Ingeniería en Sistemas ula

Bases de datos

Modelos de datos

Modelo Relacional. Transformación ERE-Relacional
Lenguaje de consulta SQL

Modelo relacional (MR)

- Base de los SGBDR
- Propuesto por E. Codd en 1970, tiene mas de 45 años!
- Ha generado una industria de muchos billones de dólares.

- Modelo SIMPLE que permite consultas mediante lenguajes de alto nivel.
- Este modelo tiene implementaciones muy eficientes y también existen implementaciones de su lenguaje de consulta.

Conceptos básicos MR

Una ***base de datos relacional es un conjunto de relaciones*** (tablas), cada una con nombre diferente y sus ocurrencias (hechos) son las **tuplas** de esas relaciones

- Una **relación** tienen un conjunto predefinido de **atributos** (columnas)
- Los valores de cada **atributo** se almacenan en **tuplas** (filas)
- Cada **atributo** tiene un tipo (**dominio**)

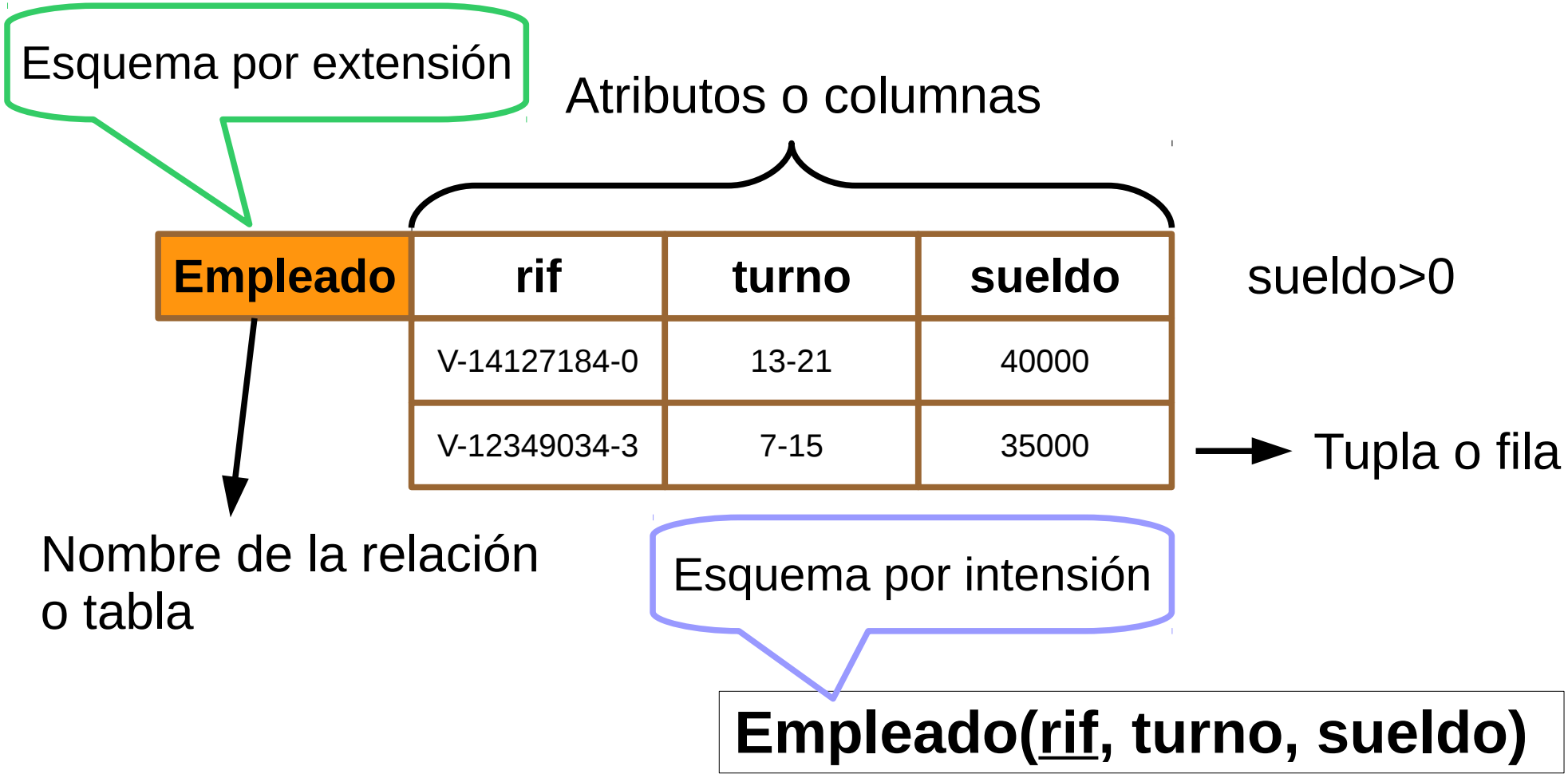
Dominio

Conjunto de valores posibles para un atributo.

Ej: Entero>0; Cadena(50) letras, digitos, . , # ‘ / -;

NULL: valor especial cuando lo desconocemos o es indefinido para una tupla

Modelo relacional



Modelo relacional- Esquema por intensión

Semestre(<u>código</u>, fechaInicio, fechaFin, fechaInscripción)
Estudiante(<u>ci</u>, nombres, apellidos, *cod_carrera*)
Carrera(<u>cod_carrera</u>, nombre)
Profesor(<u>ci</u>, nombres, apellidos, *id_dpto*)
Departamento(<u>id_dpto</u>, nombre, *jefe*, *cod_carrera*)

Atributo	Descripción	Dominio
	Semestre	
código	Código del semestre	D1: Semestres: Cadena(4), sub(código,i,1) Є {letras} i=1 y sub(código,j,1) Є {'-'} j=2 y sub(código,k,1) Є {dígitos} k=3,4
fechaInicio	Fecha de inicio del semestre	D2: Fechas: Fecha

Conceptos MR

Esquema de la base de datos contempla la *descripción estructural de sus relaciones*.

Nombre+atributos+dominio de sus atributos

Instancia de la BD: contenido real en un momento determinado

Clave primaria: atributo que tiene un valor único en cada tupla, o conjunto de valores cuya combinación es única en cada tupla

Clave foránea

Clave foránea: Si un atributo “a” que pertenece a “tabla1” es también la clave primaria de “tabla2”, entonces “a” es un atributo foráneo de “tabla1”

Reglas de formación de una BDR

1. Cada tabla contiene un solo tipo de fila
2. Cada fila contiene un número fijo de columnas
3. Cada fila es única y se identifica con su clave primaria.
4. Una clave candidata es aquel atributo o grupo de atributos que identifiquen unívoca e inequívocamente cada fila de la tabla
5. La clave primaria de una tabla se selecciona entre las claves candidatas.

Reglas de formación de una BDR

6. El orden de las filas en la tabla es irrelevante.
7. Los valores de las columnas deben pertenecer al dominio de cada atributo definido en ella.
8. Un mismo dominio puede ser usado por distintos atributos.
9. A partir de una o más tablas se pueden producir nuevas tablas diferentes usando las operaciones del álgebra relacional.

Reglas de integridad de una BDR

1. **De la relación**: ningún componente de un valor de los atributos que conforman la clave primaria puede ser nulo.
2. **De referencia**: si “a” es la clave primaria de Tabla1 y también un atributo foráneo de Tabla2, entonces para toda fila de Tabla2 donde “a” no sea nulo debe existir la tupla correspondiente en Tabla1.
3. **De los valores de un atributo**: son los predicados definidos por el administrador de BD sobre los valores de los atributos usando el lenguaje de definición de datos (LDD)

Restricciones e integridad

Semestre(<u>código</u>, fechaInicio, fechFin, fechaInscripción)
Estudiante(<u>ci</u>, nombres, apellidos, *cod_carrera*)
Carrera(<u>cod_carrera</u>, nombre)
Profesor(<u>ci</u>, nombres, apellidos, *id_dpto*)
Departamento(<u>id_dpto</u>, nombre, *jefe*, *cod_carrera*)

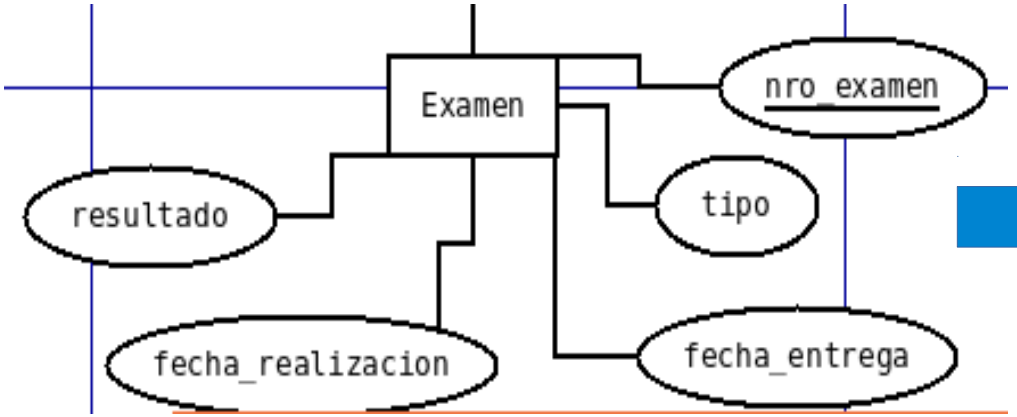
fechaInicio < fechaFin
fechaInscripcion < fechaInicio

Modelo objeto-relacional

- Puedes crear nuevos tipos de datos, funciones y operadores.
- Soporta el encadenamiento dinámico y herencia en los tipos tupla o registro.
- Soporta reglas mediante los triggers o gatillos.

Transformación de ERE a Relacional

- Cada entidad se convierte en una tabla donde sus atributos pasan a ser las columnas y el atributo clave se convierte en la clave primaria de la tabla. Cada fila de la tabla es una entidad del conjunto entidad.



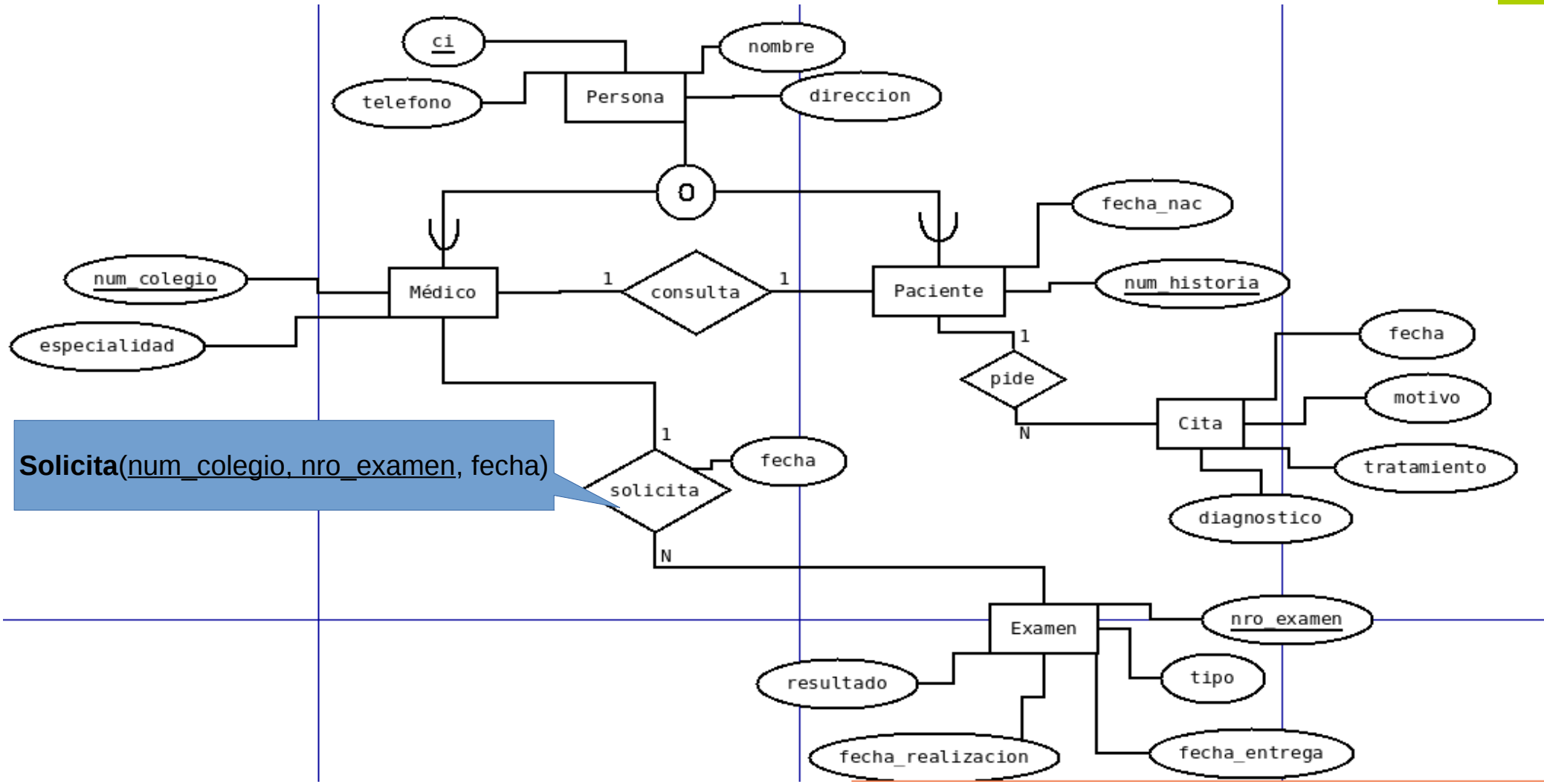
Examen(nro_examen, tipo, fecha_entrega, fecha_realizacion, resultado)
fecha_entrega>fecha_realizacion

Transformación de ERE a Relacional

- Cada conjunto relación entre los conjuntos entidades que asocia se convierte en un esquema de relación si:
  - El conjunto relación tiene atributos
  - El tipo de correspondencia del mismo es N:M

La clave primaria del esquema de relación es la concatenación de las claves primarias de los conjuntos entidad que ella asocia y sus atributos son los mismos del conjunto relación tratado

Transformación de ERE a Relacional



Solicita(num_colegio, nro_examen, fecha)

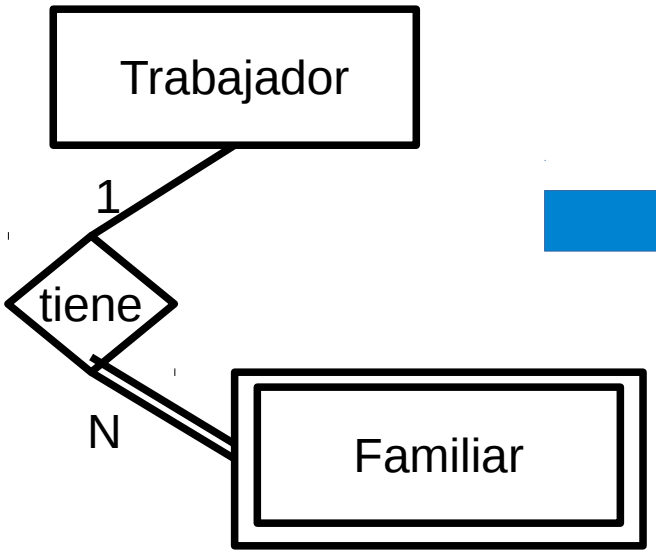
Transformación de ERE a Relacional

- Los conjuntos de valores de los atributos del diagrama ERE se convierten en los dominios del modelo relacional.

Atributo	Descripción	Dominio
num_historia	Número de historia clínica del paciente	D1: Historias: Cadena(9), sub(código,i,1) Є {dígitos} i=1 al 9
fecha_nac	Fecha de nacimiento del paciente	D2: Fechas: Fecha

Transformación de ERE a Relacional

- Las entidades débiles se convierten en tablas con clave primaria igual a la concatenación de la clave primaria de la entidad de la que depende con algún atributo propio de la entidad débil que identifique a cada fila de la tabla.

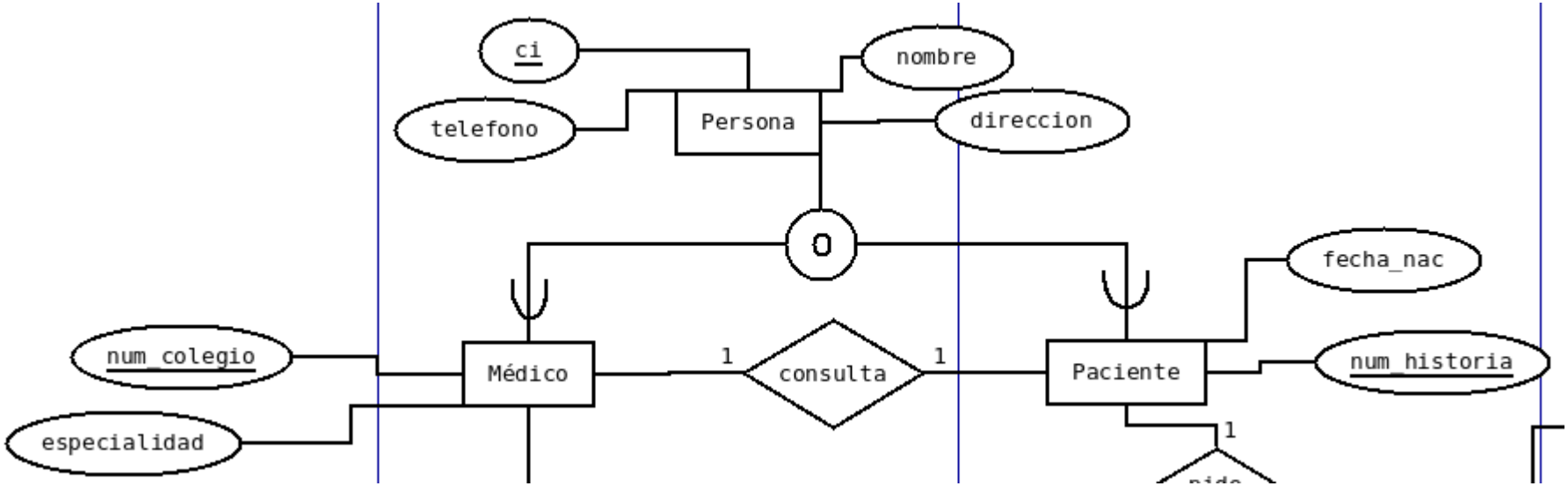


Trabajador(<u>ci</u>, rif, nombre, apellido, fecha_nac)
Familiar(<u>ci, nombre</u>, parentesco, fecha_nac)

Transformación de ERE a Relacional

- Cada especialización es una tabla con los atributos de la especialización y con la misma clave de la entidad general.

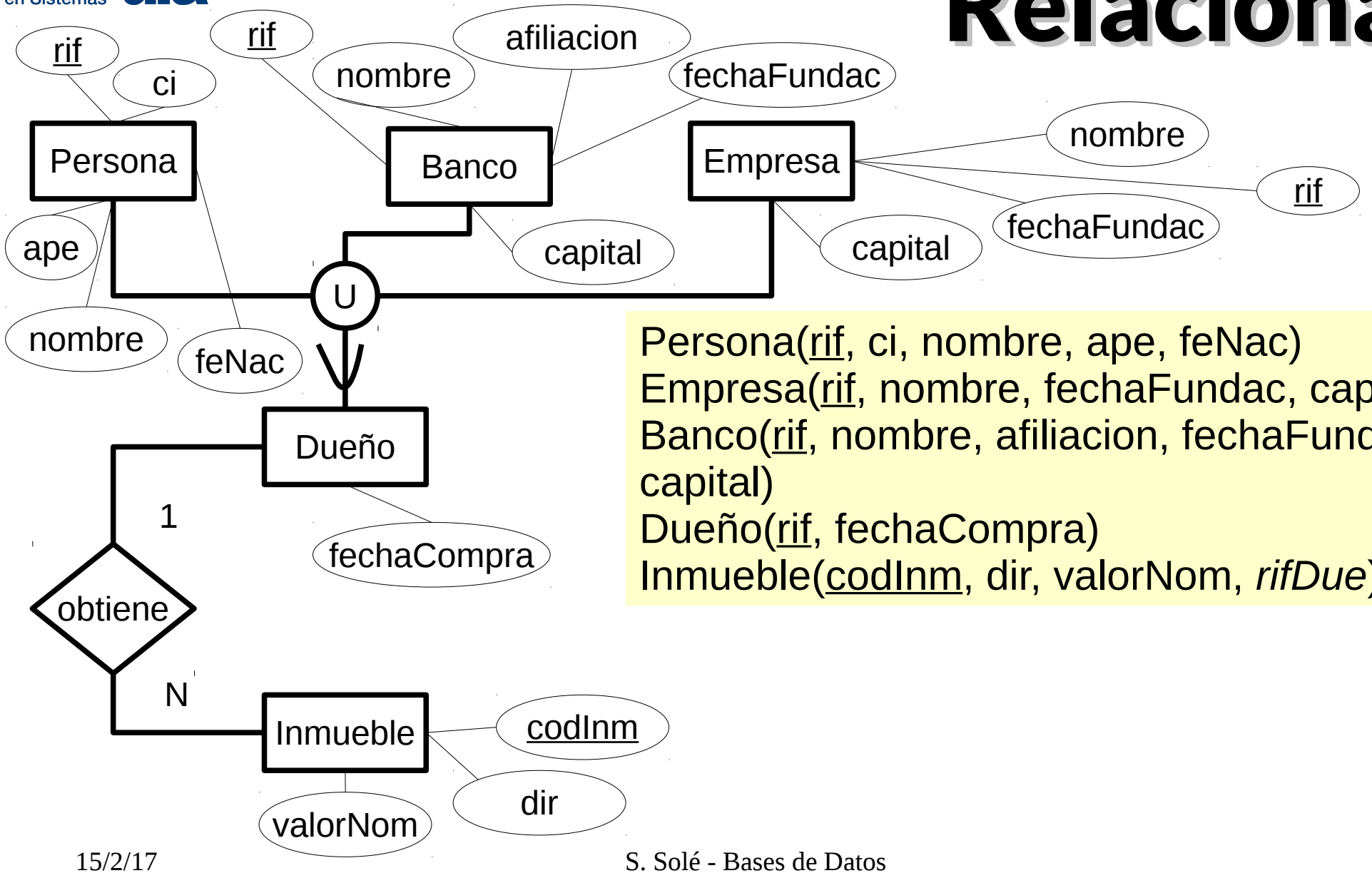
Persona(ci, nombre, dirección, teléfono)
Médico(ci, num_colegio, especialidad)
Paciente(ci, num_historia, fecha_nac)



Transformación de ERE a Relacional

- Una categoría es una subclase de la unión de dos o más superclases, por lo que se crea una clave para la categoría y la misma se coloca en los esquemas de relación de las superclases si tienen diferentes esquemas.
- Los conjuntos relación cuyos tipos de correspondencia sean 1:N o N:1 ó 1:1 indican la toma de la clave primaria del conjunto entidad asociado al 1 para colocarlo como clave foránea en el esquema de relación asociado a N

Transformación de ERE a Relacional



Persona(<u>rif</u>, ci, nombre, ape, feNac)
Empresa(<u>rif</u>, nombre, fechaFundac, capital)
Banco(<u>rif</u>, nombre, afiliacion, fechaFundac, capital)
Dueño(<u>rif</u>, fechaCompra)
Inmueble(<u>codInm</u>, dir, valorNom, *rifDue*)

Lenguaje de definición de datos

LDD para el modelo relacional: **SQL** (Structured Query Language), también es un LMD

- Lenguaje estándar de consulta para BD relacionales.
- Es la versión comercial del lenguaje SEQUEL creado por IBM.
- Lenguaje declarativo
- Puede usarse en cónsola, usando una IGU del SGBDR o embebido en programas de aplicación.
- Soportado por los principales SGBD comerciales
- Basado en álgebra relacional

LDD en SQL3

- Objetos de la BD (varían según la implementación del SGBD)

- Para crear objetos: **CREATE**
- Para destruir objetos: **DROP**
- Para modificar objetos: **ALTER**

- Objetos de la BD en *Postgresql*: server, database, table, schema, tablespace, view, function, operator, cast, sequence, extension, data type, domain, trigger

Ingeniería en Sistemas ula

LDD en SQL3

Las sentencias del LDD se usan para crear y modificar la estructura de las relaciones así como otros objetos de la base de datos.

CREATE - para crear objetos en la base de datos.

ALTER - modifica la estructura de la base de datos.

DROP - borra objetos de la base de datos.

TRUNCATE - elimina todos los registros de la tabla, incluyendo todos los espacios asignados a los registros.

Crear Relaciones

CREATE TABLE <nomRelacion> (<nomAt> <tipo> [NOT NULL | WITH DEFAULT],...);

```
CREATE TABLE Persona (
    ci VARCHAR(10) NOT NULL,
    nombre VARCHAR(30) NOT NULL,
    direccion VARCHAR(70),
    telefono VARCHAR(12),
    fechaNac DATE
);
```

Restricciones

- Atributo único (unique): no hay dos filas con el mismo valor en esa columna,
- Atributo clave primaria (primary key), es única y no acepta nulos,
- Atributo clave foránea (foreign key), hace referencia a la clave primaria de otra tabla, aceptando nulos.
- Restriciones de verificación (check) especifica una condición de búsqueda

```
CREATE TABLE Persona (
    ci VARCHAR(10) PRIMARY KEY,
    nombre VARCHAR(30) NOT NULL,
    direccion VARCHAR(70),
    telefono VARCHAR(12),
    fechaNac DATE,
    numHijos smallint,
    CHECK (numHijos>0)
);
```

```
CREATE TABLE Medico (
    ci VARCHAR(10) NOT NULL FOREIGN
KEY (ci) REFERENCES Persona,
    numcolegio VARCHAR(30) NOT NULL
UNIQUE,
    especialidad VARCHAR(70)
);
```

Restricciones atributos de una relación

```
CREATE TABLE Paciente (
    ci VARCHAR(10) PRIMARY KEY,
    numhistoria VARCHAR(30) NOT NULL UNIQUE,
    fechaNac DATE,
    CHECK (fechaNac >= TODAY()),
    CONSTRAINT pkPersona FOREIGN KEY ci
REFERENCES Persona(ci)
);
```

también

CONSTRAINT fechaNac_const CHECK (fechaNac >= TODAY())

Restricciones de dominio

```
CREATE DOMAIN colores CHAR(8) CHECK (VALUE IN
('verde', 'negro', 'rojo', 'azul'));


CREATE TABLE Producto (
    nroPro CHAR(6) PRIMARY KEY,
    nombrePro VARCHAR(64),
    cantidad int,
    color colores
);

CREATE TYPE colores AS ENUM ('verde', 'rojo', 'gris');
```

Sintaxis SQL - LDD

```
CREATE TABLE [schema.]table
        ( { column datatype [DEFAULT expr]
[column_constraint] ...
          | table_constraint}
        [, { column datatype [DEFAULT expr]
[column_constraint] ...
          | table_constraint} ]...)
        [AS subquery]
```

Sintaxis SQL - LDD

CLAVES PRIMARIAS:

A nivel de columna:
COLUMN [data type] [CONSTRAINT <constraint name> PRIMARY KEY]

A nivel de tabla:
CONSTRAINT [constraint name] PRIMARY KEY [column (s)]

Sintaxis SQL - LDD

CLAVES FORÁNEAS:

A nivel de columna:
COLUMN [data type] [CONSTRAINT] [constraint name] [REFERENCES] [table name (column name)]

A nivel de tabla:
CONSTRAINT [constraint name] [FOREIGN KEY (foreign key column name) REFERENCES] [referenced table name (referenced column name)]

Sintaxis SQL - LDD

RESTRICCIONES:

A nivel de tabla:
CONSTRAINT [name] CHECK (condition)

```
CREATE TABLE TEST
( ...,
   CONSTRAINT TEST_CHK
   CHECK (stdate < = enddate),
);
```

A nivel de columa:
COLUMN [data type] CONSTRAINT [name] [CHECK (condition)]

```
CREATE TABLE TABLE_NAME
( ...,
   GRADE char (1) CONSTRAINT TEST_CHK
   CHECK (upper (GRADE) in ('A','B','C')),
   ...
);
```

Sintaxis SQL - LDD

ALTER TABLE [TABLE NAME] RENAME TO [TABLE NAME NEW];

ALTER TABLE [TABLE NAME] ADD ([COLUMN NAME] DATATYPE);
ALTER TABLE [TABLE NAME] RENAME COLUMN [COLUMN NAME] TO [COLUMN NAME NEW];
ALTER TABLE [TABLE NAME] DROP COLUMN [COLUMN NAME];

ALTER TABLE [TABLE NAME] ADD PRIMARY KEY ([COLUMN NAME]);
ALTER TABLE [TABLE NAME] DROP PRIMARY KEY;

Sintaxis SQL - LDD

DROP TABLE [TABLE NAME] [PURGE];

TRUNCATE TABLE [TABLE NAME];

Ingeniería en Sistemas ula

Ejercicio

SQLite:

```
$ sqlite3 nombrebd.db
sqlite> CREATE TABLE Persona(
rif varchar(10) PRIMARY KEY,
ci varchar(10) UNIQUE NOT NULL,
nombre varchar(40),
ape varchar(40),
edad integer CHECK (edad>0),
sexo char DEFAULT 'F'
);
sqlite> .tables
sqlite> .schema Persona
sqlite> ALTER TABLE Persona ADD COLUMN peso REAL;
sqlite> DROP TABLE Persona;
```

SQLite

Www.sqlite.org
https://www.sqlite.org/datatype3.html - Tipos de datos en SqLite3
https://www.sqlite.org/lang_createtable.html#constraints - SQL Create table