Stored Functions

- Jayendra Khatod

Objectives

- Describe the uses of functions
- Create stored functions
- Invoke a function
- Remove a function
- Differentiate between a procedure and a function

Overview of Stored Functions

- A function is a named PL/SQL block that returns a value.
- A function can be stored in the database as a schema object for repeated execution.
- A function is called as part of an expression.

Syntax for Creating Functions

```
CREATE [OR REPLACE] FUNCTION function_name
 (parameter1 [mode1] datatype1,
  parameter2 [mode2] datatype2,
  . . .)
RETURN datatype
IS|AS
PL/SQL Block;
```

The PL/SQL block must have at least one RETURN statement.

Example

```
CREATE OR REPLACE FUNCTION get_sal
    (v_id  IN employees.employee_id%TYPE)
   RETURN NUMBER
 IS
    v_salary  employees.salary%TYPE :=0;
  BEGIN
    SELECT salary
    INTO       v_salary
    FROM       employees
    WHERE    employee_id = v_id;
    RETURN (v_salary);
  END get_sal;
 /
```

Executing Functions

- Invoke a function as part of a PL/SQL expression.
- Create a variable to hold the returned value.
- Execute the function. The variable will be populated by the value returned through a RETURN statement.

Executing Functions: Example

Create the function GET_SAL function

```
DECLARE
v_salary number;

BEGIN
v_salary := get_sal(100);
 dbms_output.put_line( v_salary);
END ;
```

Advantages of User-Defined Functions

- **Extend SQL where activities are too complex, too awkward, or unavailable with SQL**
- **Can increase efficiency when used in the WHERE clause to filter data, as opposed to filtering the data in the application**
- **Can manipulate character strings**

Invoking Functions in SQL Expressions

```
CREATE OR REPLACE FUNCTION tax(p_value IN
NUMBER)
   RETURN NUMBER IS
BEGIN
   RETURN (p_value * 0.08);
END tax;
/
```

```
SELECT empno, ename, sal, tax(sal)
FROM emp
WHERE deptno = 10;
```

Restrictions on Calling Functions from SQL

To be callable from SQL expressions, a user-defined function must:

- **Be a stored function**
- **Accept only IN parameters**
- **Accept only valid SQL data types, not PL/SQL specific types, as parameters**
- **Return data types that are valid SQL data types, not PL/SQL specific types**

Restrictions on Calling Functions from SQL

- **Functions called from SQL expressions cannot contain DML statements.**
- **Functions called from UPDATE/DELETE statements on a table T cannot contain DML on the same table T.**
- **Functions called from an UPDATE or a DELETE statement on a table T cannot query the same table.**
- **Functions called from SQL statements cannot contain statements that end the transactions.**
- **Calls to subprograms that break the previous restriction are not allowed in the function.**

Restrictions on Calling Functions from SQL

```
CREATE OR REPLACE FUNCTION dml_call_sql (p_sal NUMBER)
    RETURN NUMBER IS
BEGIN
    INSERT INTO emp(empno, ename, hiredate, job, sal)
    VALUES(1, 'employee 1', SYSDATE, 'SA_MAN', 1000);
    RETURN (p_sal + 100);
END;
/
```

```
ERROR at line 1:
ORA-04091: table scott.EMP is mutating, trigger/function may not see it
ORA-06512: at “scott.DML_CALL_SQL", line 4
```

```
UPDATE emp SET sal = dml_call_sql(2000)
WHERE empno = 7839;
```

Mutating Table

Removing Functions

```
DROP FUNCTION function_name
```

```
DROP FUNCTION get_sal;
```

- All the privileges granted on a function are revoked when the function is dropped.
- The CREATE OR REPLACE syntax is equivalent to dropping a function and recreating it. Privileges granted on the function remain the same when this syntax is used.

- All the privileges granted on a function are revoked when the function is dropped.

Procedure or Function?

Examples:
What would be the logical choice for 1 and 2?

1. ***A subprogram that accepts one value and outputs three values***

2. ***A subprogram that accepts one value and outputs one value***

Comparing Procedures and Functions

Procedures	Functions
Execute as a PL/SQL statement	Invoke as part of an expression
Do not contain RETURN clause in the header	Must contain a RETURN clause in the header
Can return none, one, or many values	Must return a single value
Can contain a RETURN Statement	Must contain at least one RETURN statement

Benefits of Stored Procedures and Functions

- **Improved performance**
- **Easy maintenance**
- **Improved data security and integrity**
- **Improved code clarity**

Summary

- **A function is a named PL/SQL block that must return a value.**
- **A function is created by using the CREATE FUNCTION syntax.**
- **A function is invoked as part of an expression.**
- **A function stored in the database can be called in SQL statements.**
- **A function can be removed from the database by using the DROP FUNCTION syntax.**
- **Generally, you use a procedure to perform an action and a function to compute a value.**

Thank You !