13019 - Diseño de bases de datos *Curso 2004-2005*
Fecha: *27 de enero de 2005*

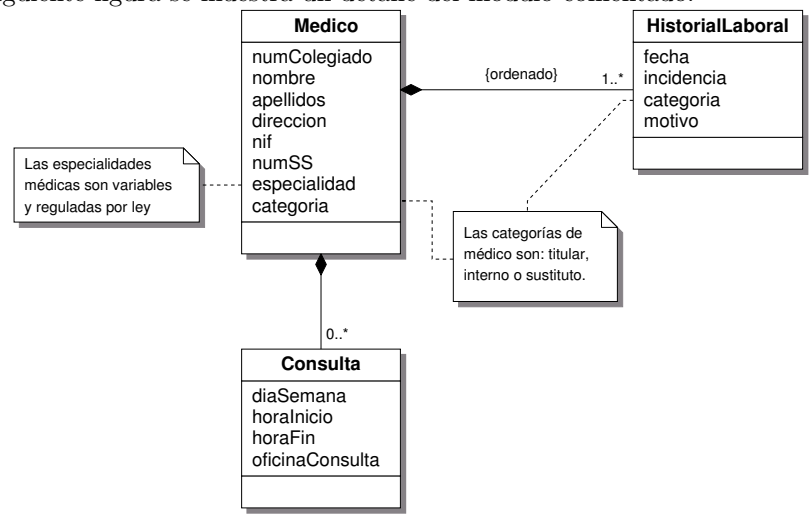
Nombre: _____ Apellidos: _____

1. (*3.00 puntos*) El *Hospital Central* desea diseñar un sistema de información para la gestión de médicos, empleados y pacientes. Se trata de un sistema de grandes dimensiones que cuenta con varios módulos y que trataremos por separado.

   Para la gestión del personal médico, se desea mantener la siguiente información: nombre, dirección (calle, población, provincia y código postal), teléfonos de contacto, NIF, número de la Seguridad Social, número de colegiado, especialidad y si es médico titular, interino o médico sustituto. Además:

   - Cada médico tiene un horario de periodicidad semanal en el que pasa consulta y que puede ser diferente cada día de la semana.
   - Por otra parte, el sistema debe mantener el historial laboral del personal médico. Para cada incidencia laboral se cuenta con un registro (alta o baja en la empresa) en el que se indica la categoría del médico en ese momento y el motivo de la incidencia (por ejemplo, un médico sustituto puede tener varias fechas de alta y baja dependiendo de las sustituciones que haya realizado).

   En la siguiente figura se muestra un detalle del módulo comentado:



   De acuerdo con esto y teniendo en cuenta que el sistema se va a implementar en una base de datos relacional, desarrolle las siguientes cuestiones:

   (a) (*0.50 puntos*) Implemente el enumerado categoria de acuerdo con los requisitos indicados en su nota adjunta.

   (b) (*0.50 puntos*) Implemente el enumerado especialidad de acuerdo con los requisitos indicados en su nota adjunta.

   (c) (*1.50 puntos*) Implemente las clases (tablas) Medico, HistorialLaboral y Consulta junto con sus relaciones y restricciones. Utilice los resultados de los apartados anteriores.

   (d) (*0.25 puntos*) Escriba una consulta SQL que resuelva el caso de uso "obtener el historial laboral de un médico dado identificado por sus apellidos".

   (e) (*0.25 puntos*) Escriba una consulta SQL que imprima la planificación semanal ordenada de consultas (día, hora de inicio y fin de consulta y oficina) de un médico identificado por sus apellidos.

(a) Se trata de un dominio enumerado con tres posibles valores con pocas posibilidades de ser modificados. La forma más conveniente de implementar el enumerado en este caso es mediante una cadena de enumeración, dejando que el DBMS mantenga la restricción. Es posible definir la restricción de varias formas, por ejemplo:

```
CREATE TABLE Medico (
  ...
  categoria CHAR(10),
  CHECK (categoria IN ('TITULAR','INTERNO','SUSTITUTO')),
  ...
);
CREATE TABLE HistorialLaboral (
  ...
  categoria CHAR(10),
  ...
);
ALTER TABLE HistorialLaboral
  ADD CONSTRAINT check_categoria
  CHECK (categoria in ('TITULAR','INTERNO','SUSTITUTO'));
```

(b) Se trata de un dominio enumerado algo más complejo, ya que cuenta con un rango mayor de posibles valores y es susceptible a variaciones de la legislación. En este caso es preferible codificar la enumeración y forzar la restricción mediante una clave ajena:

```
CREATE TABLE Especialidad (
  especialidad INTEGER PRIMARY KEY,
  nombre VARCHAR2(20)
);
CREATE TABLE Medico (
  ...
  especialidad INTEGER,
  FOREIGN KEY (especialidad) REFERENCES Especialidad,
  ...
);
```

(c) Una implementación (teniendo que cuenta que la tabla Especialidad ya exite) es:

```
CREATE TABLE Medico (
  numColegiado INTEGER PRIMARY KEY,
  nombre       VARCHAR2(50),
  apellidos    VARCHAR2(50),
  direccion    direccion_t,
  nif          CHAR(10),
  numSS        CHAR(20),
  especialidad INTEGER,
  categoria    CHAR(10),
  FOREIGN KEY (especialidad) REFERENCES Especialidad,
  CHECK (categoria IN ('TITULAR','INTERNO','SUSTITUTO'))
);
CREATE TABLE HistorialLaboral (
  idHistLab    INTEGER PRIMARY KEY,
  numColegiado INTEGER,
  fecha        DATE,
  incidencia   CHAR(4), -- ('alta','baja')
  categoria    CHAR(10),
  motivo       VARCHAR2(50),
  FOREIGN KEY (numColegiado) REFERENCES Medico
    ON DELETE CASCADE
);
CREATE SEQUENCE HistLab_seq;
ALTER TABLE HistorialLaboral
  ADD CONSTRAINT check_categoria
  CHECK (categoria in ('TITULAR','INTERNO','SUSTITUTO'));

CREATE TABLE Consulta (
  idConsulta   INTEGER PRIMARY KEY,
```

```
  numColegiado  INTEGER,
  diaSemana     INTEGER, -- (0='LUNES', 7='DOMINGO')
  horaInicio    CHAR(5), -- 'HH:MM'
  horaFin       CHAR(5), -- 'HH:MM'
  OffCon        CHAR(5)
  FOREIGN KEY (numColegiado) REFERENCES Medico
    ON DELETE CASCADE
);
```

(d) Historial laboral de un médico:

```
SELECT h.fecha, h.incidencia, h.categoria, h.motivo
  FROM HistorialLaboral h, Medico m
 WHERE m.apellidos = '_ciertos_apellidos_'
   AND m.numColegiado = h.numColegiado
 ORDER BY idHistLab;
```

(e) Horario de consulta de un médico:

```
SELECT c.dia, c.horaInicio, c.horaFin, c.OffCon
  FROM Consulta c, Medico m
 WHERE m.apellidos = '_ciertos_apellidos_'
   AND m.numColegiado = c.numColegiado
 ORDER BY c.dia, c.horaInicio;
```

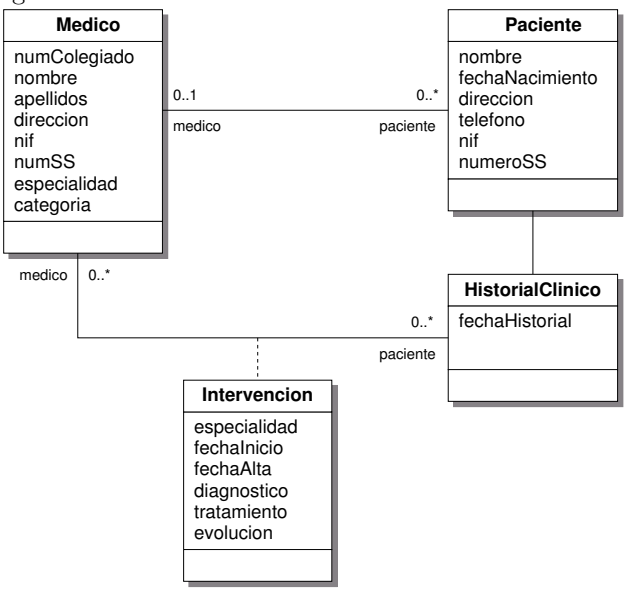
13019 - Diseño de bases de datos *Curso 2004-2005*
Fecha: *27 de enero de 2005*

Nombre: _____ Apellidos: _____

2. (*3.00 puntos*) En lo que se refiere a los pacientes del *Hospital Central*, se desea mantener la siguiente información: nombre, fecha de nacimiento, dirección (calle, población, provincia y código postal), teléfono, NIF y número de la Seguridad Social. Además:

   - A cada paciente le corresponde un médico de cabecera (cuya especialidad suele ser *Medicina General*).
   - Cada paciente tiene asociado un historial clínico, que consiste en un conjunto de intervenciones. Cada intervención está asociada a una especialidad médica y a un médico de esa especialidad, tiene fechas de inicio y alta y contiene el diagnóstico, el tratamiento y la evolución del paciente.

   En la siguiente figura se muestra un detalle del módulo comentado:



   De acuerdo con esto y siguiendo con la implementación relacional de este sistema, desarrolle las siguientes cuestiones:

   (a) (*2.00 puntos*) Implemente las clases (tablas) Paciente, HistorialClinico e Intervencion y sus relaciones. Recuerde que **no puede** modificar la clase Medico.

   (b) (*0.75 puntos*) Escriba una consulta SQL que muestre el historial clínco ordenado de un paciente dado (identificado por su número de la Seguridad Social) con indicación del nombre y número de colegiado del médico responsable de cada intervención.

   (c) (*0.25 puntos*) Escriba una consulta SQL que proporcione el número total de pacientes a cargo de un médico identificado por sus apellidos.

(a) Una posible implementación:

```
CREATE TABLE Paciente (
  numSS          CHAR(20) PRIMARY KEY,
  nombre         VARCHAR2(50),
  apellidos      VARCHAR2(50),
  direccion      direccion_t,
  telefono       telefono_t,
```

```
  nif           CHAR(10),
  medico        INTEGER,
  FOREIGN KEY (medico) REFERENCES Medico
);
CREATE TABLE HistorialClinico (
  idHistorial   INTEGER PRIMARY KEY,
  fechaHist     DATE,
  paciente      CHAR(20),
  FOREIGN KEY (paciente) REFERENCES Paciente
);
CREATE TABLE Intervencion (
  idInterv      INTEGER PRIMARY KEY,
  historial     INTEGER,
  medico        INTEGER,
  especialidad  INTEGER,
  fechaInicio   DATE,
  fechaAlta     DATE,
  diagnostico   VARCHAR2(2000),
  tratamiento   VARCHAR2(2000),
  evolucion     VARCHAR2(2000),
  FOREIGN KEY (historial) REFERENCES HistorialClinico,
  FOREIGN KEY (medico) REFERENCES Medico,
  FOREIGN KEY (especialidad) REFERENCES Especialidad
);
CREATE SEQUENCE Intervencion_seq;
```

(b) Historial clínico de un paciente:

```
SELECT i.fechaInicio, i.fechaAlta, i.diagnostico, i.tratamiento,
       i.evolucion, m.numColegiado, m.nombre, m.apellidos,
       i.especialidad
  FROM HistorialClinico h, Intervencion i, Medico m
 WHERE h.numSS = '_numero_Seguridad_Social_'
   AND h.idHistorial = i.historial
   AND i.medico = m.numColegiado
 ORDER BY i.idInterv, i.fechaInicio;
```

(c) Número total de pacientes de un médico:

```
SELECT SUM(p.numSS)
  FROM Medico m, Paciente p
 WHERE m.apellidos = '_ciertos_apellidos_'
   AND m.numColegiado = p.medico;
```

13019 - Diseño de bases de datos *Curso 2004-2005*
Fecha: *27 de enero de 2005*

Nombre: _____ Apellidos: _____

3. (*3.00 puntos*) Partiendo del diagrama de clases UML de la cuestión anterior, responda a las siguientes cuestiones:

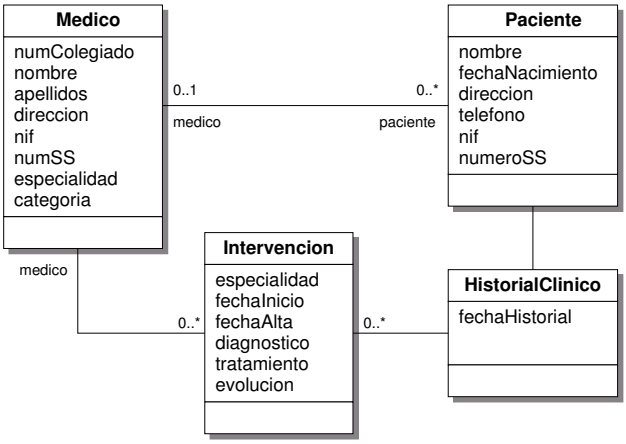
   (a) (*0.25 puntos*) Modifique el diagrama de clases mediante una transformación de equivalencia promoviendo la clase de asociación a clase.

   (b) (*2.00 puntos*) Proyecte el diagrama de clases en el estándar ODMG/C++. Todas las asociaciones deben considerarse bidireccionales.

   (c) (*0.50 puntos*) Implemente el método void Paciente::printHistoriaClinica(void). Este método debe imprimir los datos del paciente y toda su historia clínica ordenada con indicación del nombre y número de colegiado del médico responsable de la intervención.

   (d) (*0.25 puntos*) Implemente el método void Medico::printListaPacientes(void) que imprima la lista de pacientes (apellidos, nombre y número de la Seguridad Social) a cargo de un médico.

(a) El esquema modificado resulta ser:



(b) Implementación C++:

```
#include <odmg>

class Medico : public d_Object {
protected:
  d_Rel_Set<Paciente,"medico"> pacientes;
  d_Rel_Set<Intervencion,"medico"> intervenciones;
public:
  d_Integer numColegiado;
  d_String nombre, apellidos;
  Direccion direccion;          // Implementación no definida
  d_String nif;
  d_String numSS;
  Especialidad especialidad;    // Implementación no definida
  Categoria categoria;          // Implementación no definida

  void printListaPacientes(void);
};
class Paciente : public d_Object {
protected:
  d_Rel_Ref<Medico,"pacientes"> medico;
```

```
  d_Rel_Ref<HistorialClinico,"paciente"> historial;
public:
  d_String numSS;
  d_String nombre, apellidos;
  d_Date fechaNacimiento;
  Direccion direccion;        // Implementación no definida
  d_String telefono;
  d_String nif;

  void printHistoriaClinica(void)
};
class HistorialClinico : public d_Object {
protected:
  d_Rel_Ref<Paciente,"historial"> paciente;
  d_Rel_List<Intervencion,"historial"> intervenciones;
public:
  d_Date fechaHistorial;

  void printIntervenciones(void);
};
class Intervencion : public d_Object {
protected:
  d_Rel_Ref<HistorialClinico,"intervenciones"> historial;
  d_Rel_Ref<Medico,"intervenciones"> medico;
public:
  Especialidad especialidad; // Implementación no definida
  d_Date fechaInicio, fechaAlta;
  d_String diagnostico, tratamiento, evolucion;
};
```

(c) void Paciente::printHistoriaClinica(void):

```
#include <iostream>
void HistorialClinico::printIntervenciones(void) {
  d_Iterator<d_Ref<Intervencion>> ii;
  d_Ref<Intervencion> laIntervencion;

  for (ii = intervenciones.create_iterator(); ii.not_done();
       ii.advance()) {
    laIntervecion = ii.get_element();
    cout << laIntervencion->fechaInicio << "etc... "
         << laIntervencion->medico->nombre << ", "
         << laIntervencion->medico->apellidos << " ("
         << laIntervencion->medico->numColegiado << ")\n";
  }
}
void Paciente::printHistoriaClinica(void) {
  cout << "Paciente: " << nombre << " " << apellidos << "\n";
  cout << "Fecha Historial: " << historial->fechaHistorial;
  cout << "Intervenciones:\n";
  historial->printIntervenciones();
}
```

(d) Lista de pacientes de un médico:

```
#include <iostream>
void Medico::printListaPacientes(void) {
  d_Iterator<d_Ref<Paciente>> ip;
  d_Ref<Paciente> unPaciente;

  for (ip = pacientes.create_iterator(); ip.not_done();
       ip.advance()) {
    unPaciente = ip.get_element();
    cout << unPaciente->nombre << " "
         << unPaciente->apellidos << " ("
         << unPaciente->numSS << ")\n";
  }
}
```

13019 - Diseño de bases de datos *Curso 2004-2005*
Fecha: *27 de enero de 2005*

Nombre: _____ Apellidos: _____

4. (*1.00 puntos*) Almacenes de datos: Procesamiento analítico en línea (GROUPING SETS, ROLLUP y CUBE).