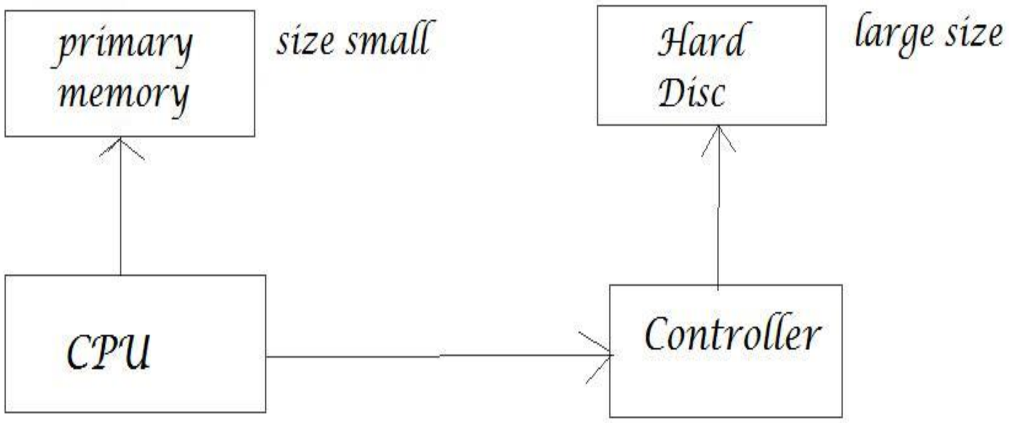
FILE STRUCTURE

File Organisation:-



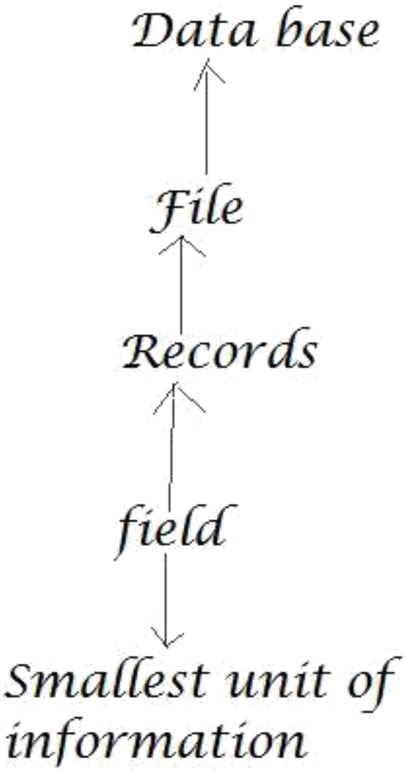
It is the organisation of records in a file.

Record:-

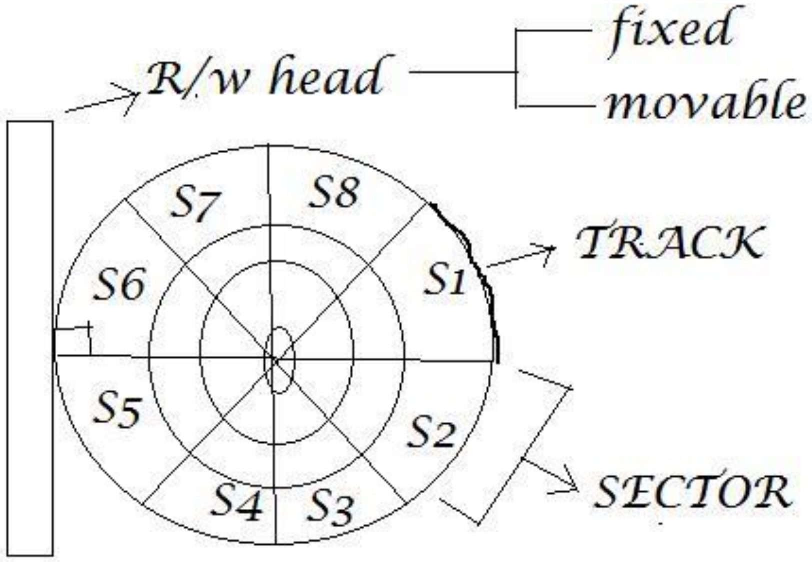
It is the collection of related fields.

Field:-

It is the smallest logically meaningful unit of information in a file.



Secondary memory structure:-



Seek time:-

It is the track access time by R/w haed.

Latency time:-

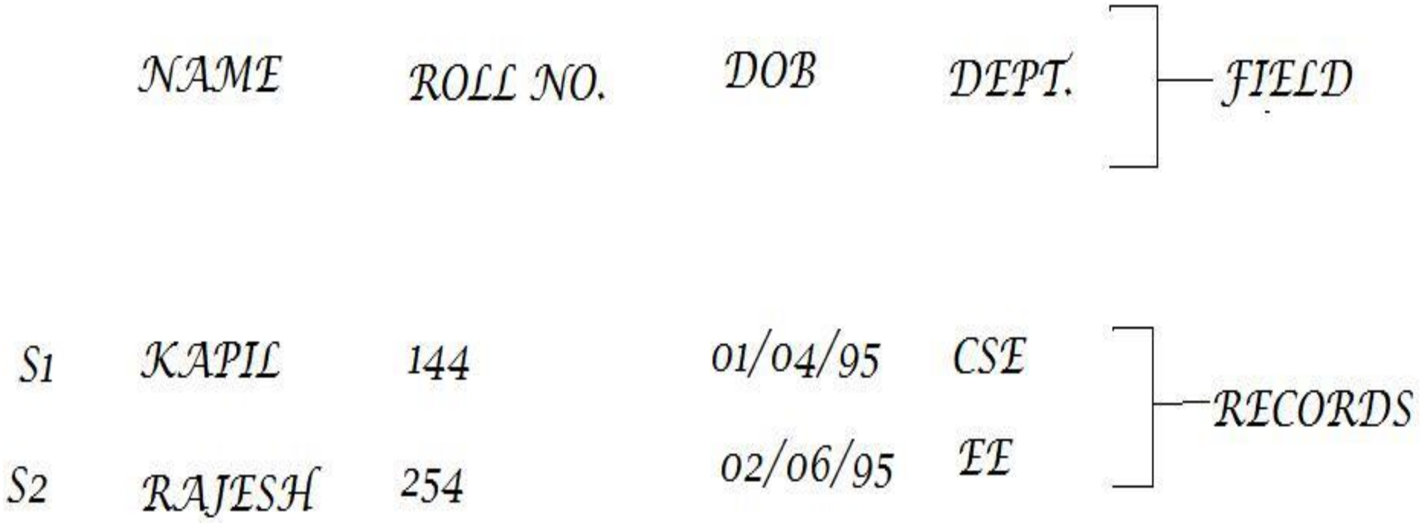
It is the time taken for the sector movement.

“L< S”

Data access time:-It is the time taken for the file movement.Student file:-

KEY:-

Key is the field which uniquely identify the records.



	NAME	ROLL NO.	DOB	DEPT.	FIELD
S1	KAPIL	144	01/04/95	CSE	RECORDS
S2	RAJESH	254	02/06/95	EE	

(1) Key must have distinct value. (no duplicate)

(2) Key must be in proper order.

File:-

Records binary relation:-

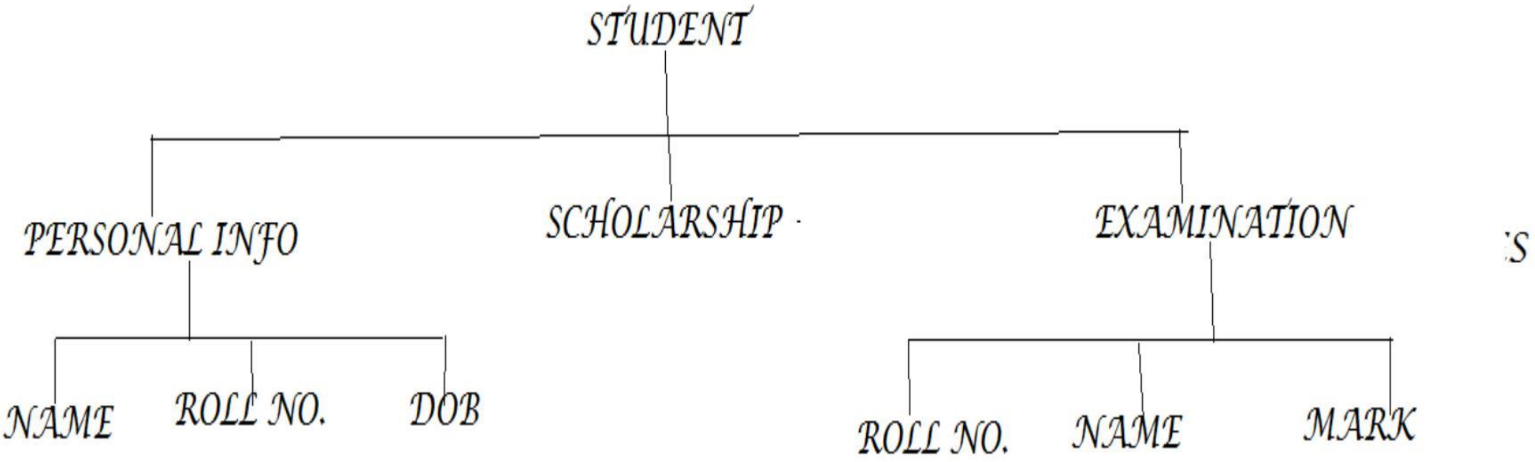
	A	B	C	D	E
r1					
r2					
r3					
r4					

$r1 \in A*B*C*D*E$

No. of records(rows) is called cardinality.

No. of fields(columns) is called degree.

APPLICATION:-



Types of files –

1. Serial Files
2. Sequential Files
3. Direct or Random Files
4. Index Sequential Files

Serial Files –

- This type of file was used in 1950-1960.
- The records are arranged one after another.
- New record is added at the end of the file.
- These files were stored in magnetic tapes.
- When the size of the file increases, the time required to access data becomes more. This is because it can only apply linear search.
- Logical ordering = Physical ordering
- These are used in secondary files.
- Examples – A material without page no. Here searching will be difficult due to lack of page number.

- The records are orders.
- . There is no key value.
- Gaps are left so that new records can be added there to maintain ordering.
- New record is added in gaps left between records according to ordering.
- These files were stored in magnetic tapes.
- When the size of the file increases, the time required to access data becomes more. This is because there are no key.
- Logical ordering may not be equal to Physical ordering.
- These are used in master files.
- Examples – Arranging cards in sequential order (A 2 3 4 5 6 7 8 9 10 J Q K)

Direct or Random Files-

- Each record is associated with a direct function.
- There are key values.
- The records have mapping.
- Disk device like CD are used.
- Searching is quit faster.
- Hashing is its extension.
- Due to mapping more space is needed.
- It is more complex than the previous 2 files.

Index Sequential File –

- This was invented by IBM.
- These uses indices.
- Access is sequential.
- Indexing and sorting is random.
- These have keys.
- A group of keys are given one index.
- Disk device is used for storage.
- Searching is fast as the index is searched and then the key is searched.
- This is used in banking.
- Example – Contents of a book. The topics are the keys. They have indices like page number. That topic can be found in that page no. When new information needs to be added, a pointer is taken to point to a new location like in appendix of a book. This saves the time and errors that occur due to shifting the later data after insertion.

Key-

A vital, crucial element notched and grooved, usually metal implement that is turned to open or close a lock. The keys have the characteristics of having unique attribute.

In database management systems, a key is a field that you use to sort data.

It can also be called a key field , sort key, index, or key word.

For example, if you sort records by age, then the age field is a key.

Most database management systems allow you to have more than one key so that you can sort records in different ways.

One of the keys is designated the primary key, and must hold a unique value for each record.

A key field that identifies records in a different table is called a foreign key.

LOCK: In general a device operated by a key, combination, or keycard and used for holding, closing or securing the data.

The key lock principle states that once lock can be opened or fastened by specific one type of keys only.
In digital electronics latch is used for temporary security while lock is used for permanent security.

FILE ORGANISATION TECHNIQUES:

The structure of a file (especially a data file), defined in terms of its components and how they are mapped onto backing store.

Any given file organization supports one or more file access methods.

Organization is thus closely related to but conceptually distinct from access methods.

The distinction is similar to that between data structure and the procedures and functions that operate on them (indeed a file organization is a large-scale data structure), or to that between a logical schema of a database and the facilities in a data manipulation language.

There is no very useful or commonly accepted taxonomy of methods of file organization: most attempts confuse organization with access methods.

Choosing a file organization is a design decision, hence it must be done having in mind the achievement of good performance with respect to the most likely usage of the file

The criteria usually considered important are:

1. Fast access to single record or collection of related records.
2. Easy record adding/update/removal, without disrupting
3. Storage efficiency.
4. Redundancy as a warranty against data corruption.

To read a specific record from an indexed sequential file, you would include the KEY= parameter in the READ (or associated input) statement.

The "key" in this case would be a specific record number (e.g., the number 35 would represent the 35th record in the file).

The direct access to a record moves the record pointer, so that subsequent sequential access would take place from the new record pointer location, rather than the beginning of the file.

Now question arises how to acces these files. We need KEYS to acess the file.

TYPES OF KEYS:

PRIMARY KEYS: The primary key of a relational table uniquely identifies each record in the table.

It can either be a normal attribute that is guaranteed to be unique (such as Social Security Number in a table with no more than one record per person) or it can be generated by the DBMS (such as a globally unique identifier, or GUID, in Microsoft SQL Server).

Primary keys may consist of a single attribute or multiple attributes in combination.

Examples: Imagine we have a STUDENTS table that contains a record for each student at a university.

The student's unique student ID number would be a good choice for a primary key in the STUDENTS table.

The student's first and last name would not be a good choice, as there is always the chance that more than one student might have the same name.

ALTERNATE KEY: The keys other than the primary keys are known as alternate key.

CANDIDATE KEY: The Candidate Keys are super keys for which no proper subset is a super key. In other words candidate keys are minimal super keys.

SUPER KEY: Super key stands for superset of a key. A Super Key is a set of one or more attributes that are taken collectively and can identify all other attributes uniquely

SECONDARY KEY: Any field in a record may be a secondary key.

The problem with secondary keys is that they are not unique and are therefore likely to return more than one record for a particular value of the key.

Some fields have a large enough range of values that a search for a specific value will produce only a few records; other fields have a very limited range of values and a search for a specific value will return a large proportion of the file.

An example of the latter would would be a search in student records for students classified as freshmen.

FOREIGN KEY : Foreign Key (Referential integrity) is a property of data which, when satisfied, requires every value of one attribute of a relation to exist as a value of another attribute in a different relation.

For referential integrity to hold in a relational database, any field in a table that is declared a foreign key can contain either a null value, or only values from a parent table's primary key or a candidate key.

In other words, when a foreign key value is used it must reference a valid, existing primary key in the parent table.

For instance, deleting a record that contains a value referred to by a foreign key in another table would break referential integrity.

Some relational database management systems can enforce referential integrity, normally either by deleting the foreign key rows as well to maintain integrity, or by returning an error and not performing the delete.

Which method is used may be determined by a referential integrity constraint defined in a data dictionary. "Referential" the adjective describes the action that a foreign key performs, 'referring' to a link field in another table.

In simple terms, 'referential integrity' is a guarantee that the target it 'refers' to will be found.

A lack of referential integrity in a database can lead relational databases to return incomplete data, usually with no indication of an error.

A common problem occurs with relational database tables linked with an 'inner join' which requires non-NULL values in both tables, a requirement that can only be met through careful design and referential integrity.

ENTITY INTEGRITY:

In the relational data model, entity integrity is one of the three inherent integrity rules.

Entity integrity is an integrity rule which states that every table must have a primary key and that the column or columns chosen to be the primary key should be unique and not NULL.

Within relational databases using SQL, entity integrity is enforced by adding a primary key clause to a schema definition.

The system enforces Entity Integrity by not allowing operation (INSERT, UPDATE) to produce an invalid primary key.

Any operation that is likely to create a duplicate primary key or one containing nulls is rejected.

The Entity Integrity ensures that the data that you store remains in the proper format as well as comprehensible.