





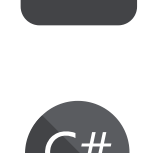



























-  Secrets
-  ABAP
-  Apex
-  C
-  C++
-  CloudFormation
-  COBOL
-  C#
-  CSS
-  Flex
-  Go
-  HTML
-  Java
-  JavaScript
-  Kotlin
-  Kubernetes
-  Objective C
-  PHP
-  PL/I
-  **PL/SQL**
-  Python
-  RPG
-  Ruby
-  Scala
-  Swift
-  Terraform
-  Text
-  TypeScript
-  T-SQL
-  VB.NET
-  VB6
-  XML

PL/SQL

PL/SQL static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your PL/SQL code













All rules188

 Vulnerability4

 Bug45

 Security Hotspot2

 Code Smell137

 Code Smell
Variables should not be initialized with "NULL"
 Code Smell
Boolean checks should not be inverted
 Code Smell
Unused local variables should be removed
 Code Smell
Package names should comply with a naming convention
 Code Smell
Variables should comply with a naming convention
 Code Smell
Return of boolean expressions should not be wrapped into an "if-then-else" statement
 Code Smell
Boolean literals should not be redundant
 Code Smell
Comments should not be nested
 Code Smell
"DBMS_UTILITY.FORMAT_ERROR_STACK" and "FORMAT_ERROR_BACKTRACE" should be used together
 Code Smell
Procedures should not contain "RETURN" statements
 Code Smell
Track uses of "TODO" tags
 Code Smell

Variables should not be initialized with "NULL"

Analyze your code

 Code Smell

 Minor?

 clumsy

Explicit variable initializations with null values are superfluous, since unassigned variables are implicitly initialized to null.

Noncompliant Code Example

```
SET SERVEROUTPUT ON

DECLARE
    foo PLS_INTEGER := NULL; -- Noncompliant, the null assignation is superfluous
    bar VARCHAR2(100) := ''; -- Noncompliant, the null assignation is superfluous
    correctInitializedString VARCHAR2(100) := 'Hello world!';

BEGIN
    IF foo IS NULL THEN
        DBMS_OUTPUT.PUT_LINE('foo is NULL');
    ELSE
        DBMS_OUTPUT.PUT_LINE('foo is NOT NULL');
    END IF;
END;
/
```

Compliant Solution

```
SET SERVEROUTPUT ON

DECLARE
    foo PLS_INTEGER;
    bar VARCHAR2(100);
    correctInitializedString VARCHAR2(100) := 'Hello world!';

BEGIN
    IF foo IS NULL THEN
        DBMS_OUTPUT.PUT_LINE('foo is NULL');
    ELSE
        DBMS_OUTPUT.PUT_LINE('foo is NOT NULL');
    END IF;
END;
/
```

Available In:

 |  |  Developer Edition