ADMINISTRACIÓN DE BASE DE DATOS.

Tema Nº7:Programación PL/SQL III.

Indicador de logro Nº7:Implementa Cursores, utilizando la programación estructurada PL/SQL.

**TEMA 01 Teoría de los**

Imagen que contiene Icono

Descripción generada automáticamente

**TEMA Nº7:**

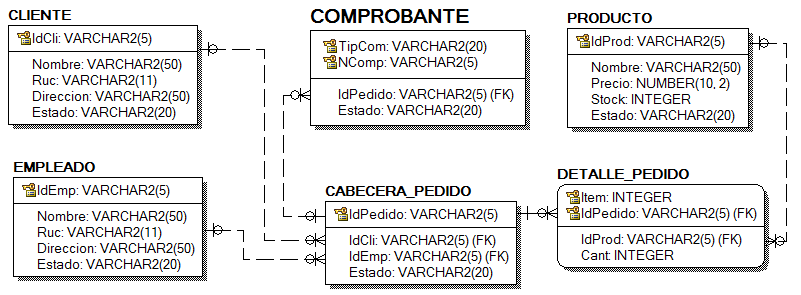
Programación PL/SQL III.

**Subtema 7.1:**

Implementa cursores explícitos e implícitos, y de actualización, utilizando programación estructuradas en PL/SQL basada en reglas de negocio.

**CURSORES IMPLICITOS y EXPLICITOS USANDO VARIAS TABLAS, CASO PRÁCTICO PARA DESARROLLAR**

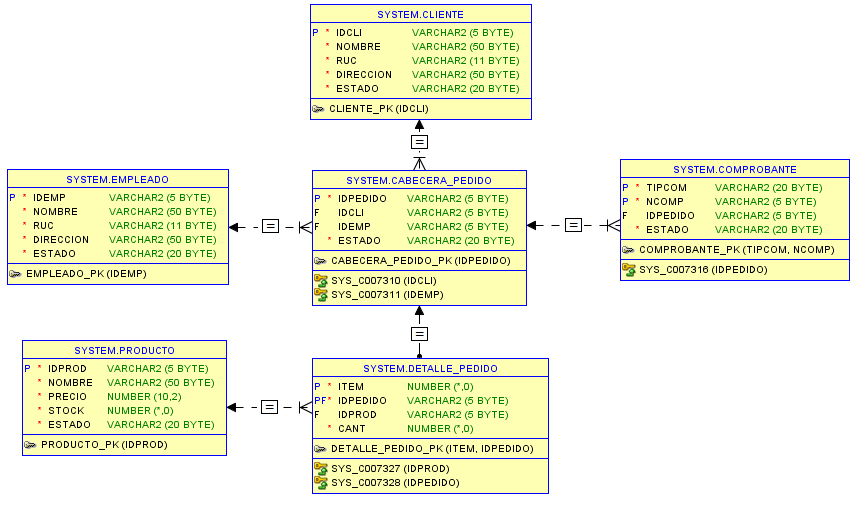
Se tiene el siguiente modelo relacional. **COMPROBANTE DE PAGO**.



El presente modelo relacional permite generar comprobantes de pago, boletas y facturas a partir de un determinado pedido, para ello se requiere la colaboración de empleados y clientes, dado que el cliente solicita el producto y el empleado atiende al cliente.

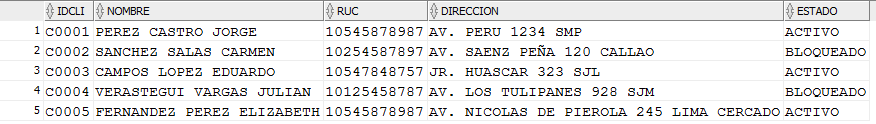
El Script de base de datos se encuentra en el archivo: **Semana07.sql**

**Modelo Relacional en Oracle, Diagrama de Base de Datos:**

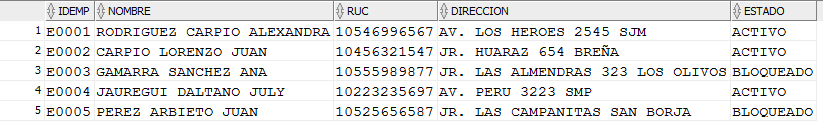


**Cuyos datos son los siguientes:**

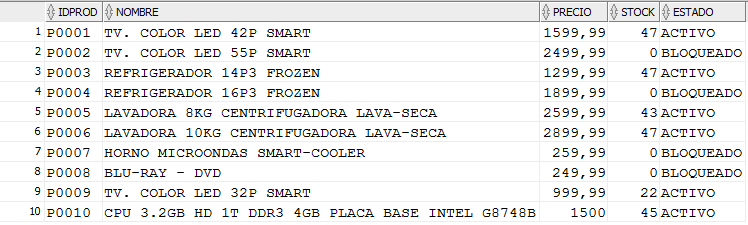
**TABLA CLIENTE:**



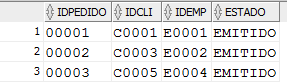
**TABLA EMPLEADO:**



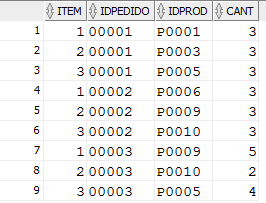
**TABLA PRODUCTO:**



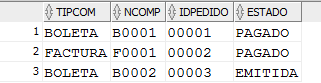
**TABLA PEDIDO:**



**TABLA DETALLE\_PEDIDO:**



**TABLA COMPROBANTE:**



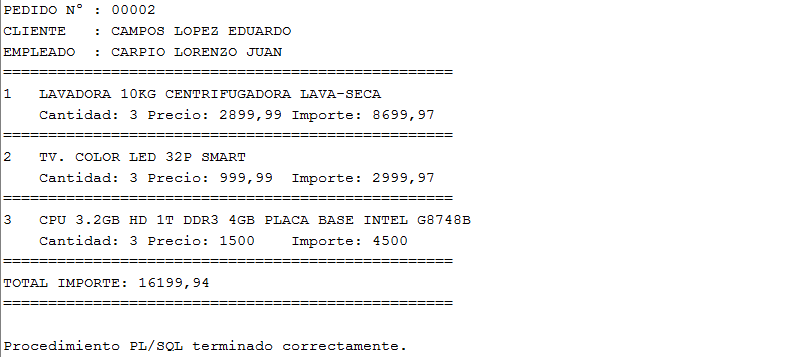
**Ejercicio 1:**

Se pide crear un cursor que sea capaz de mostrar los pedidos realizados, según la tabla **PEDIDO,** esta tabla tiene 3 pedidos, estos son: **‘00001’, ‘00002’ y ‘00003’**, **para ello se debe ingresar el número del pedido**, por intermedio de variables de **sustitución &**, y brindar la información solicitada, tal como se muestran en las siguientes imágenes.

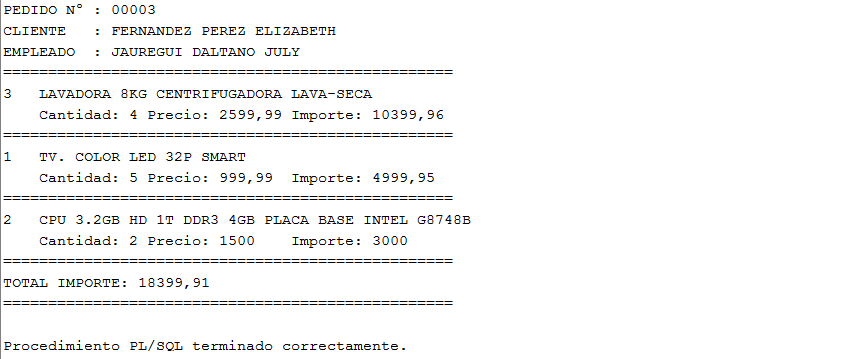
**NUMERO DE PEDIDO ‘00001’**



**NUMERO DE PEDIDO ‘00002’**



**NUMERO DE PEDIDO ‘00003’**



Cada vez que ejecute el cursor, este pedirá el **número del pedido**, en este ejercicio se usan todas las tablas, realizando diferentes consultas y obteniendo los registros solicitados.

**SOLUCION:**

SET SERVEROUTPUT ON;

DECLARE

It DETALLE\_PEDIDO.ITEM%TYPE;

CodigoP CABECERA\_PEDIDO.IDPEDIDO%TYPE:= **&CodigoP;**

CodigoC CABECERA\_PEDIDO.IDCLI%TYPE;

CodigoE CABECERA\_PEDIDO.IDEMP%TYPE;

Est CABECERA\_PEDIDO.ESTADO%TYPE;

NombreCli CLIENTE.NOMBRE%TYPE;

NombreEmp EMPLEADO.NOMBRE%TYPE;

NombreProd PRODUCTO.NOMBRE%TYPE;

CantProd DETALLE\_PEDIDO.CANT%TYPE;

PreProd PRODUCTO.PRECIO%TYPE;

ImpCompra NUMBER (10,2);

TOTAL\_IMPORTE NUMBER (10,2):=0;

CURSOR DETALLE IS (SELECT D.ITEM, P.NOMBRE, D.CANT,

P.PRECIO, (D.CANT\* P.PRECIO) IMPORTE

FROM DETALLE\_PEDIDO D, PRODUCTO P

WHERE D.IDPROD = P.IDPROD AND

D.IDPEDIDO = CodigoP);

BEGIN

SELECT IDPEDIDO,IDCLI, IDEMP, ESTADO

INTO CodigoP, CodigoC, CodigoE, Est

FROM CABECERA\_PEDIDO WHERE IDPEDIDO=CodigoP;

SELECT NOMBRE INTO NombreCli FROM CLIENTE WHERE IDCLI=CodigoC;

SELECT NOMBRE INTO NombreEmp FROM EMPLEADO WHERE IDEMP=CodigoE;

DBMS\_OUTPUT.PUT\_LINE('==================================================');

DBMS\_OUTPUT.PUT\_LINE('IMPORTADORA - LAS CAMELIAS S.A');

DBMS\_OUTPUT.PUT\_LINE('==================================================');

DBMS\_OUTPUT.PUT\_LINE('PEDIDO Nº : ' || CodigoP);

DBMS\_OUTPUT.PUT\_LINE('CLIENTE : ' || NombreCli);

DBMS\_OUTPUT.PUT\_LINE('EMPLEADO : ' || NombreEmp);

DBMS\_OUTPUT.PUT\_LINE('==================================================');

OPEN DETALLE;

FETCH DETALLE INTO It, NombreProd, CantProd, PreProd, ImpCompra;

WHiLE DETALLE%Found

LOOP

DBMS\_OUTPUT.PUT\_LINE(It || CHR(9) || NombreProd || CHR(9));

DBMS\_OUTPUT.PUT\_LINE(CHR(9) || 'Cantidad: ' ||CantProd || CHR(9) ||

'Precio: ' ||PreProd || CHR(9) || 'Importe: ' ||ImpCompra);

DBMS\_OUTPUT.PUT\_LINE('==================================================');

FETCH DETALLE INTO It, NombreProd, CantProd, PreProd, ImpCompra;

END LOOP;

SELECT SUM(D.CANT\* P.PRECIO) INTO TOTAL\_IMPORTE

FROM DETALLE\_PEDIDO D, PRODUCTO P

WHERE D.IDPROD = P.IDPROD AND D.IDPEDIDO = CodigoP;

DBMS\_OUTPUT.PUT\_LINE('TOTAL IMPORTE: ' || TOTAL\_IMPORTE);

DBMS\_OUTPUT.PUT\_LINE('==================================================');

CLOSE DETALLE;

END;

OPEN DETALLE;

FETCH DETALLE INTO It, NombreProd, CantProd, PreProd, ImpCompra;

WHiLE DETALLE%Found

LOOP

DBMS\_OUTPUT.PUT\_LINE(It || CHR(9) || NombreProd || CHR(9));

DBMS\_OUTPUT.PUT\_LINE(CHR(9) || 'Cantidad: ' ||CantProd || CHR(9) ||

'Precio: ' ||PreProd || CHR(9) || 'Importe: ' ||ImpCompra);

DBMS\_OUTPUT.PUT\_LINE('==================================================');

FETCH DETALLE INTO It, NombreProd, CantProd, PreProd, ImpCompra;

END LOOP;

SELECT SUM(D.CANT\* P.PRECIO) INTO TOTAL\_IMPORTE

FROM DETALLE\_PEDIDO D, PRODUCTO P

WHERE D.IDPROD = P.IDPROD AND D.IDPEDIDO = CodigoP;

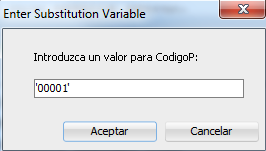
DBMS\_OUTPUT.PUT\_LINE('TOTAL IMPORTE: ' || TOTAL\_IMPORTE);

DBMS\_OUTPUT.PUT\_LINE('==================================================');

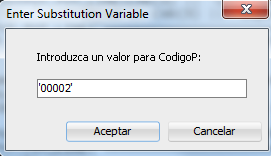
CLOSE DETALLE;

END;

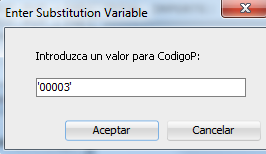
La ejecución del programa por primera vez, este solicitará el código del pedido para la variable de sustitución: **&CodigoP**: en este ‘00001’.



La ejecución del programa por segunda vez, este solicitará el código del pedido para la variable de sustitución: **&CodigoP**: en este ‘00002’.



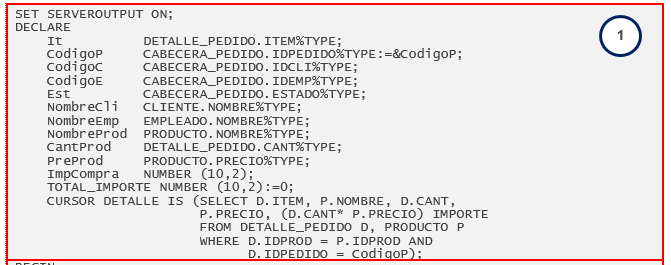
La ejecución del programa por tercera vez, este solicitará el código del pedido para la variable de sustitución: **&CodigoP**: en este ‘00003’.



**EL PROGRAMA ESTÁ DIVIDIDO EN 5 SECCIONES**

**SECCION 1:**

En esta sección se declaran las variables y cursor necesario para los procesos a realizar, lea con detenimiento que variables son, en especial la variable **&CodigoP** (Numero de Pedido) que recibe un valor desde teclado, El cursor **DETALLE** proporcionara al programa el detalle o listado de productos que se han solicitado. Las tablas que se están usando en este cursor son: **DETALLE\_PEDIDO** y **PRODUCTO**. Las variables que usan **%TYPE** heredan el tipo de dato que corresponde al campo de una determinada tabla.



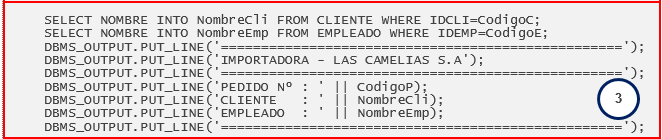
**SECCION 2:**

Al iniciar el programa, se realiza una consulta implícita, el cual captura: **IDPEDIDO**, **IDCLI**, **IDEMP** y **ESTADO**, y almacenarlos en variables temporales **INTO CodigoP, CodigoC, CodigoE, Est** consecutivamente, dependiendo del **IDPEDIDO**, los datos obtenidos serán usado en el proceso de la **sección 3.**



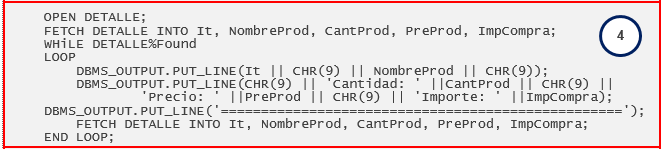
**SECCION 3:**

En esta sección se realizan dos consultas implícitas, la primera es para **capturar el nombre del cliente** y la segunda es para **capturar el nombre del empleado**, que se usaran para mostrar en la cabecera del documento.



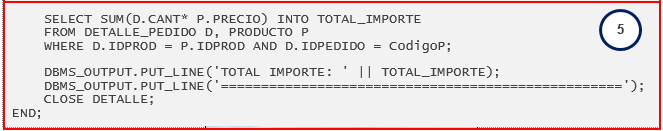
**SECCION 4:**

En esta sección se abre el cursor **DETALLE** y se procede a leer la data almacenada en el primer registro del cursor, y a continuación se usa un bucle con la finalidad de leer todo el cursor, la cantidad de veces que se requiera (**DETALLE%Found**), según la cantidad de vueltas que realice el bucle se irán imprimiendo los datos (**DBMS\_OUTPUT.PUT\_LINE**) del ítem, Nombre del producto, Precio, Cantidad y el importe de la compra.



**SECCION 5:**

En esta sección se realizar una consulta implícita, obteniendo el **total del importe** y mostrando dicho resultado en el comprobante. Finalmente cerrando el cursor.

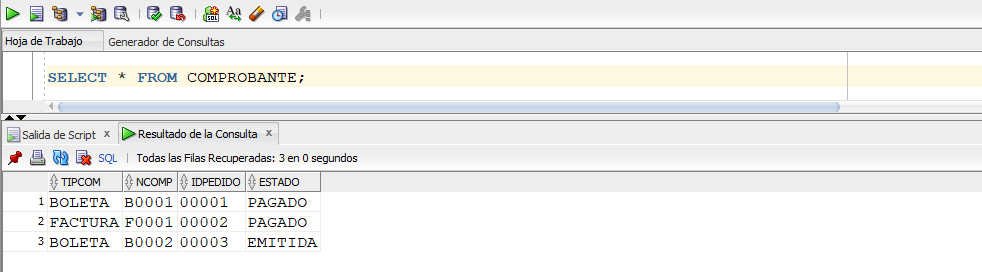


**FIN DEL EJERCICIO.**

**EJERCICIO 2:**

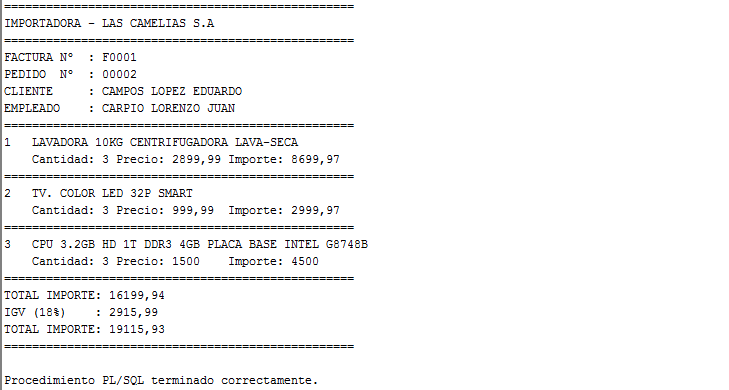
Se pide crear un cursor, que sea capaz de mostrar datos de las facturas que han sido realizadas como **PAGADAS**, según la tabla **COMPROBANTE,** esta tiene 3 documentos (1 factura y 2 boletas) estas son: ‘**F0001’**, ‘**B0001’** y ‘**B0002’**, tal como se muestra en la siguiente consulta:

**SELECT \* FROM COMPROBANTE;**



Para consultar el comprobante se debe ingresar el número del comprobante, por intermedio de variables de **sustitución** **&**, y brindar la información solicitada, tal como se muestran en las siguientes imágenes.

Por ejemplo, si ingresamos la factura ‘**F0001**’, mediante una variable de sustitución, se debe obtener el siguiente comprobante:



**SOLUCION:**

**SET SERVEROUTPUT ON;**

**DECLARE**

It DETALLE\_PEDIDO.ITEM%TYPE;

CodigoF COMPROBANTE.NCOMP%TYPE:= **&CodigoF**;

CodigoP CABECERA\_PEDIDO.IDPEDIDO%TYPE;

CodigoC CABECERA\_PEDIDO.IDCLI%TYPE;

CodigoE CABECERA\_PEDIDO.IDEMP%TYPE;

Est CABECERA\_PEDIDO.ESTADO%TYPE;

NombreCli CLIENTE.NOMBRE%TYPE;

NombreEmp EMPLEADO.NOMBRE%TYPE;

NombreProd PRODUCTO.NOMBRE%TYPE;

CantProd DETALLE\_PEDIDO.CANT%TYPE; PreProd PRODUCTO.PRECIO%TYPE;

ImpCompra NUMBER(10,2); TOTAL\_IMPORTE NUMBER(10,2):=0;

IGV NUMBER(10,2):=0; TOTAL\_GENERAL NUMBER(10,2):=0;

CURSOR DETALLE IS (SELECT D.ITEM, P.NOMBRE, D.CANT,

P.PRECIO, (D.CANT\* P.PRECIO) IMPORTE

FROM DETALLE\_PEDIDO D, PRODUCTO P, COMPROBANTE C

WHERE D.IDPROD = P.IDPROD AND

D.IDPEDIDO = C.IDPEDIDO AND

C.TIPCOM ='FACTURA' AND

C.NCOMP = CodigoF AND

C.ESTADO ='PAGADO');

**BEGIN**

SELECT CABECERA\_PEDIDO.IDPEDIDO, CABECERA\_PEDIDO.IDCLI,

CABECERA\_PEDIDO.IDEMP,COMPROBANTE.ESTADO

INTO CodigoP, CodigoC, CodigoE, Est

FROM CABECERA\_PEDIDO, COMPROBANTE

WHERE CABECERA\_PEDIDO.IDPEDIDO = COMPROBANTE.IDPEDIDO AND

COMPROBANTE.TIPCOM='FACTURA' AND

COMPROBANTE.NCOMP= CodigoF;

SELECT NOMBRE INTO NombreCli FROM CLIENTE WHERE IDCLI=CodigoC;

SELECT NOMBRE INTO NombreEmp FROM EMPLEADO WHERE IDEMP=CodigoE;

DBMS\_OUTPUT.PUT\_LINE('==================================================');

DBMS\_OUTPUT.PUT\_LINE('IMPORTADORA - LAS CAMELIAS S.A');

DBMS\_OUTPUT.PUT\_LINE('==================================================');

DBMS\_OUTPUT.PUT\_LINE('FACTURA Nº : ' || CodigoF);

DBMS\_OUTPUT.PUT\_LINE('PEDIDO Nº : ' || CodigoP);

DBMS\_OUTPUT.PUT\_LINE('CLIENTE : ' || NombreCli);

DBMS\_OUTPUT.PUT\_LINE('EMPLEADO : ' || NombreEmp);

DBMS\_OUTPUT.PUT\_LINE('==================================================');

OPEN DETALLE;

FETCH DETALLE INTO It, NombreProd, CantProd, PreProd, ImpCompra;

WHiLE DETALLE%Found

LOOP

DBMS\_OUTPUT.PUT\_LINE(It || CHR(9) || NombreProd || CHR(9));

DBMS\_OUTPUT.PUT\_LINE(CHR(9) || 'Cantidad: ' ||CantProd || CHR(9) ||

'Precio: ' ||PreProd || CHR(9) || 'Importe: ' ||ImpCompra);

DBMS\_OUTPUT.PUT\_LINE('==============================================');

FETCH DETALLE INTO It, NombreProd, CantProd, PreProd, ImpCompra;

END LOOP;

SELECT SUM(D.CANT\* P.PRECIO) INTO TOTAL\_IMPORTE FROM DETALLE\_PEDIDO D,

PRODUCTO P

WHERE D.IDPROD = P.IDPROD AND D.IDPEDIDO = CodigoP;

IGV := TOTAL\_IMPORTE \* 0.18;

TOTAL\_GENERAL := TOTAL\_IMPORTE + IGV;

DBMS\_OUTPUT.PUT\_LINE('TOTAL IMPORTE: ' || TOTAL\_IMPORTE);

DBMS\_OUTPUT.PUT\_LINE('IGV (18%) : ' || IGV);

DBMS\_OUTPUT.PUT\_LINE('TOTAL IMPORTE: ' || TOTAL\_GENERAL);

DBMS\_OUTPUT.PUT\_LINE('==================================================');

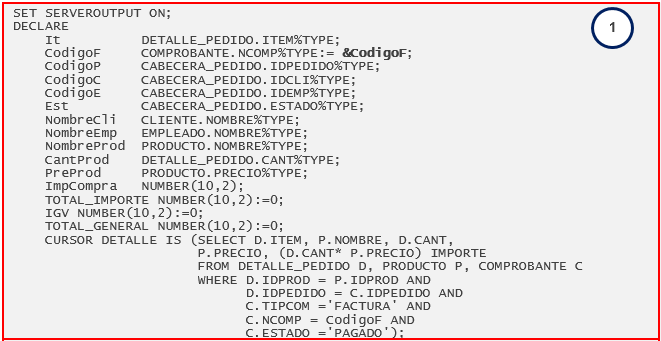
CLOSE DETALLE;

**END;**

**EL PROGRAMA ESTÁ DIVIDIDO EN 5 SECCIONES**

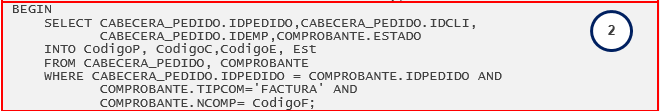
**SECCION 1:**

En esta sección se declaran las variables y cursor necesario para los procesos a realizar, lea con detenimiento que variables son, en especial la variable **&CodigoF** (Numero de Factura o Comprobante) que recibe un valor desde teclado, El cursor **DETALLE** proporcionara al programa el detalle o listado de productos que se han solicitado (**Cursor Explicito**). Las tablas que se están usando en este cursor son: **DETALLE\_PEDIDO**, **PRODUCTO** y **COMPROBANTE**. Las variables que usan **%TYPE** heredan el tipo de dato que corresponde al campo de una determinada tabla.



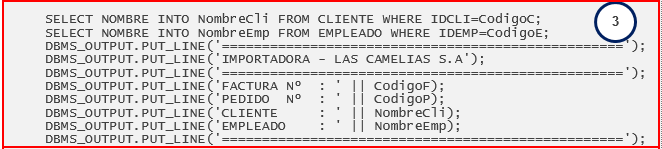
**SECCION 2:**

Al iniciar el programa, se realiza una consulta implícita, el cual captura: IDPEDIDO, IDCLI, IDEMP y ESTADO, y almacenarlos en variables temporales **INTO CodigoP, CodigoC, CodigoE, Est** consecutivamente, dependiendo del **Ncomp** (Numero de Comprobante), los datos obtenidos serán usado en el proceso de la sección 3.



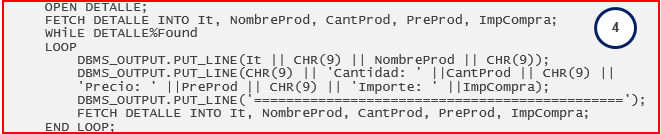
**SECCION 3:**

En esta sección, se realizan dos consultas implícitas, la primera es para **captura el nombre del cliente** y la segunda es para **captura el nombre del empleado**, que se usaran para mostrar en la cabecera del documento.



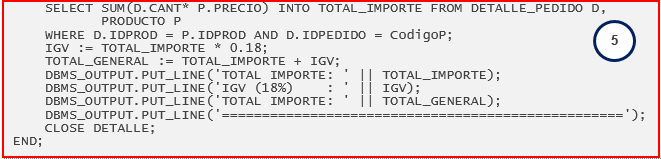
**SECCION 4:**

En esta sección se abre el cursor **DETALLE** y se procede a leer la data almacenada en el primer registro del cursor (**FETCH**), y a continuación se usa un bucle con la finalidad de leer todo el cursor la cantidad de veces que se requiera (**DETALLE%Found**), según la cantidad de vueltas que realice el bucle se irán imprimiendo los datos (**DBMS\_OUTPUT.PUT\_LINE**) de: **ítem, Nombre del producto, Precio, Cantidad y el importe de la compra**.



**SECCION 5:**

En esta sección se realizar una consulta implícita, obteniendo el **TOTAL IMPORTE** calculando **IGV y TOTAL\_GENERAL,** mostrando dichos resultados en el comprobante. Finalmente cerrando el cursor.



**FIN DEL EJERCICIO.**

**CONCLUSIONES Y RECOMENDACIONES DE LA EXPERIENCIA**

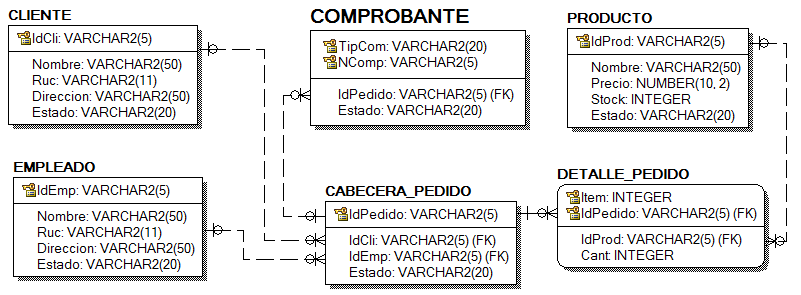
La codificación PL/SQL de Oracle es tan amplia, que ofrece a los estudiantes una alta gama de alternativas para la resolución de problemas, no difíciles de entender, con la prioridad de manejar los datos con suma responsabilidad por el programador Oracle.

Si una base de datos es demasiado grande, se debe hacer un rediseño de base de datos centralizada, para que esta sea una base de datos distribuida ya que se pueden almacenar los datos en localidades donde son utilizados con mayor frecuencia, de tal manera que la mayor parte de las operaciones sean sólo locales lo cual reduce el tráfico en la red.

**ACTIVIDAD VIRTUAL**

Revisar y analizar el material presentado en la Semana 07.

**SE TIENE EL SIGUIENTE MODELO RELACIONAL:**



**LA ESTRUCTURA Y DATOS DE LAS TABLAS DEL MODELO RELACIONAL SE ENCUENTRAN**

**EN EL ARCHIVO SCRIPT DE BASE DE DATOS.**

**Semana07\_PL\_SQL\_2.sql,**

**FAVOR DE DESCARGAR DICHO ARCHIVO.**

De acuerdo al material presentado en el Tema 07, Responda a las siguientes Preguntas:

**RESOLVER:**

Crear los siguientes programas anónimos PL/SQL:

1. Crear un cursor el cual permita mostrar un determinado comprobante de pago, con solamente ingresar el tipo de comprobante **(‘BOLETA’, ‘FACTURA’)** y el número del comprobante, estos pueden ser: **‘F0001’, ‘B0001’** y **‘B0002’**. Si en caso el comprobante no existe, agregue una excepción de no existencia.
2. Crear un cursor, el cual muestre los pedidos y sus montos correspondientes de la venta. Si en caso no existen pedidos, agregar una excepción de no existencia.