Desarrollo de una Base de Datos para Instrumentos Públicos Notariales. Enfoque Detallado

Contenido

[**FASE 1: ANÁLISIS DE REQUISITOS Y MODELADO CONCEPTUAL** 2](#_Toc200021433)

[**1. Introducción** 2](#_Toc200021434)

[**1.1 Comprensión del Dominio Notarial (solo personas físicas)** 2](#_Toc200021435)

[**1.1.1 Contexto funcional** 2](#_Toc200021436)

[**1.1.2 Objetivos del sistema** 2](#_Toc200021437)

[**1.1.3 Supuestos y exclusiones** 3](#_Toc200021438)

[**1.2 Identificación de Entidades Principales** 3](#_Toc200021439)

[**1.2.1 Criterios de selección** 3](#_Toc200021440)

[**1.2.2 Entidades seleccionadas** 3](#_Toc200021441)

[**1.3 Diagrama Entidad-Relación (ER)** 4](#_Toc200021442)

[**FASE 2: DISEÑO LÓGICO DE LA BASE DE DATOS** 19](#_Toc200021443)

[**2.1 Transformación del Modelo ER a Esquema Relacional** 19](#_Toc200021444)

[**2.2 Definición Detallada de Tablas** 20](#_Toc200021445)

[**2.3 Normalización de la Base de Datos** 24](#_Toc200021446)

[**2.4 Integridad Referencial** 25](#_Toc200021447)

[**FASE 3: IMPLEMENTACIÓN FÍSICA** 29](#_Toc200021448)

[**3.1 ELECCIÓN DEL SISTEMA GESTOR DE BASE DE DATOS (SGBD)** 29](#_Toc200021449)

[**3.2 SCRIPT DE CREACIÓN COMPLETO** 30](#_Toc200021450)

[**3.3 ÍNDICES Y OPTIMIZACIÓN** 35](#_Toc200021451)

[**3.4 VISTAS PARA CONSULTAS COMUNES** 36](#_Toc200021452)

[**3.5 PROCEDIMIENTOS ALMACENADOS** 38](#_Toc200021453)

**FASE 1: ANÁLISIS DE REQUISITOS Y MODELADO CONCEPTUAL**

**Proyecto: Base de Datos Notarial para Personas Físicas – Formularios 1-A y 1-B**

**1. Introducción**

El proceso de análisis de requisitos y modelado conceptual constituye el pilar sobre el cual se fundamenta cualquier sistema de bases de datos que aspire a ser robusto, escalable y coherente con el dominio que pretende modelar. En el contexto notarial español, las actuaciones que implican la emisión de instrumentos públicos requieren una captación rigurosa, verificable y estructurada de la información proporcionada por los comparecientes u **otorgantes**.

En esta fase del diseño se parte de los formularios oficiales 1-A y 1-B, utilizados por las notarías para recoger datos de personas físicas, tanto nacionales como extranjeras, en el proceso previo a la redacción y autorización de una escritura pública. A partir del análisis de estos formularios, se construirá un modelo conceptual que permita representar de manera fiel y exhaustiva la realidad que pretenden reflejar, teniendo en cuenta las limitaciones del marco legal, las necesidades operativas del entorno notarial y los principios de normalización de bases de datos relacionales.

Este documento desarrolla la fase 1 en profundidad, prescindiendo expresamente de la figura de la persona jurídica. Por tanto, el análisis se focaliza exclusivamente en personas físicas, en coherencia con la funcionalidad real de los formularios analizados y con el alcance limitado del presente proyecto.

**1.1 Comprensión del Dominio Notarial (solo personas físicas)**

**1.1.1 Contexto funcional**

En el ámbito notarial, todo acto público tiene como punto de partida la identificación y caracterización de los comparecientes. Estos son, en la mayoría de los casos, personas físicas que acuden a una notaría con el objetivo de otorgar una declaración de voluntad con efectos legales. Dado que la escritura pública es un documento jurídico de carácter formal y solemne, la exactitud de los datos personales es esencial, así como su validez jurídica.

El notario, como fedatario público, debe garantizar la veracidad de los datos, la identidad de los comparecientes, su capacidad para actuar, su estado civil y, en caso de representación, la existencia y vigencia del poder que lo autoriza. Esta obligación convierte el proceso de recogida de datos en una actividad reglada, repetitiva y jurídicamente relevante, lo cual justifica su sistematización en una base de datos bien diseñada.

**1.1.2 Objetivos del sistema**

El sistema a diseñar deberá permitir:

* Registrar datos completos y consistentes de personas físicas (nacionales y extranjeras).
* Asociar a cada persona sus documentos de identidad válidos.
* Identificar su domicilio actual (en España o en el extranjero).
* Indicar si actúan por sí mismos o en representación de otra persona.
* Registrar su estado civil y, si procede, su régimen económico matrimonial.
* Verificar automáticamente la mayoría de edad y profesiones consideradas de riesgo.
* Gestionar los formularios en sus versiones 1-A (nacionales) y 1-B (extranjeros).

**1.1.3 Supuestos y exclusiones**

Este proyecto:

* **Incluye**: Personas físicas mayores de edad, con nacionalidad española o extranjera.
* **Excluye**: Personas jurídicas (empresas, fundaciones), menores de edad, casos de doble nacionalidad con efectos legales complejos.
* **Asume**: Que todos los otorgantes cuentan con documentación válida (NIF, NIE, pasaporte).

**1.2 Identificación de Entidades Principales**

**1.2.1 Criterios de selección**

Una entidad debe representar un concepto autónomo del mundo real que tenga una existencia propia y que deba ser almacenado como tabla. En el contexto de los formularios notariales, estos conceptos se extraen directamente del análisis semántico del contenido de los formularios 1-A y 1-B.

**1.2.2 Entidades seleccionadas**

A continuación, se presentan las entidades seleccionadas, junto con una justificación para su inclusión:

1. **PERSONA\_FISICA**
   * Representa a cada otorgante individual. Incluye tanto españoles como extranjeros.
   * Se justifica su existencia como entidad independiente por su carácter central y por contener atributos únicos no compartidos con otras entidades (nombre, apellidos, fecha de nacimiento, profesión).
2. **DOCUMENTO\_IDENTIDAD**
   * Representa cada identificación oficial válida para la persona física (NIF, NIE, Pasaporte).
   * Su separación como entidad permite asociar múltiples documentos a una persona y facilitar validaciones cruzadas por número, tipo o formato.
3. **DIRECCION**
   * Representa el lugar de residencia habitual de la persona física.
   * Es normalizada como tabla separada para gestionar múltiples direcciones (domicilio habitual, dirección en el extranjero).
4. **FORMULARIO**
   * Registra cada proceso de recopilación de datos mediante un formulario (1-A o 1-B).
   * Permite historificar entradas y diferenciar tipos de otorgantes.
5. **REPRESENTACION**
   * Describe el vínculo jurídico mediante el cual una persona actúa en nombre de otra.
   * Esencial para aquellos casos en los que el otorgante no actúa por sí mismo.
6. **PODER**
   * Documento notarial que fundamenta la representación.
   * Puede tener múltiples campos: fecha, tipo (general/especial), archivo escaneado.
7. **ESTADO\_CIVIL**
   * Representa el estado jurídico respecto al matrimonio.
   * Se modela como entidad para permitir selección controlada (soltero, casado, separado, viudo).
8. **MATRIMONIO**
   * Relación específica entre dos personas físicas casadas.
   * Permite almacenar tanto el cónyuge como la fecha del matrimonio y su régimen.
9. **REGIMEN\_MATRIMONIAL**
   * Define el marco legal de los bienes durante el matrimonio.
   * Se mantiene como tabla para permitir parametrización, control y posibles ampliaciones normativas.

**1.3 Diagrama Entidad-Relación (ER)**

A continuación, se presenta el diagrama conceptual con las entidades principales y sus relaciones.

Изображение выглядит как текст, диаграмма, линия, План

Содержимое, созданное искусственным интеллектом, может быть неверным.

https://www.mermaidchart.com/app/projects/f9a08323-1fa4-4747-b6d2-7f2261e7eeaf/diagrams/f99d46ac-2dcf-4889-bce5-1c7e5c8aee54/version/v0.1/edit

erDiagram

PERSONA\_FISICA ||--o{ DIRECCION : tiene

PERSONA\_FISICA ||--o{ DOCUMENTO\_IDENTIDAD : posee

PERSONA\_FISICA ||--|| ESTADO\_CIVIL : tiene

PERSONA\_FISICA ||--o{ MATRIMONIO : contrae

MATRIMONIO ||--|| REGIMEN\_MATRIMONIAL : sujeto\_a

REPRESENTACION ||--|| PERSONA\_FISICA : representante

REPRESENTACION ||--|| PERSONA\_FISICA : representado

REPRESENTACION ||--|| PODER : fundamenta

FORMULARIO ||--o{ PERSONA\_FISICA : otorgante

Este diagrama refleja la realidad notarial de forma normalizada y permite la implementación posterior sin redundancias ni ambigüedades.

**1.4 Atributos Clave por Entidad**

**PERSONA\_FISICA**

Entidad central del modelo. Representa a cada individuo que comparece ante el notario.

| **Atributo** | **Tipo de dato** | **Restricciones** | **Justificación** |
| --- | --- | --- | --- |
| id | BIGINT | PK, autoincremental | Identificador único de la persona |
| nombre | VARCHAR(50) | NOT NULL | Campo obligatorio según normativa |
| primer\_apellido | VARCHAR(50) | NOT NULL | Primer apellido obligatorio en el formulario |
| segundo\_apellido | VARCHAR(50) | NULLABLE | Admite NULL (personas extranjeras pueden tener solo uno) |
| fecha\_nacimiento | DATE | NOT NULL, CHECK (>=1900, <= fecha actual) | Verifica la mayoría de edad (> 18 años) |
| profesion | VARCHAR(100) | NULLABLE | Campo útil para control de profesiones de riesgo |
| profesion\_riesgo | BOOLEAN | DEFAULT FALSE | Valor lógico útil para prevalidación documental |

Nota: Se considera que toda persona tiene un solo id que la identifica en el sistema, pero puede tener múltiples relaciones con otras entidades como dirección, documentos, matrimonios, etc.

**DOCUMENTO\_IDENTIDAD**

Representa un documento legal válido. Una persona puede tener varios (NIF, NIE, pasaporte).

| **Atributo** | **Tipo de dato** | **Restricciones** | **Justificación** |
| --- | --- | --- | --- |
| id\_documento | BIGINT | PK, autoincremental | Identificador único de documento |
| persona\_id | BIGINT | FK a PERSONA\_FISICA(id), NOT NULL | Asociación con la persona |
| tipo\_documento | ENUM('NIF','NIE','PASAPORTE') | NOT NULL | Control sobre tipos aceptados legalmente |
| numero | VARCHAR(20) | NOT NULL, UNIQUE | Verificación de unicidad en el sistema |
| fecha\_expedicion | DATE | NOT NULL | Permite validación temporal del documento |

La separación en una tabla propia permite manejar múltiples documentos por persona, validaciones de formato y futuras integraciones con APIs oficiales.

**DIRECCION**

Tabla normalizada que permite registrar múltiples direcciones por persona. Es obligatoria al menos una (domicilio habitual).

| **Atributo** | **Tipo de dato** | **Restricciones** | **Justificación** |
| --- | --- | --- | --- |
| id\_direccion | BIGINT | PK, autoincremental | Identificador único de dirección |
| persona\_id | BIGINT | FK a PERSONA\_FISICA(id), NOT NULL | Asociación con la persona física |
| tipo\_via | VARCHAR(20) | NOT NULL | Estandariza el tipo de vía pública |
| nombre\_via | VARCHAR(100) | NOT NULL | Nombre de la vía |
| numero | VARCHAR(10) | NOT NULL | Número del portal o edificio |
| escalera | VARCHAR(10) | NULLABLE | Opcional |
| piso | VARCHAR(10) | NULLABLE | Opcional |
| puerta | VARCHAR(10) | NULLABLE | Opcional |
| municipio | VARCHAR(50) | NOT NULL | Localidad obligatoria |
| provincia | VARCHAR(50) | NOT NULL | Provincia obligatoria |
| codigo\_postal | CHAR(5) | NOT NULL, CHECK (LENGTH=5) | Validación estándar española |
| pais | VARCHAR(50) | NOT NULL si formulario es tipo 1-B | Obligatorio en casos de dirección extranjera |

El diseño flexible permite gestionar tanto direcciones nacionales como extranjeras.

**ESTADO\_CIVIL**

Tabla paramétrica. Representa estados válidos según derecho civil español.

| **Atributo** | **Tipo de dato** | **Restricciones** | **Justificación** |
| --- | --- | --- | --- |
| id | TINYINT | PK | Código del estado civil |
| descripcion | VARCHAR(30) | CHECK (IN ENUM) | Control de dominio (ej. 'SOLTERO', 'CASADO', 'VIUDO') |

Se asocia directamente con la persona física mediante una FK.

**MATRIMONIO**

Permite almacenar la existencia de un matrimonio entre dos personas físicas, así como su régimen y fecha.

| **Atributo** | **Tipo de dato** | **Restricciones** | **Justificación** |
| --- | --- | --- | --- |
| id\_matrimonio | BIGINT | PK | Identificador único |
| persona\_id | BIGINT | FK a PERSONA\_FISICA(id) | Uno de los cónyuges |
| conyuge\_id | BIGINT | FK a PERSONA\_FISICA(id), CHECK (≠ persona\_id) | El otro cónyuge |
| fecha\_matrimonio | DATE | NOT NULL | Registro de la fecha oficial |
| regimen\_id | BIGINT | FK a REGIMEN\_MATRIMONIAL(id) | Relación con tipo de régimen económico |

Cada registro representa una unión matrimonial concreta. Se puede ampliar en el futuro con un campo de fecha de disolución.

**REGIMEN\_MATRIMONIAL**

Tabla de referencia para definir los tipos de regímenes patrimoniales.

| **Atributo** | **Tipo de dato** | **Restricciones** | **Justificación** |
| --- | --- | --- | --- |
| id | TINYINT | PK | Identificador único |
| nombre | ENUM | CHECK(nombre IN (...)) | Control sobre las opciones válidas legales |

Opciones:

* 'SOCIEDAD\_GANANCIALES'
* 'SEPARACION\_BIENES'

**REPRESENTACION**

Relación entre dos personas físicas: una actúa en nombre de la otra. Se puede basar en un poder.

| **Atributo** | **Tipo de dato** | **Restricciones** | **Justificación** |
| --- | --- | --- | --- |
| id | BIGINT | PK | Clave primaria |
| representante\_id | BIGINT | FK a PERSONA\_FISICA(id) | Persona que actúa |
| representado\_id | BIGINT | FK a PERSONA\_FISICA(id), CHECK(≠ representante\_id) | Persona en cuyo nombre se actúa |
| tipo | ENUM | ('LEGAL','NOTARIAL') | Define el tipo de representación |
| fecha\_inicio | DATE | NOT NULL | Fecha desde que tiene efecto |
| fecha\_fin | DATE | NULLABLE | Fecha de caducidad o disolución |
| poder\_id | BIGINT | FK a PODER(id), NULLABLE | El documento que valida la representación |

**PODER**

Documento notarial que legitima la representación.

| **Atributo** | **Tipo de dato** | **Restricciones** | **Justificación** |
| --- | --- | --- | --- |
| id | BIGINT | PK | Identificador único del poder |
| tipo | ENUM | ('GENERAL','ESPECIAL') | Diferencia entre facultades plenas o limitadas |
| documento\_digital | VARCHAR(255) | Ruta o enlace | Permite almacenamiento digital |
| fecha\_otorgamiento | DATE | Fecha en la que se emitió | Obligatorio para verificar vigencia |

**FORMULARIO**

Representa el formulario electrónico rellenado por el notariado.

| **Atributo** | **Tipo de dato** | **Restricciones** | **Justificación** |
| --- | --- | --- | --- |
| id\_formulario | BIGINT | PK | Registro único del formulario |
| tipo | ENUM | ('1-A','1-B') | Diferencia entre nacionales y extranjeros |
| fecha\_creacion | TIMESTAMP | DEFAULT CURRENT\_TIMESTAMP | Fecha de generación |
| estado | ENUM | ('BORRADOR','VALIDADO','ARCHIVADO') | Seguimiento del flujo de validación notarial |

**1.5 Relaciones Principales**

**Introducción**

En el diseño de una base de datos conceptual, las relaciones entre entidades reflejan la forma en que los objetos del dominio se conectan e interactúan entre sí. No basta con identificar las entidades de manera aislada; es necesario modelar de forma precisa los vínculos que existen entre ellas, su sentido, su obligatoriedad y su multiplicidad. A continuación, se presentan las relaciones seleccionadas, explicadas una por una.

**1.5.1 Relación: PERSONA\_FISICA – DIRECCION**

**Nombre de la relación:** reside\_en  
**Tipo:** 1:N (una persona puede tener varias direcciones)  
**Obligatoriedad:** Mínimo una dirección requerida

**Justificación:**

* En el formulario 1-A, se requiere obligatoriamente una dirección de residencia en España.
* En el formulario 1-B, puede haber dirección en España y/o en el extranjero.
* Algunas personas pueden tener más de un domicilio (residencia habitual, lugar de notificación, segunda residencia).

**1.5.2 Relación: PERSONA\_FISICA – DOCUMENTO\_IDENTIDAD**

**Nombre de la relación:** identificado\_por  
**Tipo:** 1:N  
**Justificación:**

* Toda persona debe tener al menos un documento válido.
* Extranjeros pueden tener NIE o pasaporte; españoles, el NIF.
* El sistema permite mantener documentos anteriores por motivos históricos o de verificación múltiple.

**1.5.3 Relación: PERSONA\_FISICA – ESTADO\_CIVIL**

**Nombre de la relación:** tiene\_estado\_civil  
**Tipo:** 1:1  
**Obligatoriedad:** siempre presente

**Justificación:**

* El estado civil afecta directamente la capacidad legal y la aplicación de régimen matrimonial.
* Se selecciona de un conjunto limitado de valores legales.

**1.5.4 Relación: PERSONA\_FISICA – MATRIMONIO**

**Nombre de la relación:** contrae  
**Tipo:** 1:N (una persona puede haberse casado varias veces)  
**Justificación:**

* El modelo considera el historial matrimonial (vida conyugal puede incluir varios matrimonios).
* Permite referenciar diferentes cónyuges en distintos momentos del tiempo.

**1.5.5 Relación: MATRIMONIO – REGIMEN\_MATRIMONIAL**

**Nombre de la relación:** sujeto\_a  
**Tipo:** N:1  
**Justificación:**

* Cada matrimonio se celebra bajo un régimen económico concreto.
* Un régimen puede ser compartido por muchos matrimonios.

**1.5.6 Relación: PERSONA\_FISICA – REPRESENTACION**

**Nombre de la relación:** actúa\_como  
**Tipo:** N:N  
**Justificación:**

* Una persona puede representar a otras, y a su vez, ser representada.
* La relación se formaliza mediante la tabla REPRESENTACION con atributos propios (tipo, fechas).

**1.5.7 Relación: REPRESENTACION – PODER**

**Nombre de la relación:** fundamentada\_en  
**Tipo:** N:1  
**Justificación:**

* Una representación válida debe estar basada en un documento notarial (poder).
* Un poder puede ser reutilizado mientras esté vigente.

**1.5.8 Relación: FORMULARIO – PERSONA\_FISICA**

**Nombre de la relación:** contiene\_otorgante  
**Tipo:** 1:N  
**Justificación:**

* Un formulario se asocia a una o más personas físicas (copartícipes en una escritura).
* Una persona puede figurar como otorgante en múltiples formularios a lo largo del tiempo.

**Consideraciones adicionales**

* Todas las relaciones están diseñadas bajo los principios de integridad referencial.
* Se busca siempre la **claridad semántica**: cada relación tiene un nombre explícito y un sentido jurídico.
* La cardinalidad se adapta tanto a la norma legal como a la realidad operativa del entorno notarial.

**1.5 Relaciones Principales**

**Introducción**

En el diseño de una base de datos conceptual, las relaciones entre entidades reflejan la forma en que los objetos del dominio se conectan e interactúan entre sí. No basta con identificar las entidades de manera aislada; es necesario modelar de forma precisa los vínculos que existen entre ellas, su sentido, su obligatoriedad y su multiplicidad. A continuación se presentan las relaciones seleccionadas, explicadas una por una.

**1.5.1 Relación: PERSONA\_FISICA – DIRECCION**

**Nombre de la relación:** reside\_en  
**Tipo:** 1:N (una persona puede tener varias direcciones)  
**Obligatoriedad:** Mínimo una dirección requerida

**Justificación:**

* En el formulario 1-A, se requiere obligatoriamente una dirección de residencia en España.
* En el formulario 1-B, puede haber dirección en España y/o en el extranjero.
* Algunas personas pueden tener más de un domicilio (residencia habitual, lugar de notificación, segunda residencia).

**1.5.2 Relación: PERSONA\_FISICA – DOCUMENTO\_IDENTIDAD**

**Nombre de la relación:** identificado\_por  
**Tipo:** 1:N  
**Justificación:**

* Toda persona debe tener al menos un documento válido.
* Extranjeros pueden tener NIE o pasaporte; españoles, el NIF.
* El sistema permite mantener documentos anteriores por motivos históricos o de verificación múltiple.

**1.5.3 Relación: PERSONA\_FISICA – ESTADO\_CIVIL**

**Nombre de la relación:** tiene\_estado\_civil  
**Tipo:** 1:1  
**Obligatoriedad:** siempre presente

**Justificación:**

* El estado civil afecta directamente la capacidad legal y la aplicación de régimen matrimonial.
* Se selecciona de un conjunto limitado de valores legales.

**1.5.4 Relación: PERSONA\_FISICA – MATRIMONIO**

**Nombre de la relación:** contrae  
**Tipo:** 1:N (una persona puede haberse casado varias veces)  
**Justificación:**

* El modelo considera el historial matrimonial (vida conyugal puede incluir varios matrimonios).
* Permite referenciar diferentes cónyuges en distintos momentos del tiempo.

**1.5.5 Relación: MATRIMONIO – REGIMEN\_MATRIMONIAL**

**Nombre de la relación:** sujeto\_a  
**Tipo:** N:1  
**Justificación:**

* Cada matrimonio se celebra bajo un régimen económico concreto.
* Un régimen puede ser compartido por muchos matrimonios.

**1.5.6 Relación: PERSONA\_FISICA – REPRESENTACION**

**Nombre de la relación:** actúa\_como  
**Tipo:** N:N  
**Justificación:**

* Una persona puede representar a otras, y a su vez, ser representada.
* La relación se formaliza mediante la tabla REPRESENTACION con atributos propios (tipo, fechas).

**1.5.7 Relación: REPRESENTACION – PODER**

**Nombre de la relación:** fundamentada\_en  
**Tipo:** N:1  
**Justificación:**

* Una representación válida debe estar basada en un documento notarial (poder).
* Un poder puede ser reutilizado mientras esté vigente.

**1.5.8 Relación: FORMULARIO – PERSONA\_FISICA**

**Nombre de la relación:** contiene\_otorgante  
**Tipo:** 1:N  
**Justificación:**

* Un formulario se asocia a una o más personas físicas (copartícipes en una escritura).
* Una persona puede figurar como otorgante en múltiples formularios a lo largo del tiempo.

**Consideraciones adicionales**

* Todas las relaciones están diseñadas bajo los principios de integridad referencial.
* Se busca siempre la **claridad semántica**: cada relación tiene un nombre explícito y un sentido jurídico.
* La cardinalidad se adapta tanto a la norma legal como a la realidad operativa del entorno notarial.

**1.6 Proceso de Normalización del Modelo Conceptual**

**1.6.1 Introducción a la normalización**

La normalización es el proceso de descomposición de estructuras complejas en estructuras más simples, con el objetivo de:

* Evitar duplicación de datos.
* Eliminar dependencias no deseadas.
* Facilitar la integridad de los datos.

El modelo que hemos diseñado ha sido sometido a una revisión formal basada en las tres primeras formas normales (1NF, 2NF y 3NF), ya que éstas son suficientes para la gran mayoría de los sistemas OLTP como el notarial.

**1.6.2 Primera Forma Normal (1NF)**

**Criterio**:  
Toda tabla debe tener atributos atómicos, sin grupos repetidos ni campos multivaluados.

**Aplicación en nuestro modelo**:

| **Tabla** | **Corrección aplicada** | **Resultado** |
| --- | --- | --- |
| DIRECCION | Cada dirección en una fila única | ✓ |
| DOCUMENTO\_IDENTIDAD | Cada documento en una fila separada | ✓ |
| MATRIMONIO | Relación entre dos personas separada | ✓ |
| REPRESENTACION | Se modela con su propia tabla | ✓ |

**Conclusión**: Todas las entidades cumplen 1NF.

**1.6.3 Segunda Forma Normal (2NF)**

**Criterio**:  
Todo atributo no clave debe depender de toda la clave primaria y no solo de parte de ella. Aplicable solo si la clave primaria es compuesta.

**Revisión**:

| **Entidad** | **¿Clave compuesta?** | **¿Cumple 2NF?** | **Justificación** |
| --- | --- | --- | --- |
| PERSONA\_FISICA | No | ✓ | Todos los atributos dependen de la PK simple (id) |
| DIRECCION | No | ✓ | Dependen de id\_direccion |
| DOCUMENTO\_IDENTIDAD | No | ✓ | Idem |
| REPRESENTACION | No | ✓ | Atributos como fecha\_inicio dependen de id |
| MATRIMONIO | No | ✓ | Atributos dependen de clave primaria id\_matrimonio |

**Conclusión**: Ninguna tabla tiene claves compuestas activas, por tanto, se cumple 2NF automáticamente.

**1.6.4 Tercera Forma Normal (3NF)**

**Criterio**:  
No debe haber dependencias transitivas: es decir, ningún atributo no clave debe depender de otro atributo no clave.

**Evaluación por tabla**:

| **Tabla** | **Dependencia transitiva detectada** | **Medida adoptada** |
| --- | --- | --- |
| PERSONA\_FISICA | No | — |
| MATRIMONIO | No | regimen\_id apunta a tabla externa (independiente) |
| FORMULARIO | No | tipo y estado son atómicos |
| DOCUMENTO\_IDENTIDAD | No | Validación por tipo y número única |

**Conclusión**: Todas las entidades cumplen 3NF sin necesidad de reestructuración adicional.

**1.7 Modelo Conceptual Final Integrado**

Una vez garantizada la correcta normalización, podemos representar el modelo conceptual completo, incluyendo todas las entidades, relaciones y cardinalidades de forma consolidada.

**Diagrama Mermaid del Modelo Final**

**Изображение выглядит как текст, диаграмма, План, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.**

[**https://www.mermaidchart.com/app/projects/f9a08323-1fa4-4747-b6d2-7f2261e7eeaf/diagrams/2afb7821-2093-4e56-b34e-aadae568e0aa/version/v0.1/edit**](https://www.mermaidchart.com/app/projects/f9a08323-1fa4-4747-b6d2-7f2261e7eeaf/diagrams/2afb7821-2093-4e56-b34e-aadae568e0aa/version/v0.1/edit)

erDiagram

    PERSONA\_FISICA {

        BIGINT id\_persona PK

        VARCHAR nombre

        VARCHAR primer\_apellido

        VARCHAR segundo\_apellido

        DATE fecha\_nacimiento

        VARCHAR profesion

        BOOLEAN profesion\_riesgo

        TINYINT estado\_civil\_id FK

    }

    DIRECCION {

        BIGINT id\_direccion PK

        BIGINT persona\_id FK

        VARCHAR tipo\_via

        VARCHAR nombre\_via

        VARCHAR numero

        VARCHAR piso

        VARCHAR puerta

        VARCHAR escalera

        VARCHAR municipio

        VARCHAR provincia

        CHAR codigo\_postal

        VARCHAR pais

    }

    DOCUMENTO\_IDENTIDAD {

        BIGINT id\_documento PK

        BIGINT persona\_id FK

        ENUM tipo\_documento

        VARCHAR numero

        DATE fecha\_expedicion

    }

    ESTADO\_CIVIL {

        TINYINT id\_estado PK

        VARCHAR descripcion

    }

    MATRIMONIO {

        BIGINT id\_matrimonio PK

        BIGINT persona\_id FK

        BIGINT conyuge\_id FK

        DATE fecha\_matrimonio

        TINYINT regimen\_id FK

    }

    REGIMEN\_MATRIMONIAL {

        TINYINT id\_regimen PK

        ENUM tipo

    }

    PODER {

        BIGINT id\_poder PK

        ENUM tipo\_poder

        DATE fecha\_otorgamiento

        VARCHAR documento\_digital

    }

    REPRESENTACION {

        BIGINT id\_representacion PK

        BIGINT representante\_id FK

        BIGINT representado\_id FK

        ENUM tipo

        DATE fecha\_inicio

        DATE fecha\_fin

        BIGINT poder\_id FK

    }

    FORMULARIO {

        BIGINT id\_formulario PK

        ENUM tipo

        TIMESTAMP fecha\_creacion

        ENUM estado

    }

    OTORGANTE {

        BIGINT formulario\_id PK, FK

        BIGINT persona\_id PK, FK

    }

    PERSONA\_FISICA ||--o{ DIRECCION : "reside\_en"

    PERSONA\_FISICA ||--o{ DOCUMENTO\_IDENTIDAD : "identificado\_por"

    PERSONA\_FISICA ||--|| ESTADO\_CIVIL : "tiene"

    PERSONA\_FISICA ||--o{ MATRIMONIO : "contrae"

    MATRIMONIO }|--|| REGIMEN\_MATRIMONIAL : "sujeto\_a"

    REPRESENTACION ||--|| PODER : "tiene\_base\_en"

    REPRESENTACION ||--|| PERSONA\_FISICA : "representante"

    REPRESENTACION ||--|| PERSONA\_FISICA : "representado"

    FORMULARIO ||--o{ OTORGANTE : "contiene"

    OTORGANTE }|--|| PERSONA\_FISICA : "es"

**1.8 Justificación General del Modelo**

**Coherencia con la normativa**

Todos los campos recogidos en el modelo tienen base normativa en la legislación notarial española. Se han extraído directamente de los formularios 1-A y 1-B, con el objetivo de:

* Cumplir con los requisitos del Colegio Notarial.
* Permitir verificaciones automáticas en tiempo real.
* Asegurar que la información puede ser reutilizada en futuros trámites.

**Escalabilidad**

El modelo permite fácilmente:

* Añadir múltiples matrimonios históricos.
* Gestionar direcciones temporales y permanentes.
* Incluir nuevos tipos de representación en el futuro.
* Adaptarse a nuevos formularios sin modificar el núcleo.

**Integración futura**

Puede integrarse con:

* API de verificación de NIF/NIE.
* Firmas electrónicas y validadores de poderes.
* Sistemas de gestión documental (para anexar PDFs de poderes escaneados).

**FASE 2: DISEÑO LÓGICO DE LA BASE DE DATOS**

**2.1 Transformación del Modelo ER a Esquema Relacional**

**2.1.1 Introducción al Diseño Lógico**

**Definición**: El diseño lógico de bases de datos consiste en transformar el modelo conceptual (diagrama entidad-relación) en un conjunto de tablas relacionales que cumplen con las reglas del modelo relacional. Esta fase traduce las entidades, atributos y relaciones en estructuras que pueden ser implementadas en un sistema de gestión de bases de datos (SGBD).

El objetivo principal es preservar la semántica del modelo conceptual y adaptarla a las restricciones y capacidades del entorno relacional.

**2.1.2 Reglas de transformación aplicadas**

1. **Cada entidad fuerte** se transforma en una tabla con una clave primaria.
2. **Relaciones 1:N** se transforman introduciendo claves foráneas en el lado "muchos".
3. **Relaciones N:M** se convierten en tablas intermedias.
4. **Atributos multivaluados** se normalizan creando una nueva tabla.
5. **Herencia (especialización)** se implementa mediante la técnica "una tabla por entidad", es decir, se conserva la tabla principal PERSONA\_FISICA, sin considerar subtipos.
6. **Atributos compuestos** se descomponen en atributos simples y atómicos.

**2.1.3 Esquema relacional derivado**

A continuación se muestra el esquema relacional generado a partir del modelo ER conceptual, reflejando cardinalidades, claves primarias y claves externas:

Изображение выглядит как диаграмма, текст, линия, План

Содержимое, созданное искусственным интеллектом, может быть неверным.

<https://www.mermaidchart.com/app/projects/f9a08323-1fa4-4747-b6d2-7f2261e7eeaf/diagrams/32a3d6cb-91cb-47c1-80e1-27febb1a2356/version/v0.1/edit>

erDiagram

PERSONA\_FISICA ||--o{ DIRECCION : "1:N"

PERSONA\_FISICA ||--o{ DOCUMENTO\_IDENTIDAD : "1:N"

PERSONA\_FISICA ||--|| ESTADO\_CIVIL : "1:1"

PERSONA\_FISICA ||--o{ MATRIMONIO : "1:N"

MATRIMONIO }|--|| REGIMEN\_MATRIMONIAL : "N:1"

REPRESENTACION }|--|| PERSONA\_FISICA : "N:1"

REPRESENTACION ||--|| PODER : "N:1"

FORMULARIO ||--o{ OTORGANTE : "1:N"

Este esquema proporciona una estructura limpia, sin redundancias y preparada para su implementación en MySQL o cualquier SGBD relacional compatible.

**2.2 Definición Detallada de Tablas**

**2.2.1 Tabla: PERSONA\_FISICA**

**Función**: almacenar los datos personales básicos de cada otorgante.

| **Campo** | **Tipo de dato** | **Restricciones** | **Descripción** |
| --- | --- | --- | --- |
| id | BIGINT | PK, AUTO\_INCREMENT | Identificador único |
| nombre | VARCHAR(50) | NOT NULL | Nombre legal de la persona |
| primer\_apellido | VARCHAR(50) | NOT NULL | Primer apellido |
| segundo\_apellido | VARCHAR(50) | NULLABLE | Segundo apellido, opcional |
| fecha\_nacimiento | DATE | NOT NULL, CHECK (edad >= 18) | Fecha de nacimiento |
| profesion | VARCHAR(100) | NULLABLE | Ocupación declarada |
| profesion\_riesgo | BOOLEAN | DEFAULT FALSE | Indicador de profesión considerada de riesgo |
| estado\_civil\_id | TINYINT | FK a ESTADO\_CIVIL(id) | Clave foránea al estado civil |

**2.2.2 Tabla: DOCUMENTO\_IDENTIDAD**

**Función**: registrar documentos válidos de identificación para cada persona.

| **Campo** | **Tipo de dato** | **Restricciones** | **Descripción** |
| --- | --- | --- | --- |
| id | BIGINT | PK, AUTO\_INCREMENT | Clave primaria |
| persona\_id | BIGINT | FK a PERSONA\_FISICA(id) | Clave foránea a persona |
| tipo\_documento | ENUM | CHECK ('NIF', 'NIE', 'PASAPORTE') | Tipo de documento |
| numero | VARCHAR(20) | UNIQUE, NOT NULL | Número del documento |
| fecha\_expedicion | DATE | NOT NULL | Fecha de emisión |

**2.2.3 Tabla: DIRECCION**

**Función**: almacenar información de residencia, tanto en territorio nacional como extranjero.

| **Campo** | **Tipo de dato** | **Restricciones** | **Descripción** |
| --- | --- | --- | --- |
| id | BIGINT | PK, AUTO\_INCREMENT | Identificador de dirección |
| persona\_id | BIGINT | FK a PERSONA\_FISICA(id) | Asociación con la persona |
| tipo\_via | VARCHAR(20) | NOT NULL | Calle, avenida, plaza |
| nombre\_via | VARCHAR(100) | NOT NULL | Nombre de la vía pública |
| numero | VARCHAR(10) | NOT NULL | Número o portal |
| escalera | VARCHAR(10) | NULLABLE | Escalera o bloque |
| piso | VARCHAR(10) | NULLABLE | Piso del inmueble |
| puerta | VARCHAR(10) | NULLABLE | Puerta específica |
| municipio | VARCHAR(50) | NOT NULL | Localidad |
| provincia | VARCHAR(50) | NOT NULL | Provincia o región |
| codigo\_postal | CHAR(5) | CHECK (LENGTH = 5) | Código postal español |
| pais | VARCHAR(50) | NOT NULL si extranjero | País de residencia |

**2.2.4 Tabla: ESTADO\_CIVIL**

**Función**: representar el estado civil legal según normativa.

| **Campo** | **Tipo de dato** | **Restricciones** | **Descripción** |
| --- | --- | --- | --- |
| id | TINYINT | PK | Clave primaria |
| descripcion | VARCHAR(20) | CHECK (IN ENUM) | 'SOLTERO', 'CASADO', 'VIUDO' etc. |

**2.2.5 Tabla: MATRIMONIO**

**Función**: registrar las relaciones matrimoniales con datos esenciales.

| **Campo** | **Tipo de dato** | **Restricciones** | **Descripción** |
| --- | --- | --- | --- |
| id | BIGINT | PK, AUTO\_INCREMENT | Clave primaria |
| persona\_id | BIGINT | FK a PERSONA\_FISICA(id) | Persona principal |
| conyuge\_id | BIGINT | FK a PERSONA\_FISICA(id), persona distinta | Cónyuge de la persona |
| fecha\_matrimonio | DATE | NOT NULL | Fecha legal de matrimonio |
| regimen\_id | BIGINT | FK a REGIMEN\_MATRIMONIAL(id) | Tipo de régimen legal |

**2.2.6 Tabla: REGIMEN\_MATRIMONIAL**

**Función**: tabla paramétrica que describe tipos legales.

| **Campo** | **Tipo de dato** | **Restricciones** | **Descripción** |
| --- | --- | --- | --- |
| id | TINYINT | PK | Clave primaria |
| nombre | ENUM | CHECK ('SOCIEDAD\_GANANCIALES', 'SEPARACION\_BIENES') | Tipos reconocidos por el código civil |

**2.2.7 Tabla: PODER**

**Función**: conservar información de documentos notariales habilitantes.

| **Campo** | **Tipo de dato** | **Restricciones** | **Descripción** |
| --- | --- | --- | --- |
| id | BIGINT | PK, AUTO\_INCREMENT | Clave primaria |
| tipo | ENUM | CHECK ('GENERAL', 'ESPECIAL') | Clasificación legal |
| documento\_digital | VARCHAR(255) | Ruta a documento PDF | Archivo vinculado |
| fecha\_otorgamiento | DATE | NOT NULL | Fecha legal de emisión |

**2.2.8 Tabla: REPRESENTACION**

**Función**: representar vínculos legales entre representado y representante.

| **Campo** | **Tipo de dato** | **Restricciones** | **Descripción** |
| --- | --- | --- | --- |
| id | BIGINT | PK, AUTO\_INCREMENT | Clave primaria |
| representante\_id | BIGINT | FK a PERSONA\_FISICA(id) | Persona que actúa |
| representado\_id | BIGINT | FK a PERSONA\_FISICA(id), CHECK(≠ representante) | Persona en cuyo nombre se actúa |
| tipo | ENUM | ('LEGAL', 'NOTARIAL') | Tipo de representación |
| fecha\_inicio | DATE | NOT NULL | Inicio de validez legal |
| fecha\_fin | DATE | NULLABLE | Caducidad (si aplica) |
| poder\_id | BIGINT | FK a PODER(id), NULLABLE | Documento asociado |

**2.2.9 Tabla: FORMULARIO**

**Función**: encapsular procesos de entrada de datos.

| **Campo** | **Tipo de dato** | **Restricciones** | **Descripción** |
| --- | --- | --- | --- |
| id\_formulario | BIGINT | PK, AUTO\_INCREMENT | Clave principal |
| tipo | ENUM | ('1-A', '1-B') | Tipo oficial |
| fecha\_creacion | TIMESTAMP | DEFAULT CURRENT\_TIMESTAMP | Registro de alta |
| estado | ENUM | ('BORRADOR','VALIDADO','ARCHIVADO') | Ciclo de vida |

**2.2.10 Tabla: OTORGANTE**

**Función**: relación intermedia entre formularios y personas físicas.

| **Campo** | **Tipo de dato** | **Restricciones** | **Descripción** |
| --- | --- | --- | --- |
| formulario\_id | BIGINT | FK a FORMULARIO(id\_formulario) | Formulario asociado |
| persona\_id | BIGINT | FK a PERSONA\_FISICA(id) | Persona compareciente |

**2.3 Normalización de la Base de Datos**

**2.3.1 Definición teórica**

**Normalización** es el proceso de estructuración de una base de datos con el objetivo de reducir la redundancia, evitar anomalías de inserción, actualización y eliminación, y mejorar la integridad de los datos. Se realiza a través de formas normales (NF), que son niveles progresivos de refinamiento estructural.

Cada forma normal define un conjunto de condiciones que una relación debe cumplir. En el ámbito de bases de datos transaccionales (OLTP), se suelen aplicar hasta la **Tercera Forma Normal (3FN)**.

**2.3.2 Aplicación de formas normales al modelo**

**Primera Forma Normal (1FN)**

**Criterio**: Todos los atributos deben contener valores atómicos, es decir, no multivaluados ni repetitivos.

**Evaluación**:

* Atributos como direccion, documento\_identidad o estado civil se han modelado en **tablas separadas**.
* No existen campos tipo lista, JSON o arreglos en el esquema relacional.
* Por tanto, **todo el modelo cumple 1FN**.

**Segunda Forma Normal (2FN)**

**Criterio**: Ningún atributo no clave debe depender parcialmente de una clave primaria compuesta.

**Aplicación**:

* Ninguna tabla usa clave compuesta como PK real.
* En tablas intermedias como OTORGANTE, la clave compuesta se usa como PK, pero no hay atributos adicionales dependientes de una sola parte.
* Por tanto, **el modelo cumple 2FN**.

**Tercera Forma Normal (3FN)**

**Criterio**: No debe haber **dependencias transitivas** entre atributos no clave. Es decir, un atributo no clave no debe depender de otro atributo no clave.

**Ejemplos tratados**:

* En MATRIMONIO, el atributo regimen\_id es clave foránea a la tabla REGIMEN\_MATRIMONIAL, evitando una dependencia semántica de texto.
* En PERSONA\_FISICA, el estado\_civil\_id apunta a una tabla normalizada ESTADO\_CIVIL, evitando tener el valor textual repetido.

**Conclusión**: Todas las relaciones están **normalizadas hasta 3FN**.

**2.3.3 Ventajas alcanzadas**

* **Reducción de redundancia**: la información sobre estado civil, direcciones o documentos no se repite en múltiples lugares.
* **Facilidad de mantenimiento**: las modificaciones en catálogos no afectan datos estructurales.
* **Consistencia**: cada tabla representa una sola entidad o relación.
* **Escalabilidad**: el modelo puede crecer sin pérdida de integridad.

**2.4 Integridad Referencial**

**2.4.1 Definición**

**Integridad referencial** es un principio de las bases de datos relacionales que asegura que los valores de claves foráneas (FK) coincidan con valores existentes en las claves primarias (PK) de las tablas referenciadas.

Esto garantiza que no existan referencias huérfanas (es decir, relaciones sin entidad de destino) y que se mantenga la coherencia entre las tablas relacionadas.

**2.4.2 Tipos de restricciones referenciales**

1. **ON DELETE CASCADE**  
   Si la fila referenciada se elimina, se eliminan automáticamente las filas relacionadas.
2. **ON DELETE SET NULL**  
   Si la fila referenciada se elimina, la FK se convierte en NULL (solo si está permitida).
3. **ON DELETE RESTRICT**  
   Impide la eliminación si hay referencias activas.

**2.4.3 Ejemplos SQL de implementación**

**Clave externa de PERSONA\_FISICA a ESTADO\_CIVIL**

*ALTER TABLE PERSONA\_FISICA*

*ADD CONSTRAINT fk\_estado\_civil*

*FOREIGN KEY (estado\_civil\_id)*

*REFERENCES ESTADO\_CIVIL(id)*

*ON DELETE RESTRICT;*

**Justificación**: no se permite eliminar un estado civil si existe una persona asociada a él.

**Clave externa de DIRECCION a PERSONA\_FISICA**

*ALTER TABLE DIRECCION*

*ADD CONSTRAINT fk\_direccion\_persona*

*FOREIGN KEY (persona\_id)*

*REFERENCES PERSONA\_FISICA(id)*

*ON DELETE CASCADE;*

**Justificación**: si se elimina una persona, también deben eliminarse sus direcciones.

**Clave externa de REPRESENTACION a PODER**

*ALTER TABLE REPRESENTACION*

*ADD CONSTRAINT fk\_representacion\_poder*

*FOREIGN KEY (poder\_id)*

*REFERENCES PODER(id)*

*ON DELETE SET NULL;*

**Justificación**: si un poder expira o se elimina por error, la relación se conserva pero deja constancia de que el vínculo ha perdido su documento base.

**Clave compuesta de OTORGANTE (relación FORMULARIO – PERSONA)**

*ALTER TABLE OTORGANTE*

*ADD CONSTRAINT fk\_otorgante\_formulario*

*FOREIGN KEY (formulario\_id)*

*REFERENCES FORMULARIO(id\_formulario)*

*ON DELETE CASCADE;*

*ALTER TABLE OTORGANTE*

*ADD CONSTRAINT fk\_otorgante\_persona*

*FOREIGN KEY (persona\_id)*

*REFERENCES PERSONA\_FISICA(id)*

*ON DELETE CASCADE;*

**Justificación**: la eliminación de un formulario o una persona debe reflejarse también en esta tabla intermedia.

**2.4.4 Restricciones CHECK adicionales**

**Verificación de mayoría de edad**

*ALTER TABLE PERSONA\_FISICA*

*ADD CONSTRAINT chk\_mayoria\_edad*

*CHECK (fecha\_nacimiento <= DATE\_SUB(CURDATE(), INTERVAL 18 YEAR));*

**Justificación**: toda persona otorgante debe ser legalmente mayor de edad.

**Formato válido de NIF y NIE**

*ALTER TABLE DOCUMENTO\_IDENTIDAD*

*ADD CONSTRAINT chk\_documento\_valido*

*CHECK (*

(tipo\_documento = 'NIF' AND numero REGEXP '^[0-9]{8}[A-Z]$') OR

(tipo\_documento = 'NIE' AND numero REGEXP '^[XYZ][0-9]{7}[A-Z]$')

);

**Justificación**: los documentos deben seguir el patrón legalmente establecido.

**2.4.5 Beneficios del diseño referencial**

* Prevención de errores por inconsistencia entre tablas.
* Reducción de operaciones manuales de limpieza.
* Mejor control de datos huérfanos.
* Apoyo a la lógica del negocio desde el nivel de base de datos.

**Conclusión de la Fase 2**

El diseño lógico propuesto:

* Refleja fielmente el modelo conceptual.
* Respeta las reglas del modelo relacional.
* Está completamente normalizado hasta 3FN.
* Garantiza la integridad referencial en cada entidad y relación.
* Está preparado para ser implementado en un SGBD como MySQL con mínimo riesgo de inconsistencia.

**FASE 3: IMPLEMENTACIÓN FÍSICA**

**3.1 ELECCIÓN DEL SISTEMA GESTOR DE BASE DE DATOS (SGBD)**

**Definición**

Un **Sistema Gestor de Base de Datos (SGBD)** es una aplicación de software que proporciona las herramientas necesarias para definir, construir, manipular y mantener una base de datos. El SGBD actúa como interfaz entre los usuarios y la base de datos, garantizando el acceso seguro, eficiente y coherente a los datos.

**Criterios de selección para el entorno notarial**

Para un sistema de gestión de datos notariales, donde la integridad, la auditabilidad y la escalabilidad son factores determinantes, el SGBD seleccionado debe cumplir con los siguientes requisitos:

* **Compatibilidad con integridad referencial**
* **Soporte de transacciones ACID**
* **Amplia adopción y comunidad activa**
* **Mecanismos robustos de seguridad**
* **Facilidad de integración con aplicaciones modernas (Java, Python, etc.)**
* **Disponibilidad de funciones para vistas, procedimientos almacenados e índices compuestos**

**Comparativa de SGBD candidatos**

| **Criterio** | **MySQL** | **PostgreSQL** | **Oracle** | **SQL Server** |
| --- | --- | --- | --- | --- |
| Rendimiento en consultas | Alto | Muy alto | Alto | Alto |
| Soporte de integridad | Completo | Completo | Completo | Completo |
| Transacciones ACID | Sí | Sí | Sí | Sí |
| Licencia | Open Source | Open Source | Comercial | Comercial |
| Escalabilidad | Alta | Muy alta | Muy alta | Alta |
| Procedimientos almacenados | Sí | Sí | Sí | Sí |
| Facilidad de uso | Alta | Media | Media | Alta |
| Comunidad y documentación | Muy amplia | Amplia | Limitada | Limitada |

**Elección definitiva: MySQL**

**Justificación técnica**:

1. **Motor InnoDB**: ofrece integridad referencial, transacciones ACID y bloqueo a nivel de fila.
2. **Lenguaje SQL estándar**: facilita la portabilidad y la integración.
3. **Compatibilidad con herramientas modernas**: compatible con frameworks como Hibernate, JPA, y tecnologías Java EE.
4. **Rendimiento optimizado**: gracias a su arquitectura simple y efectiva.
5. **Comunidad**: miles de tutoriales, foros, soporte empresarial de Oracle (si se requiere).
6. **Licencia abierta**: ideal para entornos educativos, institucionales y de código abierto.

Por tanto, el sistema será implementado sobre **MySQL 8.0 o superior** con codificación utf8mb4 y motor InnoDB.

**3.2 SCRIPT DE CREACIÓN COMPLETO**

A continuación, se presenta el **script completo y comentado** de creación de las tablas, claves primarias, claves externas y restricciones CHECK. Este código es directamente ejecutable en MySQL Workbench o CLI.

Изображение выглядит как текст, диаграмма, План, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

<https://www.mermaidchart.com/app/projects/f9a08323-1fa4-4747-b6d2-7f2261e7eeaf/diagrams/344546fe-b826-4ff3-96c3-89d9e19bc970/version/v0.1/edit>

/\*Creación de base de datos\*/

DROP DATABASE IF EXISTS notarial\_db;

CREATE DATABASE IF NOT EXISTS notarial\_db

DEFAULT CHARACTER SET utf8mb4

COLLATE utf8mb4\_general\_ci;

USE notaria\_db;

/\*Tabla: ESTADO\_CIVIL\*/

CREATE TABLE ESTADO\_CIVIL (

id TINYINT PRIMARY KEY,

descripcion VARCHAR(20) NOT NULL UNIQUE

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

/\*Tabla: REGIMEN\_MATRIMONIAL\*/

CREATE TABLE REGIMEN\_MATRIMONIAL (

id TINYINT PRIMARY KEY,

nombre ENUM('SOCIEDAD\_GANANCIALES', 'SEPARACION\_BIENES') NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

/\*Tabla: PERSONA\_FISICA\*/

CREATE TABLE PERSONA\_FISICA (

id BIGINT PRIMARY KEY AUTO\_INCREMENT,

nombre VARCHAR(50) NOT NULL,

primer\_apellido VARCHAR(50) NOT NULL,

segundo\_apellido VARCHAR(50),

fecha\_nacimiento DATE NOT NULL,

profesion VARCHAR(100),

profesion\_riesgo BOOLEAN DEFAULT FALSE,

estado\_civil\_id TINYINT NOT NULL,

CONSTRAINT fk\_estado\_civil FOREIGN KEY (estado\_civil\_id)

REFERENCES ESTADO\_CIVIL(id)

ON DELETE RESTRICT

-- NOTA: Eliminado CHECK con CURDATE() (MySQL no lo permite en versiones anteriores a 8.0.16)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

/\*Tabla: DIRECCION\*/

CREATE TABLE DIRECCION (

id BIGINT PRIMARY KEY AUTO\_INCREMENT,

persona\_id BIGINT NOT NULL,

tipo\_via VARCHAR(20) NOT NULL,

nombre\_via VARCHAR(100) NOT NULL,

numero VARCHAR(10) NOT NULL,

escalera VARCHAR(10),

piso VARCHAR(10),

puerta VARCHAR(10),

municipio VARCHAR(50) NOT NULL,

provincia VARCHAR(50) NOT NULL,

codigo\_postal CHAR(5) NOT NULL,

pais VARCHAR(50),

CONSTRAINT fk\_direccion\_persona FOREIGN KEY (persona\_id)

REFERENCES PERSONA\_FISICA(id)

ON DELETE CASCADE,

CONSTRAINT chk\_codigo\_postal CHECK (CHAR\_LENGTH(codigo\_postal) = 5)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

/\*Tabla: DOCUMENTO\_IDENTIDAD\*/

CREATE TABLE DOCUMENTO\_IDENTIDAD (

id BIGINT PRIMARY KEY AUTO\_INCREMENT,

persona\_id BIGINT NOT NULL,

tipo\_documento ENUM('NIF','NIE','PASAPORTE') NOT NULL,

numero VARCHAR(20) NOT NULL UNIQUE,

fecha\_expedicion DATE NOT NULL,

CONSTRAINT fk\_documento\_persona FOREIGN KEY (persona\_id)

REFERENCES PERSONA\_FISICA(id)

ON DELETE CASCADE,

CONSTRAINT chk\_documento\_formato CHECK (

(tipo\_documento = 'NIF' AND numero REGEXP '^[0-9]{8}[A-Z]$') OR

(tipo\_documento = 'NIE' AND numero REGEXP '^[XYZ][0-9]{7}[A-Z]$')

)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

/\*Tabla: MATRIMONIO\*/

CREATE TABLE MATRIMONIO (

id BIGINT PRIMARY KEY AUTO\_INCREMENT,

persona\_id BIGINT NOT NULL,

conyuge\_id BIGINT NOT NULL,

fecha\_matrimonio DATE NOT NULL,

regimen\_id TINYINT NOT NULL,

CONSTRAINT fk\_persona\_matrimonio FOREIGN KEY (persona\_id)

REFERENCES PERSONA\_FISICA(id)

ON DELETE RESTRICT,

CONSTRAINT fk\_conyuge\_matrimonio FOREIGN KEY (conyuge\_id)

REFERENCES PERSONA\_FISICA(id)

ON DELETE RESTRICT,

CONSTRAINT fk\_matrimonio\_regimen FOREIGN KEY (regimen\_id)

REFERENCES REGIMEN\_MATRIMONIAL(id)

ON DELETE RESTRICT,

CONSTRAINT chk\_conyuge\_distinto CHECK (persona\_id <> conyuge\_id)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

/\*Tabla: PODER\*/

CREATE TABLE PODER (

id BIGINT PRIMARY KEY AUTO\_INCREMENT,

tipo ENUM('GENERAL','ESPECIAL') NOT NULL,

documento\_digital VARCHAR(255) NOT NULL,

fecha\_otorgamiento DATE NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

/\*Tabla: REPRESENTACION\*/

CREATE TABLE REPRESENTACION (

id BIGINT PRIMARY KEY AUTO\_INCREMENT,

representante\_id BIGINT NOT NULL,

representado\_id BIGINT NOT NULL,

tipo ENUM('LEGAL','NOTARIAL') NOT NULL,

fecha\_inicio DATE NOT NULL,

fecha\_fin DATE,

poder\_id BIGINT,

CONSTRAINT fk\_representante FOREIGN KEY (representante\_id)

REFERENCES PERSONA\_FISICA(id)

ON DELETE CASCADE,

CONSTRAINT fk\_representado FOREIGN KEY (representado\_id)

REFERENCES PERSONA\_FISICA(id)

ON DELETE CASCADE,

CONSTRAINT fk\_poder FOREIGN KEY (poder\_id)

REFERENCES PODER(id)

ON DELETE SET NULL,

CONSTRAINT chk\_distintos CHECK (representante\_id <> representado\_id)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

/\*Tabla: FORMULARIO\*/

CREATE TABLE FORMULARIO (

id\_formulario BIGINT PRIMARY KEY AUTO\_INCREMENT,

tipo ENUM('1-A','1-B') NOT NULL,

fecha\_creacion TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

estado ENUM('BORRADOR','VALIDADO','ARCHIVADO') NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

/\*Tabla: OTORGANTE\*/

CREATE TABLE OTORGANTE (

formulario\_id BIGINT NOT NULL,

persona\_id BIGINT NOT NULL,

PRIMARY KEY (formulario\_id, persona\_id),

FOREIGN KEY (formulario\_id) REFERENCES FORMULARIO(id\_formulario)

ON DELETE CASCADE,

FOREIGN KEY (persona\_id) REFERENCES PERSONA\_FISICA(id)

ON DELETE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

**3.3 ÍNDICES Y OPTIMIZACIÓN**

**Definición teórica**

Un **índice** en bases de datos es una estructura auxiliar que permite acelerar el acceso a los datos almacenados en una tabla. Su funcionamiento es similar al índice de un libro: permite buscar con mayor rapidez un valor específico sin necesidad de revisar cada fila una a una.

Los índices se implementan mediante estructuras de datos como **árboles B+**, que permiten búsquedas, inserciones y borrados eficientes.

**Tipos de índices en MySQL**

1. **Índices primarios (PRIMARY KEY)**:
   * Se crean automáticamente con la clave primaria de la tabla.
   * Garantizan unicidad y ordenamiento.
2. **Índices únicos (UNIQUE INDEX)**:
   * Evitan duplicados en una columna específica (como el número de documento).
3. **Índices secundarios (INDEX / KEY)**:
   * Permiten optimizar búsquedas sobre columnas que no son claves.
4. **Índices compuestos (composite indexes)**:
   * Se definen sobre múltiples columnas para optimizar consultas con varias condiciones.

**Estrategia de indexación para el modelo notarial**

**Tabla PERSONA\_FISICA**

CREATE INDEX idx\_fecha\_nacimiento ON PERSONA\_FISICA(fecha\_nacimiento);

* **Justificación**: las búsquedas por edad son frecuentes en validaciones.

**Tabla DOCUMENTO\_IDENTIDAD**

CREATE UNIQUE INDEX idx\_numero\_documento ON DOCUMENTO\_IDENTIDAD(numero);

* **Justificación**: garantiza unicidad del número de documento, incluso entre NIF y NIE.

**Tabla REPRESENTACION**

CREATE INDEX idx\_representante ON REPRESENTACION(representante\_id);

CREATE INDEX idx\_representado ON REPRESENTACION(representado\_id);

* **Justificación**: optimiza la búsqueda de relaciones de representación.

**Tabla OTORGANTE**

CREATE INDEX idx\_otorgante\_formulario ON OTORGANTE(formulario\_id);

* **Justificación**: mejora las consultas que recuperan personas por formulario.

**Recomendaciones adicionales de optimización**

* **Motor InnoDB**: permite bloqueo a nivel de fila y recuperación transaccional.
* \*\*Evitar SELECT \*\*\*: seleccionar sólo columnas necesarias reduce E/S.
* **Revisar planes de ejecución** (EXPLAIN SELECT) para afinar consultas.

**3.4 VISTAS PARA CONSULTAS COMUNES**

**Definición**

Una **vista (VIEW)** es una tabla virtual definida por una consulta SELECT. Las vistas permiten:

* Simplificar consultas complejas.
* Encapsular lógica de negocio.
* Mejorar seguridad ocultando detalles de tablas.

**Vista 1: Personas y su Estado Civil**

CREATE VIEW vw\_personas\_estado\_civil AS

SELECT

p.id,

CONCAT(p.nombre, ' ', p.primer\_apellido, ' ', COALESCE(p.segundo\_apellido, '')) AS nombre\_completo,

ec.descripcion AS estado\_civil

FROM PERSONA\_FISICA p

JOIN ESTADO\_CIVIL ec ON p.estado\_civil\_id = ec.id;

**Uso**: consultar el estado civil de todos los otorgantes.

**Vista 2: Documentos de Identidad vigentes**

CREATE VIEW vw\_documentos\_vigentes AS

SELECT

d.id,

d.tipo\_documento,

d.numero,

p.id AS persona\_id,

p.nombre

FROM DOCUMENTO\_IDENTIDAD d

JOIN PERSONA\_FISICA p ON d.persona\_id = p.id

WHERE d.fecha\_expedicion >= DATE\_SUB(CURDATE(), INTERVAL 10 YEAR);

**Uso**: verificación de documentos actuales.

**Vista 3: Relaciones de representación activas**

CREATE VIEW vw\_representaciones\_activas AS

SELECT

r.id,

r.representante\_id,

r.representado\_id,

r.tipo,

r.fecha\_inicio,

r.fecha\_fin

FROM REPRESENTACION r

WHERE r.fecha\_fin IS NULL OR r.fecha\_fin > CURDATE();

**Uso**: auditoría de relaciones vigentes.

**Beneficios operativos**

* Mayor claridad para desarrolladores y usuarios avanzados.
* Centralización de la lógica de combinación de datos.
* Menor riesgo de errores en los formularios o reportes.

**3.5 PROCEDIMIENTOS ALMACENADOS**

**Definición**

Un **procedimiento almacenado (Stored Procedure)** es un bloque de código SQL compilado y almacenado en el servidor. Su propósito es encapsular operaciones frecuentes o críticas, con beneficios en:

* Rendimiento (menos tráfico red-cliente).
* Reutilización de lógica.
* Seguridad (control de acceso granular).

**Procedimiento 1: Insertar nuevo otorgante**

DELIMITER $$

CREATE PROCEDURE insertar\_otorgante(

IN p\_formulario\_id BIGINT,

IN p\_persona\_id BIGINT

)

BEGIN

INSERT INTO OTORGANTE(formulario\_id, persona\_id)

VALUES (p\_formulario\_id, p\_persona\_id);

END$$

DELIMITER ;

**Uso**: desde una aplicación externa, registrar a un nuevo firmante.

**Procedimiento 2: Eliminar persona y sus datos relacionados**

DELIMITER $$

CREATE PROCEDURE eliminar\_persona\_completa(IN p\_id BIGINT)

BEGIN

DELETE FROM PERSONA\_FISICA WHERE id = p\_id;

END$$

DELIMITER ;

**Importante**: gracias a ON DELETE CASCADE, también se eliminarán:

* Direcciones
* Documentos
* Matrimonios
* Representaciones donde figure

**Procedimiento 3: Validar existencia de documento**

DELIMITER $$

CREATE PROCEDURE verificar\_documento(

IN p\_numero VARCHAR(20),

OUT existe BOOLEAN

)

BEGIN

DECLARE contador INT;

SELECT COUNT(\*) INTO contador

FROM DOCUMENTO\_IDENTIDAD

WHERE numero = p\_numero;

SET existe = contador > 0;

END$$

DELIMITER ;

**Uso**: prevenir duplicación desde la aplicación cliente.

**Procedimiento 4: Generar resumen de formulario**

DELIMITER $$

CREATE PROCEDURE resumen\_formulario(IN p\_formulario\_id BIGINT)

BEGIN

SELECT f.id\_formulario, f.tipo, f.estado, f.fecha\_creacion,

COUNT(o.persona\_id) AS total\_otorgantes

FROM FORMULARIO f

LEFT JOIN OTORGANTE o ON f.id\_formulario = o.formulario\_id

WHERE f.id\_formulario = p\_formulario\_id

GROUP BY f.id\_formulario;

END$$

DELIMITER ;

**Uso**: en informes administrativos internos.

**Conclusión Fase 3**

Esta fase garantiza:

* Una **implementación física robusta** en MySQL 8.0+.
* Un esquema preparado para operaciones críticas con integridad referencial.
* Altas prestaciones gracias a **índices estratégicos**.
* Acceso simplificado mediante **vistas SQL**.
* Automación de operaciones clave con **procedimientos almacenados**.

1. **Estructura de Nombres de Tablas:**

* Proyecto Notario usa: @Table(name = "personas")
* Base de datos usa: PERSONA\_FISICA

1. **Campos en PERSONA\_FISICA vs Clase Persona:**

Base de datos tiene:

id BIGINT  
nie VARCHAR(20)  
dni VARCHAR(20)  
nombre VARCHAR(50)  
primer\_apellido VARCHAR(50)  
segundo\_apellido VARCHAR(50)  
genero VARCHAR(20)  
fecha\_nacimiento DATE  
profesion VARCHAR(100)  
profesion\_riesgo BOOLEAN  
estado\_civil\_id TINYINT

Clase Persona tiene:

id Long  
dni String  
nombre String  
primerApellido String  
segundoApellido String  
genero String  
fechaNacimiento LocalDate  
profesion String

1. **Cambios Necesarios en la Clase Persona:**

@Entity  
@Table(name = "PERSONA\_FISICA")  
public class Persona {  
 @Id  
 @GeneratedValue(strategy = GenerationType.IDENTITY)  
 private Long id;  
   
 private String dni;  
 private String nie;  
   
 private String nombre;  
   
 @Column(name = "primer\_apellido")  
 private String primerApellido;  
   
 @Column(name = "segundo\_apellido")  
 private String segundoApellido;  
   
 private String genero;  
   
 @Column(name = "fecha\_nacimiento")  
 private LocalDate fechaNacimiento;  
   
 private String profesion;  
   
 @Column(name = "profesion\_riesgo")  
 private Boolean profesionRiesgo;  
   
 @ManyToOne  
 @JoinColumn(name = "estado\_civil\_id")  
 private EstadoCivil estadoCivil;  
}

1. **Nuevas Entidades Necesarias:**

@Entity  
@Table(name = "ESTADO\_CIVIL")  
public class EstadoCivil {  
 @Id  
 private Byte id;  
 private String descripcion;  
}  
  
@Entity  
@Table(name = "DIRECCION")  
public class Direccion {  
 @Id  
 @GeneratedValue(strategy = GenerationType.IDENTITY)  
 private Long id;  
   
 @ManyToOne  
 @JoinColumn(name = "persona\_id")  
 private Persona persona;  
   
 @Column(name = "tipo\_via")  
 private String tipoVia;  
   
 @Column(name = "nombre\_via")  
 private String nombreVia;  
   
 private String numero;  
 private String escalera;  
 private String piso;  
 private String puerta;  
 private String municipio;  
 private String provincia;  
   
 @Column(name = "codigo\_postal")  
 private String codigoPostal;  
   
 private String pais;  
}

1. **Principales Diferencias y Recomendaciones:**

a) **Documentos de Identidad:**

* La base de datos ya incluye y directamente en dniniePERSONA\_FISICA
* También tiene una tabla separada para más detalles DOCUMENTO\_IDENTIDAD
* Se recomienda usar ambos enfoques: campos básicos en y detalles en PERSONA\_FISICADOCUMENTO\_IDENTIDAD

b) **Estado Civil:**

* La base de datos tiene una relación con ESTADO\_CIVIL
* Es necesario agregar esta relación en la clase Persona

c) **Campos Adicionales:**

* Agregar profesionRiesgo en la clase Persona
* Considerar la relación con DIRECCION
* Evaluar si se necesitan las relaciones con , y MATRIMONIOPODERREPRESENTACION

1. **Plan de Acción:**
2. Modificar el nombre de la tabla en la anotación @Table
3. Agregar todos los campos faltantes en la clase Persona
4. Crear las entidades relacionadas (EstadoCivil, Direccion, etc.)
5. Agregar las relaciones necesarias (@OneToMany, @ManyToOne)
6. Actualizar los repositorios correspondientes
7. Modificar los controladores según la nueva estructura
8. **Consideraciones de Interfaz:**

* Los formularios HTML (esp.html y extr.html) ya están alineados con la estructura de la base de datos
* La interfaz maneja correctamente los campos adicionales como estado civil y dirección