```
-- LIMPIAMOS TODO PARA QUE NO DE FALLOS AL CARGAR EL ARCHIVO
        DROP TABLE TablaResponsables; DROP TABLE TablaComerciales;
        DROP TYPE Personal FORCE;        DROP TYPE Responsable FORCE;
        DROP TYPE Zonas FORCE;           DROP TYPE Comercial FORCE;
        DROP TYPE ListaZonas FORCE;
-- 1.1 Crea el tipo de objetos "Personal"
CREATE OR REPLACE TYPE Personal AS OBJECT (
        codigo INTEGER,dni VARCHAR2(10), nombre VARCHAR2(30), apellidos  VARCHAR2(30),
sexo VARCHAR2(1),fecha_nac DATE )  NOT FINAL; /
--1.2 Crea el tipo de objeto "Responsable" como tipo heredado de "Personal
CREATE OR REPLACE TYPE Responsable UNDER Personal (
        tipo CHAR,
        antiguedad INTEGER,
        MEMBER FUNCTION getNombreCompleto RETURN VARCHAR2,
        CONSTRUCTOR FUNCTION Responsable (
                codigo INTEGER, nombre VARCHAR2, primerApellido VARCHAR2,
                segundoApellido VARCHAR2,        tipo CHAR
        ) RETURN SELF AS RESULT); /
--1.3. Crea el tipo Zonas con los atributos codigo INTEGER, nombre VARCHAR2(20), refRespon REF
Responsable, codigoPostal CHAR(5),
CREATE OR REPLACE TYPE Zonas AS OBJECT(
        codigo INTEGER, nombre VARCHAR2(20), refResponsable REF Responsable,
        codigoPostal CHAR(5)      ); /
-- Crea, como tipo heredado de Personal, el tipo de objeto "Comercial"
CREATE OR REPLACE TYPE Comercial UNDER Personal (
        zonaComercial Zonas
        ); /
-- 2. Crea un método constructor para tipo de objetos "Responsable"
CREATE OR REPLACE TYPE BODY Responsable AS
CONSTRUCTOR FUNCTION Responsable(
        codigo INTEGER, nombre VARCHAR2,
        primerApellido VARCHAR2, segundoApellido VARCHAR2,
        tipo CHAR) RETURN SELF AS RESULT IS
        BEGIN
                -- Asignar al atributo apellidos los datos de primer apellido y          segundo
apellido que se han pasado como parámetros, uniéndolos          con un espacio entre ellos
                SELF.codigo := codigo; SELF.nombre := nombre;
                SELF.apellidos := primerApellido || ' ' || segundoApellido;
                SELF.tipo := tipo;   RETURN;
        END;
-- 3. Crea un método getNombreCompleto para el tipo de objetos Responsable que permita obtener su
nombre completo con el formato apellidos nombre.
        MEMBER FUNCTION getNombreCompleto RETURN VARCHAR2 IS
        BEGIN
                RETURN self.apellidos || ' ' || self.nombre;
        END getNombreCompleto;
END; /
-- 4. Crea una tabla TablaResponsables de objetos Responsable. Inserta en dicha tabla dos objetos
Responsable.
CREATE TABLE TablaResponsables OF Responsable;
INSERT INTO TablaResponsables VALUES (
        Responsable(5, '51083099F', 'ELENA', 'POSTA LLANOS', 'F', TO_DATE('31/03/1975',
'DD/MM/YYYY'), 'N', 4) );
DECLARE
        responsable2 Responsable;
BEGIN
        responsable2 := Responsable(6, 'JAVIER', 'JARAMILLO', 'HERNANDEZ', 'C');
        INSERT INTO TablaResponsables VALUES (responsable2);
END; /

-- 5. Crea una colección VARRAY llamada ListaZonas en la que se puedan almacenar hasta 10 objetos Zonas.

CREATE TYPE ListaZonas AS VARRAY(10) OF Zonas; /
DECLARE
        listaZonas1 ListaZonas; zona1 Zonas; zona2 Zonas; responsable1 REF        Responsable;
        responsable2 REF Responsable;
BEGIN
        listaZonas1 := ListaZonas();
        -- Seleccionamos los datos referenciados
        SELECT REF(TR) INTO responsable1 FROM TablaResponsables TR WHERE TR.codigo = 6;
        SELECT REF(TR) INTO responsable2 FROM TablaResponsables TR WHERE TR.dni          =
        '51083099F';
        -- Creamos nuestros objetos Zonas
        zona1 := Zonas(1, 'zona1', responsable1, '06834');
        zona2 := Zonas(2, 'zona2', responsable2, '28003');
        -- Guarda en una instancia listaZonas1 de dicha lista, dos Zonas
        listaZonas1 := ListaZonas(zona1, zona2);
END; /
-- 6. Crea una tabla TablaComerciales de objetos Comercial
CREATE TABLE TablaComerciales OF Comercial;
DECLARE
   zona1 Zonas; zona2 Zonas;
BEGIN
   INSERT INTO TablaComerciales VALUES (
     Comercial(100, '23401092Z', 'MARCOS', 'SUAREZ LOPEZ', 'M',  TO_DATE('30/03/1990',
        'DD/MM/YYYY'), zona1)     );
INSERT INTO TablaComerciales VALUES (
      Comercial(102, '6932288V', 'ANASTASIA', 'GOMES PEREZ', 'F', TO_DATE('28/11/1984',
        'DD/MM/YYYY'), zona2)     ); END; /
-- 7. Obtener, de la tabla TablaComerciales, el Comercial que tiene el código 100,asignándoselo a una variable
unComercial
 SELECT VALUE(a) INTO unComercial FROM TablaComerciales a WHERE codigo = 100;
-- 8. Modifica el código del Comercial guardado en variable unComercial = 10
 unComercial.codigo := 101; unComercial.zonaComercial := zona2;
 INSERT INTO TablaComerciales VALUES (unComercial);  END; /
-- 9. Crea un método MAP ordenarZonas para el tipo Zonas.
 MAP MEMBER FUNCTION ordenarZonas RETURN VARCHAR2 IS
        unResponsable Responsable;
        BEGIN
                SELECT DEREF(refRespon) INTO unResponsable FROM Dual;
                RETURN (unResponsable.getNombreCompleto());
        END;   END; /
-- 10. Realiza una consulta de la tabla TablaComerciales ordenada por zonaComercial para comprobar el
funcionamiento del método MAP. */
 SELECT * FROM TablaComerciales ORDER BY zonaComercial;
```