


















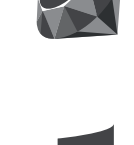




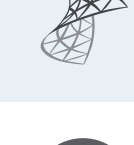









-  Secrets
-  ABAP
-  Apex
-  C
-  C++
-  CloudFormation
-  COBOL
-  C#
-  CSS
-  Flex
-  Go
-  HTML
-  Java
-  JavaScript
-  Kotlin
-  Kubernetes
-  Objective C
-  PHP
-  PL/I
-  PL/SQL
-  Python
-  RPG
-  Ruby
-  Scala
-  Swift
-  Terraform
-  Text
-  TypeScript
-  **T-SQL**
-  VB.NET
-  VB6
-  XML














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... 

Boolean checks should not be inverted
 Code Smell
Multiple variables should not be declared on the same line
 Code Smell
Unused local variables should be removed
 Code Smell
Local variable and parameter names should comply with a naming convention
 Code Smell
Empty statements should be removed
 Code Smell
Track uses of "TODO" tags
 Code Smell
A primary key should be specified during table creation
 Code Smell
Track lack of copyright and license headers
 Code Smell
SHA-1 and Message-Digest hash algorithms should not be used in secure contexts
 Vulnerability
"NOCOUNT" should be activated on "PROCEDURE" and "TRIGGER" definitions
 Code Smell
Control flow statements "IF", "WHILE" and "TRY" should not be nested too deeply
 Code Smell
"CASE" expressions should end with "ELSE" clauses

Boolean checks should not be inverted

 Code Smell  Minor   pitfall

It is needlessly complex to invert the result of a boolean comparison. The opposite comparison should be made instead.

Noncompliant Code Example




```
IF NOT (@a = 2) -- Noncompliant
BEGIN
    ...
END

IF NOT (@b < 10) -- Noncompliant
BEGIN
    ...
END
```

Compliant Solution

```
IF @a <> 2
BEGIN
    ...
END

IF @b >= 10
BEGIN
    ...
END
```

Available In:
 |  |  Developer Edition