1 – Modelo Lógico

Para convertir de conceptual a lógico, se deben resolver estos 4 problemas:

- Atributos polivalentes: agregar entidad de tipo o débil (ver Figura 1.2).
- Atributos compuestos: dejar todos sus individuales y 1 sólo que los englobe.
- Atributos derivados (redundancia): se quita sólo si se usa y actualiza poco.
- Jerarquías: se convierte en relaciones o se eliminan padres o hijos.

✗ Error típico: convertir atributo opcional (0,1) como si fuera polivalente.

1.1 – Conversión de jerarquía

Según la COBERTURA de la misma, se pueden dar los siguientes casos:

Tabla 1.1. Posibilidad de reemplazar jerarquía por asociaciones según Fig. 1.1

Dejar todo	Total	Parcial
Exclusiva	✔	✔
Superpuesta	✔	✔

Tabla 1.2. Posibilidad de mover atributos del padre a los hijos como obligatorios (1,1)

Eliminar padre	Total	Parcial
Exclusiva	✔	✗
Superpuesta	✗	✗

Tabla 1.3. Posibilidad de mover atributos de los hijos al padre como opcionales (0,1)

Eliminar hijos	Total	Parcial
Exclusiva	✔	✔
Superpuesta	✔	✔

Podemos resumir los casos y las mejores opciones de la siguiente forma:

✗ Nunca matar al padre si la cobertura es (T,E).

✗ No matar al padre si tiene muchos atributos (ej: compuestos).

✗ No matar a los hijos si tienen asociaciones con otras entidades.

⚠ Obligatorio: agregar identificador externo en hijo si NO tiene uno interno.

✔ Opcional: agregar identificador externo en hijo si ya tiene uno interno.

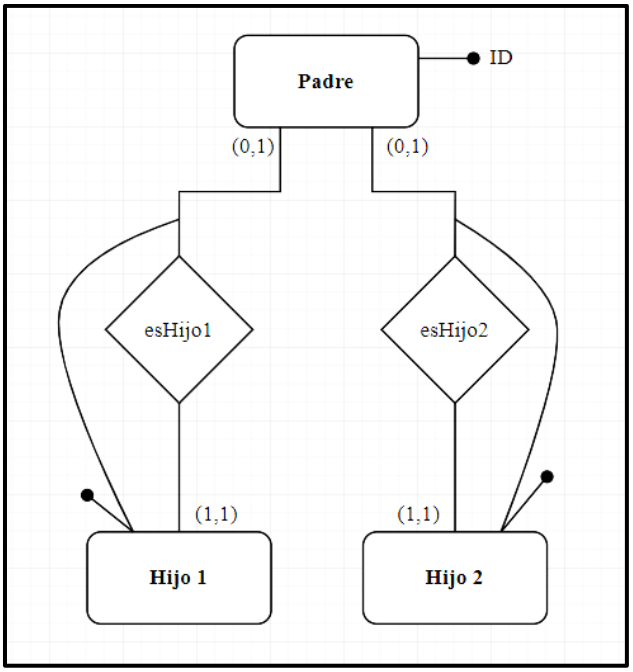

Fig. 1.1. Conversión de jerarquía dejando todas las entidades.

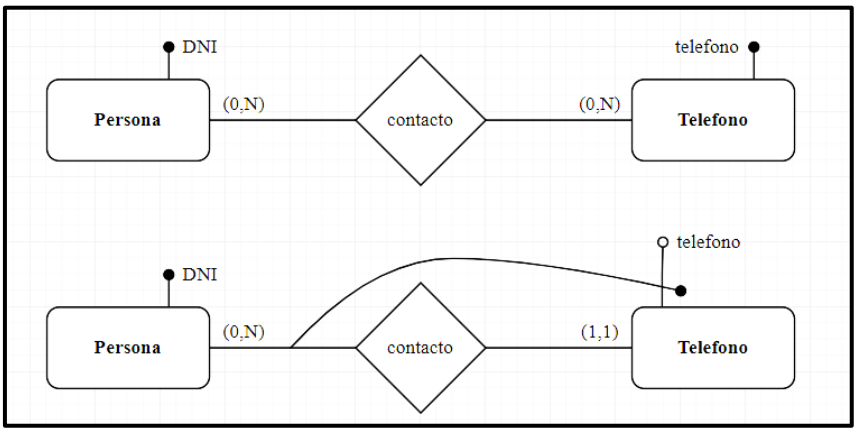

Fig. 1.2. Conversión de atributo polivalente: entidad de tipo vs entidad débil.

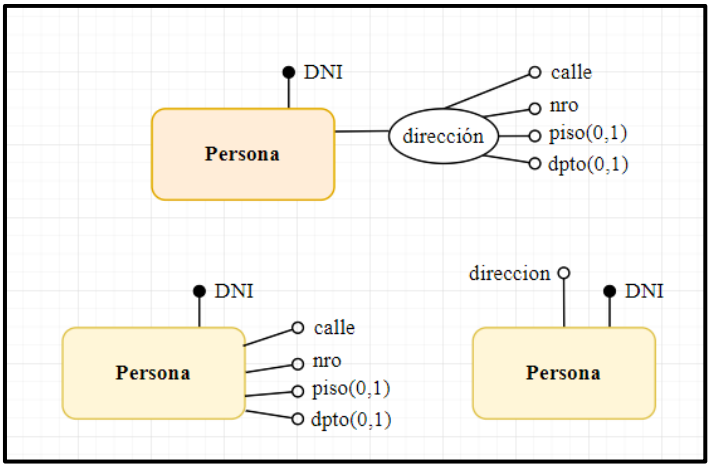

Fig. 1.3. Conversión de atributo compuesto: atributos individuales vs atributo único.

2 – Modelo Físico

Para convertir de modelo lógico a física, hay que considerar estos aspectos:

- Selección de la PK: subrayar el ID más simple y de menor tamaño.
- Atributos opcionales: se coloca un "?" al final del nombre en lugar de (0,1).
- Asociaciones entre entidades: según la cardinalidad, puede o no ser tabla.

❌ Error típico: marcar ambos ID como PK cuando con uno solo alcanza.

❌ Error típico: separar el subrayado de atributos que conforman la PK.

🔢 Cuando se usan 2 atributos de otra tabla como FK, escribir (at1,at2)(fk).

2.1 – Conversión de asociaciones

A continuación, se muestra en qué casos es necesario crear una tabla aparte:

Tabla 2.1. Resolución de la asociación entre la entidad A y B.

A / B	(0,1)	(1,1)	(0,N) o (1,N)
(1,1)	❌ (A -> B)	❌ (Juntar)	❌ (A -> B)
(0,N) o (1,N)	✔ (PK es B)	❌ (B -> A)	✔ (PK es A + B)

No olvidar que la clave primaria de una asociación puede estar formada también por un atributo de la misma, por ejemplo, la fecha de inicio (volvió a trabajar después).

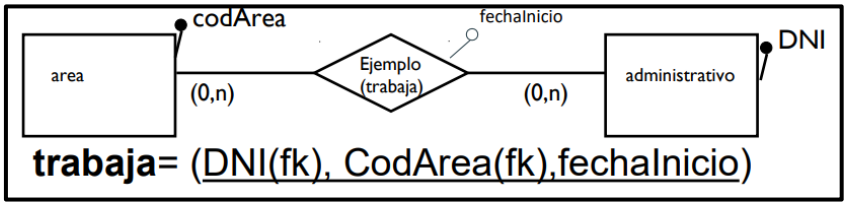

Fig. 2.1. Relación muchos a muchos, con PK también compuesta por atributo de asociación.

En el caso de que una asociación sea entre la misma entidad (ejemplo: materia con materia correlativa), se debe cambiar el nombre de una de las FK.

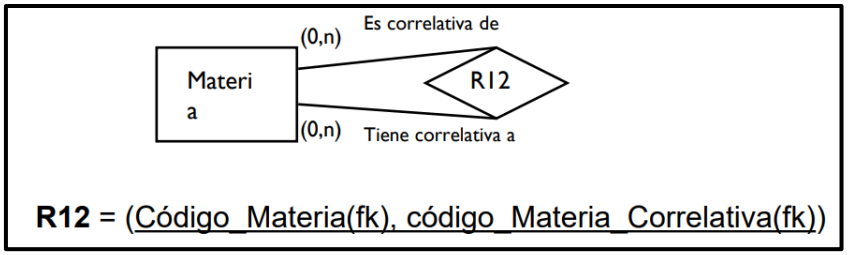

Fig. 2.2. Relación muchos a muchos, con asociación a la misma entidad.

3 – Álgebra Relacional

Tenemos operaciones unarias y binarias para realizar sobre una o dos tablas. En algunos casos se requiere que el esquema sea igual (mismo tipo y orden de columnas).

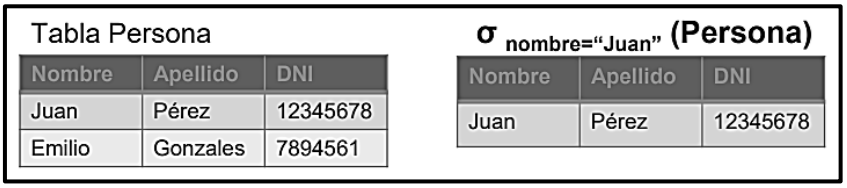

Tabla Persona

Nombre	Apellido	DNI
Juan	Pérez	12345678
Emilio	Gonzales	7894561

$\sigma_{\text{nombre="Juan"}}$ (Persona)

Nombre	Apellido	DNI
Juan	Pérez	12345678

Fig. 3.1. Operación de selección en una tabla, reduce cantidad de tuplas.

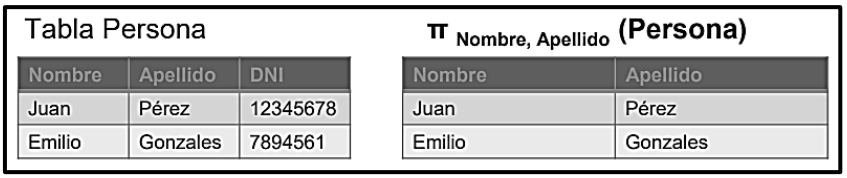

Tabla Persona

Nombre	Apellido	DNI
Juan	Pérez	12345678
Emilio	Gonzales	7894561

$\pi_{\text{Nombre, Apellido}}$ (Persona)

Nombre	Apellido
Juan	Pérez
Emilio	Gonzales

Fig. 3.2. Operación de proyección en una tabla, reduce cantidad de columnas.

Tabla Persona

Nombre	Apellido	DNI
Juan	Perez	12345678
Emilio	Gonzales	7894561

Tabla Telefono

Teléfono	DNI
224152356	12345678
221547896	12345678
011545787	7894561

Persona |X| Telefono

Nombre	Apellido	DNI	Telefono
Juan	Perez	12345678	224152356
Juan	Perez	12345678	221547896
Emilio	Gonzales	7894561	011545787

Fig. 3.3. Producto natural entre dos tablas, compara las columnas homónimas.

Tabla Persona

Nombre	Apellido	DNI
Juan	Pérez	12345678
Emilio	Gonzales	7894561

Tabla Telefono

Teléfono	DNI
224152356	12345678
221547896	12345678
011545787	7894561

Persona X Telefono

Nombre	Apellido	Persona.DNI	Telefono	Telefono.DNI
Juan	Pérez	12345678	224152356	12345678
Juan	Pérez	12345678	221547896	12345678
Juan	Pérez	12345678	011545787	7894561
Emilio	Gonzales	7894561	224152356	12345678
Emilio	Gonzales	7894561	221547896	12345678
Emilio	Gonzales	7894561	011545787	7894561

Fig. 3.4. Producto cartesiano entre dos tablas, mezcla a todos con todos. Luego se deberá hacer selección.

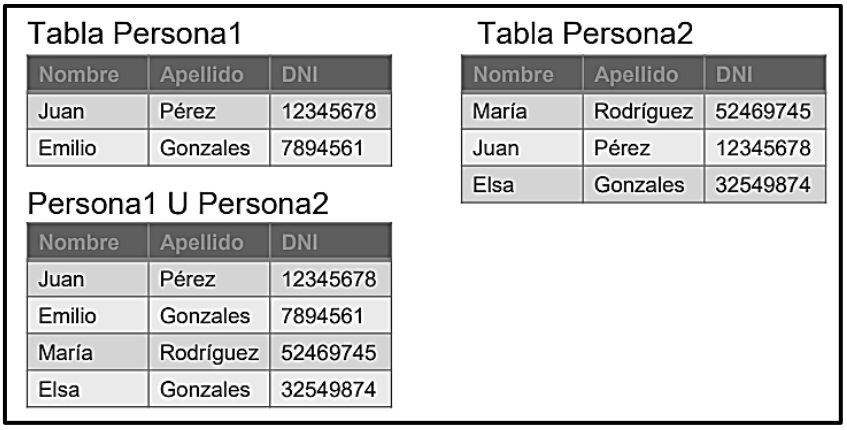

Fig. 3.5. Operación de unión entre 2 tablas de mismo esquema. No deja tuplas repetidas.

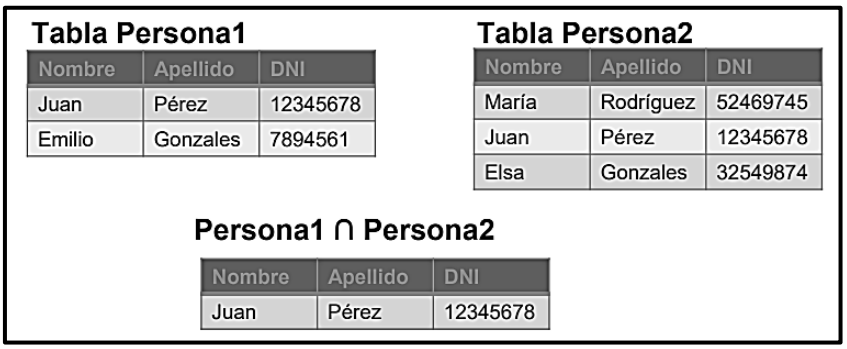

Fig. 3.6. Operación de intersección entre 2 tablas de mismo esquema. Deja sólo las tuplas repetidas.

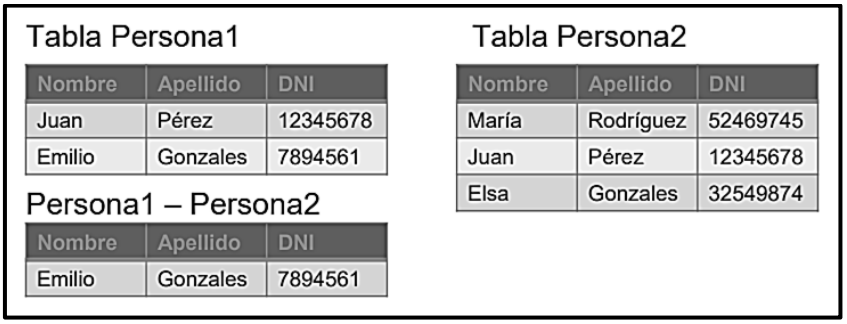

Fig. 3.7. Operación de diferencia entre 2 tablas de mismo esquema. Elimina tuplas de Tabla 2 en Tabla 1.

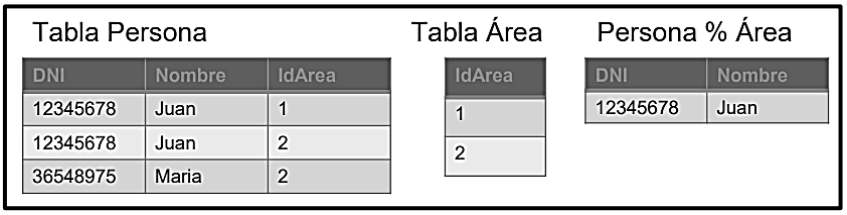

Fig. 3.8. Operación de división entre 2 tablas. Esquema de Tabla 2 debe estar incluido en Tabla 1.

Para insertar tuplas, se le asigna a la tabla la unión de los atributos: $T \leftarrow T \cup \{...\}$

Para eliminar tuplas, se le asigna a la tabla una diferencia: $T \leftarrow T - \sigma_{cond}(T)$

Para modificar tuplas, se representa el cambio mediante: $\delta_{at=nuevoValor}(T)$

NO se deben realizar cruces innecesarios si ya tengo el atributo en la tabla.

4 – SQL

Nuevamente, se tienen varias operaciones para realizar sobre las tablas.

- SELECT: se indica los nombres de los atributos a devolver, o se escribe * para no filtrar. Para eliminar tuplas repetidas, escribir DISTINCT previamente.
- FROM Tabla t1: tabla origen. Si se escriben varias: producto cartesiano.
- INNER JOIN Tabla2 t2 ON (t1.atributo = t2.atributo): producto natural.
- LEFT JOIN Tabla2 t2 ON (…): deja NULL en atributos de t2 si no hay.
- RIGHT JOIN Tabla2 t2 ON (…): deja NULL en atributos de t1 si no hay.
- WHERE: se indica la condición para filtrar las tuplas requeridas.
    - AND, OR y NOT: operadores lógicos para las condiciones.
    - Atributo BETWEEN valor1 and valor2: incluye los extremos.
    - Atributo LIKE “Ho%”: strings que empiecen con cierta cadena.
    - Atributo LIKE “%ura”: strings que terminen con cierta cadena.
    - Atributo LIKE “%abc%”: strings que contengan cierta cadena.
    - Atributo LIKE “ho_ _”: los “_” representan un carácter cualquiera.
    - Atributo IS NULL: si el atributo es nulo (no significa “” ni 0).
    - EXIST: devuelve verdadero si la subconsulta devolvió alguna tupla.
- GROUP BY: se indican los atributos para formar grupos (tuplas no repetidas), debiéndose indicar tanto la clave primaria como los atributos a devolver.
- HAVING: similar a WHERE pero aplica sobre cada grupo formado.
- ORDER BY: ordena las tuplas por atributos indicados. Por defecto es de manera ascendente (ASC), se debe escribir DESC al final para descendente.

Para aplicar sobre conjuntos tenemos las siguientes operaciones:

- UNION: devuelve las tuplas de 2 tablas de mismo esquema, sin repetidos. Si no se desea eliminar las tuplas repetidas se debe utilizar UNION ALL.
- INTERSECT: devuelve las tuplas repetidas de 2 tablas de mismo esquema.
- EXCEPT: realiza la diferencia Tabla 1 – Tabla 2, de mismo esquema.

Por otro lado, para realizar altas, bajas y modificaciones tenemos:

- INSERT INTO Tabla (atributo1, atributo2, …) VALUES (valor1, valor2, …)
- DELETE FROM Tabla WHERE (…): si no hay WHERE, borra toda la tabla.
- UPDATE Tabla SET atributo1 = valor1, atributo2 = valor2 WHERE (…)

4.1 – Mínimos y máximos

Para obtener el mínimo o máximo de un atributo, se utiliza la función de agregación MIN(atributo) o MAX(atributo) respectivamente.

En condiciones normales, devolverá el máximo o mínimo de todas las tuplas resultantes de la consulta. Sin embargo, si se aplica GROUP BY, devolverá un valor para cada grupo, debiéndose usar (en MIN o MAX) un atributo no usado para formar grupos.

Otra recomendación es renombrar la columna con AS para que sea legible.

4.2 – Conteo, promedio y suma

De manera similar para mínimos y máximos, existen las funciones COUNT, AVG y SUM, aplicables sobre un atributo. Rigen las mismas reglas bajo condiciones normales y formación de grupos. Sin embargo, resulta importante destacar que todas estas funciones no tienen en cuenta a los valores NULL, por ejemplo, si se realiza un COUNT sobre una columna tipo "?", si en todas las tuplas el valor es nulo devolverá cero.

También puede darse el caso de querer informar la cantidad de apariciones por grupo formado. Para ello, se deberá usar LEFT JOIN, ya que INNER JOIN únicamente deja las tuplas que hayan coincidido en ambas tablas.

Existe la posibilidad de usar COUNT(*), que cuenta también a las tuplas que tienen atributos nulos. Para obtener cantidad de nulos: COUNT(*) – COUNT(atributo).

4.3 – Resultados de subconsultas

Dentro de una consulta principal puede existir una subconsulta, que puede devolver 0, 1 o más tuplas. En caso de que se retornen tuplas de una única columna, se pueden utilizar los operadores IN (o su negación), SOME y ALL, todos dentro de una cláusula WHERE o HAVING. Ejemplo: WHERE atributo1 IN (SELECT …)

Para SOME y ALL se antepone un operador de comparación, como '=', '<', '<=', '>=' o '>'. En el caso de SOME se devuelve verdadero si alguno de los valores devueltos en la subconsulta lo cumple; y en ALL si todos, sin excepción, lo cumplen.

Si se está seguro que la subconsulta devuelve un único valor, se puede aplicar directamente el operador de comparación. Ejemplo: WHERE atributo1 = (SELECT …)