En este laboratorio trabajaremos excepciones, transacciones, paquetes, y vistas.

## Manejo de Excepciones en PL/SQL

Permite manejar los errores de ejecución de manera adecuada.

```
BEGIN
    execution_block
EXCEPTION
    WHEN exception_name THEN
        exception handling code
    WHEN exception_name THEN
        exception handling code
    …
    WHEN OTHERS THEN
       default_exception_handling_code
 END;
```

Ejemplo:

```
DECLARE liquidez NUMBER(10,2);
BEGIN
   SELECT activosCtes / pasivosCtes INTO liquidez
   FROM Balance;
   INSERT INTO resumen VALUES ('Liquidez', liquidez);
EXCEPTION
   WHEN ZERO_DIVIDE THEN
        INSERT INTO resumen VALUES ('Liquidez', NULL);
   WHEN OTHERS THEN
        dbms_output.put_line('Error en el cálculo de la liquidez');
END;
```

Excepciones definidas por el usuario:

```
CREATE OR REPLACE TRIGGER insEmpleado
BEFORE INSERT OR UPDATE ON EMPLEADO FOR EACH ROW
DECLARE
  invalidEmail  EXCEPTION;
BEGIN
   IF INSTR(:NEW.email, '@') = 0 THEN
      RAISE invalidEmail;
   END IF;
EXCEPTION
  WHEN invalidEmail THEN
    RAISE_APPLICATION_ERROR (-20000, 'Dirección de correo con formato
incorrecto');
END;
```

Excepciones del sistema no nombradas:

```
DECLARE liquidez NUMBER(2);
```

```
BEGIN
   SELECT activosCtes / pasivosCtes INTO liquidez
   FROM Balance;
   INSERT INTO resumen VALUES ('Liquidez', liquidez);
EXCEPTION
   WHEN OTHERS THEN
       dbms_output.put_line(SQLCODE || ' ' || SQLERRM(SQLCODE));
END;
```

Nombrar excepciones del sistema no nombradas:

```
DECLARE
    Number_over EXCEPTION;
    PRAGMA EXCEPTION_INIT (Number_over, -6502);
    liquidez NUMBER(2);
BEGIN
   SELECT activosCtes / pasivosCtes INTO liquidez
   FROM Balance;
   INSERT INTO resumen VALUES ('Liquidez', liquidez);
EXCEPTION
   WHEN Number_over THEN
       dbms_output.put_line('Error: Valor supera el máximo permitido');
END;
```

Lista de codigos de error: https://docs.oracle.com/cd/E11882_01/server.112/e17766/e0.htm

## Transacciones

Una transacción es una unidad lógica de trabajo que contiene una o más instrucciones SQL y que garantiza las propiedades ACID.

En Oracle la transacción inicia con la primera sentencia SQL y termina con COMMIT o ROLLBACK. También se pueden definir SAVE POINTS, marcadores intermedios que dividen una transacción en partes más pequeñas, y que dan la opción de hacer el ROLLBACK hasta un SAVE POINT determinado.

Ejemplo:

```
CREATE OR REPLACE PROCEDURE ejemplo (per NUMBER) AS
       activo NUMBER(12); pasivo NUMBER(12); invent NUMBER(12);
BEGIN
      SELECT activosCtes, pasivosCtes, inventario INTO activo, pasivo, invent
      FROM balance WHERE periodo = per;
      BEGIN
             SAVEPOINT cero;
             INSERT INTO resumen VALUES ('Liquidez', (activo/pasivo));
             EXCEPTION WHEN OTHERS THEN ROLLBACK TO cero;
```

```
        END;
        BEGIN
           SAVEPOINT uno;
           INSERT INTO resumen VALUES ('Capital financiero', (activo - pasivo));
           EXCEPTION WHEN OTHERS THEN ROLLBACK TO uno;
        END;
        BEGIN
           SAVEPOINT dos;
           INSERT INTO resumen VALUES ('Prueba acida', (activo - invent) / pasivo);
           EXCEPTION WHEN OTHERS THEN ROLLBACK TO dos;
        END;
        COMMIT;
END;
```

## Paquetes

Un paquete es un objeto que agrupa tipos, variables, constantes, subprogramas, cursores y excepciones. Consta de la especificación, que declara los elementos públicos, y el cuerpo, que define los elementos públicos y privados.

```
CREATE [OR REPLACE] PACKAGE nombre IS | AS
      -- definiciones públicas de tipos, variables y cursores
      -- especificaciones de subprogramas
END ;

CREATE [OR REPLACE] PACKAGE BODY nombre IS | AS
      -- Declaraciones privadas
      -- Cuerpo de subprogramas
 END ;
```

Los elementos del paquete se usan anteponiendo el nombre del paquete, separado por coma:

```
package_name.type_name
package_name.item_name
package_name.subprogram_name
```

Ejemplo:

```
CREATE OR REPLACE PACKAGE pckEmpleado IS
    TYPE t_empleado IS RECORD (empleados%rowtype) ;
    vp_empresa NUMBER := 1;

   PROCEDURE insEmpleado (pCC empleado.cc%TYPE,
             pNombre empleado.nombre%TYPE,
             pEmail empleado.email%TYPE);

   FUNCTION nombreEmpleado (pCC empleado.cc%TYPE)
         RETURN VARCHAR2;
```

```
END;

CREATE OR REPLACE PACKAGE BODY pckEmpleado IS
   PROCEDURE insEmpleado (pCC empleado.cc%TYPE,
             pNombre empleado.nombre%TYPE,
             pEmail empleado.email%TYPE) IS
   BEGIN
       INSERT INTO empleado VALUES (pCC, pNombre, 0, pEmail);
   END;

   FUNCTION nombreEmpleado (pCC empleado.cc%TYPE)
   RETURN VARCHAR2 IS
       nom empleado.nombre%TYPE;
   BEGIN
       SELECT nombre INTO nom FROM empleado WHERE cc = pCC;
       RETURN nom;
   END;
END;
```

Ejemplo del llamado a la función del paquete:

```
BEGIN
    DBMS_OUTPUT.PUT_LINE(pckEmpleado.nombreEmpleado(101));
END;
```

## Vistas

Permiten darle un nombre a una consulta de la base de datos para después usalas como una tabla.

```
CREATE VIEW name AS query;
```

Ejemplo:

```
CREATE VIEW fechasCliente AS ( SELECT cc, nombre, fecha
FROM Cliente NATURAL JOIN Compra );

SELECT COUNT(DISTINCT fecha) FROM fechasCliente;
```

## Ejercicios

Usando el esquema de base de datos creado y modificado en los laboratorios anteriores, escriba los comandos para dar respuesta a los requerimientos especificados en esta sección.

1. Seleccione 2 de las funciones o procedimientos que implementó en los laboratorios anteriores, en las que identifique que puede incluir manejo de excepciones, y agregueles el manejo de las excepciones que puedan ocurrir (excepciones del sistema nombradas o no, o excepciones de usuario).

2. Dado el siguiente procedimiento que crea un registro en infraccionParte, y sino está creado, crea también el parte; modifiquelo para agregar manejo transaccional y de excepciones.

```
CREATE OR REPLACE PROCEDURE creaInfParte
        (nroP NUMBER, cond NUMBER, placaP VARCHAR, codInf NUMBER, idInf NUMBER,
         salMin NUMBER)
AS
        existe NUMBER(2) DEFAULT 0;
        vMulta NUMBER(12);
BEGIN
    -- 1. Si no existe el parte, lo crea
    SELECT COUNT(*) INTO existe FROM Parte WHERE nroParte = nroP;
    IF existe = 0 THEN
        INSERT INTO Parte VALUES (nroP, SYSDATE, cond, placaP);
    END IF;

    -- 2. Obtiene el valor de la multa
    SELECT multaSalariosMin * salMin INTO vMulta FROM Infraccion
    WHERE  codigo = codInf;

    -- 3. Crea el registro de la infraccion (verificar que el orden de los atributos es correcto según su tabla)
    INSERT INTO infraccionParte
    VALUES ( codInf, nroP, vMulta, idInf);
END;
```

3. Cree una vista para la siguiente consulta: Liste para cada ciudad, por cada mes y año, el valor total de las multas de ese mes de los carros matriculados en esa ciudad.

4. Use la vista que creó en el punto anterior en una consulta que calcule el promedio del valor mensual de las multas en los meses del año 2019.

5. Seleccione 3 funciones que haya desarrollado y cree un paquete con ellas.

Al finalizar la sesión, cada estudiante debe enviar el script a mcpabon@javerianacali.edu.co.