Tarea de Transacciones en Modelo Relacional
Ejercicios y Preguntas

Estudiante: Castro Contreras Luis Alejandro
Código: 23200247

1. Ejercicio 1 – Control básico de transacciones

Cree un bloque anónimo PL/SQL que:

- Aumente en un 10% el salario de los empleados del departamento 90.
- Guarde un SAVEPOINT llamado punto1.
- Aumente en un 5% el salario de los empleados del departamento 60.
- Realice un ROLLBACK al SAVEPOINT punto1.
- Ejecute finalmente un COMMIT.

```
BEGIN
 -- 1. Aumentar en un 10% el salario de los empleados del depto 90
 UPDATE Employees
   SET salary = salary * 1.10
  WHERE department_id = 90;

 -- 2. Guardar SAVEPOINT
 SAVEPOINT punto1;

 -- 3. Aumentar en un 5% el salario de los empleados del depto 60
 UPDATE Employees
   SET salary = salary * 1.05
  WHERE department_id = 60;

 -- 4. Rollback parcial al SAVEPOINT
 ROLLBACK TO punto1;

 -- 5. Confirmar cambios
 COMMIT;
END;
/
```

Preguntas:

a. ¿Qué departamento mantuvo los cambios?

   El departamento 90 mantuvo el aumento del 10%, porque ese cambio ocurrió antes del SAVEPOINT y no fue revertido.

b. ¿Qué efecto tuvo el ROLLBACK parcial?

   Deshizo únicamente el aumento del 5% aplicado al departamento 60, ya que ese cambio ocurrió después del SAVEPOINT.

c. ¿Qué ocurriría si se ejecutara ROLLBACK sin especificar SAVEPOINT?

   Se desharían todos los cambios de la transacción, incluyendo el aumento del 10% en el departamento 90. Es decir, la base de datos volvería al estado inicial antes de ejecutar el bloque.

2. Ejercicio 2 – Bloqueos entre sesiones

En dos sesiones diferentes de Oracle:

- En la primera sesión, ejecute:

```
UPDATE employees
SET salary = salary + 500
WHERE employee_id = 103;
```

- Sin ejecutar COMMIT, en la segunda sesión, intente modificar el mismo registro.
- Observe el bloqueo y, desde la primera sesión, ejecute:

```
ROLLBACK;
```

- Analice el efecto sobre la segunda sesión.

```
-- Sesión 1

UPDATE employees
SET salary = salary + 500
WHERE employee_id = 103;

ROLLBACK;

-- Sesión 2

UPDATE employees
SET salary = salary + 500
WHERE employee_id = 103;
```

Preguntas:

a. ¿Por qué la segunda sesión quedó bloqueada?

   Porque Oracle aplica un bloqueo exclusivo de fila cuando una transacción modifica un registro. Hasta que la primera sesión no confirme (COMMIT) o deshaga (ROLLBACK), ninguna otra sesión puede modificar esa misma fila.

b. ¿Qué comando libera los bloqueos?

   Los bloqueos se liberan automáticamente al ejecutar COMMIT o ROLLBACK en la sesión que los generó.

c. ¿Qué vistas del diccionario permiten verificar sesiones bloqueadas?

   Las más usadas son:

   - V$LOCK → Muestra los bloqueos actuales.
   - V$SESSION → Muestra las sesiones activas y su estado.
   - DBA_BLOCKERS → Identifica sesiones que están bloqueando.
   - DBA_WAITERS → Identifica sesiones que están esperando por un bloqueo.

3. Ejercicio 3 – Transacción controlada con bloque PL/SQL

Cree un bloque anónimo PL/SQL que realice una transferencia de empleado de un departamento a otro, registrando la transacción en JOB_HISTORY.
Pasos:

- Actualice el department_id del empleado 104 al departamento 110.
- Inserte simultáneamente el registro correspondiente en JOB_HISTORY.
- Si ocurre un error (por ejemplo, departamento inexistente), haga un ROLLBACK y muestre un mensaje con DBMS_OUTPUT.

```
SET SERVEROUTPUT ON;

DECLARE
  v_start_date DATE;
  v_job_id    employees.job_id%TYPE;
  v_dept_id   employees.department_id%TYPE;
BEGIN
  -- 1. Obtener datos actuales del empleado 104
  SELECT hire_date, job_id, department_id
    INTO v_start_date, v_job_id, v_dept_id
    FROM employees
   WHERE employee_id = 104;
  -- 2. Actualizar el departamento del empleado 104
  UPDATE employees
     SET department_id = 110
   WHERE employee_id = 104
  -- 3. Insertar registro en JOB_HISTORY
  INSERT INTO job_history (employee_id, start_date, end_date, job_id, department_id)
  VALUES (104, v_start_date, SYSDATE, v_job_id, v_dept_id);
  -- 4. Confirmar cambios
  COMMIT;
  DBMS_OUTPUT.PUT_LINE('Transferencia realizada con éxito.');
EXCEPTION
  WHEN OTHERS THEN
    ROLLBACK;
    DBMS_OUTPUT.PUT_LINE('Error en la transferencia: ' || SQLERRM);
END;
/
```

Preguntas:

a. ¿Por qué se debe garantizar la atomicidad entre las dos operaciones?

   Porque si se actualiza el empleado pero no se registra en JOB_HISTORY (o viceversa), los datos quedarían inconsistentes. La atomicidad asegura que ambas operaciones ocurran juntas o ninguna.

b. ¿Qué pasaría si se produce un error antes del COMMIT?

   El ROLLBACK desharía todos los cambios de la transacción, dejando la base de datos en el estado anterior. Así se evita que quede un cambio “a medias”.

c. ¿Cómo se asegura la integridad entre EMPLOYEES y JOB_HISTORY?

   Mediante las claves foráneas: JOB_HISTORY.employee_id referencia a EMPLOYEES.employee_id, y department_id y job_id deben existir en sus tablas respectivas. Esto garantiza que solo se registren transferencias válidas.

4. Ejercicio 4 – SAVEPOINT y reversión parcial

Diseñe un bloque anónimo PL/SQL que ejecute las siguientes operaciones en una sola transacción:

- Aumentar el salario en 8% para empleados del departamento 100 → SAVEPOINT A.
- Aumentar el salario en 5% para empleados del departamento 80 → SAVEPOINT B.
- Eliminar los empleados del departamento 50.
- Revierte los cambios hasta el SAVEPOINT B.
- Finalmente, confirma la transacción con COMMIT.

```
SET SERVEROUTPUT ON;

BEGIN
  -- 1. Aumentar salario en 8% para depto 100
  UPDATE employees
    SET salary = salary * 1.08
   WHERE department_id = 100;
  SAVEPOINT A;

  -- 2. Aumentar salario en 5% para depto 80
  UPDATE employees
    SET salary = salary * 1.05
   WHERE department_id = 80;
  SAVEPOINT B;

  -- 3. Eliminar empleados del depto 50
  DELETE FROM employees
   WHERE department_id = 50;

  -- 4. Revertir hasta SAVEPOINT B
  ROLLBACK TO B;

  -- 5. Confirmar transacción
  COMMIT;

  DBMS_OUTPUT.PUT_LINE('Transacción completada con SAVEPOINT y ROLLBACK parcial.');
END;
/
```

Preguntas:

a. ¿Qué cambios quedan persistentes?

   Los aumentos de salario en departamento 100 (8%) y en departamento 80 (5%).

b. ¿Qué sucede con las filas eliminadas?

   La eliminación de los empleados del departamento 50 se revierte con el ROLLBACK TO B, por lo que esas filas permanecen en la tabla.

c. ¿Cómo puedes verificar los cambios antes y después del COMMIT?

   Antes del COMMIT, puedes hacer un SELECT en la misma sesión para ver los cambios (no serán visibles en otras sesiones).

   Después del COMMIT, los cambios quedan permanentes y ya son visibles desde cualquier sesión.