**Lesson 8**

**Objectives**

* Disadvantages of Three Level Architecture
* Functions of DBMS

**Disadvantage of Three Level Architecture**

Two stage mapping may be inefficient. By adopting three level architecture approach for database systems; operations on database cannot be made directly. Query is written on external level, then this query is mapped on internal level via conceptual schema (conceptual level). Hence query execution will be completed in two stages; at first it will be mapped on conceptual level and then will be mapped on internal level. Due to this two stage mapping; query execution may required more time as compared to direct mapping.

**Functions of DBMS**

Someone can except following functions from DBMS. In other words; we can say that a choice of DBMS can be made on following functions.

* Data Storage, Retrieval, Update
* User Accessible Catalog
* Transaction Support
* Concurrency Control
* Recover Services
* Authorization Services
* Support of Data Communication
* Integrity Services
* Services to Promote Data Independence
* Utility Services

**Data Storage, Retrieval, Update**

A DBMS must furnish users with the ability to store, retrieve, and update data in the database.

* This is the fundamental function of a DBMS. From the discussion in previous lecture (three level architecture), clearly in providing this functionality the DBMS should hide the internal physical implementation details (such as file organization and storage structures) from the user.

**A user-accessible catalog**

A DBMS must furnish a catalog in which descriptions of data items are stored and which is accessible to users.

* A key feature of three level architecture is the recognition of an integrated **system catalog** to hold data about the schemas, users, applications, and so on. The catalog is expected to be accessible to users as well as to the DBMS. A system catalog, or data dictionary, is a repository of information describing the data in the database: it is, the ‘data about the data’ or **metadata**. The amount of information and the way the information is used vary with the DBMS. Typically, the system catalog stores:
* names, types, and sizes of data items;
* names of relationships;
* integrity constraints on the data;
* names of authorized users who have access to the data;
* the data items that each user can access and the types of access allowed; for example,
* insert, update, delete, or read access;
* external, conceptual, and internal schemas and the mappings between the schemas
* usage statistics, such as the frequencies of transactions and counts on the number of accesses made to objects in the database.

The DBMS system catalog is one of the fundamental components of the system.

Some benefits of a system catalog are:

* Information about data can be collected and stored centrally. This helps to maintain control over the data as a resource.
* The meaning of data can be defined, which will help other users understand the purpose of the data.
* Communication is simplified, since exact meanings are stored. The system catalog may also identify the user or users who own or access the data.
* Redundancy and inconsistencies can be identified more easily since the data is centralized.
* Changes to the database can be recorded.
* The impact of a change can be determined before it is implemented, since the system catalog records each data item, all its relationships, and all its users.
* Security can be enforced.
* Integrity can be ensured.
* Audit information can be provided.

**Transaction support**

A DBMS must furnish a mechanism which will ensure either that all the updates corresponding to a given transaction are made or that none of them is made.

* A transaction is a series of actions, carried out by a single user or application program, which accesses or changes the contents of the database.

For example, some simple transactions for the *ATM* might be to WITHDRAW amount from account. To with draw amount series of actions to be required.

* Check amount within daily limit
* Check availability of amount
* Update account’s amount
* Withdraw amount, etc.

If the transaction fails during execution, perhaps because of a computer crash, the database will be in an **inconsistent** state: some changes will have been made and others not. Consequently, the changes that have been made will have to be undone to return the database to a consistent state again.

**Concurrency control services**

A DBMS must furnish a mechanism to ensure that the database is updated correctly when multiple users are updating the database concurrently.

* One major objective in using a DBMS is to enable many users to access shared data concurrently. Concurrent access is relatively easy if all users are only reading data, as there is no way that they can interfere with one another. However, when two or more users are accessing the database simultaneously and at least one of them is updating data, there may be interference that can result in inconsistencies.

For example, consider two transactions

* T1 and T2, which are executing concurrently on account having Rs. 100 balance
* T1 is withdrawing Rs. 10 from an account (with balance X) and T2 is depositing Rs. 100 into the same account.
* If these transactions were executed **serially**, one after the other with no interleaving of operations, the final balance would be Rs. 190 regardless of which was performed first.
* However, in this example transactions T1 and T2 start at nearly the same time and both read the balance as Rs. 100. T2 then increases X by Rs.100 to Rs. 200 and stores the update in the database. Meanwhile, transaction T1 decrements its copy of X by Rs. 10 to Rs. 90 and stores this value in the database, overwriting the previous update and thereby ‘losing’ Rs. 100.
* The DBMS must ensure that, when multiple users are accessing the database, interference cannot occur.

**Recovery services**

* A DBMS must furnish a mechanism for recovering the database in the event that the database is damaged in any way.

When discussing transaction support, we mentioned that if the transaction fails then the database has to be returned to a consistent state. This may be a result of a system crash, media failure, a hardware or software error causing the DBMS to stop, or it may be the result of the user detecting an error during the transaction and aborting the transaction before it completes. In all these cases, the DBMS must provide a mechanism to recover the database to a consistent state.

**Authorization services**

A DBMS must furnish a mechanism to ensure that only authorized users can access the database.

* It is not difficult to predict instances where we would want to prevent some of the data stored in the database from being seen by all users.

For example, we may want only branch managers to see salary-related information for staff and prevent all other users from seeing this data. Additionally, we may want to protect the database from unauthorized access. The term **security** refers to the protection of the database against unauthorized access, either intentional or accidental. We expect the DBMS to provide mechanisms to ensure the data is secure.

**Support for data communication**

A DBMS must be capable of integrating with communication software.

* Most users access the database from workstations. Sometimes these workstations are connected directly to the computer hosting the DBMS. In other cases, the workstations are at remote locations and communicate with the computer hosting the DBMS over a network.
* In either case, the DBMS receives requests as **communications messages** and responds in a similar way. All such transmissions are handled by a Data Communication Manager (DCM). Although the DCM is not part of the DBMS, it is necessary for the DBMS to be capable of being integrated with a variety of DCMs if the system is to be commercially viable. Even DBMSs for personal computers should be capable of being run on a local area network so that one centralized database can be established for users to share, rather than having a series of disparate databases, one for each user. This does not imply that the database has to be distributed across the network; rather that users should be able to access a centralized database from remote locations.

**Integrity services**

A DBMS must furnish a means to ensure that both the data in the database and changes to the data follow certain rules.

* Database integrity refers to the correctness and consistency of stored data: it can be considered as another type of database protection. While integrity is related to security, it has wider implications: integrity is concerned with the quality of data itself.
* Integrity is usually expressed in terms of *constraints*, which are consistency rules that the database is not permitted to violate.

For example, we may want to specify a constraint that no student can take more than 7 courses in one semester. Here, we would want the DBMS to check when we assign a course to a student that this limit would not be exceeded and to prevent the assignment from occurring if the limit has been reached.

**Services to promote data independence**

A DBMS must include facilities to support the independence of programs from the actual structure of the database.

* We discussed the concept of data independence in previous lecture. Data independence is normally achieved through a view or subschema mechanism. Physical data independence is easier to achieve: there are usually several types of change that can be made to the physical characteristics of the database without affecting the views. However, complete logical data independence is more difficult to achieve. The addition of a new entity, attribute, or relationship can usually be accommodated, but not their removal.

**Utility services**

A DBMS should provide a set of utility services.

* Utility programs help the DBA to administer the database effectively. Some utilities work at the external level, and consequently can be produced by the DBA. Other utilities work at the internal level and can be provided only by the DBMS vendor.

Examples of utilities of the latter kind are:

* import facilities, to load the database from flat files, and export facilities, to unload the database to flat files; monitoring facilities, to monitor database usage and operation; statistical analysis programs, to examine performance or usage statistics; index reorganization facilities, to reorganize indexes and their overflows; garbage collection and reallocation, to remove deleted records physically from the storage devices, to consolidate the space released, and to reallocate it where it is needed.