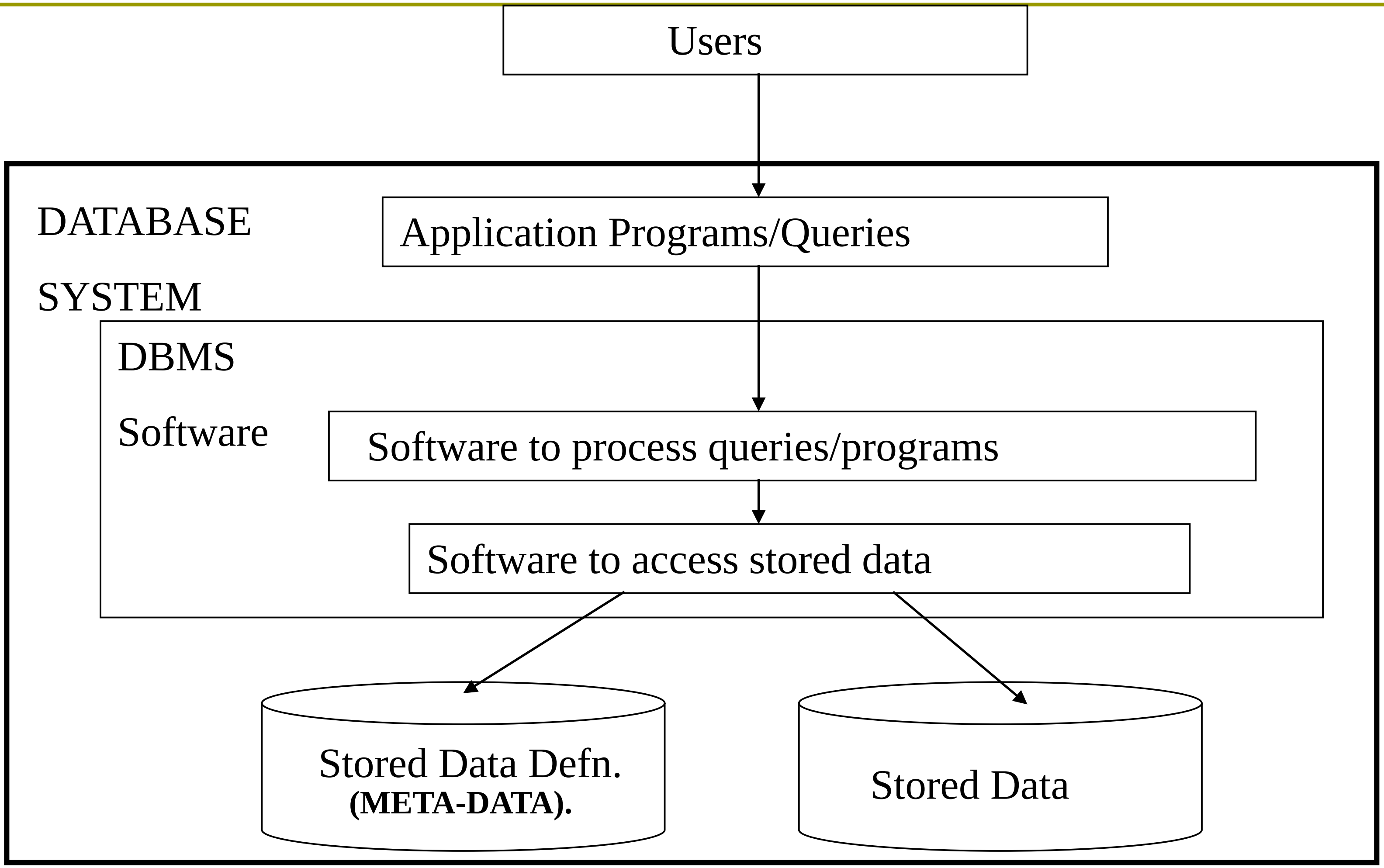
Goals of a Database Management System:

- **To provide an efficient as well as a convenient environment for accessing data in a database**
- **Enforce information security: database security, concurrency control, crash recovery**

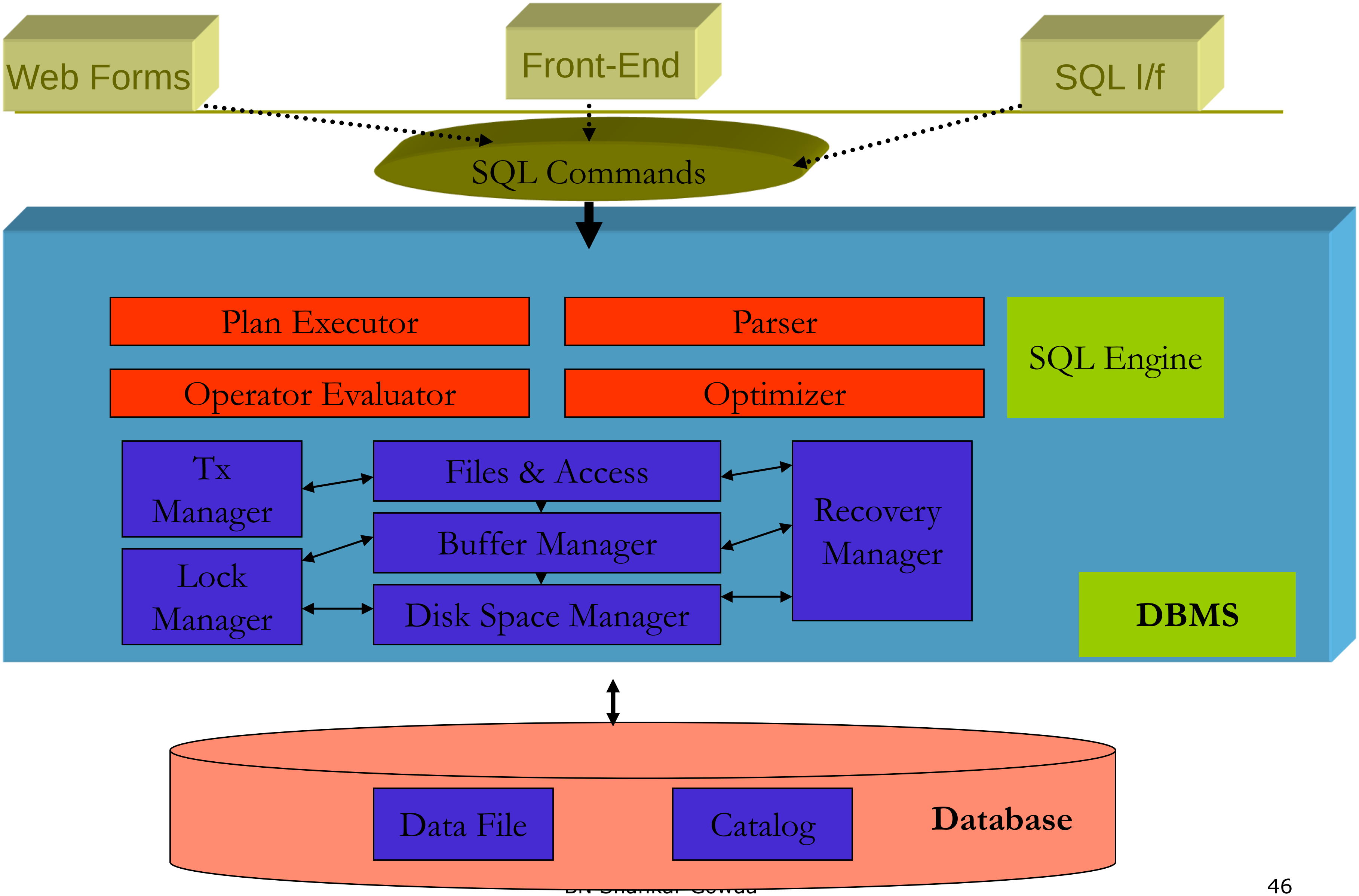
It is a general purpose facility for:

- **Defining** database
- **Constructing** database
- **Manipulating** database

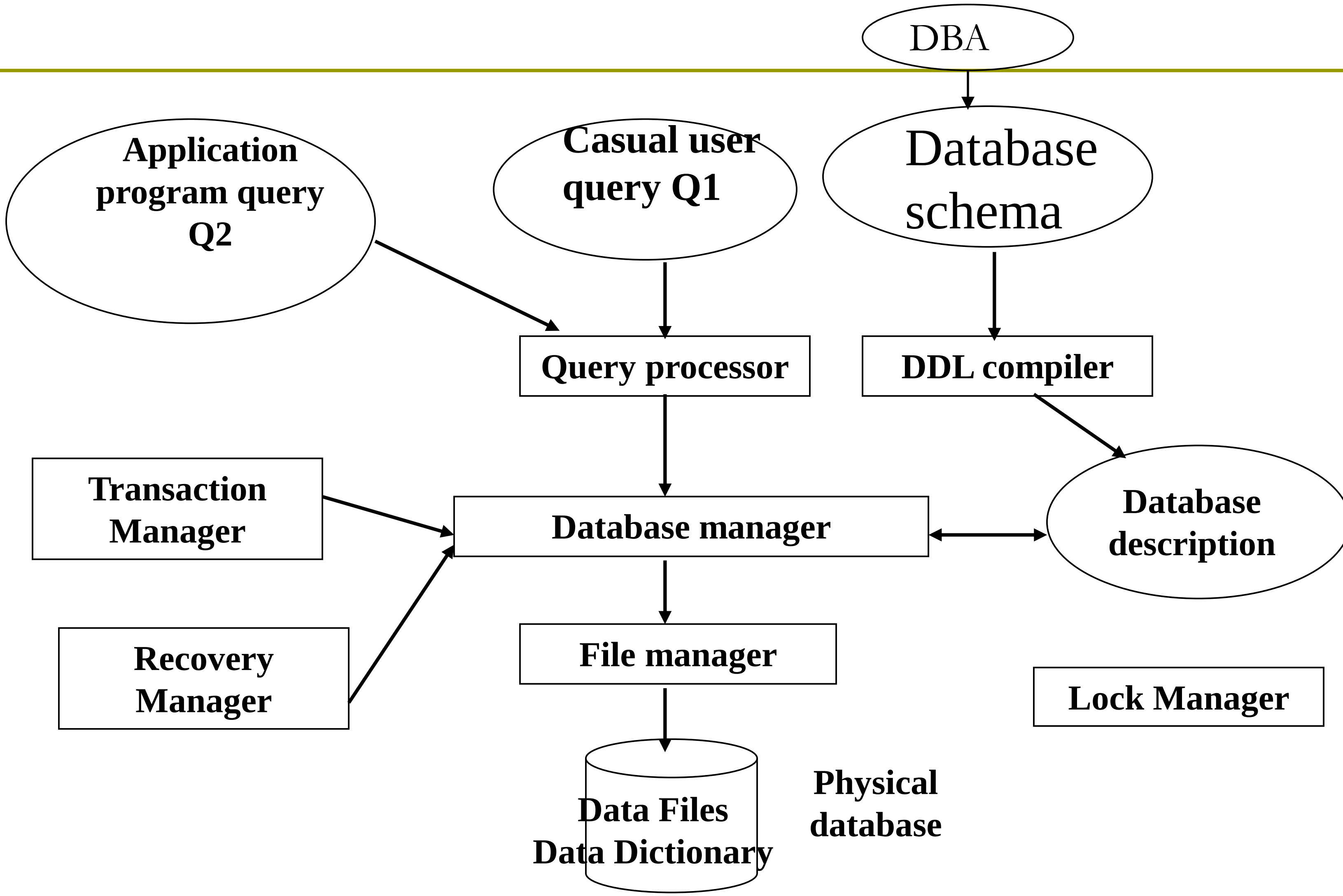
Database System



Database Structure



Structure of Database System



Structure of Database System

- **DDL Compiler:**

Converts DDL statements into set of Tables

- **Database Manager:**

Central S/w component of DBMS

1. Convert queries to Physical file system;
2. Enforce Constraints to maintain integrity & Security
3. Synchronize the simultaneous operations from concurrent users
4. Back up & Recovery mechanisms

Structure of Database System

- File Manager:

1. Structure of Files & Managing File space
2. Locate the block containing requested record
3. Send Request & receive Response from Disk manager

- Disk Manager:

1. Part of OS
2. Transfer Block of data to File manager

Structure of Database System

- **Query Processor:**

Interprets the user Query & convert into efficient series of operation that can be sent to data manager

- **Data Files:**

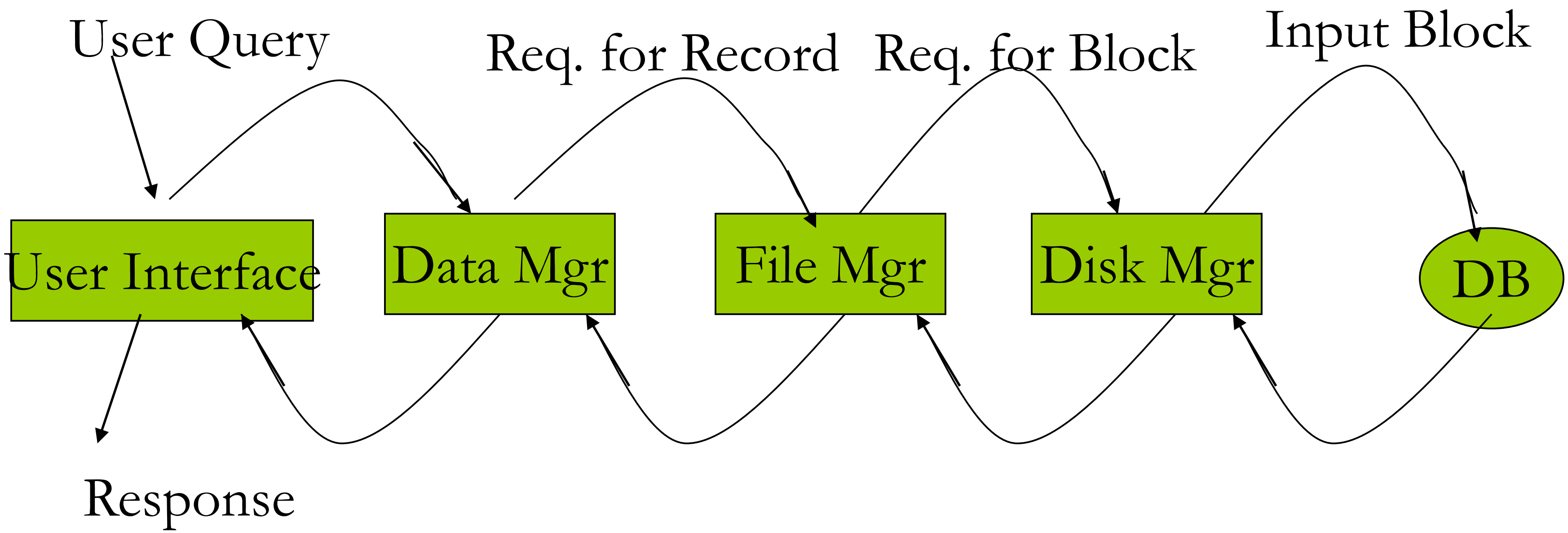
Data portion of DB

- **Data Dictionary:**

Information pertaining to the structure & usage

Describes Metadata:

Steps in Data Access



History of DBMS

- 1960 – First DBMS designed by Charles Bachman at GE. IBMs Information Management System (IMS)
- 1970 – Codd introduced the RDBMS
- 1980 – Relational model became popular and accepted as the main database paradigm. SQL, ANSI SQL, etc.
- 1980 to 1990 – New data models, powerful query languages, etc. Popular vendors are Oracle, SQL Server, IBMs DB2, Informix, etc.

Various types of data:

Images, Text, complex queries, Data Mining, etc.

- Enterprise Resource Planning (ERP)
- Management Resource Planning (MRP)
- Database in Web technologies

Current Database trends:

- Multimedia databases
- Interactive video
- Streaming data
- Digital Libraries

DBMS Functions

- Data Definition
- Data Manipulation
- Data Security and Integrity
- Data Recovery and Concurrency
- Data Dictionary
- Performance

Data Models

Definition of data model :

A tool used to describe :

- Data
- Data relationships
- Data semantics
- Consistency constraints

Types of data models

- Object based logical model
  - Entity relationship model
- Record based logical model
  - Hierarchical data model
  - Network data model
  - Relational data model

Hierarchical Model

- Data Structures
- Integrity Constraints
- Operations

- Commercial systems include IBM's IMS, MRI's System-2000 (now sold by SAS), and CDC's MARS IV

Hierarchical Database Structures

- The hierarchical model employs two main data structuring concepts: **records and parent-child relationships**.
- A **record** is a collection of **field values** that provide information on an entity or a relationship instance. Records of the same type are grouped into **record types.**
- A record type is given a name, and its structure is defined by a collection of named **fields** or **data items.** Each field has a certain data type, such as integer, real, or string.
- A **parent-child relationship type (PCR type)** is a 1:N relationship between two record types. The record type on the 1-side is called the **parent record type,** and the one on the N-side is called the **child record type** of the PCR type. An **occurrence** (or **instance**) of the PCR type consists of *one record* of the parent record type and a number of records (zero or more) of the child record type.

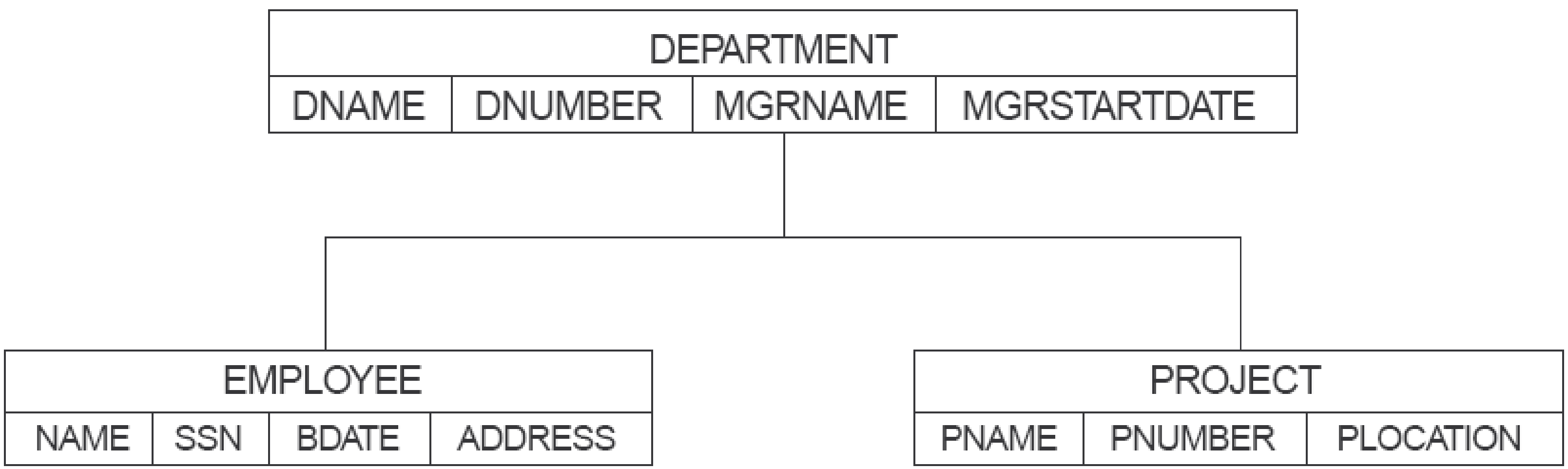
Properties of a Hierarchical Schema

- One record type, called the **root** of the hierarchical schema, does not participate as a child record type in any PCR type.
- Every record type except the root participates as a child record type in *exactly one* PCR type and A record type can participate as parent record type in any number (zero/more) of PCR types
- A record type that does not participate as parent record type in any PCR type is called a **leaf** of the hierarchical schema.
- If a record type participates as parent in more than one PCR type, then *its child record types are ordered.* The order is displayed, by convention, from left to right in a hierarchical diagram.

Hierarchical diagram

- A hierarchical schema is displayed as a **hierarchical diagram,** in which record type names are displayed in rectangular boxes and PCR types are displayed as lines connecting the parent record type to the child record type. Figure shows a simple hierarchical diagram for a hierarchical schema with three record types and two PCR types. The record types are DEPARTMENT , EMPLOYEE , and PROJECT Field names can be displayed under each record type name, as shown in Figure. In some diagrams, for brevity, we display only the record type names.

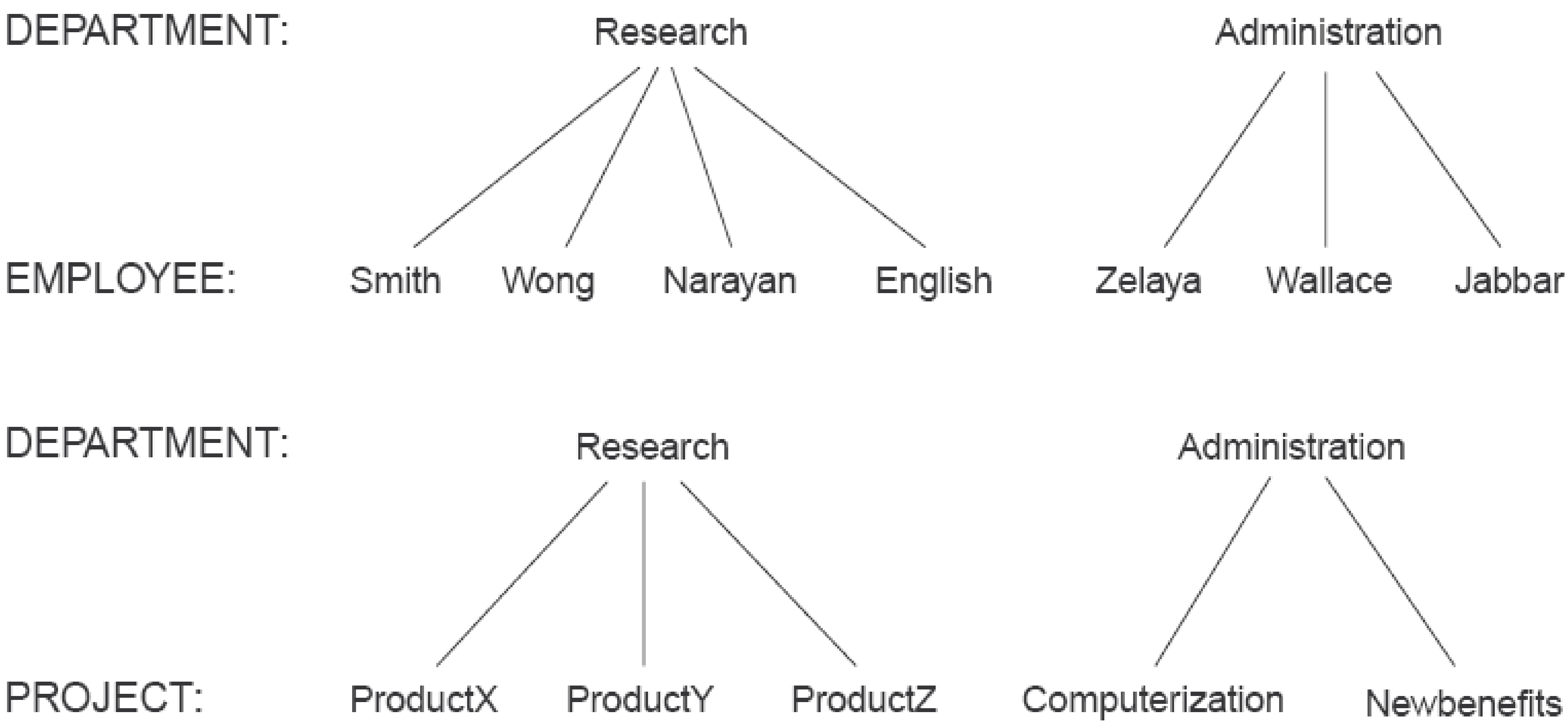
A Hierarchical schema



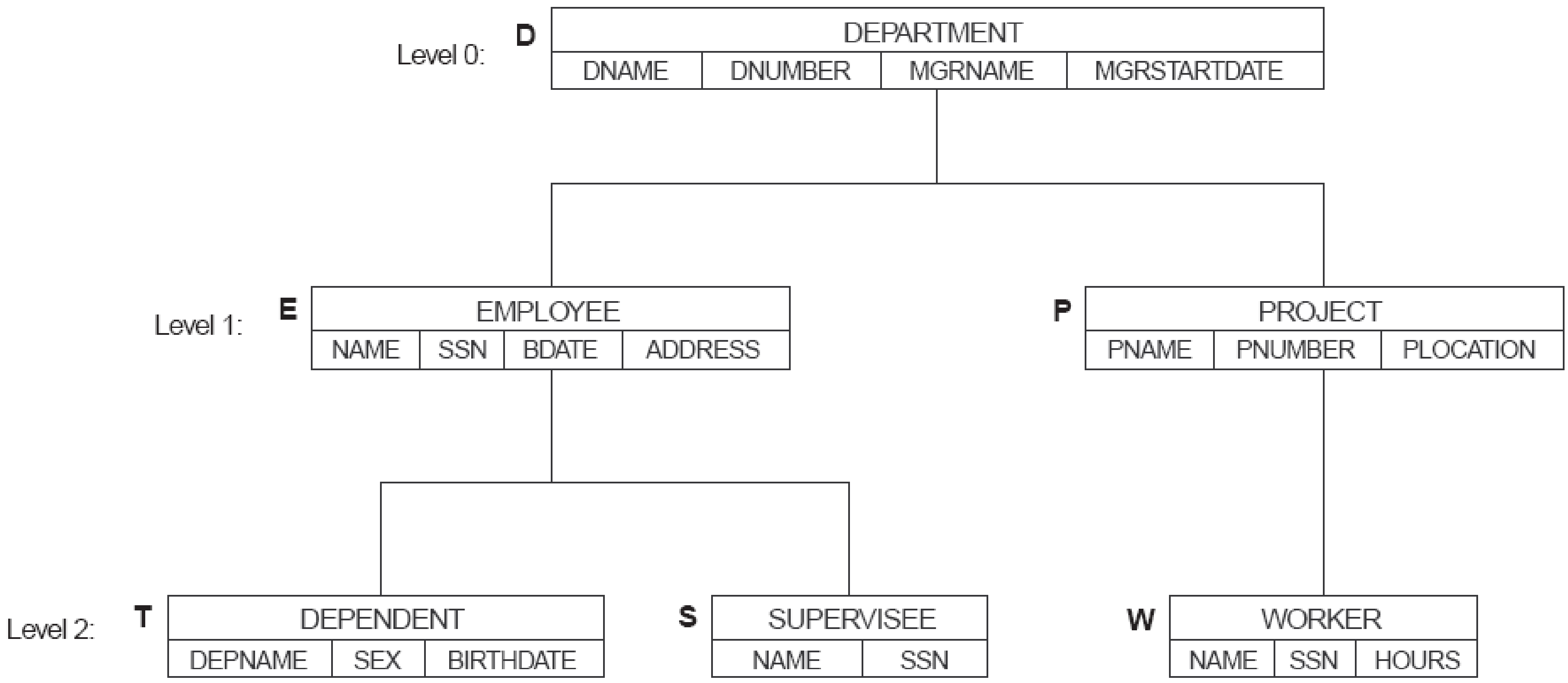
A Hierarchical schema

- The PCR type in a hierarchical schema by listed by the pair (parent record type, child record type) between parentheses. The two PCR types in Figure are (DEPARTMENT , EMPLOYEE) and (DEPARTMENT, PROJECT). Notice that PCR types *do not* have a name in the hierarchical model. In Figure each *occurrence* of the (DEPARTMENT , EMPLOYEE) PCR type relates one department record to the records of the *many* (zero or more) employees who work in that department. An *occurrence* of the (DEPARTMENT , PROJECT) PCR type relates a department record to the records of projects controlled by that department. Figure shows two PCR occurrences (or instances) for each of these two PCR types.

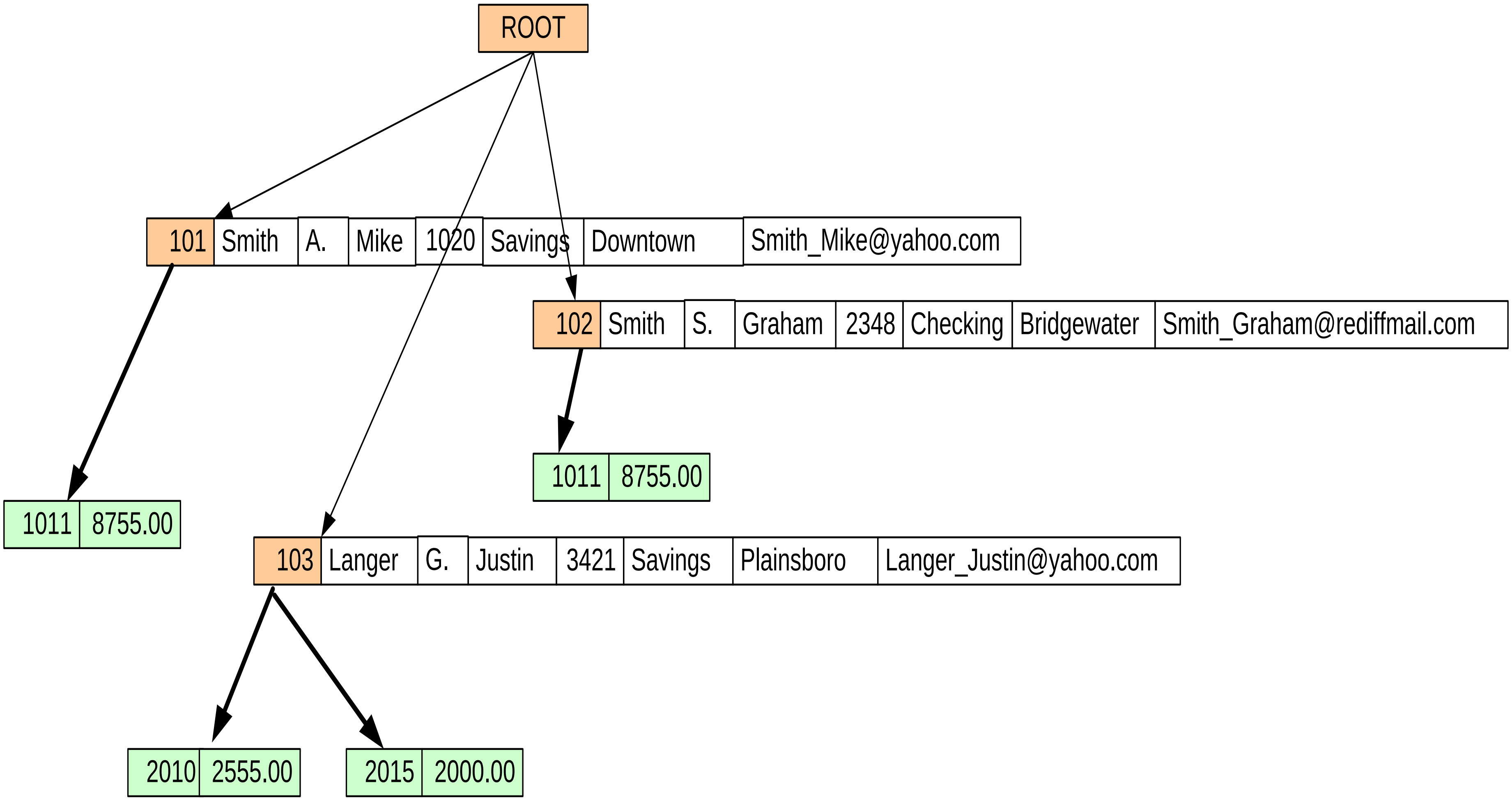
Occurrences of Parent-Child Relationships



A Hierarchical schema for part of the COMPANY database



Record based data model – Hierarchical data model



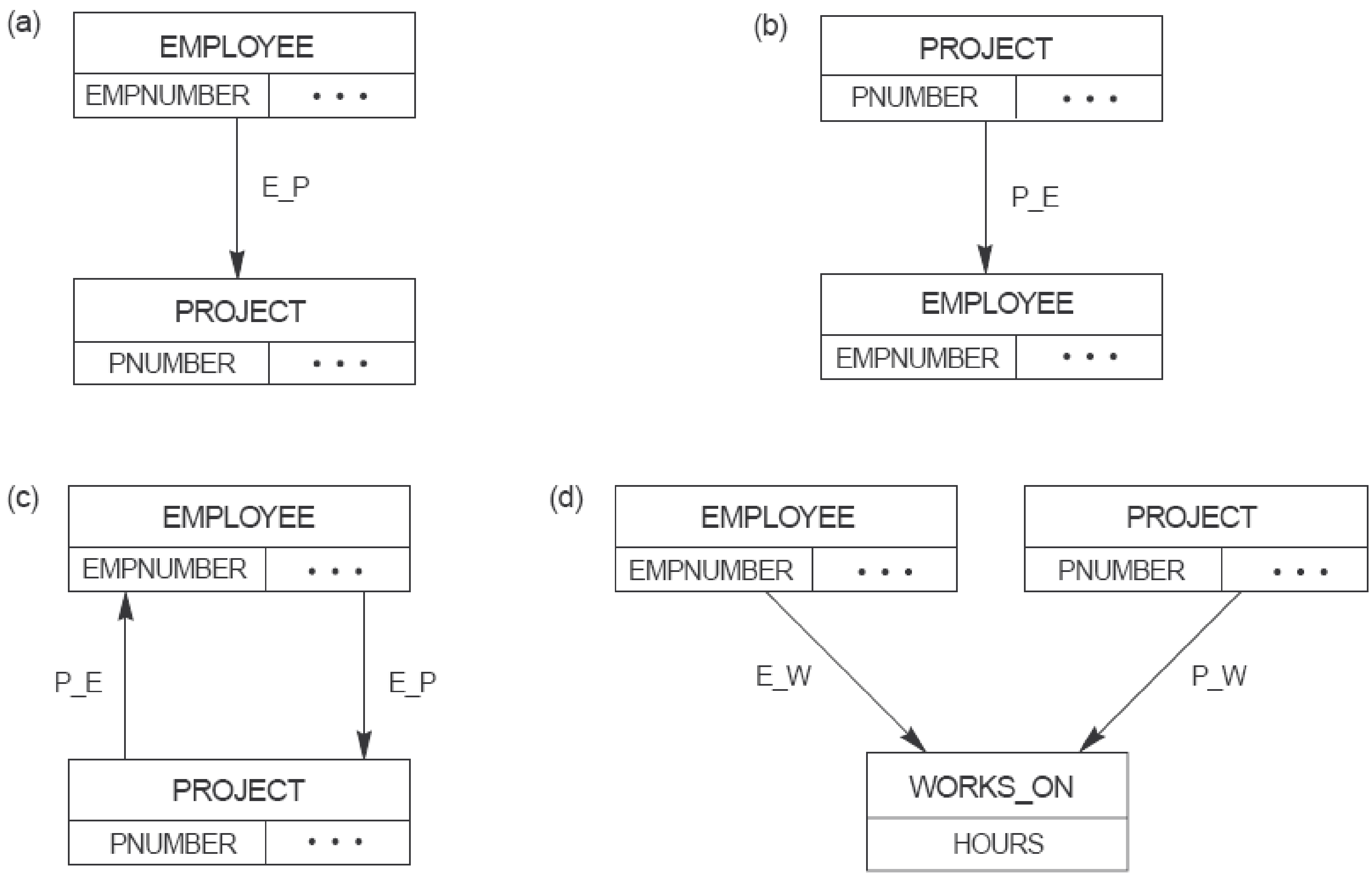
E.g.: Information Management System (IMS) from IBM

Network Model

- Data Structures
- Integrity Constraints
- Operations

- Based on the CODASYL-DBTG 1971 report
- Commercial systems include, CA-IDMS and DMS-1100

Employee database



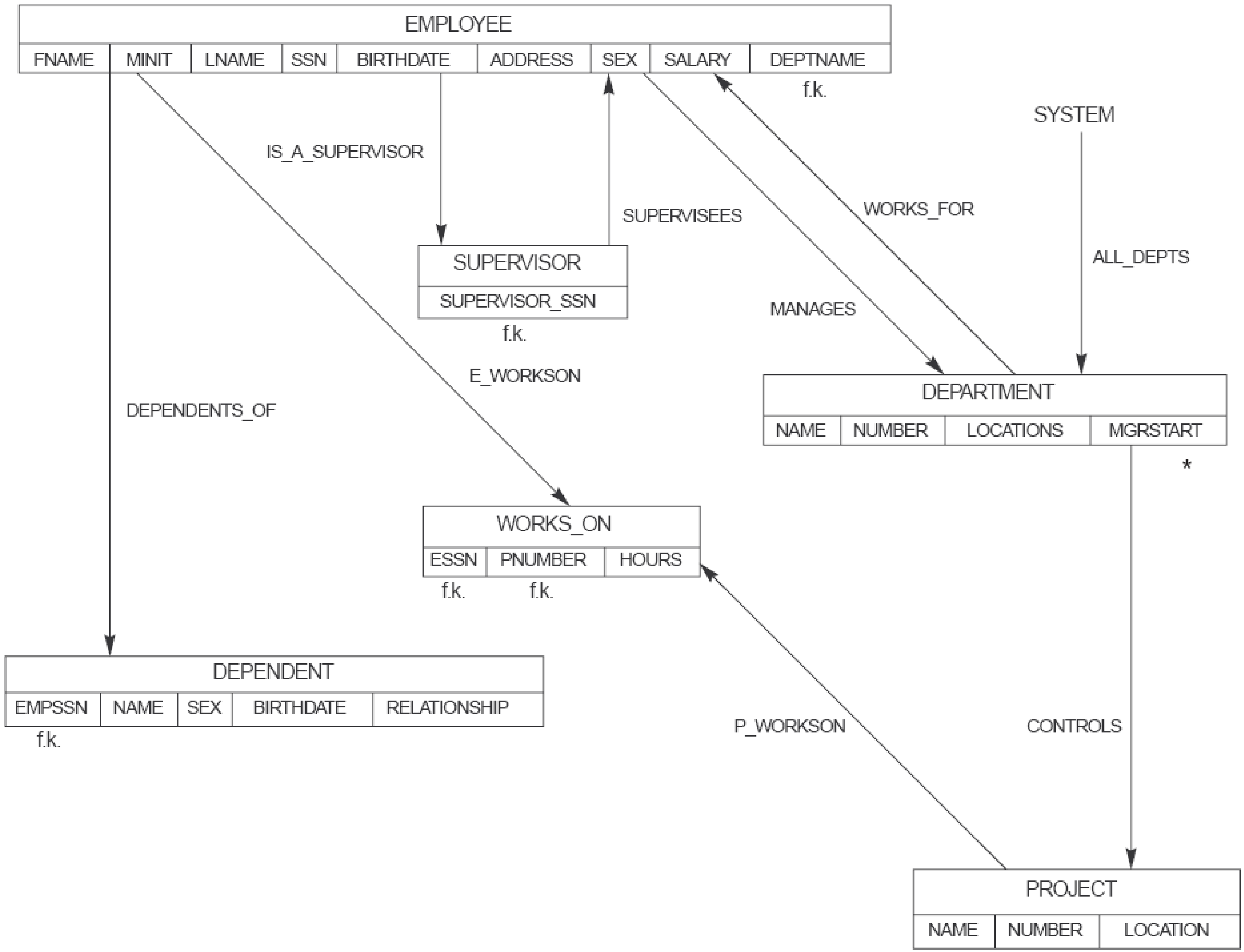
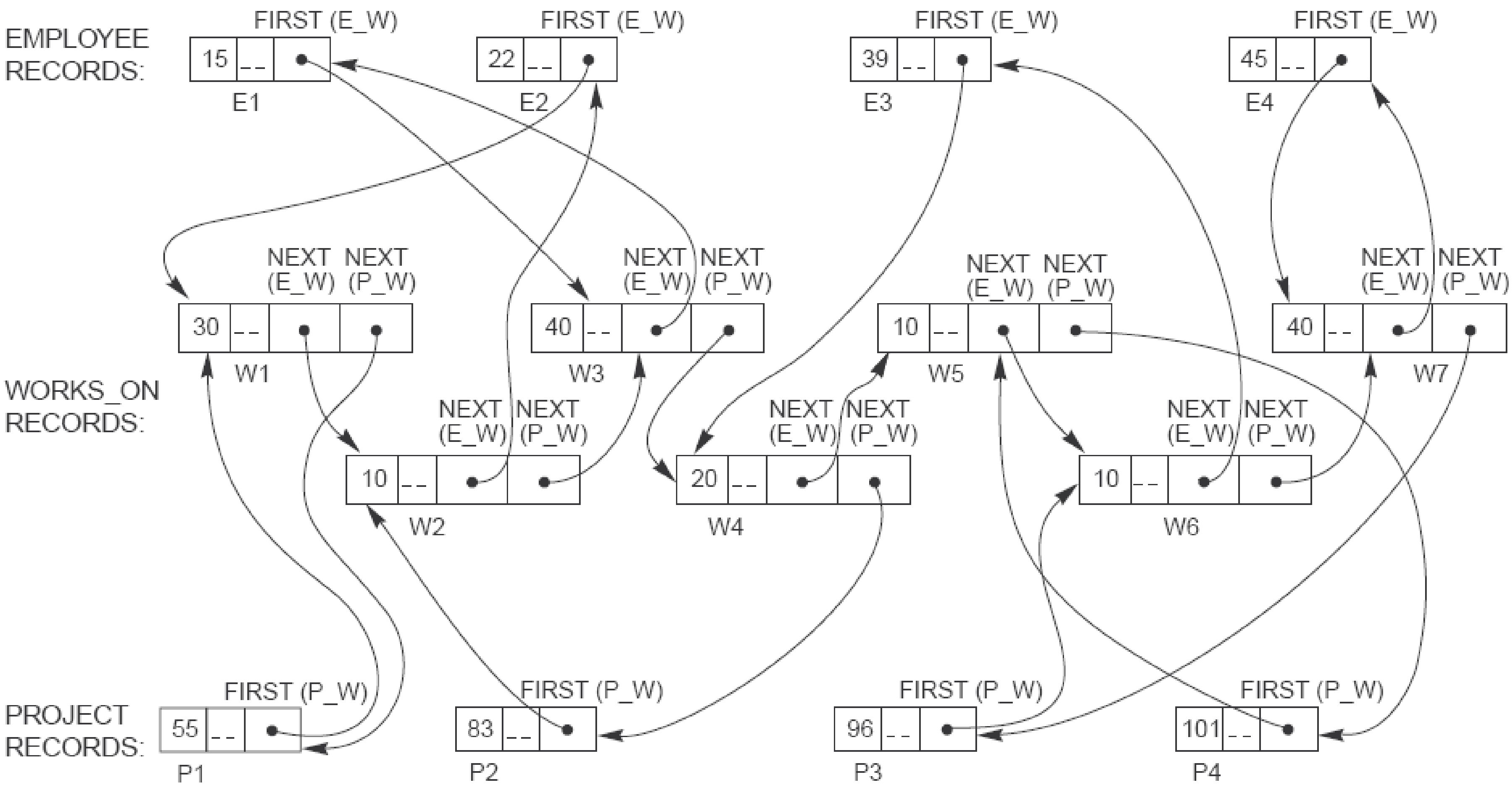
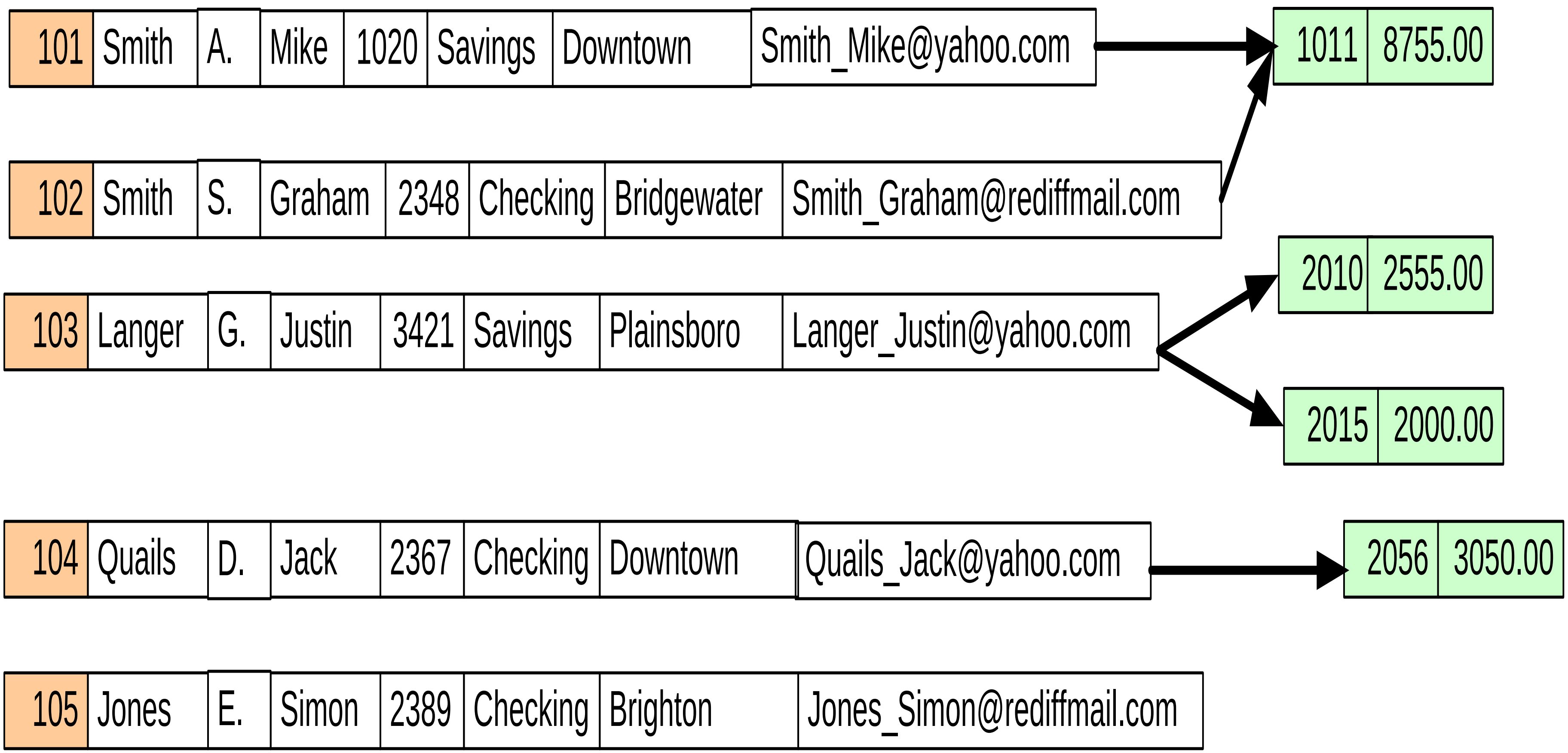


Figure E.8 A network schema diagram for the COMPANY database.

Network Model (linked list)



Record based data model – Network data model



 E.g.: Integrated Data Management System(IDMS) from Honeywell

Record based data model – Network data model

- Data in the network model is represented by a collection of records
- Relationships among data are represented by links (Pointers)
- The records in the database are collection of graphs
- E.g.: Integrated Data Management System(IDMS) from Honeywell

Relational Model

- Data Structures
- Integrity Constraints
- Operations

- Commercial systems include: ORACLE, DB2, SYBASE, INFORMIX, INGRES, SQL Server
- Dominates the database market on all platforms

Relational Database Definition

- A relational database is a collection of relations or two-dimensional tables.

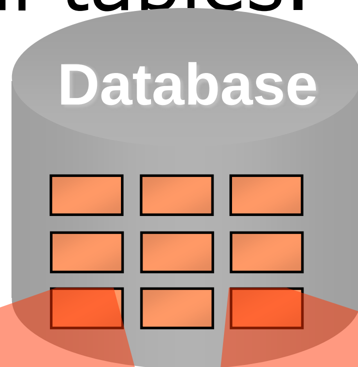


Table Name: **EMP**

EMPNO	ENAME	JOB	DEPTNO
7839	KING	PRESIDENT	10
7698	BLAKE	MANAGER	30
7782	CLARK	MANAGER	10
7566	JONES	MANAGER	20

Table Name: **DEPT**

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Relational model basics

- Data is viewed as existing in two dimensional tables known as relations
- A relation (table) consists of unique attributes (columns) and tuples (rows)
- Tuples are unique
- Sometimes the value to be inserted into a particular cell may be unknown, or it may have no value. This is represented by a **NULL**
- Null is not the same as zero, blank or an empty string
- Relational Database: Any database whose logical organization is based on relational data model.
- RDBMS: A DBMS that manages the relational database.

Record based data model – Relational data model

Attributes / Columns / Fields

Rows / Records / Tuples

Customer_Loan records from Customer_Loan table

Cust_ID	Loan_No	Amount_in_Dollars
101	1011	8755.00
103	2010	2555.00
104	2056	3050.00
103	2015	2000.00

No. of Records / Rows / Tuples: Cardinality of the Relation

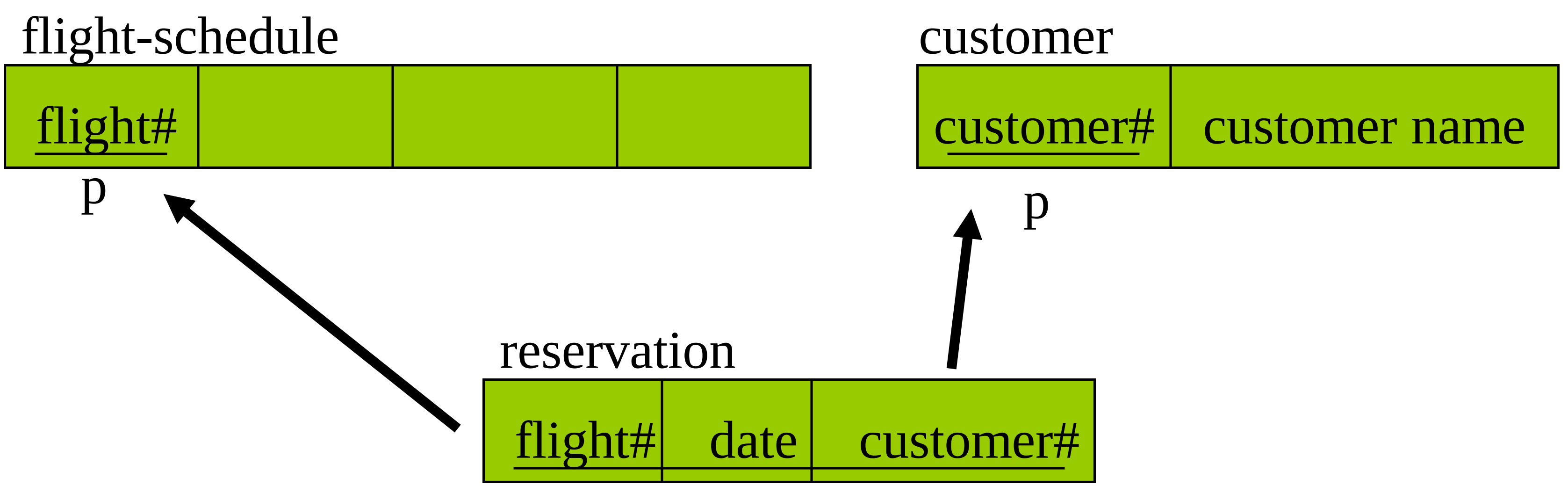
No. of Attributes / Columns / Fields : Degree of the Relation

Cust_ID	Cust_Last_Name	Cust_Mid_Name	Cust_First_Name	Account_No	Account_Type	Bank_Branch	Cust_Email
101	Smith	A.	Mike	1020	Savings	Downtown	Smith_Mike@yahoo.com
102	Smith	S.	Graham	2348	Checking	Bridgewater	Smith_Graham@rediffmail.com
103	Langer	G.	Justin	3421	Savings	Plainsboro	Langer_Justin@yahoo.com
104	Quails	D.	Jack	2367	Checking	Downtown	Quails_Jack@yahoo.com
105	Jones	E.	Simon	2389	Checking	Brighton	Jones_Simon@rediffmail.com

records from Customer_Details table

Relational Model – Integrity Constraints

- Keys
- Primary Keys
- Entity Integrity
- Referential Integrity



Classifications of constraints

Primary constraints	Secondary Constraints
Candidate Key	Not Null
Primary Key	Default
Super/Composite Key	
Alternate key	
Foreign Key	
Unique Key	
Check Constraint	

Keys in Relational model

- **Candidate key**

A Candidate key is a set of **one or more attributes(minimal)** that can uniquely identify a row in a given table.

- **Primary Key**

During the creation of the table, the Database Designer chooses one of the Candidate Key from amongst the several available, to uniquely identify row in the given table.

- **Alternate Key**

The candidate key that is chosen to perform the identification task is called the *primary key* and the remaining candidate keys are known as alternate keys.

No of Alternate Keys = No of Candidate Keys - 1

Key and Non-key Attributes in Relational Model

- **Key Attributes**

  The attributes that participate in the Candidate key are Key Attributes

- **Non-Key Attributes**
  - The attributes other than the Candidate Key attributes in a table/relation are called Non-Key attributes.

  OR

  - The attributes which do not participate in the Candidate key.

Example

Given a relation

Trainee(Empno, FirstName, LastName, Email, PhoneNo)

Assumptions:

i. **Empno for each trainee is different.**
ii. **Email for each trainee is different**
iii. **PhoneNo for each trainee is different**
iv. **Combination of FirstName and LastName for each trainee is different**

Candidate key:

{Empno},{Email},{PhoneNo},{FirstName,LastName}

Primary key:

{Empno}

Alternate Key:

{Email},{PhoneNo},{FirstName,LastName}

Exercise on Key attributes

Given a relation R1(X,Y,Z,L) and the following attribute(s) can uniquely identify the records of relation R1.

1)X

2)X,L

3)Z,L

Identify the following in relation R1?

Candidate Key(s)

Primary Key

Alternate Key

Key attribute(s)

Non-key attribute(s)

What are the candidate keys?

Case 1

Assumptions

One customer can have only one account

An account can belong to only one customer

while deciding the Candidate key do not get misguided by the data present in the table.

Cust_ID	Cust_Last_Name	Cust_Mid_Name	Cust_First_Name	Account_No	Account_Type	Bank_Branch	Cust_Email
101	Smith	A.	Mike	1020	Savings	Downtown	Smith_Mike@yahoo.com
102	Smith	S.	Graham	2348	Checking	Bridgewater	Smith_Graham@rediffmail.com
103	Langer	G.	Justin	3421	Savings	Plainsboro	Langer_Justin@yahoo.com
104	Quails	D.	Jack	2367	Checking	Downtown	Quails_Jack@yahoo.com
105	Jones	E.	Simon	2389	Checking	Brighton	Jones_Simon@rediffmail.com

Customer_Detail records from Customer_Details table

What are the candidate keys?

Case 2

Assumptions

One customer can have many accounts

An account can belong to only one customer

Cust_ID	Cust_Last_Name	Cust_Mid_Name	Cust_First_Name	Account_No	Account_Type	Bank_Branch	Cust_Email
101	Smith	A.	Mike	1020	Savings	Downtown	Smith_Mike@yahoo.com
102	Smith	S.	Graham	2348	Checking	Bridgewater	Smith_Graham@rediffmail.com
103	Langer	G.	Justin	3421	Savings	Plainsboro	Langer_Justin@yahoo.com
104	Quails	D.	Jack	2367	Checking	Downtown	Quails_Jack@yahoo.com
105	Jones	E.	Simon	2389	Checking	Brighton	Jones_Simon@rediffmail.com

Customer_Detail records from Customer_Details table

What are the candidate keys?

Case 3 :
Assumptions

One customer can have many accounts.
An account can belong to more than one customer (joint account)

Cust_ID	Cust_Last_Name	Cust_Mid_Name	Cust_First_Name	Account_No	Account_Type	Bank_Branch	Cust_Email
101	Smith	A.	Mike	1020	Savings	Downtown	Smith_Mike@yahoo.com
102	Smith	S.	Graham	2348	Checking	Bridgewater	Smith_Graham@rediffmail.com
103	Langer	G.	Justin	3421	Savings	Plainsboro	Langer_Justin@yahoo.com
104	Quails	D.	Jack	2367	Checking	Downtown	Quails_Jack@yahoo.com
105	Jones	E.	Simon	2389	Checking	Brighton	Jones_Simon@rediffmail.com

Customer_Detail records from Customer_Details table

Choosing a Primary key from Candidate keys -Guidelines

- Give preference to numeric column(s)
- Give preference to single attribute
- Give preference to minimal composite key

Primary Key of the table, Customer_Details

Cust_ID	Cust_Last_ Name	Cust_Mid _Name	Cust_First _Name	Account _No	Account_ Type	Bank_Branch	Cust_Email
101	Smith	A.	Mike	1020	Savings	Downtown	Smith_Mike@yahoo.com
102	Smith	S.	Graham	2348	Checking	Bridgewater	Smith_Graham@rediffmail.com
103	Langer	G.	Justin	3421	Savings	Plainsboro	Langer_Justin@yahoo.com
104	Quails	D.	Jack	2367	Checking	Downtown	Quails_Jack@yahoo.com
105	Jones	E.	Simon	2389	Checking	Brighton	Jones_Simon@rediffmail.com

Customer_Detail records from Customer_Details table

Foreign Key

A Foreign Key is a set of attribute (s) whose values are required to match values of a column in the same or another table.

DEPT

(Parent /Master/Referenced Table)

DeptNo	DName
D1	IVS
D2	ENR

EMP

(Child /Referencing Table)

EmpNo	EName	EDeptNo
1001	Elsa	**D1**
1002	John	**D2**
1003	Maria	**D1**
1004	Maida	**D1**

- Points to remember
  - Foreign key values do not (usually) have to be unique.
  - Foreign keys can also be *null* .
  - To enter the data in child table corresponding data must be present in master table or NULL is the default entry in child table in the referenced column (FK column)

Relating Multiple Tables

- EACH AND EVERY ROW IS IDENTIFIED BY A UNIQUE KEY CALLED PK
- LOGICAL RELATED ROWS SHARED the PKEY as FKEY.

EMP

EMPNO	ENAME	JOB	DEPTNO
7839	KING	PRESIDENT	10
7698	BLAKE	MANAGER	30
7782	CLARK	MANAGER	10
7566	JONES	MANAGER	20

DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Primary key **Foreign key** **Primary key**

Foreign Key

- Composite Foreign key example →

Demos

- Points to remember
  - A Foreign Key is a set of attributes of a table, whose values are required to match values of some Candidate Key in the same or another table
  - The constraint that values of a given Foreign Key must match the values of the corresponding Candidate Key is known as Referential constraint
  - A table which has a Foreign Key referring to its own Candidate Key is known as Self-Referencing table

Relational Model - Operations

- Powerful set-oriented query languages
- Relational Algebra: procedural; describes how to compute a query; operators like JOIN, SELECT, PROJECT
- Relational Calculus: declarative; describes the desired result, e.g. SQL, QBE
- insert, delete, and update capabilities

Object-Oriented Model(s)

- based on the object-oriented paradigm,
  e.g., Simula, Smalltalk, C++, Java

- **object-oriented model** has object-oriented repository model; adds persistence and database capabilities; (see ODMG-93, ODL, OQL)
- **object-oriented** commercial systems include GemStone, Ontos, Orion-2, Statice, Versant, O_2

- **object-relational model** has relational repository model; adds object-oriented features; (see SQL3)
- **object-relational** commercial systems include Starburst, POSTGRES

Database Design Phases

DATA ANALYSIS

Entities - Attributes - Relationships - Integrity Rules

LOGICAL DESIGN

Tables - Columns - Primary Keys - Foreign Keys

PHYSICAL DESIGN

DDL for Tablespaces, Tables, Indexes

Database Languages

- DDL – Data Definition Language
- DML – Data Manipulation Language
- DCL – Data Control Language
- TCL – Transaction Control Language
- SCC– System Control Commands
- SCC– Session Control Commands

People Who Work with Databases

- Database Implementers
- End Users
- Application Programmers
- DBA

Database designers

- Design the database elements i,e
- Responsible for identifying
- The data to be stored in the database
- Choosing appropriate datatype, constraints,size…
- Interact with clients for requirement and come up with suitable design

End Users

- Casual users

  These are people who use the database occasionally.

- Naive users

  These are users who constantly querying and updating the database.

  Eg. Reservation Clerks of Airline, Railway, Hotel, etc.

  Clerks at receiving station of Courier service, Insurance agencies, etc.

- Sophisticated Users

  People who use for their complex requirements.

  Eg. Engineers, Scientists, Business analysts…

- Standalone Users

  Who maintain database for personal use.

DBA

- Managing resources
- Creation of user accounts
- Providing security and authorization
- Managing poor system response time
- System Recovery
- Tuning the Database

Disadvantages of DBMS

- Cost
- Complexity of Backup and Recovery of data
- Problems associated with centralized control

When not to use DBMS?

- For small applications
- Concurrent access of data not required

Summary

- An Overview of Database Management
- Database
- DBMS
- Database Systems
- Why Use Database
- Database Architecture
- An Example of the Three Levels
- Schema
- Data Independence
- Types Of Database Models
- Database Design Phases

End of Chapter - 1