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

PL/SQL

PL/SQL static code analysis

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

All rules 188 | Vulnerability 4 | Bug 45 | Security Hotspot 2 | Code Smell 137

Tags

Search by name...

Identifiers should be written in lower case
Code Smell

"PLS_INTEGER" types should be used
Code Smell

Reserved words should be written in upper case
Code Smell

Parameter "IN" mode should be specified explicitly
Code Smell

Lines in a multiline comment should start with "*"
Code Smell

CASE should be used for sequences of simple tests
Code Smell

SQL tables should be joined with the "JOIN" keyword
Code Smell

Constraint names should comply with a naming convention
Code Smell

Reserved words should be written in lower case
Code Smell

Comments should not be located at the end of lines of code
Code Smell

Lines should not end with trailing whitespaces
Code Smell

The "RELIES_ON" clause should not be used
Code Smell

CASE should be used for sequences of simple tests

Analyze your code

Code Smell | Minor | clumsy

When a single primitive is tested against three or more values in an `IF`, `ELSIF` chain, it should be converted to a `CASE` instead for greater readability.

Noncompliant Code Example

```
DECLARE
  x PLS_INTEGER := 0;
BEGIN
  IF x = 0 THEN                          -- Noncompliant
    DBMS_OUTPUT.PUT_LINE('x = 0');
  ELSIF x = 1 THEN
    DBMS_OUTPUT.PUT_LINE('x = 1');
  ELSE
    DBMS_OUTPUT.PUT_LINE('x > 1');
  END IF;
END;
/
```

Compliant Solution

```
DECLARE
  x PLS_INTEGER := 0;
BEGIN
  CASE x
    WHEN 0 THEN
      DBMS_OUTPUT.PUT_LINE('x = 0');
    WHEN 1 THEN
      DBMS_OUTPUT.PUT_LINE('x = 1');
    ELSE
      DBMS_OUTPUT.PUT_LINE('x > 1');
  END CASE;
END;
/
```

Available In:

sonarlint | sonarcloud | sonarqube Developer Edition