PL/SQL static code analysis: Variables should not be shadowed

1-2 minutes

Overriding or shadowing a variable declared in an outer scope can strongly impact the readability, and therefore the maintainability, of a piece of code. Further, it could lead maintainers to introduce bugs because they think they're using one variable but are really using another.

Noncompliant Code Example

```
SET SERVEROUTPUT ON

DECLARE
  foo VARCHAR2(42) := 'foo';
BEGIN
  DECLARE
    foo VARCHAR2(42) := 'bar'; -- Noncompliant - this variable hides the one above and should be renamed
  BEGIN
    DBMS_OUTPUT.PUT_LINE(foo); -- Displays "bar", which is confusing
  END;

  DBMS_OUTPUT.PUT_LINE(foo); -- Displays "foo"
END;
```

```
/
```

Compliant Solution

```
SET SERVEROUTPUT ON

DECLARE
  foo VARCHAR2(42) := 'foo';
BEGIN
  DECLARE
    bar VARCHAR2(42) := 'bar'; -- Compliant
  BEGIN
    DBMS_OUTPUT.PUT_LINE(bar); -- Displays "bar"
  END;

  DBMS_OUTPUT.PUT_LINE(foo); -- Displays "foo"
END;
/
```

See

- CERT, DCL01-C. - Do not reuse variable names in subscopes
- CERT, DCL51-J. - Do not shadow or obscure identifiers in subscopes