Aquí tienes la teoría del **Diseño Lógico** (transformación del modelo E/R al Relacional) organizada y separada en apartados para que te resulte sencillo estudiar:

**1. Visión General del Diseño Lógico**

* **Objetivo**: Convertir el esquema conceptual E/R en un conjunto de tablas relacionales.
* **Reglas básicas** (por cada elemento E/R, se obtiene un elemento relacional) .

**2. Transformación de Entidades**

* Cada **entidad** se convierte en una **tabla**.
  + El nombre de la tabla = nombre de la entidad.
  + Los atributos de la entidad = columnas de la tabla.
  + La clave primaria (PK) de la entidad = PK de la tabla .

**3. Transformación de Relaciones Binarias**

**3.1 Conectividad 1:1**

* Se implementa con una **clave foránea** (FK) en una de las dos tablas, más restricción de unicidad en esa FK.
* La FK no admite valores repetidos (única) y, si la participación es total, no admite NULL .

**3.2 Conectividad 1:N**

* Se añade una **FK** en la tabla correspondiente al lado “N”.
* Si la participación de “1” es total, la FK no puede ser NULL .

**4. Transformación de Relaciones M:N**

* Se crea una **nueva tabla** para la relación.
  + Sus columnas incluyen las PK de ambas entidades como FKs, que juntas forman la PK compuesta.
  + Los atributos propios de la relación pasan a columnas de esta tabla .

**5. Transformación de Relaciones N-arias**

**5.1 Relaciones Ternarias**

* Igual que M:N, pero con las PK de las tres entidades:
  + **M:N:P** → PK = tres FKs.
  + **M:N:1** → PK = FKs de los lados M y N.
  + **N:1:1** / **1:1:1** → PK = combinación de dos de las tres FKs .

**5.2 Relaciones n-arias (n>3)**

* Se crea tabla con FKs de todas las entidades;
  + Si todas las conexiones son “muchos”, PK = todas las FKs.
  + Si hay algún “uno”, PK = todas las FKs salvo la de una entidad “1” .

**6. Transformación de Relaciones Recursivas**

* **1:1** o **1:N** → add FK a la misma tabla (auto-relación).
* **M:N** o n-aria → se crea tabla intermedia como en relaciones no recursivas .

**7. Entidades Débiles**

* Se crea tabla con **cascada** en actualizaciones y borrados:
  + La FK que referencia la entidad fuerte forma parte de la PK de la tabla débil .

**8. Generalización / Especialización**

* Para cada **superclase** y **subclase** se crea una tabla:
  + Tabla superclase → atributos comunes + PK.
  + Cada tabla subclase → PK (FK a superclase) + atributos específicos .
* **Opciones de simplificación**:
  + “Merge” de subclases o colapsar la superclase, si hay muchas tablas.

**9. Resumen de Transformaciones**

| **E/R** | **Relacional** |
| --- | --- |
| Entidad | Tabla |
| Relación 1:1 | FK + unicidad |
| Relación 1:N | FK en lado N |
| Relación M:N | Tabla intermedia |
| Relación n-aria | Tabla intermedia |
| Relación recursiva | Igual que no recursiva (FK o tabla) |
| Entidad débil | Tabla con cascada |
| Generalización / Especialización | Tabla superclase y tablas subclase |

**10. Restricciones Externas**

* Para garantizar **participaciones totales**, se crean **check constraints** o FKs con NOT NULL.
* Ejemplo: “Todo proyecto debe tener al menos un empleado” → restricción externa en la tabla de relaciones proyecto–empleado .

Aquí tienes ejemplos prácticos de cada caso de transformación ER→Relacional:

**1. Entidad**

**ER:** Entidad Empleado con atributos EmpID (PK), Nombre, Departamento.

CREATE TABLE Empleado (

EmpID INT PRIMARY KEY,

Nombre VARCHAR(100),

Departamento VARCHAR(50)

);

**2. Relación Binaria 1:1**

**ER:** Entidades Persona y Pasaporte relacionadas 1:1 (cada persona tiene un único pasaporte).

* Se añade FK en Pasaporte con UNIQUE y NOT NULL.

CREATE TABLE Persona (

PersonaID INT PRIMARY KEY,

Nombre VARCHAR(100)

);

CREATE TABLE Pasaporte (

PasaporteID INT PRIMARY KEY,

FechaExpedicion DATE,

PersonaID INT NOT NULL UNIQUE,

FOREIGN KEY (PersonaID) REFERENCES Persona(PersonaID)

);

**3. Relación Binaria 1:N**

**ER:** Departamento 1:N Empleado (un departamento muchos empleados).

* FK en el lado N (Empleado).

CREATE TABLE Departamento (

DeptID INT PRIMARY KEY,

Nombre VARCHAR(50)

);

CREATE TABLE Empleado (

EmpID INT PRIMARY KEY,

Nombre VARCHAR(100),

DeptID INT NOT NULL,

FOREIGN KEY (DeptID) REFERENCES Departamento(DeptID)

);

**4. Relación Binaria M:N**

**ER:** Estudiante M:N Asignatura (los estudiantes cursan varias asignaturas).

* Tabla intermedia Inscripcion.

CREATE TABLE Estudiante (

EstID INT PRIMARY KEY,

Nombre VARCHAR(100)

);

CREATE TABLE Asignatura (

AsigID INT PRIMARY KEY,

Titulo VARCHAR(100)

);

CREATE TABLE Inscripcion (

EstID INT,

AsigID INT,

FechaInscripcion DATE,

PRIMARY KEY (EstID, AsigID),

FOREIGN KEY (EstID) REFERENCES Estudiante(EstID),

FOREIGN KEY (AsigID) REFERENCES Asignatura(AsigID)

);

**5. Relación N-aria (Ternaria)**

**ER:** Proveedor suministra Producto a Sucursal (ternaria, con atributo Precio).

* Tabla intermedia con PK de las tres.

CREATE TABLE Proveedor (

ProvID INT PRIMARY KEY,

Nombre VARCHAR(100)

);

CREATE TABLE Producto (

ProdID INT PRIMARY KEY,

Nombre VARCHAR(100)

);

CREATE TABLE Sucursal (

SucID INT PRIMARY KEY,

Ubicacion VARCHAR(100)

);

CREATE TABLE Suministro (

ProvID INT,

ProdID INT,

SucID INT,

Precio DECIMAL(8,2),

PRIMARY KEY (ProvID, ProdID, SucID),

FOREIGN KEY (ProvID) REFERENCES Proveedor(ProvID),

FOREIGN KEY (ProdID) REFERENCES Producto(ProdID),

FOREIGN KEY (SucID) REFERENCES Sucursal(SucID)

);

**6. Relación Recursiva**

**ER:** Empleado supervisa Empleado (1:N).

* FK SupervisorID en la misma tabla.

CREATE TABLE Empleado (

EmpID INT PRIMARY KEY,

Nombre VARCHAR(100),

SupervisorID INT,

FOREIGN KEY (SupervisorID) REFERENCES Empleado(EmpID)

);

Para M:N recursivo (p. ej. empleados colaboran en pares), se usaría tabla intermedia.

**7. Entidad Débil**

**ER:** Factura (fuerte) y Detalle (débil) con Linea como identificador parcial.

* Tabla Detalle con PK compuesta y cascadas.

CREATE TABLE Factura (

FactID INT PRIMARY KEY,

Fecha DATE

);

CREATE TABLE Detalle (

FactID INT,

Linea INT,

Producto VARCHAR(100),

Cantidad INT,

PRIMARY KEY (FactID, Linea),

FOREIGN KEY (FactID)

REFERENCES Factura(FactID)

ON UPDATE CASCADE

ON DELETE CASCADE

);

**8. Generalización / Especialización**

**8.1 Tabla única**

Superclase Vehículo y subclases Coche, Moto.

CREATE TABLE Vehiculo (

VehID INT PRIMARY KEY,

Tipo VARCHAR(10) NOT NULL, -- 'Coche' o 'Moto'

Marca VARCHAR(50),

NumPuertas INT, -- sólo para coche

Cilindrada INT -- sólo para moto

);

**8.2 Tabla por Subclase (sin superclase)**

CREATE TABLE Coche (

VehID INT PRIMARY KEY,

Marca VARCHAR(50),

NumPuertas INT

);

CREATE TABLE Moto (

VehID INT PRIMARY KEY,

Marca VARCHAR(50),

Cilindrada INT

);

**8.3 Tabla por Jerarquía**

CREATE TABLE Vehiculo (

VehID INT PRIMARY KEY,

Marca VARCHAR(50)

);

CREATE TABLE Coche (

VehID INT PRIMARY KEY,

NumPuertas INT,

FOREIGN KEY (VehID) REFERENCES Vehiculo(VehID)

);

CREATE TABLE Moto (

VehID INT PRIMARY KEY,

Cilindrada INT,

FOREIGN KEY (VehID) REFERENCES Vehiculo(VehID)

);

**9. Restricciones de Participación**

Si en una relación 1:N la participación de Empleado en Departamento fuese **total** (todo empleado debe pertenecer):

-- Departamento ya creado...

ALTER TABLE Empleado

ADD CONSTRAINT Empl\_Dept\_total

FOREIGN KEY (DeptID) REFERENCES Departamento(DeptID)

NOT DEFERRABLE;

Aquí el NOT NULL en DeptID asegura la obligatoriedad.

Con estos ejemplos verás cómo aplicar cada regla en ejercicios y tests de conversión ER→Relacional. ¡A practicar!