Gestión de Datos

Trabajo Práctico

2º Cuatrimestre

CLÍNICA – FRBA

Estrategia

Estrategia

En el presente documento se plasmarán todas las decisiones tomadas para el modelado de la base de datos a utilizar, junto con la implementación de la aplicación asociada.

Importante: Se considera que quien ejecute la aplicación, deberá tener previamente creada la base de datos “GD2C2016” junto con la tabla maestra entregada por la cátedra.

Decisiones tomadas en el diseño del DER (ver DER adjunto):

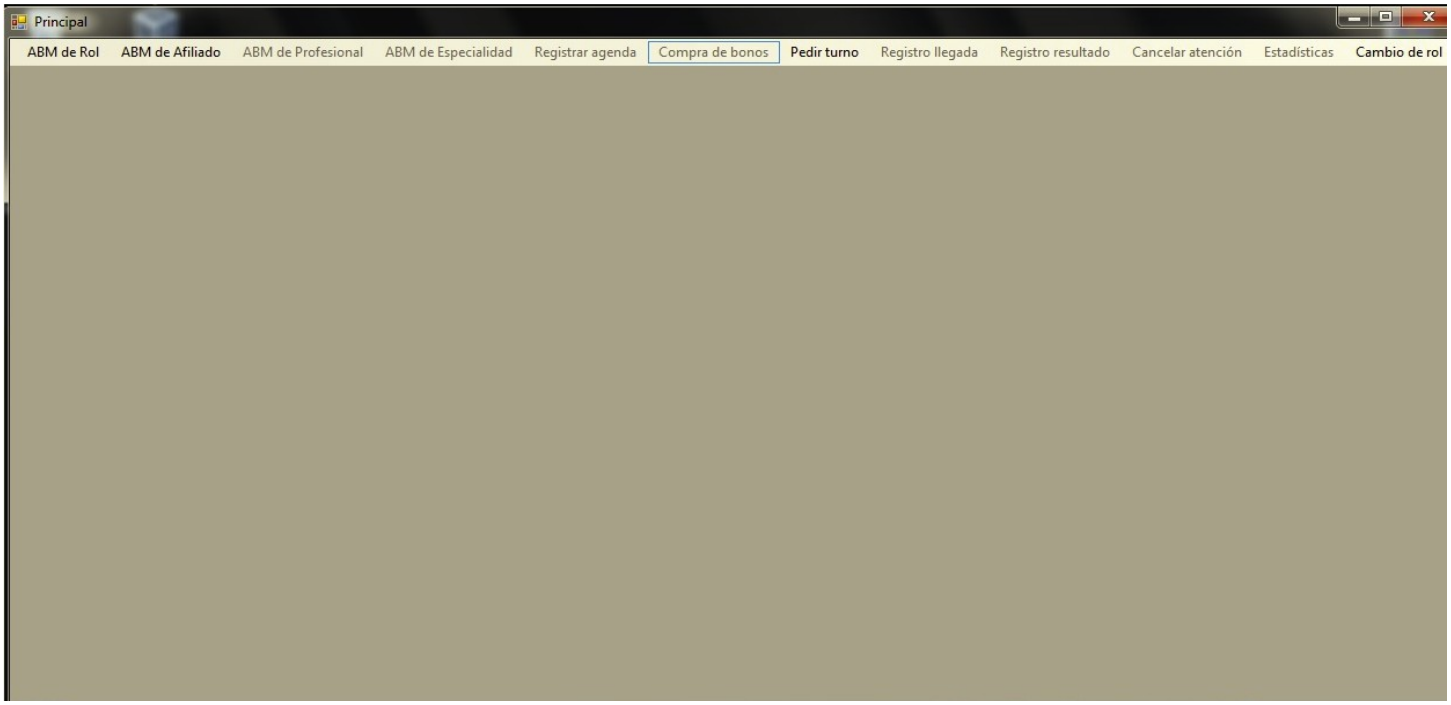
- Para la asignación de roles a usuarios, se creó una tabla intermedia “rol_de_usuario”, que facilita el acceso a todos los roles con los que un usuario puede ingresar al sistema.
- La tabla “funcionalidad_de_rol” contiene la relación entre un rol específico y las distintas funcionalidades que el mismo posee.
- Las tablas “afiliado”, “profesional” y “administrativo” contienen información necesaria para el momento en el que el usuario ingrese con alguno de estos roles. Vale aclarar que, dentro de la tabla “afiliado”, se puede referenciar a otro de la misma especie (a través del campo “afiliado_encargado”) para indicar pertenencia a un grupo familiar.
- La tabla “especialidad_de_profesional” incluye los campos necesarios para establecer la relación entre un profesional y las especialidades que atiende.
- La tabla “usuario” posee todos los datos básicos que debe tener una persona para ingresar al sistema y ser identificada, sumado al campo “intentos_fallidos”, que controla para seguridad interna, las veces que un usuario ingresó mal su contraseña y en caso de alcanzar el valor de 3, inhabilita al usuario.
- Se utilizó la tabla “Agenda”, con una fecha de inicio (“agenda_inicio”) y otra de caducidad (“agenda_fin”) para establecer la conexión entre los profesionales y los días y horarios que estos atienden.
- Se creó la tabla “afiliado_historial” para poder registrar la información necesaria al dar de baja un afiliado. De forma semejante, se creó la tabla “cancelación_turno” para poder acceder fácilmente a los datos de una posible cancelación.
- La tabla “plan”, referenciada mediante la FK “plan_id” desde “afiliado”, es la

encargada de contener toda la información correspondiente a un plan.

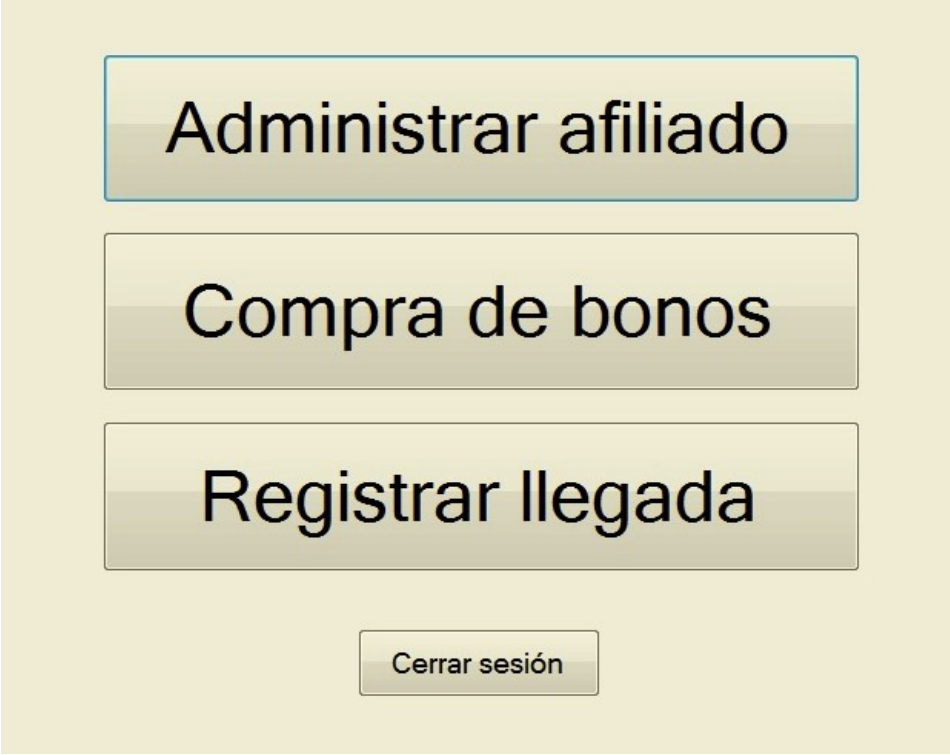
- La tabla “compra_bono”, referenciada mediante la FK “compra_bono_id” desde “bono”, incluye la información correspondiente a la compra de un bono consulta. A su vez, contiene una referencia al plan del afiliado que realizó la compra para conocer el importe final del bono.
- La tabla “bono”, referenciada mediante la FK “bono_id” desde la tabla “turno”, y referenciando al afiliado poseedor del bono (FK afiliado_id), es la tabla encargada de persistir todos los datos de un bono consulta. El mismo posee un campo “plan_id” para un acceso más rápido a la hora de encontrar el plan pertinente a un bono.
- La tabla “turno” es la encargada de persistir un turno con posibilidad de cancelación (tabla “cancelación_turno”) o efectivización (tabla “consulta_médica”).
- La tabla “consulta_médica” es la tabla que evidencia la confirmación y atención de un turno. Tiene en ella las referencias al profesional que atendió, al afiliado, al turno y a la agenda mediante FK.
- La tabla “día” evidencia mediante sus campos, el horario de atención de un profesional junto con los días de la semana que trabaja. Referencia a la agenda de un profesional para poder determinar la relación mencionada.
- La tabla “especialidad” contiene los datos necesarios a conocer de una especialidad, junto con una referencia a la tabla “tipo_especialidad” para conocer la misma.
- La tabla “funcionalidad” y “rol” son las tablas encargadas de persistir todas las funcionalidades y roles del sistema. En ambas se declaró un id del tipo identity como PK en lugar de su respectivo nombre, para que en un futuro, la agregación de roles o funcionalidades no ralentice las consultas al motor de base de datos (resulta más óptimo una búsqueda por int que por varchar(50)).

Decisiones tomadas en el desarrollo de la aplicación desktop:

- En la apertura de la aplicación, al loguearse con un usuario y elegir uno de los roles habilitados para el mismo, el programa toma estos valores ingresados y los asigna de forma permanente a dos variables, las cuales volverán a resetear sus valores una vez se regrese a la pantalla de login.
- Dentro de la creación de interfaces, se ponderó el hecho de que una persona con un rol asignado no pueda ver las funcionalidades de otro rol, por sobre la agregación de funcionalidades a un rol específico (ver imágenes 1 y 2 adjuntadas en la estrategia). Al tener interfaces distintas para evitar la visualización de funcionalidades de otro rol, si una funcionalidad es agregada o eliminada de un rol, gracias a la arquitectura de 3 capas utilizada, solo se debería agregar o quitar botones a la interfaz, y llamar al método de la clase correspondiente cuando se clickee este nuevo botón. Para arribar a esta solución, también se tomó en cuenta el contexto de una clínica común, donde los diversos roles creados, no estarían cambiando sus funcionalidades diariamente, ya que previo a ello, habría consentimiento por parte del cliente sobre qué funcionalidades asignar a cada rol. En otras palabras, el cambio de funcionalidades de cada rol, comprendemos que sería muy poco frecuente, y fácil de adaptar a nuestro código e interfaces.
- Se creó una clase llamada “BDComun” encargada de realizar todas las consultas SQL necesarias para la correcta ejecución de ciertos métodos y también, para poder establecer el string de conexión con el que nos comunicamos con la base de datos SQL Server.



Interfaz capaz de administrar agregación o eliminación de funcionalidades diariamente a un rol. En la misma se enseña un rol que puede acceder a las funcionalidades de: ABM de Rol – ABM de Afiliado – Pedir Turno – Cambio de Rol. Pese a no poder seleccionar funcionalidades que no le corresponden, el usuario puede verlas.



Interfaz utilizada para el desarrollo de la aplicación desktop. La misma muestra un ejemplo de cómo un usuario, seleccionando un rol, no puede visualizar las funcionalidades de los demás roles. En caso de una agregación de funcionalidades al rol (muy poco frecuente, tal y como lo mencionamos), solo se agregaría el botón pertinente a esta interfaz, junto con el llamado a la clase y método correspondiente en el evento "click".

Decisiones tomadas para el desarrollo de la migración de datos: