

































-  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 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

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

 Bug

"PACKAGE BODY" initialization sections should not contain "RETURN" statements

 Bug

"NULL" should not be compared directly

 Bug

"MLSLABEL" should not be used

 Bug

"CASE" expressions should end with "ELSE" clauses

Analyze your code

 Code Smell

 Critical



 cwe

The requirement for a final `ELSE` clause is defensive programming. The `CASE` expression should always provide a value.

Noncompliant Code Example

```
CASE grade -- Noncompliant, can raise a CASE_NOT_FOUND exception.
  WHEN 'A' THEN DBMS_OUTPUT.PUT_LINE('Excellent');
  WHEN 'B' THEN DBMS_OUTPUT.PUT_LINE('Very Good');
END CASE;
```

Compliant Solution

```
CASE grade
  WHEN 'A' THEN DBMS_OUTPUT.PUT_LINE('Excellent');
  WHEN 'B' THEN DBMS_OUTPUT.PUT_LINE('Very Good');
  ELSE DBMS_OUTPUT.PUT_LINE('No such grade');
END CASE;
```

See

- [MITRE, CWE-478](#) - Missing Default Case in Switch Statement
- [CERT, MSC01-C](#) - Strive for logical completeness

Available In:

 |  |  Developer Edition