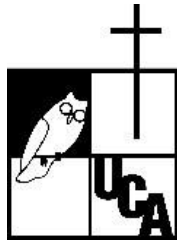


**UNIVERSIDAD CENTROAMERICANA
JOSÉ SIMEÓN CAÑAS**



**Proyecto de Administración de Bases de Datos
Sistema de Gestión Hospitalaria (ClinicaDB)**

Ingeniero:

James Edward Humberstone Morales

Integrantes:

Reynaldo Alcides Bonilla Ventura 00107117

Xochill Guissell Miguel Miranda 00114924

Julissa Uriel Patiño Figueroa 00026124

Luis Rodrigo Guzman Estrada 00003924

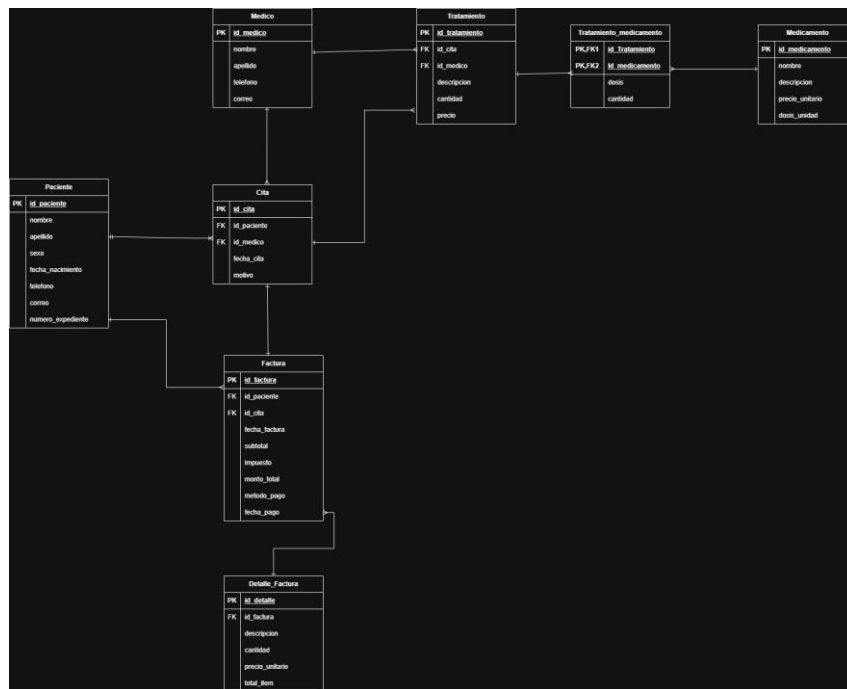
Jaime Arturo Magaña Nerio 00077320

INTRODUCCION

Este proyecto presenta el desarrollo de un Sistema de Gestión Hospitalaria simplificado, diseñado para administrar información de pacientes, médicos, citas, tratamientos y facturación. Se implementó una base de datos completa, abarcando el diseño lógico y físico, así como la creación de usuarios, roles y políticas de seguridad basadas en el principio de mínimo privilegio.

Además, se configuraron auditorías, índices y funciones ventana para mejorar el rendimiento y la trazabilidad del sistema. También se aplicaron estrategias de respaldo y recuperación, junto con la automatización de procesos mediante jobs. Finalmente, se elaboró un dashboard en Power BI conectado a SQL Server para visualizar los datos de forma clara y analítica.

Diagrama relacional



El diagrama entidad – relación fue diseñado utilizando 3 normalizaciones para poder tener un modelo bien estructurado que podemos notar que se centra en la tabla citas que conecta el flujo operativo entre paciente y médicos, algo a destacar es la integridad de los datos ya que con esto resolvemos las prescripciones con la tabla intermedia “tratamiento_medicamento” y separamos el módulo financiero de factura y detalle para escalabilidad y cero redundancias.

DESCRIPCIÓN DEL PROCESO DE INSERCIÓN DE DATOS

Este bloque describe las diferentes metodologías de inserción de datos utilizadas para poblar las tablas de la base de datos de la clinicaDB. El proceso se divide en bloques lógicos que simulan la creación de un entorno de trabajo realista con grandes volúmenes de datos. Se han

utilizando técnicas de generación de datos aleatorios y temporales (como las *Tally Tables* o Tablas de Conteo) para alcanzar las cantidades especificadas.

1. Tabla Medica.Medico (50 Registros)

Se ingresan 50 nombres, apellidos y datos de contacto específicos.

2. Tabla Admision.Paciente (5,000 Registros)

Generación Aleatoria: Se combinan 100 nombres base con 100 apellidos base y se seleccionan **5,000 registros únicos** de forma aleatoria. Se genera data simulada de fecha_nacimiento, teléfono y correo.

3. Tabla Medica.Medicamento (500 Registros)

Tabla de Conteo (Tally Table): Se usa una secuencia de 500 números para generar nombres (prefijo + dosis + sufijo) y precios unitarios **aleatorios** en lote.

4. Tabla Admision.Cita (25,000 Registros)

Uso de Tally Table: Genera 25,000 citas, asignando **id_paciente** e **id_medico** aleatorios y una fecha_cita en los últimos 5 años.

5. Tabla Medica.Tratamiento (15,000 Registros)

Asigna id_cita e id_medico aleatorios. Genera una descripción cíclica y un **precio aleatorio** para simular la facturación del servicio.

6. Tabla Finanzas.Factura (3,000 Registros)

Asigna id_paciente e id_cita aleatorios. Calcula el **impuesto** (15% sobre el subtotal) y asigna un **metodo_pago**, con un 10% de probabilidad de dejar la factura como pendiente (campos NULL).

7. Tabla Medica.Tratamiento_Medicamento (5,000 Registros)

Generación sin Duplicados: Utiliza un *staging* (preparación) temporal para generar combinaciones y luego inserta **SELECT DISTINCT TOP 5000** para asegurar que cada par (id_tratamiento, id_medicamento) sea único.

8. Tabla Finanzas.Detalle_Factura (4,000 Registros)

Asigna aleatoriamente el `id_factura`. La **descripción** y el **precio_unitario** son generados aleatoriamente para simular diferentes conceptos de cobro (consultas, análisis, curaciones).

Nota: En la base de dato se insertaron un total de 57,550 registros.

DICCIONARIO DE DATOS

En esta sección se describe la estructura general de la base de datos. Se detallan las tablas con sus atributos, tipos de dato y restricciones, lo que permite entender cómo está organizada la información y de qué manera se relacionan los distintos elementos del sistema.

○ ESQUEMAS

La base se separó por esquemas porque así es más fácil ordenar todo conforme a que área está relacionado.

Nombre del Esquema	Fecha de creación (si aplica)	Notas
Admision	Creado al inicio del proyecto	Contiene tablas relacionadas con pacientes y citas
Medica	Creado al inicio del proyecto	Contiene tablas de médicos, tratamientos y medicamentos
Finanzas	Creado al inicio del proyecto	Contiene tablas de facturación y detalle de facturas

○ TABLAS

En esta sección se detalla las tablas de la base incluidas en cada esquema, junto con sus atributos y características, señalando si el campo admite valores nulos, claves primarias y otras restricciones relevantes.

1. Paciente

Columna	Tipo	Longitud	Nulo	Identity	PK/FK/Restricciones	Notas
<code>id_paciente</code>	INT	4	NO	Sí	PK	Identificador único del paciente
<code>nombre</code>	VARCHAR	50	NO	No		Primer nombre del paciente
<code>apellido</code>	VARCHAR	50	NO	No		Apellido del paciente
<code>sexo</code>	CHAR	1	SÍ	No	CHECK (sexo IN ('M','F'))	Género del paciente
<code>fecha_nacimiento</code>	DATE	3	NO	No		Fecha de nacimiento
<code>telefono</code>	VARCHAR	20	SÍ	No		Número de contacto
<code>correo</code>	VARCHAR	100	SÍ	No		Correo electrónico
<code>numero_expediente</code>	VARCHAR	20	SÍ	No	UNIQUE	Número de expediente único

2. Médico

Columna	Tipo	Longitud	Nulo	Identity	PK/FK/Restricción	Notas
id_medico	INT	4	NO	Sí	PK	Identificador único del médico
nombre	VARCHAR	50	NO	No		Primer nombre del médico
apellido	VARCHAR	50	NO	No		Apellido del médico
telefono	VARCHAR	20	SÍ	No		Número de contacto
correo	VARCHAR	100	SÍ	No		Correo electrónico

3. Cita

Columna	Tipo	Longitud	Nulo	Identity	PK/FK/Restricciones	Notas
id_cita	INT	4	NO	Sí	PK	Identificador único de la cita
id_paciente	INT	4	NO	No	FK → Paciente(id_paciente)	Relaciona la cita con el paciente
id_medico	INT	4	NO	No	FK → Medico(id_medico)	Relaciona la cita con el médico
fecha_cita	DATE	3	NO	No		Fecha de la cita
motivo	VARCHAR	255	SÍ	No		Motivo de la cita

4. Medicamento

Columna	Tipo	Longitud	Nulo	Identity	PK/FK/Restricciones	Notas
id_medicamento	INT	4	NO	Sí	PK	Identificador único del medicamento
nombre	VARCHAR	100	NO	No		Nombre del medicamento
descripcion	TEXT	-	SÍ	No		Descripción detallada
precio_unitario	DECIMAL	10,2	NO	No		Precio por unidad
dosis_unidad	VARCHAR	50	SÍ	No		Unidad de dosis recomendada

5. Tratamiento

Columna	Tipo	Longitud	Nulo	Identity	PK/FK/Restricciones	Notas
id_tratamiento	INT	4	NO	Sí	PK	Identificador único del tratamiento
id_cita	INT	4	NO	No	FK → Cita(id_cita)	Relaciona el tratamiento con la cita
id_medico	INT	4	NO	No	FK → Medico(id_medico)	Médico responsable
descripcion	TEXT	-	SÍ	No		Detalles del tratamiento
cantidad	INT	4	SÍ	No		Cantidad de tratamientos
precio	DECIMAL	10,2	SÍ	No		Precio del tratamiento

6. Tratamiento_Medicamento

Columna	Tipo	Longitud	Nulo	Identity	PK/FK/Restricciones	Notas
id_tratamiento	INT	4	NO	No	PK, FK → Tratamiento(id_tratamiento)	Relaciona tratamiento y medicamento
id_medicamento	INT	4	NO	No	PK, FK → Medicamento(id_medicamento)	Medicamento administrado
dosis	VARCHAR	50	SÍ	No		Dosis recomendada
cantidad	INT	4	SÍ	No		Cantidad administrada

7. Factura

Columna	Tipo	Longitud	Nulo	Identity	PK/FK/Restricciones	Notas
id_factura	INT	4	NO	Sí	PK	Identificador único de la factura
id_paciente	INT	4	NO	No	FK → Paciente(id_paciente)	Paciente asociado
id_cita	INT	4	NO	No	FK → Cita(id_cita)	Cita asociada
fecha_factura	DATE	3	NO	No	DEFAULT GETDATE()	Fecha de emisión
subtotal	DECIMAL	10,2	NO	No		Monto antes de impuestos
impuesto	DECIMAL	10,2	NO	No		Impuesto aplicado
monto_total	DECIMAL	10,2	NO	No	Calculado (subtotal + impuesto)	Atributo derivado
metodo_pago	VARCHAR	50	SÍ	No		Medio de pago
fecha_pago	DATE	3	SÍ	No		Fecha de pago

8. Detalle_Factura

Columna	Tipo	Longitud	Nulo	Identity	PK/FK/Restricciones	Notas
id_detalle	INT	4	NO	Sí	PK	Identificador del detalle
id_factura	INT	4	NO	No	FK → Factura(id_factura)	Relaciona con factura
descripcion	TEXT	-	Sí	No		Detalle del item facturado
cantidad	INT	4	Sí	No		Cantidad vendida
precio_unitario	DECIMAL	10,2	Sí	No		Precio por unidad
total_item	DECIMAL	10,2	Sí	No	Calculado (cantidad * precio_unitario)	Atributo derivado

○ ROLES Y USUARIOS

En esta sección se muestra cómo se estructuraron los permisos y accesos según la función que cada uno ejecuta dentro de la clínica.

Rol	Usuario(s)	Notas
AdministradorDB	admin_clinica	Control total sobre la base de datos
MedicoRole	medico_user	Acceso a tablas de Paciente, Cita y Tratamiento (SELECT, INSERT, UPDATE)
RecepcionistaRole	recepcion_user	Acceso a tablas de Paciente y Cita (SELECT, INSERT)
FacturacionRole	facturacion_user	Acceso a Factura y Detalle_Factura (SELECT, INSERT, UPDATE)
ConsultaRole	consulta_user	Solo lectura en toda la base de datos (SELECT)
CatedraticoRole	catedratico_clinica	Permisos de SELECT, VIEW DEFINITION y EXECUTE

○ INDICES

Acá se muestran tanto los índices generados automáticamente por claves primarias y restricciones de la base, como los que se agregaron manualmente para optimizar consultas específicas.

Esquema	Tabla	Índice	Tipo	Único	Columnas	Nota
Admision	Paciente	PK__Paciente__2C2C72B8459CB3C	CLUSTERED	Si	id_paciente	Clave primaria
Admision	Paciente	UQ_Paciente_NumeroExpediente	NONCLUSTERED	Si	numero_expediente	Asegura ser único
Admision	Cita	PK__Cita__6AEC3C09B0EBEE92	CLUSTERED	Si	id_cita	Clave primaria
Admision	Cita	IX_Cita_Paciente_Fecha	NONCLUSTERED	No	id_paciente, fecha_cita	Optimiza búsquedas por paciente y fecha
Medica	Medico	PK__Medico__E038EB43ED326C0E	CLUSTERED	Si	id_medico	Clave primaria
Medica	Tratamiento	PK__Tratamiento__C8825F4C3C481576	CLUSTERED	Si	id_tratamiento	Clave primaria
Medica	Tratamiento_Medicamento	PK__TM_FADAD34FCB5A49E6	CLUSTERED	Si	id_tratamiento, id_medicamento	Clave primaria compuesta
Finanzas	Factura	IX_Factura_Fecha_Pac	NONCLUSTERED	No	fecha_factura, id_paciente	Optimiza búsquedas por fecha y paciente.

DIMENSIONAMIENTO

- Admision.Paciente

Columna	Tipo	Tamaño (bytes)
id_paciente	INT	4
nombre	VARCHAR(50)	52
apellido	VARCHAR(50)	52
sexo	CHAR(1)	1
fecha_nacimiento	DATE	3
telefono	VARCHAR(20)	22
correo	VARCHAR(100)	102
numero_expediente	VARCHAR(20)	22
Fila total	—	258
Overhead fila	—	7
Total fila	—	265
Registros	—	5,000
Tamaño total	—	1,325,000 bytes ≈ 1.26 MB

- Medica.Medico

Columna	Tipo	Tamaño (bytes)
id_medico	INT	4
nombre	VARCHAR(50)	52
apellido	VARCHAR(50)	52
telefono	VARCHAR(20)	22
correo	VARCHAR(100)	102
Fila total	—	232
Overhead fila	—	7
Total fila	—	239
Registros	—	50
Tamaño total	—	11,950 bytes \approx 11.7 KB

- Admision.Cita

Columna	Tipo	Tamaño (bytes)
id_cita	INT	4
id_paciente	INT	4
id_medico	INT	4
fecha_cita	DATE	3
motivo	VARCHAR(255)	257
Fila total	—	272
Overhead fila	—	7
Total fila	—	279
Registros	—	25,000
Tamaño total	—	6,975,000 bytes \approx 6.65 MB

- Medica.Medicamento

Columna	Tipo	Tamaño (bytes)
id_medicamento	INT	4
nombre	VARCHAR(100)	102
descripcion	TEXT	16 (puntero) + 255 promedio
precio_unitario	DECIMAL(10,2)	5
dosis_unidad	VARCHAR(50)	52
Fila total	—	434
Overhead fila	—	7
Total fila	—	441
Registros	—	500
Tamaño total	—	220,500 bytes \approx 215 KB

- Medica.Tratamiento

Columna	Tipo	Tamaño (bytes)
id_tratamiento	INT	4
id_cita	INT	4
id_medico	INT	4
descripcion	TEXT	16 + 255 promedio
cantidad	INT	4
precio	DECIMAL(10,2)	5
Fila total	—	292
Overhead fila	—	7
Total fila	—	299
Registros	—	15,000
Tamaño total	—	4,485,000 bytes \approx 4.28 MB

- Medica.Tratamiento_Medicamento

Columna	Tipo	Tamaño (bytes)
id_tratamiento	INT	4
id_medimento	INT	4
dosis	VARCHAR(50)	52
cantidad	INT	4
Fila total	—	64
Overhead fila	—	7
Total fila	—	71
Registros	—	5,000
Tamaño total	—	355,000 bytes \approx 346.7 KB

- Finanzas.Factura

Columna	Tipo	Tamaño (bytes)
id_factura	INT	4
id_paciente	INT	4
id_cita	INT	4
fecha_factura	DATE	3
subtotal	DECIMAL(10,2)	5
impuesto	DECIMAL(10,2)	5
monto_total	DECIMAL(10,2) derivado	5
metodo_pago	VARCHAR(50)	52
fecha_pago	DATE	3
Fila total	—	85
Overhead fila	—	7
Total fila	—	92
Registros	—	3,000
Tamaño total	—	276,000 bytes \approx 269.5 KB

- Finanzas.Detalle_Factura

Columna	Tipo	Tamaño (bytes)
id_detalle	INT	4
id_factura	INT	4
descripcion	TEXT	16 + 255 promedio
cantidad	INT	4
precio_unitario	DECIMAL(10,2)	5
total_item	DECIMAL(10,2) derivado	5
Fila total	—	293
Overhead fila	—	7
Total fila	—	300
Registros	—	4,000
Tamaño total	—	1,200,000 bytes \approx 1.14 MB

El tamaño real estimado de la base de datos, considerando únicamente los datos de las tablas y sin incluir índices es aproximadamente 14.17 MB. Este valor se obtuvo sumando el tamaño total estimado de cada tabla según el número de registros y el tamaño promedio por fila.

Para cubrir índices se calcula un aproximado de 1.3, en donde ese número es el factor de crecimiento y esto se hace aplicando la siguiente formula:

$$\text{Tamaño estimado (MB)} = ((\text{Tamaño fila promedio} + 7) \times \text{Filas esperadas} \times 1.3) / (1024 \times 1024)$$

Tabla	Tamaño fila promedio (bytes)	Overhead fila (bytes)	Filas esperadas	Fórmula (bytes)	Tamaño estimado (MB)
Admision.Paciente	258	7	5000	$(258 + 7) * 5000 * 1.3$	1.64
Medica.Medico	232	7	50	$(232 + 7) * 50 * 1.3$	0.0015
Admision.Cita	272	7	25,000	$(272 + 7) * 25,000 * 1.3$	8.67
Medica.Medicamento	434	7	500	$(434 + 7) * 500 * 1.3$	0.27
Medica.Tratamiento	292	7	15,000	$(292 + 7) * 15,000 * 1.3$	5.67
Medica.Tratamiento_Medicamento	64	7	5,000	$(64 + 7) * 5,000 * 1.3$	0.41
Finanzas.Factura	85	7	3,000	$(85 + 7) * 3,000 * 1.3$	0.31
Finanzas.Detalle_Factura	293	7	4,000	$(293 + 7) * 4,000 * 1.3$	1.56
Total	-	-	57,550	-	18.53

CREACIÓN DE USUARIOS, ROLES Y POLÍTICAS DE SEGURIDAD

Esta sección detalla la implementación de la gestión de usuarios, roles y políticas de seguridad en la base de datos *ClinicaDB* del sistema de gestión hospitalaria.

El propósito fue garantizar que cada tipo de usuario tuviera acceso únicamente a los recursos necesarios para cumplir sus funciones, aplicando el principio de mínimo privilegio y manteniendo la integridad y confidencialidad de la información.

1. Creación de Logins

Se crean los usuarios principales que accederán al servidor SQL: administrador, médico, recepcionista, facturación, consulta y catedrático.

Todos llevan contraseñas seguras y políticas de seguridad activas.

2. Creación de Roles

Con el fin de simplificar la administración y control de privilegios, se definieron roles específicos según las funciones del sistema.

Roles definidos:

- AdministradorDB: Control total del sistema.
- MedicoRole: Manejo de información clínica.
- RecepcionistaRole: Registro de pacientes y citas.
- FacturacionRole: Gestión de facturación.
- ConsultaRole: Solo lectura.
- CatedraticoRole: Lectura y análisis para revisión académica.

3. Asignación de Roles a Usuarios

Cada usuario fue asignado al rol correspondiente a sus funciones.

4. Políticas de Permisos

Los permisos se otorgaron según el nivel de acceso requerido por cada rol:

ROL	PERMISOS
Administrador	Control total sobre toda la base de datos.
Médico	- Consultar, insertar y actualizar paciente, cita y tratamiento. - Consultar Medicamento.
Recepcionista	- Consultar e insertar Paciente y Cita.
Facturación	- Consultar, insertar y actualizar Factura y Detalle_Factura.
Consulta	Solo lectura general de toda la base de datos.
Catedrático	- Solo lectura de datos. - Lectura de definiciones de objetos. - Permiso para ejecutar procedimientos almacenados pero no crearlos

5. Verificación de Roles y Miembros

Una consulta final muestra todos los usuarios y los roles a los que pertenecen, para confirmar que la configuración está correcta.

6. Políticas de Seguridad Adicionales

- Se aplicó CHECK_POLICY = ON en todos los logins para reforzar la seguridad de contraseñas
- Se implementó CHECK_EXPIRATION = ON Obliga a los usuarios a cambiar su contraseña periódicamente, cumpliendo el ciclo de expiración definido
- Se agregó un usuario de revisión con permisos limitados

7. Tabla Resumen de Permisos por Rol

Rol / Usuario	SELECT	INSERT	UPDATE	CONTROL (Total)	VIEW DEFINITION	EXECUTE
AdministradorDB (admin_clinica)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
MedicoRole (medico_user)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
RecepcionistaRole (recepcion_user)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
FacturacionRole (facturacion_user)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ConsultaRole (consulta_user)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
CatedraticoRole (catedratico_clinica)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

ESPECIFICACION DE AUDITORIA

1. Creación de la auditoría a nivel servidor

Se creó la auditoría *AuditoriaClinica* a nivel de servidor, indicando que los registros serían almacenados en un archivo dentro de una carpeta. Posteriormente, la auditoría fue habilitada.

2. Creación de la especificación de auditoría a nivel de base de datos

Dentro de la base de datos *ClinicaDB*, se creó la especificación *AuditoriaClinicaDB*, en la cual se definieron las tablas y los tipos de acciones que serían auditadas.

En este caso, se configuraron los eventos SELECT, INSERT y UPDATE sobre las tablas Paciente, Cita, Tratamiento, Factura y Detalle_Factura.

Con esto, cualquier usuario que interactúe con esos objetos generará registros en la auditoría.

3. Ejecución de pruebas de auditoría

Para validar el funcionamiento del sistema de auditoría, se realizaron operaciones de prueba dentro de la tabla *Paciente*, incluyendo una inserción y una actualización. Estas acciones permitieron generar eventos en los archivos de auditoría para comprobar que el sistema estaba registrando correctamente las actividades realizadas.

4. Consulta general de los logs de auditoría

Se utilizó la función del sistema sys.fn_get_audit_file para leer todos los registros generados por la auditoría desde los archivos almacenados en disco.

Esta consulta permitió visualizar información como el usuario que ejecutó la acción, el tipo de evento, la sentencia realizada y el objeto afectado.

5. Filtrado de los eventos por tipo de operación

Para obtener una vista más detallada, se ejecutaron consultas adicionales filtrando exclusivamente los eventos SELECT, INSERT, UPDATE y DELETE.

Con esto se pudo identificar de manera específica qué tipo de acciones se habían realizado en la base de datos y qué usuario lo realiza

ESTRATEGIA DE BACKUPS

En la estrategia de backups de multi-nivel para nuestra base de datos ClinicaDB utilizamos lo que es el backup full que es la base inicial para la recuperación , el backup diferencial que este lo que hace es capturar cambios desde el ultimo backup full que se realizó y por último el backup de transacciones (LOG) que este nos permite tener un point-in-time recovery con los cuales tenemos objetivos cumplidos con RTO porque tenemos una recuperación rápida con el backup diferencia que nos reduce el tiempo de restore y RPO con el cual tenemos la mínima perdida de datos gracias al backup log frecuente .

```
-- 1. BACKUP FULL
BACKUP DATABASE ClinicaDB
TO DISK = 'C:\Backups\ClinicaDB_Full.bak'
WITH
    FORMAT,
    INIT,
    NAME = 'Backup Completo ClinicaDB',
    DESCRIPTION = 'Copia completa de seguridad de ClinicaDB - Proyecto Final',
    STATS = 10,
    CHECKSUM;

PRINT ' BACKUP COMPLETO EXITOSO: C:\Backups\ClinicaDB_Full.bak';
GO
```

```

-- 3. BACKUP DIFERENCIAL
USE master;
GO

-- 4. BACKUP DIFERENCIAL
BACKUP DATABASE ClinicaDB
TO DISK = 'C:\Backups\ClinicaDB_Diff.bak'
WITH
    DIFFERENTIAL,
    INIT,
    NAME = 'Backup Diferencial ClinicaDB',
    DESCRIPTION = 'Copia diferencial de ClinicaDB - Cambios desde ultimo completo',
    STATS = 10,
    CHECKSUM;

PRINT ' BACKUP DIFERENCIAL EXITOSO: C:\Backups\ClinicaDB_Diff.bak';
GO

-- 5. BACKUP DE LOG DE TRANSACCIONES
USE master;
GO

BACKUP LOG ClinicaDB
TO DISK = 'C:\Backups\ClinicaDB_Log.trn'
WITH
    INIT,
    NAME = 'Backup del Log ClinicaDB',
    DESCRIPTION = 'Backup del log de transacciones de ClinicaDB',
    STATS = 10,
    CHECKSUM;

PRINT ' BACKUP DE LOG EXITOSO: C:\Backups\ClinicaDB_Log.trn';
GO

```

Con esto tenemos la ventaja que tenemos modelo FULL recovery para tener una máxima protección, verificación de checksum para integridad, el monitorio del backup full y tener un recovery hasta el punto de fallo.

JOBS

En el proyecto se implementaron tres **jobs en SQL Server Agent** con el objetivo de automatizar tareas críticas de respaldo y mantenimiento dentro de la base de datos *ClinicaDB*. La presencia de estos jobs garantiza la disponibilidad, integridad y continuidad operativa del sistema hospitalario, cumpliendo con buenas prácticas de administración de bases de datos.

1. Backup_Diario_ClinicaDB (Backup completo diario)

Este job ejecuta un **backup FULL** de la base de datos cada día a las 00:00 horas.

Incluye además una verificación automática mediante `RESTORE VERIFYONLY` para asegurar que el archivo generado sea válido.

Su función es proporcionar un punto de recuperación completo que respalde toda la información crítica del hospital, como pacientes, citas, diagnósticos y facturación.

2. Backup_Diferencial_6Horas (Backup diferencial periódico)

Este job realiza un **backup diferencial cada 6 horas**, copiando únicamente los cambios ocurridos desde el último respaldo completo.

Con esto se optimiza el uso de espacio, se reducen los tiempos de ejecución y se garantiza que, ante un fallo, la pérdida de información sea mínima.

Este tipo de respaldo es especialmente importante en un entorno hospitalario donde se registran datos de manera constante.

3. Limpieza_Backups_Antiguos (Mantenimiento del historial de backups)

El tercer job se ejecuta de forma **semanal** y elimina del historial de *msdb* los registros de backups con más de 30 días.

Su propósito es mantener ordenado el sistema, evitar acumulación excesiva de registros y mejorar la eficiencia en tareas administrativas relacionadas con los respaldos.

Importancia dentro del sistema

La implementación de estos jobs aporta importantes beneficios:

- **Automatizan la estrategia de respaldo**, evitando pérdida de información.
- **Reducen riesgos** mediante puntos de recuperación frecuentes.
- **Mantienen el servidor organizado y eficiente**, gracias a la limpieza automática.
- **Reflejan prácticas profesionales** de administración y mantenimiento de bases de datos.
- **Fortalecen la disponibilidad del sistema**, asegurando que los datos clínicos estén protegidos en todo momento.

En conjunto, estos jobs proporcionan una infraestructura sólida de respaldo y recuperación para la base de datos ClinicaDB, alineada a las necesidades de un sistema hospitalario real.

INDICES

Se implementaron cinco índices no agrupados con el objetivo de optimizar consultas críticas del sistema. La elección de índices NONCLUSTERED permitió mejorar significativamente la velocidad de búsqueda y filtrado sin alterar el orden físico de las tablas ni incrementar innecesariamente el costo de inserción.

Para medir el impacto real de cada índice, se usaron las instrucciones SET STATISTICS IO/TIME ON, registrando el rendimiento antes y después de su creación.

Índice	Tabla / Columnas	Uso	Problema	Antes (CPU / Elapsed)	Después (CPU / Elapsed)	Beneficio
IX_Cita_Medico	Cita(id_medico)	Consultar citas por médico	Consultas frecuentes de agenda e historial.	CPU 16 ms / 204 ms	CPU 0 ms / 57 ms	Acceso rápido a citas por médico sin escanear toda la tabla
IX_Cita_Paciente_Fecha	Cita(id_paciente, fecha_cita)	Filtrado de citas por paciente y fecha	Filtros combinados necesitaban ordenar y recorrer toda la tabla	CPU 78 ms / 545 ms	CPU 47 ms / 144 ms	Búsquedas eficientes por paciente con fechas ya ordenadas
IX_Cita_Fecha	Cita(fecha_cita)	Listados por fecha (día/semana)	Reportes diarios o semanales tardaban	CPU 0 ms / 200 ms	CPU 0 ms / 124 ms	Localiza citas por fecha de forma directa y más rápida
IX_Factura_Fecha_Pac	Factura(fecha_factura, id_paciente)	Reportes de facturación por periodo	El módulo contable necesitaba filtrar por fecha y paciente sin índice	CPU 16 ms / 204 ms	CPU 16 ms / 109 ms	Lecturas mínimas y búsqueda directa de facturas en rangos
IX_TM_Medicamento	Tratamiento_Medicamento(id_medimento)	Análisis de medicamentos usados	Consultas de inventario y frecuencia escaneaban toda la tabla	CPU 0 ms / 91 ms	CPU 0 ms / 23 ms	Recuperación rápida de todos los tratamientos por medicamento

FUNCIONES VENTANA

Se implementaron para generar rankings, totales, diferencias en consultas sobre médicos, pacientes y medicamentos. Estas consultas se benefician de los índices creados previamente, lo que optimiza la ejecución y reduce los tiempos de respuesta.

A continuación, se presenta un resumen de estas y su propósito en el sistema:

Función	Consulta	Tabla consultada	Partición / Orden	Propósito	Índice que acelera la consulta
RANK()	RankingMedicos	Medico / Cita	ORDER BY COUNT(c.id_cita) DESC	Ranking de médicos según cantidad de citas	IX_Cita_Medico
LAG()	DiferenciaCitas	Cita	PARTITION BY id_paciente ORDER BY fecha_cita	Calcular días entre citas consecutivas de cada paciente	IX_Cita_Paciente_Fecha
ROW_NUMBER()	MedicamentosUso	Medicamento / Tratamiento_Medicamento	ORDER BY [Veces Usado] DESC	Top 5 medicamentos más usados	IX_TM_Medicamento
COUNT() OVER + ROW_NUMBER()	CitasPaciente	Cita / Paciente	PARTITION BY id_paciente ORDER BY fecha_cita DESC	Top 3 de pacientes con más citas	IX_Cita_Paciente_Fecha

ANEXO

Link del video de presentacion del proyecto

<https://youtu.be/twN98rnEP2w?si=-EddtClNz37Q6f3H>