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...

Labels should not be reused in inner scopes
Code Smell

"FUNCTIONS" should not have "OUT" parameters
Code Smell

Variables should be nullable
Code Smell

"VARCHAR2" should be used
Code Smell

Native SQL joins should be used
Code Smell

"FORALL" should be used
Code Smell

"FETCH ... BULK COLLECT INTO" should be used
Code Smell

Column aliases should be defined using "AS"
Code Smell

Procedures and functions should be encapsulated in packages
Code Smell

Procedures should have parameters
Code Smell

"EXECUTE IMMEDIATE" should be used instead of DBMS_SQL procedure calls
Code Smell

"NATURAL JOIN" queries should not be used
Code Smell

"END" statements of labeled loops should be labeled
Code Smell

## Labels should not be reused in inner scopes

Analyze your code

Code Smell Major pitfall

Using the same name for multiple purposes reduces the understandability of the code and might eventually lead to bugs.

This rule verifies that no label is reused in an inner scope.

### Noncompliant Code Example

```
<<foo>>
DECLARE
  a CONSTANT PLS_INTEGER := 0;
BEGIN
  <<foo>>                               -- Noncompliant
  DECLARE
    b CONSTANT PLS_INTEGER := 42;
  BEGIN
    DBMS_OUTPUT.PUT_LINE('x = ' || foo.b); -- Confusing
  END;
END;
/
```

### Compliant Solution

```
<<foo>>
DECLARE
  a CONSTANT PLS_INTEGER := 0;
BEGIN
  <<bar>>
  DECLARE
    b CONSTANT PLS_INTEGER := 42;
  BEGIN
    DBMS_OUTPUT.PUT_LINE('x = ' || bar.b);
  END;
END;
/
```

Available In:

sonarlint sonarcloud sonarqube Developer Edition