Modelo de Datos

**Testify**

OSLO

Ojeda Valeria – Sly Eduardo

Levipichun Emilio – Oyarzo Malena

**

Un modelo es un conjunto de herramientas conceptuales para describir datos, sus relaciones, su significado y sus restricciones de consistencia.

Es el proceso de analizar los aspectos de interés para una organización y la relación que tienen unos con otros.

Resulta en el descubrimiento y documentación de los recursos de datos del negocio.

Es una tarea difícil, bastante difícil, pero es una actividad necesaria cuya habilidad solo se adquiere con la experiencia.

Tabla de contenido

[Introducción 3](#_Toc257618990)

[Propósito 3](#_Toc257618991)

[Alcance 3](#_Toc257618992)

[Referencias 3](#_Toc257618993)

[Diseño de Datos 3](#_Toc257618994)

[Modelo de Datos General 3](#_Toc257618995)

[Modelo Entidad-Relación 3](#_Toc257618996)

[Definición 3](#_Toc257618997)

[Descripción de Entidades y Atributos 3](#_Toc257618998)

[Llaves 3](#_Toc257618999)

[Relaciones Encontradas 3](#_Toc257619000)

[Diagrama E-R 3](#_Toc257619001)

[Principios de diseño 3](#_Toc257619002)

[Notación diagrama E-R 3](#_Toc257619003)

[Ejemplos Diagrama E-R 3](#_Toc257619004)

[Modelo Relacional 3](#_Toc257619005)

[Comentarios sobre el Modelo Relacional 3](#_Toc257619006)

[Normalización 3](#_Toc257619007)

[Justificación de Forma normal adoptada 3](#_Toc257619008)

[Especificación de la Distribución de Datos 3](#_Toc257619009)

Modelo de Datos

Introducción

El presente documento de Modelo de Datos tiene como objetivo proporcionar una descripción detallada de la estructura de datos que utilizará el sistema Testify, un sistema diseñado para la gestión de casos de prueba en proyectos de software. El modelo de datos es una representación conceptual que facilita la comprensión de las relaciones, restricciones y significado de los datos almacenados en el sistema, y es fundamental para asegurar la consistencia y eficiencia en la gestión de información.

Propósito

El propósito de este documento es describir el Modelo de Datos del sistema Testify. Está dirigido tanto a los desarrolladores del sistema como a los diseñadores de bases de datos que requieren una visión clara y estructurada de cómo se organizan los datos dentro del sistema. También le es útil a los analistas y arquitectos. Este documento tiene como objetivos principales:

* Registrar los requerimientos de datos de un proceso de negocio.
* Proporcionar una estructura clara que permita observar patrones de datos y usos potenciales.
* Facilitar la optimización de la gestión de datos, incluso en procesos complejos.

Alcance

Este documento aplica al sistema de gestión de casos de prueba Testify. Afecta directamente a los procesos de gestión de proyectos, asignación de roles y permisos, así como la administración de casos de prueba, hitos y usuarios. El modelo de datos cubre todas las entidades que interactúan dentro del sistema, como usuarios, proyectos, hitos, roles, permisos y casos de prueba. La información contenida en este documento influenciará las decisiones de diseño e implementación del sistema.

Referencias

* Especificación de Requerimientos del Sistema Testify, OSLO, Septiembre 2024​.
* Modelo de Casos de Uso de Testify, OSLO, Septiembre 2024.

Diseño de Datos

El diseño de datos del sistema Testify se basa en los requerimientos funcionales y no funcionales del sistema. A través de un modelo relacional, se busca maximizar la eficiencia en la gestión de la información de proyectos y casos de prueba, asegurando integridad y consistencia en las relaciones entre entidades clave como usuarios, roles, permisos y casos de prueba. El diseño incluye un modelo entidad-relación (E-R) que identifica las entidades principales y las relaciones que existen entre ellas. A su vez, este modelo permite la normalización de los datos, lo que reduce la redundancia y mejora la escalabilidad del sistema.

Modelo de Datos General

En este modelo se describen las entidades clave que componen el sistema Testify. Las principales entidades son:

* Usuario: Representa a los usuarios del sistema y almacena información como su nombre, email y rol. Un usuario puede tener uno o más roles asignados por proyecto.
* Proyecto: Cada proyecto está compuesto por hitos y casos de prueba. Tiene atributos como nombre, estado y tasa de aprobación.
* Hito: Los hitos son divisiones dentro de los proyectos que agrupan los casos de prueba.
* Caso de Prueba: Una entidad que contiene el conjunto de pruebas asociadas a un proyecto. Tiene atributos como nombre, tipo (documentación o código), estado y los usuarios involucrados.
* Rol: Define los permisos de un usuario en el sistema. Los roles están relacionados con los proyectos y casos de prueba.
* Permiso: Define las acciones que un rol puede ejecutar dentro del sistema.

Además, se establecen las relaciones entre estas entidades, como la relación entre un Usuario y un Proyecto a través de la entidad RolesAsignados, que define qué rol tiene cada usuario en un proyecto específico​.

Este modelo de datos se implementará siguiendo los principios de normalización para garantizar la eficiencia y evitar la redundancia de datos.

## Modelo Entidad-Relación

### Definición

El modelo entidad-relación (E/R) es una representación gráfica que permite visualizar de manera clara y precisa la estructura de los datos en un sistema. Está compuesto por entidades, que representan los objetos o elementos del mundo real que se desean modelar; atributos, que son las características o propiedades de dichas entidades; relaciones, que conectan las entidades entre sí, mostrando cómo interactúan; cardinalidad, que define la cantidad de instancias de una entidad que pueden estar asociadas a una instancia de otra entidad; y llaves, que identifican de forma única a cada instancia de una entidad, siendo estas las llaves primarias o foráneas. Este modelo facilita la comprensión de la organización de los datos y las interacciones entre ellos, lo que es esencial para la implementación eficiente de bases de datos.

Entidades: Incluye las entidades principales como Usuario, Proyecto, Hito, CasoDePrueba, Rol, Permiso, RolesAsignados y RolPermiso.

Atributos: Cada entidad tiene atributos con su tipo de datos, y las llaves primarias están marcadas con <<PK>>. Las llaves foráneas están marcadas con <<FK>>.

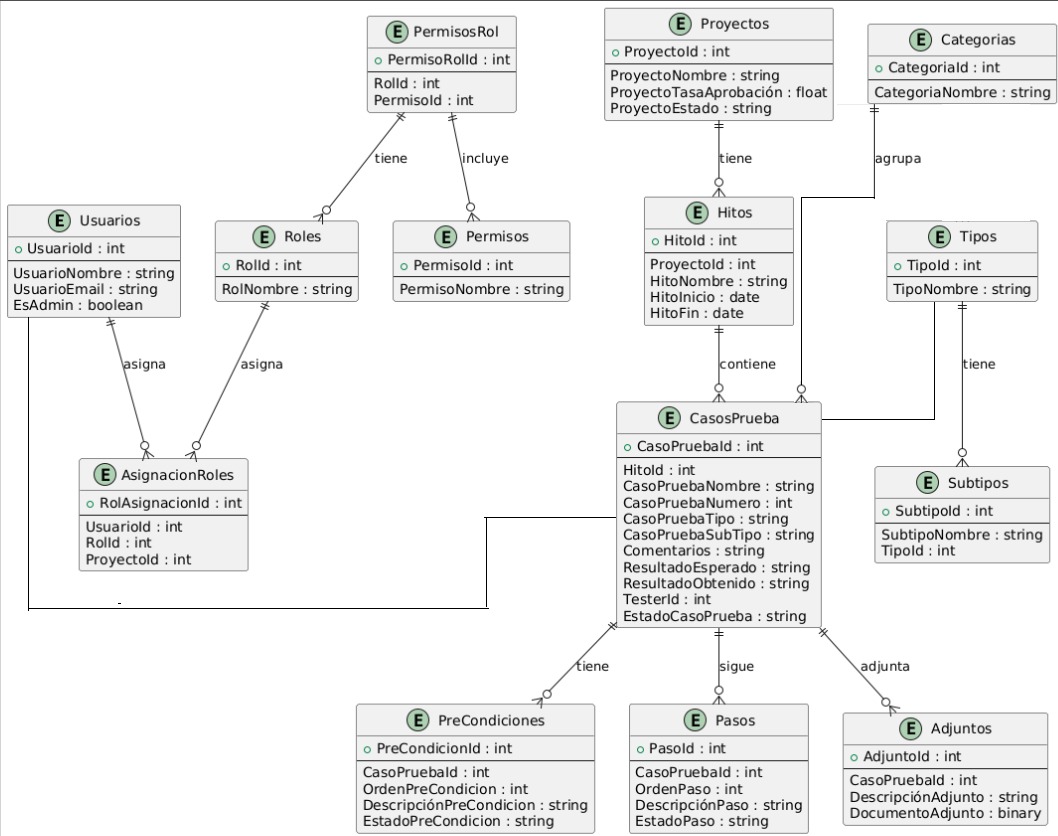
Relaciones: Se definen las relaciones entre las entidades con las cardinalidades adecuadas. Por ejemplo:

Descripción de Entidades y Atributos

A continuación, se describen todas las entidades encontradas en el proceso de análisis del sistema Testify. Para cada entidad, se listan los atributos detectados. Una entidad es todo lo que existe y es capaz de ser descrito (sustantivo). En otras palabras, es una cosa u objeto del mundo real con existencia propia y distinguible del resto. Un objeto con existencia puede ser física o abstracta. Mencionado esto, listamos las entidades identificadas, ordenadas alfabéticamente:

|  |  |  |
| --- | --- | --- |
| **Entidad** | **Atributos** | **Observaciones** |
| Proyecto | id (PK), nombre, estado (EstadoProyectoEnum), aprobacion | id: Identificador único.  - estado: Enum con valores: pendiente, enProceso, completado.  - Relación uno a muchos con Hito, muchos a muchos con Usuario a través de RolesAsignados. |
| Usuario | id (PK), nombre, email, esAdmin, autenticado | - id: Identificador único.  - esAdmin: Define si el usuario tiene privilegios de administrador.  - Relación muchos a muchos con Proyecto a través de RolesAsignados. |
| Rol | id (PK), nombre - id: Identificador único del rol. | - Relación muchos a muchos con Permiso a través de RolPermiso.  - Relación uno a muchos con RolesAsignados. |
| Permiso | id (PK), nombre - id: Identificador único del permiso. | - Relación muchos a muchos con Rol a través de RolPermiso. |
|  | id (PK), usuario\_id (FK), rol\_id (FK), proyecto\_id (FK) - id: Identificador único. | - Define la asignación de roles a usuarios dentro de proyectos.  - Llaves foráneas hacia Usuario, Rol y Proyecto. |
| RolesAsignados  - Relación uno a muchos con Revision. | Hito id (PK), nombre, fechaInicio, fechaFin, proyecto\_id (FK) - id: Identificador único. | - Relación muchos a uno con Proyecto. |
| Revision |  | - Llaves foráneas hacia Hito, Usuario (como tester y desarrollador). |
| id (PK), nombre, estado (EstadoCasoPruebaEnum), resultadoEsperado, resultadoObtenido, prioridad (PrioridadEnum), tester\_id (FK),  developer\_id (FK), hito\_id (FK) - id: Identificador único. |  | - estado: Enum con valores pendiente, enPruebas, corregir, aprobado. |
| Checklist | id (PK), descripcion, estado, revision\_id (FK) - id: Identificador único del checklist. | - estado: Indica si el checklist está activo o no.  - Relación muchos a uno con Revision. |
| Paso | id (PK), descripcion, revision\_id (FK) - id: Identificador único. | - Relación muchos a uno con Revision.  - Relación muchos a muchos entre Rol y Permiso. |
| Categoria | id (PK), nombre, proyecto\_id (FK) | - id: Identificador único de la categoría.  - Relación muchos a uno con Proyecto. |
| RolPermiso | id (PK), rol\_id (FK), permiso\_id (FK) | - id: Identificador único. |

## Diagrama E-R



**Entidades con llaves primarias, superllaves y candidatas:**

**Usuario**

* **Llave primaria (PK)**: UsuarioId
* **Superllave**: {UsuarioId}, {UsuarioEmail}, {UsuarioNombre, UsuarioEmail}
* **Llaves candidatas**: {UsuarioId}, {UsuarioEmail}

**RolAsignacion**

* **Llave primaria (PK)**: RolAsignacionId
* **Superllave**: {RolAsignacionId}, {UsuarioId, RolId, ProyectoId}
* **Llaves candidatas**: {RolAsignacionId}, {UsuarioId, RolId, ProyectoId}

**Rol**

* **Llave primaria (PK)**: RolId
* **Superllave**: {RolId}, {RolNombre}
* **Llaves candidatas**: {RolId}, {RolNombre}

**Permiso**

* **Llave primaria (PK)**: PermisoId
* **Superllave**: {PermisoId}, {PermisoNombre}
* **Llaves candidatas**: {PermisoId}, {PermisoNombre}

**PermisoRol**

* **Llave primaria (PK)**: PermisoRolId
* **Superllave**: {PermisoRolId}, {RolId, PermisoId}
* **Llaves candidatas**: {PermisoRolId}, {RolId, PermisoId}

**Proyecto**

* **Llave primaria (PK)**: ProyectoId
* **Superllave**: {ProyectoId}, {ProyectoNombre}
* **Llaves candidatas**: {ProyectoId}, {ProyectoNombre}

**Hito**

* **Llave primaria (PK)**: HitoId
* **Superllave**: {HitoId}, {ProyectoId, HitoNombre}
* **Llaves candidatas**: {HitoId}, {ProyectoId, HitoNombre}

**Revision**

* **Llave primaria (PK)**: CasoPruebaId
* **Superllave**: {RevisionId}, {HitoId, RevisionNombre}, {HitoId, RevisionNumero}
* **Llaves candidatas**: {RevisionId}, {HitoId, RevisionNombre}, {HitoId, RevisionNumero}

**CheckList**

* **Llave primaria (PK)**: PreCondicionId
* **Superllave**: {CheckListId}, {RevisionId, OrdenPreCondicion}
* **Llaves candidatas**: {PreCondicionId}, {RevisionId, OrdenPreCondicion}

**Paso**

* **Llave primaria (PK)**: PasoId
* **Superllave**: {PasoId}, {RevisionId, OrdenPaso}
* **Llaves candidatas**: {PasoId}, {RevisionId, OrdenPaso}

**Adjunto**

* **Llave primaria (PK)**: AdjuntoId
* **Superllave**: {AdjuntoId}, {RevisionId, DescripciónAdjunto}
* **Llaves candidatas**: {AdjuntoId}, {RevisionId, DescripciónAdjunto}

**Categoría**

* **Llave primaria (PK)**: CategoriaId
* **Superllave**: {CategoriaId}, {CategoriaNombre}
* **Llaves candidatas**: {CategoriaId}, {CategoriaNombre}

**Tipo**

* **Llave primaria (PK)**: TipoId
* **Superllave**: {TipoId}, {TipoNombre}
* **Llaves candidatas**: {TipoId}, {TipoNombre}

**Subtipo**

* **Llave primaria (PK)**: SubtipoId
* **Superllave**: {SubtipoId}, {SubtipoNombre, TipoId}
* **Llaves candidatas**: {SubtipoId}, {SubtipoNombre, TipoId}

**Relaciones y su Cardinalidad:**

**1. Usuario - RolAsignacion**

* **Relación**: Un usuario puede tener múltiples asignaciones de roles (relación entre Usuario y RolAsignacion).
* **Cardinalidad**:
  + **Usuario** (1) ↔ (N) **RolAsignacion**
  + Un **usuario** puede tener múltiples **asignaciones de roles**.
  + Una **asignación de rol** pertenece a un solo **usuario**.

**2. Rol - RolAsignacion**

* **Relación**: Un rol puede estar asignado a múltiples usuarios (a través de RolAsignacion).
* **Cardinalidad**:
  + **Rol** (1) ↔ (N) **RolAsignacion**
  + Un **rol** puede estar asignado a múltiples **asignaciones de roles**.
  + Una **asignación de rol** tiene un solo **rol**.

**3. Proyecto - RolAsignacion**

* **Relación**: Un proyecto puede estar relacionado con múltiples asignaciones de roles (relación entre Proyecto y RolAsignacion).
* **Cardinalidad**:
  + **Proyecto** (1) ↔ (N) **RolAsignacion**
  + Un **proyecto** puede estar relacionado con múltiples **asignaciones de roles**.
  + Una **asignación de rol** está vinculada a un solo **proyecto**.

**4. Rol - PermisoRol**

* **Relación**: Un rol puede tener múltiples permisos (a través de PermisoRol).
* **Cardinalidad**:
  + **Rol** (1) ↔ (N) **PermisoRol**
  + Un **rol** puede tener múltiples **permisos**.
  + Un **permiso de rol** está asociado a un solo **rol**.

**5. Permiso - PermisoRol**

* **Relación**: Un permiso puede estar relacionado con múltiples roles (a través de PermisoRol).
* **Cardinalidad**:
  + **Permiso** (1) ↔ (N) **PermisoRol**
  + Un **permiso** puede estar asignado a múltiples **roles**.
  + Un **permiso de rol** tiene un solo **permiso**.

**6. Proyecto - Hito**

* **Relación**: Un proyecto puede tener múltiples hitos.
* **Cardinalidad**:
  + **Proyecto** (1) ↔ (N) **Hito**
  + Un **proyecto** puede tener múltiples **hitos**.
  + Un **hito** pertenece a un solo **proyecto**.

**7. Hito - Revision**

* **Relación**: Un hito puede contener múltiples revisiones.
* **Cardinalidad**:
  + **Hito** (1) ↔ (N) **Revision**
  + Un **hito** puede tener múltiples **casos de prueba**.
  + Un **caso de prueba** pertenece a un solo **hito**.

**8. Revision - CheckList**

* **Relación**: Un caso de prueba puede tener múltiples precondiciones.
* **Cardinalidad**:
  + **Revision** (1) ↔ (N) **checklist**
  + Un **caso de prueba** puede tener múltiples **checklist**.
  + Un checklist pertenece a una sola **revision**.

**9. Revision - Paso**

* **Relación**: Un caso de prueba puede tener múltiples pasos.
* **Cardinalidad**:
  + **Revision** (1) ↔ (N) **Paso**
  + Un**a revision** puede tener múltiples **pasos**.
  + Un **paso** pertenece a una sola **revision**.

**10. Revision - Adjunto**

* **Relación**: Un caso de prueba puede tener múltiples adjuntos.
* **Cardinalidad**:
  + **Revision** (1) ↔ (N) **Adjunto**
  + Un **revision** puede tener múltiples **adjuntos**.
  + Un **adjunto** pertenece a una sola **revision**.

**11. Tester (Usuario) - Revision**

* **Relación**: Un usuario (tester) puede estar relacionado con múltiples revisiones.
* **Cardinalidad**:
  + **Usuario (Tester)** (1) ↔ (N) **Revision**
  + Un **tester** puede estar asignado a múltiples **revisiones**.
  + Una **revision** tiene un solo **tester** asignado.

**12. Tipo - Subtipo**

* **Relación**: Un tipo puede tener múltiples subtipos.
* **Cardinalidad**:
  + **Tipo** (1) ↔ (N) **Subtipo**
  + Un **tipo** puede tener múltiples **subtipos**.
  + Un **subtipo** pertenece a un solo **tipo**.

**13. Tipo - Revision**

* **Relación**: Un tipo puede tener múltiples revisiones.
* **Cardinalidad**:
  + **Tipo** (1) ↔ (N) **Revision**
  + Un **tipo** puede tener múltiples **revisiones**.
  + Un**a revision** pertenece a un solo **tipo**.

**14. Categoria - Revision**

* **Relación**: Una categoria puede tener múltiples revisiones.
* **Cardinalidad**:
  + **Tipo** (1) ↔ (N) **Revision**
  + Una **categoria** puede tener múltiples **revisiones**.
  + Un**a revision** pertenece a una sola **categoria**.

**Resumen de la Cardinalidad:**

| **Entidad 1** | **Relación** | **Entidad 2** | **Cardinalidad** |
| --- | --- | --- | --- |
| Usuario | tiene | RolAsignacion | 1 ↔ N |
| Rol | tiene | RolAsignacion | 1 ↔ N |
| Proyecto | tiene | RolAsignacion | 1 ↔ N |
| Rol | tiene | PermisoRol | 1 ↔ N |
| Permiso | tiene | PermisoRol | 1 ↔ N |
| Proyecto | tiene | Hito | 1 ↔ N |
| Hito | tiene | Revision | 1 ↔ N |
| Revision | tiene | CheckList | 1 ↔ N |
| Revision | tiene | Paso | 1 ↔ N |
| Revision | tiene | Adjunto | 1 ↔ N |
| Tester (Usuario) | asignado a | Revision | 1 ↔ N |
| Tipo | tiene | Subtipo | 1 ↔ N |
| Tipo | tiene | Revision | 1 ↔ N |
| Categoria | tiene | Revision | 1 ↔ N |

**2. Primera Forma Normalizada (1NF):**

En la **1NF**, eliminamos los valores no atómicos y nos aseguramos de que cada columna contenga valores indivisibles.

* **Separar valores atómicos**: Todas las columnas ya son atómicas. Cada fila contiene solo valores únicos.
* **Crear nuevas tablas para colecciones repetitivas**: Las colecciones, como Pasos y Adjuntos, pueden tener múltiples entradas para una sola revision.

**Tablas en 1NF:**

Tabla: Usuarios

- UsuarioId

- UsuarioNombre

- UsuarioEmail

- EsAdmin

Tabla: AsignacionRoles

- RolAsignacionId

- UsuarioId

- RolId

- ProyectoId

Tabla: Roles

- RolId

- RolNombre

Tabla: Permisos

- PermisoId

- PermisoNombre

Tabla: PermisosRol

- PermisoRolId

- RolId

- PermisoId

Tabla: Proyectos

- ProyectoId

- ProyectoNombre

- ProyectoTasaAprobación

- ProyectoEstado

Tabla: Hitos

- HitoId

- ProyectoId

- HitoNombre

- HitoInicio

- HitoFin

Tabla: Revision

- revisionId

- HitoId

- revisionNombre

- revisionNumero

- revisionTipo

- revisionSubTipo

- Comentarios

- ResultadoEsperado

- ResultadoObtenido

- TesterId

- EstadoRevision

Tabla: CheckList

- CheckListId

- revisionId

- Orden

- Descripción

- Estado

Tabla: Pasos

- PasoId

- RevisionId

- OrdenPaso

- DescripciónPaso

- EstadoPaso

Tabla: Adjuntos

- AdjuntoId

- RevisionId

- DescripciónAdjunto

- DocumentoAdjunto

Tabla: Categorias

- CategoriaId

- CategoriaNombre

Tabla: Tipos

- TipoId

- TipoNombre

Tabla: Subtipos

- SubtipoId

- SubtipoNombre

- TipoId

**3. Segunda Forma Normalizada (2NF):**

En la **2NF**, eliminamos dependencias parciales. Las columnas no clave deben depender completamente de la clave primaria.

* En tablas como revision, Hitos y Proyectos, ya estamos en 2NF porque todas las columnas dependen completamente de la clave primaria.
* Eliminamos cualquier dependencia parcial.

**Tablas en 2NF (sin cambios con respecto a 1NF):**

Las tablas en 2NF son esencialmente las mismas que en 1NF, ya que todas las dependencias parciales han sido eliminadas.

**4. Tercera Forma Normalizada (3NF):**

En la **3NF**, eliminamos dependencias transitivas, asegurándonos de que ninguna columna dependa indirectamente de la clave primaria.

* Por ejemplo, en la tabla Subtipos, la columna TipoId ya elimina una dependencia indirecta de TipoNombre.

**Tablas en 3NF (sin cambios significativos con respecto a 2NF):**

Las tablas en 3NF son prácticamente las mismas que en la 2NF, ya que hemos eliminado todas las dependencias transitivas.

**Base de Datos Final Normalizada (3NF):**

1. **Usuarios**:
   * UsuarioId (PK)
   * UsuarioNombre
   * UsuarioEmail
   * EsAdmin
2. **AsignacionRoles**:
   * RolAsignacionId (PK)
   * UsuarioId (FK -> Usuarios)
   * RolId (FK -> Roles)
   * ProyectoId (FK -> Proyectos)
3. **Roles**:
   * RolId (PK)
   * RolNombre
4. **Permisos**:
   * PermisoId (PK)
   * PermisoNombre
5. **PermisosRol**:
   * PermisoRolId (PK)
   * RolId (FK -> Roles)
   * PermisoId (FK -> Permisos)
6. **Proyectos**:
   * ProyectoId (PK)
   * ProyectoNombre
   * ProyectoTasaAprobación
   * ProyectoEstado
7. **Hitos**:
   * HitoId (PK)
   * ProyectoId (FK -> Proyectos)
   * HitoNombre
   * HitoInicio
   * HitoFin
8. **Revision**:
   * RevisionId (PK)
   * HitoId (FK -> Hitos)
   * Nombre
   * Numero
   * TipoId (FK -> Tipos)
   * SubTipoId (FK -> SubTipos)
   * Comentarios
   * ResultadoEsperado
   * ResultadoObtenido
   * TesterId (FK -> Usuarios)
   * Estado
9. **CheckList**:
   * CheckListId (PK)
   * RevisionId (FK -> Revision)
   * Orden
   * Descripción
   * Estado
10. **Pasos**:
    * PasoId (PK)
    * RevisiosnId (FK -> Revision)
    * OrdenPaso
    * DescripciónPaso
    * EstadoPaso
11. **Adjuntos**:
    * AdjuntoId (PK)
    * CasoPruebaId (FK -> Revision)
    * DescripciónAdjunto
    * DocumentoAdjunto
12. **Categorías**:
    * CategoriaId (PK)
    * CategoriaNombre
13. **Tipos**:
    * TipoId (PK)
    * TipoNombre
14. **Subtipos**:
    * SubtipoId (PK)
    * SubtipoNombre
    * TipoId (FK -> Tipos)