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...

"EXIT WHEN" should be used rather than "IF ... THEN EXIT; END IF;"
Code Smell

"FOR" loop end conditions should not be hard-coded
Code Smell

Large item lists should not be used with "IN" clauses
Code Smell

"GOTO" should not be used within loops
Code Smell

"FULL OUTER JOINS" should be used with caution
Code Smell

"CASE" should be used rather than "DECODE"
Code Smell

"CROSS JOIN" queries should not be used
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

"FOR" loop end conditions should not be hard-coded

Analyze your code

Code Smell | Major | brain-overload

Hard-coding bounds in FOR loops is a bad practice, just as magic numbers in general are. Often, those magic bounds can be replaced by dynamic values. If that is not possible, replacing the literal number with a constant is still better.

Noncompliant Code Example

```
SET SERVEROUTPUT ON

DECLARE
  TYPE myCollectionType IS VARRAY(3) OF VARCHAR2(42);
  myCollection myCollectionType := myCollectionType('David', 'John', 'Richard');

BEGIN

  FOR i IN 2 .. 3 -- Noncompliant; magic numbers used for the loop bounds
  LOOP
    DBMS_OUTPUT.PUT_LINE('name = ' || myCollection(i));
  END LOOP;

  FOR i IN 2 .. myCollection.LAST -- Noncompliant, better but still magic
  LOOP
    DBMS_OUTPUT.PUT_LINE('name = ' || myCollection(i));
  END LOOP;

END;
/
```

Compliant Solution

```
SET SERVEROUTPUT ON

DECLARE
  TYPE myCollectionType IS VARRAY(3) OF VARCHAR2(42);
  myCollection myCollectionType := myCollectionType('David', 'John', 'Richard');

BEGIN
  FOR i IN myCollection.FIRST .. myCollection.LAST
  LOOP
    DBMS_OUTPUT.PUT_LINE('name = ' || myCollection(i));
  END LOOP;
END;
/
```

Available In:

sonarlint | sonarcloud | sonarqube Developer Edition