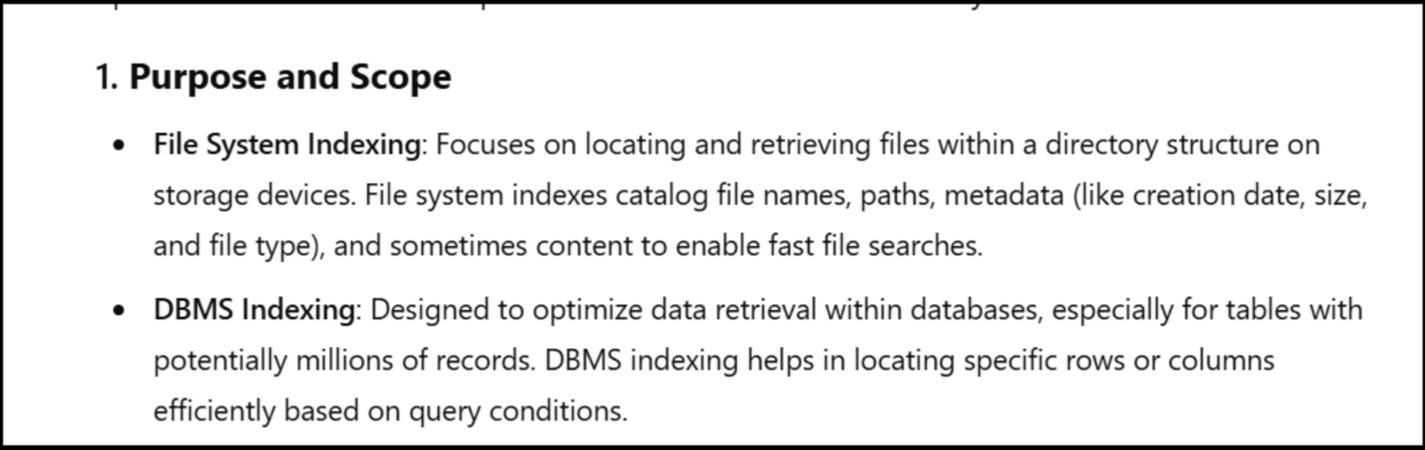
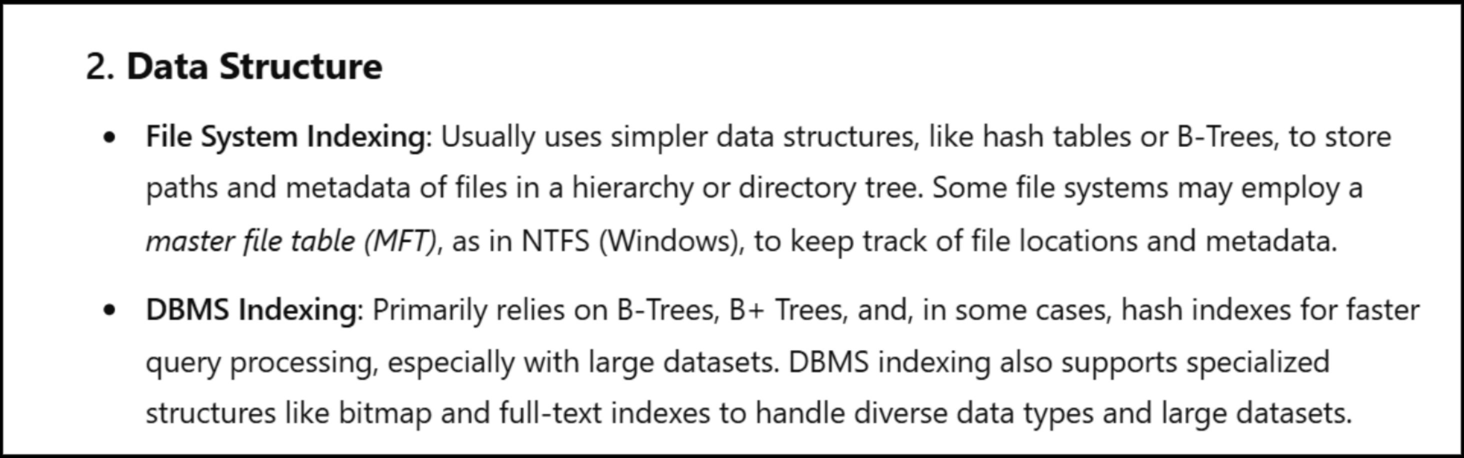
Indexing of file system
VS
Indexing of DBMS files



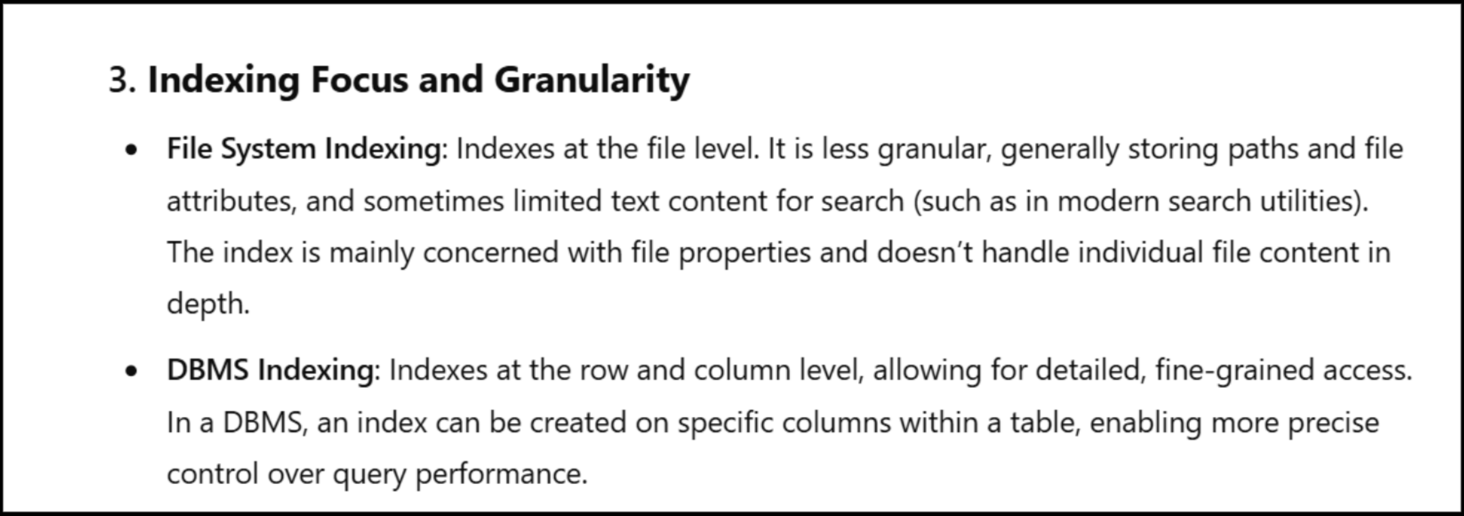
1. Purpose and Scope

- **File System Indexing**: Focuses on locating and retrieving files within a directory structure on storage devices. File system indexes catalog file names, paths, metadata (like creation date, size, and file type), and sometimes content to enable fast file searches.
- **DBMS Indexing**: Designed to optimize data retrieval within databases, especially for tables with potentially millions of records. DBMS indexing helps in locating specific rows or columns efficiently based on query conditions.



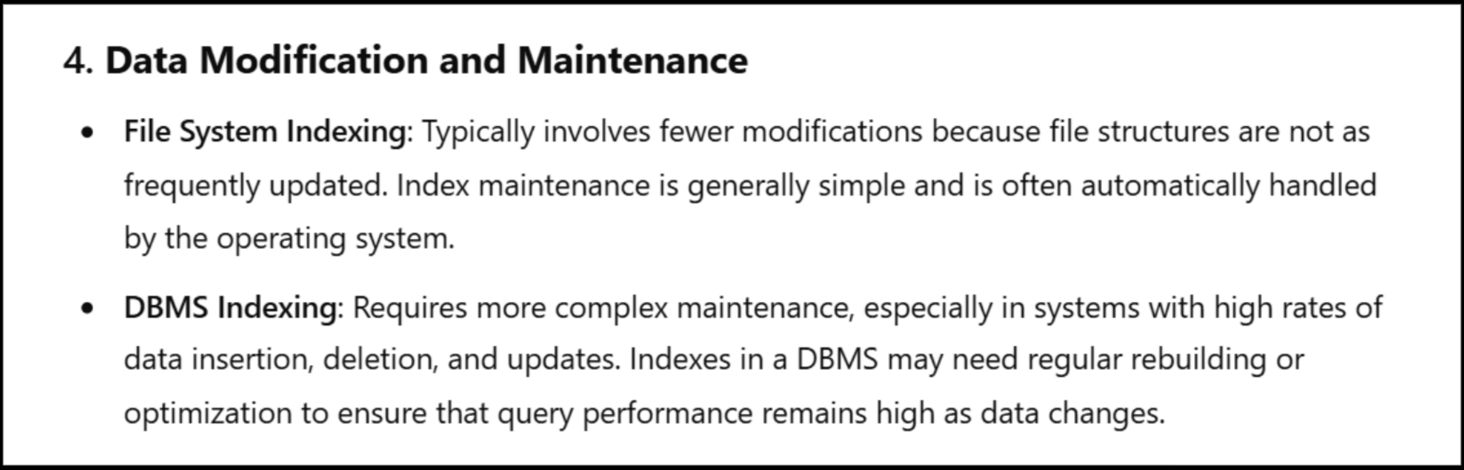
2. Data Structure

- **File System Indexing**: Usually uses simpler data structures, like hash tables or B-Trees, to store paths and metadata of files in a hierarchy or directory tree. Some file systems may employ a *master file table (MFT)*, as in NTFS (Windows), to keep track of file locations and metadata.
- **DBMS Indexing**: Primarily relies on B-Trees, B+ Trees, and, in some cases, hash indexes for faster query processing, especially with large datasets. DBMS indexing also supports specialized structures like bitmap and full-text indexes to handle diverse data types and large datasets.



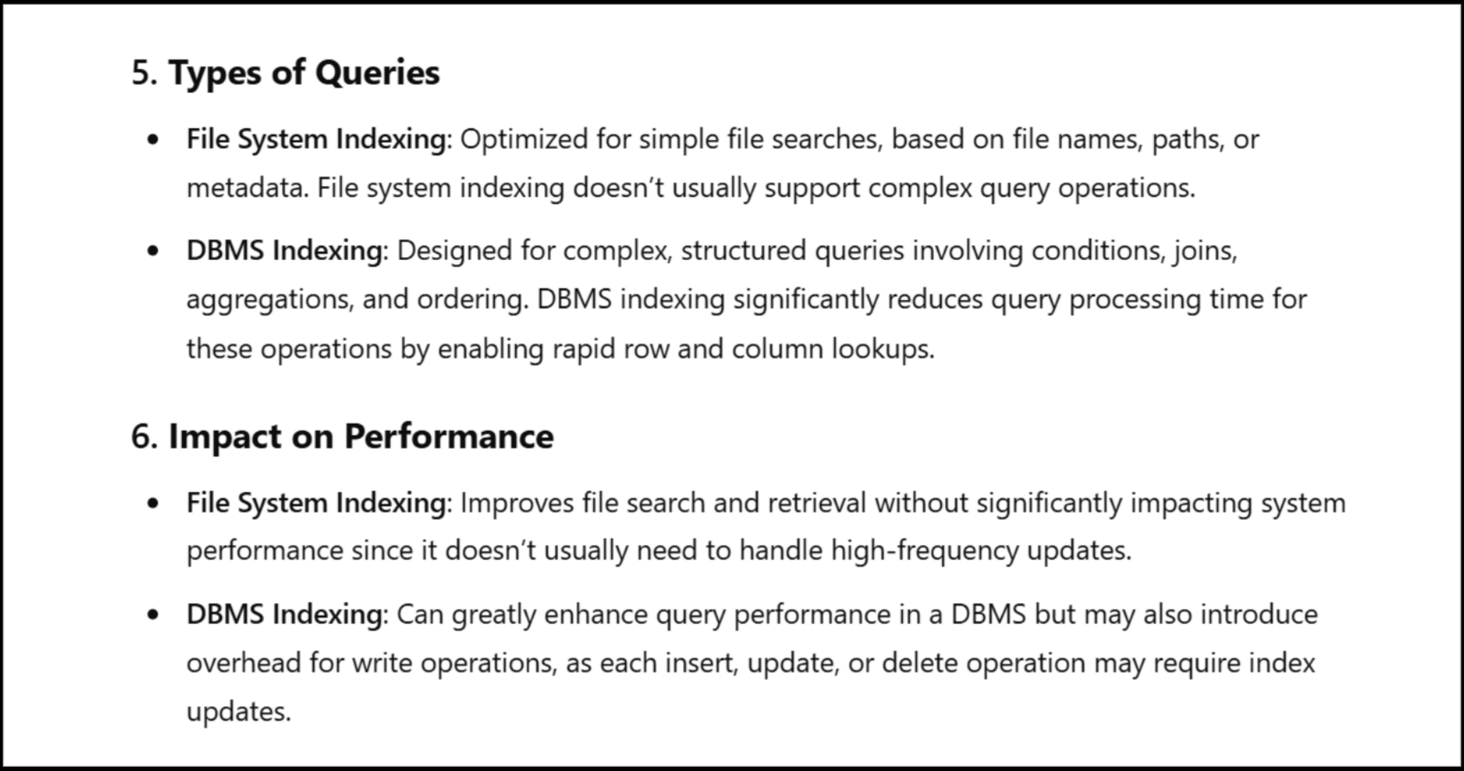
3. Indexing Focus and Granularity

- **File System Indexing**: Indexes at the file level. It is less granular, generally storing paths and file attributes, and sometimes limited text content for search (such as in modern search utilities). The index is mainly concerned with file properties and doesn't handle individual file content in depth.
- **DBMS Indexing**: Indexes at the row and column level, allowing for detailed, fine-grained access. In a DBMS, an index can be created on specific columns within a table, enabling more precise control over query performance.



4. Data Modification and Maintenance

- **File System Indexing**: Typically involves fewer modifications because file structures are not as frequently updated. Index maintenance is generally simple and is often automatically handled by the operating system.
- **DBMS Indexing**: Requires more complex maintenance, especially in systems with high rates of data insertion, deletion, and updates. Indexes in a DBMS may need regular rebuilding or optimization to ensure that query performance remains high as data changes.



5. Types of Queries

- **File System Indexing**: Optimized for simple file searches, based on file names, paths, or metadata. File system indexing doesn’t usually support complex query operations.
- **DBMS Indexing**: Designed for complex, structured queries involving conditions, joins, aggregations, and ordering. DBMS indexing significantly reduces query processing time for these operations by enabling rapid row and column lookups.

6. Impact on Performance

- **File System Indexing**: Improves file search and retrieval without significantly impacting system performance since it doesn’t usually need to handle high-frequency updates.
- **DBMS Indexing**: Can greatly enhance query performance in a DBMS but may also introduce overhead for write operations, as each insert, update, or delete operation may require index updates.