Imprimir

BDI-Proyecto 2021

PRESENTACIÓN

PROYECTO DE ESTUDIO BASES DE DATOS I

Licenciatura en Sistemas de Información
Facultad de Ciencias Exactas y Naturales y Agrimensura
Universidad Nacional del Nordeste

Año 2021

DACAPS (Digitalización y Automatización de Centros de Atención Primaria de Salud).

Integrantes Grupo 03

- Alfonzo, Micaela
- Abrahan Ramirez, Ulises Alejandro
- Benites, Matias Maximiliano
- Cabrera, Andrea Florencia
- Cáceres Braun, Juan Gabriel

CAPITULO I

Descripción del Proyecto

Pretendemos con este proyecto, poder digitalizar la documentación de una red de CAPS en Corrientes, mediante el uso de una base de datos.
Con esto esperamos lograr que la información no sea redundante sino que, esté dispuesta de forma simplificada y ordenada.

Alcance del Proyecto

Este proyecto permitirá digitilizar y automatizar los datos recopilados de cada unos de los Centros de Atención Primaria de la Salud de la provincia.

CAPITULO II

El desarrollo de este modelo permitirá almacenar imágenes de las documentaciones ya existentes en la entidad, que serán previamente escaneadas mediante dispositivos a convenir, obteniendo así, la digitalización de los archivos y/o legajos físicos a tratar.

Se innovará la manera de registrar a los pacientes, consultar su historia clínica y solicitar informes, ya que se podrá realizar a través del sistema mismo. De esta manera, no será necesario archivar físicamente la documentación, ya que estaremos agilizando las tareas involucradas, y la misma se lleva a cabo de manualmente sin el apoyo ni uso de recursos informáticos, lo cual podría implicar demoras en su realización.

Además, permitirá dar de alta, modificar y consultar a los empleados de la institución. En efecto, es primordial contar con los servicios del modelo de datos para respaldar las actividades a desarrollarse en la gestión de automatización del centro de salud en cuestión.

Temas técnicos

Procedimientos almacenados

Un procedimiento almacenado es un conjunto de instrucciones a las que se les da un nombre, que se almacena en el servidor. Permiten encapsular tareas repetitivas.Al crear un procedimiento almacenado, las instrucciones que contiene se analizan para verificar si son correctas sintácticamente. Si no se detectan errores, SQL Server guarda el nombre del procedimiento almacenado en la tabla del sistema "*sysobjects*" y su contenido en la tabla del sistema "*syscomments*" en la base de datos activa. Si se encuentra algún error, no se crea.Un procedimiento almacenado puede hacer referencia a objetos que no existen al momento de crearlo. Los objetos deben existir cuando se ejecute el procedimiento almacenado.

Ventajas:

- comparten la lógica de la aplicación con las otras aplicaciones, con lo cual el acceso y las modificaciones de los datos se hacen en un solo sitio.-permiten realizar todas las operaciones que los usuarios necesitan evitando que tengan acceso directo a las tablas.
- reducen el tráfico de red; en vez de enviar muchas instrucciones, los usuarios realizan operaciones enviando una única instrucción, lo cual disminuye el número de solicitudes entre el cliente y el servidor.

Funciones

Una función es un conjunto de sentencias que operan como una unidad lógica, una rutina que retorna un valor. Dicha función tiene un nombre, acepta parámetros de entrada y retorna un valor escalar o una tabla.

Los parámetros de entrada pueden ser de cualquier tipo, excepto timestamp, cursor y table.

Las funciones definidas por el usuario no permiten parámetros de salida.

No todas las sentencias **SQL** son válidas dentro de una función. NO es posible emplear en ellas funciones no determinísticas (como **getdate()**) ni sentencias de modificación o actualización de tablas o vistas.Si podemos emplear sentencias de asignación, de control de flujo (**if**), de modificación y eliminación de variables locales.

SQL Server *admite 3 tipos de funciones definidas por el usuario,*clasificadas según el valor retornado:

1. escalares: **retornan un valor escalar**;
2. de tabla de varias instrucciones (**retornan una tabla**) y
3. de tabla en línea (**retornan una tabla**).

*Las funciones definidas por el usuario se crean con la instrucción "**create function**" y se eliminan con "**drop function**".*

Una función escalar **retorna un único valor**. Como todas las funciones, se crean con la instrucción "**create function**".

Las funciones que retornan una tabla pueden emplearse en lugar de un "**from**" de una consulta.

Vistas

Una vista es una alternativa para mostrar datos de varias tablas. Una vista es como una tabla virtual que almacena una consulta. Los datos accesibles a través de la vista no están almacenados en la base de datos como un objeto.

Entonces, una vista almacena una consulta como un objeto para utilizarse posteriormente. Las tablas consultadas en una vista se llaman tablas base. En general, se puede dar un nombre a cualquier consulta y almacenarla como una vista.

Las vistas permiten:

- ocultar información
- simplificar la administración de los permisos de usuario.
- mejorar el rendimiento

Triggers

Un "trigger" (**disparador o desencadenador**) es un tipo de procedimiento almacenado que se ejecuta cuando se intenta modificar los datos de una tabla (o vista). Se definen para una tabla (o vista) específica.

Se crean para conservar la integridad referencial y la coherencia entre los datos entre distintas tablas.

Si se intenta modificar (agregar, actualizar o eliminar) datos de una tabla en la que se definió un disparador para alguna de estas acciones (inserción, actualización y eliminación), el disparador se ejecuta (se dispara) en forma automática.

Un trigger se asocia a un evento (inserción, actualización o borrado) sobre una tabla.

La diferencia con los procedimientos almacenados del sistema es que los triggers:

- no pueden ser invocados directamente; al intentar modificar los datos de una tabla para la que se ha definido un disparador, el disparador se ejecuta automáticamente.
- no reciben y retornan parámetros.
- son apropiados para mantener la integridad de los datos, no para obtener resultados de consultas.

Los disparadores, a diferencia de las restricciones "**check**", pueden hacer referencia a campos de otras tablas.

Permisos

Los permisos de Motor de base de datos se administran en el nivel de servidor mediante inicios de sesión y roles de servidor, y en el nivel de base de datos mediante usuarios de base de datos y roles base de datos.

Entidades de seguridad

"Entidad de seguridad" es el nombre oficial de las identidades que utilizan **SQL Server** y a las que se pueden conceder permisos para realizar acciones. Suelen ser personas o grupos de personas, pero pueden ser otras entidades que finjan ser personas.

Las entidades de seguridad se pueden crear y administrar mediante el **Transact-SQL** indicado o mediante **SQL Server Management Studio**.

Roles de nivel de base de datos

Para administrar con facilidad los permisos en las bases de datos, SQL Server proporciona varios roles , que son las entidades de seguridad que agrupan a otras entidades de seguridad. Son como los grupos del sistema operativo Microsoft Windows. Los roles de nivel de base de datos se aplican a toda la base de datos en lo que respecta a su ámbito de permisos.

Para agregar y quitar usuarios en un rol de base de datos, use las opciones **ADD MEMBER** y **DROP MEMBER** de la instrucción ***ALTER ROLE*** . Existen dos tipos de roles en el nivel de base de datos:

- los roles fijos de base de datos que están predefinidos en la base de datos y,
- los roles de base de datos definidos por el usuario que el usuario puede crear.

Los permisos de los roles de base de datos definidos por el usuario se pueden personalizar con las instrucciones **GRANT, DENY y REVOKE**.

Transacciones

Una transacción es un conjunto de operaciones **Transact SQL** que se ejecutan como un único bloque, es decir, si falla una operación Transact SQL fallan todas. Si una transacción tiene éxito, todas las modificaciones de los datos realizadas durante la transacción se confirman y se convierten en una parte permanente de la base de datos. Si una transacción encuentra errores y debe cancelarse o revertirse, se borran todas las modificaciones de los datos.

La sentencia que se utiliza para indicar el comienzo de una transacción es '**BEGIN TRAN**'. Si alguna de las operaciones de una transacción falla hay que deshacer la transacción en su totalidad para volver al estado inicial en el que estaba la base de datos antes de comenzar. Esto se consigue con la sentencia '**ROLLBACK TRAN**'.Si todas las operaciones de una transacción se completan con éxito hay que marcar el fin de una transacción para que la base de datos vuelva a estar en un estado consistente con la sentencia 'COMMIT TRAN'.Los puntos de recuperación (*SavePoints*) permiten manejar las transacciones por pasos, pudiendo hacer *rollbacks* hasta un punto marcado por el savepoint y no por toda la transacción.

También se pueden manejar tomadas como excepciones mediante un try o un catch, como también con un operador condicional if.

La variable **@ERROR** en sql server devuelve 0 si la operación fue correcta o en caso contrario devolverá un valor distinto de 0 en el caso de que haya aparecido algún error.

Índice

Un índice de SQL Server es una estructura en disco o en memoria asociada con una tabla o vista que acelera la recuperación de filas de la tabla o vista. Un índice contiene claves generadas a partir de una o varias columnas de la tabla o la vista. En el caso de los índices en disco, dichas claves están almacenadas en una estructura de árbol (árbol B) que permite que SQL Server busque de forma rápida y eficiente la fila o las filas asociadas a los valores de cada clave.

Los índices almacenan los datos organizados de forma lógica como una tabla con filas y columnas, y físicamente almacenados en un formato de datos por fila llamado almacén de filas 1, o bien en un formato de datos por columna llamado almacén de columnas.

El diseño eficaz de los índices tiene gran importancia para conseguir un buen rendimiento de una base de datos y una aplicación.

Existen diversos tipos de índices disponibles:

- Hash
- Índice no agrupado optimizado para memoria
- Clúster
- No agrupado
- Único
- Columnstore
- Índice con columnas incluidas
- Índice con columnas calculadas
- Filtered
- Espacial
- XML
- Texto completo

Backup

Las copias de seguridad de SQL Server proveen una importante solución para proteger datos críticos que están almacenados en bases de datos SQL. Y para minimizar el riego de pérdida de datos, se necesita asegurarse de que se respalda las bases de datos regularmente tomando en consideración los cambios aplicados a los datos.

SQL Server ofrece muchos tipos de copias de seguridad, lo cual depende del modelo de recuperación de la base de datos, que controla cómo el registro de transacciones es manejado en la base de datos: Copias de seguridad completas, diferenciales, de archivos, de grupos de archivos y copias de seguridad transaccionales del registro.

La toma de copias de seguridad en SQL Server es simple de hacer vía SQL Server Management Studio, usando el comando T-SQL BACKUP DATABASE o el comando de PowerShell Backup-SqlDatabase.

CAPITULO III

Metodología

Ante la visualización del problema en cuestión, en principio se llevó a cabo una puesta en común de manera grupal sobre cuáles son los requisitos a implementar en esta versión, mediante tormenta de ideas logrando tener una base para poder iniciar el trabajo.

Luego decidimos investigar aplicando entrevistas, buscando datos e información por diferentes medios para profundizar el tema.

Una vez obtenida esta, planteamos la forma en la cual abordamos la realización del trabajo estableciendo objetivos, utilizando diversas herramientas volcando toda la información obtenida.

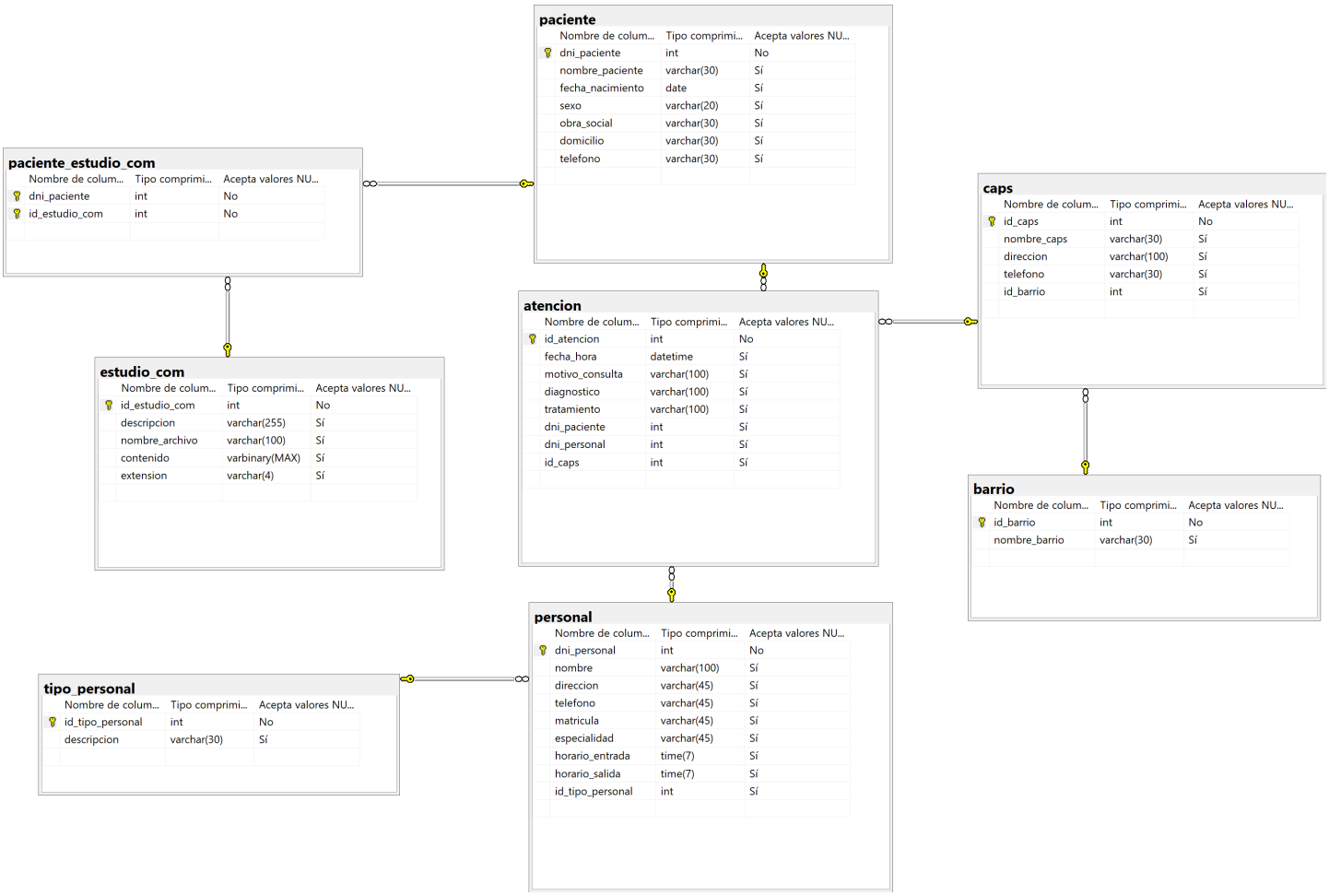
Herramientas

- SQL Server 2019
- Google Docs
- MySQL Workbench
- Microsoft SQL Server Management Studio

CAPITULO IV

Con este propósito se tiene como objetivo, lograr registrar los datos de los pacientes, los profesionales a cargo, información de las atenciones médicas, como ser las fichas médicas personales, los horarios de ingreso y salida del personal, así como también, de los pacientes mismos.

Diagrama del Modelo de datos



Entidades

- PACIENTE
- ESTUDIO_COM
- PACIENTE_ESTUDIO_COM
- CAPS
- BARRIO
- ATENCIÓN
- PERSONAL
- TIPO PERSONAL

Diccionario de datos

BARRIO

Nombre de Columna	Tipo	Longitud	NOT NULL	Descripción
id_barrio (PK)	Int	4	X	Valor autoincremental de barrio.
nombre	Varchar	45		Nombre del barrio.

- ID Barrio
- Nombre del Barrio

CAPS

Nombre de Columna	Tipo	Longitud	NOT NULL	Descripción
id_caps (PK)	Int	4	X	Valor autoincremental de CAPS.
nombre	Varchar	45		Nombre o razón social del CAPS.
direccion	Varchar	45		Dirección del CAPS.
telefono	Varchar	45		Teléfono del CAPS.

id_barrio(FK)	int	4		Numero de id del Barrio.

- ID CAPS.
- Nombre o razón social.
- Dirección.
- Teléfono.
- Id_Barrio.

PACIENTE

Nombre de Columna	Tipo	Longitud	NOT NULL	Descripción
dni_paciente (PK)	int	10	X	Valor autoincremental de paciente.
nombre	Varchar	100		Nombre y apellido del paciente.
fecha_nacimiento	Varchar	30		Fecha de nacimiento del paciente
sexo	Varchar	30		Sexo del paciente
obra_social	Varchar	30		Obra social del paciente.
domicilio	Varchar	30		Domicilio del paciente.
telefono	Varchar	30		Teléfono del paciente.

- DNI Paciente
- Nombre Paciente
- Fecha de Nacimiento
- Sexo
- Obra social
- Domicilio
- Teléfono

ATENCIÓN (Principal)

Nombre de Columna	Tipo	Longitud	NOT NULL	Descripción
id_atencion (PK)	Int	10	X	Valor autoincremental de Atención.
dni_paciente (FK)	Int	10		DNI del paciente.
fecha_hora	Datetime	20		Fecha y hora de atención
motivo_consulta	Varchar	100		Motivo de la consulta.
diagnostico	Varchar	100		Diagnostico del paciente
tratamiento	Varchar	100		Tratamiento del paciente.
dni_personal(FK)	int	10		DNI del personal
id_caps (FK)	int	10		ID del CAPS.

- ID atención
- DNI Paciente
- Fecha y hora
- Motivo de consulta
- Diagnóstico
- Tratamiento
- DNI personal
- ID CAPS

Tipo Personal

Nombre de Columna	Tipo	Longitud	NOT NULL	Descripción
id_tipo_personal (PK)	int	10	X	Valor autoincremental de Especialidad

descripcion	Varchar	30	X	Nombre de la especialidad

- ID tipo personal
- Nombre de la especialidad

Personal

Nombre de Columna	Tipo	Longitud	NOT NULL	Descripción
dni_personal (PK)	int	10	X	Valor autoincremental del Personal .
nombre	Varchar	100	X	Nombre y apellido del personal .
direccion	Varchar	45	X	Dirección del personal .
telefono	Varchar	45	X	Teléfono del personal .
matricula	Varchar	45		Matrícula del Profesional
especialidad	Varchar	45		Especialidad del Profesional
horario_entrada	Time			Hora laboral de ingreso
hora de salida	Time			Hora laboral de salida
id_tipo_personal(FK)	int	10	X	ID de tipo de personal

- DNI personal
- Nombre Personal
- Dirección
- Teléfono
- Matrícula
- Especialidad
- Horario de entrada
- Horario de salida
- ID Tipo Personal

Estudio_com

Nombre de Columna	Tipo	Longitud	NOT NULL	Descripción
id_estudio_com (PK)	int	10	X	Valor autoincremental del estudio .
descripcion	Varchar	100		Descripción
nombre_archivo	Varchar	45		Nombre del archivo .
contenido	Varbinary	MAX		Imágen .
extensión	Varchar	45		tipo de Extensión.

- ID_estudio_com.
- Descripción.
- Nombre_archivo.
- Contenido.
- Extension.

Paciente_Estudio_com

Nombre de Columna	Tipo	Longitud	NOT NULL	Descripción
dni_paciente(PK)	Int	10	X	Dni del paciente
id_estudio_com(PK)	Int	100	X	Id autoincremental. .

- Dni_paciente
- Id_estudio_com

CAPITULO V

Durante el desarrollo del proyecto nos encontramos con diversas dificultades a la hora de implementar los scripts de la base de datos, así como también, generar una estructura adecuada al sistema en cuestión. El principal problema se presentó al momento de investigar, por distintos medios, sobre el procedimiento adecuado para cargar datos de tipo multimedia, lo cuál estaba por fuera de nuestros conocimientos hasta el momento.

En relación al prototipo final, creemos que es un trabajo para destacar en lo que al grupo respecta, ya que nos enfocamos en solucionar una necesidad de la realidad que nos rodea y logramos llevarlo a cabo en forma conjunta y ordenada mediante reuniones sincrónicamente programadas.

BIBLIOGRAFÍA

- https://docs.microsoft.com/en-us/documentation/
- Investigación mediate herramientas de busqueda en Internet.-
- https://www.w3schools.com/sql/
- Material de la Catedra
- https://www.netveloper.com/listado-de-indices-en-sql-server
- https://www.programandoamedianoche.com/2009/09/indices-filtrados-en-sql-server-2008/

ANEXO

Videos

Videos y capturas

- **https://drive.google.com/drive/folders/1rcX5a-ifLLwmimSh38vabvi6FRUkwUsx**

Script SQL

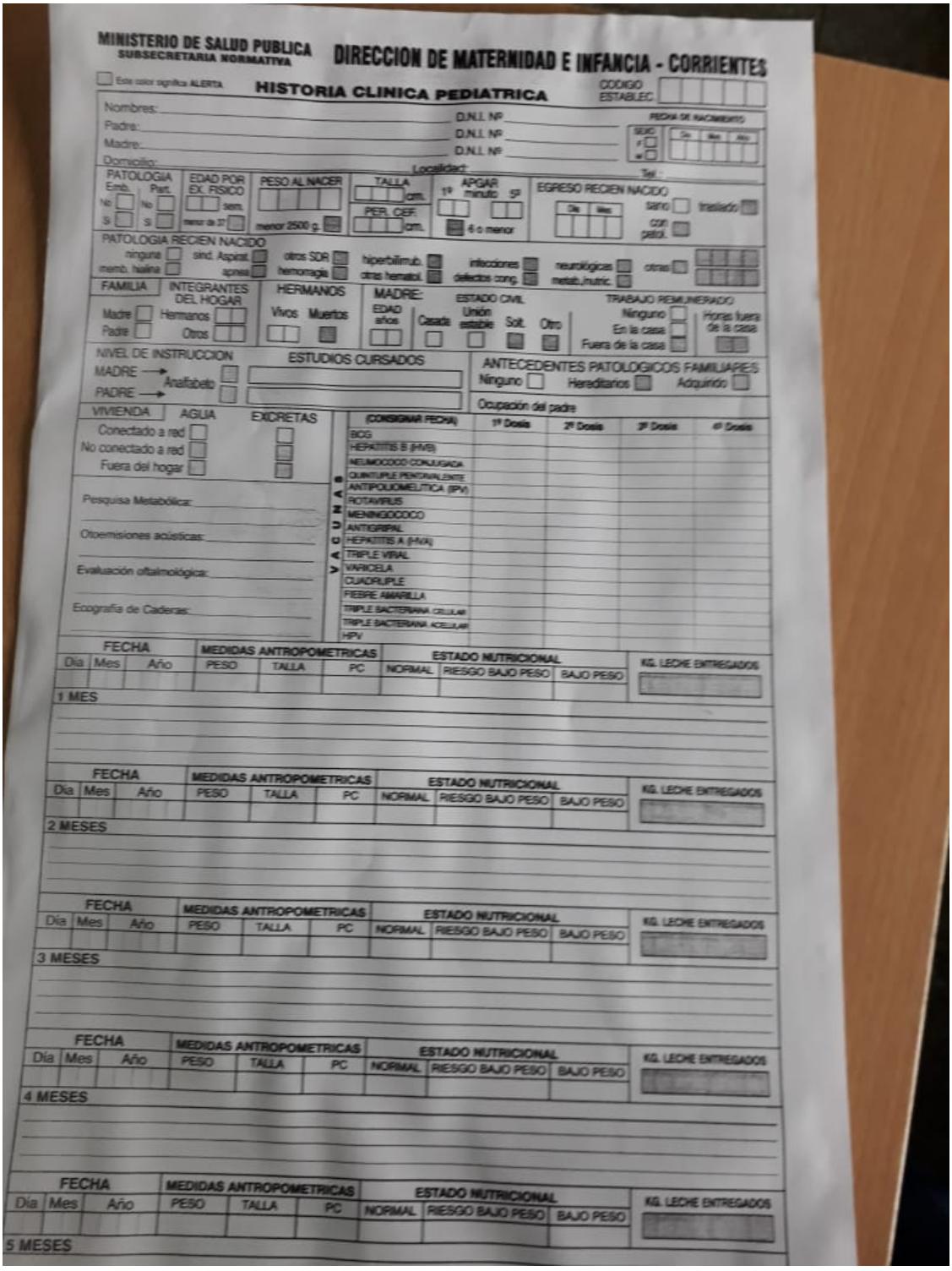
- Archivo Script de creación de la Base de datos CAPS y sus respectivas tablas en base al Modelo de datos.
- Archivo Script de lote de Datos para la carga inicial de la Base CAPS.
- Archivo Script de la creación de los índices no agrupados y filtrados en la Base de Datos CAPS.
- Archivo Script de todos los índices existentes en la Base de Datos CAPS.
- Archivo Script de Permisos de usuarios de la Base CAPS.
- Archivo Script de Triggers de la Base CAPS.
- Archivo Script de Procedimientos almacenados de la Base CAPS.
- Archivo Script de Vistas.
- Archivo Script de Transacciones

Documentación complementaria

Carga de archivo: es preciso ubicar una imagen en la ruta "C:/IMG.jpg"

Se debe respetar el nombre y el formato o puede cambiarse debajo si así se desea.

Imagen de ejemplo



MINISTERIO DE SALUD PUBLICA
SUBSECRETARIA NORMATIVA

DIRECCION DE MATERNIDAD E INFANCIA - CORRIENTES

HISTORIA CLINICA PEDIATRICA

Este color significa ALERTA

CODIGO ESTABLEC.

Nombres:
Padre:
Madre:
Domicilio:
Localidad:
D.N.I. Nº
Tel.:
FECHA DE NACIMIENTO

PATOLOGIA Embl. Pert. EDAD POR EX. FISICO PESO AL NACER TALLA PER. CEF. APGAR 1º minuto 5º EGRESO RECIEN NACIDO

PATOLOGIA RECIEN NACIDO

FAMILIA INTEGRANTES DEL HOGAR HERMANOS MADRE: ESTADO CIVIL TRABAJO REMUNERADO

NIVEL DE INSTRUCCION ESTUDIOS CURSADOS ANTECEDENTES PATOLOGICOS FAMILIARES

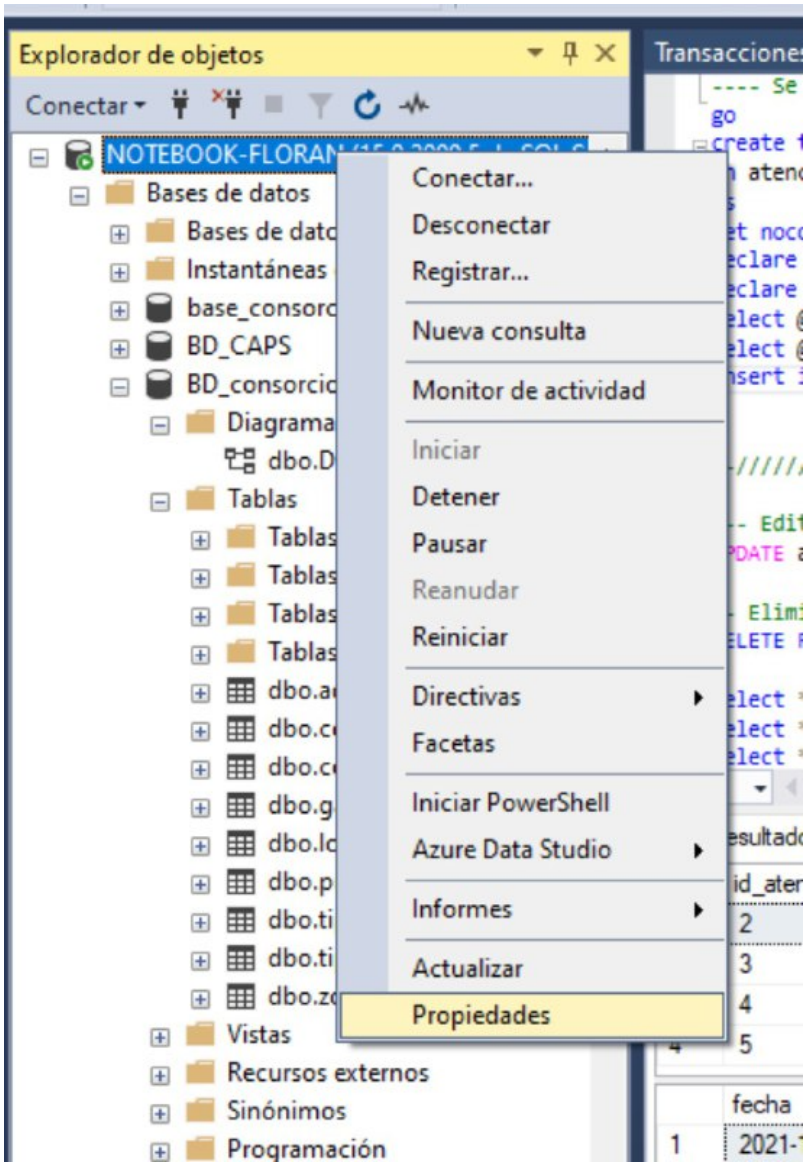
VIVIENDA AGUA EXCRETAS

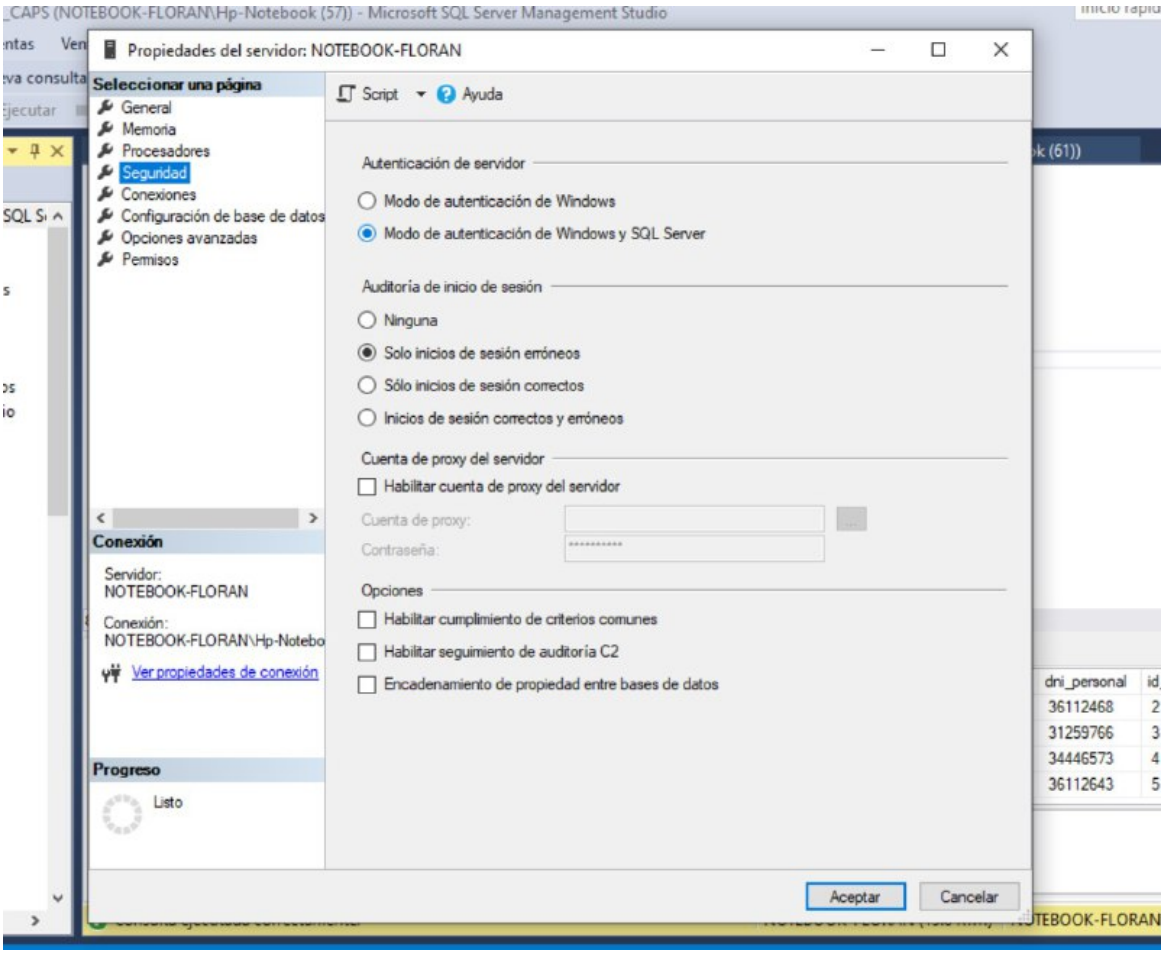
Pesquisa Metabólica:
Otoemisiones acústicas:
Evaluación oftalmológica:
Ecografía de Caderas:

(CONSIGNAR FECHA)	1ª Dosis	2ª Dosis	3ª Dosis	4ª Dosis
BCG				
HEPATITIS B (HVB)				
NEUMOCOCO CONJUGADA				
QUINTUPLE PENTAVALENTE				
ANTIPOLIOMELITICA (IPV)				
ROTAVIRUS				
MENINGOCOCO				
ANTIGRIPAL				
HEPATITIS A (HVA)				
TRIPLE VIRAL				
VARICELA				
CUADRUPLE				
FIEBRE AMARILLA				
TRIPLE BACTERIANA CELULAR				
TRIPLE BACTERIANA ACELULAR				
HPV				

FECHA Día	Mes	Año	MEDIDAS ANTROPOMETRICAS PESO	TALLA	PC	ESTADO NUTRICIONAL NORMAL	RIESGO BAJO PESO	BAJO PESO	KG. LECHE ENTREGADOS
1 MES									
2 MESES									
3 MESES									
4 MESES									
5 MESES									

Para los permisos es indispensable configurar para que el motor de base de datos pueda loguear metiante Autentificación de Window y SQL Server


Explorador de objetos
Conectar
NOTEBOOK-FLORAN
Bases de datos
Bases de datc
Instantáneas
base_consorc
BD_CAPS
BD_consorcic
Diagrama
dbo.D
Tablas
Tablas
Tablas
Tablas
Tablas
dbo.a
dbo.c
dbo.c
dbo.g
dbo.lc
dbo.p
dbo.ti
dbo.ti
dbo.zc
Vistas
Recursos externos
Sinónimos
Programación
Conectar...
Desconectar
Registrar...
Nueva consulta
Monitor de actividad
Iniciar
Detener
Pausar
Reanudar
Reiniciar
Directivas
Facetas
Iniciar PowerShell
Azure Data Studio
Informes
Actualizar
Propiedades
Transacciones
go
id_ater
2
3
4
5
fecha
1
2021-


Propiedades del servidor: NOTEBOOK-FLORAN
Seleccionar una página
General
Memoria
Procesadores
Seguridad
Conexiones
Configuración de base de datos
Opciones avanzadas
Permisos
Script
Ayuda
Autenticación de servidor
Modo de autenticación de Windows
Modo de autenticación de Windows y SQL Server
Auditoría de inicio de sesión
Ninguna
Solo inicios de sesión erróneos
Sólo inicios de sesión correctos
Inicios de sesión correctos y erróneos
Cuenta de proxy del servidor
Habilitar cuenta de proxy del servidor
Cuenta de proxy:
Contraseña:
Opciones
Habilitar cumplimiento de criterios comunes
Habilitar seguimiento de auditoría C2
Encadenamiento de propiedad entre bases de datos
Conexión
Servidor:
NOTEBOOK-FLORAN
Conexión:
NOTEBOOK-FLORAN\Hp-Notebo
Ver propiedades de conexión
Progreso
Listo
Aceptar
Cancelar
dni_personal
36112468
31259766
34446573
36112643