.1. Función que compruebe si existe un pedido con el número que se le pase. Devolverá verdadero o falso.

Tengo mi función:

```
CREATE OR REPLACE FUNCTION comprobarPedido(v_num_ped Pedidos.NUM
%TYPE)
RETURN boolean
IS
l_count number;
BEGIN
SELECT count(*)
INTO l_count
FROM Pedidos
WHERE NUM = v_num_ped;
IF( l_count = 0 )
THEN
RETURN false;
ELSE
RETURN true;
END IF;
END comprobarPedido;
```

o la tuya:

```
--Función que devuelve si existe el pedido

CREATE OR REPLACE FUNCTION EXISTE_PEDIDO(P_NUMERO IN PEDIDOS.NUM
%TYPE)

 RETURN BOOLEAN

 AS

 BOOL_EXISTE BOOLEAN;

 NUMERO_PEDIDO NUMBER;

BEGIN

 SELECT COUNT(NUM) INTO NUMERO_PEDIDO FROM PEDIDOS WHERE NUM =
P_NUMERO;

 IF NUMERO_PEDIDO = 0 THEN BOOL_EXISTE := FALSE;

 ELSE BOOL_EXISTE := TRUE;

 END IF;

 RETURN BOOL_EXISTE;
```

```
EXCEPTION

 WHEN OTHERS THEN

 DBMS_OUTPUT.PUT_LINE('Se ha producido un error');

END EXISTE_PEDIDO;
```

2.2. Función que devuelve todos los datos de un pedido a partir de su número (toda la fila de la tabla pedidos)

```
CREATE OR REPLACE FUNCTION DATOS_PEDIDOS(PARAM1 IN NUMBER)

RETURN VARCHAR2 IS DATOS VARCHAR2(1000);

BEGIN

        SELECT to_char('Numero_pedido: ')||' '||NUM||

        ' - Fecha: '||fecha||

        ' - Gastos envio: '||nvl(gastos_envio,0)||

        ' - Fecha prevista: '||fecha_prevista||

        ' - Total: '||total||

        ' - Cliente: '||cliente

        INTO DATOS

        FROM PEDIDOS

        WHERE NUM=PARAM1;

        RETURN DATOS;

END DATOS_PEDIDOS;
```

3. Procedimiento que devuelve los datos de un cliente a partir del código de cliente

```
create or replace PROCEDURE prClientes (v_cod_cli Clientes.CODIGO%TYPE)
AS
cli_tab Clientes%ROWTYPE;
BEGIN
SELECT *
INTO cli_tab
FROM Clientes
WHERE CODIGO = v_cod_cli;
```

```
EXCEPTION
when no_data_found then
dbms_output.put_line('no encontrado');
END prClientes;
```

4. Procedimiento que muestra un listado con las líneas de un pedido (a partir de su número), de la siguiente manera:
Nº Línea NombreProducto Precio Cantidad Importe

```
CREATE OR REPLACE PROCEDURE GET_DATOS_CLIENTE(P_CODIGO IN CLIENTES.CODIGO%TYPE) AS

    CNOMBRE CLIENTES.NOMBRE%TYPE;

    CAPELLIDOS CLIENTES.APELLIDOS%TYPE;

    CEDAD CLIENTES.EDAD%TYPE;

BEGIN

    SELECT C.NOMBRE, C.APELLIDOS, C.EDAD INTO CNOMBRE, CAPELLIDOS, CEDAD FROM CLIENTES C WHERE P_CODIGO = CODIGO;

    DBMS_OUTPUT.PUT_LINE('DATOS DEL CLIENTE: ' || CNOMBRE || ' ' || CAPELLIDOS || ' ' || CEDAD);

EXCEPTION

    WHEN NO_DATA_FOUND THEN

    DBMS_OUTPUT.PUT_LINE('Error: El cliente con el número '|| P_CODIGO || ' no existe');

END GET_DATOS_CLIENTE;
```

5. Procedimiento o bloque anónimo que a partir de un número de pedido, si existe, nos muestre todos los datos del pedido, del cliente y el listado de todas las líneas que tiene, utilizando los subprogramas anteriores.

```
declare
v_pedido Pedidos%rowtype;
begin
select * into v_pedido from Pedidos where NUM = '1';
end;
```

6. Deben tratarse las excepciones oportunas en cada uno de los subprogramas (si no existe el pedido, o no tiene líneas,….).

del ejerc 3 tenemos:

```
create or replace PROCEDURE prClientes (v_cod_cli Clientes.CODIGO%TYPE)
AS
cli_tab Clientes%ROWTYPE;
BEGIN
SELECT *
INTO cli_tab
FROM Clientes
WHERE CODIGO = v_cod_cli;
EXCEPTION
when no_data_found then
dbms_output.put_line('no encontrado');
END prClientes;
```

del ejerc 4 tenemos:

```
CREATE OR REPLACE PROCEDURE GET_DATOS_CLIENTE(P_CODIGO IN CLIENTES.CODIGO%TYPE) AS

    CNOMBRE CLIENTES.NOMBRE%TYPE;

    CAPELLIDOS CLIENTES.APELLIDOS%TYPE;

    CEDAD CLIENTES.EDAD%TYPE;

BEGIN

    SELECT C.NOMBRE, C.APELLIDOS, C.EDAD INTO CNOMBRE, CAPELLIDOS, CEDAD FROM CLIENTES C WHERE P_CODIGO = CODIGO;

    DBMS_OUTPUT.PUT_LINE('DATOS DEL CLIENTE: ' || CNOMBRE || ' ' || CAPELLIDOS || ' ' || CEDAD);

EXCEPTION

    WHEN NO_DATA_FOUND THEN

    DBMS_OUTPUT.PUT_LINE('Error: El cliente con el número '|| P_CODIGO || ' no existe');

END GET_DATOS_CLIENTE;
```

ACTIVIDAD 2.

1. Cada vez que se vaya a insertar o modificar una línea de un pedido debe de actualizarse correctamente el importe de la misma (cantidad X precio del producto

```
CREATE OR REPLACE TRIGGER biu_punto1
before INSERT OR UPDATE ON LINEAS
FOR EACH ROW
DECLARE
N_PRECIO NUMBER(7,2);
begin
SELECT PRECIO INTO N_PRECIO FROM PRODUCTOS WHERE
CODIGO=:new.PRODUCTO;
:new.importe := :new.cantidad * N_PRECIO;
end;
```