Structure of DBMS

Data Definition Language Compiler

The DDL Compiler converts the data definition statements into a set of tables. These tables contain the metadata concerning the database and are in a form that can be used by other components of DBMS.

Structure of DBMS...

Data Manager

The data manager is the central software component of the DBMS. It is sometimes referred to as the database control system. One of the functions of the data manager is to convert operations in the user's queries coming directly via the query processor or indirectly via an application program from user's logical view to a physical file system. The data manager is responsible for interfacing with the file system. In addition, the tasks of enforcing constraints to maintain the consistency and integrity of the data, as well as its security, are also performed by the data manager. Synchronizing the simultaneous operations performed by concurrent users is under the control of the data manager. It is also entrusted with the backup and recovery operations.

Structure of DBMS...

File Manager

Responsibility for the structure of the files and managing the file space rests with the file manager. It is also responsible for locating the block containing the required record, requesting this block from the disk manager, and transmitting the required record to the data manager. The file manager can be implemented using an interface to the existing file subsystem provided by the operating system of the host computer or it can include a file subsystem written especially for DBMS.

Disk Manager

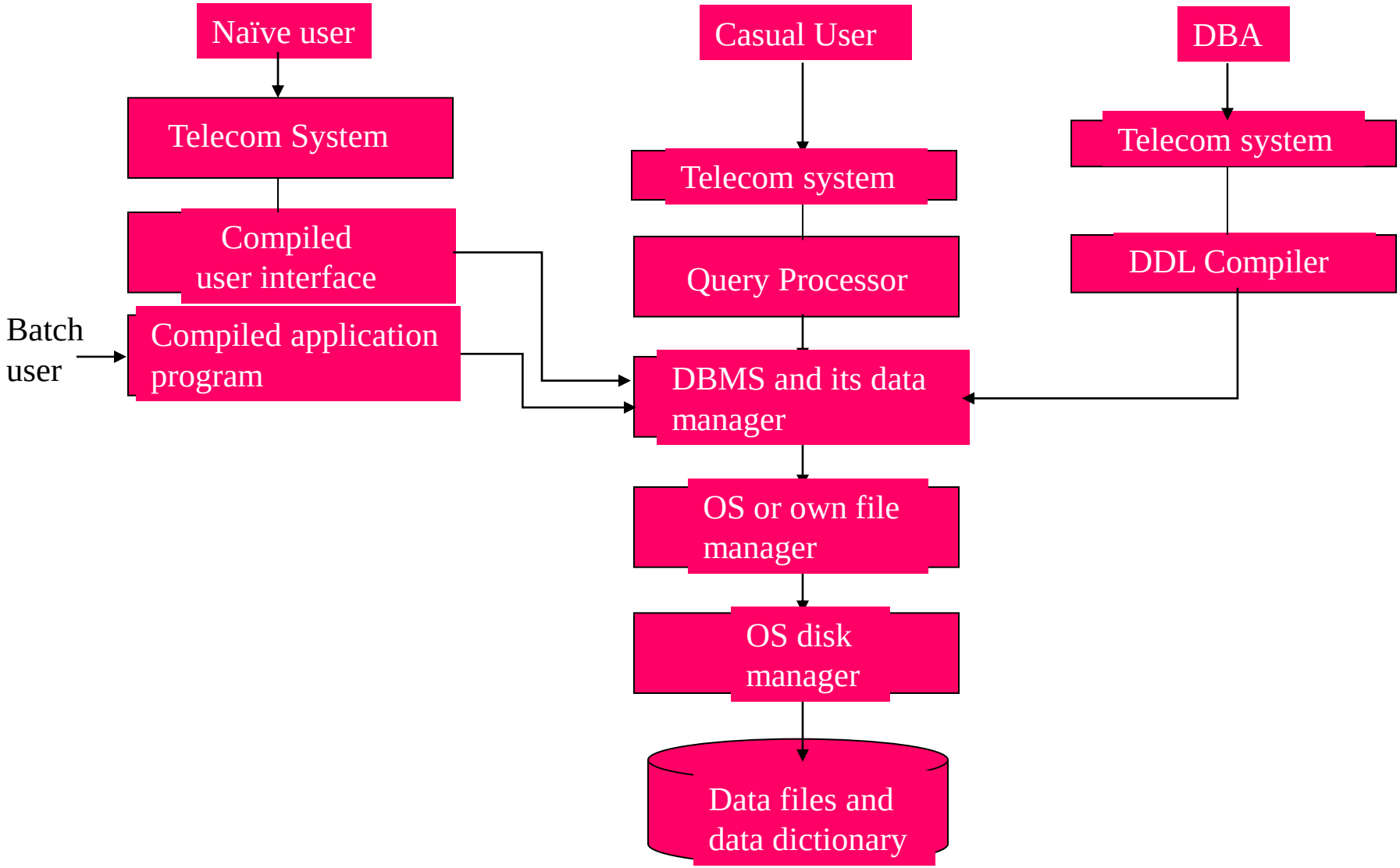
The disk manager is part of the operating system of the host computer and all physical input and output operations are performed by it. The disk manager transfers the block or page requested by the file manager so that the latter need not be concerned with the physical characteristics of the underlying storage media.

Structure of DBMS...

Query Processor

The database user retrieves data by formulating a query in the data manipulation language provided with the database. The query processor is used to interpret the online user's query and convert it into an efficient series of operations in a form capable of being sent to the data manager for execution. The query processor uses the data dictionary to find the structure of the relevant portion of the database and uses the information in modifying the query and preparing an optimal plan to access the database.

Structure of DBMS...



Structure of DBMS...

Data Files

Data files contain the data portion of the database.

Data dictionary

Information pertaining to the structure and usage of data contained in the database, the metadata, is maintained in a data dictionary. The term system catalog also describes this meta data. The data dictionary, which is a database itself, documents the data. Each database user can consult the data dictionary to learn what each piece of data and various synonyms of the data fields mean.

Structure of DBMS...

Data dictionary…

In an integrated system (i.e., in a system where the data dictionary is a part of the DBMS) the data dictionary stores information concerning the external, conceptual, and internal levels of the database. It contains the source of each data-field value, the frequency of its use, and an audit trail concerning updates, including the who and when of each update.

Currently data dictionary systems are available as add-ons to the DBMS. Standards have yet to be evolved for integrating the data dictionary facility with the DBMS so that the two databases, one for metadata and the other for data, can be manipulated using an unified DDL/DML.

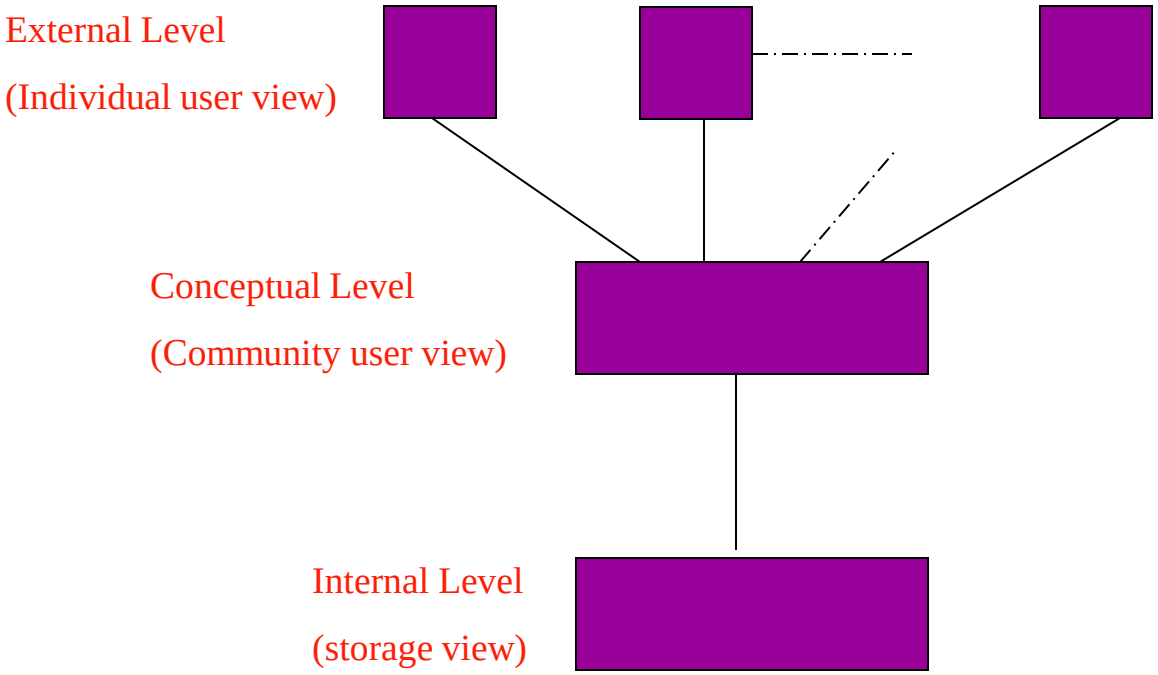
Structure of DBMS...

Access Facilities

To improve the performance of a DBMS, a set of access facilities in the form of indexes are usually provided in a database system. Commands are provided to build and destroy additional temporary indexes.

An Architecture for a Database System

The architecture is divided into three general levels: internal, conceptual, and external



An Architecture for a Database System…

Broadly speaking,

The ***internal level*** is the one closest to the physical storage.

The ***external level*** is the one closest to the users, that is, the one concerned with the way in which the data is viewed by individual users.

The ***conceptual level*** is a “level of indirection” between the other two.

An Architecture for a Database System…

If the ***external level*** is concerned with the individual user views, the ***conceptual level*** may be thought of as defining a community user view. In other words, there will be many ***"external views,"*** each consisting of a more or less abstract representation of some portion of the database, and there will be a single ***"conceptual view,"*** consisting of a similarly abstract representation of the database in its entirety.(Remember that most users will not be interested in the total database, but only in some restricted portion of it.) Likewise, there will be a single ***"internal view,"*** representing the total database as actually stored.

An Architecture for a Database System…

An Example of Three Levels

External (PL/I)	External (COBOL)
DCL 1 EMPP, 2 EMP# CHAR(6), 3 SAL FIXED BIN(31);	01 EMPC. 02 EMPNO PIC X(6). 03 DEPTNO PIC X(4).

Conceptual

```
EMPLOYEE
    EMPLOYEE_NUMBER      CHARACTER (6)
    DEPARTMENT_NUMBER CHARACTER (4)
    SALARY                        NUMRIC      (5)
```

Internal

```
STORED_EMP LENGTH=18
    PREFIX    TYPE=BYTE(6), OFFSET=0
    EMP#      TYPE=BYTE(6), OFFSET=6, INDEX=EMPX
    DEPT#     TYPE=BYTE(4), OFFSET=12
    PAY       TYPE=FULLWORD, OFFSET=16
```

An Architecture for a Database System…

Figure shows the conceptual structure of a simple personnel database, the corresponding internal structure, and two corresponding external structure (one for a PL/I user, the other for COBOL user). The example is completely hypothetical – it is not intended to resemble any actual system – and many irrelevant details have been deliberately omitted.

An Architecture for a Database System…

- At the conceptual level, the database contains information concerning an entity type called EMPLOYEE. Each EMPLOYEE has an EMPLOYEE_NUMBER (six character), a DEPARTMENT_NUMBER (four characters), and a SALARY (five digits).

An Architecture for a Database System…

- At the internal level, employees are represented by a stored record type called STORED_EMP, eighteen bytes long. STORED_EMP contains four stored field types: a six-byte prefix (presumably containing control information such as flags or pointers) and three data fields corresponding to the three properties of EMPLOYEE. In addition, STORED_EMP records are indexed on the EMP# field by an index called EMPX.

An Architecture for a Database System…

- The PL/I user has an external view of the database in which each employee is represented by a PL/I record containing two fields (department numbers are of no interest to this user and are omitted from the view). The record type is defined by an ordinary Pl/I structure declaration in accordance with the normal Pl/I rules.

An Architecture for a Database System…

- Similarly, the COBOL user has an external view in which each employee is represented by a COBOL record containing, again, two fields (salaries are omitted). The record type is defined by an ordinary COBOL record description in accordance with normal COBOL rules.

An Architecture for a Database System…

- Notice that corresponding objects at the three levels can have different names at each level. For example, the COBOL employee number field is called EMPNO, the corresponding stored field is called EMP#, and the corresponding conceptual object is called EMPLOYEE_NUMBER.(The system must be aware of the correspondences. For example, it must be told that the COBOL field EMPNO is derived from the conceptual object EMPLOYEE_NUMBER, which in turn is represented by the stored field EMP#. Such correspondences, or mapping, have not been shown in the figure).

Components of Architecture

Users

The users are either application programmers or on-line terminal users of any degree of sophistication. Each user has a language at his or her disposal. For the application programmer it will be a conventional programming language, such as COBOL or PL/I; for the terminal user it will be either a query language or a special purpose language tailored to that user's requirements and supported by an on-line application program.

Components of Architecture...

External View

An individual user will generally be interested only in some portion of the total database; moreover, the user's view of that portion will generally be some what abstract when compared with the way in which the data is physically stored. In ANSI terms an individual user's view is called an external view.

Components of Architecture...

External View…

An external view is thus the content of the database as it is seen by some particular user, for example, a user from personnel department, users in the purchasing department. In general, then, an external view consists of multiple occurrences of multiple types of external records. An external record is not necessarily the same as a stored record. The user's data sub language is defined in terms of external records.

Components of Architecture...

External View…

Each external view is defined by means of an external schema, which consists basically of definitions of each of the various types of external record in the that external view. The external schema is written using the DDL portion of the data sub language. That DDL is therefore sometimes called an external DDL.

Components of Architecture...

Conceptual View

The conceptual view is a representation of the entire information content of the database, again in a form that is somewhat abstract in comparison with the way in which the data is physically stored.(It may also be quite different from the way in which the data is viewed by any particular user. Broadly speaking, it is intended to be a view of the data “as it really is,” rather than as users are forced to see it by the constraints of the particular language or hardware they are using.

Components of Architecture...

Conceptual View…

A conceptual record is not necessarily the same as either an external record, on the one hand, or a stored record, on the other. The conceptual view is defined by means of the conceptual schema, which includes definitions of each of the various types of conceptual record. The conceptual schema is written using another data definition language – the conceptual DDL.

Components of Architecture...

Conceptual View…

The conceptual view, then, is a view of the total database content, and the conceptual schema is a definition of this view. However, it would be misleading to suggest that the conceptual schema is nothing more than a set of definitions much like the simple record definitions in a COBOL program today. The definitions in the conceptual schema are intended to include a great many additional features, such as authorization checks and validation procedures.

Components of Architecture...

Internal Level

The internal view is a very low level representation of the entire database; it consists of multiple occurrences of multiple types of internal record. “Internal record” is the ANSI term for the construct that we have been calling a stored record; the internal view is thus still at one remove from the physical level, since it does not deal in terms of physical records or blocks, nor with any device-specific constraints such as cylinder or track sizes.

Components of Architecture...

Internal Level…

The internal view is described by means of the internal schema, which not only defines the various types of stored record but also specifies what indexes exist, how stored fields are represented, what physical sequence the stored record are in, and so on. The internal schema is written in using yet another data definition language - the internal DDL.

Components of Architecture...

Mapping between Conceptual & Internal

The conceptual/internal mapping defines the correspondence between conceptual view and the stored database; it specifies how conceptual records and fields map into their stored counterparts. If the structure of the stored database is changed – i.e., if a change is made to the storage structure definition – the conceptual/internal mapping must be changed accordingly, so that the conceptual schema may remain invariant. In other words, the effects of such changes must be contained below the conceptual level, so that data independence can be achieved.

Mapping between External & Conceptual

An external/conceptual mapping defines the correspondence between a particular external view and the conceptual view. In general, the same sort of differences may exist between these two levels as may exist between the conceptual view and stored database. For example, fields may have different data types, records may be differently sequenced and so on. Any number of external views may exist at the same time; any numbers of users may share a given external view; different external views may overlap.