

































-  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... 

"DELETE" and "UPDATE" statements should contain "WHERE" clauses	 Bug
"FORALL" statements should use the "SAVE EXCEPTIONS" clause	 Bug
Pipelined functions should have at least one "PIPE ROW" statement and not return an expression (PLS-00633)	 Bug
GOTO should not be used to jump backwards	 Code Smell
Quoted identifiers should not be used	 Code Smell
Loop start and end labels should match	 Code Smell
Block start and end labels should match	 Code Smell
Cipher algorithms should be robust	 Vulnerability
"COMMIT" should not be used inside a loop	 Bug
Individual "WHERE" clause conditions should not be unconditionally true or false	 Bug
Nullable subqueries should not be used in "NOT IN" conditions	 Bug
"COMMIT" and "ROLLBACK" should not be called from non-autonomous transaction triggers	 Bug
Positional and named arguments should not be mixed in invocations	 Bug

"FETCH ... BULK COLLECT INTO" should not be used without a "LIMIT" clause

Analyze your code

 Bug  Blocker 

A `FETCH ... BULK COLLECT INTO` without a `LIMIT` clause will load all the records returned by the cursor at once. This may lead to memory exhaustion. Instead, it is better to process the records in chunks using the `LIMIT` clause.

Noncompliant Code Example

```
SET SERVEROUTPUT ON

-- Fetches all records at once, requiring lots of memory
DECLARE
    TYPE largeTableRowArrayType IS TABLE OF largeTable%ROWTYPE;
    largeTableRowArray largeTableRowArrayType;
    CURSOR myCursor IS SELECT * FROM largeTable;
BEGIN
    OPEN myCursor;

    FETCH myCursor BULK COLLECT INTO largeTableRowArray; -- Non-compliant

    DBMS_OUTPUT.PUT_LINE('Alternative 1: ' || largeTableRowArray.COUNT || ' records');

    CLOSE myCursor;
END;
/
```

Compliant Solution

```
SET SERVEROUTPUT ON

-- fetches one chunk at a time, requiring constant memory
DECLARE
    TYPE largeTableRowArrayType IS TABLE OF largeTable%ROWTYPE;
    largeTableRowArray largeTableRowArrayType;
    CURSOR myCursor IS SELECT * FROM largeTable;
    counter PLS_INTEGER := 0;
BEGIN
    OPEN myCursor;

    LOOP
        FETCH myCursor BULK COLLECT INTO largeTableRowArray LIMIT 1000; -- Compliant

        counter := counter + largeTableRowArray.COUNT;

        EXIT WHEN myCursor%NOTFOUND;
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('Alternative 1: ' || counter || ' records');

    CLOSE myCursor;
END;
/

DROP TABLE largeTable;
```

Available In:

 |  |  