**TRANSACCIONES**

1. **Transacciones**
   1. Al ser Oracle un sistema de base de datos puramente transaccional no existe la instrucción BEGIN TRANSACTION, las transacciones se pueden componer de puntos de controles SAVEPOINT, además de sus sentencias de finalización como lo son COMMIT el cual se usa cuando la transacción termina correctamente, ROLLBACK se rechaza la transacción y cualquier posible cambio se deshace, estas sentencias para su correcto funcionamiento se puede declarar su funcionamiento hasta tal punto usando COMMIT TO \_\_\_\_; O ROLLBACK TO \_\_\_\_;

Por lo que su estructura es:

DECLARE

BEGIN

COMMIT;

ROLLBACK;

END;

* 1. En Oracle se manejan 3 niveles de aislamiento.
     1. Nivel de aislamiento READ COMMITTED:
        1. En este nivel solo se usan los diferentes datos que fueron confirmados (Commited Data) antes del comienzo de la ejecución, en este nivel se logra evitar una lectura sucia, ya que los cambios que no han sido confirmados no son visibles para otra transacción, hasta que se confirme el cambio.
     2. Nivel de aislamiento SERIALIZABLE:
        1. En este nivel cuando se ejecuta se realiza un bloqueo de rango en el conjunto de datos haciendo que cuando se ejecuta una transacción se logre bloquear todos los registros y recursos a los que se tiene acceso impidiendo cualquier cambio que haga otro usuario como insertar o actualizar hasta que la transacción se complete, es el aislamiento más fuerte posible, en este nivel se logra evitar el fenómeno de fantasma.
     3. Nivel de aislamiento READ ONLY:
        1. En este nivel las transacciones no realizan modificaciones en la base de datos, por lo que puede ser útil para consultas de solo lectura.
  2. El aislamiento por defecto de Oracle es READ-COMMITTED

1. **Vistas**
   1. Para crear una vista en Oracle se sigue la siguiente estructura;

CREATE VIEW nombre\_vista AS (subconsulta);

En cambio, para eliminarla se sigue la siguiente

DROP VIEW nombre\_vista;

Cuando se elimina una tabla a la que hace referencia una vista, esta no se elimina, toca eliminarla explícitamente

* 1. Restricción de consulta, en esta restricción solo se podrá hacer lectura sobre ella por lo que evita que la vista se use para agregar, cambiar o suprimir datos de la tabla subyacente, en opción de comprobación prohíbe cualquier cambio en la tabla subyacente que produzca filas que no estén incluidas en esta vista.

1. **Modularidad Paquetes**
   1. Los paquetes en Oracle ayudan para agrupar y organizar funcionalidades en una base de datos, ya que agrupa objetos PL/SQL (funciones, procedimientos, tipos y demás) que permite tener programas estructurados agripados por funcionalidades.
   2. Para crear un paquete, lo primero que se hace es crear la cabecera donde se definen procedimientos, funciones, variables y demás, en esta parte solo se declar, para luego crear el cuerpo del paquete, en donde se define el bloque de código para las funciones y procedimientos declarados en la cabecera.
   3. Ejemplo:

CREATE PACKAGE nombre\_paquete AS

PROCEDURE procedimiento1;

FUNCTION funcion1 RETURN NUMBER;

END nombre\_paquete;

CREATE PACKAGE BODY nombre\_paquete AS

PROCEDURE procedimiento1 IS

BEGIN

-- lógica del procedimiento1

END procedimiento1;

FUNCTION funcion1 RETURN NUMBER IS

BEGIN

-- lógica de la función1

END funcion1;

END nombre\_paquete;

**Para poder invocar un procedimiento o función de un paquete se debe usar**  **la siguiente sintaxis:**

nombre\_paquete.procedimiento1;

variable := nombre\_paquete.funcion1;

**Para modificar el encabezado se realiza la siguiente instrucción:**

ALTER PACKAGE nombre\_paquete ADD (

PROCEDURE nuevo\_procedimiento;

);

**Para el cuerpo se sigue la siguiente sintaxis**

ALTER PACKAGE BODY nombre\_paquete ADD (

PROCEDURE nuevo\_procedimiento IS

BEGIN

-- lógica del nuevo\_procedimiento

END nuevo\_procedimiento;

);

**Para la eliminación del paquete:**

DROP PACKAGE nombre\_paquete;

1. **SYS\_REFCURSOR**
   1. Es un tipo de dato especial que se usa para representar un cursor o conjunto de resultados. Es útil cuando se necesita devolver resultados de un procedimiento almacenado o una función.
   2. Se define de la siguiente manera, pondremos un ejemplo de una declaración de una variable de cursor de un paquete:

CREATE OR REPLACE PACKAGE my\_pkg

AS

TYPE emp\_cur\_type IS REF CURSOR RETURN emp%ROWTYPE;

my\_rec emp\_cur\_type;

END my\_pkg

Para retornar usamos este ejemplo donde la función abre un cursor y lo devuelve como resultado

CREATE OR REPLACE FUNCTION obtener\_datos() RETURN SYS\_REFCURSOR AS

mi\_cursor SYS\_REFCURSOR;

BEGIN

OPEN mi\_cursor FOR

SELECT columna1, columna2

FROM mi\_tabla;

RETURN mi\_cursor;

END obtener\_datos;

PRACTICANDO

En auto5.sql