**Assignment 5**

**Demonstrate the creation of an index on a table and discuss how it improves query performance. Use a DROP INDEX statement to remove the index and analyze the impact on query execution.**

**Demonstration of Creating an Index**

Let's use the Books table from our library system to demonstrate the creation of an index.

**Create an Index**

We'll create an index on the title column of the Books table. This is useful for speeding up queries that search for books by their titles.

-Create an index on the title column

CREATE INDEX idx\_books\_title ON Books(title);

**How an Index Improves Query Performance**

Indexes are used to quickly locate data without having to search every row in a table. They function similarly to the index in a book, which allows you to find information quickly without having to read every page.

* Faster Search: When you query the Books table to find a book by its title, the database engine uses the index to directly locate the rows with the matching title, significantly speeding up the search.
* Efficiency: An index reduces the number of disk I/O operations. Instead of scanning the entire table, the database can quickly navigate the index to find the relevant rows.

Query Example:

- Query that benefits from the index

SELECT \* FROM Books WHERE title = 'The Great Gatsby';

Without an index, this query would require a full table scan, which is time-consuming, especially for large tables. With an index, the database engine can quickly find the rows where title = 'The Great Gatsby'.

**Dropping the Index**

Now, let's drop the index and analyze the impact on query execution.

-- Drop the index on the title column

DROP INDEX idx\_books\_title ON Books;

**Impact of Dropping the Index**

When the index is removed, queries that search by the title column will no longer benefit from the quick lookup provided by the index. This results in:

* Slower Search: The database engine will have to perform a full table scan to find rows that match the query condition, which involves reading each row sequentially until the match is found.
* Increased I/O: Full table scans increase the number of disk I/O operations, leading to higher resource consumption and longer query execution times.

**Comparison of Query Performance**

**With Index**

-- Execution plan before dropping the index

EXPLAIN SELECT \* FROM Books WHERE title = 'The Great Gatsby';

**Expected Result:**

* The execution plan shows the use of the index idx\_books\_title.
* The number of rows to be scanned is significantly reduced.

**Without Index**

-- Execution plan after dropping the index

EXPLAIN SELECT \* FROM Books WHERE title = 'The Great Gatsby';

**Expected Result:**

* The execution plan shows a full table scan (typically indicated by ALL in the type column).
* The number of rows to be scanned is the total number of rows in the table.

**Summary**

* Creating an Index: Improves query performance by allowing quick lookups and reducing the number of rows the database needs to scan.
* Dropping an Index: Results in slower query performance as the database must perform full table scans, increasing the number of disk I/O operations and overall query execution time.

Indexes are essential for optimizing database performance, especially for queries involving search conditions on large tables. However, they also come with trade-offs, such as increased storage requirements and slower performance for write operations (INSERT, UPDATE, DELETE), which must update the indexes as well. Therefore, indexes should be used judiciously based on query patterns and performance considerations.