SQL Internship – Task 11 Solution

Indexing and Query Performance Optimization

This document demonstrates how to improve SQL query performance using indexes. It includes queries without indexes, index creation scripts, EXPLAIN usage, and best practices.

SQL Script:

```
-- Sample Table
CREATE TABLE employees (
    emp_id INT PRIMARY KEY,
    emp_name VARCHAR(100),
    department VARCHAR(50),
    salary INT
);

-- Query WITHOUT index (slow)
SELECT * FROM employees WHERE department = 'IT';

-- Create Index on frequently searched column
CREATE INDEX idx_department ON employees(department);

-- Query WITH index (faster)
SELECT * FROM employees WHERE department = 'IT';

-- Use EXPLAIN to analyze query plan
EXPLAIN SELECT * FROM employees WHERE department = 'IT';

-- Clustered vs Non-Clustered (MySQL uses clustered index on PRIMARY KEY)
-- Primary key index
ALTER TABLE employees ADD PRIMARY KEY(emp_id);

-- Non-clustered index example
CREATE INDEX idx_salary ON employees(salary);
```

Best Indexing Practices:

- Create indexes on frequently searched columns (WHERE, JOIN, ORDER BY).
- Avoid indexes on very small tables.
- Too many indexes slow down INSERT, UPDATE, DELETE operations.
- Use EXPLAIN to check query execution plan.
- Composite indexes help when filtering on multiple columns.

Interview Questions & Answers: 1. What is an index? An index is a data structure that improves the speed of data retrieval operations. 2. Why do indexes improve performance? They reduce the number of rows scanned by storing sorted references to data. 3. When should indexes be avoided? On small tables or frequently updated columns because they slow write operations. 4. What is EXPLAIN used for? It shows how MySQL executes a query and whether indexes are used. 5. Difference between clustered and non-clustered index? Clustered index stores data physically in sorted order, while non-clustered stores pointers to data.