**SISTEMA DE GESTION DE HOSPITAL**

**SISTEMAS TRANSACCIONALES Y BASES DE DATOS**

**PRESENTADO POR:**

**DYLAN SANTIAGO RODRIGUEZ PORTILLA**

**SAN JUAN DE PASTO, 19 NOVIEMBRE DEL 2024**

**UNIVERSIDAD COOPERATIVA DE COLOMBIA**

**CAMPUS PASTO**

**Cardinalidad Compleja: Relaciones Uno a Muchos y Muchos a Muchos**

En el sistema de gestión hospitalaria, las relaciones de **uno a muchos** y **muchos a muchos** son esenciales para modelar cómo interactúan las entidades entre sí.

**Relación de uno a muchos**:  
La relación **Médico a Cita** es un ejemplo de una relación de uno a muchos. Un **médico** puede atender muchas **citas**, pero cada **cita** solo puede estar asociada a un **médico**. Esta relación permite gestionar eficientemente la agenda de los médicos y las citas asignadas a cada uno. Además, facilita la visualización de las citas programadas para un médico en particular.

**Justificación**:  
Esta relación es necesaria para organizar la carga de trabajo de los médicos, permitiendo asignar múltiples citas a un solo médico, y asegurando que cada cita esté correctamente asignada a un único médico para evitar conflictos o errores en la programación.

**Relación de muchos a muchos**:  
Un ejemplo clásico de relación de muchos a muchos es la relación entre **Paciente y Tratamiento**. Un **paciente** puede recibir múltiples **tratamientos**, y un **tratamiento** puede ser administrado a varios **pacientes**. Este tipo de relación es adecuada para representar la diversidad de tratamientos que un paciente puede recibir, y cómo un mismo tratamiento puede ser utilizado por diferentes pacientes.

**Justificación**:  
Esta relación refleja la flexibilidad de los tratamientos en el sistema hospitalario, donde un tratamiento puede ser común a varios pacientes, como en el caso de procedimientos quirúrgicos o terapias estándar. Esto facilita la administración y seguimiento de la evolución de los pacientes en su tratamiento.

**Relación Recursiva: Mentor-Aprendiz entre Médicos**

La relación **mentor-aprendiz** entre médicos se presenta como una relación recursiva, donde un **médico** puede ser mentor de otro **médico**. Cada **médico** puede tener un **mentor** asignado, y esa relación puede ser útil en la formación profesional dentro del hospital.

**Justificación**:  
Este tipo de relación es relevante para los programas de formación médica, donde médicos con más experiencia pueden guiar a médicos más jóvenes o inexpertos. La implementación de esta relación permite modelar mejor la jerarquía profesional dentro del hospital, además de proporcionar una vía para registrar el progreso de los médicos a lo largo de su carrera.

**Asignación de Habitaciones: Unicidad en la Ocupación**

Es importante que cada **habitación** esté ocupada solo por un **paciente** a la vez. La relación entre **Paciente y Habitación** es crucial para garantizar que no haya conflictos de asignación.

**Justificación**:  
Esto asegura que cada habitación esté asignada a un solo paciente en un momento dado, evitando la sobreocupación de las habitaciones, lo cual podría comprometer la comodidad, la privacidad y la seguridad de los pacientes. Además, permite un manejo eficiente de los recursos del hospital, asegurando que cada habitación se administre de manera óptima.

**Justificación de Entidades y Funcionalidades**

**Entidades de Roles**

La **entidad de roles** es fundamental para el sistema, ya que permite asignar diferentes niveles de permisos a los usuarios del sistema, como **administradores**, **médicos**, **enfermeros** y **recepcionistas**. Cada rol tendrá permisos específicos, como la capacidad de acceder a ciertos módulos del sistema o modificar ciertos datos.

**Justificación**:  
Los roles garantizan la seguridad y la organización dentro del sistema. Por ejemplo:

**Administrador**: Permite gestionar todos los aspectos del sistema, como usuarios, citas y recursos hospitalarios.

**Médico**: Solo tiene acceso a los pacientes asignados a él, sus diagnósticos y tratamientos, sin poder modificar la configuración global del sistema.

**Enfermero**: Tiene acceso limitado a la información del paciente, principalmente sobre su estado de salud y tratamiento, pero no puede modificar la estructura del sistema.

Esto asegura que la información sea tratada de manera confidencial y que solo el personal autorizado tenga acceso a determinadas acciones.

**Hospitalizaciones de Pacientes**

La **entidad de habitación** es crucial para manejar las hospitalizaciones de los pacientes. Cada paciente puede ser asignado a una habitación específica durante su hospitalización, y es importante registrar el estado de cada habitación (ocupada o libre).

**Justificación**:  
El sistema debe garantizar que cada habitación esté ocupada por un solo paciente en un momento determinado, lo que implica la gestión de recursos físicos dentro del hospital. También es necesario para controlar la duración de la hospitalización y facilitar la asignación de habitaciones basadas en la gravedad del caso o las necesidades del paciente.

**¿Cómo asegurarías que un paciente no tenga dos citas asignadas para la misma fecha y hora?**

Para garantizar que un paciente no tenga dos citas asignadas en el mismo horario, se puede implementar una restricción de unicidad en el sistema, de manera que se verifique si ya existe una cita para el mismo paciente y fecha antes de agregar una nueva. También se puede hacer mediante una validación en la interfaz de usuario, donde el sistema avise si el paciente ya tiene una cita para ese horario.

**Solución**:  
En la base de datos, agregar una restricción de clave única sobre la combinación de **paciente** y **fecha**. Además, en el sistema, antes de crear una nueva cita, realizar una consulta para verificar que no exista una cita en esa fecha para el mismo paciente.

**¿Cómo modelarías el sistema para que los registros de visitas de los pacientes mantengan integridad histórica y puedan ser revisados sin alterarse?**

Para mantener la integridad histórica, se puede implementar un sistema de **auditoría** que registre todas las visitas de los pacientes con un **timestamp** (marca de tiempo) y un identificador único para cada evento. Así, aunque los datos puedan cambiar con el tiempo, siempre se podrá acceder a la versión histórica de los registros.

**Solución**:  
Crear una tabla o colección separada para el historial de visitas de cada paciente. Cada registro de visita debe incluir detalles como el médico que atendió al paciente, diagnóstico, tratamiento y fecha/hora de la consulta.

**¿Cómo optimizarías el sistema si la base de datos llegara a manejar miles de citas y tratamientos simultáneamente?**

Si el sistema debe manejar miles de citas y tratamientos simultáneamente, se pueden aplicar varias técnicas de optimización, tales como:

**Particionamiento de tablas**: Dividir las tablas de citas y tratamientos en particiones basadas en fechas o rangos específicos (por ejemplo, particionar citas por mes o año).

**Replicación**: Utilizar bases de datos replicadas para distribuir la carga de trabajo entre varios servidores y mejorar la disponibilidad.

**Sharding**: Dividir la base de datos en fragmentos más pequeños distribuidos entre diferentes servidores, de manera que cada fragmento contenga una parte de los datos.

**Índices eficientes**: Crear índices adecuados para las búsquedas frecuentes, como por ejemplo, índices sobre los campos **fecha** y **paciente** en las tablas de citas.

**Solución**:  
Implementar particionamiento en las tablas de citas y tratamientos por fecha, y considerar el uso de replicación para asegurar que el sistema pueda manejar grandes volúmenes de datos sin pérdida de rendimiento. También se pueden usar vistas y cachés para consultas frecuentes.