

Bases de Datos Orientado a Objetos

Un sistema de bases de datos OO proporciona una identidad única a cada objeto independiente almacenado en la base de datos, esta identidad se implementa mediante un identificador de objeto único generado por el sistema u OID.

En las bases de datos OO, el estado (valor actual) de un objeto complejo puede construirse a partir de otros objetos (u otros valores) utilizando ciertos constructores de tipos.

Disponen de una teoría adicional (**ObjetosComplejos_TeoríaComplementaria.pdf**), como guía para representar gráficamente a estos objetos.

Una forma de representar dichos objetos es ver a cada objeto como un trío (i, c, v) , donde **i** es el **identificador de objeto (OID)** único, **c** es un **constructor de tipo** (es decir, una identificación de cómo se construye el estado del objeto) y **v** es el **estado del objeto** (o valor actual). El modelo de datos normalmente incluirá varios constructores de tipos. Los tres constructores más básicos son **atom**, **tuple** y **set**.

El constructor atom se utiliza para representar todos los valores atómicos básicos, como enteros, números reales, cadenas de caracteres, booleanos y cualquier tipo de datos básico que el sistema soporte directamente. El estado **v** de un objeto (i, c, v) se interpreta basándose en el constructor **c**.

Si **c** = atom, el estado (valor) **v** es un valor atómico del dominio de valores básicos soportado por el sistema.

Si **c** = set, el estado **v** es un conjunto de identificadores de objetos $\{i_1, i_2, \dots, i_n\}$, que son los OIDs de un conjunto de objetos que normalmente son de algún tipo.

Si **c** = tuple, el estado **v** es una tupla de la forma $\langle a_1:i_1, a_2:i_2, \dots, a_n:i_n \rangle$, donde cada a_j es un nombre de atributo y cada i_j es un OID.

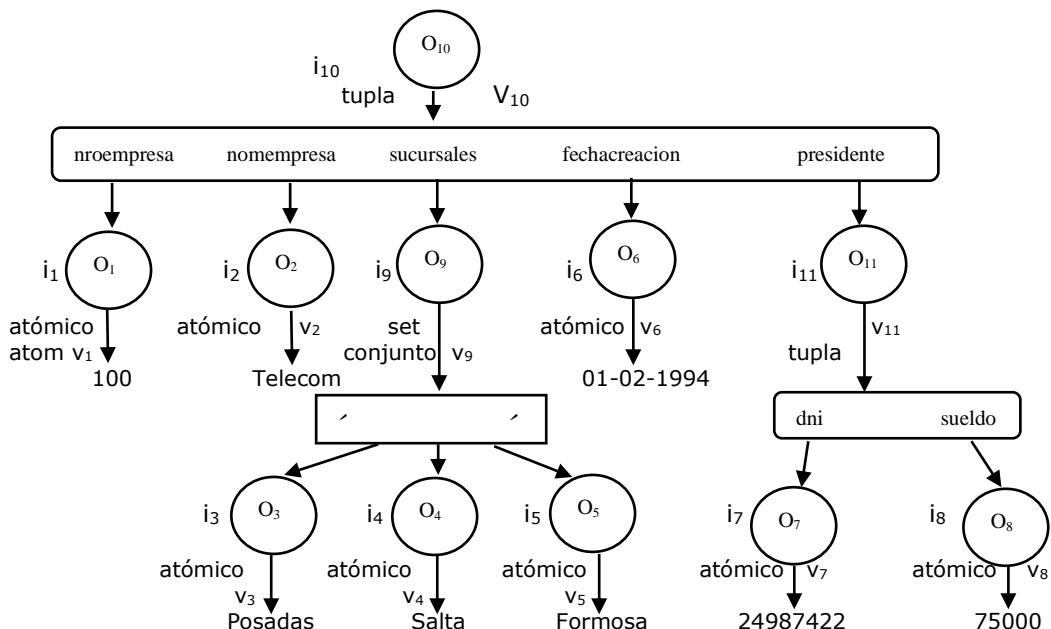
- 1)** Considerar la numeración dada para cada dato, para identificar a los objetos (o_i, i_i) :

a)

- O₁**= $(i_1, \text{atom}, 100)$
- O₃**= $(i_3, \text{atom}, \text{'Posadas'})$
- O₅**= $(i_5, \text{atom}, \text{'Formosa'})$
- O₇**= $(i_7, \text{atom}, 24987422)$
- O₉**= $(i_9, \text{set}, \{i_3, i_4, i_5\})$
- O₁₀**= $(i_{10}, \text{tuple}, \langle \text{nroempresa}:i_1, \text{nomempresa}:i_2, \text{sucursales}:i_9, \text{fechacreacion}:i_6, \text{presidente}:i_{11} \rangle)$
- O₁₁**= $(i_{11}, \text{tuple}, \langle \text{dni}:i_7, \text{sueldo}:i_8 \rangle)$

- O₂**= $(i_2, \text{atom}, \text{'Telecom'})$
- O₄**= $(i_4, \text{atom}, \text{'Salta'})$
- O₆**= $(i_6, \text{atom}, \text{'01-02-1994'})$
- O₈**= $(i_8, \text{atom}, 75000)$

- b)** Representar el objeto complejo con los siguientes símbolos:

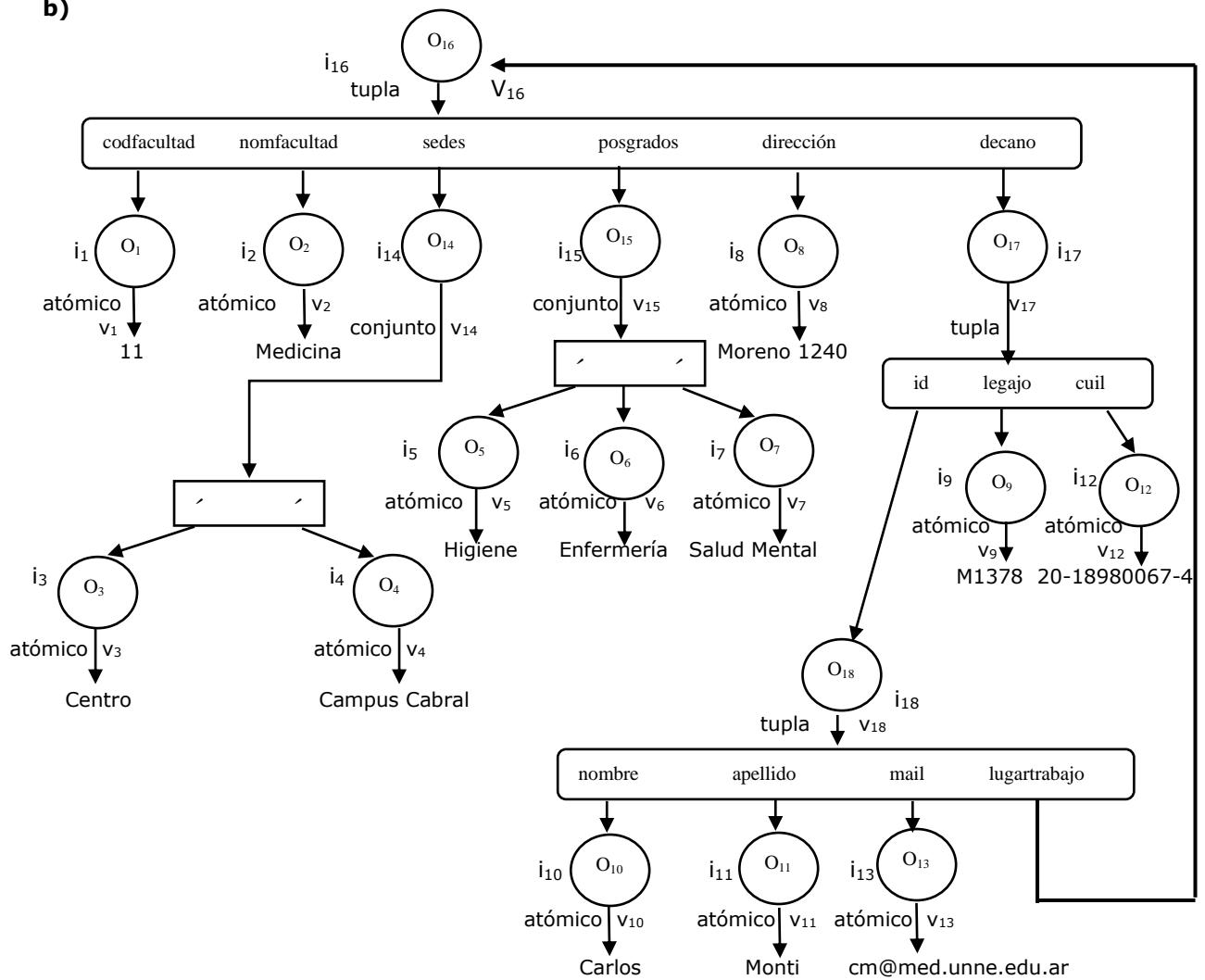


2)

a)

- $O_1 = (i_1, \text{atom}, 11)$
- $O_2 = (i_2, \text{atom}, \text{'Medicina'})$
- $O_3 = (i_3, \text{atom}, \text{'Centro'})$
- $O_4 = (i_4, \text{atom}, \text{'Campus Cabral'})$
- $O_5 = (i_5, \text{atom}, \text{'Higiene'})$
- $O_6 = (i_6, \text{atom}, \text{'Enfermería'})$
- $O_7 = (i_7, \text{atom}, \text{'Salud Mental'})$
- $O_8 = (i_8, \text{atom}, \text{'Moreno 1240'})$
- $O_9 = (i_9, \text{atom}, \text{'M1378'})$
- $O_{10} = (i_{10}, \text{atom}, \text{'Carlos'})$
- $O_{11} = (i_{11}, \text{atom}, \text{'Monti'})$
- $O_{12} = (i_{12}, \text{atom}, \text{'20-18980067-4'})$
- $O_{13} = (i_{13}, \text{atom}, \text{'cm@med.unne.edu.ar'})$
- $O_{14} = (i_{14}, \text{set}, \{i_3, i_4\})$
- $O_{15} = (i_{15}, \text{set}, \{i_5, i_6, i_7\})$
- $O_{16} = (i_{16}, \text{tuple}, \langle \text{codfacultad}:i_1, \text{nomfacultad}:i_2, \text{sedes}:i_{14}, \text{posgrados}:i_{15}, \text{dirección}:i_8, \text{decano}:i_{17} \rangle)$
- $O_{17} = (i_{17}, \text{tuple}, \langle \text{id}:i_{18}, \text{legajo}:i_9, \text{cuil}:i_{12} \rangle)$
- $O_{18} = (i_{18}, \text{tuple}, \langle \text{nombre}:i_{10}, \text{apellido}:i_{11}, \text{mail}:i_{13}, \text{lugartrabajo}:i_6 \rangle)$

b)

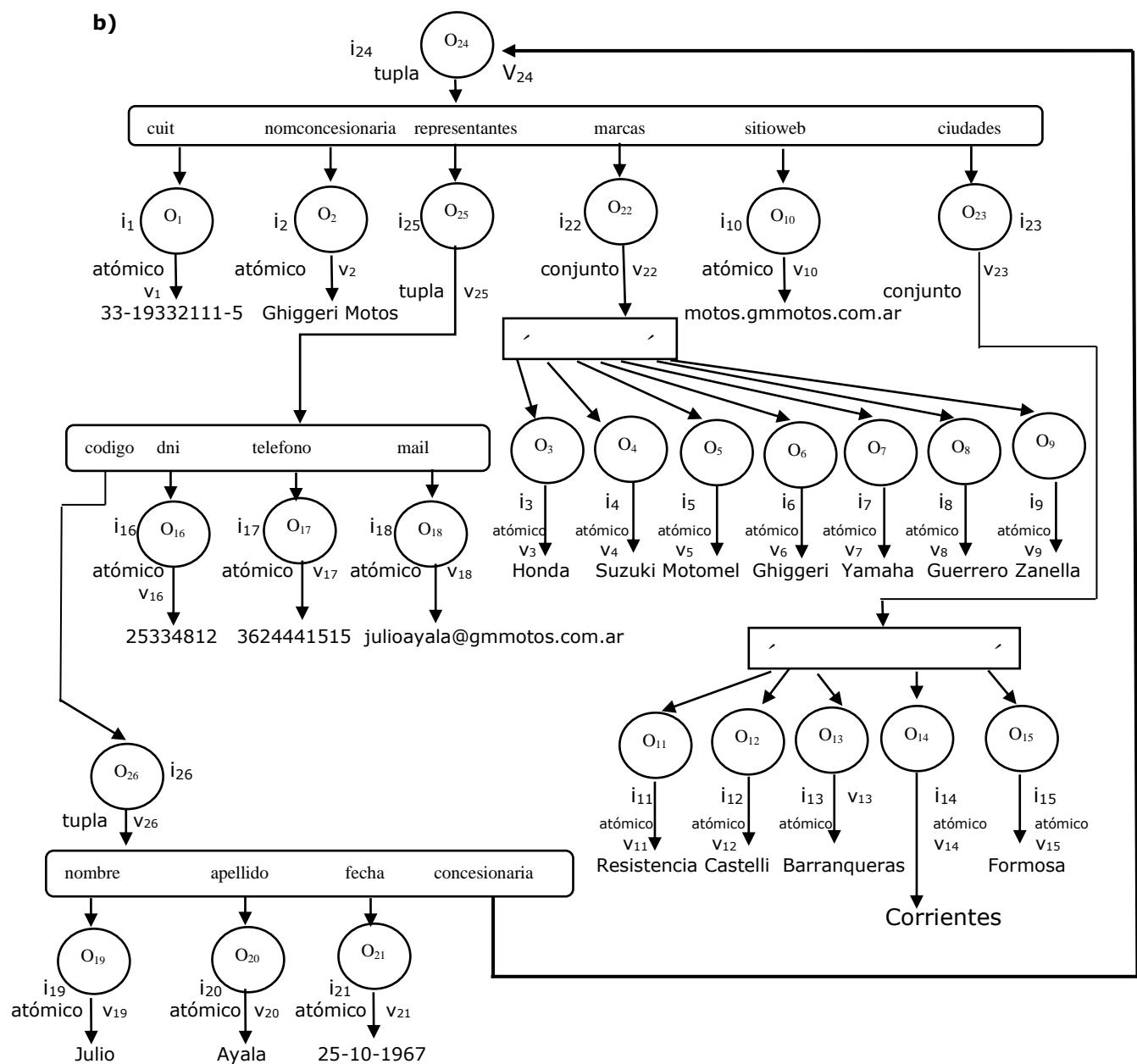


3)

a)

- $O_1 = (i_1, \text{atom}, '33-19332111-5')$
- $O_3 = (i_3, \text{atom}, 'Honda')$
- $O_5 = (i_5, \text{atom}, 'Motomel')$
- $O_7 = (i_7, \text{atom}, 'Yamaha')$
- $O_9 = (i_9, \text{atom}, 'Zanella')$
- $O_{11} = (i_{11}, \text{atom}, 'Resistencia')$
- $O_{13} = (i_{13}, \text{atom}, 'Barranqueras')$
- $O_{15} = (i_{15}, \text{atom}, 'Formosa')$
- $O_{17} = (i_{17}, \text{atom}, 3624441515)$
- $O_{19} = (i_{19}, \text{atom}, 'Julio')$
- $O_{21} = (i_{21}, \text{atom}, '25-10-1967')$
- $O_{22} = (i_{22}, \text{set}, \{i_3, i_4, i_5, i_6, i_7, i_8, i_9\})$
- $O_{24} = (i_{24}, \text{tuple}, <\text{cuit}:i_1, \text{nomconcesionaria}:i_2, \text{representantes}:i_{25}, \text{marcas}:i_{22}, \text{sitioweb}:i_{10}, \text{ciudades}:i_{23}>)$
- $O_{25} = (i_{25}, \text{tuple}, <\text{codigo}:i_{26}, \text{dni}:i_{16}, \text{telefono}:i_{17}, \text{mail}:i_{18}>)$
- $O_{26} = (i_{26}, \text{tuple}, <\text{nombre}:i_{19}, \text{apellido}:i_{20}, \text{fechanac}:i_{21}, \text{concesionaira}:i_{24}>)$

b)

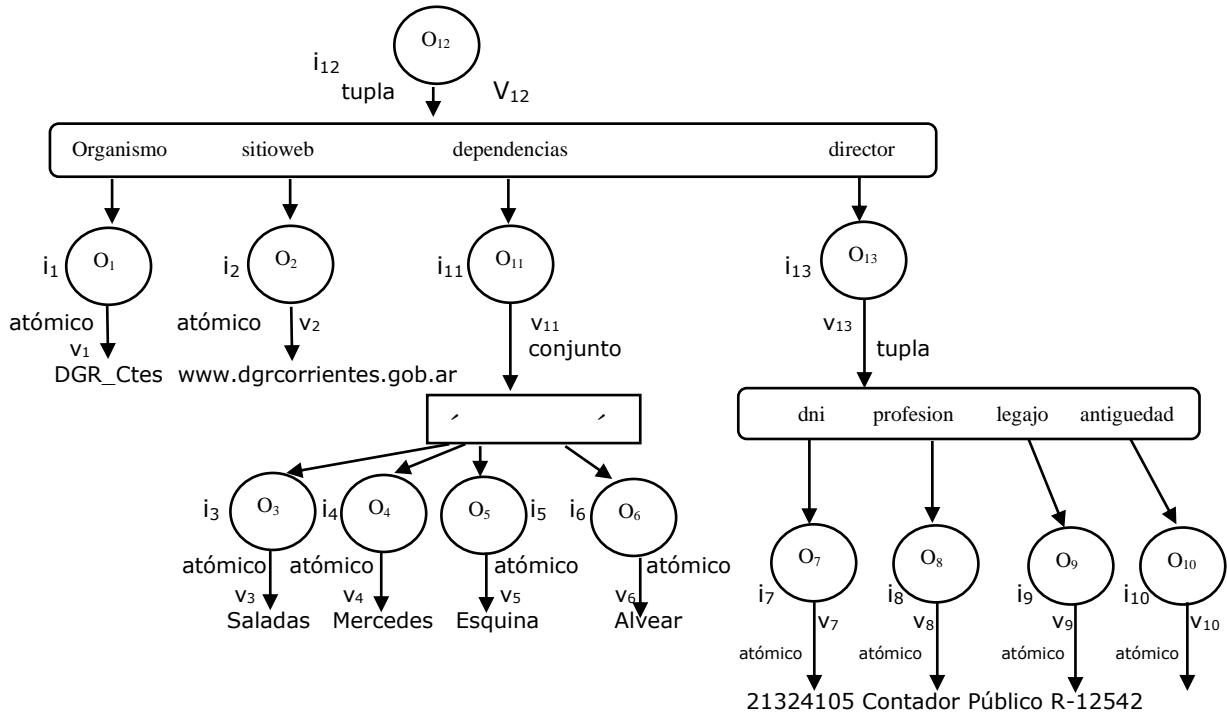


4)

a)

- O₁**= (i₁, atom, 'DGR_Ctes')
- O₃**= (i₃, atom, 'Saladas')
- O₅**= (i₅, atom, 'Esquina')
- O₇**= (i₇, atom, '21324105')
- O₉**= (i₉, atom, 'R-12542')
- O₁₁**= (i₁₁, set, {i₃, i₄, i₅, i₆})
- O₁₂**= (i₁₂, tuple, <organismo:i₁, sitioweb:i₂, dependencias:i₁₁, director:i₁₃>)
- O₁₃**= (i₁₃, tuple, <dni:i₇, profesion:i₈, legajo:i₉, antiguedad:i₁₀>)

b)



Tener en cuenta para la representación de los objetos complejos, de especificar cada:

- identificador de cada objeto (**i_x**),
- el objeto (**O_x**),
- de qué tipo de objeto se trata (atómico, conjunto, tupla o atom, set, tuple), y
- su valor actual (**V_x**)

Possibles soluciones para los ejercicios 5,6 y 7

Para los ejercicios 5,6 y 7, se deben de especificar las sentencias del lenguaje de definición de objetos ODL, bajo un sistema de gestión de bases de datos OO, con el fin de definir las clases, atributos, relaciones, etc.

Disponen de una teoría adicional (**LenguajeODL_TeoriaComplementaria.pdf**), respecto al Lenguaje ODL, que les puede ayudar a evaluar estos ejercicios, esta teoría es solo a los efectos de la resolución de los mismos, no se agrega esta parte teórica a las unidades que se contemplan para los parciales que rindan bajo la opción de Promoción.

Anteriormente esta parte de Lenguaje ODL, estaba incorporada a la Unidad de Bases de Datos Orientada a Objetos, que luego fue modificada y quedo como una parte complementaria para estos casos prácticos.

Las resoluciones que se indican a continuación, como saben, pueden sufrir las modificaciones que quieran incorporar en cada caso, ya que para estos ejercicios indicamos las soluciones modelos, que también pueden variar de acuerdo al sistema de gestión bases de datos OO con el cual uno opere.

5.

Clase Persona:

```
class Persona (extent personas key DNI)
{
    /*      Definición de atributos      */
    attribute string DNI;
    attribute struct nombrePersona {string nombre1, string nombre2, string apellido1,
                                    string apellido2} nombre;
    attribute date fechanac;
    attribute struct direPersona {string calle, integer numero, string codpostal} direccion;
}
```

```
class Profesor extent Persona (extent profesores)
{
    /*      Definición de atributos      */
    attribute integer categoria;
    attribute string despacho;
    attribute float salario;
    attribute string telefono;

    /*      Definición de relaciones      */
    relationship Departamento trabaja_en
        inverse Departamento::tiene_profesores;
}
```

```
class Estudiante extent Persona (extent estudiantes)
{
    attribute integer LU;

    relationship set<Calificacion> materias_cursadas
        inverse Calificacion::estudiante;
}
```

```
class Departamento (extent departamentos key nombre)
{
    attribute string nombre;
    attribute string despacho;
    attribute string telefono;
    attribute Profesor director;
```

```

attribute string area;

relationship set<Profesor> tiene_profesores
    inverse Profesor::trabaja_en;

relationship set<Curso> oferta
    inverse Curso::ofertado_por;

}

class Curso (extent cursos key numero)
{
    attribute string nombre;
    attribute integer numero;
    attribute string descripcion;

    relationship set<Dictado> de_dictado
        inverse Dictado::tiene_cursos;

    relationship Departamento ofertado_por
        inverse Departamento::oferta;
}

class Dictado (extent dictados)
{
    attribute short numero;
    attribute integer año;

    relationship set<Calificacion> estudiante_de
        inverse Calificacion::cursa;

    relationship set<Curso> tiene_cursos
        inverse Curso::de_dictado;
}

class Calificacion (extent calificaciones)
{
attribute float nota;

    relationship Dictado cursa
        inverse Dictado::estudiante_de;

    relationship Estudiante estudiante
        inverse Estudiante::materias_cursadas;
}

```

6.

Clase Libro:

```

class Libro (key ISBN)
{
attribute string ISBN;
attribute string titulo;
attribute integer anio;
attribute float precio;

relationship Editorial publicadoPor inverse Editorial::publica;
relationship Autor escritoPor inverse Autor::escribe;
};

```

Clase Editorial:

```
class Editorial (key nombre)
  {attribute string nombre;
   attribute string sitioWeb;

   relationship Libro publica
     inverse Libro::publicadoPor;
  };
```

Clase Autor:

```
class Autor (key nombreApellido)
  {attribute Struct nombreApellido
    (string nombre, string apellido);
   attribute string email;
   attribute Struct direccion
    (string calle, integer numero, integer codPostal);

   relationship Libro escribe
     inverse Libro::escritoPor;
  };
```

7.**Clase Facultad:**

```
class Facultad (key nombre)
  {attribute string nombre;
   attribute string sitioWeb;
   attribute string direccion;
   attribute string telefono;

   relationship Universidad administradoPor
     inverse Universidad::depende;
   relationship Alumno tieneAlumnos
     inverse Alumno::seInscribe;
   relationship Decano dirigidoPor
     inverse Decano::dirige;
  };
```

Clase Universidad:

```
class Universidad (key nombre)
  {attribute string nombre;
   attribute string sitioWeb;
   attribute string direccion;
   attribute string telefono;

   relationship Facultad depende
     inverse Facultad::administradoPor;
  };
```

Clase Alumno:

```
class Alumno (key LU)
  {attribute integer LU;
   attribute string nombre;
   attribute string direccion;
   attribute string telefono;
   attribute string mail;
   attribute string carrera;
   attribute Enum genero ('M', 'F');
   relationship Facultad seInscribe
     inverse Facultad::tieneAlumnos; };
```

Clase Decano:

```
class Decano (key dni)
  {attribute integer dni;
   attribute integer cuil;
   attribute string nombre;
   relationship Facultad dirige
     inverse Facultad::dirigidoPor;
};
```

- 8.** Pendiente para que agreguen su resolución.