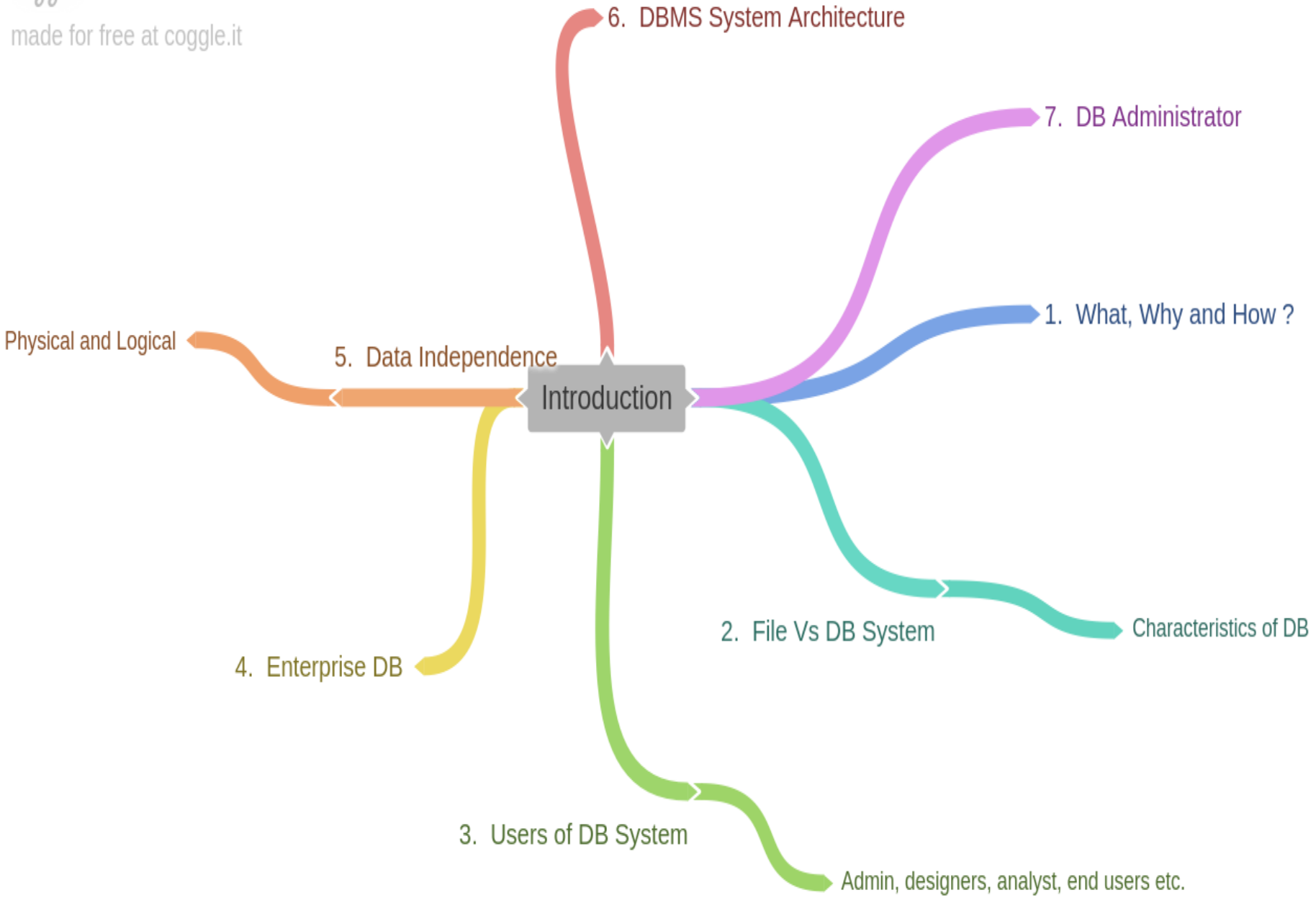
Introduction to Database Management System

-By
Jyoti Tryambake



Basic Definitions

Data:

- Known facts that can be recorded and have an implicit meaning.

Database:

- A well organized collection of data that are related in a meaningful way which can be accessed in different logical orders but are stored only once. The data in the database is therefore integrated, structured and shared.

Basic Definitions (cont.)

Database Management System (DBMS):

- Collection of interrelated data
- Set of programs to access the data
- An environment that is both *convenient* and *efficient* to use

- Def: A software package/ system to facilitate the creation and maintenance of a computerized database.

Database System:

- The DBMS software together with the data itself. Sometimes, the applications are also included.
- Database System: DBMS + Data (+ applications)

Database Applications

- Enterprise Information
  - Sales
    - For customer, product and purchase information
  - Accounting
    - For payments, receipts, account balances, assets
  - Human Resources
    - For employee, salaries, payroll taxes
  - Manufacturing
    - For management of the supply chain and for tracking production of items , orders of item

Database Applications

Enterprise Information (cont.)

- Various Enterprise Database Management System :

There are many enterprise databases such as :

- Oracle Database 18c
- Microsoft SQL Server
- IBM DB2
- SAP Sybase ASE
- PostgreSQL
- MariaDB Enterprise etc

Database Applications (cont.)

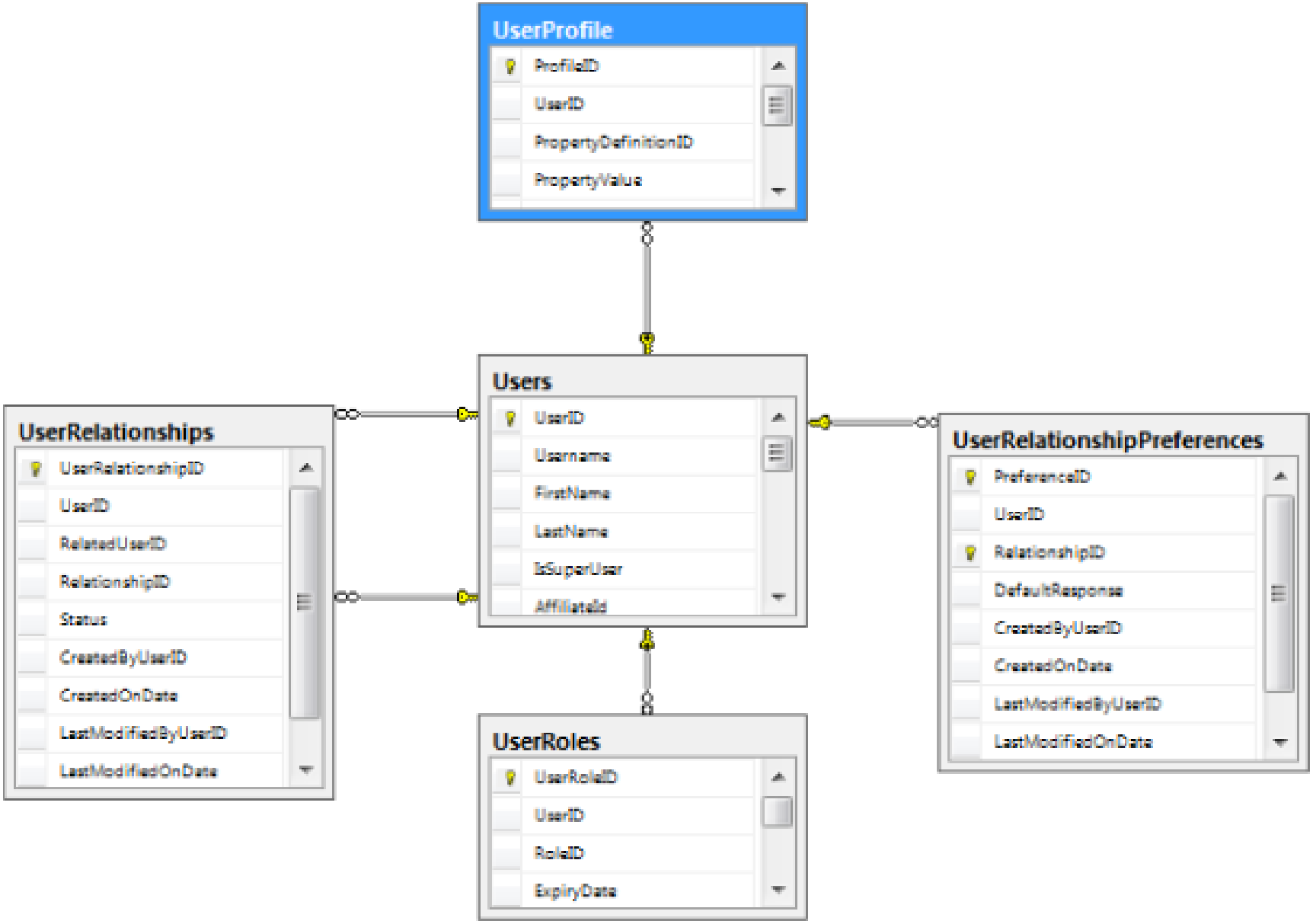
- Banking and Finance
  - Banking
    - For customer info, accounts, loans and banking transactions
  - Credit and transactions
    - For purchase on credit cards and generation of monthly statements
  - Finance
    - Sales and purchase of financial instruments such as stocks, storing real time market data
- Universities
  - For student info, course registration, grades
- Airlines
  - Reservations and schedule info

Many more…..

RDBMS

- RDBMS
  - Relational Database Management System
  - Data stored in table form
  - All the tables have relationship between them
  - Relationships are created by means of keys
  - Follows ACID Property and Normalization

RDBMS Example



Father of DBMS

Dr Edgar F Codd

- had propounded 12 rules. According to him, a DBMS is fully relational if it abides by all his twelve rules.
- If a management system or software follows any of 5-6 rules proposed by E.F.Codd, it qualifies to be a Database Management System (DBMS).
- If a management system or software follows any of 7-9 rules proposed by E.F.Codd, it qualifies to be a semi-Relational Database Management System (semi-RDBMS).
- If a management system or software follows 9-12 rules proposed by E.F. Codd, it qualifies to be a complete Relational Database Management System (RDBMS).

Dr Edgar F Codd's Twelve Commandments

- **Rule 0: Foundation rule**
  - a RDBMS should be able to manage the stored data in its entirety through its relational capabilities.
- **Rule 1: Rule of Information**
  - Relational Databases should store the data in the form of relations.
  - it is important to store the value as an entity in the table cells.
- **Rule 2: Rule of Guaranteed Access**
  - Every data entity which is atomic in nature should be accessed logically by using a right combination of the name of table, primary key represented by a specific row value and column name represented by attribute value.
- **Rule 3: Rule of Systematic Null Value Support**
  - Null values are completely supported in relational databases. They should be uniformly considered as 'missing information'. Null values are independent of any data type. They should not be mistaken for blanks or zeroes or empty strings. Null values can also be interpreted as 'inapplicable data' or 'unknown information.'

Dr Edgar F Codd's Twelve Commandments

- **Rule 4: Rule of Active and online relational Catalog**
  - The active online catalog that stores the metadata is called '***Data dictionary'***. The so called data dictionary is accessible only by authored users who have the required privileges and the query languages used for accessing the database should be used for accessing the data of data dictionary.
- **Rule 5: Rule of Comprehensive Data Sub-language**
  - should be able to define integrity constraints, views, data manipulations, transactions and authorizations.
- **Rule 6: Rule of Updating Views**
  - Views should reflect the updates of their respective base tables and vice versa. A view is a logical table which shows restricted data. Views generally make the data readable but not modifiable. Views help in data abstraction.

Dr Edgar F Codd's Twelve Commandments

- **Rule 7: Rule of Set level insertion, update and deletion**
  - A single operation should be sufficient to retrieve, insert, update and delete the data.
- **Rule 8: Rule of Physical Data Independence**
  - Batch and end user operations are logically separated from physical storage and respective access methods.
- **Rule 9: Rule of Logical Data Independence**
  - Batch and end users can change the database schema without having to recreate it or recreate the applications built upon it.
- **Rule 10: Rule of Integrity Independence**
  - Integrity constraints should be available and stored as metadata in data dictionary and not in the application programs.

Dr Edgar F Codd's Twelve Commandments

- **Rule 11: Rule of Distribution Independence**
  - The Data Manipulation Language of the relational system should not be concerned about the physical data storage and no alterations should be required if the physical data is centralized or distributed.
- **Rule 12: Rule of Non Subversion**
  - Any row should obey the security and integrity constraints imposed. No special privileges are applicable.

File System Vs Database System

In the early days, database applications were built directly on top of file systems

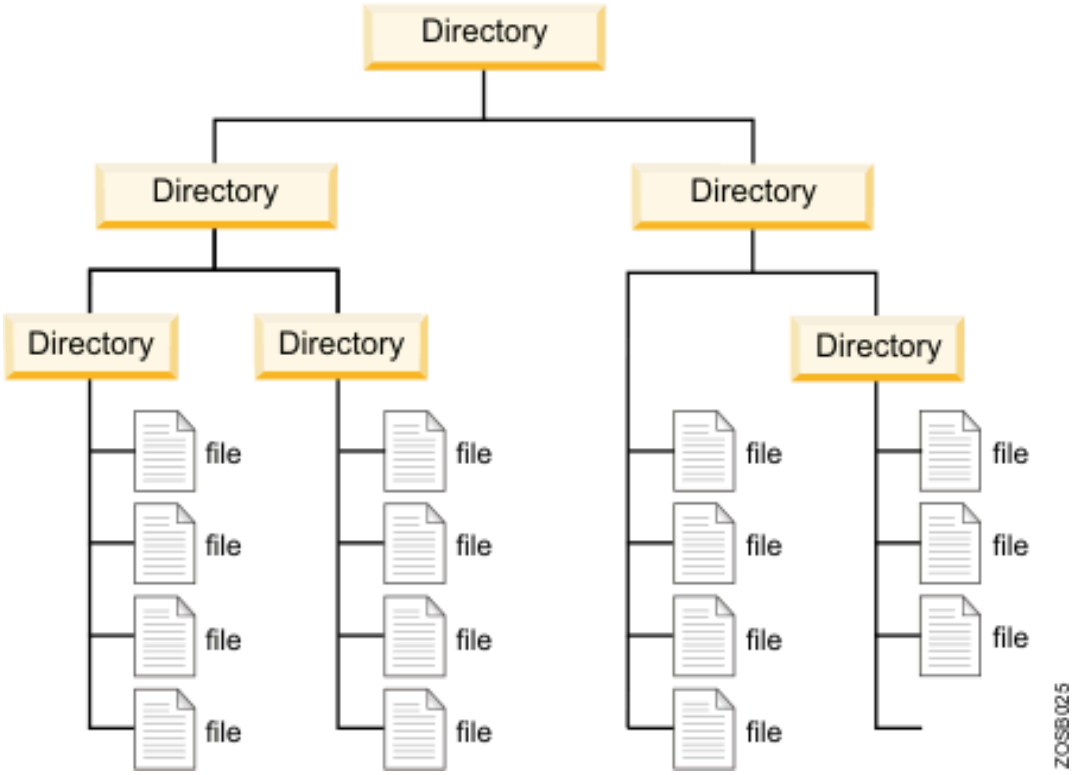


Fig. a) File System

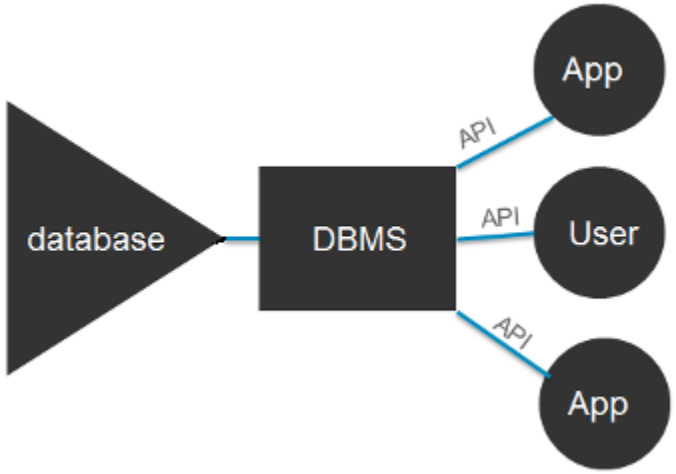


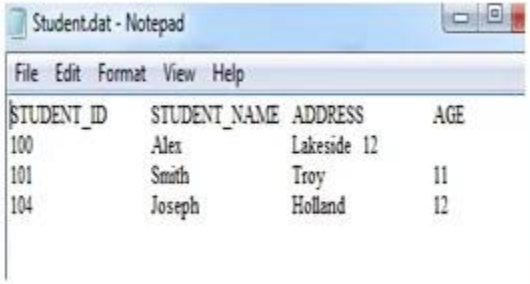
Fig. b) Database System

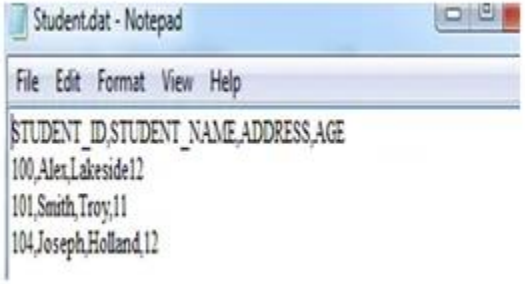
Fig. 1 File System Vs Database System

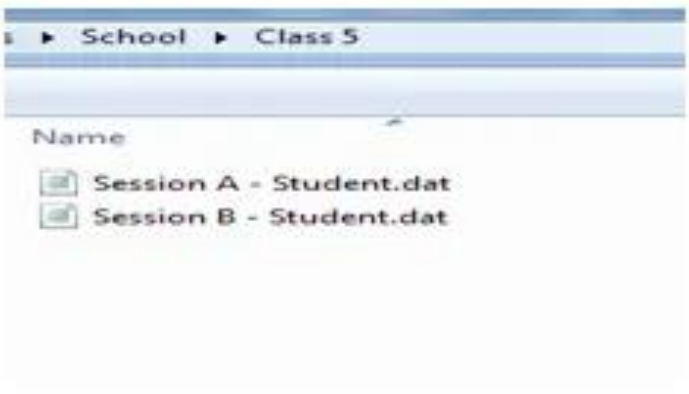
File System

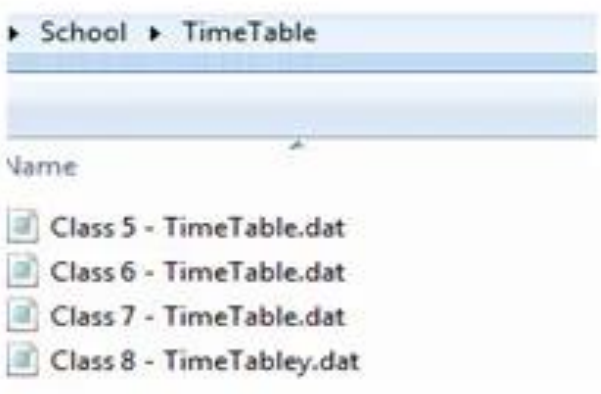
- Example – Student Information stored in Notepad.

Student files for each class were bundled inside different folders to identify it quickly.









File system Vs Database System (cont.)

- **Drawbacks of using file systems to store data**
  - Data redundancy and inconsistency
    - Multiple file formats, duplication of information in different files
    - No methods to validate insertion of duplicate data

STUDENT_ID	STUDENT_NAME	ADDRESS	SUBJECT
100	Joseph	Alaiedon Township	Mathematics
101	Allen	Fraser Township	Chemistry
100	Joseph	Alaiedon Township	Physics
102	Chris	Clinton Township	Mathematics
103	Patty	Troy	Physics

File system Vs Database System (cont.)

- **Drawbacks of using file systems to store data (cont..)**
  - Difficulty in accessing data
    - Any two independent files are not linked
    - Need to write a new program to carry out each new task
    - Time consuming and inefficient to search particular information with huge amount of data
  - Data Isolation
    - Multiple files and formats

File system Vs Database System (Cont.)

- **Drawbacks of using file systems to store data (cont.)**
  - Integrity problems
    - Integrity constraints (e.g., account balance > 0) become "buried" in program code rather than being stated explicitly
    - Hard to add new constraints or change existing ones
  - Atomicity of updates
    - Failures may leave database in an inconsistent state with partial updates carried out
    - Example: Transfer of funds from one account to another should either complete or not happen at all

File system Vs Database System (Cont.)

- **Drawbacks of using file systems to store data (cont.)**
  - Concurrent access by multiple users
    - Concurrent access needed for performance
    - Uncontrolled concurrent accesses can lead to inconsistencies
    - Example: Two people reading a balance (say 100) and updating it by withdrawing money (say 50 each) at the same time
  - Security problems
    - Each file can be password protected.
    - For example, the user has to be given access to view only their bank account information in the file. This is very difficult in the file system.

File system Vs Database System (Cont.)

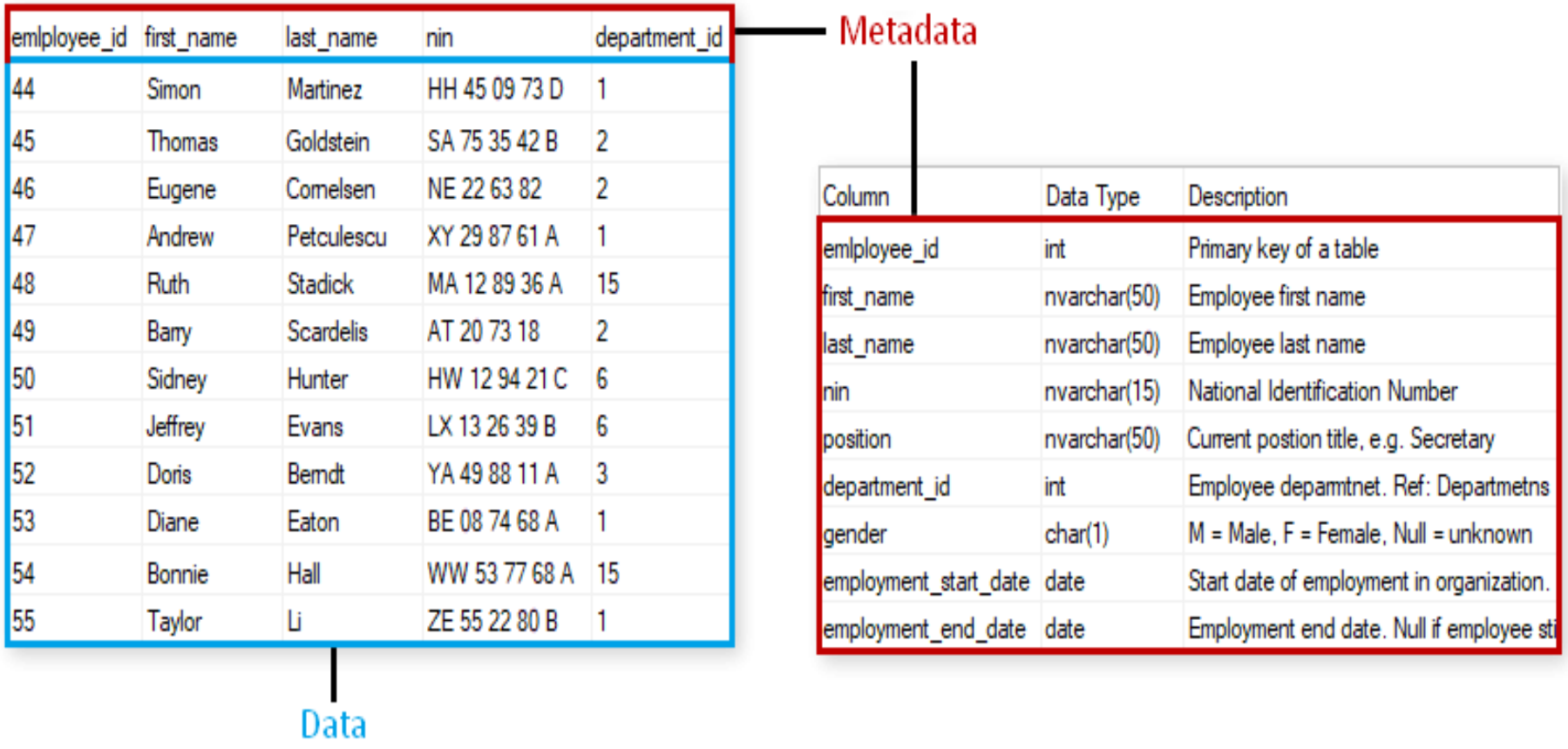
- **Drawbacks of using file systems to store data (cont.)**
  - Data Dependence
    - Program dependency on file
    - Difficulty during file processing

File system Vs Database System (Cont.)

Database systems offer solutions to all the above problems

Characteristics of Databases

- **Self-describing nature of a database system**
  - Contains database as well as metadata (description of data structure and its constraints)



Characteristics of Databases

- **Self-describing nature of a database system**
  - Catalog used by; DBMS Software and database users

RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE

Figure 1.3
An example of a database catalog for the database in Figure 1.2.

Characteristics of Databases (cont.)

- **Insulation between program and data**

  Physical-data independence:

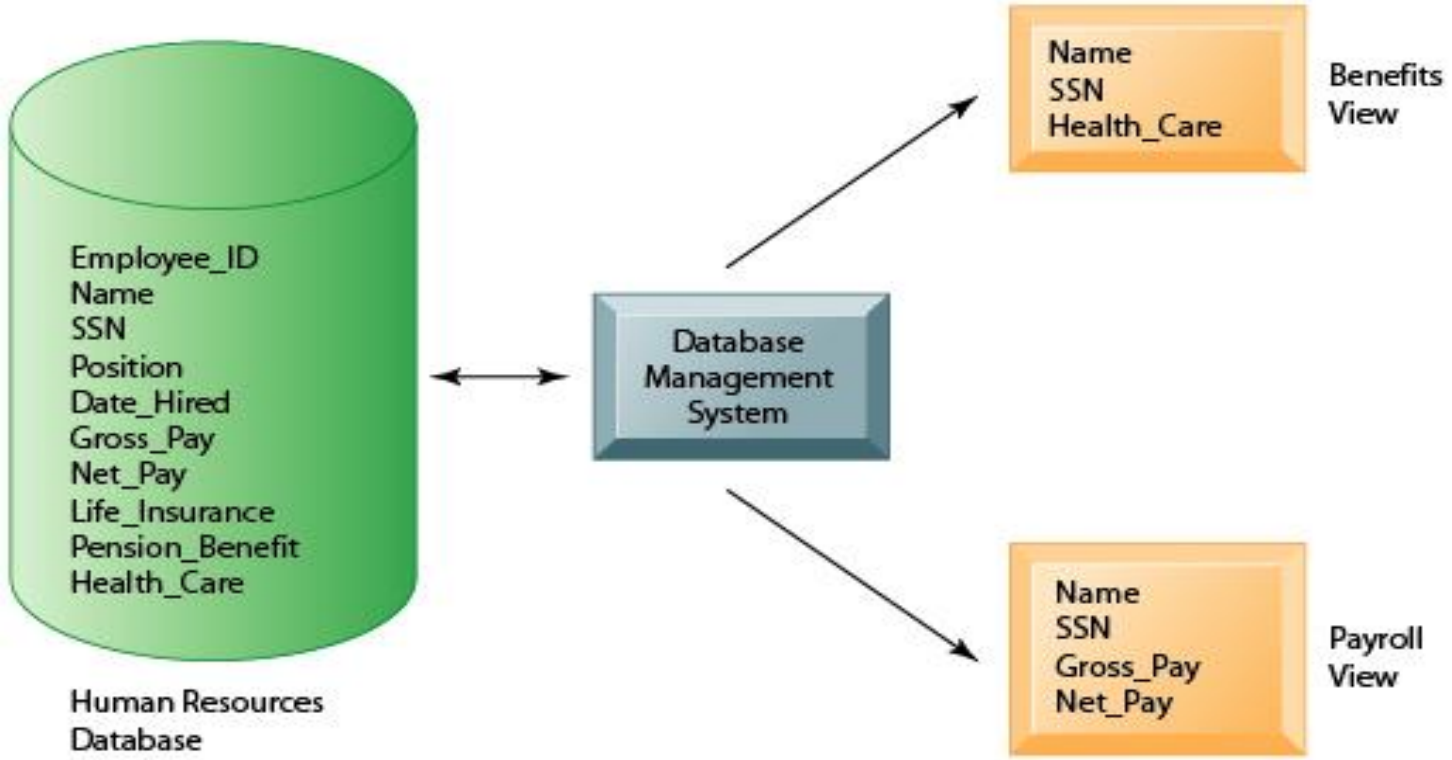
  – The ability to separate db definition from physical storage organization of the database
    - Structure of data file is stored in DBMS Catalog separately from access program
  – Any change in the physical structure of the database do not impact application program

Characteristics of Databases (cont.)

- **Data Abstraction**
  - Conceptual representation of data
    - Does not include details of how data is stored or how operations are implemented
    - (user need to know what data and what operations)
  - Data model
    - Type of data abstraction used to provide conceptual representation

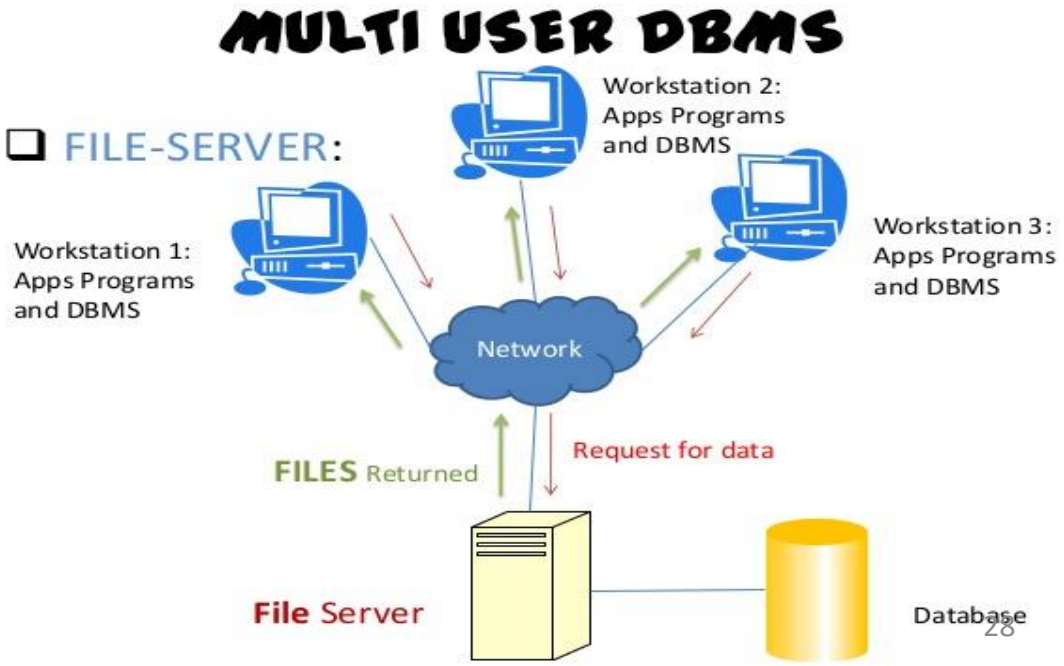
Characteristics of Databases (cont.)

- **Support for multiple views of data**
  - A *view* is a subset of the database, which is defined and dedicated for particular users of the system.
  - Each view might contain only the data of interest to a user or group of users.



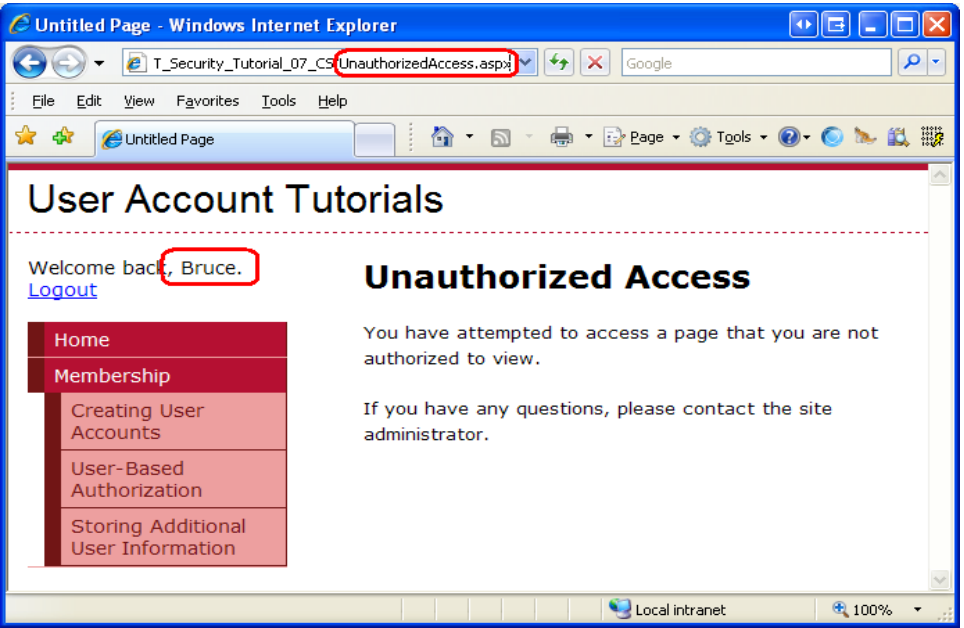
Characteristics of Databases (cont.)

- **Sharing of data and multiuser system**
  - Current database systems are designed for multiple users. That is, they allow many users to access the same database at the same time.
  - This access is achieved through features called concurrency control strategies. These strategies ensure data integrity .



Characteristics of Databases (cont.)

- **Access Flexibility and Security (Restriction of unauthorized access)**
  - A database management system should provide a security subsystem to create and control different types of user accounts and restrict unauthorized access.
  - Security is enhanced by concept of views – controlling access to the db system



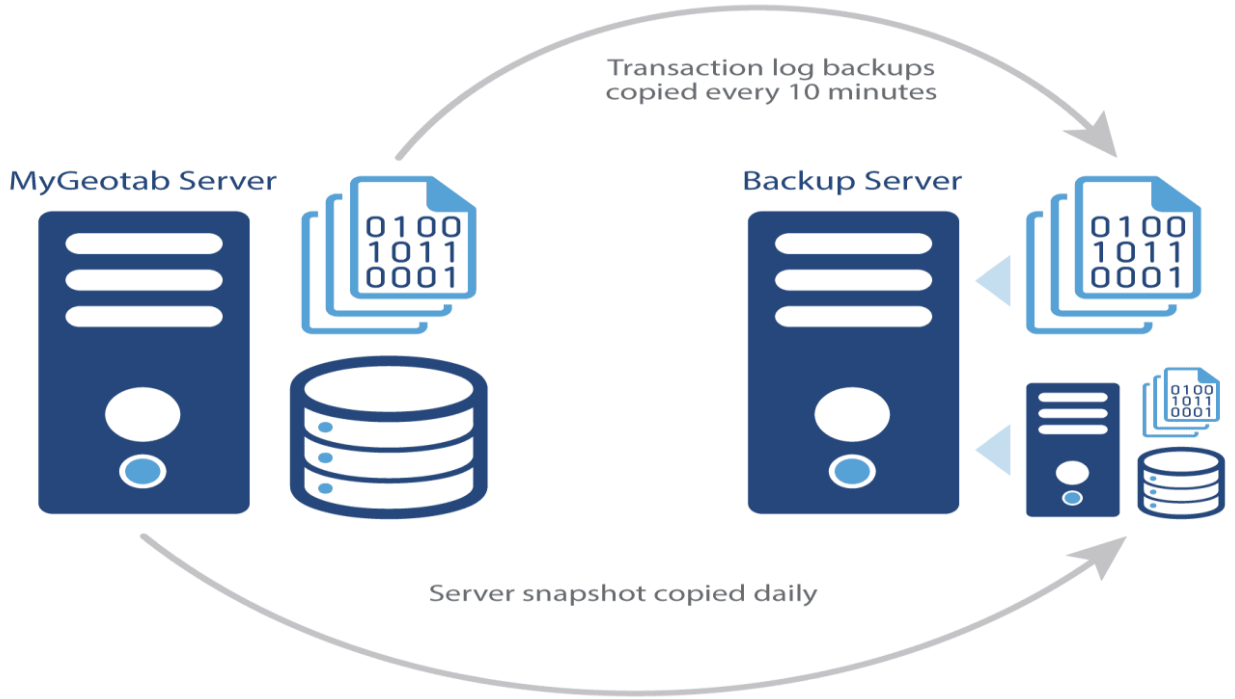
General Office	Library	Hostel	Account Office
RollNo	RollNo	RollNo	RollNo
Name	No_of_books_issued	RoomNo	Fee
Class	Fine	Mess_Bill	Installments
Father_Name	etc.	etc.	Discount
Address			Balance
Phone - No			Total
Date_of_birth			etc.
Previous_Record			
Attendance			
Marks			
etc.			

Characteristics of Databases (cont.)

- **Transaction processing**
  - A transaction is a program including a collection of database operations, executed as a logical unit of data processing.
  - The operations performed in a transaction include one or more of database operations like insert, delete, update or retrieve data.
  - It is an atomic process that is either performed into completion entirely or is not performed at all.
  - A database management system must include concurrency control subsystems.

Characteristics of Databases (cont.)

- **Backup and recovery facilities**
  - To protect your data from loss, the database system provides a separate process, from that of a network backup, for backing up and recovering data.



S.No	Difference factor	File System	DBMS
1	Definition	**Database management system (DBMS)** is a collection of interrelated data and a set of programs to access those data. Some of the very well known DBMS are Microsoft Access, Microsoft SQL Server, Oracle, SAP, dBASE, FoxPro, IBM dB2, SQLite etc.	**A file management system** is an abstraction to store, retrieve, management and update a set of files. A File Management System keep track on the files and also manage them.
2	Data Redundancy	In file system approach, each user defines and implements the needed files for a specific application to run. For example in sales department of an enterprise, One user will be maintaining the details of how many sales personnel are there in the sales department and their grades. Another user will be maintaining the sales person salary details.	Although the database approach does not remove redundancy completely, it controls the amount of redundancy in the database because in database approach, a single repository of data is maintained that is defined once and then accessed by many users. The fundamental characteristic of database approach is that the database system not only contains data's but it contains complete definition or description of the database structure and constraints.
3	Sharing of data	File system doesn't allow sharing of data or data sharing is very complex.	In DBMS data can be shared very easily due to centralized system
4	Data Consistency	When data is redundant, it is difficult to update. For e.g. if we want to change or update employee's address, then we have to make changes at all the places where data of that employee is stored. If by mistake, we forgot to change or update the address at one or more place then data inconsistency will occur i.e. the appearance of same data will differ from each other.	In DBMS, as there is no or less data redundancy, data remains consistent.

5	Difficult to search/access data	In conventional file system, if we want to search/retrieve/access some data item, it becomes very difficult because in file system for every operation we have to we have to make different programs.	In DBMS searching/retrieval/accessing of data item is very easy and user-friendly because searching and querying operations are already available in the system.
6	Data isolation	In file system there is no standard format of data or we can say data is scattered in various formats or files which also make data retrieval difficult.	In DBMS, due to centralized system the format of similar type of data remains same.
7	Data Integrity	The value of data in database must follow or satisfy some rules or consistency constraints. For e.g. A company have a policy that the age of an employee must be >=18. The value which is not satisfying this constraint must not be stored in the respective column. 8In file system, there is no procedure to check these constraints automatically.	DBMS maintains the data integrity by enforcing the constraints by adding appropriate code.
8	Security problems	In file system there is no or very less security. General security provided by file system are locks, guards etc.	DBMS have high level security like encryption, passwords, biometric security (fingerprint matching, face and voice detection etc) etc.

9	Atomicity	Atomicity means a transaction must be all-or-nothing i.e. the transaction must either fully happen, or not happen at all. It must not complete partially. E.g. if A want to transfer 5000rs to B's a/c. In this case A's a/c should be debited and B's a/c should be credited with the same amount. Let suppose A's a/c is debited with 5000rs and then transaction fails. Now the transaction is incomplete because B's a/c is not credited. These type of problems occur in file system because there is no procedure to stop such type of anomalies.	Transaction atomicity is a special feature of DBMS. In DBMS either a transaction completed fully or none of the action is performed. For this, DBMS maintains the transaction log in which intermediate values are stored.
10	Concurrent Access Anomalies	Any multi-user database application has to have some method for dealing with concurrent access to data -- when more than one user is accessing the same data at the same time. A problem occurs when user X reads a row for editing, user Y reads the same row for editing, user Y saves changes, then user X saves changes. The changes made by user Y are lost unless something prevents user X from blindly overwriting the row. File system does not provide any procedure to stop such type of anomalies.	DBMS along with an appropriate application provides safety towards concurrent access. For this locks are available in DBMS. If 2 or more transactions want to change/update or write a data item, an exclusive lock is issued to one of these transactions. Until and unless the transaction release that lock no other transaction can acquire the lock and hence cannot update/write the data item.

Users of Database

- **Database administrators:**
  - Responsible for authorizing access to the database, for coordinating and monitoring its use, acquiring software and hardware resources, controlling its use and monitoring efficiency of operations.
- **Database Designers:**
  - Responsible to define the content, the structure, the constraints, and functions or transactions against the database. They must communicate with the end-users and understand their needs.
  - Two types: Logical DB Designer and Physical DB Designer
    - Logical: identify data and relationship between them
    - Physical: how the way db model can be best mapped to physical storage

Users of Database (cont.)

- **System Analysts and Application Programmers:**
  - These are technical users determine the requirement of end user.
    - System Analyst develop specifications for application program, responsible for the design, structure and properties of database
    - Application programmer implement these specifications (interaction through DML calls)
- **End-users:**
  - They use the data for queries, reports and some of them update the database content.
  - Casual, parametric, sophisticated, standalone users etc.

Users of Database (cont.)

- **Casual users –**
  - The casual end users access the database occasionally. Each time they access the database their request will change.
  - They use sophisticated database query language to retrieve the data from the database.
  - Example – High level managers who access the data weekly basis.

- **Naïve or parametric end users:**
  - Require constant querying and updating db, using standard type of queries and updates (canned transactions)
  - Naïve: no need to understand DBMS. Ex. Bank teller, cashier, reservation agent.

Users of Database (cont.)

- **Sophisticated users:**
  - Sophisticated: familiar with DBMS, likely to be using SQL. Ex. Engineer, Scientist, Business analyst
  - They directly interact with the database by means of SQL.
  - This category includes designers and developers of DBMS and SQL.
- **Stand-alone Users :**
  - These users will have a stand-alone database for their personal use.
  - These kinds of the database will have readymade database packages which will have menus and graphical interfaces.

Users of Database (cont.)

- **Workers behind the scene:**
  - System Designers and implementers – Design and implement of DBMS modules and interfaces as package.
  - Tool developers for db design, performance monitoring, graphical interfaces
  - Operators and maintenance personnel, system admin personnel responsible for actual running and maintenance of the h/w and s/w environment for the db system

Concerns when using an enterprise database

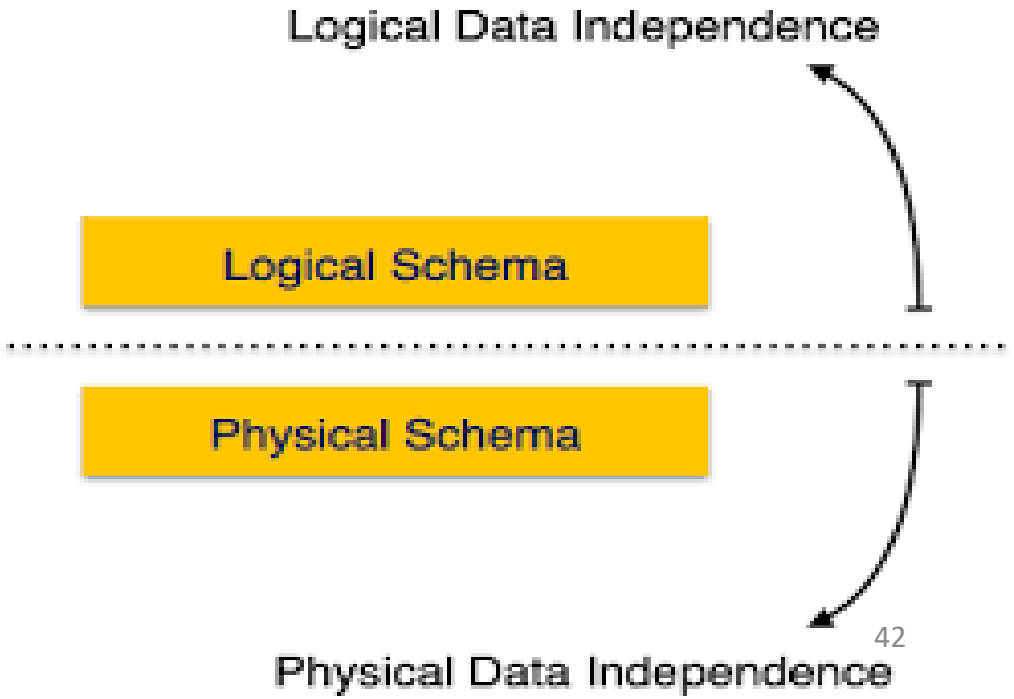
- **Enterprise vulnerability:**
  - Database has to be kept safe against destruction, unauthorized modification etc. in order to make full time availability.
- **Confidentiality, privacy and security:**
  - Centralized database must be protected from various security threats.
  - Necessary to take technical, administrative and possibly legal measures

Concerns when using an enterprise database (cont.)

- **Data quality:**
  - Controlling data quality and maintaining integrity
- **Cost of building and using a DBMS**
- **Inability to change the database when necessary:**
  - Easily adaptable to new changes

Data Independence

- Deals with independence between the way the data is structured and the programs that manipulate it.
- Two levels of data independence:
  - Physical Data Independence
  - Logical Data Independence



Data Independence (cont.)

- Physical Data Independence
  - modify the physical schema without impacting the schema or logical data.
  - For 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 Independence (cont.)

- Logical Data Independence
  - The ability to change the logical (conceptual) schema without changing the External schema (User View) is called logical data independence.
  - For example,
    - the addition or removal of new entities, attributes, or relationships to the conceptual schema.
    - some changes on table format (Any changes to the database objects like changes to table structure, size or addition/removal of columns from the table will not affect user views and data residing on the disk.

Data Independence (cont.)

- Application Program should not need to know:
  - The ordering of the data fields in a record
  - The ordering of records in a file
  - The size of each record
  - The size of each field
  - The format and type of each data item
  - Whether a file is sorted or not
  - The type of data structure being used to store some of the data

Data Dictionary

- A data dictionary contains metadata i.e data about the database.
- It contains information such as what is in the database, who is allowed to access it, where is the database physically stored etc.
- The users of the database normally don't interact with the data dictionary, it is only handled by the database administrators.
- The data dictionary in general contains information about the following −
  - Names of all the database tables and their schemas.
  - Details about all the tables in the database, such as their owners, their security constraints, when they were created etc.
  - Physical information about the tables such as where they are stored and how.
  - Table constraints such as primary key attributes, foreign key information etc.
  - Information about the database views that are visible.

Data Dictionary example

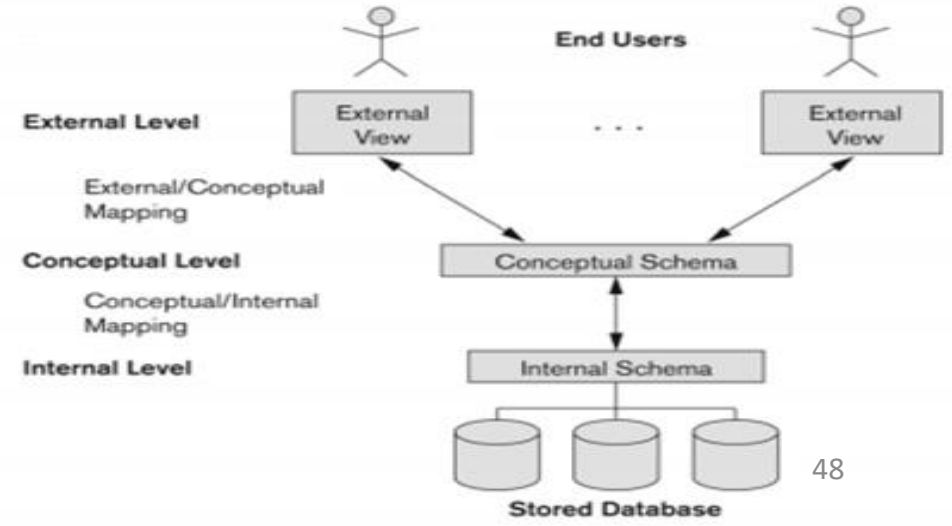
This is a data dictionary describing a table that contains employee details.

Field Name	Data Type	Field Size for display	Description	Example
Employee Number	Integer	10	Unique ID of each employee	1645000001
Name	Text	20	Name of the employee	David Heston
Date of Birth	Date/Time	10	DOB of Employee	08/03/1995
Phone Number	Integer	10	Phone number of employee	6583648648

Three-level DBMS Architecture

- External view
  - Each user has his own view of his enterprise db.
  - Ex. Department head may only require departmental finances
- Conceptual view
  - View of whole enterprise
  - Represented by conceptual schema includes details of each of various types of data
- Internal view
  - Physical or storage view concern about data structure,

Access paths, indexing, keys, encryption techniques etc.



DBMS System Architecture (*Navathe*)

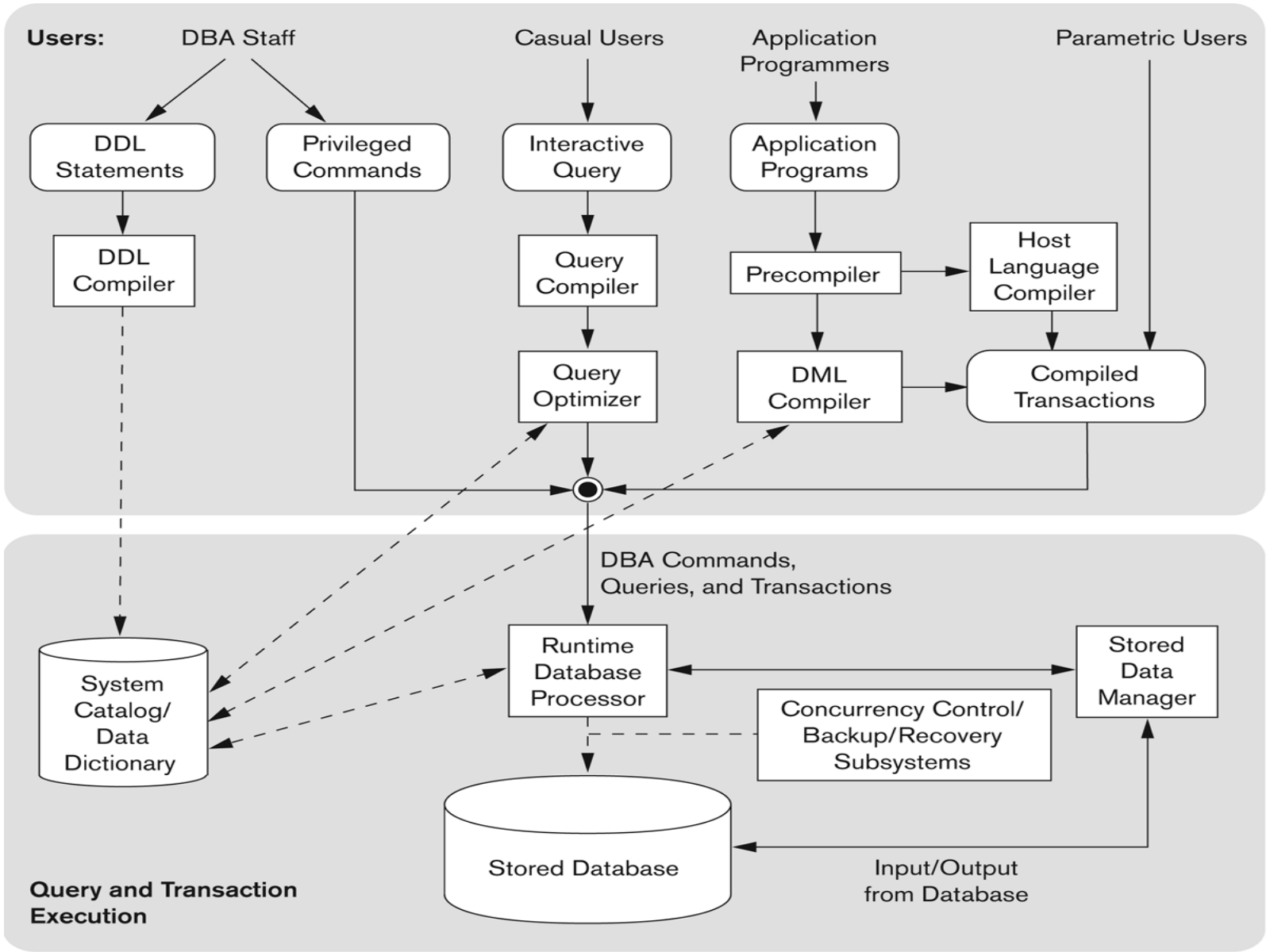


Figure 2.3
Component modules of a DBMS and their interactions.

DBMS System Architecture (*Navathe*)

- DDL Compiler : Process schema definition and stores metadata in catalog
- Query Optimizer: rearrangement/reordering of operations, eliminating redundancies using optimized algo for query execution
- Application Prog: written in C, python, java etc.
- Precompiler: seperates DML statements from programming language statements
- DML Compiler: generates object codes
- Object code and canned transactions are submitted to runtime processor
- Runtime Processor: handles privileged commands, query plans, canned transactions, system directory
- Stored db manager: carry out I/O db operations between disk and memory

DBMS System Architecture (cont.)

Query Processor

- It interprets the requests (queries) received from end user via an application program into instructions. It also executes the user request which is received from the DML compiler.

  DML Compiler –

  It processes the DML statements into low level instruction (machine language), so that they can be executed.

  DDL Interpreter –

  It processes the DDL statements into a set of table containing meta data (data about data).

  Query Optimizer –

  It executes the instruction generated by DML Compiler.

  Query evaluation engine-

  This engine will execute low-level instructions generated by the DML compiler on DBMS.

DBMS System Architecture (cont.)

Storage Manager

- Storage Manager is a program that provides an interface between the data stored in the database and the queries received.

- It is also known as Database Control System. It maintains the consistency and integrity of the database by applying the constraints and executes the DCL statements. It is responsible for updating, storing, deleting, and retrieving data in the database.

Authorization Manager –

It ensures role-based access control, i.e. checks whether the particular person is privileged to perform the requested operation or not.

Integrity Manager –

It checks the integrity constraints when the database is modified.

DBMS System Architecture (cont.)

Storage Manager (cont.)

Transaction Manager –

It controls concurrent access by performing the operations in a scheduled way that it receives the transaction. Thus, it ensures that the database remains in the consistent state before and after the execution of a transaction.

File Manager –

It manages the file space and the data structure used to represent information in the database.

Buffer Manager –

It is responsible for cache memory and the transfer of data between the secondary storage and main memory.

DBMS System Architecture (cont.)

Disk Storage

Data Files – It stores the data.

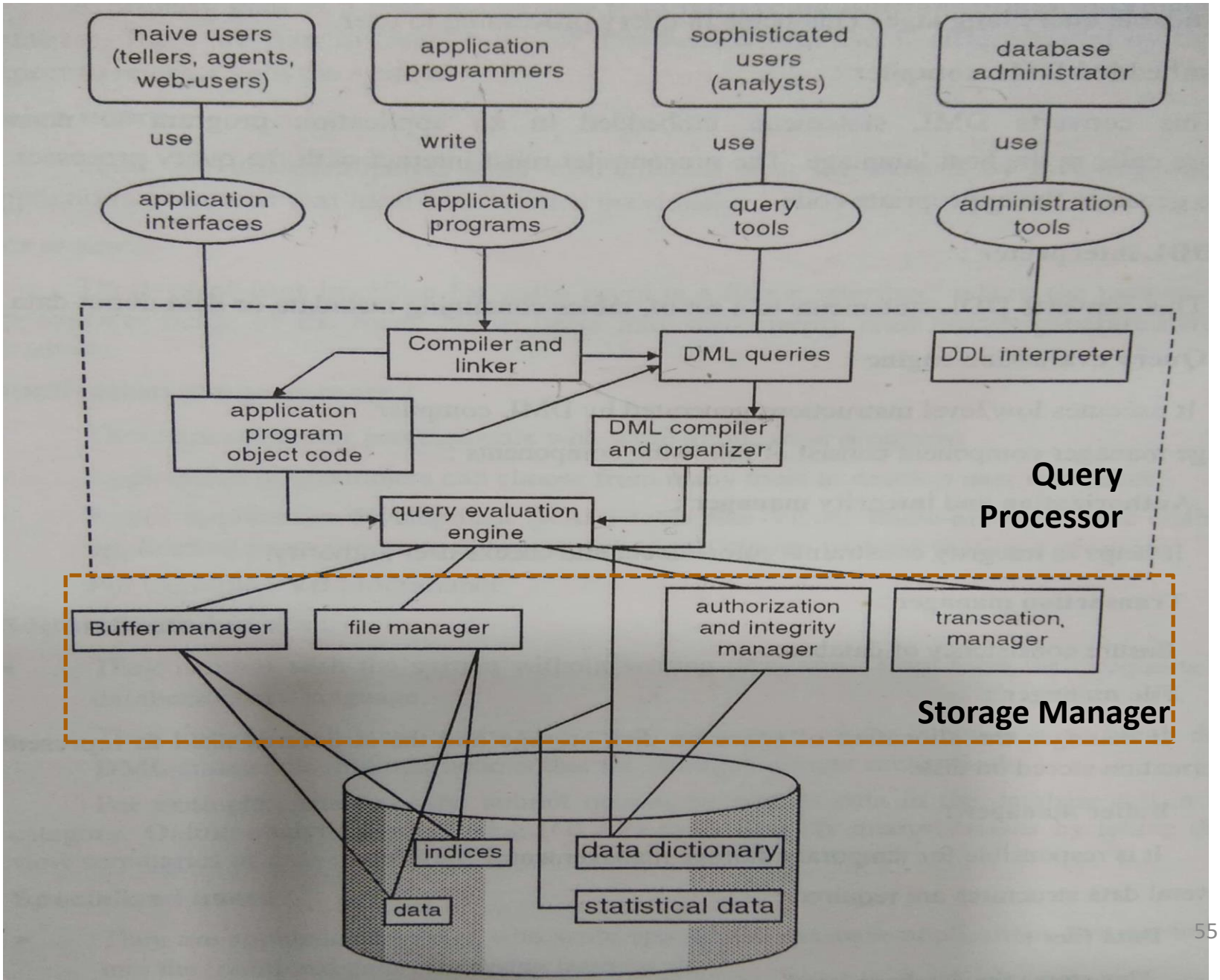
Data Dictionary –

It contains the information about the structure of any database object. It is the repository of information that governs the metadata.

Indices –

It provides faster retrieval of data item

DBMS System Architecture (*korth*)



DBMS System Architecture

Various components -

- User communication, authentication and authorization:
  - Mechanism for user to log in and be authenticated
- Query Processor:
  - Subcomponents - query parser, query rewriter, query optimizer, query executor
  - Query processor for interpreting DDL, DML statements, correct syntax and translating into q. execution plan
  - Query optimizer for finds best plan
  - Plans are passed to db manager for execution

DBMS System Architecture (cont.)

Various components -

- Transaction Management and Concurrency Control:
  - Transactions in multi-user system must be managed concurrently
  - No. of components – Concurrency control, lock manager, log manager, recovery manager
- File and Access Methods:
  - Interacts with physical storage of the database
  - Responsible for storing db on the disk and provide DBA no of data structure options for storing db
  - Assist in maintaining metadata of db as well as indexes for the various files

DBMS System Architecture (cont.)

Various components -

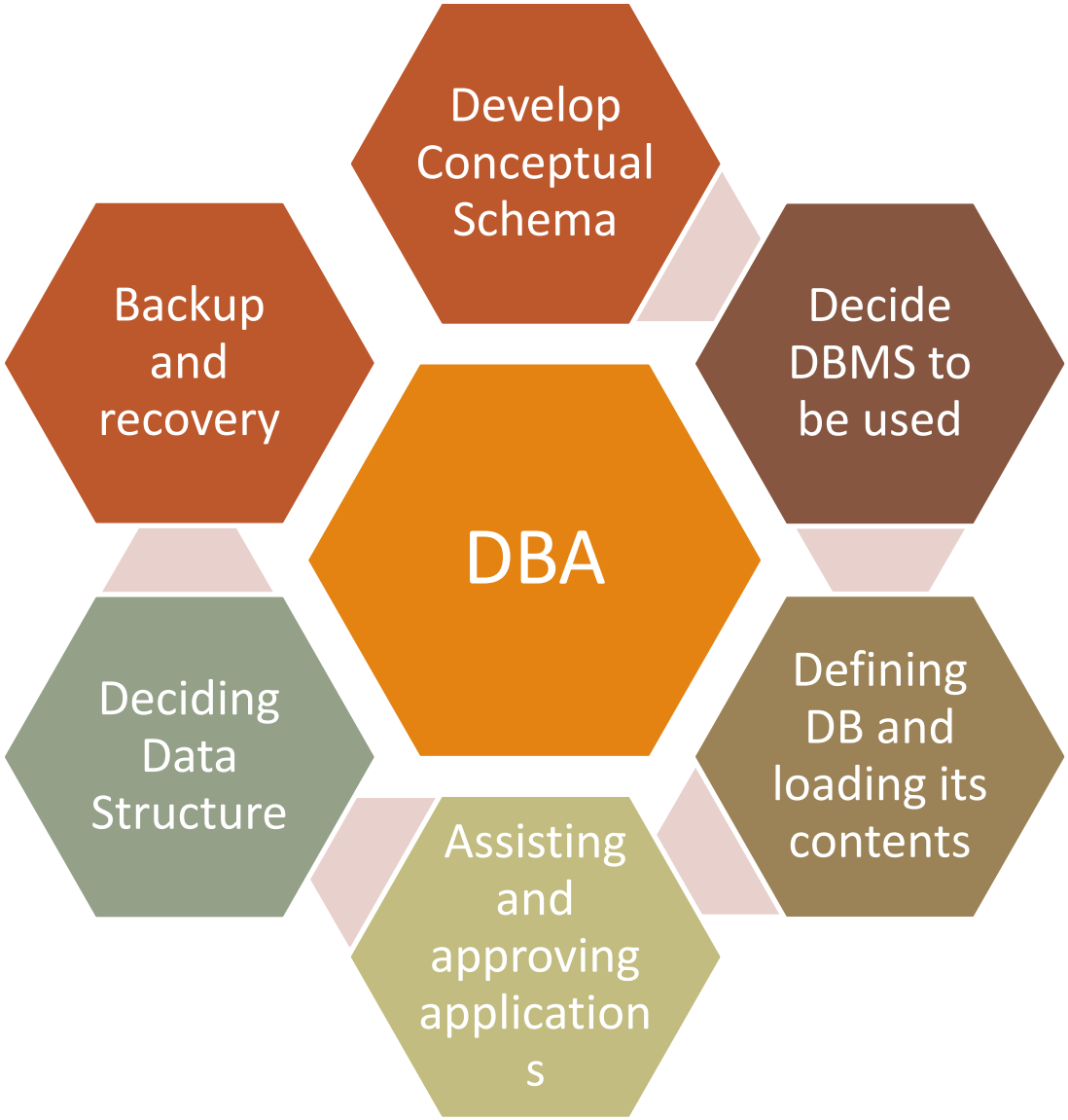
- Buffer Manager:
  - Responsible for handling the interfacing between the main memory and disk
  - Maintains no of pages of same size and a large buffer size is important for the performance of a large and active db system
- Storage Manager:
  - Support for management of access paths and indexes
- Catalog Manager:
  - Metadata maintained in catalog and stored as relational tables, hence need to be queried as per requirement

DBMS System Architecture

Various components -

- Physical Database:
  - Available on disk. The performance of a large db depends on how efficiently the data has been stored on the disk

Database Administrator (1)



Database Administrator (2)

- **Decides hardware –**

  Based upon cost, performance and efficiency of hardware, and best suits organization. It is hardware which is interface between end users and database.

- **Manages data integrity and security –**

  Data integrity need to be checked and managed accurately as it protects and restricts data from unauthorized use.

- **Database design –**

  DBA is held responsible and accountable for logical, physical design, external model design, and integrity and security control.

- **Database implementation –**

  DBA implements DBMS and checks database loading at time of its implementation.

Database Administrator (3)

- **Query processing performance –**

  DBA enhances query processing by improving their speed, performance and accuracy.

- **Tuning Database Performance –**

  If user is not able to get data speedily and accurately then it may loss organization business. So by tuning SQL commands DBA can enhance performance of database.

References

- G K Gupta, Database Management Systems
- Elmasri Navathe, Fundamentals of Database Systems
- Korth, Database System Concepts