PROYECTO FINAL DE BASE DE DATOS
Ejercicio: Sistema de Gestión de Hospital

Desarrollar un sistema para gestionar un hospital que incluya pacientes, médicos y turnos de consulta médica.

- **Modelo del Sistema:** Representar entidades como Pacientes, Médicos y Turnos con fecha y horario.
- **Datos inciales:** La base de datos inicial debe contener al menos diez (10) pacientes y diez (10) médicos.

Ejercicio: Consignas para el Menú de Gestiones

1. Gestión de Pacientes: Registrar, actualizar, ver y eliminar información de pacientes.
2. Gestión de Doctores: Agregar, actualizar y ver detalles de los doctores, incluyendo especialidades.
3. Manejo de Turnos: Programar, actualizar o cancelar turnos.
4. Búsquedas Avanzadas: Recuperar pacientes o médicos mediante diferentes atributos (nombre, especialidad, ID de paciente). Tener en cuanta que la búsqueda puede ser por parte del texto a buscar.
5. Reporte de turnos: Mostrar un reporte donde muestre los tres (3) médicos que más turnos tienen y la cantidad de turnos de cada uno.
6. Cancelación de turnos: Se debe poder cancelar para un médico dado y un rango de fechas dado.

DESARROLLO:

Restricciones

- Cada paciente tiene un identificador único (pacienteId) y puede estar asociado a un médico mediante turnos.
- Cada médico tiene un identificador único (medicoId) y puede tener múltiples turnos programados con diferentes pacientes.
- Un turno conecta a un paciente y un médico en una fecha y hora específicas. Los turnos se identifican de manera única por un idTurno.
- Los pacientes pueden buscarse por atributos como nombre o DNI.
- Los médicos pueden buscarse por atributos como nombre, especialidad o matrícula.
- Un médico puede tener varios turnos el mismo día, pero no dos turnos al mismo tiempo.
- Los turnos se pueden cancelar o reprogramar según las necesidades del sistema.

Esquema DB

Se diseñaron las tablas de acuerdo a las entidades requeridas y para evitar redundancias cumpliendo con la 3FN.

`Pacientes<pacienteId, nombre, apellido, dni, telefono, email, direccion, obraSocial>`

`Medicos<medicoId, nombre, apellido, matricula, telefono, especialidad>`

`Turnos<idTurno, idPaciente, idMedico, fecha, hora>`

1. Dependencias Funcionales (DFs)

TABLA PACIENTES
`pacienteID` **-> nombre, apellido, dni, teléfono, email, direccion, obraSocial**
La **clave primaria** `pacienteID` identifica de forma única a un paciente.

`dni` **→ nombre, apellido, dni, teléfono, email, direccion, obraSocial**
El `dni` (documento) también es único y podría determinar todos los datos del paciente.

TABLA MÉDICOS
`medicoId` **-> nombre, apellido, matricula, especialidad, teléfono**
La **clave primaria** `medicoId` identifica de forma única a un médico.

`matricula` **-> nombre, apellido, especialidad, teléfono**
El número de matrícula también es único y podría determinar todos los datos del médico.

TABLA TURNOS
`idTurno` **-> idPaciente, idMedico, fecha, hora**
La **clave primaria** `idTurno` identifica un turno de forma única.

2. Claves Candidatas

- **Pacientes**:
  Clave primaria: `pacienteID`
  Clave candidata alternativa: `dni`

- **Médicos**:
  Clave primaria: `medicoId`
  Clave candidata alternativa: `matricula`

- **Turnos**:
  Clave primaria: `idTurno`
  Clave candidata compuesta: `pacienteID, medicoId, fecha, hora`
  Aunque `idTurno` es la clave primaria, la combinación de pacienteID, medicoId, fecha y hora también identifica un turno de forma única.

3. Normalización a Tercera Forma Normal (3FN)

- Las tres tablas cumplen con la Primera Forma Normal (1FN), ya que no hay atributos multivaluados o repetidos.

- Cumplen también con la Segunda Forma Normal (2FN), porque los atributos no clave dependen completamente de la clave primaria en cada tabla.

- Para la Tercera Forma Normal (3FN):

  - Verificamos que en la tabla Pacientes no hay dependencias funcionales transitivas, ya que todos los atributos dependen directamente de la clave primaria `pacienteID`, ya está en 3FN.

  - Revisando la tabla Médicos, no hay dependencias funcionales transitivas, ya que todos los atributos dependen directamente de la clave primaria `medicoId`, así que es 3FN.

- Finalmente en la tabla Turnos, no hay dependencias transitivas, ya que todos los atributos dependen directamente de la clave primaria `idTurno`, también es 3FN.

No hay redundancias significativas, cada tabla tiene una clave primaria bien definida y las dependencias funcionales son claras y perfectamente representadas, quedando la composición final de las tablas de la siguiente forma:

1 - **Tabla `TURNOS`**
- `idTurno` (Clave primaria)
- `idPaciente` (Clave foránea: Referencia a PACIENTES(pacienteId))
- `idMedico` (Clave foránea: Referencia a MEDICOS(medicoId))
- `fecha`
- `hora`

Clave primaria: `idTurno`

2 - **Tabla `PACIENTES`**
- `pacienteId` (Clave primaria)
- `nombre`
- `apellido`
- `dni`
- `telefono`
- `email`
- `direccion`
- `obraSocial`

Clave primaria: `pacienteId`

3 - **Tabla `MEDICOS`**
- `medicoId` (Clave primaria)
- `nombre`
- `apellido`
- `matricula`
- `telefono`
- `especialidad`

Clave primaria: `medicoId`

Funcionamiento del Proyecto y manejo de archivos

El proyecto se ejecuta mediante un archivo principal `app.py` que accede a cuatro archivos más: `manejo_medicos.py`, `manejo_pacientes.py`, `manejo_turnos.py` y `busqueda_avanzada.py` (a su vez los cuatro acceden al archivo `conexion.py` que los conecta con la base de datos). Estos archivos contienen funciones para realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar), es decir que se utiliza a través de ellos el manejo de las entidades de la base de datos, generadas por la creación de objetos con los archivos `medicos.py`, `pacientes.py` y `turnos.py`.

Junto con los archivos python se encuentran tres archivos sql, que se usan como referencia para crear tablas y métodos en el Workbench (`tablas.sql`, `datos tablas.sql` y `procedimentos.sql`).