Inserción de Datos

En nuestro sistema, la inserción de datos se realiza principalmente al registrar nuevos inmuebles. Veamos un ejemplo detallado:

INSERT INTO Inmuebles (Tipo, Superficie, Direccion, Zona, PrecioVenta, PrecioAlquiler,

Propietario, TelefonoPropietario, OficinaID, TieneLlaves)

VALUES ($1, $2, $3, $4, $5, $6, $7, $8, $9, $10)

RETURNING Ref;

Esta consulta utiliza parámetros ($1, $2, etc.) para prevenir inyección SQL. El `RETURNING Ref` nos devuelve el identificador del inmueble recién insertado, que necesitaremos para registrar sus características específicas.

Cuando insertamos las características, la consulta varía según el tipo de inmueble. Para una vivienda:

INSERT INTO CaracteristicasVivienda (Ref, NumHabitaciones, NumBanos, NumAseos, NumCocinas,

TieneGasCiudad, TienePuertaBlindada, TieneParquet)

VALUES ($1, $2, $3, $4, $5, $6, $7, $8);

Modificación de Datos

En nuestro sistema, las modificaciones son especialmente importantes cuando actualizamos características de inmuebles. Por ejemplo, si necesitáramos actualizar el precio de un inmueble:

UPDATE Inmuebles

SET PrecioVenta = $1, PrecioAlquiler = $2

WHERE Ref = $3;

Este tipo de consulta utiliza la cláusula `WHERE` para asegurarse de que solo se modifica el inmueble específico. El uso de parámetros ($1, $2, $3) nuevamente protege contra la inyección SQL y mejora la seguridad.

Eliminación de Datos

La eliminación en nuestro sistema es particularmente interesante porque involucra múltiples tablas relacionadas. Cuando eliminamos un inmueble, necesitamos considerar todas sus dependencias:

-- Primero eliminamos las visitas asociadas

DELETE FROM Visitas

WHERE RefInmueble = $1;

-- Luego las características específicas

DELETE FROM CaracteristicasVivienda

WHERE Ref = $1;

-- o

DELETE FROM CaracteristicasLocal

WHERE Ref = $1;

-- Finalmente el inmueble

DELETE FROM Inmuebles

WHERE Ref = $1;

Gracias a las restricciones `ON DELETE CASCADE` en nuestras tablas, podríamos simplificar esto a una sola consulta:

DELETE FROM Inmuebles WHERE Ref = $1;

Y PostgreSQL se encargará automáticamente de eliminar todos los registros relacionados.

Consideraciones Importantes

En nuestro sistema, todas estas operaciones están encapsuladas en transacciones para mantener la integridad de los datos. Por ejemplo:

work W(c);

try {

// Realizar operaciones SQL

W.commit();

} catch (const exception &e) {

// Manejar el error

// La transacción se revierte automáticamente

}

Esta estructura asegura que todas las operaciones se completen exitosamente o ninguna se aplique, manteniendo la consistencia de la base de datos.

Además, nuestro sistema implementa validaciones antes de ejecutar cualquier consulta. Por ejemplo, antes de eliminar un inmueble, verificamos su existencia:

SELECT COUNT(\*) FROM Inmuebles WHERE Ref = $1

Esta práctica de verificación previa, combinada con el uso de transacciones y parámetros preparados, hace que nuestro sistema sea robusto y seguro en el manejo de datos.