















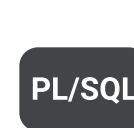


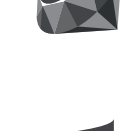




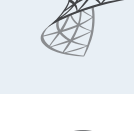









-  Secrets
-  ABAP
-  Apex
-  C
-  C++
-  CloudFormation
-  COBOL
-  C#
-  CSS
-  Flex
-  Go
-  HTML
-  Java
-  JavaScript
-  Kotlin
-  Kubernetes
-  Objective C
-  PHP
-  PL/I
-  PL/SQL
-  Python
-  RPG
-  Ruby
-  Scala
-  Swift
-  Terraform
-  Text
-  TypeScript
-  **T-SQL**
-  VB.NET
-  VB6
-  XML














T-SQL static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your T-SQL code

- All rules 80
-  Vulnerability 1
-  Bug 16
-  Security Hotspot 4
-  Code Smell 59

Tags 

Search by name... 

String literals should not be duplicated	 Code Smell
Expressions should not be too complex	 Code Smell
"WHERE" clauses should not contain redundant conditions	 Bug
Track lack of SQL Server session configuration	 Code Smell
Duplicate values should not be passed as arguments	 Code Smell
Track parsing failures	 Code Smell
Functions and stored procedure should not have too many lines of code	 Code Smell
Track uses of "NOSONAR" comments	 Code Smell
Statements should be on separate lines	 Code Smell
"WHEN" clauses should not have too many lines of code	 Code Smell
Files should not have too many lines of code	 Code Smell
Lines should not be too long	

String literals should not be duplicated

[Analyze your code](#)

 Code Smell  Critical   design

Duplicated string literals make the process of refactoring error-prone, since you must be sure to update all occurrences.

On the other hand, constants can be referenced from many places, but only need to be updated in a single place.

Noncompliant Code Example

With the default threshold of 3:

```
IF @x='Yes '
    SELECT ...
        FROM ...
        WHERE field='Yes '
...
...
IF @x='Yes '
    ...
```

Compliant Solution

```
DECLARE @Yes VARCHAR(3) = 'Yes '
IF @x=@Yes
    SELECT ...
        FROM ...
        WHERE field=@Yes
...
...
IF @x=@Yes
    ...
```

Exceptions

To prevent generating some false-positives, literals having less than 5 characters are excluded.

Available In:

sonarlint  | **sonarcloud**  | **sonarqube**  Developer Edition