

































-  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... 

Dynamically executing code is security-sensitive
 Security Hotspot
SQL "JOIN" conditions should involve all joined tables
 Code Smell
"SELECT" statements used as argument of "EXISTS" statements should be selective
 Code Smell
"LIKE" clauses should not be used without wildcards
 Code Smell
Weak "REF CURSOR" types should not be used
 Code Smell
Whitespace and control characters in string literals should be explicit
 Code Smell
Blocks containing "EXECUTE IMMEDIATE" should trap all exceptions
 Code Smell
"EXCEPTION_INIT -20,NNN" calls should be centralized
 Code Smell
"CREATE OR REPLACE" should be used instead of "CREATE"
 Code Smell
Block labels should appear on the same lines as "END"
 Code Smell
"LOOP ... END LOOP;" constructs should be avoided
 Code Smell
"IF" statements should not be nested too deeply
 Code Smell
"CASE" expressions should end with "ELSE" clauses
 Code Smell
String literals should not be duplicated

Block start and end labels should match

Analyze your code

 Code Smell

 Blocker

  confusing

Labeled blocks are useful, especially when the code is badly indented, to match the begin and end of each block. This rule verifies that block start and end labels match, when both are specified.

Noncompliant Code Example

```
BEGIN
    NULL;
END; -- Compliant, no labels at all
/

<<myBlockLabel1>>
BEGIN
    NULL;
END; -- Compliant, only starting label
/

BEGIN
    NULL;
END myBlockLabel2; -- Compliant, only ending label
/

<<myBlockLabel3>>
BEGIN
    NULL;
END myBlockLabel4; -- Noncompliant, labels mismatch
/

<<myBlockLabel6>>
<<myBlockLabel6>>
BEGIN
    NULL;
END myBlockLabel6; -- Noncompliant, several starting labels
/
```

Compliant Solution

```
BEGIN
    NULL;
END;
/

<<myBlockLabel1>>
BEGIN
    NULL;
END;
/

BEGIN
    NULL;
END myBlockLabel2;
/

<<myBlockLabel3>>
BEGIN
    NULL;
END myBlockLabel3;
/

<<myBlockLabel6>>
BEGIN
    NULL;
END myBlockLabel6;
/
```

Available In:

   Developer Edition