

Auxiliar Seis - CC20A

Computación II - Sección 2

Oscar E. Alvarez Callaú
oalvarez@dcc.uchile.cl

Santiago - Chile, Sep/27/2007

1. P1 aux5, versión resumida

Tomaremos este ejercicio para continuar, pero antes simplificaremos unos detalles (esta simplificación es por razones de simplicidad del problema y este pueda servir de buena materia pedagógica). Ahora el flujo que demos seguir para obtener una buena implementación de nuestro modelo E/R o nuestra solución en un plano *lógico* (meramente en papel y nuestra cabezas) a uno *físico* (en secuencia de bits que el computador interprete para mostrarnos los resultados esperados) se puede simplificar en los siguientes.

1. Obtener el modelo E/R corregido y sin ningún detalle ambiguo [ver sección 1.1]
2. Convertir el modelo E/R a un modelo Relacional [ver sección 1.2]
3. Transcribir cada relación en el modelo relacional a una tabla en el sistema administrador de base de datos (Access, Mysql, ...) [ver sección 2.1]
4. Construir por cada requerimiento o consulta una *query* en SQL para solucionarla en función a las tablas creadas en el punto anterior [ver sección 2.2]

Esta es una pauta muy reducida y simplificada que por ejemplo no incluye la normalización que debe hacerse a las relaciones en el modelo relacional (esto justamente para optimizar la estructura de la base de datos creada y garantizar su correcto funcionamiento en la mayoría de los casos) o como es el caso de la inserción de las tuplas o filas de cada tabla ya creada en SQL, vital para poder probar nuestras consultas, pero nada interesante en sí, ya que representa una sola instrucción `insert into ...`, dejamos como trabajo del lector entonces completar esta pauta definida en la auxiliar.

1.1. Modelo E/R

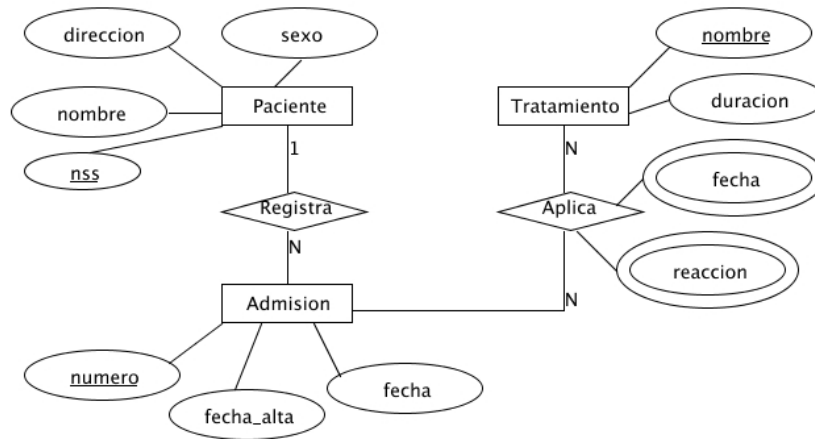


Figura 1: Modelo E/R del problema reducido.

1.2. Modelo Relacional

Paciente (nss, nombre, direccion, sexo).

Por la entidad Paciente, toda entidad debe tener su correspondiente relación en el modelo Relacional con minimamente la misma cantidad de atributos que tiene en el modelo entidad-relación (se excluye en caso de los atributos derivados o calculables, escritos en líneas segmentadas).

Admision (numero, nss, fecha, fecha_alta), llave externa (nss) que referencia a Paciente(nss).

Relación creada por la entidad Admisión, observemos que se agrega un atributo o columna más que en si es la llave externa que se pasa de la relación existente en Paciente y Admisión \rightarrow *Registra* con cardinalidad 1:N, en estos caso la llave de la entidad con cardinalidad 1 le pasa su llave a la entidad con cardinalidad N (del uno va al muchos) y de existir atributos en la relación (*Registra*) estos también pasaría a la entidad Admisión (caso particular son los atributos multivaluados que si pasan a la entidad y forman parte de la llave o el caso de los derivados que se simplifican).

Tratamiento (nombre, duracion). Por la entidad Tratamiento.

Aplicacion Tratamiento (numero, nombre, fecha, reaccion), llave externa (numero) que referencia a Admision(numero) y llave externa (nombre) que referencia a Tratamiento(nombre).

Se crea por la relación *Aplica* entre las entidades Tratamiento y Admisión, el caso concreto es que toda relación N:N necesariamente crea una tabla o relación en el modelo relacional y las llaves de las entidades involucradas en la relación pasan a esta nueva relación a formar parte de la llave en conjunción con una llave interna que pueda tener y los atributos multivaluados, como es el caso de fecha y reacción y este es el motivo por el cual forman parte de la llave.

Este ejemplo es muy útil y sencillo de seguir, solo faltaría definir que pasa con las relaciones 1:1 en la cardinalidad del modelo E/R, pero no es un caso muy complicado ya que se sigue la técnica de pasar la llave de la relación con participación parcial a la entidad con participación total. Y el caso de las entidades débiles que se las considera como entidades normales con

la diferencia de que reciben la llave de la entidad o entidades fuertes de las que depende y agrega estas llaves externas como parte de su llave.

2. SQL

SQL el lenguaje estandar de administración y consulta de base de datos.

2.1. Creación de tablas

```
Paciente create table Paciente(  
    nss int not null,  
    nombre varchar(50) not null default 'NN',  
    dir varchar(100),  
    sexo enum('f','m'),  
    primary key (nss)  
);
```

```
Admisión create table Admision(  
    numero int not null,  
    nss int not null,  
    fecha date not null default today(),  
    fecha_alta date,  
    primary key (numero),  
    foreign key (nss) references Paciente(nss)  
);
```

```
Tratamiento create table Tratamiento(  
    nombre varchar(20) not null,  
    duracion int default 1,  
    primary key (nombre)  
);
```

```
AplicaciónTratamiento create table AplicacionTratamiento(  
    numero int not null,  
    nombre varchar(20) not null,  
    fecha date not null,  
    reaccion varchar(50) not null,  
    primary key (numero,nombre,fecha,reaccion),  
    foreign key (numero) references Admision(numero),  
    foreign key (nombre) references Tratamiento(nombre)  
);
```

2.2. Consultas

- Todos los pacientes que entraron desde el 30/Ene/01 hasta hoy.

```
select p.nss,p.nombre  
from Paciente p, Admision as a  
where p.nss=a.nss and fecha >= '30/1/01'
```

- Contar todos los pacientes del género femenino.

```
select count(*)  
from Paciente  
where sexo='f'
```

- Contar cuantos paciente por tratamiento hay actualmente.

```
select t.nombre,count(a.nss)
from AplicacionTratamiento t, Admision a
where a.numero=t.numero and a.fecha_alta is null
group by t.nombre
```

- Lista de los pacientes que han sido dados de alta ordenados por nombre.

```
select p.nombre
from Paciente p, Admision a
where p.nss=a.nss and a.fecha_alta is not null
order by p.nombre
```

- Los paciente con mas de 5 reacciones al tratamiento X en una misma aplicación.

```
select a.nss, count(t.reaccion)
from Admision a, AplicacionTratamiento t
where t.nombre ='x' and a.numero=t.numero
group by a.nss,t.fecha
having count(t.reaccion) > 5
```