PL/SQL static code analysis: Collections should not be iterated in "FOR" loops

2 minutes

The `FOR` loop at first seems like a convenient way of iterating over the elements of a collection, but doing so will raise a `VALUE_ERROR` exception if the collection is empty. Looping instead from 1 to `COUNT` doesn't work either if the collection is sparse; that leads to a `ORA-01403: no data found` error.

Instead, a `WHILE` loop should be used.

Noncompliant Code Example

```
DECLARE
  TYPE fooType IS TABLE OF VARCHAR2(42);
  foo fooType := new fooType('Strawberry', 'Apple', 'Banana');
BEGIN
  foo.DELETE(2);                          -- The collection is now
sparse

  FOR i IN 1 .. foo.COUNT                 -- Noncompliant - leads
to ORA-01403: no data found
  LOOP
    DBMS_OUTPUT.PUT_LINE(i || ' = ' || foo(i));
  END LOOP;
```

```
END;
/
```

Compliant Solution

```
DECLARE
  TYPE fooType IS TABLE OF VARCHAR2(42);
  foo fooType := new fooType('Strawberry', 'Apple', 'Banana');
  i PLS_INTEGER;
BEGIN
  foo.DELETE(2);                          -- The collection is now
sparse

  i := foo.FIRST;

  WHILE (i IS NOT NULL)                   -- Compliant - works
as expected
  LOOP
    DBMS_OUTPUT.PUT_LINE(i || ' = ' || foo(i));
    i := foo.NEXT(i);
  END LOOP;
END;
/
```