1. ***Identificación de Entidades:***

- Revisa el modelo E/R y lista todas las entidades que debes convertir en tablas. Cada entidad se

convertirá en una tabla en el modelo relacional.

2. ***Definición de Atributos:***

- Para cada entidad, identifica sus atributos y clasifícalos en:

- ***Atributos simples***: Se convierten directamente en columnas de la tabla.

- ***Atributos compuestos***: Divide en atributos simples y crea columnas correspondientes.

- ***Atributos multivaluados***: Crea una nueva tabla para almacenar estos atributos. La nueva tabla

tendrá una relación con la tabla de la entidad original.

3. ***Asignación de Claves:***

- Identifica la clave primaria de cada tabla:

- Selecciona un atributo o conjunto de atributos que sirva como identificador único (no puede ser

nulo).

- Asegúrate de que no se repita en otras filas.

- Determina si necesitas claves ajenas para mantener la integridad referencial entre tablas

relacionadas.

4***. Transformación de Relaciones***:

- Para cada relación entre entidades, determina su cardinalidad (1:1,1:N, N:M) y realiza lo

siguiente:

**Transformación de Relaciones**

**1. Relaciones Muchos a Muchos (N:M)**

 Se crea una **tabla intermedia**.

* **Claves primarias**: Esta tabla intermedia tendrá como clave primaria la combinación de las claves primarias de las dos tablas originales.
* **Claves foráneas**: Los atributos que forman parte de la clave primaria en la tabla intermedia son **claves foráneas** que referencian las claves primarias de las tablas originales.

 **No se mueven otros atributos** a la tabla intermedia.

**Ejemplo**:

* **AUTOR (Cod-Autor, ...)**
* **LIBRO (Cod-Libro, ...)**
* **ESCRIBE (Cod-Autor, Cod-Libro, ...)** (tabla intermedia)

**2. Relaciones Uno a Muchos (1:N)**

Existen **dos enfoques** para transformarlas:

* **Propagación del atributo principal**:
* Se lleva la clave primaria de la tabla con cardinalidad (1) a la tabla con cardinalidad (N). que seran Claves foráneas que referencia a la tabla cardinaliad 1
* Esta clave no permite valores nulos.
* **No eliminas otros atributos**: No es necesario borrar otros atributos de la tabla, pero la relación se simplifica.

**Ejemplo**:

* **LIBROS (Cod-Libro, Cod-editorial, ...)**, donde Cod-editorial no permite nulos.
* **Transformación a relación N**:
* Esto se recomienda en los siguientes casos:
* Si hay muchos nulos debido a pocos elementos relacionados.
* Si se prevé que la relación cambiará a N

en el futuro.

* Si la relación tiene **atributos propios**.

**3. Relaciones Uno a Uno (1:1)**

* Se puede aplicar cualquiera de las reglas anteriores, pero se recomiendan las siguientes consideraciones:
* Si las entidades tienen cardinalidades (0,1) (0,1), es mejor crear una relación para evitar muchos nulos.
* Si las entidades tienen cardinalidades (0,1) (1,1), es mejor propagar la clave de la entidad con cardinalidad (1,1) a la entidad (0,1). como clave foranea con referencia a la de 1,1
* Si las entidades tienen cardinalidades (1,1) (1,1), la propagación es indiferente. Se decide según la **frecuencia de acceso** (consultas, modificaciones).

**Propagación**

La **propagación** se refiere a la acción de llevar una clave de una tabla a otra para establecer una relación entre ellas, facilitando así la integridad referencial. Las decisiones sobre cómo propagar las claves dependen de la cardinalidad de las relaciones y de cómo se espera que evolucionen las tablas en el futuro.

**Integridad referencial**: La propagación ayuda a mantener la integridad de los datos y a simplificar las consultas, evitando redundancias innecesarias.

**Resumen**

* **M:N**

: Crear tabla intermedia, Pk(relaciones) =PK y claves foráneas en la nueva tabla.

* **1:n**

: Propagar clave de 1 a N o transformarla en N

en ciertos casos.

* **1:1**: Propagar clave según la cardinalidad y la posibilidad de nulos.

***Interrelaciones con Atributos:***

Si la interrelación incluye atributos, se transforma en una **tabla**. Esta tabla representará la relación y sus atributos como columnas de la nueva tabla.

La tabla resultante incluirá:

* **Primary key:** son las fk las dos fk forman la prmary key
* **Claves foráneas**: Las claves primarias de las entidades relacionadas (por ejemplo, Cod\_Estudiante y Cod\_Curso).
* **Atributos de la relación**: Los atributos que describen la relación (por ejemplo, Fecha\_Inscripción).

**Definición de Relaciones Reflexivas**

Una relación reflexiva ocurre cuando una entidad tiene una relación con otra instancia de la misma entidad. Por ejemplo, en un modelo de empleados, un empleado puede ser el jefe de otro empleado.

**Modelado en el Modelo Relacional DE REFLEXIVAS**

Para representar una relación reflexiva, puedes seguir estos pasos **2 CASO:**

**CASO 1:1**

**a. Tabla Original**

Tienes una tabla que representa la entidad. Por ejemplo, una tabla Empleados con atributos como:

* Cod\_Empleado (clave primaria)
* Nombre
* Puesto

**b. Claves Foráneas en la Misma Tabla**

* Para modelar la relación reflexiva, se agrega un nuevo atributo en la misma tabla que actúa como una clave foránea. Este atributo hará referencia a la clave primaria de la misma tabla.
* En el caso del ejemplo de empleados, puedes agregar un atributo como Jefe\_ID, que hace referencia a Cod\_Empleado.
*  Cod\_Empleado es la clave primaria.
*  Jefe\_ID es una clave foránea que se refiere a Cod\_Empleado. Esto permite establecer la jerarquía de empleados, donde Juan es el jefe de María, y María es la jefa de Carlos y Ana.
* puede ser NULL

***CASO 2 N:M***

* **N:M o N-aria**:
* Se crea una **nueva relación**.

**Relación recursiva M**

**(Empleado y Mentor)**

* **Entidad**: Empleado.
* **Relación**: Un empleado puede ser mentor de varios empleados, y un empleado puede tener varios mentores (M

).

Aquí se debe crear una **nueva tabla** para representar la relación de mentoría.

**Nueva tabla**:

|  |  |
| --- | --- |
| **ID\_Mentor** | **ID\_Empleado** |
| E001 | E002 |
| E001 | E003 |
| E002 | E003 |

En este caso, **Juan** (E001) es mentor de **María** y **Carlos**. **María** (E002) también es mentora de **Carlos**.

**Crear una Tabla Separada para el Atributo Multivaluado**

Cuando encuentras un atributo multivaluado, como los **números de teléfono** de un **estudiante** (un estudiante puede tener varios números de teléfono), la solución correcta es **crear una nueva tabla**.

Esta nueva tabla contendrá :

* La **clave primaria** de la entidad original (en este caso, Estudiante).**ACTUA COMO FORANEA**
* DOS OPCIONES PK:
* Clave primaria compuesta: Combina la clave primaria de la entidad original con el atributo multivaluado en la nueva tabla.Cada estudiante tenga teléfonos únicos asociados.Evitar duplicados
* Clave primaria independiente: Crear un nuevo campo (por ejemplo, ID\_Telefono) como clave primaria de la tabla de atributos multivaluados.Quieres un identificador único y autónomo para cada registro en la tabla de teléfonos.Necesitas flexibilidad para el mismo número de teléfono que podría estar asociado a múltiples estudiante
* El atributo multivaluado (Teléfono).
* ejemplo::En la tabla **Teléfonos**, Cod\_Estudiante actúa como una **clave foránea** que hace referencia a la tabla **Estudiantes**.La clave primaria en la tabla **Teléfonos** podría ser una combinación de Cod\_Estudiante y Teléfono, o simplemente un nuevo campo si se necesita una clave única.
* fk=pk entida orignal con referencia a la talab orignal

Esto convierte la relación **1 a muchos** (un estudiante puede tener varios números de teléfono) en una relación normalizada.

**Ejemplo:**

Supongamos que tienes la entidad **Estudiante** con un atributo multivaluado Teléfono.

**Antes (Entidad Estudiante con Atributo Multivaluado):**

|  |  |  |
| --- | --- | --- |
| **Cod\_Estudiante** | **Nombre** | **Teléfonos** |
| 1 | Juan | 123456, 789012 |
| 2 | María | 345678 |
| 3 | Carlos | 987654, 321098, 567890 |

**Después (Entidad Normalizada):**

**Tabla Estudiantes:**

|  |  |
| --- | --- |
| **Cod\_Estudiante** | **Nombre** |
| 1 | Juan |
| 2 | María |
| 3 | Carlos |

**Tabla Teléfonos (Nueva tabla):**

|  |  |
| --- | --- |
| **Cod\_Estudiante** | **Teléfono** |
| 1 | 123456 |
| 1 | 789012 |
| 2 | 345678 |
| 3 | 987654 |
| 3 | 321098 |
| 3 | 567890 |

**Diseño lógico: Transformación de interrelaciones N-arias**

**Interrelación N-aria**

**Una interrelación N-aria implica que participan más de dos entidades en una relación. La N significa que pueden ser 3, 4 o más entidades. Cuando tienes una interrelación N-aria, en el modelo relacional se debe crear una nueva tabla**

* **Transformación al modelo relacional**:
* Se crea una **nueva relación**.
* **Atributos** de la relación = claves primarias de todas las entidades interrelacionadas + atributos de la interrelación.

**Casos:**

* **Todas las entidades están conectadas con “muchos”**:
* **Clave primaria** = será la **combinación de las claves** de todas las entidades que participan.
* **Alguna entidad conectada con “uno”**:
* **Clave primaria** = todos los atributos de las claves, menos una entidad conectada con "uno".

**Diseño lógico: Transformación de entidades débiles**

**Entidad débil**

* **Transformación**:

Se aplica **actualización y borrado en cascada**.

**Entidad débil con dependencia en identificación**

* **Clave primaria** = combinación de la clave de la entidad fuerte + clave propia de la entidad débil.
* **CLAVE FORANEA**=PK DE LA ENTIDAD FUERTE CON REFERENCIA ALA ENTIDAD FUERTE

**Diseño lógico: Transformación de generalización/especialización**

**Generalización/especialización**

* **Transformación**:
* Se crea una **relación para la superclase** y otra para cada subclase.
* **Clave primaria** de la superclase pasa a ser la **clave primaria** de la relación subclase. Y ASU VEZ FORANEA CON REFERENCIA A LA ENTIDAD SUPERCLASE
* **Transformación al Modelo Relacional**: Hay tres enfoques comunes para modelar jerarquías de generalización/especialización en el modelo relacional:
* **a. Enfoque de Tabla Única (Tabla para el Supertipo)**
* **Descripción**: Se utiliza una sola tabla para la entidad general y todas las entidades específicas.
* **Implementación**:
* Crear una sola tabla que incluya **todos los atributos del supertipo y de los subtipos**.
* Añadir una columna de **discriminador** para identificar el subtipo de cada registro.
* **Ventaja**: Simple y eficiente en accesos.
* **Desventaja**: Puede haber muchos nulos si los subtipos tienen atributos muy diferentes.

**Ejemplo**: Una tabla Empleado con columnas adicionales para Especializacion (solo relevante para Ingeniero) y Depto\_Gestion (solo para Gerente).

* **b. Enfoque de Tablas Separadas para Cada Subtipo**
* **Descripción**: Crear una tabla separada para el supertipo y cada uno de los subtipos.
* **Implementación**:
* Crear una tabla para el **supertipo** que contenga su clave primaria y atributos comunes.
* Crear una **tabla para cada subtipo**, que contenga su clave primaria (que actúa también como clave foránea hacia el supertipo) y sus atributos específicos.
* **Ventaja**: No hay valores nulos.
* **Desventaja**: Requiere JOINs para consultar datos completos de cada subtipo.
* **Ejemplo**: Una tabla Empleado y, por separado, tablas Ingeniero y Gerente, cada una referenciando la clave primaria de Empleado.
* **c. Enfoque de Tabla Única para Cada Subtipo**
* **Descripción**: Crear solo una tabla para cada subtipo sin tener una tabla para el supertipo.
* **Implementación**:
* Crear una **tabla para cada subtipo** que incluya tanto los atributos comunes del supertipo como los atributos específicos.
* **Ventaja**: Consulta directa sin necesidad de JOINs.
* **Desventaja**: Duplica los atributos comunes del supertipo, lo que incrementa la redundancia.
* **Ejemplo**: Crear tablas Ingeniero y Gerente con sus atributos específicos y los atributos comunes de Empleado.

**Diseño lógico: Resumen de transformación del modelo E/R al modelo relacional**

* **Entidad** → **Relación**.
* **Interrelación 1:1** → **Clave externa**.
* **Interrelación 1**

→ **Clave externa**.

* **Interrelación M**

→ **Nueva relación**.

* **Interrelación N-aria** → **Nueva relación**.
* **Interrelación recursiva** → Como una relación no recursiva.
* **Entidad débil** → Actualización y borrado en cascada (clave foránea en la clave primaria).
* **Generalización/especialización** → Relación para superclase y para subclase.

**Restricciones externas en el modelo relacional**

* **Relación 1**

**con participación total** → Nueva restricción externa.

* **Participación total en relación N**

→ Nueva restricción externa.

* **Relación 1:1 con participación total** → Nueva restricción externa.

5. ***Definición de Dominios:***

- Establece el tipo de datos para cada columna según el dominio que cada atributo puede tomar

(por ejemplo, enteros, cadenas, fechas).

- Define restricciones como NOT NULL, UNIQUE, etc.

6. ***Implementación de Restricciones***:

- Define las restricciones de integridad para cada tabla:

- Claves primarias y ajenas.

- Restricciones de unicidad y no nulos.

- Otras restricciones de dominio y validación (por ejemplo, CHECK).

7. ***Diseño de Vistas (Opcional):***

- Considera crear vistas para proporcionar un acceso simplificado o seguro a los datos. Las vistas

pueden ayudar a ocultar la complejidad de la estructura de las tablas.

8. ***Documentación:***

- Documenta cada tabla, incluyendo su nombre, descripción, atributos, tipos de datos, claves y

restricciones para futuras referencias.

***Ejemplo Práctico:***

Imagina que tienes las siguientes entidades y relaciones:

- ***Entidad***: Empleado (Atributos: ID, Nombre, Apellido, Departamento)

- ***Entidad:*** Departamento (Atributos: ID, Nombre)

***- Relación***: Un empleado pertenece a un departamento (1:N).

Proceso:

1. ***Tablas:***

- Crear tabla 'Empleado' con columnas: 'ID' (PK), 'Nombre', 'Apellido', 'Departamento\_ID' (FK).

- Crear tabla 'Departamento' con columnas: 'ID' (PK), 'Nombre'.

2. Claves:

- Clave primaria en 'Empleado' es 'ID'.

- Clave primaria en 'Departamento' es 'ID'.

- 'Departamento\_ID' en 'Empleado' se convierte en clave ajena que referencia a 'ID' en

'Departamento'.

3. ***Restricciones:***

- Asegúrate de que 'Departamento\_ID' no sea nulo en 'Empleado'.