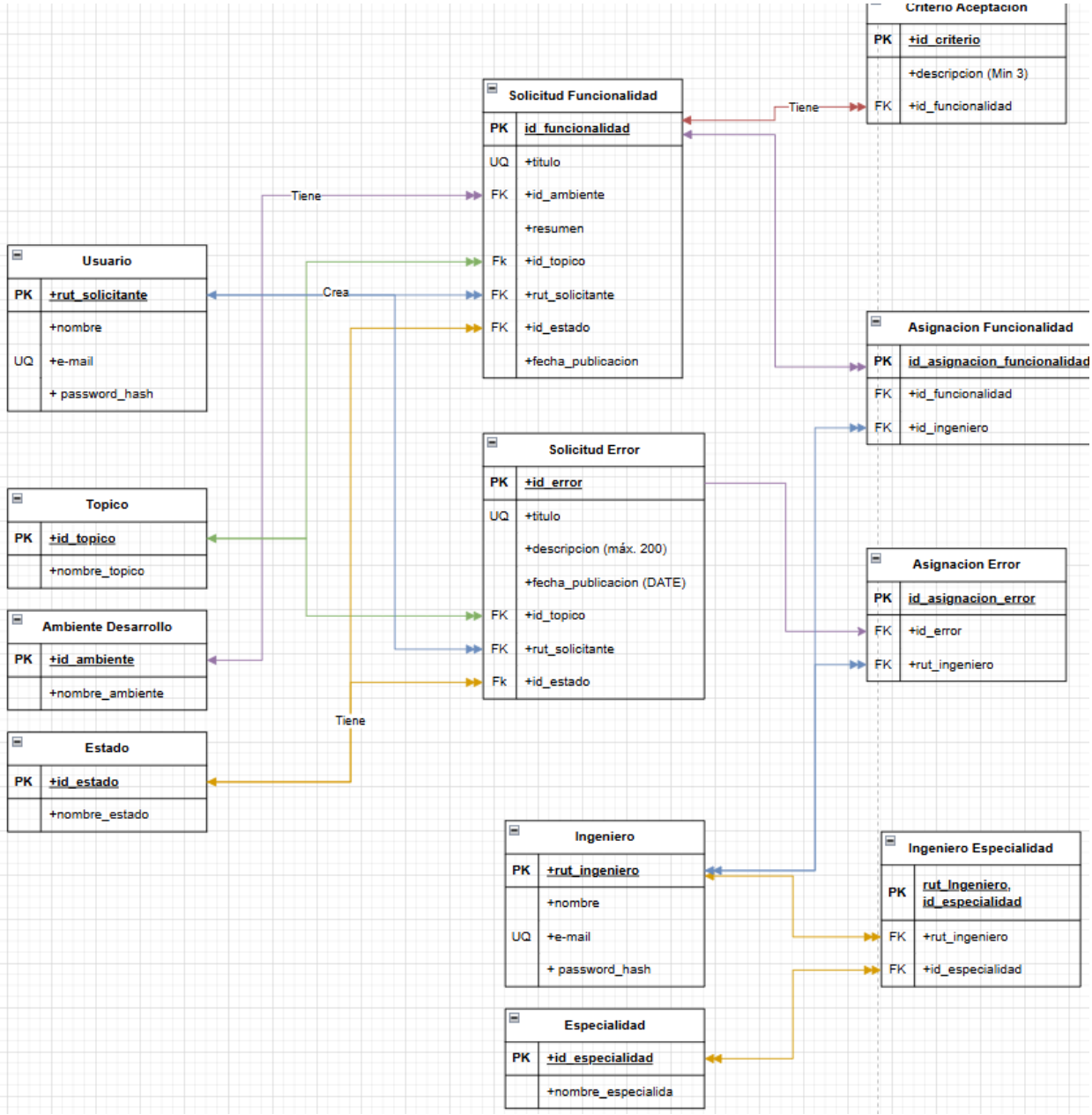


Justificación de la Normalización del Modelo

El modelo de base de datos presentado fue diseñado y construido aplicando directamente los principios de la normalización, lo que resultó en una estructura que cumple de manera inherente con la **Tercera Forma Normal (3FN)**.

La razón principal es que cada tabla se enfoca en representar una única entidad bien definida (como `Usuario`, `Solicitud Funcionalidad` o `Tópico`), eliminando así la redundancia de datos. Todos los atributos en cada tabla dependen de manera directa y exclusiva de su clave primaria. Por ejemplo, en la tabla `Solicitud Funcionalidad`, atributos como el `título` y

el `resumen` dependen únicamente del `id_funcionalidad`. La información del solicitante o del estado no se almacena directamente en esta tabla; en su lugar, se utilizan claves foráneas (`rut_solicitante`, `id_estado`) que apuntan a sus respectivas tablas (`Usuario`, `Estado`), evitando así cualquier dependencia transitiva.

Al seguir esta estructura desde el diseño inicial, se garantiza que la base de datos sea eficiente, escalable y libre de las anomalías de modificación que la normalización busca resolver.

Proceso de Verificación de la Normalización

Para que una tabla esté en **Tercera Forma Normal (3FN)**, debe cumplir secuencialmente con las siguientes reglas:

1. **Primera Forma Normal (1FN):**
    - Tener una clave primaria única.
    - Todos los valores en sus columnas deben ser atómicos (indivisibles).
    - No debe haber grupos de columnas repetidos.
2. **Segunda Forma Normal (2FN):**
    - Debe estar en 1FN.
    - Todos los atributos que no forman parte de la clave primaria deben depender *completamente* de la clave primaria. (Esto es crucial para claves compuestas).
3. **Tercera Forma Normal (3FN):**
    - Debe estar en 2FN.
    - Ningún atributo no clave debe depender de otro atributo no clave. Esto se conoce como evitar **dependencias transitivas**.

Análisis Detallado por Tabla

1. Tablas de Catálogo (Entidades Simples)

Estas tablas almacenan información descriptiva y son la base del modelo.

- **Tabla: Usuario**
    - **Clave Primaria (PK):** `rut_solicitante`
    - **1FN:** Cumple. Tiene una PK, y los atributos `nombre`, `e-mail` y `password_hash` son atómicos.
    - **2FN:** Cumple. La clave primaria es simple, por lo que no pueden existir dependencias parciales.

    - **3FN:** Cumple. Los atributos `nombre`, `e-mail` y `password_hash` dependen directamente de `rut_solicitante` y no dependen entre sí. No hay dependencias transitivas.
- **Tabla: Ingeniero**
    - **PK:** `rut_ingeniero`
    - **1FN, 2FN, 3FN:** Cumple por las mismas razones que la tabla `Usuario`.
- **Tablas: Tópico, Ambiente Desarrollo, Estado, Especialidad**
    - **PK:** `id_topico`, `id_ambiente`, `id_estado`, `id_especialidad`
    - **1FN:** Cumplen. Cada una tiene su PK y un único atributo de nombre que es atómico.
    - **2FN:** Cumplen. Sus claves primarias son simples.
    - **3FN:** Cumplen. Al tener un solo atributo no clave (`nombre_...`), es imposible que exista una dependencia transitiva. El nombre depende directamente del ID.

2. Tablas Transaccionales (Eventos y Registros)

Estas tablas registran las acciones y solicitudes en el sistema.

- **Tabla: Solicitud Funcionalidad**
    - **PK:** `id_funcionalidad`
    - **1FN:** Cumple. Tiene una PK y todos sus campos son atómicos.
    - **2FN:** Cumple. La clave primaria es simple.
    - **3FN:** Cumple. Los atributos como `titulo`, `resumen` y `fecha_publicacion` dependen directamente del `id_funcionalidad`. Las claves foráneas (`id_ambiente`, `rut_solicitante`, etc.) también dependen directamente de la solicitud. No hay un atributo no clave que dependa de otro no clave. Por ejemplo, `resumen` no depende de `titulo`, ambos describen a la funcionalidad.
- **Tabla: Solicitud Error**
    - **PK:** `id_error`
    - **1FN, 2FN, 3FN:** Cumple por la misma lógica que `SolicitudFuncionalidad`.
- **Tabla: Criterio Aceptación**
    - **PK:** `id_criterio`
    - **1FN:** Cumple.
    - **2FN:** Cumple.
    - **3FN:** Cumple. El atributo `descripcion` depende directamente de `id_cterio`. La clave foránea `id_funcionalidad` establece a qué solicitud pertenece el criterio, por lo que también depende de `id_criterio`. No hay dependencias transitivas.
- **Tabla: Reseña**
    - **PK:** `id_reseña`

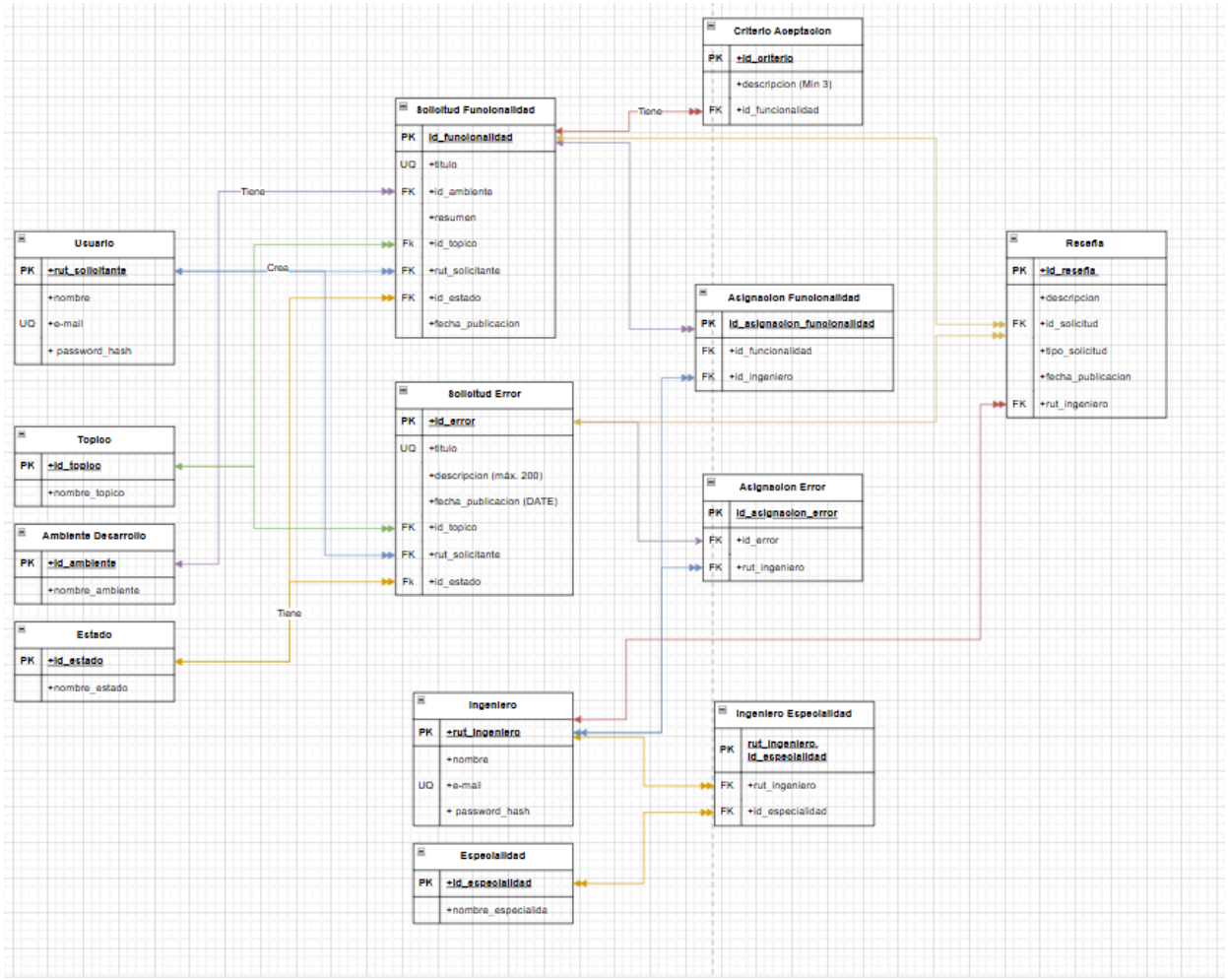
    - **1FN:** Cumple. Tiene una PK (`id_reseña`), y sus atributos (`descripcion`, `fecha_publicacion`, `tipo_solicitud`) son atómicos.
    - **2FN:** Cumple. La clave primaria es simple, por lo que no pueden existir dependencias parciales.
    - **3FN:** Cumple. Los atributos no clave (`descripcion`, `fecha_publicacion`, `tipo_solicitud`) dependen directamente de la PK `id_reseña`. Las claves foráneas (`id_solicitud`, `rut_ingeniero`) también dependen directamente de `id_reseña`, ya que identifican la solicitud y el ingeniero asociados *a esta reseña específica*. No existen dependencias transitivas.

3. Tablas de Unión (Relaciones Muchos a Muchos)

Estas tablas conectan otras tablas.

- **Tabla: Ingeniero Especialidad**
    - **PK (Compuesta):** (`rut_ingeniero`, `id_especialidad`)
    - **1FN:** Cumple.
    - **2FN:** Cumple. La tabla no tiene atributos no clave, por lo que no puede haber dependencias parciales. Su único propósito es vincular las dos claves.
    - **3FN:** Cumple. Al no tener atributos no clave, no pueden existir dependencias transitivas.
- **Tablas: Asignación Funcionalidad / Asignacion Error**
    - **PK:** `id_asignacion_funcionalidad`, `id_asignacion_error`
    - **1En:** Cumplen.
    - **2FN:** Cumplen. Utilizan una clave primaria simple (subrogada), por lo que no hay dependencias parciales.
    - **3FN:** Cumplen. Sus únicos otros atributos son las claves foráneas que dependen directamente del ID de la asignación. Por ejemplo, `id_ingeniero` depende del `id_asignacion_funcionalidad`, no del `id_funcionalidad`.

Como se aprecia, el diseño original sufrió 3 cambios principales debido a que ya se encontraba normalizado, se agregaron a las tablas usuario e ingeniero el atributo contraseña, y se creo la tabla (en 3FN) de reseñas para almacenar estas últimas.


Criterio Aceptacion
PK +id_criterio
+descripcion (Min 3)
FK +id_funcionalidad
Tiene
Solicitud Funcionalidad
PK id_funcionalidad
UQ +titulo
FK +id_ambiente
+resumen
Fk +id_topico
FK +rut_solicitante
FK +id_estado
+fecha_publicacion
Tiene
Usuario
PK +rut_solicitante
+nombre
UQ +e-mail
+ password_hash
Crea
Asignacion Funcionalidad
PK id_asignacion_funcionalidad
FK +id_funcionalidad
FK +id_ingeniero
Reseña
PK +id_reseña
+descripcion
FK +id_solicitud
+tipo_solicitud
+fecha_publicacion
FK +rut_ingeniero
Solicitud Error
PK +id_error
UQ +titulo
+descripcion (máx. 200)
+fecha_publicacion (DATE)
FK +id_topico
FK +rut_solicitante
Fk +id_estado
Topico
PK +id_topico
+nombre_topico
Ambiente Desarrollo
PK +id_ambiente
+nombre_ambiente
Asignacion Error
PK id_asignacion_error
FK +id_error
FK +rut_ingeniero
Tiene
Estado
PK +id_estado
+nombre_estado
Ingeniero
PK +rut_ingeniero
+nombre
UQ +e-mail
+ password_hash
Ingeniero Especialidad
PK rut_ingeniero, id_especialidad
FK +rut_ingeniero
FK +id_especialidad
Especialidad
PK +id_especialidad
+nombre_especialida