


































-  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... 

Related "IF"/"ELSE IF" statements and "WHEN" clauses in a "CASE" should not have the same condition

 Bug

Identical expressions should not be used on both sides of a binary operator

 Bug

All code should be reachable

 Bug

Loops with at most one iteration should be refactored

 Bug

Variables should not be self-assigned

 Bug

"GOTO" statements should not be used

 Code Smell

Deprecated system tables and views should not be used

 Code Smell

"ANSI_NULLS", "ANSI_PADDING" and "CONCAT_NULL_YIELDS_NULL" should not be configured

 Code Smell

"@@IDENTITY" should not be used

 Code Smell

"COALESCE", "IIF", and "CASE" input expressions should not contain subqueries

 Code Smell

"CHECK" or "NOCHECK" should be specified explicitly when constraints are activated

 Code Smell

Related "IF"/"ELSE IF" statements and "WHEN" clauses in a "CASE" should not have the same condition

Analyze your code

 Bug  Major   unused pitfall

A `CASE` and a chain of `IF/ELSE IF` statements is evaluated from top to bottom. At most, only one branch will be executed: the first one with a condition that evaluates to `true`.

Therefore, duplicating a condition automatically leads to dead code. Usually, this is due to a copy/paste error. At best, it's simply dead code and at worst, it's a bug that is likely to induce further bugs as the code is maintained, and obviously it could lead to unexpected behavior.

Noncompliant Code Example

```
IF @x = 1
    PRINT 'A'
ELSE IF @x = 2
    PRINT 'B'
ELSE IF @x = 1 -- Noncompliant
    PRINT 'C'

SELECT
    CASE coll
        WHEN 1
            THEN 'A'
        WHEN 2
            THEN 'B'
        WHEN 1 -- Noncompliant
            THEN 'C'
        ELSE 'D'
    END
FROM table1
```

Compliant Solution

```
IF @x = 1
    PRINT 'A'
ELSE IF @x = 2
    PRINT 'B'
ELSE IF @x = 3
    PRINT 'C'

SELECT
    CASE coll
        WHEN 1
            THEN 'A'
        WHEN 2
            THEN 'B'
        WHEN 3
            THEN 'C'
        ELSE 'D'
    END
FROM table1
```

See

- [CERT, MSC12-C](#). - Detect and remove code that has no effect or is never executed

Available In:

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