SECURITY POLICY SP-REF-001

SQL Injection Prevention Policy

Document ID:	SP-REF-001
Version:	1.0
Date:	2025-11-07
Classification:	Internal Use
Owner:	Information Security Team

1. EXECUTIVE SUMMARY

This security policy establishes mandatory requirements for preventing SQL injection vulnerabilities in all software applications developed, maintained, or deployed by the organization. SQL injection represents one of the most critical web application security risks, capable of compromising data confidentiality, integrity, and availability. This policy mandates the use of parameterized queries, input validation, and secure coding practices across all database interactions.

2. PURPOSE

The purpose of this policy is to:

• Eliminate SQL injection vulnerabilities from all application codebases

• Establish secure coding standards for database interactions

• Ensure compliance with industry security frameworks and regulations

• Protect sensitive data from unauthorized access, modification, or deletion

• Maintain the integrity and availability of database systems

3. SCOPE

This policy applies to:

• All web applications, APIs, and services that interact with databases

• All software development teams and contractors

• Legacy applications undergoing maintenance or updates

• Third-party integrations that perform database queries

• Development, staging, and production environments

4. RISK STATEMENT

SQL injection vulnerabilities allow attackers to execute arbitrary database queries through unsanitized user input, potentially leading to:

• Unauthorized data access: Retrieval of sensitive customer data, credentials, and intellectual property

• Data modification: Alteration or deletion of critical business data

• Authentication bypass: Circumventing login mechanisms and access controls

• Privilege escalation: Gaining administrative access to database systems

• System compromise: Executing operating system commands in certain database configurations

The impact of successful SQL injection attacks can result in regulatory penalties, reputational damage, financial loss, and legal liability.

5. POLICY STATEMENT

All database interactions must implement the following mandatory security controls:

5.1 Parameterized Queries

All SQL queries must use parameterized statements or prepared statements. String concatenation of user input into SQL queries is strictly prohibited. Use of Object-Relational Mapping (ORM) frameworks with parameterized queries is recommended.

5.2 Input Validation

Implement strict input validation on all user-supplied data before database operations. Validation must include type checking, length restrictions, format verification, and whitelisting of acceptable characters. Reject invalid input rather than attempting sanitization.

5.3 Least Privilege Access

Database accounts used by applications must follow the principle of least privilege. Grant only the minimum permissions necessary for application functionality. Avoid using administrative or root database accounts in application connections.

5.4 Error Handling

Database error messages must not expose sensitive information such as table names, column names, or query structure. Implement generic error messages for end users while logging detailed errors securely for debugging purposes.

6. COMPLIANCE MAPPING

This policy aligns with the following security frameworks and standards:

Framework	Control ID	Control Title
NIST CSF	PR.DS-5	Protections against data leaks are implemented
NIST CSF	PR.AC-4	Access permissions and authorizations are managed
NIST CSF	DE.CM-1	The network is monitored to detect potential cybersecurity events
ISO 27001	A.14.2.5	Secure system engineering principles
ISO 27001	A.8.22	Segregation in networks
ISO 27001	A.8.3	Handling of assets
OWASP Top 10	A03:2021	Injection
PCI DSS	6.5.1	Injection flaws, particularly SQL injection

7. REMEDIATION PLAN

Upon detection of SQL injection vulnerabilities, the following remediation process must be executed:

Phase	Action	Timeline	Responsible Party
1. Immediate	Notify security team and development lead	0-2 hours	Security Operations
2. Assessment	Analyze vulnerability scope and impact	2-8 hours	AppSec Team
3. Mitigation	Deploy WAF rules if immediate fix not possible	8-12 hours	Security Engineering
4. Development	Implement parameterized queries in code	24-48 hours	Development Team
5. Testing	Conduct code review and security testing	48-72 hours	QA & Security
6. Deployment	Deploy fix to production environment	72-96 hours	DevOps Team
7. Verification	Re-scan and validate vulnerability closed	96-120 hours	Security Operations

8. ROLES AND RESPONSIBILITIES

Role	Responsibilities
Development Team	• Implement parameterized queries in all code • Follow secure coding standards • Conduct peer code reviews
Security Team	• Perform vulnerability assessments • Provide security training • Monitor for SQL injection attempts

QA Team	• Test for SQL injection vulnerabilities • Validate remediation effectiveness • Verify compliance with policy
DevOps Team	• Deploy security patches promptly • Configure WAF rules • Maintain audit logging
Management	• Allocate resources for security initiatives • Enforce policy compliance • Review policy effectiveness quarterly

9. MONITORING AND DETECTION

The following monitoring controls must be implemented to detect SQL injection attempts:

Web Application Firewall (WAF): Deploy WAF rules to detect and block SQL injection patterns in real-time

Database Activity Monitoring (DAM): Implement DAM solutions to track and alert on suspicious query patterns

SIEM Integration: Forward security logs to SIEM for correlation and alerting on injection attempts

Automated Scanning: Conduct quarterly SAST and DAST scans to identify vulnerabilities

Penetration Testing: Perform annual penetration testing focused on injection vulnerabilities

Security Logging: Enable comprehensive logging of all database queries and access attempts

10. SUCCESS CRITERIA

Policy effectiveness will be measured by the following success criteria:

• Zero SQL injection vulnerabilities detected in production environments

• 100% of database queries use parameterized statements (verified through code audits)

• All developers complete secure coding training annually

• WAF successfully blocks 100% of simulated SQL injection attacks during testing

• Database access follows least privilege principle (verified through access reviews)

• No security incidents related to SQL injection in the past 12 months

11. ENFORCEMENT

Non-compliance with this policy may result in:

• Code deployment blocked until vulnerabilities are remediated

• Mandatory security training for development team members

• Escalation to management for repeated violations

• Performance review implications for persistent non-compliance

• Termination of vendor contracts for third-party violations

12. REVIEW AND UPDATES

This policy will be reviewed and updated:

• Annually by the Information Security team

• Following any security incident related to SQL injection

• When significant changes occur to application architecture

• Upon updates to compliance frameworks (NIST CSF, ISO 27001, etc.)

• Based on feedback from development teams and security assessments

13. REFERENCES

• OWASP Top 10 2021: A03 Injection - https://owasp.org/Top10/A03_2021-Injection/

• NIST Cybersecurity Framework v1.1

• ISO/IEC 27001:2022 - Information security management

• PCI DSS v4.0 - Payment Card Industry Data Security Standard

• CWE-89: SQL Injection - https://cwe.mitre.org/data/definitions/89.html

• OWASP SQL Injection Prevention Cheat Sheet

Document Generated: 2025-11-07 16:24:18
Classification: Internal Use Only
Approved By: Chief Information Security Officer