





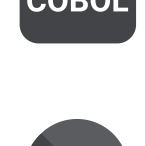



























-  Secrets
-  ABAP
-  Apex
-  C
-  C++
-  CloudFormation
-  COBOL
-  C#
-  CSS
-  Flex
-  Go
-  HTML
-  Java
-  JavaScript
-  Kotlin
-  Kubernetes
-  Objective C
-  PHP
-  PL/I
-  **PL/SQL**
-  Python
-  RPG
-  Ruby
-  Scala
-  Swift
-  Terraform
-  Text
-  TypeScript
-  T-SQL
-  VB.NET
-  VB6
-  XML

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... 

 Code Smell
"END" statements of labeled blocks should be labeled
 Code Smell
Two branches in a conditional structure should not have exactly the same implementation
 Code Smell
Unused assignments should be removed
 Code Smell
"LIKE" clauses should not start with wildcard characters
 Code Smell
Column names should be used in a SQL "ORDER BY" clause
 Code Smell
Procedures and functions should be documented
 Code Smell
SQL statements should not join too many tables
 Code Smell
Function and procedure names should comply with a naming convention
 Code Smell
Columns to be read with a "SELECT" statement should be clearly defined
 Code Smell
"CASE" structures should not have too many "WHEN" clauses
 Code Smell
Deprecated LONG and LONG RAW datatypes should no longer be used
 Code Smell

"END" statements of labeled blocks should be labeled

Analyze your code

 Code Smell

 Major 

 convention

Labeled blocks are useful, especially when the code is badly indented, to match the begin and end of each block. This check detects labeled blocks which are missing an ending label.

Noncompliant Code Example

```
<<myBlockLabel1>>
BEGIN
    NULL;
END; -- Noncompliant; this labeled loop has no ending label
/

BEGIN
    NULL; -- Compliant; not a labeled block
END;
/
```

Compliant Solution

```
<<myBlockLabel2>>
BEGIN
    NULL;
END myBlockLabel2;
/

BEGIN
    NULL;
END;
/
```

Available In:

sonarlint 

sonarcloud 

sonarqube  Developer Edition