| GUÍA 2.1.5:
Construcción de funciones almacenadas para procesar información

Sigla	Asignatura	Experiencia de Aprendizaje
BDY1103	Taller de Base de Datos	EA2: Implementa programas PL/SQL en la base de datos para construir una solución integral de procesamiento y generación de información o ser usados en otros procesos y/o aplicaciones.
Tiempo	**Modalidad de Trabajo**	**Indicadores de logro**
4h	Individual	IL2.1

Antecedentes generales

En esta guía encontrarás los contenidos asociados a la construcción de funciones almacenadas para procesar información, junto con ejemplos y actividades prácticas a desarrollar.

Requerimientos para esta actividad

En esta actividad, los y las estudiantes deberán utilizar SQL Developer y seguir las instrucciones indicadas por el/la docente.

Sesión 2: Construcción de funciones almacenadas para procesar Información

Objetivos de Aprendizaje

1. Comprender cómo construir funciones almacenadas con y sin parámetros.
2. Aprender a procesar información masiva utilizando funciones.
3. Integrar procedimientos y funciones en otros programas PL/SQL y sentencias SQL.

1. Introducción a las Funciones Almacenadas

1.1 ¿Qué son?

- Funciones Almacenadas: Subprogramas que realizan cálculos y devuelven un valor. Pueden ser usadas en consultas SQL, dentro de PL/SQL, o en expresiones.

1.2 Ventajas

- Reutilización de Código: Facilita el uso repetitivo de cálculos y transformaciones.
- Versatilidad: Se pueden usar en SQL, PL/SQL, y expresiones.
- Modularidad: Permiten organizar cálculos complejos en componentes reutilizables.

1.3 Estructura Básica

```
CREATE OR REPLACE FUNCTION function_name RETURN return_type IS
BEGIN
   -- Código de la función
END;
/
```

2. Funciones sin Parámetros

2.1 Concepto

- Funciones sin Parámetros: Realizan cálculos o transformaciones estándar sin requerir datos externos que sean pasados como parámetros.

2.2 Ejemplo Práctico: Calcular Total de Empleados

Problema: Contar el total de empleados en la tabla emp.

```
CREATE OR REPLACE FUNCTION count_total_employees RETURN NUMBER IS
   v_total_employees NUMBER;
BEGIN
   SELECT COUNT(*) INTO v_total_employees FROM emp;
   RETURN v_total_employees;
END;
/
```

2.3 Uso de la Función en PL/SQL y SQL

En PL/SQL:

```
DECLARE
   v_total NUMBER;
BEGIN
   v_total := count_total_employees;
   DBMS_OUTPUT.PUT_LINE('Total Employees: ' || v_total);
END;
/
```

En SQL:

```
SELECT            count_total_employees            FROM            dual;
```

3. Funciones con Parámetros

3.1 Concepto

- Funciones con Parámetros: Aceptan datos de entrada para realizar cálculos específicos y devolver resultados.

3.2 Ejemplo Práctico: Calcular Total de Ventas por Producto

Problema: Calcular el total de ventas de un producto dado su ID.

```
CREATE       OR       REPLACE       FUNCTION       calculate_total_sales(
   p_product_id                          IN                          NUMBER
)                    RETURN                    NUMBER                    IS
   v_total_sales                                                    NUMBER;
BEGIN
   SELECT             SUM(unit_price             *             quantity)
   INTO                                                   v_total_sales
   FROM                                                     order_items
   WHERE             product_id             =             p_product_id;

   RETURN                                                 v_total_sales;
END;
/
```

3.3 Uso de la Función en PL/SQL y SQL

En PL/SQL:

```
DECLARE
   v_sales NUMBER;
BEGIN
   v_sales := calculate_total_sales(46);
   DBMS_OUTPUT.PUT_LINE('Total Sales: ' || v_sales);
END;
```

```
/
```

En SQL:

```
SELECT calculate_total_sales(46) FROM dual;
```

4. Ejemplos Prácticos

Ejemplo 1: Función sin Parámetros

Objetivo: Calcular el total de productos.

```
CREATE OR REPLACE FUNCTION count_total_products RETURN NUMBER IS
   v_total_products NUMBER;
BEGIN
   SELECT COUNT(*) INTO v_total_products FROM products;
   RETURN v_total_products;
END;
/

select count_total_products()
from dual;
/
```

Ejemplo 2: Función con Parámetros

Objetivo: Calcular el promedio de salarios por departamento.

```
CREATE OR REPLACE FUNCTION calculate_avg_salary_by_dept(
   p_deptno IN NUMBER
) RETURN NUMBER IS
   v_avg_salary NUMBER;
BEGIN
   SELECT AVG(sal)
   INTO v_avg_salary
   FROM emp
   WHERE deptno = p_deptno;

   RETURN v_avg_salary;
END;
/

select calculate_avg_salary_by_dept(30)
from dual;
/
```

5. Actividades Prácticas

Actividad 1: Función sin Parámetros

1. Objetivo: Crear una función que devuelva el total de órdenes.

2. Tabla: orders
3. Pasos:
    - Crear la función count_total_orders.
    - Usar COUNT(*) para calcular el total.

Actividad 2: Función con Parámetros

1. Objetivo: Crear una función que devuelva el total de pedidos de un cliente dado su ID.
2. Tabla: orders
3. Pasos:
    - Crear la función calculate_total_orders_by_customer.
    - Aceptar customer_id como parámetro.
    - Usar las tablas ORDERS y ORDER_ITEMS para calcular el total de pedidos.

Conclusión de la Sesión 2

- Resumen: Las funciones almacenadas son cruciales para realizar cálculos y transformaciones reutilizables.

Evaluación y Mejores Prácticas

1. Validación de Parámetros: Siempre valida los parámetros para asegurar datos correctos y prevenir errores.
2. Documentación: Documenta claramente el propósito y uso de cada procedimiento y función.
3. Pruebas: Realiza pruebas exhaustivas para asegurar que los subprogramas funcionan correctamente en diferentes escenarios.

Recursos Adicionales

PL/SQL. Procedimientos y Funciones. (2016, enero 1). PL/SQL. Procedimientos y Funciones. https://elbauldelprogramador.com/plsql-procedimientos-y-funciones/

DiscoDurodeRoer [@DiscoDurodeRoer]. (2018, agosto 23). *Funciones y procedimientos | Ejercicios PL/SQL #4*. Youtube. https://www.youtube.com/watch?v=lRp2ReMONx0