SQL injection is a security vulnerability that occurs when an attacker manipulates a SQL query by injecting malicious input. This allows the attacker to modify the query's behavior and potentially gain unauthorized access to sensitive data. Common forms of SQL injections include using logical operators like `OR 1=1`, comments like `--`, query terminators like `;`, or manipulating string literals with `'`. Detecting these malicious patterns is crucial for preventing unauthorized data access.

In the `run\_query` method, the goal is to detect these signs of SQL injection before executing the query. To do this, the SQL query is first converted to lowercase, which makes it easier to detect injection attempts regardless of case. The method then searches for common suspicious patterns, such as `OR`, `--`, `;`, and `'`, using `std::string::find`. If any of these patterns are found, the method flags the query as potentially dangerous and halts execution by returning `false`. This approach helps in preventing basic SQL injection attacks from being executed on the database.

While detecting SQL injection patterns is a useful safeguard, the most effective way to prevent SQL injection is to use parameterized queries or prepared statements. These techniques separate the SQL code from user input, ensuring that any input provided by the user cannot alter the structure of the SQL query. By doing so, SQL injection vulnerabilities are eliminated.

In summary, by detecting suspicious SQL patterns and halting execution of potentially harmful queries, this method helps protect the database from SQL injections. However, for more robust protection, it is recommended to use parameterized queries or prepared statements to fully prevent SQL injection attacks. This ensures that user input is handled securely, safeguarding sensitive information.

A screenshot of a computer program

Description automatically generated