Automatización de Procesos en Zapatería Zapataz

La empresa Zapatería Zapataz registra ventas en distintas ciudades de Chile. Los procesos actuales de cálculo manual de comisiones y pagos a vendedores son lentos, propensos a errores y poco escalables, afectando la eficiencia operativa y precisión en pagos.

Integrantes: Cristofer Lobos, Remi Garcia, Sebastian Rojas

Introducción al Problema: Zapatería Zapataz

La empresa Zapatería Zapataz registra ventas en distintas ciudades de Chile.

- **Procesos actuales:** cálculo manual de comisiones y pagos a vendedores.
- **Problema:** procesos lentos, propensos a errores y poco escalables.
- **Impacto:** afecta eficiencia operativa y precisión en pagos.

Objetivo

Automatizar cálculos con PL/SQL para asegurar eficiencia y confiabilidad.

Datos a Procesar

Vendedores

RUT, sueldo, comisión, escolaridad, fecha de contrato

Documentos

Clientes y documentos de venta (boletas/facturas)

Productos

Detalle de productos vendidos (nacionales e importados)

Tramos

Tablas de antigüedad y escolaridad para bonos

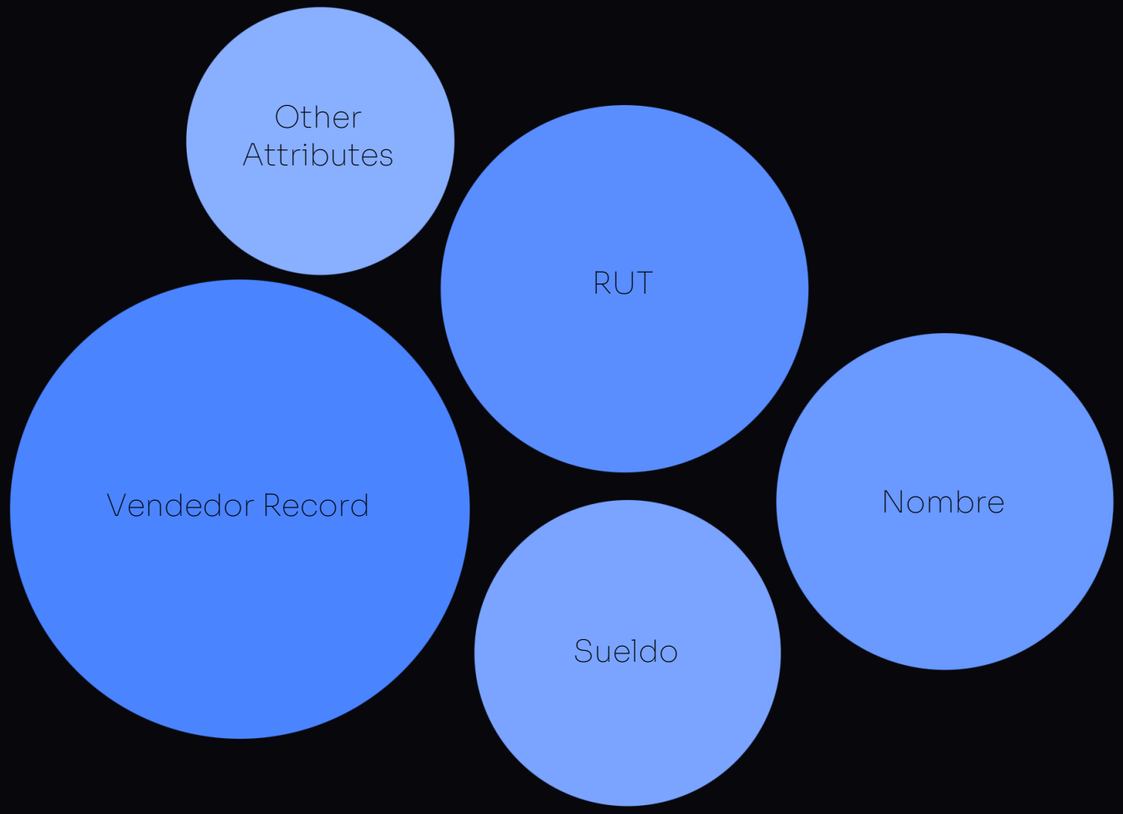
Resultado esperado: Reportes confiables de ventas, comisiones y pagos.

Uso de Tipos de Datos Compuestos

RECORD

Agrupa información de un vendedor en una sola variable

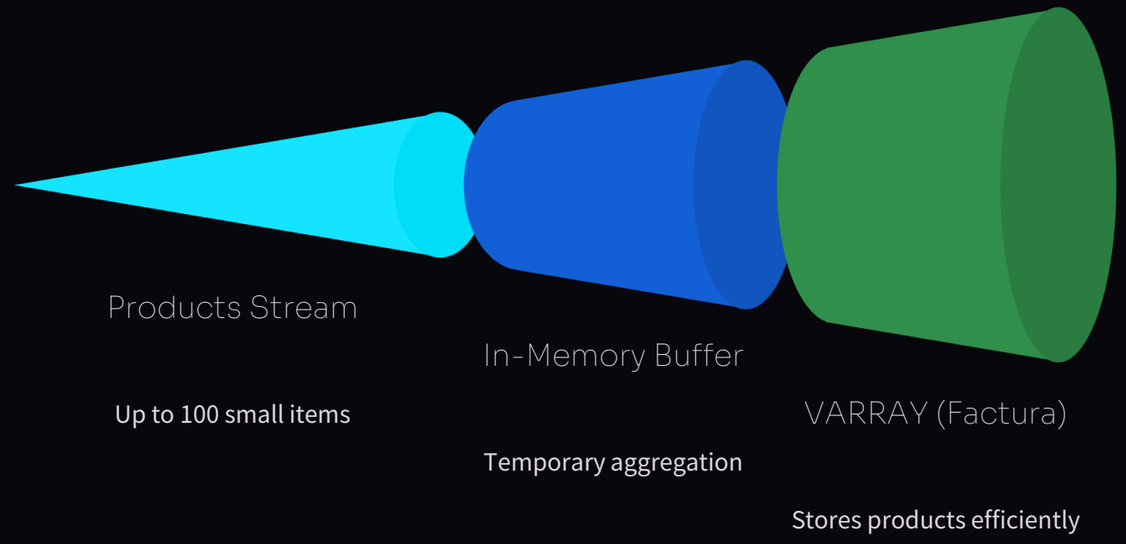
Facilita manipulación de atributos relacionados (rut, nombre, sueldo, etc.)



VARRAY

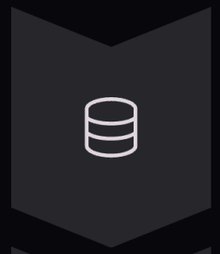
Permite almacenar hasta 100 productos por factura en memoria

Eficiencia: menos accesos a disco y operaciones más rápidas



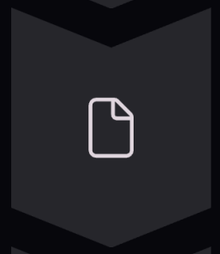
Justificación: simplificación y eficiencia en el manejo de datos complejos.

Cursores y Bucles Anidados



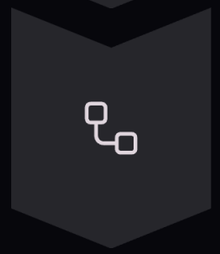
Cursores sin parámetros

Listar todos los vendedores



Cursores con parámetros

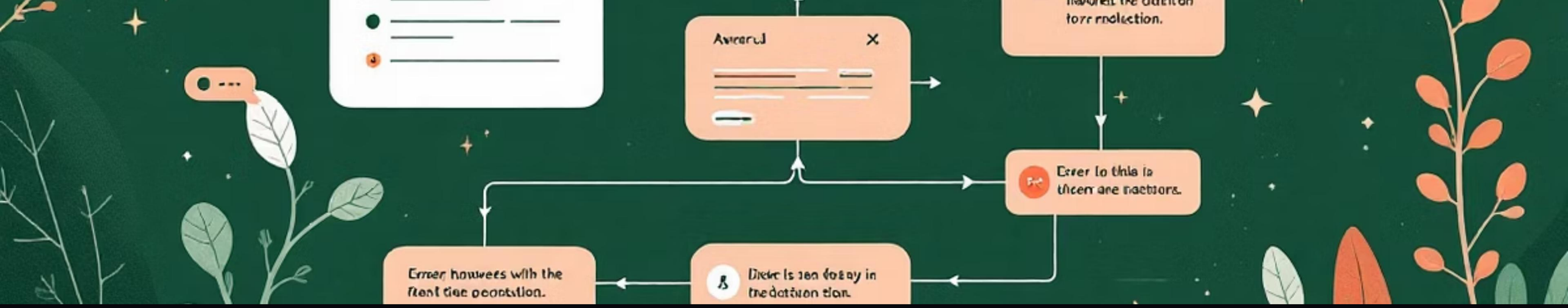
Obtener facturas por vendedor y productos por factura



Bucles anidados

Vendedor → Factura → Detalle

> Refleja la estructura jerárquica de los datos (1:N:N)
>
> **Justificación:** modularidad, eficiencia y claridad en el procesamiento.



Integración de Excepciones

Excepciones predefinidas (Oracle)

NO_DATA_FOUND, TOO_MANY_ROWS, ZERO_DIVIDE, OTHERS

Excepciones personalizadas

Ejemplo: ex_sin_facturas para vendedores sin ventas

Integración en bloques PL/SQL

Sección EXCEPTION al final de cada bloque. Captura errores previstos y no previstos

Beneficio: robustez, continuidad del proceso y aseguramiento de integridad de datos.

Beneficios del Manejo de Excepciones

Prevenir errores críticos

Evita que fallos interrumpan el flujo completo del proceso

Integridad de datos

Permite rollback, valores de seguridad y registro de incidencias

Robustez del sistema

Los vendedores siempre obtendrán un cálculo de pago

Mejor trazabilidad

Facilita detección y corrección de problemas mediante logs

Implementación del Sistema

```
DECLARE
  -- VARRAY para guardar productos de una factura
  TYPE t_productos IS VARRAY(100) OF NUMBER;     -- Definimos un tipo de arreglo (máximo 100) de números (códigos de producto)
  v_productos t_productos;                       -- Variable que usará ese VARRAY para almacenar los productos de cada factura

  -- RECORD para agrupar datos de un vendedor
  TYPE t_vendedor IS RECORD (                    -- Definimos un tipo de registro que agrupa info de un vendedor
    rut          VENDEDOR.RUTVENDEDOR%TYPE,      -- Rut del vendedor (mismo tipo que en la tabla VENDEDOR)
    nombre       VENDEDOR.NOMBRE%TYPE,           -- Nombre del vendedor
    sueldo_base  VENDEDOR.SUELDO_BASE%TYPE,      -- Sueldo base
    comision     VENDEDOR.COMISION%TYPE,         -- Porcentaje de comisión
    fecha_cont   VENDEDOR.FECHA_CONTRATO%TYPE,   -- Fecha de contrato
    escolaridad  VENDEDOR.ESCOLARIDAD%TYPE       -- Nivel de escolaridad
  );
  r_vend t_vendedor;                             -- Variable del tipo RECORD para cargar datos de cada vendedor
```

01

Definición de estructuras

Creación de RECORD y VARRAY para manejo eficiente de

```
-- Cursores
CURSOR c_vendedores IS
  SELECT RUTVENDEDOR, NOMBRE, SUELDO_BASE, COMISION, FECHA_CONTRATO, ESCOLARIDAD
  FROM VENDEDOR;                               -- Cursor que devuelve todos los vendedores con sus datos

CURSOR c_ventas_vendedor(p_rut VENDEDOR.RUTVENDEDOR%TYPE) IS
  SELECT NUMFACTURA, TOTAL
  FROM FACTURA
  WHERE RUTVENDEDOR = p_rut;                   -- Cursor con parámetro: devuelve las facturas de un vendedor específico

CURSOR c_detalle_factura(p_numfactura FACTURA.NUMFACTURA%TYPE) IS
  SELECT CODPRODUCTO, TOTALLINEA
  FROM DETALLE_FACTURA
  WHERE NUMFACTURA = p_numfactura;             -- Cursor con parámetro: devuelve el detalle de una factura (productos y montos)

-- Variables de cálculo
total_ventas     NUMBER := 0;                  -- Total de ventas acumuladas por vendedor
monto_comision   NUMBER := 0;                  -- Monto de la comisión calculada
porc_antiguedad  NUMBER := 0;                  -- Porcentaje por antigüedad
porc_escolaridad NUMBER := 0;                  -- Porcentaje por escolaridad
total_bonos      NUMBER := 0;                  -- Bonos sumados (antigüedad + escolaridad)
total_pagar      NUMBER := 0;                  -- Monto total final a pagar al vendedor
```

02

Configuración de cursores

Cursores explícitos con y sin parámetros para procesamiento jerárquico

Implementación del Sistema

03

Cálculos automatizados

Comisiones, bonos por antigüedad y escolaridad

```
BEGIN
  -- Recorremos todos los vendedores
  FOR v IN c_vendedores LOOP                    -- Bucle que recorre cada fila devuelta por el cursor c_vendedores

    -- llenamos el record
    r_vend.rut := v.rutvendedor;                -- Guardamos rut en el RECORD
    r_vend.nombre := v.nombre;                  -- Guardamos nombre
    r_vend.sueldo_base := v.sueldo_base;        -- Guardamos sueldo base
    r_vend.comision := NVL(v.comision, 0);      -- Guardamos comisión (si es NULL, lo reemplazamos por 0)
    r_vend.fecha_cont := v.fecha_contrato;      -- Guardamos fecha de contrato
    r_vend.escolaridad := v.escolaridad;        -- Guardamos escolaridad

    -- inicializamos valores
    total_ventas := 0;                          -- Reiniciamos el acumulador de ventas
    monto_comision := 0;                        -- Reiniciamos comisión
    total_bonos := 0;                           -- Reiniciamos bonos
    total_pagar := 0;                           -- Reiniciamos total a pagar

    -- calcular antigüedad
    DECLARE anos NUMBER := 0;                   -- Variable local para almacenar la antigüedad en años
    BEGIN
      IF r_vend.fecha_cont IS NOT NULL THEN     -- Si la fecha de contrato no es nula...
        SELECT TRUNC(MONTHS_BETWEEN(SYSDATE, r_vend.fecha_cont) / 12)
        INTO anos FROM DUAL;                    -- Calculamos la diferencia en años entre hoy y la fecha de contrato
      END IF;
```

Implementación del Sistema

```
  BEGIN
    SELECT PORCENTAJE
    INTO porc_antiguedad
    FROM TRAMO_ANTIGUEDAD
    WHERE anos BETWEEN ANNOS_CONT_INF AND ANNOS_CONT_SUP; -- Buscamos el porcentaje correspondiente en la tabla de tramos
  EXCEPTION
    WHEN NO_DATA_FOUND THEN
      porc_antiguedad := 0;                -- Si no existe tramo para esa antigüedad, dejamos 0
  END;
END;

-- calcular escolaridad
BEGIN
  SELECT PORC_ASIG_ESCOLARIDAD
  INTO porc_escolaridad
  FROM TRAMO_ESCOLARIDAD
  WHERE ID_ESCOLARIDAD = r_vend.escolaridad; -- Buscamos porcentaje asociado al nivel de escolaridad
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    porc_escolaridad := 0;                 -- Si no existe registro, dejamos 0
END;
```

```
   -- recorrer facturas
   FOR f IN c_ventas_vendedor(r_vend.rut) LOOP -- Recorremos cada factura del vendedor
     v_productos := t_productos();             -- Inicializamos el VARRAY vacío para guardar productos de esta factura

     -- recorrer detalle de factura
     FOR d IN c_detalle_factura(f.numfactura) LOOP -- Recorremos los productos de la factura
       total_ventas := total_ventas + NVL(d.totallinea, 0); -- Sumamos cada línea de la factura al total de ventas
       v_productos.EXTEND;                     -- Aumentamos el tamaño del VARRAY
       v_productos(v_productos.LAST) := d.codproducto; -- Guardamos el código del producto en la última posición
     END LOOP;

     DBMS_OUTPUT.PUT_LINE('Factura '||f.numfactura||' - Productos: '|| v_productos.COUNT);
     -- Mostramos número de factura y cuántos productos tenía
   END LOOP;

   -- cálculos finales
   monto_comision := total_ventas * r_vend.comision; -- Calculamos la comisión según el total vendido
   total_bonos := ROUND(r_vend.sueldo_base * ((porc_antiguedad + porc_escolaridad) / 100), 2);
   -- Calculamos bonos como % del sueldo base (antigüedad + escolaridad), redondeado a 2 decimales
   total_pagar := r_vend.sueldo_base + monto_comision + total_bonos; -- Sumamos todo para obtener el pago final
```

04

Control de excepciones

Manejo robusto de errores predefinidos y personalizados

Implementación del Sistema

```
    -- mostrar resumen
    DBMS_OUTPUT.PUT_LINE(
      'Vendedor: ' || r_vend.nombre ||                          -- Nombre
      ' | Ventas: ' || TO_CHAR(total_ventas, '999G999G999') || -- Total de ventas con formato
      ' | Comisión: ' || TO_CHAR(monto_comision, '999G999G999') || -- Comisión calculada
      ' | Bonos: ' || TO_CHAR(total_bonos, '999G999G999') ||     -- Bonos
      ' | Total a pagar: ' || TO_CHAR(total_pagar, '999G999G999') -- Pago final
    );
    DBMS_OUTPUT.PUT_LINE('-------------------------------------------------'); -- Separador visual
  END LOOP;

-- manejo de excepciones
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Error inesperado: ' || SQLERRM); -- Captura cualquier error y lo muestra
END;
/
```

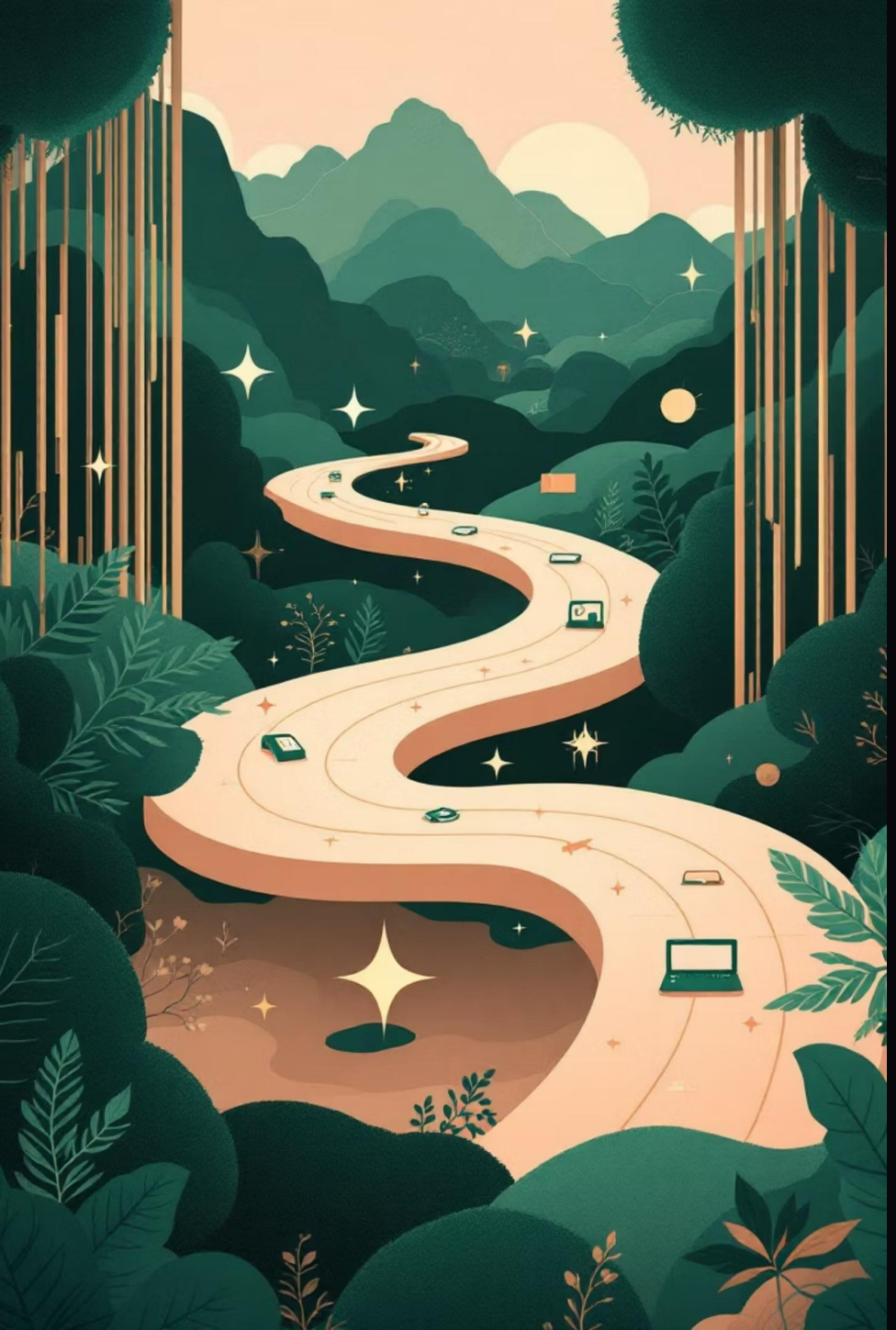
```
Procedimiento PL/SQL terminado correctamente.
Factura 11520 - Productos: 1
Factura 11522 - Productos: 3
Factura 11525 - Productos: 1
Vendedor: LEOPOLDO ROJAS | Ventas:      339.400 | Comisión:       33.940 | Bonos:       28.800 | Total a pagar:      302.740
-------------------------------------------------
Factura 11521 - Productos: 2
Factura 11527 - Productos: 1
Factura 11528 - Productos: 1
Vendedor: MARIO SOTO  | Ventas:      208.400 | Comisión:       41.680 | Bonos:       34.450 | Total a pagar:      341.130
-------------------------------------------------
Factura 11523 - Productos: 1
Factura 11524 - Productos: 1
Vendedor: SALVADOR ALVARADO | Ventas:       95.955 | Comisión:       28.787 | Bonos:       35.000 | Total a pagar:      313.787
-------------------------------------------------
Factura 11526 - Productos: 1
Factura 11529 - Productos: 1
Factura 11530 - Productos: 1
Factura 11531 - Productos: 1
Vendedor: LUIS MUÑOZ | Ventas:       88.600 | Comisión:       35.440 | Bonos:       32.400 | Total a pagar:      337.840
-------------------------------------------------

Procedimiento PL/SQL terminado correctamente.
```

04

Resultado final

Resultado obtenido tras la implementacion del Script.



Recomendaciones Futuras

1

Migración a Procedimientos

Convertir bloques anónimos en procedimientos almacenados reutilizables

2

Parámetros Dinámicos

Implementar fechas de corte variables y rangos personalizables

3

Extensiones Funcionales

Incorporar cálculo de impuestos y reportes analíticos por período

Conclusiones

El proyecto implementa exitosamente un sistema automatizado de comisiones que cumple con todos los requisitos técnicos establecidos.

Impacto Técnico

Uso estratégico de RECORD, VARRAY, cursores complejos y manejo robusto de excepciones

Impacto Operacional

Solución escalable y mantenible para procesamiento de datos transaccionales

Valor Agregado

Información confiable para toma de decisiones y planeación estratégica

Zapatería Zapataz ahora cuenta con un sistema robusto, eficiente y preparado para el crecimiento futuro.