Database Management System

Lecture: 2

File Oriented Approach:

The conventional approach for data processing is to store data locally and develop program(s) for each type of application. In the convenient file processing system, each and every subsystem of the information system will have its own set of files.

In early processing systems, an organization's information was stored as groups of records in separate files. These file processing systems constituted of a few data files and many application programs. Each file called a flat file contained and processed information for one specific function, such as accounting or inventory.

Programmers used programming languages such as COBOL to write applications that directly accessed flat files to perform data management services and provide information for users.

In creating the files and applications, developers focused on business processes or how business was transacted and their interactions. However, business processes are dynamic, requiring continuous changes in files and applications.

As system becomes more complex, file processing systems offered little flexibility, presented many limitations and were difficult to maintain.

Limitations of File Processing System:

- **Separated and Isolated Data**: To make a decision, a user might need data from two separate files. First, the files were evaluated by analysts and programmers to determine the specific data required from each file and the relationships between the data. Then, applications could be written in a third generation language to process and extract the needed data.
- **Data Redundancy**: Often, the same information was stored in more than one file. In addition to taking up more file space on the system, the replication of data caused loss of data integrity. For instance if a customer's address was stored in four different files, an address change would have to be updated in each file separately. If a user was not consistent in updating all files, no one would know which information was correct.
- **Program-data interdependence involving file formats and accessing techniques:** In file processing system, files and records were described by specific physical formats that were coded into the application program

by the programmers. If the format of a certain record was changed, the code in each file containing that format must be updated. E.g. a field in the sales file might be coded as “decimal” while the same field in the customer file could be coded as “binary”. In order to combine these files into one application, a programmer would have to write code to convert every value of the decimal field in the sales file to a binary field (or the reverse) in addition to coding the application.

- **Difficulty in representing data from user’s view:** To create useful applications for the user often data from various files must be combined. In file processing, it was difficult to determine relationships between isolated data in order to meet user requirements.
- **Data Inflexibility**: Program data interdependency and data isolation limited the flexibility of file processing system in providing users with ad hoc information requests.

File Oriented System versus Database System:

The difference between file-oriented system and database system can be discussed in the context of following points:

- **Data Independence:** Database independence can be defined as the immunity of applications to change in storage structure and access strategy. This is a main advantage of database. Both the database and the user program can be modified independently of each other. In file oriented approach, applications (programs) are data dependent and data files depend on applications (programs). If a data file is to be added to a master data file, all such programs that access the master file would have to be modified to allow this new file to be added to the master record. In this way, programs become dependent on data files and data files become dependent on programs
- **Data Redundancy**: In non-database system each application has its own private files. Due to decentralization of the data, the file oriented system leads in uncontrolled duplication of data. This can lead to redundancy in stored data which results in the wastage of storage space. It also costs time and money to enter data more than once. In a database approach, data can be stored centrally at a single storage location with controlled redundancy under DBMS, which saves space; thus eliminated redundancy and does not permit inconsistency.
- **Data Integrity:** It describes the problem of ensuring that the data in the database is accurate. Inconsistencies between two entries representing the same fact give an example of lack of integrity (caused by redundancy in the database). Integrity constraints can be viewed as a set of

assertions to be obeyed when updating a database to preserve an error-free state. Even if redundancy is eliminated, the database may still contain incorrect data. Centralized control of database ensures that adequate checks are incorporated in the DBMS to provide data integrity. In conventional file based system checks are implemented in the application that manipulates the file.

- **Centralized Control:** With central control of the database, the DBA can ensure that standards are followed in the representation of data. In a file-based system the representation of data is controlled by the application.
- **Security:** Having control over the database the DBA can ensure that access to database is through proper channels and can define the access rights of any user to any data items or defined subset of the database. The security system prevents corruption of the existing data either accidentally or maliciously. It is very difficult to implement security mechanism in a file-based system.
- **Performance and Efficiency:** In view of the size of databases and of demanding database accessing requirements good performance and efficiency are the major requirements. Knowing the overall requirements of the organization, the DBA can structure the database system to provide an overall service that is best for the enterprise. The performance and efficiency of a file bases system depends on the performance and efficiency of the application that handles the files.