Assignment :- 6

(1) Explain indexed file organization.

→ Indexed file organization is a method ~~using~~ used in data structures and algorithms (DSA) to improve the efficiency of data retrieval from files.

→ In this method, an index is created for a file that helps quickly locate the data records without having to scan the entire file.

→ An index is a separate table that stores key attributes along with pointers to the actual locations of data in the file.

→ These are the actual data entries in the file.

→ These are links or references in the index that point to the location of records in the file.

→ The data file is divided into blocks (or pages) that store the actual records.

→ An index table is maintained separately, which stores key values along with the address of the corresponding record in the data file.

→ The index helps in reducing the no. of searches needed to find a record.

→ Searching for records becomes faster, as you do not need to scan the entire file.

→ In some cases, updating or deleting records can be more efficient when using indexes.

→ Indexes requires additional storage.

→ Whenever data in the file is updated, the index may also need to be updated, which can increase the complexity.

(2) Explain random file organization.

→ Random file organization, also known as direct or hashed file organization, is a method in data structures and algorithms where records are stored at specific locations in a file based on a hashing function applied to a key field.

→ This method allows for direct access to records, making data retrieval very efficient.

→ A mathematical function that takes a key and converts it into a unique address where the record is stored in the file.

→ A bucket is a location or block in memory where the record is stored.

→ Multiple records may share the same bucket if their hash values collide.

→ When two different keys produce the same hash value and therefore point to the same loch.

→ Collisions are resolved using techniques like chaining or open addressing.

→ Because of the hashing function, records can be accessed directly without the need for sequential search, resulting in fast retrieval times.

→ Each record in the file has a unique key (like an ID).

→ A hashing function is applied to the key to calculate an address where the record should be stored.

→ When retrieving a record, the same hashing function is used to compute the address, and the record is fetched from that specific location.

(3) Explain sequential file organizations and list its advantages and disadvantages.

→ Sequential file organization is a method in data structures and algorithms (DSA) where records are stored in a sequential order based on a key field.

→ In this method, records are typically written

one after the other, and they are accessed sequentially.

→ To retrieve, insert, or delete a record, the file is usually searched from the beginning until the desired record is found.

→ Records are stored in a specific, ordered sequence based on a key, often a primary key such as an ID number.

→ The records are stored contiguously in the file, making it easy to read one record after another.

→ To find a record, the file is often searched linearly from the start, unless indexing is applied.

→ Sequential files are suitable for batch processing, where operations are performed on multiple records at once.

• advantages :-

→ The concept is easy to understand and implement. It works well for systems that require simple data structures.

→ Sequential access to records is fast and efficient since records are stored contiguously.

→ This method is highly efficient for systems that process large volumes of data at once, such as payroll systems or billing systems.

→ There is no need for extra structures like indexes or hash tables, which reduces overhead in terms of space.

(4) → How access of record is performed in multi key file organization ?

→ In multi-key file organization in Data structures and Algorithms, accessing records involves using multiple keys to locate and retrieve records from a file.

→ This type of file organization supports accessing data through more than one field, allowing for flexible and efficient searching based on different keys such as an emp ID, dep. or job title.

→ • How access of Records is performed in Multi-key file organization.

→ An index is created based on a primary key. This index directly maps the primary key to the record's location in the file.

→ Additional indexes are created for other keys. These indexes store pointers to the records in the file.

→ Access can be performed through either the primary key or secondary keys based on the requirement.

→ The primary index is used for direct access to records. The system uses the key to directly locate the record in the file.

→ When a secondary key is used, the process is slightly more complex because multiple records may match the key.

→ In some cases, a multi-key file organization uses the inverted file method. This involves maintaining separate inverted lists for each secondary key.

→ Sometimes, hashing functions are used for each key. Each key has its own hash function that directly points to the record location in the file.

→ In more complex scenarios, multi-level indexing is used where multiple levels of indexes are created for the keys.

(5) Define terms with respect to file : fields, records, database.

(1) Fields :- A field is a single unit of data or a specific piece of information within a record.

→ It represents a specific attribute of an entity and can be thought of as a column in a table.

→ Each field has a name and a data type.

→ For example, in an employee records, fields might include:
Employee ID (integer)
Name (string)
Department (string)
Salary (float).

(2) Records: A record is a collection of related fields that represent a single entry or instance of an entity within a file.

→ It is analogous to a row in a table.

→ Each record contains multiple fields, with each field providing specific information about that record.

(3) Database :- A database is a structured collection of records stored in files.

→ It is a comprehensive system that enables efficient storage, retrieval, and management of data.

→ A database can contain multiple tables, each consisting of records and fields. The tables are often related to one another through keys.

→ For eg, a simple employee database might consist of:

→ employees table: Contains records of employees with fields for Employee ID, Name, Department, and salary.

→ Departments Table: Contains records of department with fields for Department ID and Department Name

→ Databases can be managed using Database Management systems, which provide functionalities for querying, updating, and managing the data

(6) Explain basic file operations.

→ Basic file operations are fundamental actions that can be performed on files in a computer systems.

→ These operations are essential for managing data stored in files and can be applied in various programming languages and environments.

→ • Creating a File:- This operation involves initializing a new file in the filesystem. It typically involves specifying the file name and path.

→ Creating file is often the first step in data storage, allowing algorithms to save results or data structures to disk.

→ • opening a file: Before any operation can be performed on a file, it must be opened. This establishes a connection between the program and the file.

→ Opening file in the correct mode is crucial for algorithms that need to read or write data efficiently, ensuring that data integrity is maintained.

→ • Reading from a file :- This operation retrieves data from the file. It can be done in various ways, such as reading the entire file at once or line by line.

→ Reading data efficiently is vital in algorithms that requires input data, such as sorting or searching algorithms.

→ • Writing to a file: This operation stores data into a file. It can either overwrite existing data or append new data, depending on how the file is opened.

→ Writing results of computations or storing intermediate data structures is essential for many algorithms, particularly in data analysis and processing tasks.

• Closing a file :- After performing the necessary file operations, the file must be closed to release resources and ensure that all data is properly saved.

→ closing files helps prevent data corruption and ensures that resources are freed, which is particularly important in memory-constrained environments.

→ • Deleting a file :- This operation removes a file from the filesystem permanently.

→ In certain algorithms, particularly those involving temporary data storage or caching, files may need to be deleted once they are no longer needed.

→ • Renaming a file :- This operation changes the name of an existing file without affecting its content.

→ Renaming is useful for organizing files or creating backups during data processing tasks.

→ • Copying a file :- This open creates a duplicate of an existing file.

→ Copying files is often used in algorithms that need to maintain ~~various~~ versions of data or create backups.

(7) What is file organization? Briefly summarize different file organizations.

→ File organization refers to the way data is stored in a file on storage devices, such as a hard disk or SSD, in a structured manner.

→ It determines how records are arranged, how data can be accessed, and how efficiently the data can be retrieved.

→ In Data structure and algorithms the choice of file organization impacts the speed of data access, storage efficiency, and performance of operations like searching, updating, or deleting records.

(i) Sequential file organization :- Records are stored in a sequential ~~m~~order, usually sorted by a primary key.

→ Data is read in sequence from the beginning to the end. To find a specific record, you may need to scan through previous records.

→ simple and easy to implement; efficient for large batch processing.

→ Slow for searching and inserting records especially when the file grows large.

(2) Indexed file organization :- An index is created for each record or group of records, and the actual data is stored separately.

→ The index stores pointers to the location of records.

→ Records can be accessed through the index, allowing for efficient search without scanning the entire file.

→ efficient for searching and retrieval, especially when searching based on non-primary keys.

→ Requires additional space for storing the index; index maintenance is necessary during updates.

(3) Multi-key File organization :- The file is organized in such a way that records can be accessed using multiple keys, rather than just a primary key.

→ Multiple indexes are created for different keys, allowing flexible searching.

→ Flexible and efficient for searching based on multiple attributes.

→ Increased complexity and storage overhead due to maintaining multiple indexes.

(4) Direct / Hashed file organization: Records are stored at specific locations determined by a hash function that calculates the locn based on a key.

→ Allow for direct access to records, bypassing the need for sequential search.

→ Very fast for searching, inserting, and deleting records.

→ Requires a good hash function to avoid collisions; handling collisions can be complex.

(8) Give a brief note on indexing.

→ Indexing is used to speed up retrieval of records.

→ It is done with the help of a separate sequential file. Each record of in the index file consists of two fields, a key field and a pointer into the main file.

→ To find a specific record for the given key value, index is searched for the given key value.

→ Binary search can used to search in Index file. After getting the address of record from index file, the record in main file can easily be retrieved.

Search Key | Pointer

Key	Pointer
1000	
1010	
1012	
1015	
1016	

Index File

Main File

Name	Roll No	year	Marks
AMIT	1010	1	70
KALPESH	1016	1	80
JITENDRA	1000	3	65
RAVI	1012	2	78
NILESH	1015	1	95

→ Index file is ordered on the ordering key Roll No. each record of index file points to the corresponding record. Main file is not sorted.

- Advantages :-

→ Sequential file can be searched effectively on ordering key. When it is necessary to search for a record on the basis of some other attribute than the ordering key field, sequential file representation is inadequate.

→ Multiple indexes can be maintained for each type of field used for searching.

→ Thus, indexing provides much better flexibility

→ An index file usually requires less storage space than the main file.

→ A binary search on sequential file will require accessing of more blocks.

→ This can be explained with the help of the following eg.

→ Consider the example of a sequential file with r = 1024 records of fixed length with record size R = 128 bytes stored on disk with block size B = 2048 bytes.

→ No. of blocks required to store the file
$= \frac{1024 \times 128}{2048} = 64.$

→ No. of block access for searching a record $= \log_2 64 = 6.$

→ Suppose, we want to construct an index on a key field that is V = 4 bytes long and the block pointer is P = 4 bytes long.

→ A record of an index file is of the form <Vi; Pj> and it will need 8 bytes per entry.

→ Total no. of index entries = 1024.

→ No. of block b' required to store file
$= \frac{1024 \times 8}{2048} = 4.$

→ No. of block accesses for searching a record $= \log_2 4 = 2$

→ With indexing, new records can be added at the end of the main file.

→ It will not require movement of records as in the case of sequential file.

→ Updation of index file requires fewer block accesses compare to sequential file.