**ST. XAVIER’S COLLEGE**

**(Affiliated to Tribhuvan University)**

Maitighar, Kathmandu



**DATABASE MANAGEMENT SYSTEM**

**LAB ASSIGNMENT#2**

**Submitted by:**

Shreesha Pokharel

013BSCCSIT038

**Submitted to:**

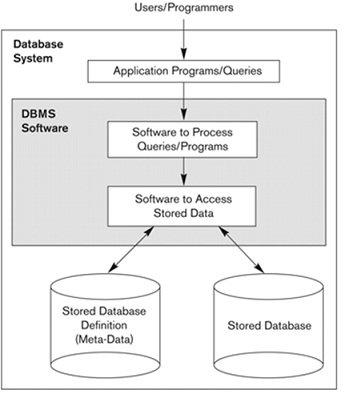
|  |  |
| --- | --- |
| **Er. Sanjay Kumar Yadav** |  |

Lecturer

Department of Computer Science

Date of submission: 27th July, 2015

**SIMPLIFIED DATABASE SYSTEM:**



Databases range from being ultra-simple to use to ultra-complex. In the world of personal computers, there are two main types of database programs. At the simple end of the scale are flat-file databases, also called single-file or list managers. These programs are as easy to learn as a word processor or spreadsheets. The initial concepts may take a little more time to absorb than word processing, but they’re well within anyone’s grasp. If you’re using a database at home, in a class at school or in a small club or organization, chances are the simple flat-file database will fill your needs.

**APPROACHES TO MANAGEMENT OF DATA:**

Data management is the development and execution of architectures, policies, practices and procedures in order to manage the information lifecycle needs of an enterprise in an effective manner.

* **Database approach:**

In order to remove all limitations of the File Based Approach, a new approach was required that must be more effective known as Database approach. The Database is a shared collection of logically related data, designed to meet the information needs of an organization. A database is a computer based record keeping system whose over all purpose is to record and maintains information.

The database is a single, large repository of data, which can be used simultaneously by many departments and users. Instead of disconnected files with redundant data, all data items are integrated with a minimum amount of duplication.

* **File system approach:**

A file system is a more unstructured data store for storing arbitrary, probably unrelated data. The file system is more general, and databases are built on top of the general data storage services provided by file systems. Transactions are not quite possible in this approach. It generally saves data in temporary locations. Data will be accessed through single or multiple files. Data are less secured in this approach.

**DATABASE vs. FILE SYSTEM APPROACH:**

* **Data abstraction:**

**File System**: Electronic data are directly stored in a set of files. If only one table is stored in a file, they are called flat files. They contain values at each row separated with a special delimiter like commas. In order to query some random data, first it is required to parse each row and load it to an array at run time. But for this file should be read sequentially (because, there is no control mechanism in files), therefore it is quite inefficient and time consuming. The burden of locating the necessary file, going through the records (line by line), checking for the existence of a certain data, remembering what files/records to edit is on the user. The user either has to perform each task manually or has to write a script that does them automatically with the help of the file management capabilities of the operating system. Because of these reasons, File Systems are easily vulnerable to serious issues like inconsistency, inability for concurrency, data isolation, threats on integrity and lack of security.

**DBMS:** For the system to be usable, it must retrieve data efficiently. The need for efficiency has led designers to use complex data structures to represent data in the database. Since many database-systems users are not computer trained, developers hide the complexity from users through several levels of abstraction, to simplify users’ interactions with the system:

* **Physical Level:** The lowest level of abstraction describes how the data are actually stored. The physical level describes complex low-level data structures in detail.
* **Logical Level:**The next-higher level of abstraction describes what data are stored in the database, and what relationships exist among those data. The logical level thus describes the entire database in terms of a small number of relatively simple structures. Although implementation of the simple structures at the logical level may involve complex physical-level structures, the user of the logical level does not need to be aware of this complexity. Database administrators, who must decide what information to keep in the database, use the logical level of abstraction.
* **View Level:** The highest level of abstraction describes only part of the entire database. Even though the logical level uses simpler structures, complexity remains because of the variety of information stored in a large database. Many users of the database system do not need all this information; instead, they need to access only a part of the database. The view level of abstraction exists to simplify their interaction with the system. The system may provide many views for the same database.
* **Reliability:**

Typically when using a File System, most tasks such as storage, retrieval and search are done manually and it is quite tedious whereas a DBMS will provide automated methods to complete these tasks. Because of this reason, using a File System will lead to problems like data integrity, data inconsistency and data security, but these problems could be avoided by using a DBMS. Unlike File System, DBMS are efficient because reading line by line is not required and certain control mechanisms are in place.

Hence, we can conclude that the reliability in storage of data is to be done in database management system rather than on file system.

* **Efficiency\ performance:**

A Database Management System provides:

* + Data Consistency and Integrity - by controlling access and minimizing data duplication
  + Application program independence - by storing data in a uniform fashion
  + Data Sharing - by controlling access to data items, many users can access data concurrently
  + Check pointing and Recovery
  + Security and Privacy
  + Multiple views of data
  + Expandability, Flexibility, Scalability
  + Reduced application development times once the system is in place
  + Standards enforcement

On the other hand, a file system fails to provide all above mentioned points. Hence, in terms of performance and efficiency, one can be assured to choose DBMS in place of file system.

**THREE LAYER ARCHITECTURE:**

3-tier or 3-layer architecture separates its tiers from each other based on the complexity of the users and how they use the data present in the database. It is the most widely used architecture to design a DBMS.



* **Database (Data) Tier** − At this tier, the database resides along with its query processing languages. We also have the relations that define the data and their constraints at this level.
* **Application (Middle) Tier** − At this tier reside the application server and the programs that access the database. For a user, this application tier presents an abstracted view of the database. End-users are unaware of any existence of the database beyond the application. At the other end, the database tier is not aware of any other user beyond the application tier. Hence, the application layer sits in the middle and acts as a mediator between the end-user and the database.
* **User (Presentation) Tier** − End-users operate on this tier and they know nothing about any existence of the database beyond this layer. At this layer, multiple views of the database can be provided by the application. All views are generated by applications that reside in the application tier.

Multiple-tier database architecture is highly modifiable, as almost all its components are independent and can be changed independently.

**Advantages of three layer architecture:**

* It makes the database abstract. It hides the details of how the data is stored physically in an electronic system, which makes it easier to understand and easier to use for an average user. It also allows the user to concentrate on the data rather than worrying about how it should be stored.
* It lets users view the same data, but it makes the data customizable to fit each user. Each user should be able to access a source of data, and any change made to their customized data source should not affect data sources being used by other users. This means that the data in your database is independent.
* The three tiered architecture model also allows migration to another system to be seamless. The database appears the same on different systems, even if the physical storage is changed. Its structure remains the same, which makes it portable.
* The model also allows a database admin to make changes to the database structure or make upgrades to it without disturbing a user currently on the system.
* The model allows a database admin to change the storage medium of the database without disturbing a user who is currently on the system.

**ADVANTAGES AND DISADVANTAGES OF DBMS:**

**Advantages:**

1. **Controlling Redundancy:** In file system, each application has its own private files, which cannot be shared between multiple applications. 1:his 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 that all redundancy should be eliminated. 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.
2. **Integrity can be enforced:** Integrity of data means that data in database is always accurate, such that 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.
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.
4. **Data can be shared:** As explained earlier, the data can be shared by multiple applications in centralized DBMS as compared to file system so now applications 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 standard can be enforced easily may be at Company level, Department level, National level or International level. The standardized data is very helpful during migration or interchanging of data. The file system is an independent system so standard 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 others are 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. A 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 variety 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".
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.
9. **Cost of developing and maintaining system is lower:**It is much easier to respond to unanticipated requests when data is centralized in a database than when it is stored in a conventional file system. Although the initial cost of setting up of a database can be large, but the cost of developing and maintaining application programs to be far lower than for similar service using conventional systems. The productivity of programmers can be higher in using non-procedural languages that have been developed with DBMS than using procedural languages.
10. **Data** **Model can be developed:**The centralized system is able to represent the complex data and interfile relationships, which results better data modeling properties. The data madding properties of relational model is based on Entity and their Relationship, which is discussed in detail in chapter 4 of the book.
11. **Concurrency Control:**DBMS systems provide mechanisms to provide concurrent access of data to multiple users.

**Disadvantages of DBMS:**

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 ~vailabi1ity 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 requirements 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 to 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.

**DRAWBACKS USING FILE SYSTEM TO STORE DATA:**

1. **Data redundancy**: Same information can be duplicated in several files. This leads to data redundancy. The files are created in the file system as and when required by an enterprise over its growth path. So in that case the repetition of information about an entity cannot be avoided.
2. **Data Inconsistency**: Data redundancy leads to greater problem than just wasting the storage i.e. it may lead to inconsistent data. Same data which has been repeated at several places may not match after it has been updated at some places.
3. **Difficulty in Accessing Data:** For generating ad hoc reports the programs will not already be present and only options present will to write a new program to generate requested report or to work manually. This is going to take impractical time and will be more expensive.
4. **Data Isolation:** Since the data files are created at different times and supposedly by different people the structures of different files generally will not match. The data will be scattered in different files for a particular entity. So it will be difficult to obtain appropriate data.
5. **Integrity Problems**: All the consistency constraints have to be applied to database through appropriate checks in the coded programs. This is very difficult when number such constraint is very large
6. **Security and access control:** Database should be protected from unauthorized users. Every user should not be allowed to access every data. Since application programs are added to the system.
7. **Concurrency Problems:** When more than one users are allowed to process the database. If in that environment two or more users try to update a shared data element at about the same time then it may result into inconsistent data.