Lecture Handout

Database Management System

Lecture No. 04

Reading Material

"Database Systems Principles, Design and Implementation" written by Catherine Ricardo, Maxwell Macmillan.	4.1.3, 4.1.4
Hoffer	Chapter 2

Overview of Lecture

- Internal Schema of the Database Architecture
- Data Independence
- Different aspects of the DBMS

Internal or Physical View / Schema

This is the level of the database which is responsible for the storage of data on the storage media and places the data in such a format that it is only readable by the DBMS. Although the internal view and the physical view are so close that they are generally refereed to a single layer of the DBMS but there lies thin line which actually separated the internal view form the physical view. As we know that data when stored onto a magnetic media is stored in binary format, because this is the only data format which can be represented electronically, No matter what is the actual format of data, either text, images, audio or video. This binary storage mechanism is always implemented by the Operating system of the Computer. DBMS to some extent decides the way data is to be stored on the disk. This decision of the DBMS is based on the requirements specified by the DBA when implementing the database. Moreover the DBMS itself adds information to the data which is to be stored. For example a DBMS has selected a specific File organization for the storage of data on disk, to implement that specific file system the DBMS needs to create specific indexes. Now whenever the DBMS will attempt to retrieve the data back form the file organization system it will use the same indexes information for data retrieval. This index information is one example of additional

information which DBMS places in the data when storing it on the disk. At the same level Storage space utilization if performed so that the data can be stored by consuming minimum space, for this purpose the data compression can be performed, this space optimization is achieved in such a way that the performance of retrieval and storage process is not compromised. Another important consideration for the storage of data at the internal level is that the data should be stored in such a way that it is secure and does not involve any security risks. For this purpose different data encryption algorithms may be used. Lines below detail further tidbits of the internal level.

The difference between the internal level and the external level demarcates a boundary between these two layers, now what is that difference, it in fact is based on the access or responsibility of the DBMS for the representation of data. At the internal Level the records are presented in the format that are in match with schema definition of the records, whereas at the physical level the data is not strictly in record format, rather it is in character format., means the rules identified by the schema of the record are not enforced at this level. Once the data has been transported to the physical level it is then managed by the operating system. Operating system at that level uses its own data storage utilities to place the data on disk.

Inter Schema Mapping:

The mechanism through which the records or data at one level is related to the changed format of the same data at another level is known as mapping. When we associate one form of data at the external level with the same data in another form is know as the external/Conceptual mapping of the data. (We have seen examples of external /conceptual mapping in the previous lecture) In the same way when data at the conceptual level is correlated with the same data at the internal level, this is called the conceptual/Internal mapping.

Now the question arises that how this mapping is performed. Means how is it possible to have data at one level in date format and at a higher level the same data show us the age. This hidden mechanism, conversion system or the formula which converts the date of birth of an employee into age is performed by the mapping function and it is defined in the specific ext/con mapping, for example, when the data at the conceptual level is presented as the age of the employee is done by the external schema of that specific user. Now in this scenario the ext/con mapping is performing the mapping with the internal view and is retrieving the data in desire format of the user.

In the same way the mapping between an internal view and conceptual view is performed. The figure below gives a clear picture of this mapping process and informs where the mapping between different levels of the database is performed.

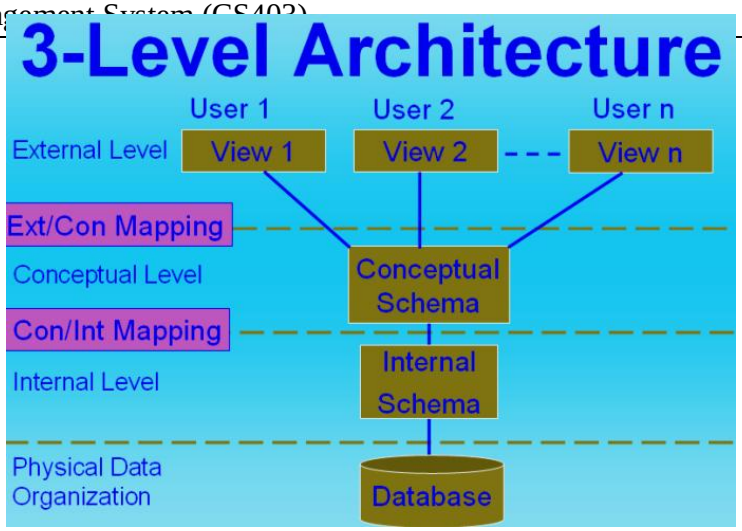

Fig: 1: Mapping between External/Conceptual and Conceptual/Internal levels

In Figure-1 we can see clearly where the mapping or connectivity is performed between different levels of the database management system. Figure-1 is showing another very important concept that the internal layer and the physical layers lie separately the Physical layer is explicitly used for data storage on disk and is the responsibility of the Operating system. DBMS has almost no concern with the details of the physical level other than that it passes on the data along-with necessary instructions required to the store that data to the operating system.

Figure-2 on the next page shows how data appears on different levels of the database architecture and also at that of physical level. We can clearly see that the data store on the physical level is in binary format and is separate from the internal view of data in location and format. Separation of the physical level from the internal level is of great use in terms of efficiency of storage and data retrieval.

Fig: 2. Representation of data at different levels of data base Architecture and at the physical level at bottom

At the internal level we can see that data is prefixed with Block Header and Record header RH, the Record header is prefixed to every record and the block header is prefixed to a group of records; because the block size is generally larger than the record size, as a result when an application is producing data it is not stored record wise on the disk rather block wise which reduces the number of disk operations and in-turn improves the efficiency of writing process.

Data Independence:

Data Independence is a major feature of the database system and one of the most important advantages of the Three Level Database Architecture. As it has been discussed already that the file processing system makes the application programs and the data dependent on each other, i-e if we want to make a change in the data we will have to make or reflect the corresponding change in the associated applications also.

The Three Level Architecture facilitates us in such a way that data independence is automatically introduced to the system. In other words we can say the data independence is major most objective of the Three Level Architecture. If we do not have data independence then whenever there will be a change made to the internal or physical level or the data accessing strategy the applications running at the external level will demand to be changed because they will not be able to properly access the changed internal or physical levels any more. As a result these applications will stop working and ultimately the whole system may fail to operate.

The Data independence achieved as a result of the three level architecture proves to be very useful because once we have the data , database and data applications independent of each other we can easily make changes to any of the components of the system, without effecting the functionality and operation of other interrelated components.
Data and program independence is on advantage of the 3-L architecture the other major advantage is that ant change in the lower level of the 3-L architecture does not effect the structure or the functionality on upper levels. I-e we get external/conceptual and conceptual/internal independence by the three levels Architecture.
Data independence can be classified into two type based on the level at which the independence is obtained.

- **Logical Data Independence**
- **Physical Data Independence**

Logical data independence
Logical data independence provides the independence in a way that changes in conceptual model do not affect the external views. Or simply it can be stated at the Immunity of external level from changes at conceptual level.

Although we have data independence at different levels, but we should be careful before making a change to anything in database because not all changes are accepted transparently at different levels. There may be some changes which may cause damage or inconsistency in the database levels. The changes which can be done transparently may include the following:

- Adding a file to the database
- Adding a new field in a file
- Changing the type of a specific field

But a change which may look similar to that of the changes stated above could cause problems in the database; for example: Deleting an attribute from the database structure, This could be serious because any application which is using this attribute may not be able to run any more. So having data independence available to us we still get problem after a certain change, it means that before making a certain change its impact should also be kept in mind and the changes should be made while remaining in the limits of the data independence.

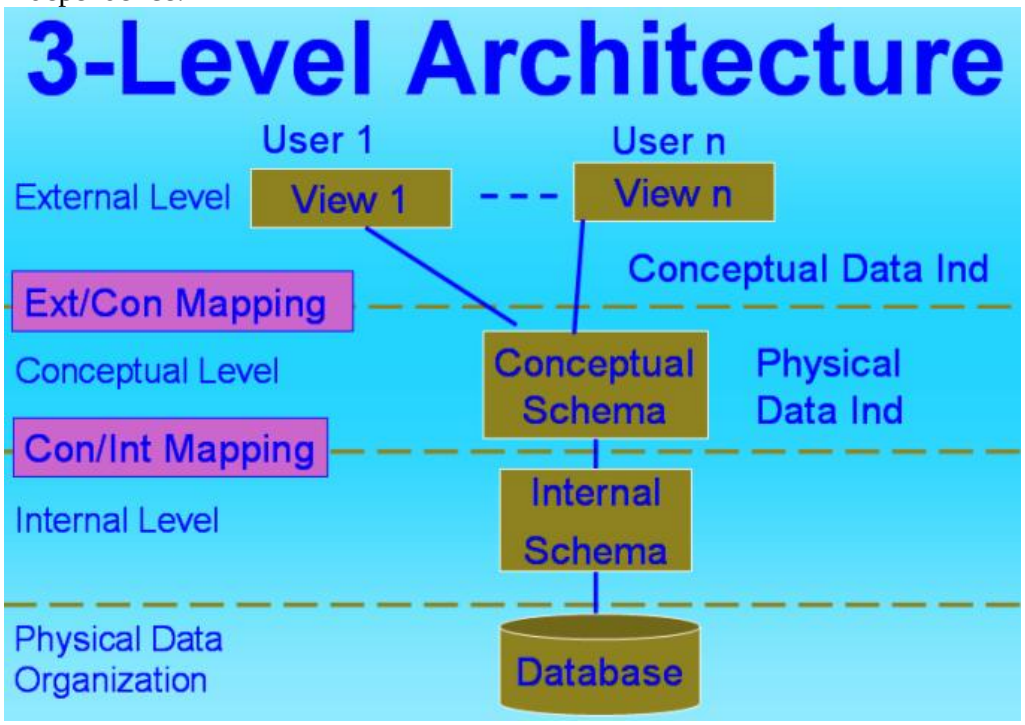

Fig:3. The levels where the Conceptual and Physical data independence are effective

Physical Data Independence

Physical data independence is that type of independence that provides us changes transparency between the conceptual and internal levels. i-e the changes made to internal level shall not affect the conceptual level. Although the independence exist but as we saw in the previous case the changes made should belong to a specific domain and should not exceed the liberty offered by the physical data independence. For example the changes made to the file organization by implementing indexed or sequential or random access at

a later stage, changing the storage media, or simply implement a different technique for managing file indexes or hashes.

Functions of DBMS

- **Data Processing**
- **A user accessible Catalog**
- **Transaction Support**
- **Concurrency Control Services**
- **Recovery Services**
- **Authorization Services**
- **Support for Data Communication**
- **Integrity Services**

DBMS lies at the heart of the course; it is the most important component of a database system. To understand the functionality of DBMS it is necessary that we understand the relation of database and the DBMS and the dissection of the set of functions the DBMS performs on the data stored in the database.
Two important functions that the DBMS performs are:

User management
Data Management

The detailed description of the above two major activities of DBMS is given below;

- **Data Processing**

By Data management we mean a number of things it may include certain operations on the data such as: creation of data, Storing of the data in the database, arrangement of the data in the databases and data-stores, providing access to the data in the database, and placing of the data in the appropriate storage devices. These action performed on the data can be classified as data processing.

- **A User Accessible Catalog**

DBMS has another very important task known as access proviso to catalog. Catalog is an object or a place in the DBMS which stores almost all of the information of the database, including schema information, user information right of the users, and many more things about the database. Modern relational DBMS require that the Administrative users of the database should have access to the catalog of the database.

- **Transaction Support**

DBMS is responsible for providing transaction support. Transaction is an action that is used to perform some manipulation on the data stored in the database. DBMS is responsible for supporting all the required operations on the database, and also manages the transaction execution so that only the authorized and allowed actions are performed.

- **Concurrency Support**

Concurrency support means to support a number of transactions to be executed simultaneously, Concurrency of transactions is managed in such a way that if two or more transaction are making certain processing on the same set of data, in that case the result of all the transactions should be correct and no information should be lost.

- **Recovery Services**

Recovery services mean that in case a database gets an inconsistent state to gets corrupted due to any invalid action of someone, the DBMS should be able to recover itself to a consistent state, ensuring that the data loss during the recovery process of the database remains minimum.

- **Authorization Services**

The database is intended to be used by a number of users, who will perform a number of actions on the database and data stored in the database, The DBMS is used to allow or restrict different database users to interact with the database. It is the responsibility of the database to check whether a user intending to get access to database is authorized to do so or not. If the user is an authorized one than what actions can he/she perform on the data.

- **Support for Data Communication**

The DBMS should also have the support for communication of the data indifferent ways. For example if the system is working for such an organization which is spread across the country and it is deployed over a number of offices throughout the country, then the DBMS should be able to communicate to the central database station. Or if the data regarding a product is to be sent to the customers worldwide it should have the facility of sending the data of the product in the form of a report or offer to its valued customers.

- **Integrity Services**

Integrity means to maintain something in its truth or originality. The same concept applies to the integrity in the DBMS environment. Means the DBMS should allow the operation on the database which are real for the specific organization and it should not allow the false information or incorrect facts.

DBMS Environments:

- **Single User**
- **Multi-user**
  - **Teleprocessing**
  - **File Servers**
  - **Client-Server**

- **Single User Database Environment**

This is the database environment which supports only one user accessing the database at a specific time. The DBMS might have a number of users but at a certain time only one user can log into the database system and use it. This type of DBMS systems are also called Desktop Database systems.

- **Multi-User Database systems**

This is the type of DBMS which can support a number of users simultaneously interacting with the database indifferent ways. A number of environments exist for such DBMS.

- **Teleprocessing**

This type of Multi user database systems processes the user requests at a central computer, all requests are carried to the central computer where the database is residing, transactions are carried out and the results transported back to the terminals. It has become obsolete now.

- **File Servers**

This type of multi-user database environment assumes another approach for sharing of data for different users. A file server is used to maintain a connection between the users of the database system. Each client of the network runs its own copy of the DBMS and the database resides on the file server. Now whenever a user needs data from the file server it makes a request the whole file containing the required data was sent to the client. At this stage it is important to see that the user has requested one or two records from the database but the server sends a complete file, which might contain hundreds of records. Now if the client after making the desired operation on the desired data wants to write back the data on the database he will have to send the whole file back to the server, thus causing a lot of network overhead. The Good thing about this approach is, that the server does not have lots of actions to do rather it remains idle for lots of the time in contrast with that of the teleprocessing systems approach.

- **Client-Server**

This type of multi-user environment is the best implementation of the network and DBMS environments. It has a DBMS server machine which runs the DBMS and to this machine are connected the clients having application programs running for each user. Once a users want to perform a certain operation on data in the database it sends it requests to the DBMS through its machine's application software; the request is forwarded to the DBMS server which performs the required operation on data in the database stored in the dame computer and then passes back the result to the user intending the result. This environment is best suited for large enterprises where bulk of data is processed and requests are very much frequent.

This concludes the topics discusses in the lecture No4.In the next lecture Database application development process will be discussed

Exercises:

- Extend the format of data from the exercise of previous lecture to include the physical and internal levels. Complete your exercise by including data at all three levels
- Think of different nature of changes at all three levels of database architecture and see, which ones will have no effect on the existing applications, which will be adjusted in the inter-schema mapping and which will effect the existing applications.

Thanks and good luck