**HẠN CHẾ TẤN CÔNG SQL INJECTION**

# Lựa chọn phương án hạn chế tấn công SQL Injection

1. **Khái niệm SQL Injection:**

* SQL Injection là một kỹ thuật lợi dụng những lỗ hổng về câu truy vấn của các ứng dụng. Được thực hiện bằng cách chèn thêm một đoạn SQL để làm sai lệch đi câu truy vấn ban đầu, từ đó có thể khai thác dữ liệu từ database. SQL injection có thể cho phép những kẻ tấn công thực hiện các thao tác như một người quản trị web, trên cơ sở dữ liệu của ứng dụng.
* Đây là một trong những lỗ hổng bảo mật phổ biến và nguy hiểm nhất, thường xảy ra khi các ứng dụng xử lý các truy vấn SQL mà không kiểm tra hoặc lọc dữ liệu đầu vào từ người dùng

1. **Các cách để hạn chế tấn công SQL Injection:**

* Sử dụng lớp trừu tượng (abstraction layer) -> tạo ra một schema chỉ chứa các view,  các kết nối từ ứng dụng chỉ thao tác được trên view -> không thể xóa hay thay đổi được dữ liệu: Để giảm thiểu rủi ro SQL Injection, hãy áp dụng tiêu chuẩn lưu trữ yêu cầu các ứng dụng sử dụng lớp trừu tượng khi truy cập dữ liệu.
* Chỉ cho phép truy cập thông qua stored procedures: Đảm bảo rằng việc truy cập công cụ cơ sở dữ liệu chỉ được thực hiện qua các stored procedure, thay vì sử dụng các câu lệnh SQL tự do (Ad-hoc SQL statements).
* Việc áp dụng những biện pháp này sẽ giúp tăng cường bảo mật cho hệ thống cơ sở dữ liệu, giảm thiểu rủi ro từ các cuộc tấn công SQL Injection.
* Mã hóa dữ liệu

1. **Phương án lựa chọn:**

Nhóm quyết định lựa chọn phương án Sử dụng thủ tục lưu trữ (Stored Procedures)và mã hóa dữ liệu để hạn chế tấn công SQL Injection vì nhận thấy được một số lợi ích mà các phương án này mang lại như:

1. **Sử dụng thủ tục lưu trữ**

* Thủ tục lưu trữ giúp tách biệt logic ứng dụng khỏi truy vấn SQL. Điều này giảm nguy cơ việc chuỗi truy vấn SQL bị chèn mã độc hại từ phía người dùng, bởi các tham số truyền vào thủ tục được xử lý trước khi thực thi.
* Thủ tục lưu trữ nhận tham số đầu vào, giúp loại bỏ hoàn toàn việc phải nối chuỗi trực tiếp trong truy vấn SQL. Điều này ngăn chặn kẻ tấn công chèn mã độc (payload) thông qua đầu vào người dùng
* Tăng hiệu năng và tối ưu hóa truy vấn
* Dễ bảo trì và quản lý: Với các dự án lớn, việc quản lý các thủ tục lưu trữ tập trung trong cơ sở dữ liệu dễ dàng hơn so với việc sửa đổi logic SQL phân tán trong ứng dụng. Thủ tục lưu trữ có thể tái sử dụng, giúp giảm bớt khối lượng mã cần viết và kiểm thử
* Tăng cường kiểm soát và bảo mật:
* Quản trị viên có thể áp dụng các quyền hạn cụ thể cho thủ tục lưu trữ, đảm bảo chỉ người dùng hoặc ứng dụng được cấp quyền mới có thể gọi thủ tục.
* Các thủ tục có thể được sử dụng để che giấu cấu trúc cơ sở dữ liệu, giúp giảm thiểu khả năng kẻ tấn công tìm hiểu được chi tiết bảng, cột hoặc logic liên quan.

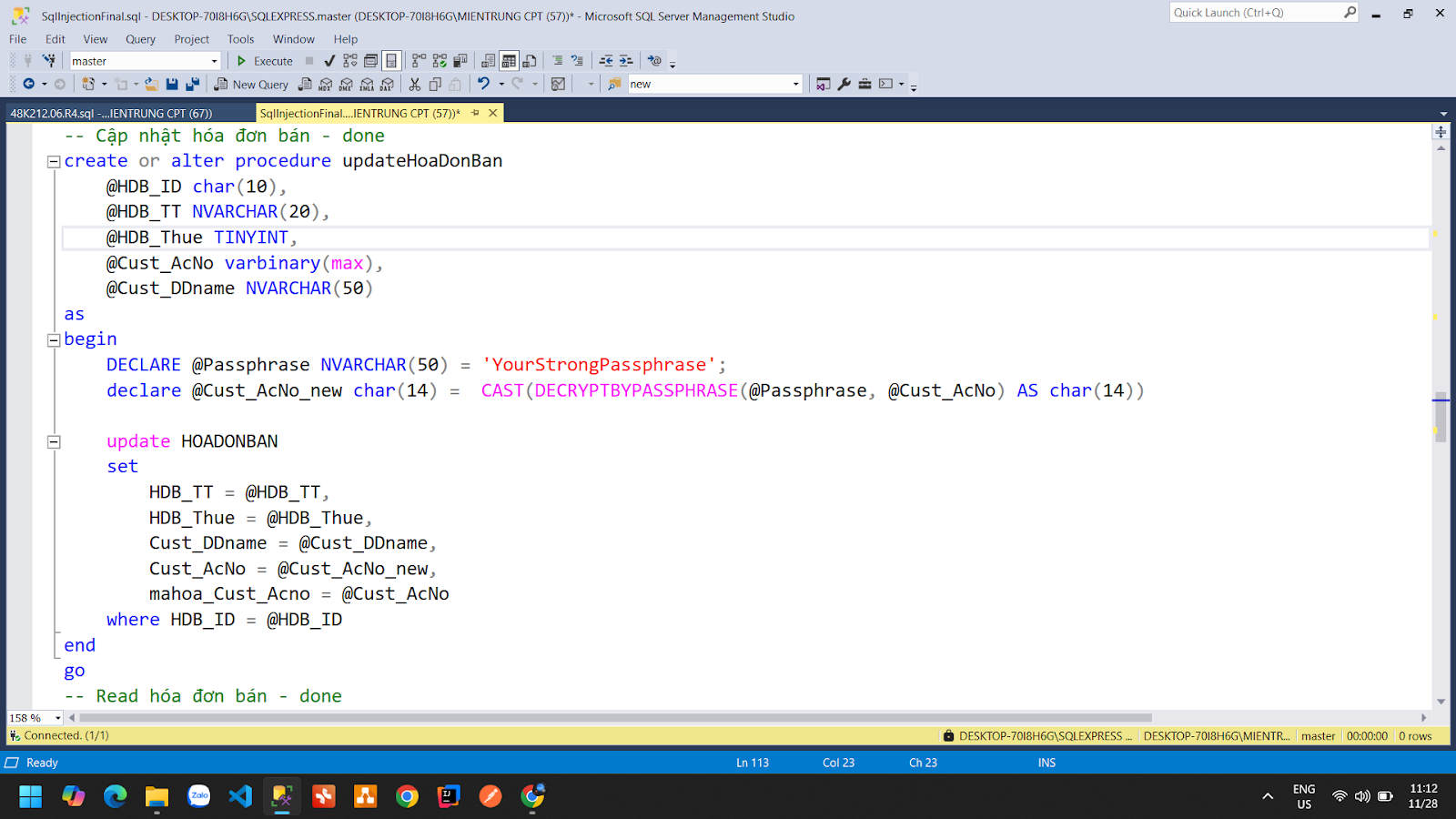
1. **Sử dụng mã hóa dữ liệu**

* Bảo vệ dữ liệu nhạy cảm:
  + Dữ liệu nhạy cảm như mật khẩu, số thẻ tín dụng, hoặc thông tin cá nhân thường là mục tiêu chính của các cuộc tấn công SQL Injection
  + Khi dữ liệu được mã hóa trong cơ sở dữ liệu, ngay cả khi kẻ tấn công thực hiện thành công một cuộc tấn công SQL Injection, chúng sẽ chỉ thấy dữ liệu mã hóa vô nghĩa thay vì nội dung thực
* Giảm thiểu tác hại từ lỗ hổng ứng dụng: Trong một số trường hợp, ứng dụng có thể không xử lý đầu vào người dùng một cách an toàn, tạo ra nguy cơ SQL Injection. Nếu dữ liệu đã được mã hóa trước khi lưu vào cơ sở dữ liệu, dữ liệu bị đánh cắp vẫn không thể sử dụng ngay cả khi ứng dụng có lỗ hổng.
* Hạn chế truy cập không được phép: Khi dữ liệu được mã hóa, việc giải mã thường yêu cầu khóa mã hóa hoặc các quyền đặc biệt. Điều này giúp ngăn chặn việc sử dụng trái phép dữ liệu ngay cả khi kẻ tấn công truy cập vào cơ sở dữ liệu
* Ngăn chặn tấn công tái sử dụng dữ liệu (Replay Attack): Trong trường hợp kẻ tấn công đánh cắp được dữ liệu từ các truy vấn SQL, dữ liệu đã được mã hóa sẽ không thể được sử dụng để thực hiện các tấn công tiếp theo

# Thực hiện phương án

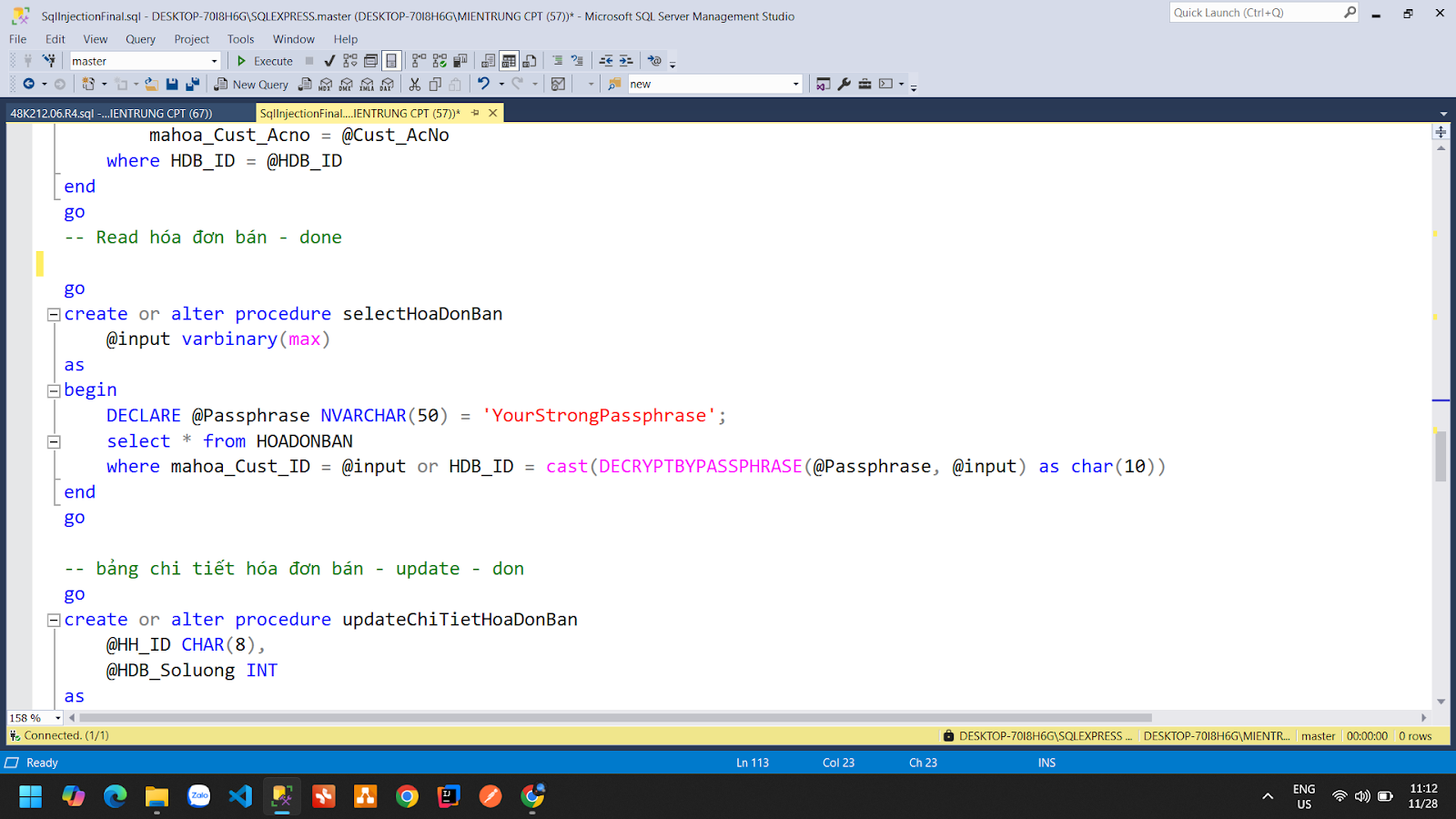
Các bước triển khai phương án sử dụng thủ tục lưu trữ và mã hóa dữ liệu để hạn chế tấn công SQL Injection được thực hiện như sau:

1. Cập nhật bảng HOADONBAN



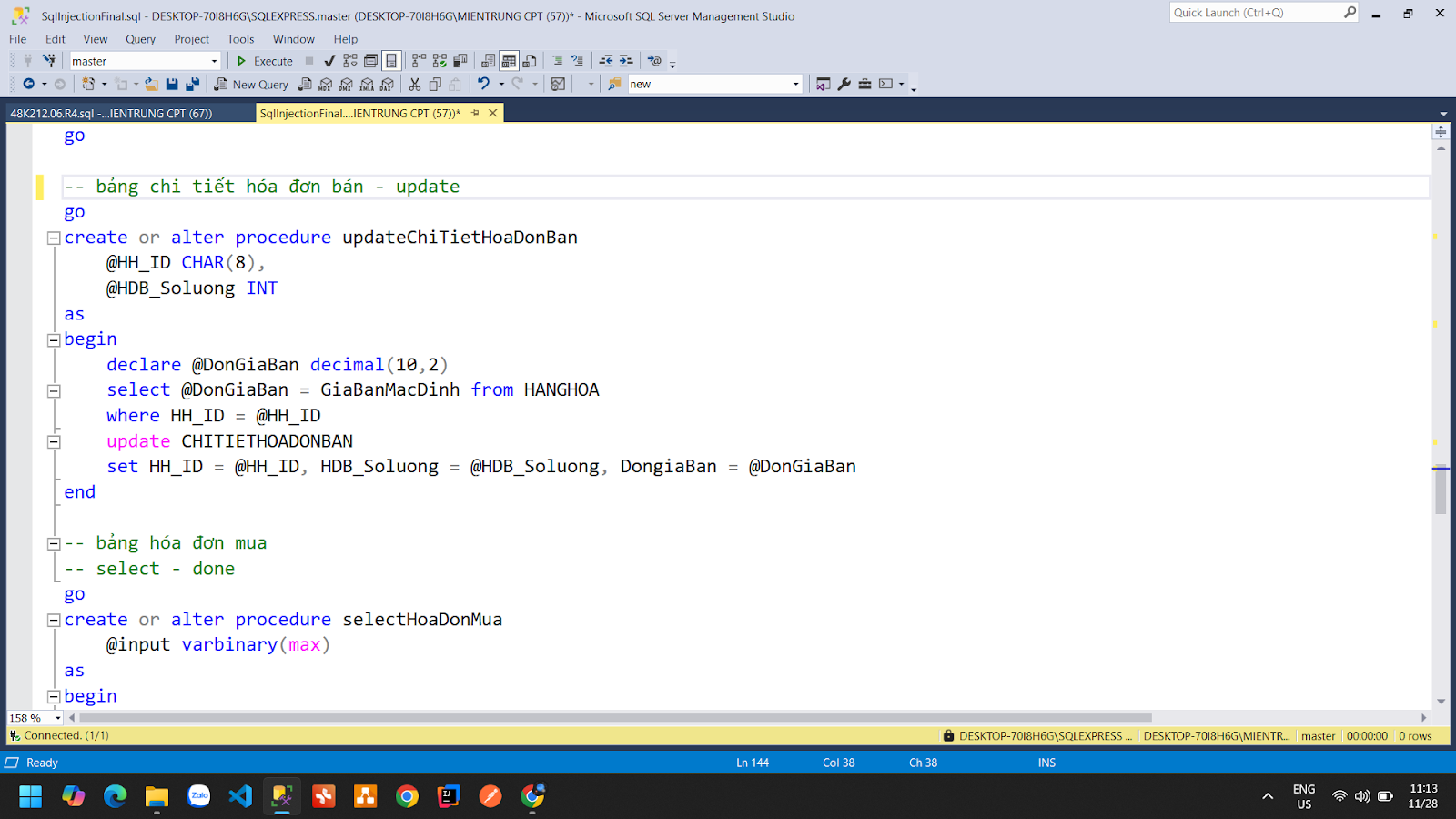
Hình 1. Demo thực hiện hạn chế tấn công SQL Injection

1. Tìm kiếm dữ liệu trong bảng HOADONBAN



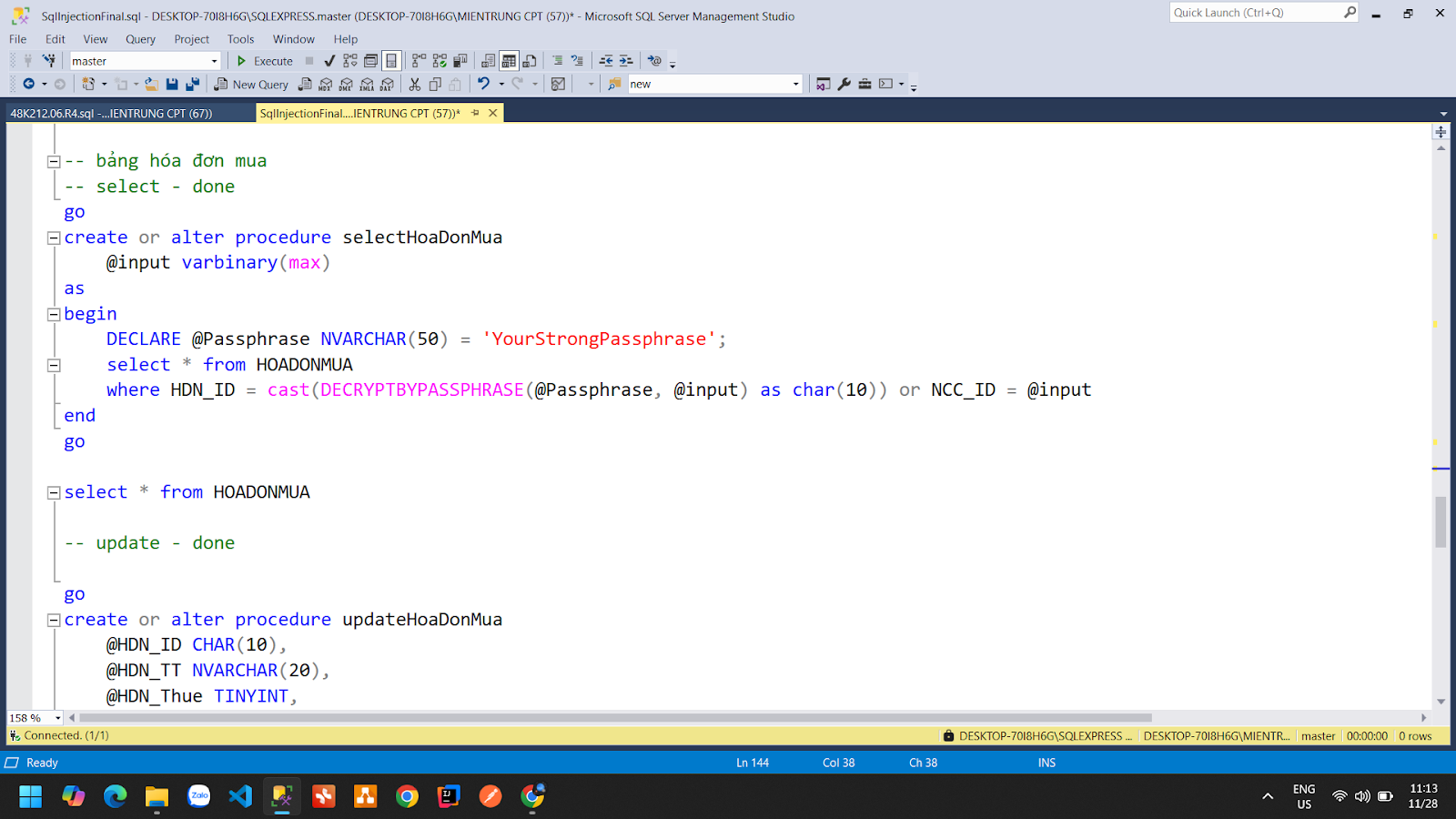
Hình 2. Demo thực hiện hạn chế tấn công SQL Injection

1. Cập nhật bảng CHITIETHOADONBAN



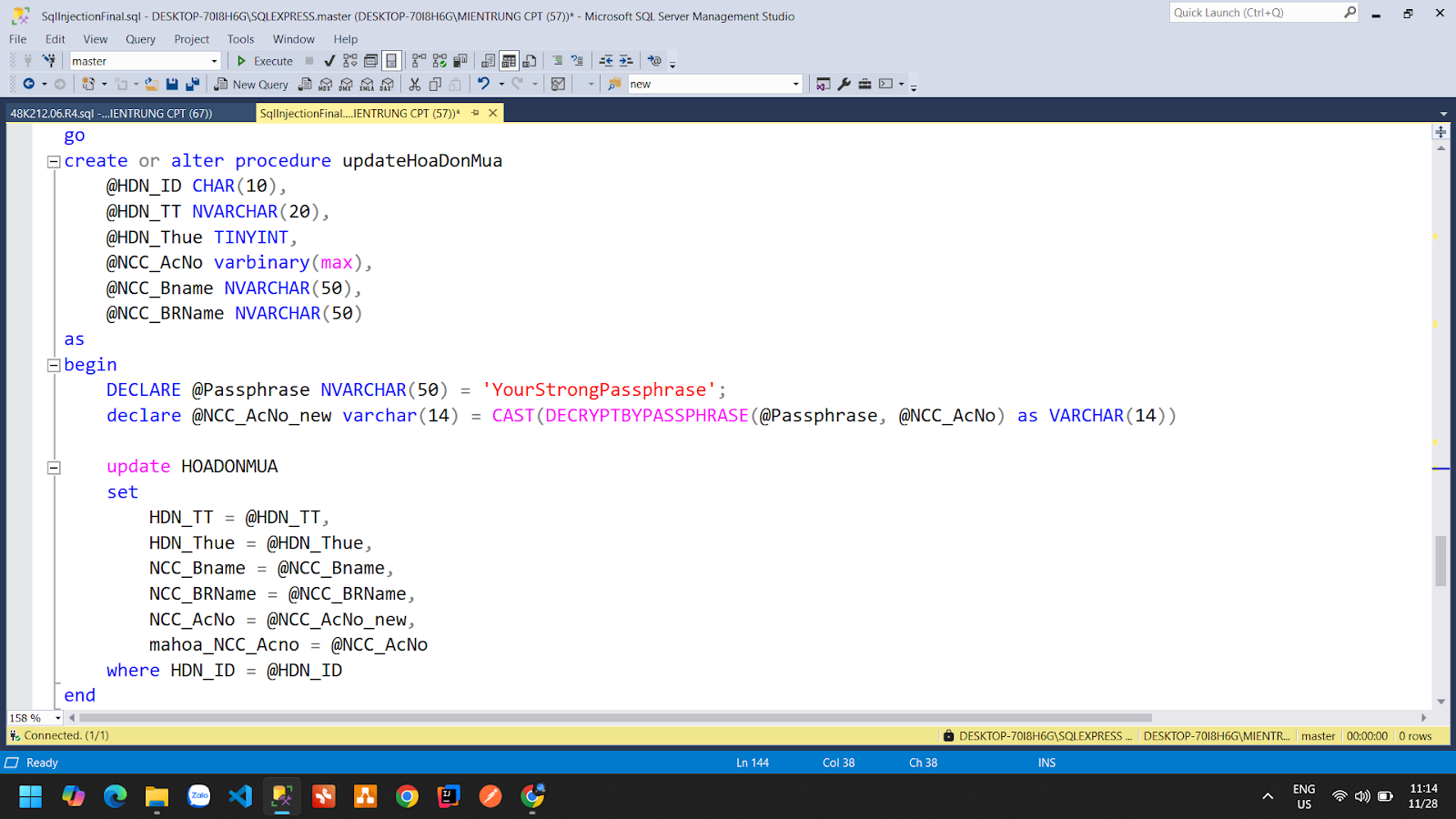
Hình 3. Demo thực hiện hạn chế tấn công SQL Injection

1. Tìm kiếm dữ liệu trong bảng HOADONMUA



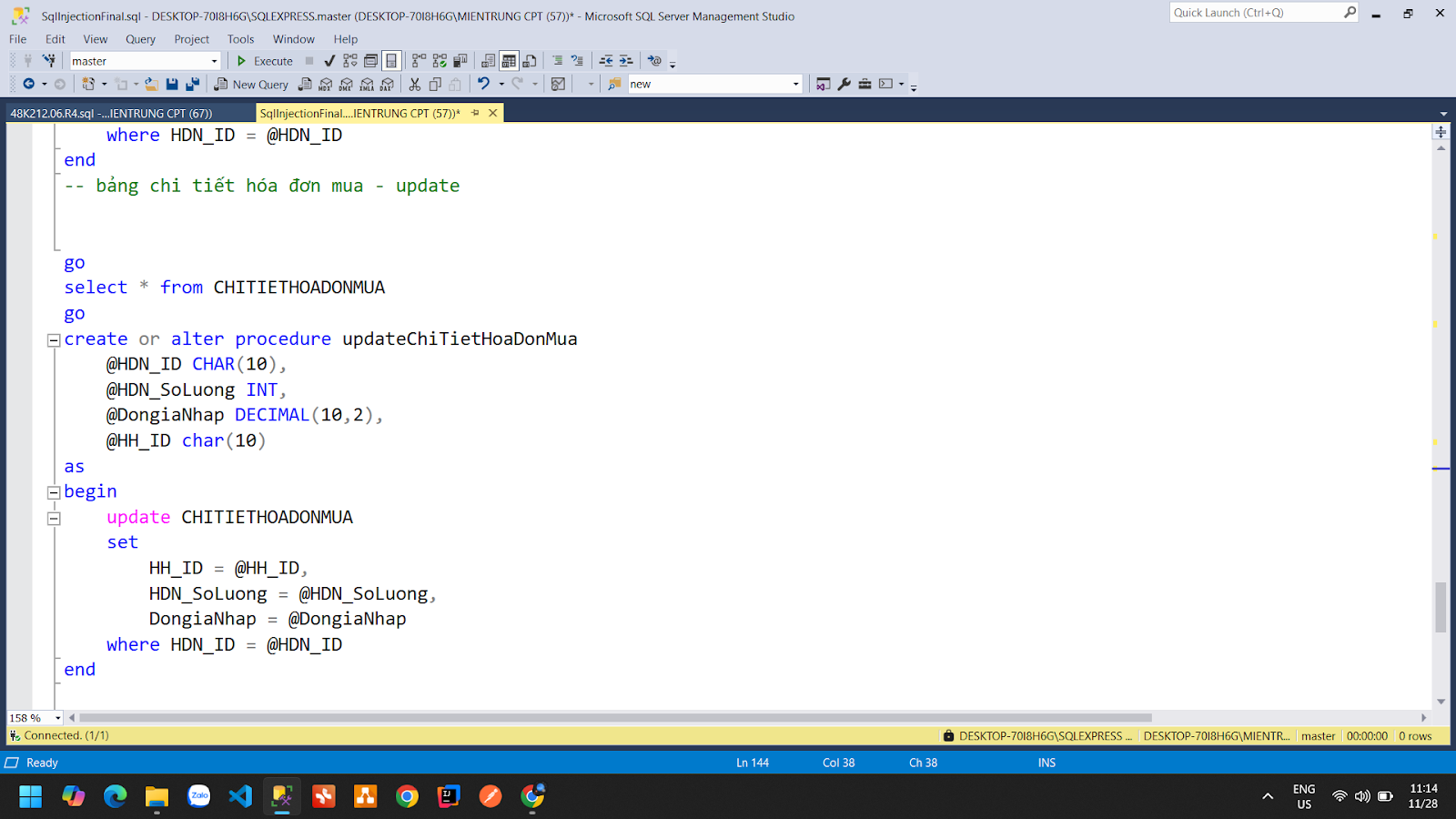
Hình 4. Demo thực hiện hạn chế tấn công SQL Injection

1. Cập nhật bảng HOADONMUA



Hình 5. Demo thực hiện hạn chế tấn công SQL Injection

1. Cập nhật bảng CHITIETHOADONMUA



Hình 6. Demo thực hiện hạn chế tấn công SQL Injection

1. Đối với thủ tục Xóa, Tạo mới các bảng
   1. Thủ tục xóa:

* Không sử dụng đầu vào là văn bản
* Lưu ý: Một số trường hợp khác cũng không nhận đầu vào là văn bản nên không thực hiện chống SQL Injection
  1. Thủ tục tạo mới: Được sử dụng triển khai ở phần xây dựng 10 module xử lý nghiệp vụ phức tạp

1. Một số đề xuất triển khai khác (R9) của nhóm để chống SQL Injection:

* Kiểm tra dữ liệu đầu vào ở tầng Frontend và Backend
* Tham số hóa dữ liệu đầu vào ở Backend thông qua một số thư viện