INTRODUCTION

Structure:

Introduction:

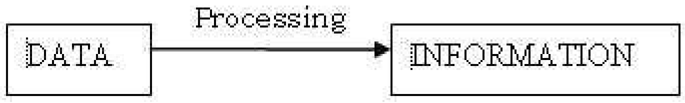
Data is termed as the collection of meaningful unorganized facts, concepts or instructions. Information is processed form of data. Traditional File Processing System is a method for storing and organizing data stored in computer files. But Traditional File Processing has significant disadvantages of data redundancy, lack of flexibility, data dependency etc.
In this lesson, we first provide the formal definitions of data, information, and traditional file processing system. Then we define the limitations of traditional file processing system and finally discuss the concepts of database

Data and Information

Data is defined as a collection of meaningful facts which can be stored and processed by computer or humans. The main examples of data are names, phone numbers, marks, age of employees etc.

Information is processed form of data which helps us in making decisions. For examples: the roll no and marks of all the students is the data and when it processed and report cards of the students are prepared for taking the decision that which student stood first, second and third in the class , then it becomes the information.

Pictorially we can show it as:



The term data refers to factual information, especially that used for analysis and based on reasoning or calculation. Data itself has no meaning, but becomes information when it is interpreted. Information is a collection of facts or data that is communicated. However, in many contexts they are considered and are used as synonyms. Data, by the way, is the plural of datum. Information comes from Latin informationem 'concept, idea' or 'outline'.

The relationship between data and information works this way:

Data = the facts about a topic.

Information = evaluated data used to answer a question.

Traditional File Processing System:

Let's take the example of our university to explain this system. Our objective is to computerize the records of various departments of the university. By not going so deep into the records, we just try to computerize the records of employees of the university. In the past system, records were stored on the registers manually. Now in this system, we are using computer for storing the records of employees. For storing the records, we use files in computer. Each file has a specified format for storing the file. Suppose we are only storing the name, date of birth, date of joining, department, permanent address, correspondence address of employee in the file. So, all these are fields of a record. The information of all fields for a particular employee will become a record of a particular employee in that file. All the fields are separated by a tab in a file and one record is stored in a line in file.

Thus, for example following will be the format of the file containing the records of the university employees maintained by the establishment department of the university:

S.No.	Name	DOJ	Department	Permanent Address	Correspondence address
1.	Narinder	19-02-2006	Accounts Branch	Patiala	Accounts Branch, P.U.Patiala
…….					
…….					
…….					

In the similar manner, Accounts Branch, Punjabi University will also maintain the record of its employee for recording the day to day activities of the employees. Thus this file maintained by Accounts Branch will also contain the name, DOB, permanent address, correspondence address and other field specific to branch.

Formally, a record is a collection of related fields that are treated as a single unit. Further all the records are grouped together to form a file. All the related files are grouped together to form a database. Thus, in the above example, all the files of the university are interrelated to each other, when grouped together they form database of the university. The database of the university consists of account file, student record file, employee record file etc.

So, Traditional file processing system has for each application a separate master file and its own set of personal files. Such file based approaches which came into being with the first commercial application of computers did provide an increased efficiency in the data processing compared to earlier manual paper record-based systems. As the demand for efficiency and speed increased, the computer-based simple file oriented approach to information processing started suffering from number of limitations which is explained in the following section.

Limitations of Traditional File Processing System

Following are the limitations of traditional file processing system:

1. Data Redundancy: By 'Data Redundancy' we mean the duplication of same data at various places. In Traditional File Based Systems, each application has its own private files. So, same data is stored in different files. For example, In the above example of university, Department is storing the personal details of employees and Accounts Department is also storing the personal details of the employees for their purpose. This fact leads to considerable redundancy in stored data, with resultant wastage in storage space.
2. Data Inconsistency: By 'Data Inconsistency' we mean that same data stored in different files do not match with each other at particular moment of time. For example, when the employee had joined the university, he has given some correspondence address. But after some period of time, his correspondence address changes and he has informed his department only and forgets to inform other departments. Thus one file (maintained by his own department) is updated with new correspondence address, but the other files (maintained by other departments) are still storing the obsolete correspondence address of that employee. This is known as Data Inconsistency.
3. Difficulty in sharing data: In this system, each file has its own format for storing data. By Format we mean that delimiters separating each record in a file and further for each field in a record can or cannot be same for different files in an application. Thus for using the data from other file in a file requires to know the

format of the file from which the data is to be shared. So, it becomes very difficult.

4. **Incompatible File Formats:** Each programmer stores the data in the files in the format as per his choice as there is no standard file format for storing the files. Thus, it becomes very difficult to share data among the files having different format. Even If some other programmer has to work on a file stored by different programmer, he hasto first understand the file format of that file and then start working on that file.

5. **Lack of Data Security:** The data stored in the database must be protected from unauthorized access. Since in Traditional File Processing System, the application programs are added to the system on the basis of queries which are not predefined so it is difficult to enforce security measures on these application programs.

6. **Lack of data integrity:** Data Integrity means data correctness. For example: Basic Pay cannot be negative. Such type of data constraints can be imposed in the traditional file processing system by writing appropriate code in the application programs. But in future, if more constraints are to be added then, we need to modify the application program and sometimes it is not possible to change the application code. Thus it may lead to poor data which may result in bad/wrong decisions.

7. **Data Dependency:** Data Dependence means it is impossible to change the storage structure without affecting the application program. For example, if we change the delimiter separating the fields in the records of a file from tab to double space, we must change the code in the application program that access data from that file as the structure of file gets changed. Due to this we cannot change the storage structure without changing the application code.

8. **Lack of Flexibility:** In Traditional File Processing System, programmers are already told about which type of queries need to be answered by the application. And programmers code that query using the data files for that application. But in the fast moving and competent business environment of today, apart from such regular queries, there is a need for responding to un-anticipatory queries, and then the system will fail. Then the programmer again has to code for such queries. Thus this limitation leads to lack of flexibility of the file system.

9. **Inadequate data modeling of real world:** The file system approach has inability to design a database which shows the basic entities, relationships and events facing the organization every day. Complex data and interfile relationships cannot be formally defined to the system.

10. **Concurrency Problem:** Concurrency means simultaneous access of the same file by two or more users. When data in a file is simultaneously accessed by two or more users for updating the data, there must be a system that finally does not lead to inconsistent data. But in this file system, it is not possible to implement such feature. If possible, it is very difficult to implement.

Database and Database System: Database is collection of related data. The database can be of varying sizes and varying complexity. In order to overcome the limitations of the traditional file processing system, the concept of Database Systems was introduced. A database system is basically a computerized record keeping system i.e. it is a computerized system whose overall purpose is to maintain information and to make the information available on demand.

In other words, a database is a collection of interrelated data stored in a database server. These data will be stored in the form of tables. The primary aim of database is to provide a way to store and retrieve database information in fast and efficient manner. There are number of characteristics that differ from traditional file management system. 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. These details will be stored and maintained in a separate file. Another, (d) user will be maintaining the salesperson salary details working in the concern. The detailed salary report will be stored and maintained in a separate file. Although both the users are interested in the data of the salespersons they will be having their details in a separate file and they need different programs to manipulate their files. This will lead to wastage of space and redundancy or replication of data, which may lead to confusion. Sharing of data among various users is not possible due to which, data inconsistency may occur. These files will not be having any inter-relationship among the data stored in these files. Therefore in traditional file processing, every user will be defining his own constraints and implement the files needed for the applications.

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 but it contains complete definition or description of the database structure and constraints. These definitions are stored in a system catalog, which contains the information about the structure and definitions of the database. The information stored in the catalog is called the metadata, it describes the primary database. Hence this approach will work on any type of database for example, insurance database, Airlines, banking database, Finance details, and Enterprise information database. But in traditional file processing system the application is developed for a specific purpose and they will access specific database only.

The other main characteristic of the database is that it will allow multiple users to access the database at the same time and sharing of data is possible. The database must include concurrency control software to ensure that several users trying to update the same data at the same time, do so in a controlled manner. In file system approach many programmers will be creating files over a long period and various files have different format in various application languages.

Therefore there is possibility of information getting duplicated. This redundancy in storing same data multiple times leads to higher costs and wastage of space. This may result in data inconsistency in the application; this is because update is done to some of the files only and not all the files. Moreover in database approach multiple views can be created. View is a tailored representation of information contained in one or more tables. View is also called as “Virtual table” because view does not contain physically stored records and will not occupy any space.

A multi-user database whose users have variety of applications must provide facilities for defining multiple views. In traditional file system, if any changes are made to the structure of the files, it will affect all the programs, so changes to the structure of a file may require changing all the programs that access the file. But in case of database approach the structure of the database is stored separately in the system catalog from the access of the application programs. This property is known as program-data independence.

Database can be used to provide persistent storage for program objects and data structures that resulted in object oriented database approach. Traditional systems suffered from impedance mismatch problem and difficulty in accessing the data, which is avoided in object oriented database system. Database can be used to represent complex relationships among data as well as to retrieve and update related data easily and efficiently.

It is possible to define and enforce integrity constraints for the data stored in the database. The database also provides facilities for recovering from hardware and software failures. The backup and recovery subsystem is responsible for recovery. It reduces the application development time considerably when compared to the file system approach and there is availability of up-to-date information of all the users. It also provides security to the data stored in the database system.

Example: University Database

Used to maintain information concerning students, courses and grades in a university environment.

The database has five files, each of which stores data records of the same type.

The Student file stores data of each student,

The Course file stores data on each course,

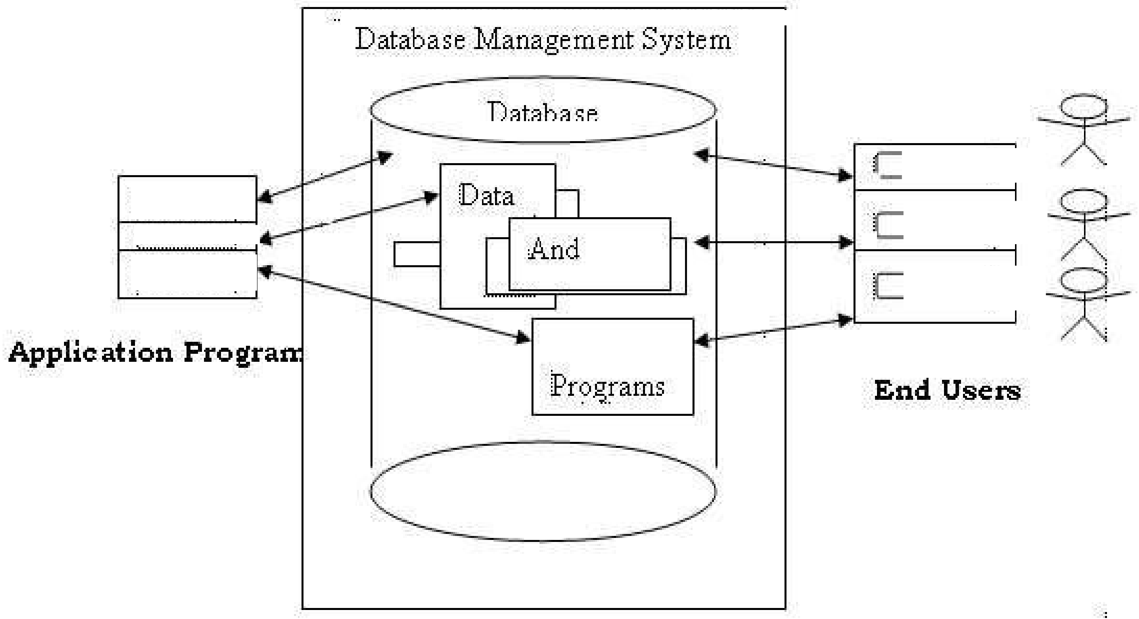
The Section file stores data on each section of a course, The Grade_Report stores information on the grades students receive in the various sections they have completed,

The Prerequisite file stores the prerequisite of each course.

Database Approach vs. Traditional File Processing

- Self contained nature of database systems (database contains both data and meta-data).
- Data Independence: application programs and queries are independent of how data is actually stored.
- Data sharing.
- Controlling redundancies and inconsistencies.
- Secure access to database; Restricting unauthorized access. Enforcing Integrity Constraints.
- Backup and Recovery from system crashes.
- Support for multiple-users and concurrent access.

Components of Database System: A database system comprises of four major components namely, data, hardware, software, and users. The simplified view of a database system is as follows:



Application Programs
End Users

Data: It is very important component of the database system. Most of the organizations generate, store and process large amount of data. The users which directly access it or access it through some application programs. In general, the data in the database will be both integrated and shared. Let us explain why we mean by the terms “integrated and shared”:

By Integrated we mean that the database can be thought of as unification of several distinct files, with any redundancy among those files wholly or partly eliminated.

By shared, we mean that individual pieces of data in the database can be shared among several different users, in the sense that each of those users can have access to the same piece of data. Such sharing is the consequence of the fact that database is integrated.

Hardware:

The hardware of the database system consists of:

1. The secondary storage devices, usually, magnetic disks, hard disks, CD-ROMs, Floppy Disks - that are used to hold the stored data, together with the associated I/O devices, device controllers etc.
2. The processor(s) and associated main memory that are used to support the execution of the database system software.

Software:

Software layer is present in the database system between the physical database itself i.e. where the data is actually stored and the users of the system and is known as Database Management System (DBMS). All the requests from the user for access to the database are handled by DBMS. One general function of DBMS is thus the shielding of database users from hardware level details. Basically, DBMS acts like a bridge between users and database. Database management systems are programs that are written to store, update, and retrieve information from a database. There are many databases available in the market. The most popular are the Oracle and SQL Server. The Oracle database is from the Oracle Corporation and the SQL Server is from the Microsoft Corporation. There are freely available database like MySQL. These are open source databases. Database Management Systems are available for personal computers and for huge systems like mainframes. DB2 is a database from IBM for Mainframe systems.

Users:

There are a number of users who can access or retrieve data on demand using application programs and interfaces provided by DBMS. Each type of user needs different software capabilities. Following are the categories of the users:

1. Application Programmers
2. End Users
3. Database Administrator (DBA)

Application Programmers: Users who are responsible for writing application programs that use database. These programs operate on data for retrieving existing information, inserting new information, deleting or changing existing information.

End Users: End Users are those who need not know about the presence of database system or any other system supporting their usage. These users interact with the system via the interface (menu- or- form driven) provided by DBMS. For example: Users of Automatic Teller machines (ATM) falls under this category.

Database Administrators (DBA): The database administrator is the person who implements the strategic and policy decisions regarding the data of the enterprise. Thus DBA is responsible for the overall control of the system at a technical level. Detailed responsibilities of DBA will be discussed in Lesson 3.

Summary:

Data is defined as a collection of meaningful facts which can be stored and processed by computer or humans. **Information** is processed form of data which helps us in making decisions. **Traditional file processing system** has for each application a separate master file and its own set of personal files. **A record** is a collection of related fields that are treated as a single unit. Further all the records are grouped together to form **a file**. All the related files are grouped together to form **a database**. The computer-based simple file oriented approach to information processing started suffering from number of limitations like data redundancy, data inconsistency, difficulty in sharing data, concurrency problems etc. In order to overcome the limitations of the traditional file processing system, the concept of

Database Systems was introduced. A database system is basically a computerized record keeping system i.e. it is a computerized system whose overall purpose is to maintain information and to make the information available on demand. A database system comprises of four major components namely, data, hardware, software, and users.

Self Understanding:

Q1. Explain the limitations of the Traditional file Processing System?

Q2. What is a database and database System?

Q3. Explain the Components of a database system?

Q4. Discuss the categories of users in a database system.

Q5. Define DBMS.

Q6. Define file, record and field.

DATABASE MANAGEMENT SYSTEM-I

Structure:

Introduction:

In this lesson, we first provide the formal definitions of Database Management System. Then we define the Characteristics of DBMS and finally we will explain the advantages of DBMS over traditional file processing system.

The Database Management System (DBMS):

A Database Management System is the software system that allows users to define, create and maintain a database and provides controlled access to the data.

It is basically a collection of programs that enables users to store, modify and extract information from a database as per the requirements. DBMS is an intermediate layer between programs and the data. Programs range from small systems that run on personal computers to huge systems that run on mainframes. Some of common examples of database applications are as follows:

1. Billing System at Super Stores
2. Patient Management System
3. Student Record Management System
4. Computerized Account Department at Educational Institutes
5. Computerized Flight Reservation System

The DBMS relieves the user from knowing how data is stored physically and the complex algorithms used for performing operations on the database.

Conceptually, what happens is the following:

1. A user issues an access request, using some particular data sublanguage (DDL, DML, and DCL using SQL).
2. The DBMS intercepts that request and analyzes it.
3. The DBMS inspects, in turn, the external schema for that user, the corresponding external/conceptual mapping, the conceptual schema, the conceptual/internal mapping, and the storage structure definition.
4. The DBMS executes the necessary operations on the stored database.

In other words, DBMS is a collection of programs that enables you to store, modify, and extract information from a database. There are many different types of DBMSs, ranging from small systems that run on personal computers to huge systems that run on mainframes. The following are examples of database applications:

- computerized library systems automated
- teller machines flight reservation systems
- computerized parts
- inventory systems

From a technical standpoint, DBMSs can differ widely. The terms relational, network, flat, and hierarchical all refer to the way a DBMS organizes information internally. The internal organization can affect how quickly and flexibly you can extract information.

Requests for information from a database are made in the form of a query, which is a stylized question. For example, the query

SELECT ALL WHERE NAME = “SMITH” AND AGE > 35

requests all records in which the NAME field is SMITH and the AGE field is greater than 35. The set of rules for constructing queries is known as a query language. Different DBMSs support different query languages, although there is a semi-standardized query language called SQL (structured query language). Sophisticated languages for managing database systems are called fourth-generation languages, or 4GLs for short.

The information from a database can be presented in a variety of formats. Most DBMSs include a report writer program that enables you to output data in the form of a report. Many DBMSs also include a graphics component that enables you to output information in the form of graphs and charts.

Characteristics of DBMS:

Following are the characteristics that a DBMS must include:

1. DBMS must be able to accept data definitions (external schemas, the conceptual schemas, and all associated mappings) in source form and convert them to the appropriate object form. In other words, the DBMS include language processor components for each of the various data definitions languages (DDLs). The DBMS must also “understand” the DDEL definitions, in the sense that, for example, it “understands” the EMPLOYEE external records include a SALARY field; it must then be able to use this knowledge in interpreting and responding to user requests.

2. The DBMS must be able to handle requests from the user to retrieve, update, or delete existing data in the database, or to add new data to the database. In other words, the DBMS must include a data manipulation language (DML) processor component.
3. The DBMS must monitor user requests and reject any attempt to violate the security and integrity rules defined by the DBA.
4. The DBMS must enforce certain recovery and concurrency controls.
5. The DBMS must provide a data dictionary function. The data dictionary can be regarded as a database in its own right. The dictionary contains “data about the data” i.e. definitions of other objects in the system – rather than just “raw data”. In particular, all the various schemas and mappings will physically be stored, in both source and object form, in the dictionary. The dictionary might even be integrated into the database it defines, and thus include its own definition. It should certainly be possible to query the dictionary just like any other database, so that, for example, it is possible to tell which program and/ or user are likely to be affected by some proposed change to the system.
6. DBMS should perform all of the functions as efficiently as possible.
7. Data should be stored with minimum redundancy to ensure consistency in the data across different applications.

Advantages of DBMS over Traditional File Processing System:

1. Redundancy can be reduced: In Traditional File Processing System, each application has its own private files, which cannot be shared between multiple applications. This can often lead to considerable redundancy in the stored data, which results in wastage of storage space. By having centralized database most of this can be avoided. It is not possible to eliminate all the redundancy. Sometimes there are sound business and technical reasons for maintaining multiple copies of the same data. In a database system, however this redundancy can be controlled.

For example: In case of college database, there is some common data of the student which has to be mentioned in each application, like Rollno, Name, Class, Phone no, Address etc. This will cause the problem of redundancy which results in wastage of storage space and is difficult to maintain, but in case of centralized database, data can be shared by number of applications and the whole college can maintain its computerized data with the database containing Roll no, Name, Class, Father Name, Address, Phone-No, Date of birth. This data which was stored repeatedly in file system in each application, need not be stored repeatedly, because every other application can access this information by joining of relations on the basis of common column i.e. Roll no. Suppose any user of Library system needs the Name and Address of any particular student, then by joining of Library and General Office relations on the basis of column Roll no, he\she can easily retrieve this information. Thus we can say that centralized system of DBMS reduces the redundancy of data to a great extent but cannot eliminate the redundancy because Roll no is still repeated in all the relations.

2. Integrity can be enforced

Integrity of data means that data in the database is always accurate, that is incorrect information cannot be stored in database. In order to maintain the integrity of data, some integrity constraints are enforced on the database. A DBMS should provide capabilities for defining and enforcing the constraints.

For Example: Let us consider the case of college database and suppose that college is having only BA,BCA, BIT, BBA and BCOM classes. But if a user enters the class -CA, then this incorrect information must not be stored in the database and must be prompted that this is an invalid data entry . In order to enforce this, the integrity constraints must be applied to the class attribute of the student entity. But, in case of file system this constraint must be enforced on all the applications separately (because all applications have class field).

In case of DBMS, this integrity constraint is applied only once on the class field of the General Office (because class field appears only once in the whole database), and all other applications will get the class information about the student from the General Office table so the integrity constraint is applied to the whole database. Now, we can say that integrity constraint is applied to the whole database. Therefore, we can say that integrity constraint can be easily enforced in centralized DBMS system as compared to file system.

3. Inconsistency can be avoided

When the same data is duplicated and changes are made at one site, which is not propagated to the other site, it gives rise to inconsistency and the two entries regarding the same data will not agree. At such times the data is said to be inconsistent. So if the redundancy is removed chances of having inconsistent data is also removed.

Let us again consider the college system and suppose that in case of General Office file it is indicated that Roll-no 5 lives in Amritsar but in library file it is indicated that Roll-Number 5 lives in Jalandhar. Then this is a state at which the two entries of the same object do not agree with each other (that is one is updated and other is not). At such time the database is said to be inconsistent.

An inconsistent database is capable of supplying incorrect or conflicting information. So, there should be no inconsistency in database. It can be clearly shown that inconsistency can be avoided in centralized system very well as compared to file system.

Let us consider again the example of college system and suppose that RollNo5 is shifted from Amritsar to Jalandhar. The address information of Roll Number 5 must be updated, where ever Roll number and address occurs in the system. In case of file system, the information must be updated separately in each application, but if we make updation only at three places and forget to make updation at fourth application, the whole system shows the inconsistent results about Roll Number 5. In case of DBMS, Roll number and address occur together only once in General-Office table. So, it needs single updation and then all other applications retrieve the address information from General-Office which is updated. So, all applications will get current and latest information by providing single update operation and this single update operation is propagated to the whole database all
other application automatically. This property is called as Propagation of Update.

We can say that the redundancy of data greatly affects the consistency of data. If redundancy is less, it is easy to maintain consistency of data. Thus DBMS system can avoid inconsistency to a great extent.

4. Data can be shared

As explained earlier, the data about Name, Class, Father-name etc. of General-Office is shared by multiple applications in DBMS as compared to file system. So now application s can be developed to operate against the same stored data. The applications may be developed without having to create any new stored files.

5. Standards can be enforced

Since DBMS is a central system, so standards can be enforced easily may be at Company level, Department level, National level or International level. The file system is an independent system so standards cannot be easily enforced on multiple independent applications.

6. Restricting unauthorized access

When multiple users share a database, it is likely that some users will not be authorized to access all information in the database. For example account office data is often considered confidential, and hence only authorized persons are allowed to access such data. In addition, some users may be permitted only to retrieve data, whereas other is allowed both to retrieve and to update. Hence the type of access operation retrieval or update must also be controlled. Typically, users or user groups are given account numbers protected by passwords, which they can use to gain access to the database. DBMS should provide a security and authorization subsystem, which the DBA uses to create accounts and to specify account restrictions. The DBMS should then enforce these restrictions automatically.

7. Solving enterprise requirement than individual requirement

Since many types of users with varying level of technical knowledge use a database, a DBMS should provide a virility of user interface. The overall requirements of the enterprise are more important than the individual user requirements. So the DBA can structure the database system to provide an overall service that is “best for the enterprise”.

For example: A representation can be chosen for the data in storage that gives fast access for the most important application but at the cost of poor performance in some other application. But the file system favors the individual requirements than the enterprise requirements.

8. **Providing Backup and Recovery**

A DBMS must provide facilities for recovering from hardware or software failures. The backup and recovery subsystem of the DBMS is responsible for recovery. For example, if the computer system fails in the middle of a complex update program, the recovery subsystem is responsible for making sure that the database is restored to the state it was in before the program started executing.

Disadvantages of DBMS

The disadvantages of the database approach are summarized as follows:

1. **Complexity**

The provision of the functionality that is expected of a good DBMS makes the DBMS an extremely complex piece of software. Database designers, developers, database administrators and end-users must understand this functionality to take full advantage of it. Failure to understand the system can lead to bad design decisions, which can have serious consequences for an organization.

2. **Size**

The complexity and breadth of functionality makes the DBMS an extremely large piece of software, occupying many megabytes of disk space and requiring substantial amounts of memory to run efficiently.

3. **Performance**

Typically, a File Based system is written for a specific application, such as invoicing. As result, performance is generally very good. However, the DBMS is written to be more general, to cater for many applications rather than just one. The effect is that some applications may not run as fast as they used to.

4. **Higher impact of a failure**

The centralization of resources increases the vulnerability of the system. Since all users and applications rely on the availability of the DBMS, the failure of any component can bring operations to a halt.

5. **Cost of DBMS**

The cost of DBMS varies significantly, depending on the environment and functionality provided. There is also the recurrent annual maintenance cost.

6. **Additional Hardware costs**

The disk storage requirement for the DBMS and the database may necessitate the purchase of additional storage space. Furthermore, to achieve the required performance it may be necessary to purchase a larger machine, perhaps even a machine dedicated to running the DBMS. The procurement of additional hardware results in further expenditure.

7. **Cost of Conversion**

In some situations, the cost of the DBMS and extra hardware may be insignificant compared with the cost of converting existing applications run on the new DBMS and hardware. This cost also includes the cost of training staff to use these new systems and possibly the employment of specialist staff to help with conversion and running of the system. This cost is one of the main reasons why some organizations feel tied to their current systems and cannot switch to modern database technology.

Summary:

DBMS is basically a collection of programs that enables users to store, modify, and extract information from a database as per the requirements. DBMS is an intermediate layer between programs and the data. Programs range from small systems that run on personal computers to huge systems that runs on mainframes. DBMS must be able to accept data definitions (external schemas, the conceptual schemas, and all associated mappings) in source form and convert them to the appropriate object form. The DBMS must include a data manipulation language (DML) processor component. It must enforce certain recovery and concurrency controls. It must provide a data dictionary function.

Self Understanding:

Q1. Define DBMS?

Q2. What are the advantages of DBMS over traditional File Processing System

Q3. Explain the Characteristics of DBMS.

Q4. What are disadvantages of DBMS?

DATABASE MANAGEMENT SYSTEM-II

Structure:

Introduction:

The database administrator is the person who implements the strategic and policy decisions regarding the data of the enterprise. Thus DBA is responsible for the overall control of the system at a technical level. The overall structure of the database is called database schema i.e. it is the description of a database, which is specified during the database design and is not expected to change frequently. The actual data in a database may change quite frequently. The data in the database at a particular moment of time is called a database state or snapshot. It is also called the current set of occurrences or instances in the database. In the given database state, each schema construct has its own current set of instances.

Implications of the Database Approach

Potential for Enforcing Standards. The database approach permits the DBA to define and enforce standards among database users in a large organization. This facilitates communication and cooperation among various departments, projects, and users within the organization. Standards can be defined for names and formats of data elements, display formats, report structures, terminology, and so on. The DBA can enforce standards in a centralized database environment more easily than in an environment where each user group has control of its own files and software.

Reduced Application Development Time. A prime selling feature of the database approach is that developing a new application such as the retrieval of certain data from the database for printing a new report-takes very little time. Designing and implementing a new database from scratch may take more time than writing a single specialized file application . However, once a database is up and running, substantially less time is generally required to create new applications using DBMS facilities. Development time using a DBMS is estimated to be one-sixth to one-fourth of that for a traditional file system.

Flexibility: It may be necessary to change the structure of a database as requirements change, For example, a new user group may emerge that needs information not currently in the database. In response, it may be necessary to add a file to the database or to extend the data elements in an existing file. Modern DBMSs allow certain types of changes to the structure of the database without affecting the stored data and the existing application programs.

Availability of Up-to-Date Information: A DBMS makes the database available to all users. As soon as one user's update is applied to the database, all other users can immediate see this update. The availability of up-to-date information is essential for many transaction-processing applications, such as reservation systems of a DBMS.

Economies of Scale: The DBMS approach permits consolidation of data and applications, thus reducing the amount of wasteful overlap between activities of data-processing personnel in different projects or departments. This enables the whole organization to invest in more powerful processors, storage devices or communication gear, rather than having each department purchase its own (weaker)equipment. This reduces overall costs of operation and management.

User of Database:

There are a number of users who can access or retrieve data on demand using application programs and interfaces provided by DBMS. Each type of user needs different software capabilities. Following are the categories of the users:

1. Application Programmers
2. End Users
3. Database Administrator (DBA)

Application Programmers: Users who are responsible for writing application programs that use database. These programs operate on data for retrieving existing information, inserting new information, deleting or changing existing information.

End Users: End Users are those who need not know about the presence of database system or any other system supporting their usage. These users interact with the system via the interface (menu- or- form driven) provided by DBMS. For example: Users of Automatic Teller machines (ATM) fall under this category.

Database Administrators (DBA): The database administrator is the person who implements the strategic and policy decisions regarding the data of the enterprise. Thus DBA is responsible for the overall control of the system at a technical level. Detailed responsibilities of DBA will be discussed in the following section.

DBA and its responsibilities

The data administrator is the person who makes the strategic and policy decisions regarding the data of the enterprise. Database Designer or Data Administration is the person responsible for indentifying the data to be stored in the data base and for choosing appropriaate structures to represent and store this data. It is the responsibility of the data base designer to communicate with all the prospective users in order to understand their requirements and to create a design that meets these requirements. And the Data Base Administrator (DBA) is the person who provides the necessary technical support for implementing the decisions taken by Data Administrator. Thus, the DBA is responsible for the overall control of the system at a technical level. In general, those functions will include the following.

Defining the conceptual schema

It is the data administrator's job to decide exactly what information is to be held in the database-in other words, to identify the entities of interest to the enterprise and to identify the information to be recorded about those entities. This process is usually referred to as logical-sometimes conceptual-database design. Once the data administrator has thus decided the content of the database at an abstract level, the DBA will then create the corresponding conceptual schema, using the conceptual DDL. The object (compiled) form of that schema will be used by the DBMS in responding to access requests. The source (uncompiled) form will act as a reference document for the users of the system.

Defining the internal schema

The DBA must also decide how the data is to be represented in the stored database. This process is usually referred to as physical database design. Having done the physical design, the DBA must then create the corresponding storage structure definition (i.e., the internal schema), using the internal DDL. In addition, he or she must also define the associated mapping between the internal and conceptual schemas. In practice either the conceptual DDL, or the internal DDL-most likely the former-will probably include the means for defining that mapping, but the two functions (creating the schema, defining the mapping) should be clearly separable. Like the conceptual schema, the internal schema and corresponding mapping will exist in both source and object form.

Liaising with users

It is the business of the DBA to liaise with users, to ensure that the data they require is available, and to write (or help the users write) the necessary external schemas, using the applicable external DDL. In addition, the mapping between any given external schema and the conceptual schema must also be defined. In practice, the external DDL will probably include the means for specifying that mapping should be clearly separable . Each external schema and corresponding mapping will exist in both source and object form.

Other aspects of the user liaison function include consulting on application design, providing technical education, assisting with problem determination and resolution, and similar system-related professional services.

Defining security and integrity procedures

The conceptual DDL should include facilities for specifying security and integrity rules that can be regarded as part of the conceptual schema.

Defining backup and recovery procedures

Once an enterprise is committed to a database system, it becomes critically dependent on the successful operation of that system. In the event of damage to any portion of the database-caused by human error, say, or a failure in the hardware or supporting operating system-it is essential to be able to repair the data concerned with the minimum of delay and with as little effect as possible on the rest of the system. For example, the availability of data that has not been damaged should ideally not be affected. The DBA must define and implement an appropriate recovery scheme, involving, e.g., periodic unloading "dumping" of the database to backup storage, and procedures for reloading the database when necessary from the most recent dump.

Incidentally, the foregoing discussion provides one reason why it might be a good idea to spread the total data collection across several databases, instead of keeping it all in one place, the individual database might very well form the unit for dump and reload purposes. Nevertheless, we will continue to talk as if there were in fact just a single database, for simplicity.

Monitoring performance and responding to changing requirements

The DBA is responsible for so organizing the system as to get the performance that is "best for the enterprise" and for making the appropriate adjustments as requirements change. For example, it might be necessary to reorganize the stored database on a periodic basis to ensure that performance levels remain acceptable. As already mentioned, any change to the physical storage (internal) level of the system must be accompanied by a corresponding change to the definition of the mapping from the conceptual level, so that the conceptual schema can remain constant.

Of course, the foregoing is not an exhaustive list it is merely intended to give some idea Of the extent and nature of the DBA's responsibilities.

Database Schema and Instance:

The overall structure of the database is called database schema i.e it is the description of a database, which is specified during the database design and is not expected to change frequently. The diagram that displays the structure of each record type but not the actual instance of records is called a schema diagram. A schema diagram displays only some aspects of a schema, such as names of the records types and data items, and some types of constraints. For example:

University_Dept:

Dept_Id Dept_name Date_Of_Estb Head

Student_Record

Stu_Id Stu_Name D_O_Adm Class Session

Employee_Record

Emp_ID Emp_Name Emp_Add Dept D_O_J

The actual data in a database may change quite frequently. For Example: the database shown above changes every time we add a student. The data in the database at a particular moment of time is called a database state or snapshot. It is also called the current set of occurrences or instances in the database. In the given database state, each schema construct has its own current set of instances. For example, the Student_Record construct will contain the set of individual student records as its instances. Every time we insert or delete a record, or change the value of a data item in a record, we change one state of the database into another state.

The distinction between database schema and database state is very important. When we define a new database, we specify its database schema only to the DBMS. At this point, the corresponding database state is the empty state with no data. We get the initial state of the database when the database state is first populated or loaded with the initial data. From then on, every time an update operation is applied to the database, we get another database state. At any point in time, the database has a current state. The DBMS is partly responsible for ensuring that every state of the database has a valid state i.e. a state that satisfies the structure and constraints specified in the schema. Hence, specifying a correct schema to the DBMS is extremely important, and the schema must be designed with the utmost care. The DBMS stores the description of the schema constructs and constraints – also called the Meta-data- in the DBMS catalog so that the DBMS software can refer to the schema whenever it needs to. The schema is sometimes called the intension and database state is called the extension of the schema. As we have already mentioned, the database schema do not change frequently, but in case there is a need to add a data item to the record type of database, this is known as schema evolution. For example – We need to add data item Emp_DOB to the record type Employee_Record.

Summary:

There are a number of users who can access or retrieve data on demand using application programs and interfaces provided by DBMS. Each type of user needs different software capabilities. Following are the categories of the users: Application Programmers, End Users, Database Administrator (DBA). The data administrator is the person who makes the strategic and policy decisions regarding the data of the enterprise. And the database administrator (DBA) is the person who provides the necessary technical support for implementing those decisions. Thus, the DBA is responsible for the overall control of the system at a technical level. In general, the functions of DBA are Defining the conceptual schema, Defining the internal schema, Liaising with users, Defining security and integrity procedures, Monitoring performance and responding to changing requirements and lot more.

Self Understanding:

Q1. Explain the Implications of Database approach.

Q2. Explain the users of database.

Q3. Define DBA and explain its responsibilities.

Q4. What do you mean by Database Schema ands instance?

Q5. Differentiate between Data Administrator and Data Base Administrator.

DBMS ARCHITECTURE

Structure:

Introduction:

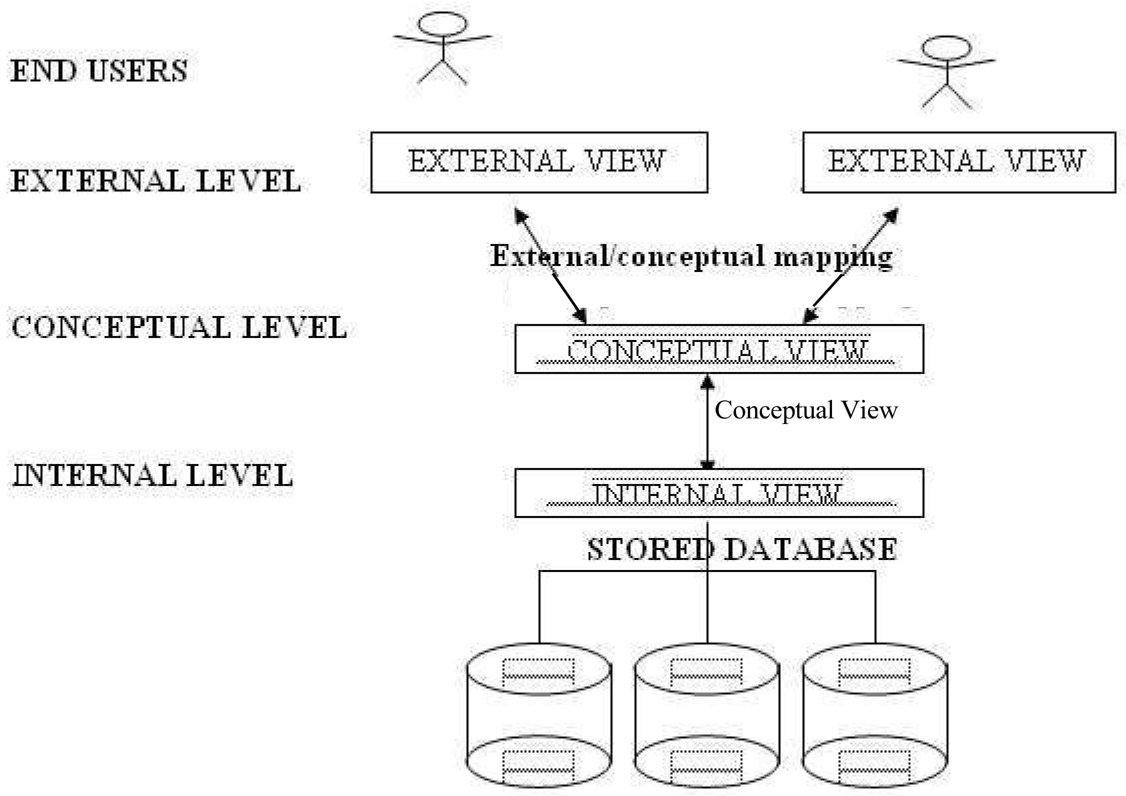
The Three Levels of the Architecture of DBMS is also known as ANSI/SPARC Model. The goal of this three level architecture is to separate the user applications and the physical database. The ANSI/SPARC model is divided into three levels, known as the internal, conceptual, and external levels. In a DBMS based on three-schema architecture, each user group refers to its own external schema. Hence, the DBMS must transform a request specified on an external schema into a request against the conceptual schema, and then into a request on the internal schema for processing over the stored database. The process of transforming requests and results between levels are called mappings. There are two levels of mappings in the architecture – Conceptual/Internal mapping and External/Conceptual mapping. The three level architecture is then used to explain the concept of data independence, which can be defined as the capacity to change the schema at one level of a database system without having to change the schema at the next higher level. There are two types of data independence – Logical data Independence and Physical data independence.

Three-Level Architecture of DBMS or ANSI/SPARC Model:

Several different frameworks have been suggested over the last several years. For example, a framework may be developed based on the functions that the various components of a DBMS must provide to its users. It may also be based on different views of data that are possible within a DBMS. Commonly used views of data approach is the three-level architecture suggested by ANSI/SPARC (American National Standards Institute/Standards Planning and Requirements Committee). ANSI/SPARC produced an interim report in 1972 followed by a final report in 1977. The reports proposed an architectural framework for databases. A large number of commercial systems and research database models fit into this framework. The three levels of the architecture are three different views of the data:

1. External - individual user view
2. Conceptual - community user view
3. Internal - physical or storage view

The three level database architecture allows a clear separation of the information meaning (conceptual view) from the external data representation and from the physical data structure layout. A database system that is able to separate the three different views of data is likely to be flexible and adaptable. This flexibility and adaptability is data independence



The view at each of these levels is described by a scheme. A Scheme is an outline or a plan that describes the records and relationships existing in the view. The word scheme, which means a systematic plan for attaining some goal, is used interchangeably in the database literature for the plural instead of schema. The word schema is used in the database literature for the plural instead of schemata, the grammatically correct word. The scheme also describes the way in which entities at one level of abstraction can be mapped to the next level.

External or User View

The external or user view is at the highest level of database abstraction where only those portions of the database that are of concern to a user or application program are included. Any number of user views (some of which may be identical) may exist for a given global or conceptual view.

Each external view is described by means of a scheme called an external schema. The external schema consists of the definition of the logical records and the relationships in the external view. The external schema also contains the method of deriving the objects in the external view from the objects in the conceptual view. The objects include entities, attributes, and relationships. (The terms view, scheme, and schema are sometimes used interchangeably when there is no confusion as to what is implied.)

Conceptual or Global View

At this level of database abstraction all the database entities and the relationships among them are included. One conceptual view represents the entire database. This conceptual view is defined by the conceptual schema. It describes all the records and relationships included in the conceptual view and, therefore, in the database. There is only one conceptual schema per database. This schema also contains the method of deriving the objects in the conceptual view from the objects in the internal view.

The description of data at this level is in a format independent of its physical representation. It also includes features that specify the checks to retain data consistency and integrity.

Internal View

It is at the lowest level of abstraction, closest to the physical storage method used. It indicates how the data will be stored and describes the data structures and access methods to be used by the database. The internal view is expressed by the internal schema, which contains the definition of the stored record, the method of representing the data fields, and the access aids used. At least the following aspects are considered at this level:

1. Storage allocation e.g. B-trees, hashing etc.
2. Access paths e.g. specification of primary and secondary keys, indexes and pointers and sequencing.
3. Miscellaneous e.g. data compression and encryption techniques, optimization of the internal structures.

Efficiency considerations are the most important at this level and the data

structures are chosen to provide an efficient database. The internal view does not deal with the physical devices directly. Instead it views a physical device as a collection of physical pages and allocates space in terms of logical pages.

The separation of the conceptual view from the internal view enables us to provide a logical description of the database without the need to specify physical structures. This is often called physical data independence. Separating the external views from the conceptual view enables us to change the conceptual view without affecting the external views. This separation is sometimes called logical data independence.

Assuming the three level view of the database, a number of mappings are needed to enable the users working with one of the external views. For example, the payroll office may have an external view of the database that consists of the following information only:

1. Staff number, name and address.
2. Staff tax information e.g. number of dependents.
3. Staff bank information where salary is deposited.
4. Staff employment status, salary level, leave information etc.

The conceptual view of the database may contain academic staff, general staff, casual staff etc. It will be needed to create a mapping where all the staff in the different categories are combined into one category for the payroll office. The conceptual view would include information about each staff's position, the date employment started, full-time or part-time, etc. This will need to be mapped to the salary level for the salary office. Also, if there is some change in the conceptual view, the external view can stay the same if the mapping is changed.

Data Independence:

For simplifying the concept of data independence, let us explain the opposite of data independence. Applications implemented on older systems tend to be data dependent. Data dependent means the way in which the data is organized in secondary storage, and the technique for accessing it, are both dictated by the requirements of the application under consideration and moreover that knowledge of that data organization and access technique is built into the application logic and code but on the other hand, in a database system, data dependence is extremely undesirable, at least for following two reasons:-

1. Different applications will need different views of the same data.

2. The DBA must have the freedom to change the storage structure or access technique in response to changing requirements, without having to modify existing applications.

Thus, Data Independence can be defined as the capacity to change the schema at one level of a database system without having to change the schema at the next higher level. There are two types of data independences:-

1. Logical data Independence.
2. Physical data Independence.

Difference between Logical Data Independence and Physical Data Independence:

1. Logical data independence is usually required for changing the conceptual schema without having to change the external schema or application programs while physical data independence is required for changing the internal schema without having to change the external schema or conceptual schema
2. Logical data independence specifies that the application programs need not to be changed if new fields are added to, or deleted from a database. It is concerned with changing the logical structure without having to change the application programs while physical data independence is concerned with changing the internal schema i.e. changing the physical storage of the data without having to change the corresponding external schema.
3. Logical data Independence is usually more difficult to achieve than physical data independence because the programs required to retrieve the data are heavily dependent on the logical structure of data while physical data independence is comparatively easy to achieve than the logical data Independence.
4. Logical data independence is concerned with changing the structure of the data or changing the data definition while physical data independence helps to migrate the files from one kind of storage device to another.

Advantages of Data Independence:

1. It provides the capacity to change the schema at one level without having the need to change the schema at the next higher level.
2. It also hides the implementation details or hardware level details from its users so that the users can concentrate on the program only.
3. The various changes made at different levels are absorbed by mapping at different level.

4, Physical data independence allows the changes to be made in storage of data without affecting application programs.

5. Due to logical data independence, there is no effect to application programs even if new fields are added or old records are deleted from the existing data.
6. Physical data independence allows the files to migrate from one kind of storage device to another.
7. Logical data independence does not require the external schema to be changed.
8. Physical data independence does not require the internal schema to be changed.

Mapping between Different Levels:

The conceptual database is the model or abstraction of the objects of concern to the database. Thus, the conceptual record of above Figure is the conceptual database and represents the abstraction of all the applications involving the entity set EMPLOYEE. The view is the subset of the objects modeled in the conceptual database that is used by an application. These could be any number of views of a conceptual database. A view can be used to limit the portion of the database that is known and accessible to a given application.

Two mappings are required in a database system with three different views. A mapping between the external and conceptual views gives the correspondence among the records and the relationship of the external and conceptual views. The external view is an abstraction of the conceptual view, which in its turn is an abstraction of the internal view. The user of the external view sees and manipulates a record corresponding to the external view. There is a mapping from a particular logical record in the external view to one (or more) conceptual record(s) in the conceptual view. A number of differences could exist between the two. Names of the fields and records, for instance, may be different. A number of conceptual fields can be combined into a single logical field, for example, Last_Name and First_Name at the conceptual level but Name at the logical level. A given logical record could be derived from a number of conceptual records.

Similarly, there is a mapping from a conceptual record to an internal one. An internal record is a record at the internal level, not necessarily a stored record on a physical storage device. The internal record of above figure may be split up into two or more physical records. The physical database is the data that is stored on secondary storage devices. It is made up of records with certain data structures and organized in files. Consequently, there is an additional mapping from the internal record to one or more stored records on secondary storage devices. This may have been implemented using some form of nonlinear addressing. The internal record is assumed to be linearly addressed. However, this complexity is managed by the DBMS and the user need not be aware of its presence nor be concerned with it.

Mapping between the conceptual and the internal level specifies the method of deriving the conceptual record from the physical database. Again, differences similar to those that exist between external and conceptual views could exist between the conceptual and internal views. Such differences are indicated and resolved in the mapping.

Differences that could exist, besides the difference in names, include the following;

- Representation of numeric values could be different in the two views. One view could consider a field to be decimal, whereas the other view may regard the field as binary. A two-way transformation between such values can be easily incorporated in the mapping. If, however, the values are stored in a binary format, the range of values may be limited by the underlying hardware.
- Representation of string data can be considered by the two views to be coded differently. One view may perceive the string data to be in ASCII code, the other view may consider the data to be in EBCDIC code. Again, two-way transformation can be provided.
- The value for a field in one view could be computed from the values in one or more fields of the other view. For example, the external view may use a field containing a person's age, whereas the conceptual view contains the date of birth. The age value could be derived from the date of birth by using a date function available from the operating system. Another example of a

computed field would be where an external view requires the value of the hours worked during a week in a field, whereas the conceptual view contains fields representing the hours worked each day of the week. The former can be derived from the later by simple addition. These two examples of transformation between the external and conceptual views are not bidirectional. One cannot uniquely reflect a change in the total hours worked during a week to hours worked during each day of the week. Therefore, a user's attempt to modify the corresponding external fields will not be allowed by the DBMS.

Such mapping between the conceptual and internal levels is a correspondence that indicates how each conceptual record is to be stored and the characteristics and size of each field of the record. Changing the storage structure of the record involves changing the conceptual view to internal view mapping so that the conceptual view does not require any alteration.

The conceptual view can assume that the database contains a sequence of records for each conceptual record type. These records could be accessed sequentially or randomly. The actual storage could have been done to optimize performance. A conceptual record may be split into two records, with the less frequently used record (part of the original record) on a slower storage device and the more frequently used, record, on a faster device. The stored record could be in a physical sequence, or one or more indices may be implemented for faster access to record occurrences by the index fields. Pointers may exist in the physical records to access the next record occurrence in various orders. These structures are hidden from the conceptual view by the mapping between the two.

Summary:

The three level architecture is divided into three levels: the external level, the conceptual level, and the internal level. The external or user view is at the highest level of database abstraction where only those portions of the database of concern to a user or application program are included. One conceptual view represents the entire database. The conceptual view is defined by the conceptual schema. It describes all the records and relationships included in the conceptual view and, therefore, in the database. The internal view indicates how the data will be scored and describes the data structures and access methods to be used by the database. It is expressed by the internal schema, which contains the definition of the stored record, the method of representing the data fields, and the access aids used. The DBMS provides users with a method of abstracting their data requirements and removes the drudgery of specifying the details of the storage and maintenance of data. The DBMS insulates users from changes that occur in the database. Two levels of database independence are provided by the system. Physical independence allows changes in the physical level of data storage without affecting the conceptual view. Logical independence allows the conceptual view to be changed without affecting the external view.

Self Understanding:

Q1. Define Data Independence.

Q2. Explain the difference between Logical data Independence and Physical data Indepence.

Q3. Explain ANSI/SPARC Model or The Three Level Architecture or DBMS.

Q4. Explain the mapping between different levels of DBMS Architecture.

Q5. What do you mean by data abstraction?

DATABASE LANGUAGES

Structure:

Introduction:

There is variety of users supported by a DBMS. Thus, DBMS must provide appropriate languages and interfaces for each category of users. Once the design of a database is completed and a DBMS is chosen to implement the database, the first order of the day is to specify conceptual and internal schema for the database and any mappings between the two. In many DBMSs where no strict separation of levels is maintained, one language, called the Data Definition Language (DDL), is used by the DBA and by database designers to define both schemas. Once the database schemas are compiled and the database is populated with data, users must have some means to manipulate the database. Typical manipulations include retrieval, insertion, deletion and modification of the data. The DBMS provides a data manipulation language (DML) for these purposes. DBMS also provides a language named as Data Control Language (DCL) for granting and revoking the privileges for using the database.

A Key is a property of the entity set, rather than of the individual entities. Any two individual entities in the set are prohibited from having the same value of the key attributes at the same time. There are different type of keys – Super, Candidate, Primary, Unique and Foreign. Keys help in maintaining the data integrity and consistency.

Objectives

After completing this lesson, you will be able to:

- What is DBMS Language?
  DDL, DML, DCL
- Keys

Database Languages

There is variety of users supported by a DBMS. Thus, DBMS must provide appropriate languages and interfaces for each category of users.

Following are the languages provided for various categories of users for different operations:

1. Data Definition Language (DDL)
2. Data Manipulation Language (DML)
3. Data Control Language (DCL)

Data Definition Language

Once the design of a database is completed and a DBMS is chosen to implement the database, the first order of the day is to specify conceptual and internal schema for the database and any mappings between the two. In many DBMSs where no strict separation of levels is maintained, one language, called the Data Definition Language (DDL), is used by the DBA and by database designers to define both schemas. The DBMS will have a DDL compiler whose function is to process DDL statements in order to identify descriptions of the schema constructs and to store the schema description in the DBMS catalog. DDL statements are used to define, alter, or drop database objects. The following table gives an overview about the usage of DDL statements;

S.No.	Purpose	SQL DDL Statement
1.	Create database schema objects	Create
2.	Alter database schema objects	Alter
3.	Delete database Schema objects	Drop
4.	Rename schema objects	Rename

Data Manipulation Language

Once the database schemas are compiled and the database is populated with data, users must have some means to manipulate the database. Typical manipulations include retrieval, insertion, deletion and modification of the data. The DBMS provides a Data Manipulation Language (DML) for these purposes. There are two main types of DMLs - A high-level or nonprocedural DML and a low-level or procedural DML. A high-level or nonprocedural DML can be used on its own to specify complex database operations in a concise manner. Many DBMSs allow high-level DML statements either to be entered interactively from a terminal or to be embedded in a general-purpose programming language. In the latter case, DML statements must be identified within the program so that they can be extracted by a pre-compiler and processed by the DBMS. A low-level or procedural DML must be embedded in a general-purpose programming language. This type of DML typically retrieves individual records or objects from the database and processes each separately. Hence, it needs to use programming language constructs, such as looping, to retrieve and process each record from a set of records. Low-level DMLs are also called record-at-a-time DMLs because of this property. High-level DMLs, such as SQL, can specify and retrieve many records in asingle DML statement and are hence called set-at-a-time or set-oriented DMLs. A query in a high-level DML often specifies which data to retrieve rather than how to retrieve it; hence, such languages are also called declarative.

The following are basic SQL commands that used as DML statements:

S.No.	Purpose	SQL statement
1.	Retrieve data from one or more tables	Select
2.	Add new rows in a table	Insert
3.	Remove rows from a table	Delete
4.	Change data in rows in a table	Update

Data Control Language

DBMS also provides a language named as Data Control Language (DCL) for granting and revoking the privileges for using the database. A privilege can either be granted to a user with the help of GRANT statement. The privileges assigned can be Select, Alter, Delete, Execute, Insert, Index etc. In addition to granting of privileges, you can also revoke it by using Revoke command. Following are the SQL Statements that can be used as DCL:

S.No.	Purpose	SQL statement
1.	Granting the privileges	Grant
2.	Taking away the privileges	Revoke
3.	Adding a comment to the data dictionary	Comment

Keys

Before we start discussing the concept of Key and various types of keys - Consider the Table PGDCA_Students_Details (Roll_No, SSNo, Student_Name, Father_name, Address, Phone_No). Here in this table, PGDCA_Students_Details is the name of the table and (Roll_No, SSNo, Student_Name, Father_name, Address, Phone_No) are the names of the attributes (fields) of the table. It is storing the details of the students of the PGDCA class.

Table is also called entity set and the rows in the table are also called entities.

A key is a single attribute or combination of two or more attributes of an entity set that is used to identify one or more instances of the set. The attribute Roll_No uniquely identifies an instance of the entity set PGDCA_Students_Details. Thus, a key is a property of the entity set, rather than of the individual entities. Any two individual entities in the set are prohibited from having the same value of the key attributes at the same time.

There are various types of keys that are as follows:

1. Super Key
2. Candidate Key
3. Primary Key
4. Alternative Key
5. Unique Key
6. Foreign Key

Details of above keys will be discussed in chapter 6 .

Summary

DBMS must provide appropriate languages for various categories of DBMS users. After database design, a DBMS is chosen to implement the database. Most common DBMS languages are DDL, DML, and DCL. Data Definition Language (DDL) is used by the DBA and by database designers to define database schemas. The DBMS provides a Data Manipulation Language (DML) for manipulating data in database. Data Control Language (DCL) is used for granting and revoking the privileges to the users for using the database.

A key is a single attribute or combination of two or more attributes of an entity set that is used to identify one or more instances of the set uniquely. There is different type of keys – Super, Candidate, Primary, Unique and Foreign. Keys help in maintaining the data integrity and consistency.

Self Understanding

Q1. What do you mean by DBMS Language?

Q2. List the various Database Languages? Explain each of them.

Q3. Write short notes on DDL, DML, DCL.

Q4. What are types of DML? Explain each of them.

Q5. What do you mean by key? List down the various keys available.

DATA MODELS AND KEYS

Structure:

Introduction:

DBMSs have database utilities that help the DBA in managing the database system. Common utilities include loading, backup, file reorganization and performance monitoring. One fundamental characteristic of the database approach is that it provides some level of data abstraction by hiding details of data storage that are not needed by most database users. A data model is a collection of concepts that can be used to describe the structure of a database and that provides the necessary means to achieve the data abstraction. We can categorize the data models into High-level or conceptual data models and Low-level or physical data models. High-level or conceptual data models are those that provide concepts that are close to the way many users perceive data. Entity Relationship model is a popular example of high-level data model. Low-level or physical data models are those that provide concepts that describe the details of how data is stored in the computer. Between the two broad categories of data models is representational or implementation data model which provide concepts that may be understood by end users. Hierarchical, Network and relational are the popular examples of this category of data model. A Key is a property of the entity set, rather than of the individual entities. Any two individual entities in the set are prohibited from having the same value of the key attributes at the same time. There are different type of keys – Super, Candidate, Primary, Unique and Foreign. Keys help in maintaining the data integrity and consistency.

Objectives

After completing this lesson, you will be able to:

- Explain the database utilities
- Define Data Model and various data models
- Define Key
- Explain different type of keys – Super, Candidate, Primary, Unique and Foreign

Database Utilities

DBMSs have database utilities that help the DBA in managing the database system. Common utilities have the following types of functions:

1. **Loading:** A loading utility is used to load existing data files — such as text files or sequential files—into the database. Usually, the current format of the data file and the desired database file structure are specified to the utility,

which then automatically reformats the data and stores it in the database. With the proliferation of DBMSs, transferring data from one DBMS to another is becoming common in many organizations. Some vendors are offering products that generate the appropriate loading programs, given the existing source and target database storage descriptions (internal schemas). Such tools are also called conversion tools.

2. Backup: A backup utility creates a backup copy of the database, usually by dumping the entire database onto tape. The backup copy can be used to restore the database in case of catastrophic failure. Incremental backups are also often used, where only changes since the previous backup are recorded. Incremental backup is more complex but it saves space.
3. File reorganization: This utility can be used to reorganize a database file into a different file organization to improve performance.
4. Performance monitoring: Such a utility monitors database usage and provides statistics to the DBA. The DBA uses the statistics in making decisions such as whether or not to reorganize files to improve performance.

   Other utilities may be available for sorting files, handling data compression, monitoring access by users, and performing other functions.

Data Models:

A data model is a collection of concepts that can be used to describe the structure of a database and that provides the necessary means to achieve the data abstraction. A Data Model is a collection of concepts that can be used to describe the structure of the database including data types, relationship and constraints that should apply on the data.

The main purpose of data modeling is to help in understanding the meaning of the data and to facilitate communication about information requirements. A data model supports communication between the users and database designers.

WHY DATA MODELLING IS IMPORTANT?

The goal of data model is to make sure that all data objects required by the database are completely and accurately represented because the data model uses easily understood notations and natural language which can be reviewed and verified as correct by the end users.

The data model must be detailed enough to be used by the database designers for building the physical database. The information contained in data model will be used to define the relational tables, primary and foreign keys and other information. A poorly designed database will require more time in the long term. Without careful planning you may create a database that omits data required to create reports, produce results that are incorrect or inconsistent and is unable to accommodate changes in the user's requirements. A data model should possess the following characteristics so that the best possible data representation can be obtained:

- Diagrammatic representation of the Data Model
- Simplicity in designing and expressibility to distinguish between data and their relationships. Also it is detailed enough to be used by a database designer to build the database
- Application independent i.e. it could be shared by different applications
- No duplication in representation of data
  Consistency and structure validity is maintained.

It follows a bottom up approach. A basic model, representing entities and relationships is developed first, and then details are added.

We can categorize the data models into three broad categories:

High-level or conceptual data models: High-level or conceptual data models are those that provide concepts that are close to the way many users perceive data. This category is also known as Object based data models. Most of the common data models under this category are:

a. Entity Relationship Model (ER Model): The ER Model is a high level conceptual data model which describes the structure of the database and the associated operations like retrieval and updation on the database. The basic concepts of ER Model include entity type, their attributes and relationships. It also represents certain constraints which are used on the database.

b. Object Oriented Model: The object oriented model is based on collection of objects, attributes and relationships which together form the static properties. It also consists of integrity rules over objects and dynamic properties such as operations or rules defining new database states. An object is a collection of data and methods. When different objects of same type are grouped together they form a class. This model is used basically for multimedia applications as well as data with complex relationships. The object model is represented graphically with object diagrams containing object classes. Classes are arranged into hierarchies sharing common structure and behavior and are associated with other classes.

c. Semantic Data Models: These models are used to express greater interdependencies among entities of interest. These interdependencies enable the models to present the semantic of the data in the databases. This class of data models is influenced by the work done by artificial intelligence researchers. Semantic data models are developed to organize and represent knowledge but not data. This type of data models is able to express greater interdependencies among entities of interest. Mainframe database are increasingly adopting semantic data models. Also its growth usage is also seen in PC's. In coming times database management system will be partially or fully intelligent.

d. Functional Data Model: The functional data model describes those aspects of a system concerned with transformations of values-functions, mappings, constraints and functional dependencies. The functional data model describes the computations within a system. It shows how output value in computation are derived from input values without regard for the order in which the values are computed. It also includes constraints among values. It consists of multiple data flow diagrams. Data flow diagrams show the dependencies between values and computation of output values from input values and functions, without regard for when (or if) the functions are executed. Traditional computing concepts such as expression trees are examples of functional models, as are less traditional concepts such as spreadsheets.

High level data models use concepts such as entities, attributes and relationships. An entity represents a real-world object or concept, such as an employee or a project, that is described in the database. An attribute represents some property of interest that further describes an entity, such as the employee's name or salary. A relationship among two or more entities represents an interaction among the entities. For example, a works-on relationship between an employee and a project. The object oriented data model not only extends the state of the object but also the actions that are associated with the object, that is, its behavior. The object is said to encapsulate both state and behavior.

Low-level or physical data models: Low-level or physical data models are those that provide concepts that describe the details of how data is stored in the computer. It represents information such as record formats, record ordering, and access paths. An access path is a structure that makes the search for particular database records efficient. One of the most important low-level data model is unifying data model.

Representational or implementation data models: Between the two broad categories of data models is representational or implementation data model which provide concepts that may be understood by end users. These are used in describing data at logical and view levels of the three level architecture of DBMS. In contrast to object based data models, these are used to specify the overall logical structure of the database and to provide a higher-level description of the implementation. The most popular representational data models are

a. Hierarchical data model
b. Network data model
c. Relational data model

Representational data models represents data by using record structures and hence are sometimes called record-based data models.

Key:

Before we start discussing the concept of Key and various types of keys - Consider the Table PGDCA_Students_Details (Roll_No, SSNo, Student_Name, Father_name, Address, Phone_No). Here in this table, PGDCA_Students_Details is the name of the table and (Roll_No, SSNo, Student_Name, Father_name, Address, Phone_No) are the names of the attributes (fields) of the table. It is storing the details of the students of the PGDCA class.

Table is also called entity set and the rows in the table are also called entities.

Definition: A key is a single attribute or combination of two or more attributes of an entity set that is used to identify one or more instances of the set. The attribute Roll_No uniquely identifies an instance of the entity set PGDCA_Students_Details. Thus, a key is a property of the entity set, rather than of the individual entities. Any two individual entities in the set are prohibited from having the same value of the key attributes at the same time.

There are various types of keys that are as follows:

1. Super Key
2. Candidate Key
3. Primary Key
4. Alternative Key
5. Unique Key
6. Foreign Key

Super Key

It is a set of one or more attributes that, taken collectively, allows us to identify uniquely an entity in the entity set. It has the property of Uniqueness and Reducibility. For Example: SK1{Roll_No}, SK2{SSNO}, SK3{Roll_No,SSNo}, SK3{SSNo,Father_Name}, SK4{Roll_No,Student_Name}, SK5{Roll_No,Student_Name,Father_Name} ....

Candidate Key

If K is the Super Key, then so is any superset of K. We are basically interested in Super keys for which no proper subset is a super key. Such minimal super keys are called candidate keys. It has the property of Uniqueness and Irreducibility.
For Example:CK1{Roll_No},CK2{SSNO}.

Primary Key

A candidate key that is chosen by the database designer as the principal means of identifying entities within an entity set. Such a minimal super key or candidate key is known as Primary Key. It has the property of Uniqueness and Irreducibility.

For Example: PK1 {Roll_No}

Alternate Key

Rest of the Candidate keys that are not taken as primary key are alternate keys.

AK1 {SSNO}, AK2 {Roll_No, Father_Name}.....

Unique

It is same as primary key. The only difference is that Unique allows NULLs but Primary key does not allow NULL.

Foreign Key

Let R1 be a relation (table) in which PK1 has been assigned as primary key. Let R2 be another relation in which PK2 is the primary key but PK1 (From Relation R1) has been marked as Foreign Key. The value of PK1 in any row of relation R2 must match with the value of PK1 of any row in relation R1.

For Example: In relation PGDCA_Student_Records {Roll_No} is acting as Primary Key. We have another relation Fee_Records (Slip_ID, Roll_No, Fee_paid, Date_of_Payment), Slip_Id is acting as Primary Key and PK1 {Roll_No} is acting as Foreign Key.

Is it possible to have a foreign key when there is only one table in a database?

Yes. There can be a case in which referencing and referenced relations are same and not distinct. Consider the relation EMP such that EMP (emp-name, mgr-ID)

In the above relation, emp_id is primary key and mgr_id is foreign key. SO, EMP Referencing emp_ id of EMP relation itself. So, EMP relation is acting as referencing as will as referenced. Thus, from the above example we see that there is only one table i.e. EMP and this table has a foreign key i.e. mgr_id.

Summary

The main purpose of data modeling is to help in understanding the meaning of the data and to facilitate communication about information requirements. A data model supports communication between the users and database designers. The goal of a data model is to make sure that all data objects required by the database are completely and accurately represented because the data model uses easily understood notations and natural language which can be reviewed and verified as correct by the end users. A key is a property of the entity set, rather than of the individual entities. Any two individual entities in the set are prohibited from having the same value of the key attributes at the same time. There are different type of keys – Super, Candidate, Primary, Unique and Foreign.

Self Understanding

Q1. What do you mean by database utility? Explain the various database utilities.

Q2. What do you mean by data model? Explain in brief various data models.

Q3. Define key. Explain different types of keys along with suitable examples.

Q4. What is the difference between Unique and Primary key?

Q5. Define Foreign Key? Is it possible to have a foreign key when there is only one table in a database?

Q6. List down the characteristics of Primary Key.