
















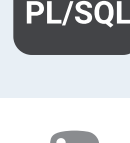
















-  Secrets
-  ABAP
-  Apex
-  C
-  C++
-  CloudFormation
-  COBOL
-  C#
-  CSS
-  Flex
-  Go
-  HTML
-  Java
-  JavaScript
-  Kotlin
-  Kubernetes
-  Objective C
-  PHP
-  PL/I
-  **PL/SQL**
-  Python
-  RPG
-  Ruby
-  Scala
-  Swift
-  Terraform
-  Text
-  TypeScript
-  T-SQL
-  VB.NET
-  VB6
-  XML

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
Nested loops should be labeled	
	Code Smell
Nested blocks should be labeled	
	Code Smell
"RESULT_CACHE" should not be used	
	Code Smell
Columns should be aliased	
	Code Smell
Track parsing failures	
	Code Smell
Files should not be too complex	
	Code Smell
Function and procedure parameters should comply with a naming convention	
	Code Smell
Magic literals should not be used	
	Code Smell
"UNION" should be used with caution	
	Code Smell
"GROUP BY" should not be used in SQL "SELECT" statements	
	Code Smell
Track breaches of an XPath rule	
	Code Smell
Track uses of "NOSONAR" comments	
	Code Smell

Nested loops should be labeled

Analyze your code

 Code Smell

 Major ?

 convention confusing

Labeled loops are useful, especially when the code is badly indented, to match the begin and end of each loop. When loops are nested, labeling them can improve the code's readability. This rule detects nested loops which do not have a start label.

Noncompliant Code Example

```
BEGIN
  LOOP
    LOOP -- Noncompliant, this nested loop is not labeled
      EXIT;
    END LOOP;

    EXIT;
  END LOOP;

  FOR i IN 1..10  LOOP
    WHILE true LOOP -- Noncompliant, this nested loop has no start label
      EXIT;
    END LOOP nestedLoopLabel1;

    EXIT;
  END LOOP;

  WHILE true LOOP
    <<nestedLoopLabel2>>
    LOOP -- Compliant, but better with an end label
      EXIT;
    END LOOP;

    EXIT;
  END LOOP;
END;
/
```

Compliant Solution

```
BEGIN
  LOOP
    <<nestedLoopLabel0>>
    LOOP
      EXIT;
    END LOOP nestedLoopLabel0;

    EXIT;
  END LOOP;

  FOR i IN 1..10  LOOP
    <<nestedLoopLabel1>>
    WHILE true LOOP
      EXIT;
    END LOOP nestedLoopLabel1;

    EXIT;
  END LOOP;

  WHILE true LOOP
    <<nestedLoopLabel2>>
    LOOP
      EXIT;
    END LOOP nestedLoopLabel2;

    EXIT;
  END LOOP;
END;
/
```

Available In:

sonarlint  | sonarcloud  | sonarqube  Developer Edition