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.

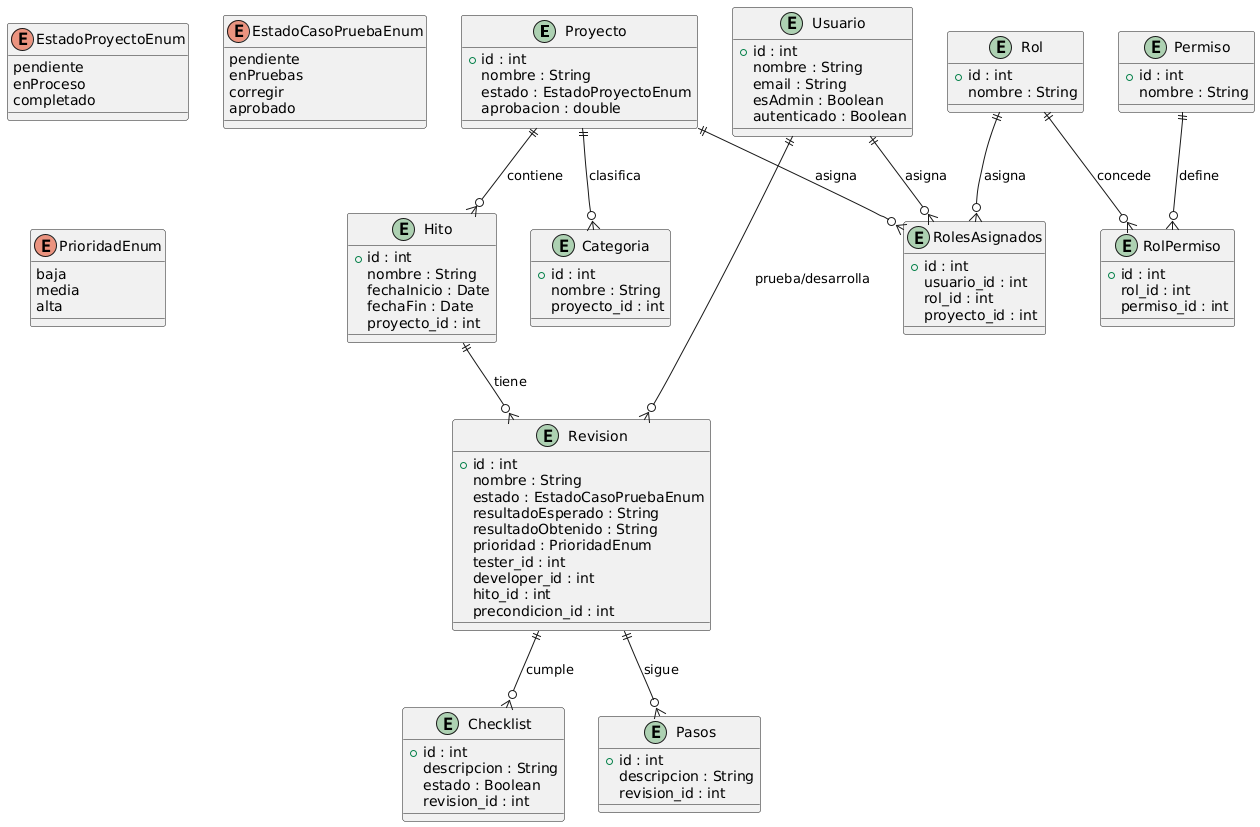
Atributos: Cada entidad tiene atributos con su tipo de datos, y las llaves primarias están marcadas con <<PK>>. Las llaves foráneas están marcadas con <<FK>>.

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

Descripción de Entidades y Atributos

[En esta sección se describen todas las entidades encontradas en el proceso de análisis para cada entidad se pueden listar sus 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. |

### Llaves

* Super llave: conjunto de uno o más atributos que "juntos" identifican de manera única a una entidad
* Llave candidata: es una super llave mínima
* Llave primaria: la seleccionada para identificar a los elementos de un conjunto de entidades.

Entidades y Atributos:

Proyecto

Atributos:

id (PK): Llave primaria.

nombre: Nombre del proyecto.

estado: Enum EstadoProyectoEnum (pendiente, enProceso, completado).

aprobacion: Porcentaje de aprobación.

Usuario

Atributos:

id (PK): Llave primaria.

nombre: Nombre del usuario.

email: Correo electrónico.

esAdmin: Booleano que indica si es administrador.

autenticado: Booleano que indica si el usuario está autenticado.

Rol

Atributos:

id (PK): Llave primaria.

nombre: Nombre del rol.

Permiso

Atributos:

id (PK): Llave primaria.

nombre: Nombre del permiso.

RolesAsignados

Atributos:

id (PK): Llave primaria.

usuario\_id (FK): Llave foránea hacia Usuario.

rol\_id (FK): Llave foránea hacia Rol.

proyecto\_id (FK): Llave foránea hacia Proyecto.

Hito

Atributos:

id (PK): Llave primaria.

nombre: Nombre del hito.

fechaInicio: Fecha de inicio.

fechaFin: Fecha de fin.

proyecto\_id (FK): Llave foránea hacia Proyecto.

Revision

Atributos:

id (PK): Llave primaria.

nombre: Nombre de la revisión.

hito\_id (FK): Llave foránea hacia Hito.

tester\_id (FK): Llave foránea hacia Usuario.

developer\_id (FK): Llave foránea hacia Usuario.

precondicion\_id (FK): Llave foránea hacia Checklist.

estado: Enum EstadoCasoPruebaEnum (pendiente, enPruebas, corregir, aprobado).

resultadoEsperado: Resultado esperado.

resultadoObtenido: Resultado obtenido.

prioridad: Enum PrioridadEnum (baja, media, alta).

Checklist

Atributos:

id (PK): Llave primaria.

descripcion: Descripción del checklist.

estado: Booleano que indica si está activo o no.

revision\_id (FK): Llave foránea hacia Revision.

Pasos

Atributos:

id (PK): Llave primaria.

descripcion: Descripción del paso.

revision\_id (FK): Llave foránea hacia Revision.

Categoria

Atributos:

id (PK): Llave primaria.

nombre: Nombre de la categoría.

proyecto\_id (FK): Llave foránea hacia Proyecto.

RolPermiso

Atributos:

id (PK): Llave primaria.

rol\_id (FK): Llave foránea hacia Rol.

permiso\_id (FK): Llave foránea hacia Permiso.

Relaciones y Cardinalidad:

Relación entre Proyecto y Usuario:

Tipo: Muchos a muchos.

Intermediaria: RolesAsignados.

Cardinalidad: Un proyecto puede tener múltiples usuarios asignados, y un usuario puede estar asignado a múltiples proyectos (dependiendo del rol).

Relación entre Proyecto y Hito:

Tipo: Uno a muchos.

Cardinalidad: Un proyecto contiene varios hitos, pero un hito solo pertenece a un proyecto.

Relación entre Hito y Revision:

Tipo: Uno a muchos.

Cardinalidad: Un hito puede tener muchas revisiones, pero una revisión pertenece solo a un hito.

Relación entre Usuario y Revision (Tester y Desarrollador):

Tipo: Muchos a uno (para cada rol).

Cardinalidad: Un usuario puede estar asignado como tester o desarrollador en varias revisiones, pero una revisión tiene solo un tester y un desarrollador.

Relación entre Revision y Checklist:

Tipo: Uno a muchos.

Cardinalidad: Una revisión puede tener múltiples checklist, pero un checklist pertenece solo a una revisión.

Relación entre Revision y Pasos:

Tipo: Uno a muchos.

Cardinalidad: Una revisión puede tener varios pasos a seguir, pero un paso pertenece solo a una revisión.

Relación entre Rol y Permiso:

Tipo: Muchos a muchos.

Intermediaria: RolPermiso.

Cardinalidad: Un rol puede tener varios permisos, y un permiso puede estar asociado a varios roles.

Relación entre Categoria y Proyecto:

Tipo: Muchos a uno.

Cardinalidad: Una categoría pertenece a un proyecto, pero un proyecto puede tener varias categorías.

Relaciones Encontradas

[Tras haber realizado un análisis del entorno se detectaron relaciones entre las entidades, se genera una lista de todas las relaciones encontradas.

**Relaciones**: la conexión que existe entre 2 entidades (verbo).]

Relación 1:

* Entidades relacionadas:
* Nombre
* Cardinalidad:

Relación 2:

## Diagrama E-R

[Una vez recolectada toda información necesaria podemos disponernos a realizar el diagrama E-R.

Este diagrama es una herramienta gráfica que sirve para describir el modelo lógico de la base de datos.]

### Principios de diseño

**Fidelidad:** se debe crear siempre un modelo que satisfaga las necesidades del problema, no sirve un modelo correcto si no cumple con la realidad que se pretende representar.

**Evitar redundancia:** una de las ventajas del diagrama e-r es que nos permite distinguir de una manera fácil y visual todos los entes y sus relaciones, de manera que es muy fácil identificar si un atributo se esta repitiendo en varias entidades o si una relación es innecesaria.

**Simplicidad:** siempre hay que procurar hacer el modelo tan simple como sea posible (sin olvidar la fidelidad) de manera que sea fácil de entender, fácil de extender y fácil de implementar.

**Escoger los elementos correctos:** es ocasiones es difícil identificar si una relación, elemento o atributo es correcto, para ello hay que analizar en perspectiva el diagrama y, por ejemplo, si se observa una entidad con solo un atributo y que únicamente presenta relaciones de 1, entonces probablemente estamos hablando de un atributo y no de una entidad.

**Relaciones n-arias:** Aún cuando se pueden presentar casos en los que una relación terciaria o n-aria parezca más conveniente, es mejor siempre pensar en términos de relaciones binarias únicamente. En el peor de los casos de que exista una relación n-aria forzosa, lo que se debe hacer es convertir esa relación R en entidad E y corregir todas las relaciones que tenía R de manera que ahora esa nueva entidad se relacione con todas las entidades que anteriormente esta.

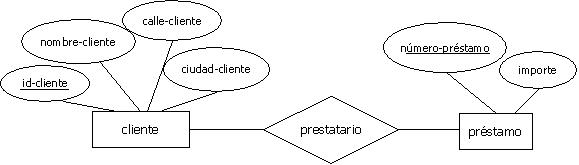
### 

### Notación diagrama E-R

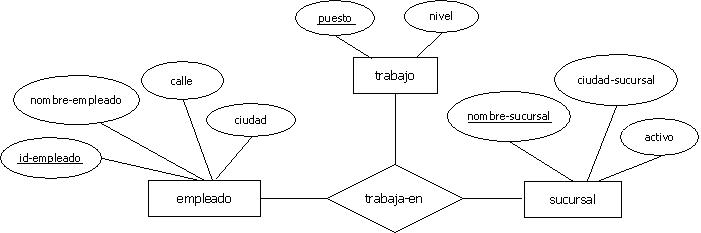


Notación Diagrama E-R

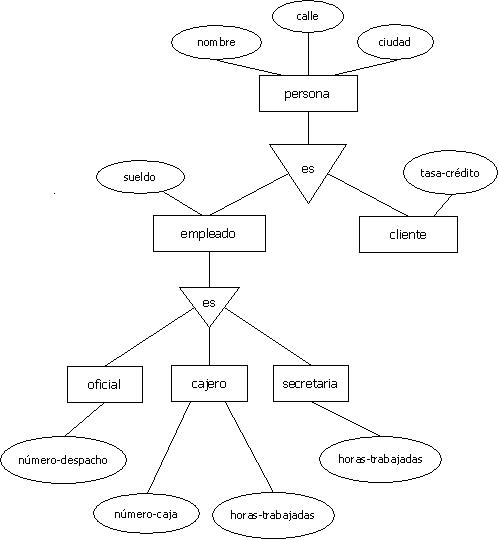
### Ejemplos Diagrama E-R



[Primer ejemplo de un diagrama Entidad-Relación]

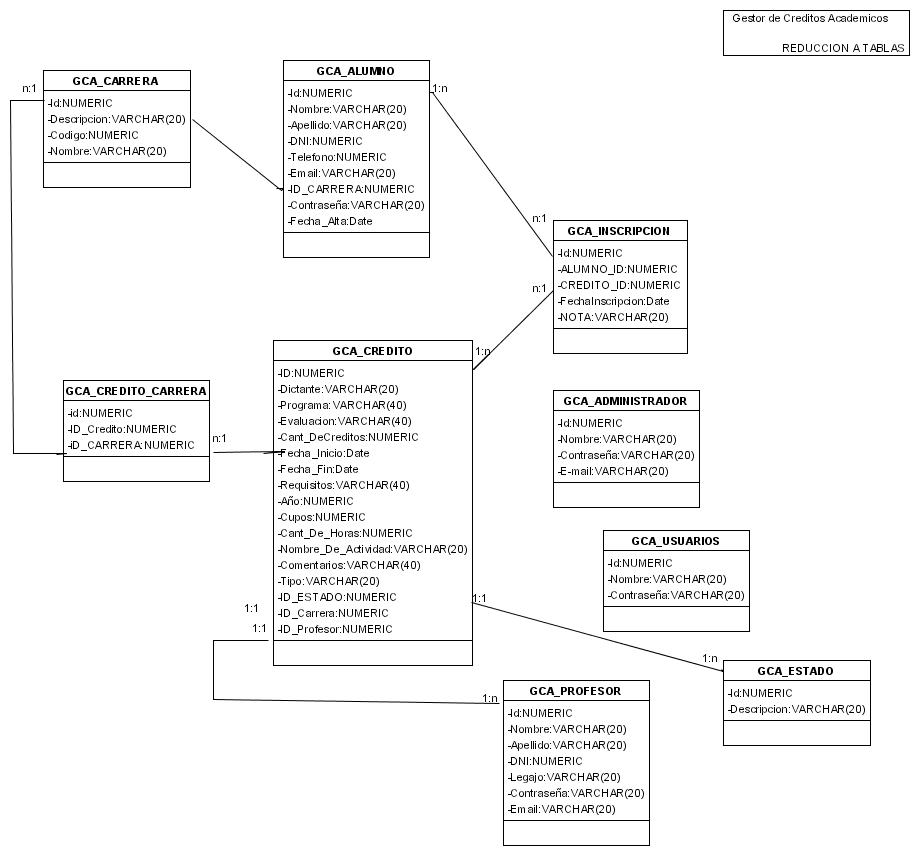


[Segundo ejemplo de un diagrama Entidad-Relación]



[Tercer ejemplo: Especialización y Generalización]

## Modelo Relacional



## Comentarios sobre el Modelo Relacional

[Esta sección está destinada a todos los comentarios y aclaraciones que se deban hacer sobre el modelo Relacional generado]

## Normalización

[Una vez creadas las tablas hay que verificarlas y revisar si aún se puede reducir u optimizar de alguna manera, para esto se aplican algunos principios de normalización para mitigar los siguientes problemas:

**Redundancia**: la información se repite innecesariamente en muchas tuplas.

**Anomalías de actualización**: cuando al cambiar la información en una tupla se descuida el actualizarla en otra.

**Anomalías de eliminación**: si un conjunto de valores llega a estar vacíos y se llega a perder información relacionada como un efecto de la eliminación.]

## Justificación de Forma normal adoptada

[Luego de adoptar alguna forma normal (primera forma normal, segunda forma normal, tercera forma normal o bien la Forma normal de Boyce-Codd (BCNF)) se debe justificar por que se selecciono esa forma normal y aclarar sus mejorías hacia el modelo relacional.]

Especificación de la Distribución de Datos

[En esta sección se especifica el modelo de distribución de datos indicando la ubicación de los manejadores de bases de datos o sistemas de archivos, así como los distintos elementos de la estructura física de datos (base de datos, tablas, índices), en los nodos correspondientes.

Para elaborar esta sección el Arquitecto trabajará en conjunto con el Especialista Técnico de Base de Datos.]