```
-- CHULETA PLS/SQL

-- FUNCIONES
	create or replace function nombre_funcion (num1 in number, num2 in number)return
	number
	is
		total number;
	begin
		total:= num1 + num2;
		return total;
	end;
	/
-- COMPROVAR SI FUNCIONA

	declare
		resultado number;
	begin
		resultado := nombre_funcion (5,10);
		DBMS_OUTPUT.PUT_LINE('el numero es: ' || resultado);
	end;

	-- PARA SACAR EL RESTO DE UNA DIVISION: MOD(num,2)
	-- LOOPS PARA FUNCIONES (DENTRO DE BEGUIN)
	loop
	for i in 1..num loop
	resultado := resultado*1
	end loop;
	end;
	/
-------------------------------------------------------------------------------------------
-- PROCEDIMIENTOS

	create or replace procedure nombre_procedimiento(variable1 in number, variable2 in
varchar2)
	is
	variable que acomulemos un resultadp (sum, avg....);
	begin
		insert into nombre_tabla (nombre_columa1,nombre columna2) values (valor1,
valor2) where (condiciones_varias);
		update nombre_tabla set columna =nuevo valor de la casilla where condicion;
		delete from nombre_tabla where condicion;
		select nombre_tabla from tabla(inner join tabla2 on (id_tabla1= id_tabla2)) where
(condicion);
	end;
-------------------------------------------------------------------------------------------
```

```
-- CURSORES

	declare
		cursor nombre_cursor is
		select nombre_tabla from tabla(inner join tabla2 on (id_tabla1= id_tabla2)) where
(condicion);
		v_nombre_variable tabla.columna%type;
		v_nombre_variable que no viene del select :=0;
	begin
		open nombre_cursor;
			loop
			fetch nombre_cursor into v_nombre_variable;
			exit when nombre_cursor %NOTFOUND;
				aqui puedes trabajar con la variable que no viene del select;
				aquio puedes ejecutar otra sentencia sql: update tabla set columa =
columna*5;
			end loop;
		close nombre_cursor;
	end;
-------------------------------------------------------------------------------------------
-- TRIGGERS

	create or replace trigger nombre_trigger after/before insert, update,delete on nombre_tabla
	for each row
	declare
		v_nombre_variable tipo_variable(number,varchar2);
	begin
		-- update nombre_tabla set columna =  columna /*+- :new.nombre_columna (esta
viene de la columna sobre la que ejecutamos el trigger)
		where columna = :new.nombre_columna;
		-- puedes ejecutar varias sentencias metiendo el resultado de la primera en una
variable creada;
		-- tambien se pueden hacer bucles for, if y while con los resultados de las variables
almacenadas;
	end;
-------------------------------------------------------------------------------------------
-- EJERCICIO 5 TRIGGERS

	CREATE OR REPLACE TRIGGER t_actualizar_stock AFTER insert or update ON
detalles_pedido
	for each row
	declare
		cursor c_detalle_pedido is
		select * from detalles_pedido where codigo_pedido:=new.codigo_pedido;
	begin
		for v_detalle in c_detalle_pedido loop
			update productos
			set cantidad_en_stock = cantidad_en_stock - v_detalle.cantidad
			where codigo_producto = v_detalle.codigo_producto;
		end loop;
	end;
-------------------------------------------------------------------------------------------
```

```
-- EXCEPCIONES

        DECLARE
        IS
        BEGIN
        EXCEPTION
        WHEN  NO_DATA_FOUND THEN
                dbms_output.put_line('Error no hay datos');

        WHEN too_many_rows THEN
                dbms_output.put_line('Error mas de una columna');

        WHEN others THEN
                dbms_output.put_line('Errorotros');
        end;
-------------------------------------------------------------------------------------------------------------
-- BUCLES
-- FOR COMPLEJO
        for i in nombre_cursor loop
                la sentencia que sea
        end loop;
```