Mario Rubio Avila BD_Tarea 7.1

Se adjunta el fichero sql con todas las sentencias ejecutadas.

1. Crea el tipo de objetos "MiembroEscolar" con los siguientes atributos:

```
--1.0 - MiembroEscolar
CREATE OR REPLACE TYPE MiembroEscolar AS OBJECT (
    codigo INTEGER,
    dni VARCHAR2(10),
    nombre VARCHAR2(20),
    apellidos VARCHAR2(30),
    sexo VARCHAR2(1),
    fecha_nac DATE
) NOT FINAL;
/
```

```
Type MIEMBROESCOLAR compilado
```

2. Crea, como tipo heredado de "MiembroEscolar", el tipo de objeto "Profesor" con los siguientes atributos y métodos:

```
--2.0 - Creamos Profesor tipo profesor simple sin nada
CREATE OR REPLACE TYPE Profesor UNDER MiembroEscolar (
especialidad VARCHAR2(20),
antiguedad INTEGER
);
/
```


```
Type PROFESOR compilado
```

Modificamos el tipo profesor añadiendo su constructor y su función para devolver

(apellidos + " " + nombre)

```
--2.1 -Creamos  tipo profesor con el contructor
CREATE OR REPLACE TYPE Profesor UNDER MiembroEscolar (
    especialidad VARCHAR2(20),
    antiguedad INTEGER,
    MEMBER FUNCTION getNombreCompleto RETURN VARCHAR2,
    CONSTRUCTOR FUNCTION Profesor(self in out nocopy Profesor, codigo INTEGER,
                                                               nombre VARCHAR2,
                                                               primerApellido VARCHAR2,
                                                               segundoApellido VARCHAR2,
                                                               especialidad VARCHAR2)
    RETURN SELF AS RESULT
);
/
```


```
Type PROFESOR compilado
```

Ahora creamos el cuerpo de la clase

```
--2.2 -El cuerpo de constructor
CREATE OR REPLACE TYPE BODY Profesor AS
    CONSTRUCTOR FUNCTION Profesor(codigo INTEGER, nombre VARCHAR2, primerApellido VARCHAR2, segundoApellido VARCHAR2, especialidad VARCHAR2)
    RETURN SELF AS RESULT IS
    BEGIN
        SELF.codigo := codigo;
        SELF.nombre := nombre;
        SELF.apellidos := (primerApellido||' '||segundoApellido);
        SELF.especialidad := especialidad;
        RETURN;
    END Profesor;
--2.4 Funcion para obtener nombre completo 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;
/
```

```
Salida de Script
Tarea terminada en 0,215 segundos
Type Body PROFESOR compilado
```

3. Crea el tipo de objeto "Cursos" con los siguientes atributos y métodos:

```
--3.0 -Crea el tipo de objeto "Cursos" con los atributos y métodos
CREATE OR REPLACE TYPE Cursos AS OBJECT (
    codigo INTEGER,
    nombre VARCHAR2(20),
    refProfe REF Profesor,
    max_Alumn INTEGER,
    fecha_Inic DATE,
    fecha_Fin DATE,
    num_Horas INTEGER,
    MAP MEMBER FUNCTION ordenarCursos RETURN VARCHAR2 --3.1 Ordenar map
);
/
```

```
Salida de Script
Tarea terminada en 0,527 segundos
Type CURSOS compilado
```

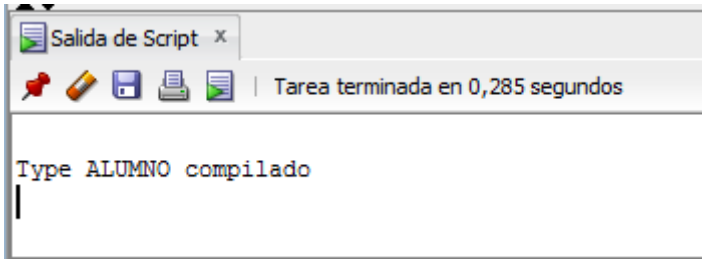
Método MAP "ordenarCursos" para el tipo "Cursos". Este método debe retornar el nombre completo del profesor al que hace referencia cada curso. Para obtener el nombre debes utilizar el método getNombreCompleto que se ha creado anteriormente.

```
--3.1
CREATE OR REPLACE TYPE BODY Cursos AS
    MAP MEMBER FUNCTION ordenarCursos RETURN VARCHAR2
    IS
     mapProfesor Profesor;
    BEGIN
        SELECT DEREF(refProfe) INTO mapProfesor FROM Dual;
        RETURN (mapProfesor.getNombreCompleto);
    END ordenarCursos;
END;
/
```


```
Type Body CURSOS compilado
```

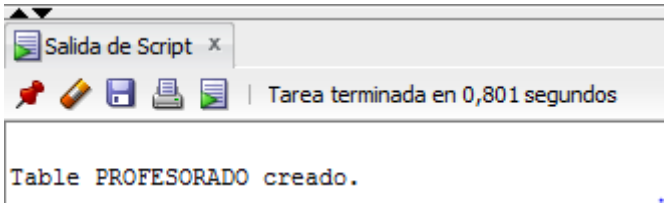
4. Crea, como tipo heredado de "MiembroEscolar", el tipo de objeto "Alumno" con los siguientes atributos:

```
--4.0
CREATE OR REPLACE TYPE Alumno UNDER MiembroEscolar (
    cursoAlumno Cursos
);
```



5. Crea un tipo de tabla "Profesorado" de objetos "Profesor".

```
/
--5.0 - Crea un tipo de tabla "Profesorado" de objetos "Profesor".
CREATE TABLE Profesorado OF Profesor;
/
```



6. Inserta en dicha tabla dos objetos "Profesor". El primero de ellos con los datos:

```
--6.0 - Inserta en dicha tabla dos objetos "Profesor". El primero de ellos con los datos:
DECLARE
    profesorAInsertar1 Profesor; --Variable para el primer profesor
    --profesorAInsertar2 Profesor; --Variable para el segundo profesor
BEGIN
    profesorAInsertar1 := NEW Profesor(2, '51083099F', 'MARIA LUISA', 'FABRE BERDUN', 'F', '31/03/1975', 'TECNOLOGIA', 4); --Iniciamos profesor 1
    --profesorAInsertar2 := NEW Profesor(3, 'JAVIER' , 'JIMENEZ', 'HERNANDO', 'LENGUA'); --Iniciamos profesor 2
    INSERT INTO Profesorado VALUES (profesorAInsertar1);
    INSERT INTO Profesorado VALUES (Profesor(3, 'JAVIER', 'JIMENEZ', 'HERNANDO', 'LENGUA'));
    --INSERT INTO Profesorado VALUES (profesorAInsertar2);
END;
/
```

```
Procedimiento PL/SQL terminado correctamente.
```

7. Crea un tipo colección VARRAY llamada "ListaCursos" en la que se puedan almacenar hasta 10 objetos "Cursos".

```
--7.0 - Crea un tipo colección VARRAY llamada "ListaCursos" en la que se puedan almacenar hasta 10 objetos "Cursos".
CREATE OR REPLACE TYPE ListaCursos IS VARRAY(10) OF Cursos;
/
```

```
Type LISTACURSOS compilado
```

8. Crea una tabla "Alumnado" de objetos "Alumno".

```
--8 -Crea una tabla "Alumnado" de objetos "Alumno".
CREATE TABLE Alumnado OF Alumno;
/
```

```
Table ALUMNADO creado.
```

9. Crea un bloque de código que haga todo lo siguiente:

```
--9.0 -Crea un bloque de código que haga todo lo siguiente:
--a) Guarda en una instancia "listaCursos1" de dicha lista, los dos cursos siguientes:
DECLARE
listaCursos1 ListaCursos;
refProfe REF Profesor;
profe Profesor;
unAlumno Alumno;
BEGIN
listaCursos1 := ListaCursos(NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);--Inicializamos todo como null
--El primer curso que debes almacenar en dicha lista debe tener los siguientes datos:
-- codigo: 1
-- nombre: Curso 1
-- refProfe: Referencia al profesor cuyo codigo es 3.
-- max_Alumn: 20
-- fecha_Inic: 1/6/2011
-- fecha_Fin: 30/6/2011
-- num_Horas: 30
SELECT REF(prof) INTO refprofe FROM profesorado prof WHERE prof.codigo = 3;
SELECT VALUE(prof) INTO profe FROM profesorado prof WHERE prof.codigo = 3;
listaCursos1(1) := NEW Cursos(1,'Curso 1',refprofe,20,'1/6/2011','30/6/2011',30);
```

```
SELECT REF(prof) INTO refprofe FROM profesorado prof WHERE prof.codigo = 3;
SELECT VALUE(prof) INTO profe FROM profesorado prof WHERE prof.codigo = 3;
listaCursos1(1) := NEW Cursos(1,'Curso 1',refprofe,20,'1/6/2011','30/6/2011',30);
-- El segundo curso que debes almacenar en dicha lista debe tener los siguientes datos:
-- codigo: 2
-- nombre: Curso 2
-- refProfe: Referencia al profesor cuyo DNI es 51083099F.
-- max_Alumn: 20
-- fecha_Inic: 1/6/2011
-- fecha_Fin: 30/6/2011
-- num_Horas: 30
SELECT REF(prof1) INTO refprofe FROM profesorado prof1 WHERE prof1.dni = '51083099F';
SELECT VALUE(prof1) INTO profe FROM profesorado prof1 WHERE prof1.dni = '51083099F';
listaCursos1(2) := NEW Cursos(2,'Curso 2',refprofe,20,'1/6/2011','30/6/2011',30);

--Vamos a imprimir el nombre del curso para probar que funciona.
DBMS_OUTPUT.put_line(listaCursos1(1).nombre);
DBMS_OUTPUT.put_line(listaCursos1(2).nombre);
```

```
-- b) Inserta en la tabla "Alumnado" las siguientes filas:
-- codigo: 100
-- dni: 76401092Z
-- nombre: MANUEL
-- apellidos: SUAREZ IBAÑEZ
-- sexo: M
-- fecha_nac: 30/6/1990
-- cursoAlumno: objeto creado anteriormente para el primer curso
-- codigo: 102
-- dni: 6915588V
-- nombre: MILAGROSA
-- apellidos: DIAZ PEREZ
-- sexo: F
-- fecha_nac: 28/10/1984
-- cursoAlumno: objeto que se encuentre en la segunda posición de "listaCursos1" (debe tomarse de la lista)
INSERT INTO Alumnado VALUES(NEW alumno(100,'76401092Z','MANUEL','SUAREZ IBAÑEZ','M','30/6/1990',listaCursos1(1)));
INSERT INTO Alumnado VALUES(NEW alumno(102,'6915588V','MILAGROSA','DIAZ PEREZ','F','28/10/1984',listaCursos1(2)));
```

```
--c) Obtener, de la tabla "Alumnado", el alumno que tiene el código 100, asignándoselo a una variable "unAlumno".
SELECT VALUE(a) INTO unAlumno FROM Alumnado a WHERE a.codigo = 100;

--d) Modifica el código del alumno guardado en esa variable "unAlumno" asignando el valor 101, y su curso debe ser el segundo que se había creado anterio
unAlumno.codigo := 101;
unAlumno.cursoAlumno := listaCursos1(2);
INSERT INTO Alumnado VALUES (unAlumno);

END;
/
```

```
Procedimiento PL/SQL terminado correctamente.
```

10. Realiza una consulta de la tabla "Alumnado" ordenada por "cursoAlumno" para comprobar el funcionamiento del método MAP.

```
--10.-Realiza una consulta de la tabla "Alumnado" ordenada por "cursoAlumno" para comprobar el funcionamiento del método MAP.
SELECT * FROM Alumnado order by cursoAlumno;
```

Salida de Script | Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 3 en 0,053 segundos

	CODIGO	DNI	NOMBRE	APELLIDOS	SEXO	FECHA_NAC	CURSOALUMNO
1	102	6915588V	MILAGROSA	DIAZ PEREZ	F	28/10/84	[SYSTEM.CURSOS]
2	101	76401092Z	MANUEL	SUAREZ IBAÑEZ	M	30/06/90	[SYSTEM.CURSOS]
3	100	76401092Z	MANUEL	SUAREZ IBAÑEZ	M	30/06/90	[SYSTEM.CURSOS]