

































-  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

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	 Code Smell
Constant names should comply with a naming convention	 Code Smell
Output parameters should be assigned	 Bug
"ROWNUM" should not be used at the same query level as "ORDER BY"	 Bug
All branches in a conditional structure should not have exactly the same implementation	 Bug
Strings should only be moved to variables or columns which are large enough to hold them	 Bug
"WHERE" clause conditions should not be contradictory	 Bug
Unary prefix operators should not be repeated	 Bug
DML events clauses should not include multiple "OF" clauses	

Block labels should appear on the same lines as "END"

Analyze your code

 Code Smell

 Critical

 convention

Labeled blocks are useful to help maintainers match-up the beginning and ending of each section of code, especially when that code is badly indented. However, if used, those labels must appear on the same line as the "END" keyword in order to avoid confusion. Otherwise, the label might be misread by maintainers as a procedure call.

Noncompliant Code Example

```
SET SERVEROUTPUT ON

DECLARE
  PROCEDURE foo AS
  BEGIN
    DBMS_OUTPUT.PUT_LINE('foo was called!');
  END;
BEGIN
  BEGIN
    NULL;
  END -- Semicolon was forgotten?

  foo; -- Noncompliant; looks like a procedure call, but is actually END block label

  <<myBlockLabel>>
  BEGIN
    NULL;
  END
  myBlockLabel; -- Noncompliant
END;
/
```

Compliant Solution

```
SET SERVEROUTPUT ON

DECLARE
  PROCEDURE foo AS
  BEGIN
    DBMS_OUTPUT.PUT_LINE('foo was called!');
  END;
BEGIN
  BEGIN
    NULL;
  END;

  foo; -- The method "foo" was actually meant to be called

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

Available In:

 |  |  Developer Edition