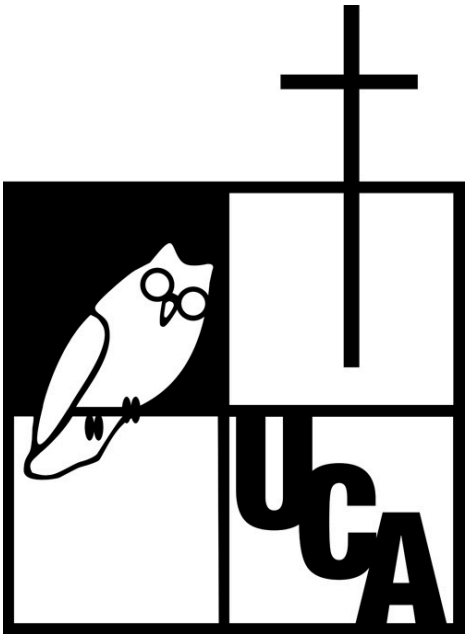


Universidad Centroamericana José Simeón Cañas
Proyecto de Cátedra Administración de Base de Datos

Proyecto: Gestión Hospitalaria

Integrantes:
Daniel Alexander Ayala Escobar 00045824
Gisela Elizabeth Rivas Rivera 00016124
Emily Eunice Orellana Mendez 00147124
Daniel Emilio Lopez Quintanilla 00140323
Edgardo Baltazar Pérez Molina 00400525

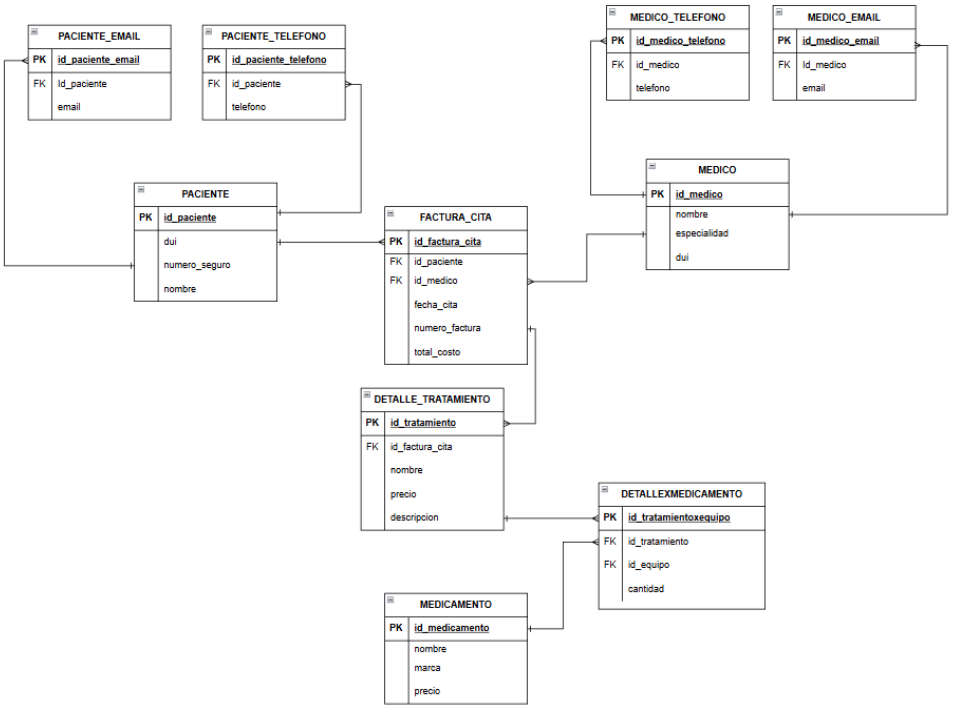
Catedrático: James Edward Humberstone Morales

Fecha de entrega: 26/11/2025

❖ Descripción del proyecto

El sistema de Gestión Hospitalaria permite administrar de forma completa y organizada toda la información relacionada con la atención médica en un hospital. El proyecto abarca el registro de pacientes y médicos, la gestión de sus correos y teléfonos, el control de medicamentos disponibles y la creación de citas médicas que generan facturas. Cada factura integra automáticamente el tratamiento realizado, los medicamentos utilizados y el cálculo total del costo mediante funciones especializadas, además, el sistema incorpora mecanismos para importar grandes volúmenes de datos, validar precios y asegurar la integridad mediante triggers, así como aplicar auditoría para registrar operaciones importantes realizadas por los usuarios. También incluye vistas analíticas que permiten conocer el total gastado del paciente, los médicos con mayor productividad y los medicamentos más recetados. Finalmente, se definieron roles y permisos que garantizan que cada usuario tenga acceso únicamente a las funciones que necesita. En conjunto, el sistema automatiza, controla y asegura toda la operación hospitalaria desde el registro de personas hasta la facturación y análisis de información clínica.

❖ Diagrama relacional



❖ Diccionario de datos

Tabla	Campo	Tipo de dato	Descripción
PACIENTE	id_paciente	INT IDENTITY PK	Llave primaria e identificador único del PACIENTE auto incrementable
PACIENTE	dui	VARCHAR(10) NOT NULL UNIQUE FK	Identificador único personal del paciente
PACIENTE	numero_seguro	VARCHAR(20)	Código identificador del paciente
PACIENTE	nombre	NVARCHAR(100) NOT NULL	Nombre de paciente

Tabla	Campo	Tipo de dato	Descripción
PACIENTE_EMAIL	id_paciente_email	INT IDENTITY PK	Llave primaria e identificador único de PACIENTE_EMAIL auto incrementable
PACIENTE_EMAIL	Id_paciente	INT FK	Llave foránea hacia la tabla de PACIENTE (id_paciente)
PACIENTE_EMAIL	email	NVARCHAR(100)	Correo electrónico del paciente

Tabla	Campo	Tipo de dato	Descripción
MEDICO_EMAL	id_medico_email	INT IDENTITY PK	Llave primaria e identificador único de MEDICO_EMAIL auto incrementable
MEDICO_EMAIL	id_medico	INT NOT NULL FK	Llave foránea hacia la tabla de MEDICO (id_medico)
MEDICO_EMAIL	email	NVARCHAR (100) NOT NULL	Especialidad del medico

Tabla	Campo	Tipo de dato	Descripción
PACIENTE_TELEFONO	id_paciente_telefono	INT IDENTITY PK	Llave primaria e identificador único de PACIENTE_TELEFONO auto incrementable
PACIENTE_TELEFONO	Id_paciente	INT FK	Llave foránea hacia la tabla de PACIENTE (id_paciente)
PACIENTE_TELEFONO	telefono	NVARCHAR (100)	Número de teléfono del paciente

Tabla	Campo	Tipo de dato	Descripción
MEDICO	id_medico	INT IDENTITY PK	Llave primaria e identificador único de MEDICO auto incrementable
MEDICO	nombre	NVARCHAT (100) NOT NULL	Nombre del medico
MEDICO	especialidad	NVARCHAR (100)	Especialidad del medico
MEDICO	dui	VARCHAT (10)	Identificador único personal del medico

Tabla	Campo	Tipo de dato	Descripción
MEDICO_TELEFONO	id_medico_telefono	INT IDENTITY PK	Llave primaria e identificador único de MEDICO_TELEFONO auto incrementable
MEDICO_TELEFONO	Id_medico	INT NOT NULL FK	Llave foránea hacia la tabla de MEDICO (id_medico)
MEDICO_TELEFONO	telefono	VARCHAR (15)	Número de teléfono del medico

Tabla	Campo	Tipo de dato	Descripción
FACTURA_CITA	id_factura_cita	INT IDENTITY PK	Llave primaria e identificador único del FACTURA_CITA auto incrementable
FACTURA_CITA	Id_paciente	INT NOT NULL FK	Referencia a PACIENTE
FACTURA_CITA	Id_medico	INT NOT NULL FK	Referencia a MEDICO
FACTURA_CITA	Fecha_cita	INT NOT NULL	Fecha de la cita atendida
FACTURA_CITA	numero_factura	INT NOT NULL	Numero único de la factura
FACTURA_CITA	total_costo	INT NOT NULL	Costo total de la factura

Tabla	Campo	Tipo de dato	Descripción
DETALLE_TRATAMIENTO	id_tratamiento	INT IDENTITY PK	Llave primaria e identificador único del DETALLE_TRATAMIENTO auto incrementable
DETALLE_TRATAMIENTO	Id_factura_cita	INT NOT NULL FK	Llave foránea hacia la tabla FACTURA_CITA (id_factura_cita)
DETALLE_TRATAMIENTO	nombre	NVARCHAR (100) NOT NULL	Nombre del tratamiento
DETALLE_TRATAMIENTO	precio	DECIMAL	Costo del tratamiento
DETALLE_TRATMIENTO	descripcion	NVARCHAR (255)	Descripción del tratamiento

Tabla	Campo	Tipo de dato	Descripción
MEDICAMENTO	id_medicamento	INT IDENTITY PK	Llave primaria e identificador único de MEDICAMENTO auto incrementable
MEDICAMENTO	nombre	NVARCHAR (100) NOT NULL	Nombre del medicamento
MEDICAMENTO	marca	NVARCHAR (100)	Marca comercial del medicamento
MEDICAMENTO	precio	DECIMAL	Precio unitario del medicamento

Tabla	Campo	Tipo de dato	Descripción
DETALLEXMEDICAMENTO	id_tratamientoxequipo	INT IDENTITY PK	Llave primaria e identificador único de DETALLEXMEDICAMENTO auto incrementable
DETALLEXMEDICAMENTO	id_tratamiento	INT NOT NULL FK	Relación con DETALLE_TRATAMIENTO
DETALLEXMEDICAMENTO	id_medicamento	INT NOT NULL FK	Relación con MEDICAMENTO
DETALLEXMEDICAMENTO	cantidad	DECIMAL	Cantidad de medicamento utilizada

❖ Creación de la base de datos

El presente informe describe el proceso de creación de la base de datos implementada para una adecuada administración de la base GestiónHospitalaria. En él se detallan la estructura de las tablas, las relaciones definidas entre ellas y el procedimiento de carga inicial de datos necesarios para su correcto funcionamiento.En primer lugar, se procedió a crear la base de datos, asignándole el nombre GestionHospitalaria, la cual servirá como repositorio central de la información. Posteriormente, se seleccionó esta base de datos para trabajar sobre ella.

1. **Tabla PACIENTE**: Se creó la tabla PACIENTE, destinada a almacenar la información principal de los pacientes. Contiene campos como el DUI, número de seguro, y nombre del paciente. El campo id_paciente se definió como clave primaria mediante un IDENTITY autoincremental.
2. **Tablas PACIENTE_EMAIL y PACIENTE_TELEFONO**: Para cumplir con buenas prácticas de normalización, se separaron los correos electrónicos y teléfonos en tablas independientes, ambas tablas incluyen un identificador propio y una clave foránea hacia la tabla PACIENTE, con reglas ON DELETE CASCADE y ON UPDATE CASCADE, permitiendo mantener la integridad referencial.
3. **Tabla MEDICO**: Se creó la tabla MEDICO, que almacena los datos del personal médico, incluyendo nombre, especialidad y DUI. Su clave primaria es id_medico, también autoincremental.

4. **Tablas MEDICO_EMAIL Y MEDICO_TELEFONO**: Al igual que con los pacientes, se dividieron los correos y teléfonos del médico en dos tablas separadas, las dos poseen claves foráneas hacia MEDICO y las mismas reglas de cascada para asegurar consistencia.
5. **Tabla FACTURA_CITA**: Esta tabla registra la información sobre cada cita facturada. Incluye referencias tanto al paciente como al médico mediante claves foráneas. También almacena la fecha de la cita, número de factura y el costo total.
6. **Tabla DETALLE_TRATAMIENTO**: En esta tabla se describen los tratamientos o servicios que formaron parte de una factura específica. Incluye nombre del tratamiento, precio y descripción. Se relaciona con FACTURA_CITA mediante una clave foránea.
7. **Tabla MEDICAMENTO**: Se creó la tabla MEDICAMENTO para registrar los distintos medicamentos disponibles, indicando su nombre, marca y precio.
8. **Tabla DETALLEXMEDICAMENTO**: Finalmente, se creó la tabla DETALLEXMEDICAMENTO, la cual permite relacionar los tratamientos con los medicamentos utilizados. Contiene la cantidad y claves foráneas hacia DETALLE_TRATAMIENTO y MEDICAMENTO.

❖ Población de datos

Después de crear la estructura completa de la base de datos GestionHospitalaria, se procedió a realizar la población inicial de datos. Este proceso tuvo como finalidad contar con registros base que permitieran probar el correcto funcionamiento de las tablas, validar las relaciones establecidas y garantizar que las reglas de integridad referencial se aplicaran adecuadamente.
La inserción de datos se realizó utilizando sentencias INSERT INTO, siguiendo un orden lógico basado en la dependencia de las claves foráneas.

1. **Inserción en tablas principales**: En primer lugar, se registraron datos en las tablas que no dependen de ninguna otra:
    a. PACIENTE
    b. MEDICO
    c. MEDICAMENTO

Estas tablas contienen la información principal del sistema, por lo que sus registros se utilizan como referencia en el resto del modelo. Al insertarse, los campos IDENTITY generaron automáticamente los identificadores necesarios para posteriores relaciones.

2. **Inserción en tablas relacionadas con pacientes y médicos**: Una vez disponibles los identificadores generados, se procedió a poblar las tablas asociadas
    a. PACIENTE_EMAIL
    b. PACIENTE_TELEFONO
    c. MEDICO_EMAIL
    d. MEDICO_TELEFONO

Para estas inserciones fue necesario utilizar los valores de id_paciente y id_medico obtenidos de las tablas principales Las restricciones ON DELETE CASCADE y ON UPDATE CASCADE se verificaron correctamente, asegurando la coherencia entre registros.

3. **Inserción en FACTURA_CITA**: Luego se registraron datos en la tabla FACTURA_CITA, la cual requiere referencias válidas tanto de pacientes como de médicos.En esta etapa se asignaron fechas de cita, números de factura y montos de costo total, permitiendo simular escenarios reales de atención hospitalaria.
4. **Inserción en DETALLE_TRATAMIENTO**: Posteriormente, se incorporaron los tratamientos asociados a cada factura en la tabla DETALLE_TRATAMIENTO, especificando el nombre del tratamiento, precio y descripción correspondiente. Esta tabla depende directamente de FACTURA_CITA, por lo que cada registro se vinculó mediante la clave foránea id_factura_cita.
5. **Inserción en DETALLEXMEDICAMENTO**: Finalmente, se registraron los medicamentos utilizados en cada tratamiento dentro de la tabla DETALLEXMEDICAMENTO. Para ello fue necesario disponer previamente de los identificadores generados las tablas:
    a. DETALLE_TRATAMIENTO
    b. MEDICAMENTO

Con esta última etapa, se completó la carga inicial necesaria para probar consultas, relaciones y operaciones en cascada dentro del modelo.

❖ Creación de Roles

1. Roles comunes

Rol	Select	Execute
r_recepcionista	VW_Paciente	USP_RegistrarPaciente
	VW_EmailPaciente	USP_RegistrarEmailPaciente
	VW_TelefonoPaciente	USP_RegistrarTelefonoPaciente
r_admin_medico	VW_Medico	USP_RegistrarMedico
	VW_EmailMedicos	USP_RegistrarEmailMedico
	VW_TelefonoMedicos	USP_RegistrarTelefonoMedico
r_farmacia	VW_Medicamento	USP_RegistrarMedicamento
r_cajero	FN_FacturasPorFechas	
	VW_Paciente	USP_RegistrarFactura
	VW_Medico	
	VW_Medicamento	
r_analista	VW_Paciente	N/A
	VW_Medico	
	VW_Medicamento	
	VW_EmailPaciente	
	VW_TelefonoPaciente	
	VW_EmailMedicos	
	VW_TelefonoMedicos	
	VW_Indices	
	FN_FacturasPorFechas	

2 . Roles especiales

Rol	Backup	Create	Otros Permisos
r_admin	N/A	TABLE	SELECT
		VIEW	ALTER
		PROCEDURE	ALTER ANY ROLE
		FUNCTION	ALTER ANY USER
		SEQUENCE	VIEW DEFINITION
r_backup	BACKUP DATABASE	N/A	N/A
	BACKUP LOG		

Descripción de los roles:

- **Administrador de la Base de Datos (r_admin):** tiene permisos explícitos para manejar usuarios, roles, esquemas y crear o modificar objetos dentro de la base.
- **Backup (r_backup):** realiza copias de seguridad de la base sin poder ver ni manipular información.
- **Recepcionista (r_recepcionista):** tiene permisos para ejecutar los procedimientos de registro de pacientes y ver su información básica. No puede modificar datos médicos ni facturas.

- **Administrador de Médicos (r_admin_medico):** tiene permisos para registrar nuevos médicos y actualizar su información de contacto. También puede consultar vistas relacionadas con los médicos del hospital.
- **Farmacia (r_farmacia)**: tiene permisos para registrar medicamentos y consultar el catálogo de medicinas disponibles, pero no puede ver datos de pacientes o facturas.
- **Cajero (r_cajero):** puede registrar facturas, calcular costos y consultar datos necesarios para la atención del paciente, pero no puede modificar datos clínicos.
- **Analista (r_analista):** rol pensado para personal que genera reportes. Tiene acceso de solo lectura a las vistas que contienen información general del hospital.

❖ Organización en Esquemas

Un esquema en SQL Server es un contenedor lógico que agrupa y organiza objetos de la base de datos, tales como tablas, vistas, procedimientos almacenados, funciones, sinónimos, triggers, etc.

En la base de datos de Gestión Hospitalaria se optado por la creación de esquemas como una forma de mejorar la organización de la base de datos y brindar seguridad al proveer a los permisos a los roles por medio de esquemas. Para esto se crearon esquemas para cada una de las tablas creadas dentro de la base de datos, con nombres descriptivos que ayudan a la ubicación de las tablas.

Se crearon los siguientes esquemas:

- **Paciente:** Contiene la tabla de los datos del paciente, además de las tablas de datos multivariados de email y teléfono.
- **Médico**: Contiene la tabla de los datos del médico , además de las tablas de datos multivariados de email y teléfono.
- **Cita**: Contiene la tabla FACTURA_CITA, esto para facilitar su ubicación dentro de la base de datos.
- **Tratamiento**: Contiene las tablas de DETALLE_TRATAMIENTO y la tabla cruzada con medicamentos y detalle_tratamiento, este esquema es útil para agrupar las tablas relacionadas con tratamientos.
- **Catalogo**: Contiene la tabla de medicamentos, útil para dejar apartado el tema de logística de los medicamentos.

❖ Especificaciones de Auditoría

La auditoría es el proceso mediante el cual un sistema de base de datos registra y supervisa las acciones realizadas por los usuarios su propósito es controlar, rastrear y evidenciar lo que sucede dentro del sistema, permitiendo detectar actividades sospechosas, cumplir normas de seguridad y reconstruir eventos ante incidentes.

En la base de datos de Gestión Hospitalaria se ha optado por la creación de auditorías, esto para poder llevar un control de qué acciones se realizan dentro de la base de datos y que usuario fue el que realizó la acción, la auditoría registra las acciones que se han considerado vitales o relevantes el saber que se está realizando pues estas tablas y procesos conllevan a modificar datos importantes para lo que es el tema financiero y de logística. Por lo tanto se han agregado a la auditoría los siguientes objetos:

- USP_RegistrarFactura
- USP_RegistrarMedicamento
- USP_RegistrarPaciente
- USP_RegistrarMedico
- Cita.FACTURA_CITA
- Catalogo.MEDICAMENTO
- USP_ImportarPacientes
- USP_ImportarMedicos
- USP_ImportarMedicamentos

Además se ha creado una vista para poder apreciar a manera de tabla la auditoría de las diferentes acciones realizadas, mostrando la fecha de realización del evento, la acción realizada, la secuencia que se realizó, la base de datos, el esquema, el usuario que realizó la acción, el id de sesión y el ip cliente, todos estos datos se tomaron en cuenta para poder llevar un buen registro de las acciones realizadas y quien la realizó.

```
EXEC USP_RegistrarPaciente 'Pepito Juancito Perez Lopez','7781220','123456','pjuancito@gmail.com','77788811'
```

Results | Messages

FechaEvento	TipoAccion	SentenciaSQL	BaseDatos	Esquema	Objeto	UsuarioServidor	IdSesion	IPCliente
2025-11-21 03:50:58.6987515	AUSC					sa	60	local machine
2025-11-21 15:05:51.1095751	AUSC					sa	27	Unknown
2025-11-24 18:03:49.0183347	EX	EXEC USP_RegistrarPaciente 'Pepito Juancito Per...	GestionHospitalaria	dbo	USP_RegistrarPaciente	sa	54	local machine

❖ Dimensionamiento de la base

Para determinar el tamaño total en bytes de cada tabla, primero calculamos el tamaño en bytes de cada uno de sus campos según la tabla de equivalencias establecida. Posteriormente sumamos los tamaños individuales y multiplicamos el resultado por la cantidad de registros que contiene la tabla. De esta manera obtenemos el total de bytes por tabla.
Finalmente, para las tablas que incluyen índices, multiplicamos el resultado por un factor de 1.3, con el fin de considerar el espacio adicional requerido por dichos índices. A continuación se muestran las tablas con los cálculo

Tabla PACIENTE	Tipo de dato	Tamaño(bytes)
id_paciente	INT	4
dui	VARCHAR(10)	12
numero_seguro	VARCHAR(20)	22
nombre	NVARCHAR(100)	202
Registros en la tabla	10000	
TOTAL		2400000

Tabla MEDICO	Tipo de dato	Tamaño (bytes)
id_medico	INT	4
nombre	NVARCHAR(100)	202
especialidad	NVARCHAR(100)	202
dui	VARCHAR(10)	12
Registros en la tabla	100	
TOTAL		42000

Tabla PACIENTE_EMAIL	Tipo de dato	Tamaño(bytes)
id_paciente_email	INT	4
id_paciente	INT	4
email	VARCHAR(100)	102
Registros en la tabla	10000	
TOTAL		1100000

Tabla MEDICO_EMAIL	Tipo de dato	Tamaño(bytes)
id_medico_email	INT	4
id_medico	INT	4
email	VARCHAR(100)	102
Registros en la tabla	100	
TOTAL		11000

Tabla PACIENTE_TELEFONO	Tipo de dato	Tamaño(bytes)
id_paciente_telefono	INT	4
id_paciente	INT	4
telefono	VARCHAR(15)	17
Registros en la tabla	10000	
TOTAL		250000

Tabla MEDICO_TELEFONO	Tipo de dato	Tamaño(bytes)
id_medico_telefono	INT	4
id_medico	INT	4
telefono	VARCHAR(15)	17
Registros en la tabla	100	
TOTAL		2500

Tabla MEDICAMENTO	Tipo de dato	Tamaño (bytes)
id_medicamento	INT	4
nombre	NVARCHAR(100)	202
marca	NVARCHAR(100)	202
precio	DECIMAL	5
Registros en la tabla	50	
TOTAL		20650

TABLAS QUE CONTIENEN INDICES

Tabla FACTURA_CITA	Tipo de datos	Tamaño (bytes)
id_factura_cita	INT	4
id_paciente	INT	4
id_medico	INT	4
fecha_cita	DATE	3
numero_factura	VARCHAR(20)	22
total_costo	DECIMAL	5
Indice		1.3
Registros en la tabla	10000	
TOTAL		546000

Tabla DETALLE_TRATAMIENTO	Tipo de dato	Tamaño (bytes)
id_tratamiento	INT	4
id_factura_cita	INT	4
nombre	NVARCHAR(100)	202
precio	DECIMAL	5
descripcion	NVARCHAR(255)	512
Indice		1.3
Registro en la tabla	10000	
TOTAL		9451000

Tabla DETALLEXMEDICAMENTO	Tipo de dato	Tamaño(bytes)
id_tartamientoxequipo	INT	4
id_tratamiento	INT	4
id_medicamento	INT	4
cantidad	INT	4
indice		1.3
Registros en la tabla	10000	
TOTAL		208000

❖ Plan de backup y restauración (Estrategia + calendario + jobs en SQL Agent)

Lo más desastroso para un negocio que aplique una metodología digital, es que todo se venga abajo, todo se corrompa, o, en pocas palabras, que todo se pierda. Al igual que ser una buena práctica para casi cualquier ocasión, en este proyecto de implementación digital de facturación, se buscó que la topología del negocio fuera respaldada por completo por un calendario minuciosamente construido para proteger justamente el negocio de algún inconveniente que pudiera pasar. Definamos entonces el plan de backup para este proyecto.

Un plan de backup o respaldo, es un calendario de planificación, donde se detalla, e idealmente se justifica, la metodología que tendrá el plan de restauración, orientado a la lógica que pueda poseer el negocio. Para este negocio, se busco planificar este backup basándose en un horario donde el tráfico fuera menor, o no se estuviera ocupando idealmente, esto con el objetivo de no provocar alguna lentitud en el sistema en uso y esto provocar inconvenientes de cualquier índole producto a la lentitud.

Dicho lo anterior, en la siguiente tabla se detalla y justifica el plan de backup que el negocio tendrá, considerando todo lo antes dicho.

Plan:	Estrategia de backup para sistema de facturacion en linea hospitalaria
Negocio:	El sistema actual es una plataforma de facturación en línea diseñada para operar de manera digital. Sin embargo, ante escenarios de contingencia donde los datos se han corrompido, por alguna causa adyacente a la normal, se ha contemplado un procedimiento manual basado en papel para garantizar la continuidad operacional. Este respaldo en papel es temporal, pues luego de respaldar la base de datos, estos papeles tendrán que ser ingresados en línea con la ayuda de las secretarias que en papel escribieron y realizaron las respectivas transacciones del proceso de facturación

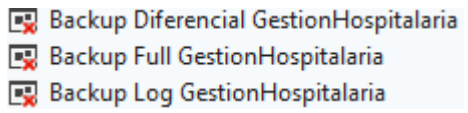
Objetivos:	RPO: 15 minutos.	RTO: 2 horas.	WRT: 1 hora.	MTD: 3 horas.	Backup: Completo, Diferencial, Transaccional
Motivo:	Con la utilización de “jobs” de sql server, se ha establecido y programado que las recuperaciones de datos solo pueden perder 15 minutos de información antes de que la base de datos realice otro backup del lote de información	Tiempo estimado en el que: se siente, se diagnostica e identifica el problema de la base de datos o la corrupción de datos de la base de datos y se recupera. Para nuestro negocio, y sus 100k registros, el tiempo de recuperación por el peso de los registros es despreciable, comparado con el diagnóstico y el modo de afrontar el problema y su respectiva restauración	Este tiempo se ha designado para que el negocio se sincronice con los valores perdidos del: RPO y RTO; dado que la base de datos se cayó y quedó inoperativa, y los últimos 15 minutos no se guardaron, es necesario sincronizar todo ese papeleo que se generó en ese transcurso de tiempo en 1 hora, la cual es la designada para este apartado	Este tiempo es el total que se ocupará para realizar toda la sincronización y levantamiento de base de datos. Cabe recalcar que mientras se levante la base de datos, queda prohibido el uso a nivel de negocio la base de datos, pues puede ocurrir una inconsistencia de datos si alguien llegara a ingresar un registro mientras se recupera la base de datos	Para mantener la integridad de los datos, se ha propuesto el siguiente horario de backups: Completo: Se hará cada domingo de cada semana a las 2:00 am Diferencial: Se hará cada día en el horario de las 11:00 pm Transaccional: Se hará cada 15 minutos de las 24 horas del día Para verificar el backup, se ha propuesto hacer una restauración completa cada semana de manera manual en una máquina de pruebas

❖ Creación de Jobs

```sql
USE master;
GO
ALTER DATABASE GestionHospitalaria SET RECOVERY FULL;
GO

--Para backup
--Full
BACKUP DATABASE GestionHospitalaria
TO DISK = 'C:\Backups\GestionHospitalaria_FULL.bak'
WITH FORMAT, NAME = N'GestionHospitalaria_FULL', COMPRESSION, STATS = 5;


--Diferencial
BACKUP DATABASE GestionHospitalaria
TO DISK = 'C:\Backups\GestionHospitalaria_DIFF.bak'
WITH DIFFERENTIAL, NAME = N'GestionHospitalaria_DIFF', COMPRESSION, STATS = 5;

--Log
BACKUP LOG GestionHospitalaria
TO DISK = 'C:\Backups\GestionHospitalaria_LOG.trn', COMPRESSION;


--Para restaurar
--Full
RESTORE DATABASE GestionHospitalaria
FROM DISK = 'C:\Backups\GestionHospitalaria_FULL.bak'
WITH NORECOVERY;

--Diferencial
RESTORE DATABASE GestionHospitalaria
FROM DISK = 'C:\Backups\GestionHospitalaria_DIFF.bak'
WITH NORECOVERY;

--Log
RESTORE LOG GestionHospitalaria
FROM DISK = 'C:\Backups\GestionHospitalaria_LOG.trn'
WITH RECOVERY;
```

Para la creación de “jobs”, se utilizó las query “para backup”. De ser necesario, se podrían utilizar las query de “para restaurar”, para comprobar la integridad del backup.

Los jobs construidos fueron:



❖ Consultas avanzadas, funciones ventanas, índices, etc.

➢ Vistas y funciones ventanas

Una vista es un objeto de la base de datos que representa una consulta almacenada.
Actúa como una tabla virtual que muestra datos provenientes de una o varias tablas, pero sin almacenar físicamente esos datos, excepto en el caso de vistas indexadas.

Para la base de datos de Gestión Hospitalaria se han implementado diferentes vistas, para facilitar la visualización rápida y eficiente de los diferentes datos dentro de las tablas.
Se han creado las siguientes vistas básicas

- **VW_Paciente** : Permite ver los datos de los pacientes registrados
- **VW_EmailPaciente:** Permite ver todos los emails del paciente registrados a su nombre.
- **VW_TelefonoPaciente:** Permite ver todos los teléfonos del paciente registrados a su nombre.
- **VW_Medico**: Permite ver los datos de los médicos registrados
- **VW_EmailMedico**: Permite ver todos los emails del médico registrados a su nombre.
- **VW_TelefonoMedico**: Permite ver todos los teléfonos del médico registrados a su nombre.
- **VW_Medicamento:** Permite ver todos los medicamentos del catálogo.
- **VW_FacturaCita**: Permite ver todas las facturas, mostrando el nombre del paciente, el tratamiento realizado y el médico que realizó el tratamiento de esa factura.

Se han creado las siguientes vistas usando funciones ventanas

- **VW_TotalPorPaciente:** Esta vista permite ver el total de dinero que cada paciente ha generado con las facturas de sus diversas citas.
- **VW_DemandaMedicamento**: Permite ver un ranking de los medicamentos más recetados.
- **VW_MedicosRanking**: Esta vista permite ver un ranking de los médicos que más ingresos han generado.
- **VW_IngresosAcumuladosFechas**: Permite ver el dinero generado por fechas.

	vista	object_id	create_date
1	vw_TotalPorPaciente	18099105	2025-11-21 11:42:56.967
2	vw_DemandaMedicamentos	50099219	2025-11-21 12:05:04.490
3	vw_MedicosRanking	66099276	2025-11-21 12:11:47.820
4	vw_IgresosAcumuladosFechas	82099333	2025-11-21 12:39:00.813
5	VW_FActuraCita	130099504	2025-11-24 12:42:12.523
6	vw_IndicesDetalles	338100245	2025-11-27 17:25:32.050
7	VW_Paciente	1749581271	2025-11-15 20:15:26.607
8	VW_Medico	1765581328	2025-11-15 20:21:30.970
9	VW_Medicamento	1781581385	2025-11-15 20:30:47.037
10	VW_EmailPaciente	1797581442	2025-11-15 20:40:55.463
11	VW_TelefonoPaciente	1813581499	2025-11-15 20:54:27.540
12	vw_UsuariosRoles	1973582069	2025-11-16 19:08:34.290
13	VW_EmailMedicos	2005582183	2025-11-16 19:13:14.603
14	VW_TelefonoMedicos	2021582240	2025-11-16 19:13:23.987
15	VW_Indices	2037582297	2025-11-16 19:13:37.393
16	VW_Roles	2053582354	2025-11-16 19:13:54.307
17	vw_AuditoriaEventos	2133582639	2025-11-20 23:03:32.217

➢ **Funciones**

Una función es un objeto de la base de datos que contiene un conjunto de instrucciones SQL diseñadas para recibir parámetros, realizar un cálculo o proceso, y devolver un valor.
Siempre retorna un resultado: ya sea un valor escalar, una tabla o un conjunto de valores.

- FN_FacturasPorFecha : Esta función devuelve una tabla que muestra las facturas dentro de un espacio de fechas que se pasa como parámetro, útil para ver facturas dentro de un espacio de tiempo.
- FN_CalcularCosto: Esta función permite calcular el costo total de una factura, sumando el precio del tratamiento y el precio de los medicamentos utilizados, teniendo en cuenta la cantidad de medicamentos usados.

	Funcion	Esquema	Tipo	create_date	modify_date
1	FN_CalcularCosto	dbo	SQL_SCALAR_FUNCTION	2025-11-15 21:17:02.377	2025-11-23 15:32:25.253
2	fn_diagramobjects	dbo	SQL_SCALAR_FUNCTION	2025-11-15 15:10:11.347	2025-11-15 15:10:11.347
3	FN_FacturasPorFechas	dbo	SQL_INLINE_TABLE_VALUED_FUNCTION	2025-11-16 19:12:22.033	2025-11-23 15:21:35.583

➢ **Procedimientos almacenados**

Un proceso almacenado es un objeto de la base de datos que contiene un conjunto de instrucciones SQL precompiladas, diseñadas para ejecutar operaciones específicas como inserciones, actualizaciones, eliminaciones, consultas o lógica de negocio completa.

Para la base de datos de Gestión Hospitalaria se han creado diferentes procesos almacenados para poder insertar datos en las diferentes tablas, así evitando que se hagan insert directos a las tablas de las bases de datos y facilitando el dar permisos solo de los procesos almacenados a usuarios y sus roles.

Se han creado los siguientes procesos almacenados

- **USP_RegistrarMedico**: Permite registrar a un médico, con su nombre, dui, especialidad, además de un teléfono y un email.
- **USP_RegistrarPaciente**: Permite registrar a un paciente, con su nombre, dui, número de seguro, además de un teléfono y un email.
- **USP_RegistrarEmailMedico** : Registra un email validando que este tenga forma de un email válido y lo guarda en la tabla de EmailMedico
- **USP_RegistrarTelefonoMedico** : Registra un teléfono validando que tenga +503 es decir que sea de El Salvador y lo agrega en la tabla TelefonoMedico.
- **USP_RegistrarEmailPaciente**: Registra un email validando que este tenga forma de un email válido y lo guarda en la tabla de EmailPaciente
- **USP_RegistrarTelefonoPaciente:** Registra un teléfono y lo agrega en la tabla TelefonoPaciente.
- **USP_RegistrarMedicamento**: Registra un medicamento a la base de datos.

- **USP_RegistrarFactura**: Registra una factura, haciendo un insert en las tablas de detalle_tratamiento, detallexmedicamento y factura_cita, además de agregar el id del paciente y médico por medio del dui, por último para generar el número de cita utiliza la secuencia SEQ_FacturaNumero.

9	USP_RegistrarEmailMedico	dbo	2085582468	2025-11-16 19:14:30.643	2025-11-23 15:25:24.550
10	USP_RegistrarEmailPaciente	dbo	1861581670	2025-11-15 22:27:33.457	2025-11-23 15:33:13.903
11	USP_RegistrarFactura	dbo	1909581841	2025-11-15 23:32:46.720	2025-11-23 15:34:51.777
12	USP_RegistrarMedicamento	dbo	2117582582	2025-11-16 19:14:57.523	2025-11-23 15:26:44.623
13	USP_RegistrarMedico	dbo	2069582411	2025-11-16 19:14:14.540	2025-11-23 15:25:18.100
14	USP_RegistrarPaciente	dbo	1845581613	2025-11-15 22:08:58.367	2025-11-23 15:33:00.350
15	USP_RegistrarTelefonoMedico	dbo	2101582525	2025-11-16 19:14:42.717	2025-11-23 15:25:52.270
16	USP_RegistrarTelefonoPaciente	dbo	1877581727	2025-11-15 22:32:04.930	2025-11-23 15:33:28.793

➢ Índices

Un índice es una estructura de datos especial que mejora la velocidad de búsqueda y recuperación de información en una tabla o vista. Para la base de datos de Gestión Hospitalaria se han creado cuatro índices para mejorar la velocidad de búsqueda en general de las consultas y de las vistas, estos índices se han creado como NONCLUSTERED para tener varios índices por tablas y para no alterar el orden físico de la tabla, los índices se realizaron en la tabla factura_cita en las columnas del los id de médicos y pacientes, para mejorar la búsquedas de los ids en las vistas, además de los id de tratamiento y medicamento, las cuales son columnas de la tabla cruzada DETALLEXMEDICAMENTO y la tabla DETALLE_TRATAMIENTO.

Además para mejorar y facilitar el proceso de reconstrucción de índices se ha realizado un proceso almacenado para reconstruir los índices. El cual es SP_ReconstruirIndices.

	Tabla	Columna	Indice	Tipo	Fragmentacion
1	FACTURA_CITA	id_medico	IX_FacturaCita_Medico	NONCLUSTERED INDEX	5.55555555555556
2	FACTURA_CITA	id_paciente	IX_FacturaCita_Paciente	NONCLUSTERED INDEX	5.55555555555556
3	DETALLE_TRATAMIENTO	id_tratamiento	IX_DetalleTratamiento_Tratamiento	NONCLUSTERED INDEX	7.69230769230769
4	DETALLEXMEDICAMENTO	id_medicamento	IX_DetalleMedicamento_Medicamento	NONCLUSTERED INDEX	5.55555555555556

➢ Triggers

Un trigger (disparador) es un objeto de la base de datos que contiene un conjunto de instrucciones SQL que se ejecutan automáticamente cuando ocurre un evento específico en una tabla o en la base de datos.

En la base de datos se han creado tres triggers como forma de garantizar y validar que los valores ingresados relacionados con dinero no puedan ser negativos, esto para enviar problemas en cálculos o análisis de las ganancias, los triggers se han implementado en las tablas DETALLE_TRATAMIENTO, MEDICAMENTO y FACTURA_CITA, al momento de insertar o actualizar.

	Trigger	Esquema	ObjetoPadreID	ObjetoPadre	Tipo	create_date	modify_date
1	TRG_ValidarPrecioMedicamento	Catalogo	1317579732	MEDICAMENTO	SQL_TRIGGER	2025-11-15 23:54:23.073	2025-11-20 22:30:50.037
2	TRG_ValidarCostoTotalFactura	Cita	1189579276	FACTURA_CITA	SQL_TRIGGER	2025-11-15 23:55:18.950	2025-11-20 22:30:20.930
3	TRG_ValidarPrecioTratamiento	Tratamiento	1269579561	DETALLE_TRATAMIENTO	SQL_TRIGGER	2025-11-15 23:53:30.560	2025-11-20 22:30:25.583

❖ Migración/carga de Datos

Para el proceso de migración de datos hacia la base de datos Gestión Hospitalaria, se implementaron dos métodos complementarios:

- Procedimientos almacenados en SQL Server
- Comandos ejecutados desde la terminal utilizando la utilidad bcp

El objetivo de estos métodos es garantizar una carga de datos controlada, segura y replicable, tanto para archivos con extensión .csv como para .dat, los cuales fueron los formatos utilizados en el proyecto.

➢ Carga de datos mediante procedimientos almacenados

Con el fin de generalizar la importación y permitir reutilizar la misma lógica, se seleccionaron las tablas: PACIENTE, MEDICO y MEDICAMENTO, debido a que representan las entidades con mayor volumen de información y con datos considerados críticos dentro del sistema, esto hace necesario un manejo más controlado durante los procesos de importación, por esto mismo, se optó por desarrollar procedimientos almacenados específicos para cada entidad, para garantizar consistencia y seguridad en la carga de datos. En el sistema se crearon 3 procedimientos almacenados: USP_ImportarPacientes, USP_ImportarMedicos y USP_ImportarMedicamento, aunque cada uno inserta información en tablas distintas según su propósito, los 3 comparten exactamente la misma lógica interna, cada procedimiento recibe como parámetro la ruta de un archivo externo y valida que su extensión sea .csv o .dat, evitando formatos no permitidos. Una vez validado, construye dinámicamente un comando BULK INSERT, el cual importa todos los registros del archivo hacia la tabla correspondiente, ignorando la primera fila por ser el encabezado y utilizando comas como separadores entre campos, finalmente este comando se ejecuta para realizar la carga de datos de manera rápida, eficiente y segura.

Este enfoque permite que la importación se ejecute siempre con una estructura definida, minimizando errores de formato y asegurando un control centralizado, asegurando que cada tabla tenga su propio proceso estandarizado.

➢ Importación y exportación de datos desde la terminal (BCP)

Como segundo método, se utilizó la herramienta nativa de SQL Server bcp (Bulk Copy Program), la cual permite importar y exportar datos desde la línea de comandos sin necesidad

de abrir SQL Server Management Studio. Este método es ideal para flujos automatizados, scripts o migraciones masivas.

- **Exportación de datos a un archivo CSV de la tabla PACIENTE**

```
C:\Users\oquij>bcp GestionHospitalaria.Paciente.PACIENTE out "C:\exportaciones\paciente2.csv" -c -t, -T -S localhost

Starting copy...
1000 rows successfully bulk-copied to host-file. Total received: 1000
1000 rows successfully bulk-copied to host-file. Total received: 2000
1000 rows successfully bulk-copied to host-file. Total received: 3000
1000 rows successfully bulk-copied to host-file. Total received: 4000
1000 rows successfully bulk-copied to host-file. Total received: 5000
1000 rows successfully bulk-copied to host-file. Total received: 6000
1000 rows successfully bulk-copied to host-file. Total received: 7000
1000 rows successfully bulk-copied to host-file. Total received: 8000
1000 rows successfully bulk-copied to host-file. Total received: 9000
1000 rows successfully bulk-copied to host-file. Total received: 10000

10004 rows copied.
Network packet size (bytes): 4096
Clock Time (ms.) Total     : 1      Average : (10004000.00 rows per sec.)
```

- **Exportación de datos a un archivo DAT de la tabla PACIENTE**

```
C:\Users\oquij>bcp GestionHospitalaria.Paciente.PACIENTE out "C:\exportaciones\paciente3.dat" -c -t, -T -S localhost

Starting copy...
1000 rows successfully bulk-copied to host-file. Total received: 1000
1000 rows successfully bulk-copied to host-file. Total received: 2000
1000 rows successfully bulk-copied to host-file. Total received: 3000
1000 rows successfully bulk-copied to host-file. Total received: 4000
1000 rows successfully bulk-copied to host-file. Total received: 5000
1000 rows successfully bulk-copied to host-file. Total received: 6000
1000 rows successfully bulk-copied to host-file. Total received: 7000
1000 rows successfully bulk-copied to host-file. Total received: 8000
1000 rows successfully bulk-copied to host-file. Total received: 9000
1000 rows successfully bulk-copied to host-file. Total received: 10000

10004 rows copied.
Network packet size (bytes): 4096
Clock Time (ms.) Total     : 1      Average : (10004000.00 rows per sec.)
```

- **Exportación de datos a un archivo CSV de la tabla MEDICAMENTO**

```
C:\Users\oquij>bcp GestionHospitalaria.Catalogo.MEDICAMENTO out "C:\exportaciones\medicamento2.csv" -c -t, -T -S localho
st

Starting copy...

54 rows copied.
Network packet size (bytes): 4096
Clock Time (ms.) Total     : 1      Average : (54000.00 rows per sec.)
```

- **Exportación de datos a un archivo DAT de la tabla MEDICAMENTO**

```
C:\Users\oquij>bcp GestionHospitalaria.Catalogo.MEDICAMENTO out "C:\exportaciones\medicamento3.dat" -c -t, -T -S localho
st

Starting copy...

54 rows copied.
Network packet size (bytes): 4096
Clock Time (ms.) Total     : 1      Average : (54000.00 rows per sec.)
```

- **Exportación de datos a un archivo CSV de la tabla MEDICO**

```
C:\Users\oquij>bcp GestionHospitalaria.Medico.MEDICO out "C:\exportaciones\medico2.csv" -c -t, -T -S localhost

Starting copy...

104 rows copied.
Network packet size (bytes): 4096
Clock Time (ms.) Total     : 1      Average : (104000.00 rows per sec.)
```

- **Exportación de datos a un archivo DAT de la tabla MEDICO**

```
C:\Users\oquij>bcp GestionHospitalaria.Medico.MEDICO out "C:\exportaciones\medico3.dat" -c -t, -T -S localhost

Starting copy...

104 rows copied.
Network packet size (bytes): 4096
Clock Time (ms.) Total     : 1      Average : (104000.00 rows per sec.)
```

- **Importación desde un archivo CSV para la tabla de PACIENTES**

```
C:\Users\oquij>bcp GestionHospitalaria.Paciente.PACIENTE in "C:\importaciones\paciente.csv" -c -t, -S localhost -T -F 2

Starting copy...

2 rows copied.
Network packet size (bytes): 4096
Clock Time (ms.) Total     : 1      Average : (2000.00 rows per sec.)
```

- **Importación desde un archivo CSV para la tabla MEDICAMENTO**

```
C:\Users\oquij>bcp GestionHospitalaria.Catalogo.MEDICAMENTO in "C:\importaciones\medicamento.csv" -c -t, -S localhost -T
 -F 2

Starting copy...

2 rows copied.
Network packet size (bytes): 4096
Clock Time (ms.) Total     : 1      Average : (2000.00 rows per sec.)
```

- **Importación desde un archivo CSV para la tabla MEDICO**

```
C:\Users\oquij>bcp GestionHospitalaria.Medico.MEDICO in "C:\importaciones\medico.csv" -c -t, -S localhost -T -F 2

Starting copy...

2 rows copied.
Network packet size (bytes): 4096
Clock Time (ms.) Total     : 16     Average : (125.00 rows per sec.)
```

- **Ejemplo de importación desde un archivo DAT**

```
C:\Users\oquij>bcp GestionHospitalaria.dbo.PACIENTE in "C:\backups\pacientes_ejemplo.dat" -c -t, -T -S localhost
```

❖ Dashboard en PowerBI

Para el proyecto de la base de datos Gestión Hospitalaria se ha creado una dashboard para el análisis de valores relacionados con dinero, este puede ser filtrado por paciente, medico, fechas(rango de fechas y por meses), permite ver los resultados de esos filtros en dos gráficos, uno de barras que muestra las ganancias por mes en general y otra que muestra un gráfico de líneas de los ingresos a través de los años, muestra también unas tarjetas con datos puntuales que van cambiando dependiendo de los filtros colocados, se pueden ver datos como los ingresos, número de citas, los pacientes totales, el valor promedio por consulta, los ingresos mensuales del mes y del anterior, además de la variación de ingresos con respecto al mes anterior, por último se creó una texto dinámico que resume todos estos datos mencionados anteriormente.