# CWE Detail – CWE-943

## Description

The product generates a query intended to access or manipulate data in a data store such as a database, but it does not neutralize or incorrectly neutralizes special elements that can modify the intended logic of the query.

## Extended Description

Depending on the capabilities of the query language, an attacker could inject additional logic into the query to: Modify the intended selection criteria, thus changing which data entities (e.g., records) are returned, modified, or otherwise manipulated Append additional commands to the query Return more entities than intended Return fewer entities than intended Cause entities to be sorted in an unexpected way The ability to execute additional commands or change which entities are returned has obvious risks. But when the product logic depends on the order or number of entities, this can also lead to vulnerabilities. For example, if the query expects to return only one entity that specifies an administrative user, but an attacker can change which entities are returned, this could cause the logic to return information for a regular user and incorrectly assume that the user has administrative privileges. While this weakness is most commonly associated with SQL injection, there are many other query languages that are also subject to injection attacks, including HTSQL, LDAP, DQL, XQuery, Xpath, and "NoSQL" languages.

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** CVE-2024-50672: NoSQL injection in product for building eLearning courses allows password resets using a query processed by the Mongoose find function

**•** CVE-2021-20736: NoSQL injection in team collaboration product

**•** CVE-2020-35666: NoSQL injection in a PaaS platform using a MongoDB operator

**•** CVE-2014-2503: Injection using Documentum Query Language (DQL)

**•** CVE-2014-2508: Injection using Documentum Query Language (DQL)

## Related Attack Patterns (CAPEC)

* CAPEC-676

## Modes of Introduction

**•** Implementation: REALIZATION: This weakness is caused during implementation of an architectural security tactic.

## Common Consequences

**•** Impact: Bypass Protection Mechanism, Read Application Data, Modify Application Data, Varies by Context — Notes:

## Applicable Platforms

**•** None (Class: Not Language-Specific, Prevalence: Undetermined)

## Demonstrative Examples

**•** The query that this code intends to execute follows:

**•** Because the code fails to neutralize the address string used to construct the query, an attacker can supply an address that includes additional LDAP queries.

**•** The Java code used to retrieve the home directory based on the provided credentials is:

## Notes

**•** Relationship: It could be argued that data query languages are effectively a command language - albeit with a limited set of commands - and thus any query-language injection issue could be treated as a child of CWE-74. However, CWE-943 is intended to better organize query-oriented issues to separate them from fully-functioning programming languages, and also to provide a more precise identifier for the many query languages that do not have their own CWE identifier.