UNIVERSIDAD NACIONAL DE INGENIERIA

PROFESOR:

Eric Gustavo Coronel Castillo

CURSO:

Oracle PL SQL

TEMA:

Sistema de gestión de Ventas para la empresa mayorista de los Hermanos Yucra

INTEGRANTES:

Estudiantes
Cardenas Grandez, Jorge
Espinoza Casio, Eligio
Zarate Ramos, Carlos

PUNTOS QUE RESOLVER DEL EXAMEN FINAL DE ORACLE PL SQL

1. Explicar el caso 2
2. Generar el script 4
3. Script de carga de datos 4
4. Programar un paquete a un CRUD 5
5. Programar un proceso simple 6
6. Consulta de resumen de datos (2 consultas) 7
7. Proceso complejo. Procesar varias filas 8

RESOLUCION DEL EXAMEN FINAL DE ORACLE PL SQL

1. Explicar el caso

Empresa que se encarga de las ventas de productos de abarrote tanto mayoristas como minoristas. Por lo que se trata de cubrir la funcionalidad principalmente de registrar los pedidos de los clientes para así poder generar una boleta con los productos de su compra.

Por lo tanto, se debe crear las tablas correspondientes que soporten tal funcionalidad que son:

- Tabla producto
- Tabla cliente
- Tabla trabajador
- Tabla pedido
- Tabla producto_pedido

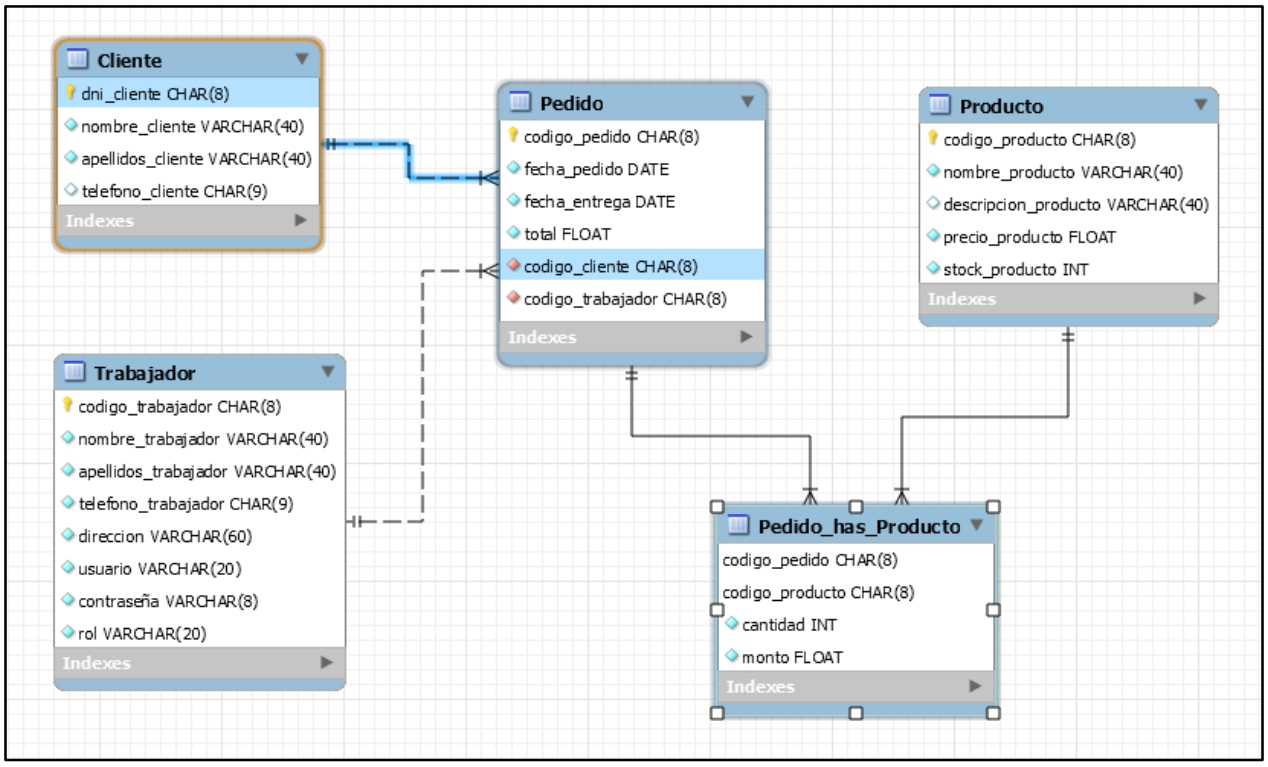
```
CREATE TABLE PRODUCTO(
    CODIGO_PRODUCTO CHAR(8) NOT NULL,
    NOMBRE_PRODUCTO VARCHAR2(40) NOT NULL,
    DESCRIPCION_PRODUCTO VARCHAR2(40) NOT NULL,
    PRECIO_PRODUCTO FLOAT NOT NULL,
    STOCK_PRODUCTO INT NOT NULL,
    PRIMARY KEY (CODIGO_PRODUCTO)
);

CREATE TABLE CLIENTE(
    DNI_CLIENTE CHAR(8) NOT NULL,
    NOMBRE_CLIENTE VARCHAR(40) NOT NULL,
    APELLIDOS_CLIENTE VARCHAR(40) NOT NULL,
    TELEFONO_CLIENTE CHAR(9) NOT NULL,
    PRIMARY KEY (DNI_CLIENTE)
);

CREATE TABLE TRABAJADOR(
    CODIGO_TRABAJADOR CHAR(8) NOT NULL,
    NOMBRE_TRABAJADOR VARCHAR(40) NOT NULL,
    APELLIDOS_TRABAJADOR VARCHAR(40) NOT NULL,
    TELEFONO CHAR(9) NOT NULL,
    DIRECCION VARCHAR2(60) NOT NULL,
    USUARIO VARCHAR2(20) NOT NULL,
    CONTRASEÑA VARCHAR2(8) NOT NULL,
    ROL VARCHAR(20) NOT NULL,
    PRIMARY KEY (CODIGO_TRABAJADOR)
);
```

```sql
CREATE TABLE PEDIDO(
    CODIGO_PEDIDO CHAR(8) NOT NULL,
    DNI_CLIENTE CHAR(8) NOT NULL,
    CODIGO_TRABAJADOR CHAR (8) NOT NULL,
    FECHA_PEDIDO DATE NOT NULL,
    FECHA_ENTREGA DATE NOT NULL,
    TOTAL FLOAT NOT NULL,
    PRIMARY KEY (CODIGO_PEDIDO),
    FOREIGN KEY (DNI_CLIENTE) REFERENCES CLIENTE(DNI_CLIENTE),
    FOREIGN KEY (CODIGO_TRABAJADOR) REFERENCES TRABAJADOR(CODIGO_TRABAJADOR)
);

CREATE TABLE PRODUCTO_PEDIDO(
    CODIGO_PEDIDO CHAR(8) NOT NULL,
    CODIGO_PRODUCTO CHAR(8) NOT NULL,
    CANTIDAD INT NOT NULL,
    MONTO DOUBLE NOT NULL,
    FOREIGN KEY (CODIGO_PEDIDO) REFERENCES PEDIDO(CODIGO_PEDIDO),
    FOREIGN KEY (CODIGO_PRODUCTO) REFERENCES PRODUCTO(CODIGO_PRODUCTO),
    CONSTRAINT PK_PRODPED PRIMARY KEY (CODIGO_PEDIDO,CODIGO_PRODUCTO)
);
```



2. Generar el script

Todos los procedimientos almacenados que se requieren para el examen final están desarrollados en los archivos SQL dentro de la carpeta "SCRIPT" que son:

- Creacion de usuario y privilegios: Creación de la tablespace junto con el usuario y su respectivo privilegio
- Creación de tablas: Que se encarga de la creación de tablas necesarias para el negocio junto con las restricciones y constraint.
- Creación de paquetes: Se generan la vista y el body del CRUD del negocio
- Creación de procedimientos almacenados: Para la consulta de resumen de datos
- Proceso simple y el complejo.

3. Script de carga de datos

```
/*CARGA DE DATOS DE LA TABLA PRODUCTOS*/

INSERT INTO PRODUCTO VALUES ('P0000001','arroz','graneado',3.50,100);
INSERT INTO PRODUCTO VALUES ('P0000002','aceite','oliva',2.50,100);
INSERT INTO PRODUCTO VALUES ('P0000003','fideo','molitalia',1.50,100);
INSERT INTO PRODUCTO VALUES ('P0000004','san mateo','agua de 3L',5.00,100);
INSERT INTO PRODUCTO VALUES ('P0000005','nick','galleta de costa',3.00,100);
INSERT INTO PRODUCTO VALUES ('P0000006','avena','3 ositos',4.00,100);
INSERT INTO PRODUCTO VALUES ('P0000007','atun','grate',3.00,50);
INSERT INTO PRODUCTO VALUES ('P0000008','papel higienico','suave 2 rollos',2.00,80);
INSERT INTO PRODUCTO VALUES ('P0000009','huevo','por kilo',2.00,100);
INSERT INTO PRODUCTO VALUES ('P0000010','gaseosa','coca cola 3 litros',7.00,100);
INSERT INTO PRODUCTO VALUES ('P0000011','leche','gloria grande',3.50,100);

/*CARGA DE DATOS DE LA TABLA TRABAJADORES*/

INSERT INTO TRABAJADOR VALUES ('T0000001','Juan','Yucra','987654321','Olivos','admi','admi','Administrador');
INSERT INTO TRABAJADOR VALUES ('T0000002','Pedro','Alvarez','998765432','Surco','empleado1','pedro','Empleado');
INSERT INTO TRABAJADOR VALUES ('T0000003','Carlos','Medina','987654333','Lurin','empleado2','carlos','Empleado');

/*CARGA DE DATOS DE LA TABLA DE CLIENTES*/

INSERT INTO CLIENTE VALUES ('33333333','Arnulfo','Perez','987654326');
INSERT INTO CLIENTE VALUES ('44444444','Pepe','Perales','987654322');
INSERT INTO CLIENTE VALUES ('55555555','Coco','Garmendia','987654323');
INSERT INTO CLIENTE VALUES ('66666666','Luis','Soto','987654324');
INSERT INTO CLIENTE VALUES ('77777777','Carlos','Diaz','987654325');
```

```sql
/*CARGA DE DATOS DE LA TABLA PEDIDOS*/

INSERT INTO PEDIDO VALUES ('B0000004','33333333','T0000003','01-01-2019','01-01-2019',13.00);
INSERT INTO PEDIDO VALUES ('B0000005','44444444','T0000003','01-01-2019','01-01-2019',10.00);
INSERT INTO PEDIDO VALUES ('B0000006','55555555','T0000003','01-01-2019','01-01-2019',7.00);
INSERT INTO PEDIDO VALUES ('B0000007','66666666','T0000002','02-01-2019','02-01-2019',17.00);
INSERT INTO PEDIDO VALUES ('B0000008','33333333','T0000002','02-01-2019','02-01-2019',3.00);

/*CARGA DE DATOS DE LA TABLA PRODUCTO_PEDIDO*/

INSERT INTO PRODUCTO_PEDIDO VALUES ('B0000004','P0000004',2, 10.00);
INSERT INTO PRODUCTO_PEDIDO VALUES ('B0000004','P0000005',1, 3.00);

INSERT INTO PRODUCTO_PEDIDO VALUES ('B0000005','P0000006',1, 4.00);
INSERT INTO PRODUCTO_PEDIDO VALUES ('B0000005','P0000001',1, 3.50);
INSERT INTO PRODUCTO_PEDIDO VALUES ('B0000005','P0000002',1, 2.50);

INSERT INTO PRODUCTO_PEDIDO VALUES ('B0000006','P0000010',1, 7.00);

INSERT INTO PRODUCTO_PEDIDO VALUES ('B0000007','P0000011',3, 7.00);
INSERT INTO PRODUCTO_PEDIDO VALUES ('B0000007','P0000008',5, 10.00);

INSERT INTO PRODUCTO_PEDIDO VALUES ('B0000008','P0000003',2, 3.00);
```

4. Programar un paquete a un CRUD

```sql
        /*---------- PAKETE DEL CRUD DEL PRODUCTO----------*/

CREATE OR REPLACE PACKAGE Producto_CRUD AS

    FUNCTION VERIFICAR_EXISTENCIA_PRODUCTO(
        v_codigoProducto IN PRODUCTO.CODIGO_PRODUCTO%TYPE) RETURN VARCHAR2;

    PROCEDURE MOSTRAR_PRODUCTO;

    PROCEDURE INSERTAR_PRODUCTO(
        v_codprod IN PRODUCTO.CODIGO_PRODUCTO%TYPE,
        v_nomprod IN PRODUCTO.NOMBRE_PRODUCTO%TYPE,
        v_desprod IN PRODUCTO.DESCRIPCION_PRODUCTO%TYPE,
        v_precioprod IN PRODUCTO.PRECIO_PRODUCTO%TYPE,
        v_stockprod IN PRODUCTO.STOCK_PRODUCTO%TYPE);

    PROCEDURE MODIFICAR_PRODUCTO(
        v_codigoProducto IN PRODUCTO.CODIGO_PRODUCTO%TYPE,
        v_nombreProducto IN PRODUCTO.NOMBRE_PRODUCTO%TYPE,
        v_descripcionProducto IN PRODUCTO.DESCRIPCION_PRODUCTO%TYPE,
        v_precioProducto IN PRODUCTO.PRECIO_PRODUCTO%TYPE,
        v_stockProducto IN PRODUCTO.STOCK_PRODUCTO%TYPE);

    PROCEDURE ELIMINAR_PRODUCTO(
        v_codigoProducto IN PRODUCTO.CODIGO_PRODUCTO%TYPE);

END Producto_CRUD;
```

En la imagen de anterior se muestra la vista del paquete del CRUD del producto, **la parte del body esta desarrollado en un archivo SQL dentro de la carpeta "SCRIPT".**

5. Programar un proceso simple

En este punto tenemos 4 procesos simples que son los siguientes:

- Obtener los datos de un trabajador

```
CREATE OR REPLACE PROCEDURE EEC_PRSIMPLE
(p_id IN TRABAJADOR.CODIGO_TRABAJADOR%TYPE,
 p_codigo OUT TRABAJADOR.CODIGO_TRABAJADOR%TYPE,
 p_nombre OUT TRABAJADOR.NOMBRE_TRABAJADOR%TYPE,
 p_apellido OUT TRABAJADOR.APELLIDOS_TRABAJADOR%TYPE,
 p_telefono OUT TRABAJADOR.TELEFONO%TYPE,
 p_direccion OUT  TRABAJADOR.DIRECCION%TYPE,
 p_rol OUT TRABAJADOR.ROL%TYPE
)
AS
BEGIN
    SELECT CODIGO_TRABAJADOR, NOMBRE_TRABAJADOR, APELLIDOS_TRABAJADOR,
           TELEFONO, DIRECCION, ROL
    INTO p_codigo, p_nombre,p_apellido,p_telefono,p_direccion,p_rol
    FROM TRABAJADOR
    WHERE CODIGO_TRABAJADOR = p_id;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        raise_application_error(-20001,'trabajador no esta registrado');
END;
```

- Modificar el precio de un producto

```
CREATE OR REPLACE PROCEDURE EEC_PRSIMPLE_PRODUCTO_4
(p_id_producto IN PRODUCTO.CODIGO_PRODUCTO%TYPE,
 p_cod_admin IN  TRABAJADOR.CODIGO_TRABAJADOR%TYPE,
 p_precio_nuevo IN  PRODUCTO.PRECIO_PRODUCTO%TYPE
)
AS
    v_rol TRABAJADOR.ROL%TYPE;
BEGIN
    SELECT ROL INTO v_rol
    FROM TRABAJADOR
    WHERE CODIGO_TRABAJADOR = p_cod_admin;
    IF (v_rol = 'Administrador') THEN
        UPDATE PRODUCTO
        SET PRECIO_PRODUCTO = p_precio_nuevo
        WHERE CODIGO_PRODUCTO = p_id_producto;
        COMMIT;
    ELSE
        raise_application_error(-20004, 'no eres administrador');
    END IF;
EXCEPTION
    WHEN OTHERS THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE('codigo: ' || SQLCODE);
        DBMS_OUTPUT.PUT_LINE('mensaje: ' || SQLERRM);
        raise_application_error(SQLCODE, SQLERRM);
END;
```

- Obtener los datos de un producto
- Modificar el precio de un producto

Los 4 procesos simples se encuentran desarrollados en la carpeta "SCRIPT".

6. Consulta de resumen de datos (2 consultas)

Se ha creado un **paquete** el cual nos **muestra** todos los **productos de los pedidos registrados** en la base de datos **que ha hecho un cliente**

```
CREATE OR REPLACE PACKAGE GENERAR_INFORME_PEDIDOS AS
    PROCEDURE PEDIDO_DEL_CLIENTE(
        v_codigoCliente IN CLIENTE.DNI_CLIENTE%TYPE
    );

    PROCEDURE MOSTRAR_PRODUCTOS_CLIENTE(
        v_codigoPedido PEDIDO.CODIGO_PEDIDO%TYPE,
        v_montoTotal PEDIDO.TOTAL%TYPE,
        v_fechaEntrega PEDIDO.FECHA_ENTREGA%TYPE,
        v_nombreTrabajador TRABAJADOR.NOMBRE_TRABAJADOR%TYPE
    );
END GENERAR_INFORME_PEDIDOS;
```

```
CREATE OR REPLACE PACKAGE BODY GENERAR_INFORME_PEDIDOS AS

    PROCEDURE PEDIDO_DEL_CLIENTE(
        v_codigoCliente IN CLIENTE.DNI_CLIENTE%TYPE
    )
    IS
        CURSOR pedido_cursor IS SELECT p.CODIGO_PEDIDO,p.TOTAL,p.FECHA_ENTREGA,t.NOMBRE_TRABAJADOR
        FROM PEDIDO p JOIN TRABAJADOR t ON p.CODIGO_TRABAJADOR = t.CODIGO_TRABAJADOR
        WHERE DNI_CLIENTE=v_codigoCliente;

        v_codigoPedido PEDIDO.CODIGO_PEDIDO%TYPE;
        v_montoTotal PEDIDO.TOTAL%TYPE;
        v_fechaEntrega PEDIDO.FECHA_ENTREGA%TYPE;
        v_nombreTrabajador TRABAJADOR.NOMBRE_TRABAJADOR%TYPE;
    BEGIN
        OPEN pedido_cursor;
            IF pedido_cursor%NOTFOUND THEN
                DBMS_OUTPUT.PUT_LINE('EL CLIENTE NO TIENE PEDIDO.');
                CLOSE pedido_cursor;
                RETURN;
            END IF;
            LOOP
                FETCH pedido_cursor INTO v_codigoPedido,v_montoTotal,v_fechaEntrega,v_nombreTrabajador;
                EXIT WHEN pedido_cursor%NOTFOUND;
                MOSTRAR_PRODUCTOS_CLIENTE(v_codigoPedido,v_montoTotal,v_fechaEntrega,v_nombreTrabajador);
            END LOOP;
        CLOSE pedido_cursor;
    END PEDIDO_DEL_CLIENTE;
```

```
    PROCEDURE MOSTRAR_PRODUCTOS_CLIENTE(
        v_codigoPedido PEDIDO.CODIGO_PEDIDO%TYPE,
        v_montoTotal PEDIDO.TOTAL%TYPE,
        v_fechaEntrega PEDIDO.FECHA_ENTREGA%TYPE,
        v_nombreTrabajador TRABAJADOR.NOMBRE_TRABAJADOR%TYPE
    )
    IS
        CURSOR product_cur IS SELECT p.NOMBRE_PRODUCTO,p.PRECIO_PRODUCTO,pp.CANTIDAD,pp.MONTO
        FROM PRODUCTO_PEDIDO pp JOIN PRODUCTO p
        ON p.CODIGO_PRODUCTO = pp.CODIGO_PRODUCTO WHERE pp.CODIGO_PEDIDO=v_codigoPedido;
    BEGIN
        DBMS_OUTPUT.PUT_LINE('******************   BOLETA  **********************');
        DBMS_OUTPUT.PUT_LINE('Codigo pedido: '||v_codigoPedido);
        DBMS_OUTPUT.PUT_LINE('Nombre del trabajador: '||INITCAP(v_nombreTrabajador));
        DBMS_OUTPUT.PUT_LINE('----------------------------------------');
        FOR producto_cursor IN product_cur LOOP
            DBMS_OUTPUT.PUT_LINE('Nombre del Producto: '||INITCAP(producto_cursor.NOMBRE_PRODUCTO)||' - '||
                                 'Precio: '||producto_cursor.PRECIO_PRODUCTO||' soles - '||
                                 'Cantidad: '||producto_cursor.CANTIDAD||' unidades - '||
                                 'PrecioxCantidad: '||producto_cursor.MONTO||' soles.');
        END LOOP;
        DBMS_OUTPUT.PUT_LINE('----------------------------------------');
        DBMS_OUTPUT.PUT_LINE('                                    MONTO TOTAL: '||v_montoTotal||' soles'||'  -  '||
                              'Fecha de entrega: '||v_fechaEntrega);
        DBMS_OUTPUT.PUT_LINE('************************************************');
    END MOSTRAR_PRODUCTOS_CLIENTE;


END GENERAR_INFORME_PEDIDOS;
```

7. Proceso complejo. Procesar varias filas

Se ha desarrollado el ingreso de varios productos a su respectiva tabla usando el paquete "PKG_UTIL" desarrollado en clase.

```
CREATE OR REPLACE PACKAGE BODY USUARIOYUCRA.PKG_UTIL AS
    FUNCTION SPLIT(P_DATA VARCHAR2, P_DELIMITADOR VARCHAR2 )
    RETURN T_ARRAY_STRING
    IS
        I         NUMBER    := 0;
        POS       NUMBER    := 0;
        V_DATA    CLOB      := P_DATA;
        STRINGS   T_ARRAY_STRING;
    BEGIN
      V_DATA := TRIM( V_DATA );
      POS := INSTR( V_DATA, P_DELIMITADOR, 1, 1 );
      WHILE ( POS != 0) LOOP
          I := I + 1;
          STRINGS(i) := SUBSTR( V_DATA, 1, POS - 1 );
          V_DATA :=  SUBSTR( V_DATA, POS + 1, LENGTH(V_DATA) );
          pos := instr(V_DATA, P_DELIMITADOR, 1, 1);
          IF POS = 0 THEN
              STRINGS( I + 1 ) := V_DATA;
          END IF;
      END LOOP;
      IF I = 0 AND LENGTH( V_DATA ) > 0 THEN
        STRINGS( I + 1 ) := V_DATA;
      END IF;
      RETURN strings;
    END SPLIT;

END PKG_UTIL;
```

```
CREATE OR REPLACE PROCEDURE PR_INSERTA_PRODUCTO
( P_DATOS IN VARCHAR2 )
AS
  V_FILAS    PKG_UTIL.T_ARRAY_STRING;
  V_CAMPOS   PKG_UTIL.T_ARRAY_STRING;

BEGIN
  V_FILAS := PKG_UTIL.SPLIT(P_DATOS,'¬');
  FOR I in 1 .. V_FILAS.COUNT LOOP

    V_CAMPOS := PKG_UTIL.SPLIT(V_FILAS(I),'|');

    INSERT INTO PRODUCTO(CODIGO_PRODUCTO,NOMBRE_PRODUCTO,DESCRIPCION_PRODUCTO,PRECIO_PRODUCTO,STOCK_PRODUCTO)
    VALUES(V_CAMPOS(1), V_CAMPOS(2), V_CAMPOS(3), CAST(to_number(V_CAMPOS(4)) AS FLOAT), CAST(to_number(V_CAMPOS(5)) AS NUMBER(38)) );
  END LOOP;
  COMMIT;
  DBMS_OUTPUT.PUT_LINE('PROCESO OK');
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE(SQLERRM);
    ROLLBACK;
END;
```

```
DECLARE
  V_DATOS VARCHAR2(6000);
BEGIN
  V_DATOS := 'P0000031|crema dental|dento triple accion|10.5|60¬
             P0000032|bloqueador solar| spf 50 neutrogena|54.00|200¬
             P0000033|jabon tripack |avena nivea|8.10|80';
  PR_INSERTA_PRODUCTO (V_DATOS);
END;
commit;
```