UNIVERSIDAD CENTROAMERICANA
JOSÉ SIMEÓN CAÑAS
DEPARTAMENTO DE ELECTRÓNICA E INFORMÁTICA

ADMINISTRACIÓN DE BASES DE DATOS

Proyecto de Cátedra

Sistema de Gestión Hospitalaria

Estudiantes:

Penado Diaz, Christian Adonay, 00046724

Jaime Antonio Perez Shupan, 00202124

Diego Pérez Herrera, 00543924

Samuel Alejando Perez Hernandez, 00052424

Juan Carlos Rivera Meléndez 00029324

David Alessandro Ventura Montoya, 00089724

Encargado de la asignatura:

Nombre: James Edward Humberstone Morales

Correo: jehumberstone@uca.edu.sv

Entrega del reporte: 27/11/25.

Descripción del sistema elegido: Sistema de Gestión Hospitalaria

El sistema que se ha seleccionado a desarrollar es el Sistema de Gestión Hospitalaria, el cual tiene como objetivo administrar eficazmente la información de pacientes, médicos, consultas, tratamientos y facturación del Hospital.

La base de datos creada tiene como nombre PHospital, y fue desarrollada en el sistema de Administración de bases de datos relacionales llamada SQL Server Management Studio. Se tomó como prioridad poner en práctica los procesos más comunes que ocurren en un hospital, garantizando la optimización y seguridad en el manejo de datos.

Las funciones más destacadas del sistema son:

- **Gestión de pacientes:** Registro de datos personales, tipo de sangre, aseguradora, dirección y contacto.
- **Gestión de médicos:** Administración de información personal y especialización médica.
- **Consultas médicas:** Registro de citas, motivos de consulta, estado y tratamientos aplicados.
- **Tratamientos:** Catálogo de procedimientos y servicios médicos con sus respectivos costos.
- **Facturación:** Generación automática de facturas y detalle de servicios prestados a cada paciente.
- **Seguridad y control:** Uso de procedimientos almacenados para garantizar consistencia en la inserción de datos y control de operaciones críticas.
- **Optimización de consultas:** Implementación de vistas y funciones ventana para obtener estadísticas como el costo promedio de consultas por médico.
- **Compatibilidad sanguínea:** Procedimiento que permite identificar pacientes con tipos de sangre compatibles, útil en escenarios de transfusión.

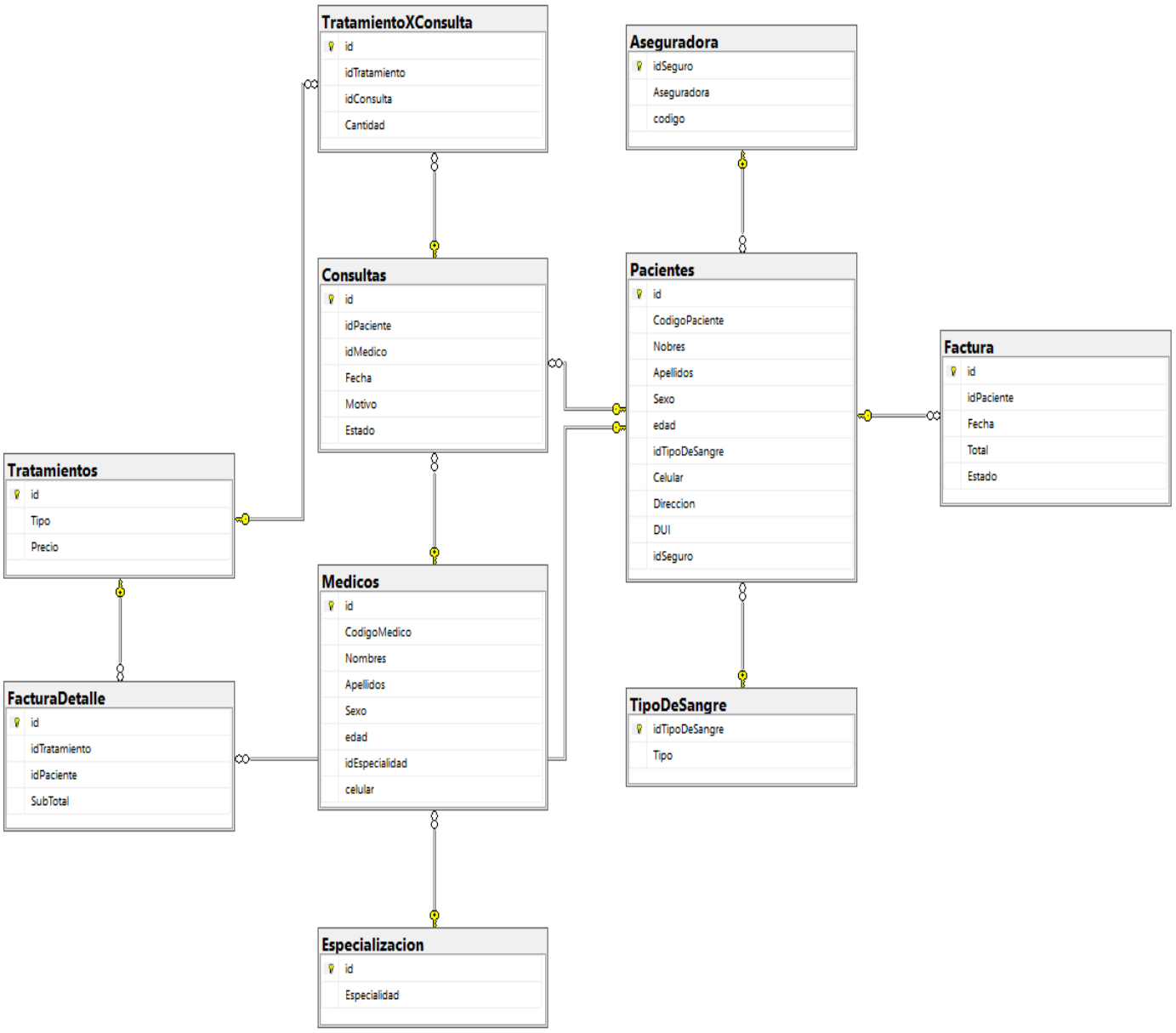
El desarrollo se enfoca en la implementación de un modelo relacional robusto, la definición de políticas de seguridad, la optimización de consultas mediante funciones avanzadas y la integración con herramientas de visualización como Power BI. Este enfoque nos permite consolidar competencias técnicas en diseño lógico y físico de bases de datos, automatización de procesos con procedimientos almacenados, y análisis de datos en entornos reales de gestión hospitalaria.

Diariamente la administración hospitalaria en su mayoría, enfrenta problemas de duplicidad de información, dificultad en el acceso a historiales médicos y errores en la facturación. Este sistema busca resolver dichas limitaciones mediante un modelo relacional centralizado que garantice integridad y seguridad de datos, con el fin de mejorar la eficiencia administrativa y calidad de servicio.

Objetivos Específicos:

- Garantizar y asegurar las relaciones entre tablas mediante claves primarias y foráneas.
- Diseñar y ejecutar consultas optimizadas mediante funciones ventana, complementadas con índices adecuados, que permitan obtener estadísticas relevantes de la gestión hospitalaria
- Implementar procedimientos almacenados que aseguren consistencia y seguridad en las operaciones.
- Facilitar el análisis de datos con la adición de la herramienta Power BI.

DIAGRAMA DE ENTIDAD RELACION - PHHOSPITAL



Modelado Físico de la Base de Datos

Propósito de la tabla: Esta tabla almacena la información personal y médica de cada paciente registrado en el hospital.

Tabla	Columna	Tipo de Dato	Propósito	Restricción	Llave
Pacientes	id	int	Vincular de forma única a cada paciente	NOT NULL	PK
	Nombres	varchar	Registrar el nombre de cada paciente	NOT NULL	
	Apellidos	varchar	Registrar el apellido de cada paciente.	NOT NULL	
	Sexo	varchar	Inidicar el sexo del paciente.	NOT NULL	
	edad	int	Registrar la edad del paciente.	NOT NULL	
	idTipodeSangre	int	Relacionar el tipo de sangre del paciente.	NOT NULL	FK a TipoDeSangre(idTipodeSangre)
	Celular	int	Registrar el número telefónico del paciente.	NULL	
	Dirección	varchar	Registrar la dirección completa del paciente.	NULL	
	DUI	int	Registrar el Documento Único de Identidad del paciente.	UNIQUE NOT NULL	
	idSeguro	int	Relacionar la aseguradora médica del paciente.	NOT NULL	

Propósito de la tabla: Almacena la información personal y profesional de los médicos que atienden en el hospital.

Tabla	Columna	Tipo de Dato	Propósito	Restricciones	Llave
Médicos	id	int	Vincular de forma única a cada médico.	NOT NULL	PK
	Nombres	varchar	Registrar el nombre del médico.	NOT NULL	
	Apellidos	varchar	Registrar el apellido del médico.	NOT NULL	
	Sexo	varchar	Indicar el sexo del médico.	NOT NULL	
	Edad	int	Registrar la edad del médico.	NOT NULL	
	idEspecialidad	int	Relacionar la especialidad médica del profesional.	NOT NULL	FK a Especializacion(id)
	celular	int	Registrar el número de contacto del médico.	NULL	

Propósito de la tabla: Registra cada consulta médica realizada, incluyendo paciente, médico, fecha y motivo.

Tabla	Columna	Tipo de Dato	Propósito	Restricciones	Llave
Consultas	id	int	Vincular cada consulta médica realizada.	NOT NULL	PK
	idPaciente	int	Relacionar el paciente atendido.	NOT NULL	FK a Pacientes(id)
	idMedico	int	Relacionar el médico que atendió la consulta.	NOT NULL	FK a Medicos(id)
	Fecha	datetime	Registrar la fecha de la consulta.	NOT NULL	
	Motivo	varchar	Reportar el motivo de la consulta.	NOT NULL	
	Estado	varchar	Indicar el estado de la consulta.	NOT NULL	

Propósito de la tabla: Catálogo de tratamientos médicos disponibles en el hospital.

Tabla	Columna	Tipo de Dato	Propósito	Restricciones	Llave
Tratamientos	id	int	Identificar cada tratamiento disponible.	NOT NULL	PK
	Tipo	varchar	Describir el tipo de tratamiento.	NOT NULL	
	Precio	int	Registrar el costo por unidad del tratamiento.	NOT NULL	

Propósito de la tabla: Registra la facturación de servicios médicos prestados a cada paciente.

Tabla	Columna	Tipo de Dato	Propósito	Restricciones	Llave
Factura	id	int	Identificar cada factura realizada.	NOT NULL	PK
	idPaciente	int	Relacionar el paciente que se ha hecho la factura.	NOT NULL	FK a Pacientes(id)
	Fecha	datetime	Registrar la fecha de emisión de la factura.	NOT NULL	
	Total	int	Registrar el monto total de la factura.	NOT NULL	
	Estado	varchar	Indicar el estado de la factura.	NOT NULL	

Propósito de la tabla: Catálogo de tipos sanguíneos disponibles para clasificar pacientes.

Tabla	Columna	Tipo de Dato	Propósito	Restricciones	Llave
TipoDeSangre	idTipoDeSangre	int	Es el identificador único del tipo de sangre.	NOT NULL	PK
	Tipo	varchar	Representa el grupo sanguíneo del paciente.	NOT NULL	

Propósito de la tabla: Catálogo de aseguradoras médicas disponibles en el hospital.

Tabla	Columna	Tipo de Dato	Propósito	Restricciones	Llave
Aseguradora	idSeguro	int	Es el identificador único de la aseguradora.	NOT NULL	PK
	Aseguradora	varchar	Nombre de la empresa aseguradora.	NOT NULL	
	Código	int	Código interno de la aseguradora	NOT NULL	

Propósito de la tabla: Catálogo de especialidades médicas disponibles en el hospital.

Tabla	Columna	Tipo de Dato	Propósito	Restricciones	Llave
Especialización	id	int	Es el identificador único de la especialidad médica.	NOT NULL	PK
	Especialidad	varchar	Nombre de la especialidad.	NOT NULL	

Propósito de la tabla: Registra los tratamientos aplicados durante cada consulta médica.

Tabla	Columna	Tipo de Dato	Propósito	Restricciones	Llave
TratamientoXConsulta	int	int	Es el identificador único de cada registro	NOT NULL	PK
	idTratamiento	int	Relaciona el tratamiento aplicado.	NOT NULL	FK a Tratamientos(id)
	idConsulta	int	Relaciona la consulta médica correspondiente.	NOT NULL	FK a Consultas(id)
	Cantidad	int	Número de	NOT NULL	

			veces que se aplicó el tratamiento en esa consulta		

Propósito de la tabla: Detalla los tratamientos incluidos en cada factura.

Tabla	Columna	Tipo de Dato	Propósito	Restricciones	Llave
FacturaDetalle	id	int	Es el identificador único del detalle de cada factura.	NOT NULL	PK
	idTratamiento	int	Relaciona el tratamiento con el cual se ha hecho la factura.	NOT NULL	FK a Tratamientos(id)
	idPaciente	int	Relaciona al paciente con el cual se le ha facturado el tratamiento.	NOT NULL	FK a Pacientes(id)
	SubTotal	int	Monto parcial correspondiente al tratamiento aplicado-	NOT NULL	

POLÍTICAS DE SEGURIDAD IMPLEMENTADAS

Cantidad de Objetos Definidos por Esquema

Esquema	Tabla	Vistas	Funciones	Procedimientos Almacenados	Triggers
dbo	0	0	2	0	0
Pacientes	3	1	0	2	4
Personal	2	1	0	0	0
Secretaria	2	6	3	2	0
Tesorería	3	2	2	4	1

En la base de datos PHHospital los objetos se encuentran distribuidos por esquemas de la siguiente manera:

- **Esquema Pacientes:** Contiene tres tablas principales (Pacientes, TipoDeSangre y Aseguradora). Además, cuenta con una vista asociada y dos procedimientos almacenados (SangreCompatible y SP_Buscar_Paciente_Por_DUI). Este esquema es el que concentra mayor número de triggers, con un total de cuatro, orientados a validar datos críticos como edad, DUI y celular, así como a evitar la eliminación de pacientes con consultas registradas.
- **Esquema Personal:** Posee dos tablas (Médicos y Especialización) y una vista relacionada con la gestión del personal médico. No posee funciones, procedimientos y triggers, lo que refleja su carácter más estático y de catálogo.
- **Esquema Secretaria:** está conformado por dos tablas (Consultas y TratamientoXConsulta), seis vistas que permiten generar reportes y seguimientos clínicos, tres funciones de apoyo para cálculos y resúmenes, y dos procedimientos almacenados. No tiene triggers definidos
- **Esquema Tesorería:** Concentra la lógica de facturación y pagos, con tres tablas (Factura, FacturaDetalle y Tratamientos), dos vistas de apoyo para reportes financieros, dos funciones (fn_TotalFactura y fn_DescuentoPorSeguro), y cuatro procedimientos almacenados (CrearFactura, CrearFacturaPorConsulta, Ver_Historial_de_Facturas y Ver_Ultima_Factura). Además, cuenta con un trigger para validar precios de tratamientos.
- **Esquema dbo:** no contiene tablas, pero sí una vista (CostoConsultaPorMedico) y una función (RankingAseguradoras()), que cumplen un rol transversal en el sistema.

Permisos por Rol sobre Tablas

Rol: rol_administrador

Tabla	Select	Insert	Update	Delete	Execute
Pacientes	✅	✅	✅	✅	✅
TipoDeSangre	✅	✅	✅	✅	✅
Aseguradora	✅	✅	✅	✅	✅
Médicos	✅	✅	✅	✅	✅
Especialización	✅	✅	✅	✅	✅
Consultas	✅	✅	✅	✅	✅
TratamientoX Consulta	✅	✅	✅	✅	✅
Factura	✅	✅	✅	✅	✅
Factura Detalle	✅	✅	✅	✅	✅
Tratamientos	✅	✅	✅	✅	✅

Rol: rol_alumno

Tabla	Select	Insert	Update	Delete	Execute
Pacientes	✅	✅	✅	✅	✅
TipoDeSangre	✅	✅	✅	✅	✅
Aseguradora	✅	✅	✅	✅	✅
Médicos	✅	✅	✅	✅	✅
Especialización	✅	✅	✅	✅	✅
Consultas	✅	✅	✅	✅	✅
TratamientoX Consulta	✅	✅	✅	✅	✅
Factura	✅	✅	✅	✅	✅
Factura Detalle	✅	✅	✅	✅	✅

Tratamientos	✅	✅	✅	✅	✅

Rol: rol_analista

Tabla	**Select**	**Insert**	**Update**	**Delete**	**Execute**
Pacientes	❌	❌	❌	❌	❌
TipoDeSangre	❌	❌	❌	❌	❌
Aseguradora	❌	❌	❌	❌	❌
Médicos	❌	❌	❌	❌	❌
Especialización	❌	❌	❌	❌	❌
Consultas	❌	❌	❌	❌	❌
TratamientoX Consulta	❌	❌	❌	❌	❌
Factura	❌	❌	❌	❌	❌
Factura Detalle	❌	❌	❌	❌	❌
Tratamientos	❌	❌	❌	❌	❌

Nota: El rol analista solo tiene SELECT sobre la vista Secretaria.Vista_PowerBI, por lo que no posee permisos sobre tablas.

Rol: rol_invitado

Tabla	**Select**	**Insert**	**Update**	**Delete**	**Execute**
Pacientes	❌	❌	❌	❌	❌
TipoDeSangre	❌	❌	❌	❌	❌
Aseguradora	❌	❌	❌	❌	❌
Médicos	❌	❌	❌	❌	❌
Especialización	❌	❌	❌	❌	❌
Consultas	❌	❌	❌	❌	❌
TratamientoX Consulta	❌	❌	❌	❌	❌

Factura	❌	❌	❌	❌	❌
Factura Detalle	❌	❌	❌	❌	❌
Tratamientos	❌	❌	❌	❌	❌

Rol: rol_profesor

Tabla	**Select**	**Insert**	**Update**	**Delete**	**Execute**
Pacientes	✅	❌	❌	❌	❌
TipoDeSangre	✅	❌	❌	❌	❌
Aseguradora	✅	❌	❌	❌	❌
Médicos	✅	❌	❌	❌	❌
Especialización	✅	❌	❌	❌	❌
Consultas	✅	❌	❌	❌	❌
TratamientoX Consulta	✅	❌	❌	❌	❌
Factura	✅	❌	❌	❌	❌
Factura Detalle	✅	❌	❌	❌	❌
Tratamientos	✅	❌	❌	❌	❌

Tabla de Usuarios, Roles y Privilegios

Usuarios	**Roles**	**Privilegios**
DBAdmin	rol_administrador	Control total sobre todos los esquemas y permisos de backup.
Alumno	rol_alumno	Control total sobre todos los esquemas y permisos de backup.
Profesor	rol_profesor	Lectura y visualización de definición en todos los esquemas.
Analista	rol_analista	Lectura sobre vista específica para PowerBI.
Invitado	rol_invitado	Lectura sobre vistas públicas del sistema.

Política de Login

Todos los logins creados en el sistema Phospital fueron configurados con políticas de seguridad activas:

- CHECK_POLICY = ON
- CHECK_EXPIRATION = ON

Esto garantiza que las contraseñas utilizadas por los usuarios cumplan con los estándares de complejidad y que sean renovadas periódicamente. Estas políticas refuerzan la seguridad del sistema, evitando accesos por contraseñas débiles o comprometidas. Además, permiten que el servidor SQL se integre con las políticas de seguridad del sistema operativo, en caso de estar conectado a un dominio.

EVIDENCIA DE CONSULTAS OPTIMIZADAS E ÍNDICES APLICADOS

Funciones Ventana Aplicadas en el Sistema:

1. **Función RankingAseguradoras**

```
CREATE FUNCTION RankingAseguradoras()
RETURNS TABLE
AS
RETURN
(
    WITH FrecuenciaAseguradoras AS (
        SELECT
            a.Aseguradora,
            COUNT(F.id) AS TotalFacturas
        FROM Pacientes.Aseguradora AS a
        INNER JOIN Pacientes.Pacientes AS p ON a.idSeguro = p.idSeguro
        INNER JOIN Tesoreria.Factura AS f ON p.id = f.idPaciente
        GROUP BY a.Aseguradora
    )

    SELECT
        Aseguradora,
        TotalFacturas,
        RANK() OVER (ORDER BY TotalFacturas DESC) AS Ranking
    FROM FrecuenciaAseguradoras
);
```

Esta función permite generar un ranking de aseguradoras médicas en función de la cantidad de facturas emitidas por cada una. Para ello, se realiza un conteo agrupado de facturas por aseguradora y se aplica la función ventana Rank() la cual ordena los resultados de forma descendente. Esto nos ayuda a identificar cuáles aseguradoras tienen mayor actividad en el hospital.

2. **Vista Costo_de_Consulta_por_Medico**

```
CREATE VIEW Costo_de_Consulta_por_Medico AS
SELECT
    c.id AS idConsulta,
    m.Nombres + ' ' + m.Apellidos AS Medico,
    p.Nobres + ' ' + p.Apellidos AS Paciente,
    SUM(t.Precio * txc.Cantidad) AS CostoConsultaIndividual,

    AVG(SUM(t.Precio * txc.Cantidad)) OVER (PARTITION BY c.idMedico) AS PromedioGastoMedico
FROM
    Consultas c
INNER JOIN
    TratamientoXConsulta txc ON c.id = txc.idConsulta
INNER JOIN
    Tratamientos t ON txc.idTratamiento = t.id
INNER JOIN
    Medicos m ON c.idMedico = m.id
INNER JOIN
    Pacientes p ON c.idPaciente = p.id
GROUP BY
    c.id, c.idMedico, m.Nombres, m.Apellidos, p.Nobres, p.Apellidos
```

Esta vista calcula el costo individual de cada consulta médica, sumando los precios de los tratamientos aplicados, y además determina el promedio de gasto por médico utilizando la función ventana AVG() OVER (PARTITION BY idMedico). Esta implementación permite obtener las estadísticas más importantes sobre el comportamiento económico de los médicos.

3. Vista - Vista_Seguimiento_Medico:

```
CREATE VIEW Secretaria.Vista_Seguimiento_Medico AS
SELECT
    p.Nobres + ' ' + p.Apellidos AS Paciente,
    c.Fecha AS FechaActual,
    c.Motivo,
    LAG(c.Fecha, 1) OVER (PARTITION BY c.idPaciente ORDER BY c.Fecha) AS FechaAnterior,
    DATEDIFF(DAY, LAG(c.Fecha, 1) OVER (PARTITION BY c.idPaciente ORDER BY c.Fecha), c.Fecha) AS DiasEntreSeguimiento,
    CASE
        WHEN DATEDIFF(DAY, LAG(c.Fecha, 1) OVER (PARTITION BY c.idPaciente ORDER BY c.Fecha), c.Fecha) > 180 THEN 'Seguimiento Tardio'
        WHEN DATEDIFF(DAY, LAG(c.Fecha, 1) OVER (PARTITION BY c.idPaciente ORDER BY c.Fecha), c.Fecha) > 90 THEN 'Seguimiento Regular'
        WHEN DATEDIFF(DAY, LAG(c.Fecha, 1) OVER (PARTITION BY c.idPaciente ORDER BY c.Fecha), c.Fecha) <= 90 THEN 'Seguimiento Frecuente'
        ELSE 'Primera Consulta'
    END AS TipoSeguimiento
FROM Secretaria.Consultas c
INNER JOIN Pacientes.Pacientes p ON c.idPaciente = p.id;
GO
```

Esta vista permite analizar el seguimiento clínico de los pacientes comparando la fecha actual de consulta con la anterior mediante la función ventana LAG(). Se calcula la diferencia en días entre ambas fechas y se clasifica el tipo de seguimiento como tardío, regular o frecuente. Esta funcionalidad es clave para detectar casos de abandono o seguimiento insuficiente, lo cual es util para priorizar la atención de los pacientes según su historial.

4. Vista - Vista_Historial_Paciente:

```
CREATE VIEW Secretaria.Vista_Historial_Paciente AS
SELECT
    ROW_NUMBER() OVER (PARTITION BY c.idPaciente ORDER BY c.Fecha) AS NumeroConsulta,
    p.Nobres + ' ' + p.Apellidos AS Paciente,
    p.DUI,
    m.Nombres + ' ' + m.Apellidos AS Medico,
    e.Especialidad,
    c.Fecha,
    c.Motivo,
    c.Estado
FROM Secretaria.Consultas c
INNER JOIN Pacientes.Pacientes p ON c.idPaciente = p.id
INNER JOIN Personal.Medicos m ON c.idMedico = m.id
INNER JOIN Personal.Especializacion e ON m.idEspecialidad = e.id;
GO
```

La vista asigna un número secuencial a cada consulta realizada por un paciente utilizando la función ventana ROW_NUMBER() OVER (PARTITION BY idPaciente ORDER BY Fecha). Esto permite visualizar el historial médico completo de cada paciente de forma ordenada, incluyendo el médico que lo atendió, la especialidad, el motivo y el estado de la consulta. Esta numeración facilita el seguimiento longitudinal y la revisión cronológica de la atención médica recibida.

Para mejorar el rendimiento de las consultas avanzadas que se implementaron en el sistema, se crearon índices no agrupados en las columnas que más se utilizaron en particiones, ordenamientos y relaciones entre tablas. Las funciones ventana que se aplicaron (RANK(), AVG() OVER, LAG(), ROW_NUMBER()) permiten obtener estadísticas importantes como el ranking de aseguradoras, el promedio de gasto por médico, el seguimiento entre consultas y el historial médico de cada paciente. Los índices se colocaron en campos como idPaciente, idMedico, Fecha, idConsulta, idSeguro y idEspecialidad, lo cual ayuda a que las vistas y funciones se ejecuten más rápido y con menos consumo de recursos. También se hicieron pruebas con SET STATISTICS IO y SET STATISTICS TIME para verificar el impacto de los índices, y se aplicaron los comandos de mantenimiento aprendidos a lo largo del ciclo, como REBUILD y REORGANIZE según el nivel de fragmentación. Finalmente, se eliminaron algunos índices que eran redundantes para evitar sobrecargar la base. Todo esto se hizo con el objetivo de que las consultas no solo funcionen bien, sino que estén optimizadas para un entorno real de gestión hospitalaria. Para ello se ha armado una tabla donde resume la labor hecha por los índices.

Esquema	**Tabla**	**Nombre del índice**	**Columnas Indexadas**	**Tipo de Índice**
Pacientes	Pacientes	IDX_Pacientes_DUI	DUI	UNIQUE NONCLUSTERED
Pacientes	Pacientes	IDX_Pacientes_Apellidos	Apellidos	NONCLUSTERED
Pacientes	Pacientes	IDX_Pacientes_TipoSangre	idTipoDe Sangre	NONCLUSTERED (eliminado)
Personal	Médicos	IDX_Medicos_Especialidad	idEspeciali dad	NONCLUSTERED
Secretaria	Consultas	IDX_Consultas_Fecha	Fecha	NONCLUSTERED
Secretaria	Consultas	IDX_Consultas_Estado	Estado	NONCLUSTERED

Secretaria	Consultas	IDX_Consultas_Paciente	idPaciente	NONCLUSTERED
Secretaria	Consultas	IDX_Consultas_Medico	idMedico	NONCLUSTERED
Secretaria	Consultas	IDX_Consultas_Paciente_Fecha	idPaciente, Fecha	NONCLUSTERED (eliminado)
Tesorería	Factura	IDX_Factura_Fecha	Fecha	NONCLUSTERED
Tesorería	Factura	IDX_Factura_Paciente	idPaciente	NONCLUSTERED
Tesorería	Factura	IDX_Factura_Estado	Estado	NONCLUSTERED
Secretaría	TratamientoXConsulta	IDX_TratamientoXConsulta_Consulta	idConsulta	NONCLUSTERED (eliminado)

Durante el proceso de optimización se crearon varios índices para mejorar el rendimiento de las consultas. Sin embargo, al analizar su uso y nivel de fragmentación, se identificaron algunos índices redundantes o de baja selectividad que no brindaban mejoras significativas. Por esta razón se decidió eliminarlos, con el objetivo de evitar tener más índices de los que necesitamos y mantener la base de datos más ligera y eficiente.

- El índice IDX_Pacientes_TipoSangre fue eliminado porque la columna idTipoDeSangre se utiliza principalmente en relaciones de clave foránea y no en filtros frecuentes.
- El índice IDX_Consultas_Paciente_Fecha fue eliminado ya que las columnas idPaciente y Fecha ya cuentan con índices individuales que cubren las mismas operaciones.
- El índice IDX_TratamientoXConsulta_Consulta fue eliminado debido a que la columna idConsulta no se emplea de forma recurrente en filtros directos, por lo que su impacto en rendimiento era mínimo.

ESTRATEGIA DE DIMENSIONAMIENTO, RESPALDO Y RECUPERACIÓN

Tabla	Componente	Tamaño (bytes)
Pacientes	id (INT)	4
	Nombres (Varchar(80))	80 + 2 = 82
	Apellidos(Varchar(80)	80 + 2 = 82
	Sexo(Varchar(1))	1 + 2 = 3
	Edad(INT)	4
	idTipoDeSangre(INT)	4
	Celular(INT)	4
	Direccion(Varchar(150))	150 + 2 = 152
	DUI(INT)	4
	idSeguro(INT)	4
	Cabecera + Null bitmap	10
	Total Estimado por fila	**≈411 bytes**

Tamaño Total (MB) = Total Estimado por fila * Cantidad de Registros / (1024*1024)

Tamaño Total (MB) = 411 * 1000 / (1024 * 1024) = 0.39 MB

Solamente en datos, sin contar índices y logs.

A la tabla Pacientes se le aplicaron dos índices así que:

Tamaño Total (MB) contando los índices = 0.39 * (1.3 * 2) = 1.014 MB

Tabla	Componente	Tamaño (bytes)
Médicos	id (INT)	4
	Nombres (Varchar(80))	80 + 2 = 82
	Apellidos (Varchar(80))	80 + 2 = 82
	Sexo (Varchar(1))	1 + 2 = 3
	edad (INT)	4
	idEspecialidad (INT)	4
	Celular (INT)	4
	Cabecera + Null bitmap	10
	Total Estimado por fila	**≈199 bytes**

Tamaño Total (MB) = Total Estimado por fila * Cantidad de Registros / (1024*1024)

Tamaño Total (MB) = 199 * 70 / (1024 * 1024) = 0.013 MB

Solamente en datos, sin contar índices y logs.

A la tabla Médicos se le aplicó un índices así que:

Tamaño Total (MB) contando los índices = 0.013 * (1.3) = 0.016 MB

Tabla	Componente	Tamaño (bytes)
Consultas	id (INT)	4
	idPaciente (INT)	4
	idMedico (INT)	4
	Fecha (DATETIME)	8
	Motivo (Varchar(100))	100 + 2 = 102
	Estado (Varchar(20))	20 + 2 = 22
	Cabecera + Null bitmap	10
	Total Estimado por fila	**≈154 bytes**

Tamaño Total (MB) = Total Estimado por fila * Cantidad de Registros / (1024*1024)

Tamaño Total (MB) = 154 * 8004 / (1024 * 1024) = 1.17 MB

Solamente en datos, sin contar índices y logs.

A la tabla Consultas se le aplicaron dos índices así que:

Tamaño Total (MB) contando los índices = 1.17 * (1.3 * 4) = 6.08 MB

Tabla	**Componente**	**Tamaño (bytes)**
Tratamientos	id (INT)	4
	Tipo (Varchar(20))	20 + 2 = 22
	Precio (INT)	4
	Cabecera + Null bitmap	10
	Total Estimado por Fila	**≈40 bytes**

Tamaño Total (MB) = Total Estimado por fila * Cantidad de Registros / (1024*1024)

Tamaño Total (MB) = 40 * 9855 / (1024 * 1024) = 0.37 MB

Solamente en datos, sin contar índices y logs.

A esta tabla no se le aplicó ningún índice.

Tabla	**Componente**	**Tamaño (bytes)**
Factura	id (INT)	4
	idPaciente (INT)	4
	Fecha (DATETIME)	8
	Total (INT)	4
	Estado (Varchar(20))	20 + 2 = 22
	Cabecera + Null bitmap	10
	Total Estimado por fila	**≈52 bytes**

Tamaño Total (MB) = Total Estimado por fila * Cantidad de Registros / (1024*1024)

Tamaño Total (MB) = 52 * 6350 / (1024 * 1024) = 0.314 MB

Solamente en datos, sin contar índices y logs.

A la tabla Factura se le aplicaron tres índices así que:

Tamaño Total (MB) contando los índices = 0.314 * (1.3 * 3) = 1.22 MB

Tabla	Componente	Tamaño (bytes)
FacturaDetalle	id (INT)	4
	idTratamiento (INT)	4
	idPaciente (INT)	4
	SubTotal (INT)	4
	Cabecera + Null bitmap	10
	Total Estimado por Fila	**≈26 bytes**

Tamaño Total (MB) = Total Estimado por fila * Cantidad de Registros / (1024*1024)

Tamaño Total (MB) = 26 * 7825 / (1024 * 1024) = 0.19 MB

Solamente en datos, sin contar índices y logs.

A esta tabla no se le aplicó ningún índice.

Tabla	Componente	Tamaño (bytes)
TratamientoXConsulta	id (INT)	4
	idTratamiento(INT)	4
	idConsulta(INT)	4
	Cantidad(INT)	4
	Cabecera + NULL bitmap	10
	Total Estimado por Fila	**≈26 bytes**

Tamaño Total (MB) = Total Estimado por fila * Cantidad de Registros / (1024*1024)

Tamaño Total (MB) = 26 * 9855 / (1024 * 1024) = 0.24 MB

Solamente en datos, sin contar índices y logs.

A esta tabla no se le aplicó ningún índice.

Tabla	Componente	Tamaño (bytes)
TipoDeSangre	idTipoDeSangre(INT)	4
	Tipo(Varchar(5))	5 + 2 = 7
	Cabecera + NULL bitmap	10
	Total Estimado por Fila	**≈21 bytes**

Tamaño Total (MB) = Total Estimado por fila * Cantidad de Registros / (1024*1024)

Tamaño Total (MB) = 21 * 8 / (1024 * 1024) = 0.00016 MB (menos de 1kb)

Solamente en datos, sin contar índices y logs.

A esta tabla no se le aplicó ningún índice.

Tabla	Componente	Tamaño (bytes)
Aseguradora	idSeguro (INT)	4
	Aseguradora (Varchar(30))	30 + 2 = 32
	codigo (INT)	4
	Cabecera + NULL bitmap	10
	Total Estimado por Fila	**≈50 bytes**

Tamaño Total (MB) = Total Estimado por fila * Cantidad de Registros / (1024*1024)

Tamaño Total (MB) = 50 * 6 / (1024 * 1024) = 0.00029 MB (menos de 1kb)

Solamente en datos, sin contar índices y logs.

A esta tabla no se le aplicó ningún índice.

Tabla	Componente	Tamaño(bytes)
Especialización	id (INT)	4
	Especialidad (Varchar(20))	20 + 2 = 22
	Cabecera + NULL bitmap	10
	Total Estimado por Fila	**≈36 bytes**

Tamaño Total (MB) = Total Estimado por fila * Cantidad de Registros / (1024*1024)

Tamaño Total (MB) = 36 * 9 / (1024 * 1024) = 0.00031 MB (menos de 1kb)

Solamente en datos, sin contar índices y logs.

A esta tabla no se le aplicó ningún índice.

DIMENSIONAMIENTO TOTAL DE LA BASE DE DATOS		
Tabla	**Tamaño en MB (sin contar índices)**	**Tamaño en MB (contando con los índices)**
Pacientes	0.39 MB	1.014 MB
Médicos	0.013 MB	0.016 MB
Consultas	1.17 MB	6.08 MB
Factura	0.314 MB	1.22 MB
TratamientoXConsulta	0.24 MB	0.24 MB
FacturaDetalle	0.19 MB	0.19 MB
TipoDeSangre	0.00016 MB	0.00016 MB
Aseguradora	0.00029 MB	0.00029 MB
Especialización	0.00031 MB	0.00031 MB
Tratamientos	0.37 MB	0.37 MB
Total:	**2.68 MB**	**9.13 MB**

Estrategia de Backup y Restauración Automatizada

Se definieron tres tipos de respaldo o Backups para la base de datos:

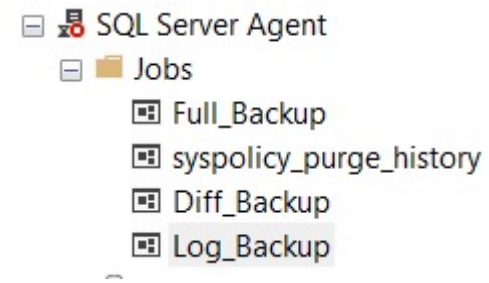
- FULL: respaldo completo semanal (jueves a las 2:00 AM)
- DIFFERENTIAL: respaldo diferencial cada 6 horas, todos los días
- LOG: respaldo de logs cada 1 hora y 30 minutos, desde la 1:30 AM hasta las 10:30 PM

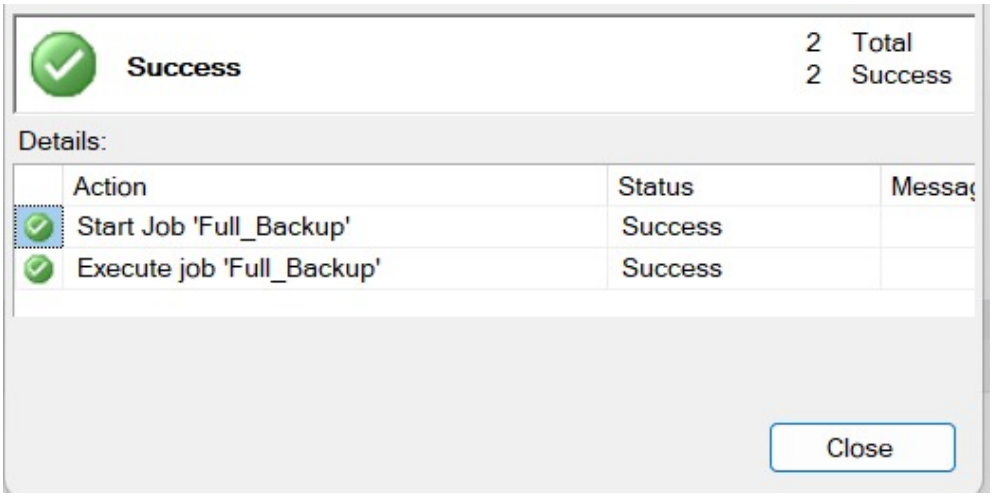
Esta estrategia permite mantener la base protegida ante fallas, con una pérdida mínima de datos y tiempos de recuperación aceptables.

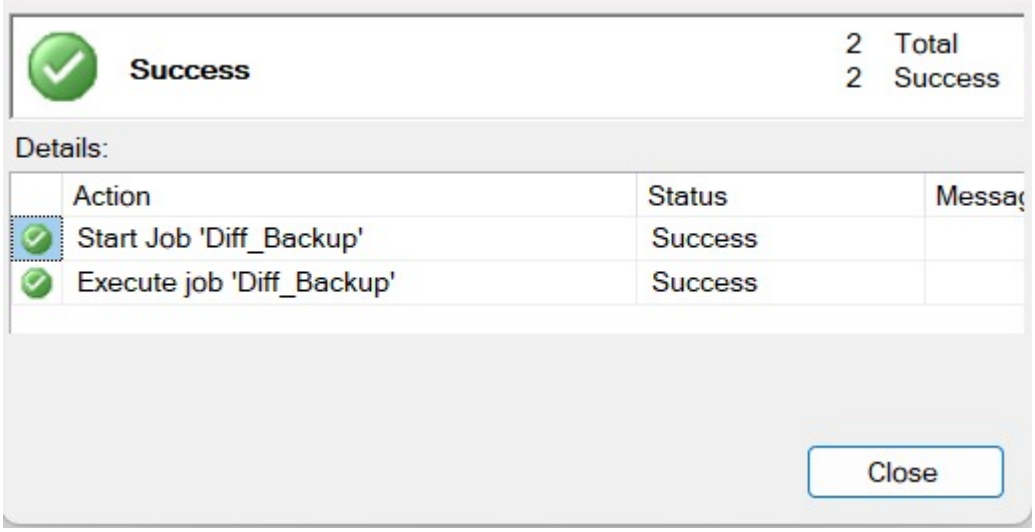
Posteriormente realizamos la creación de tres jobs:

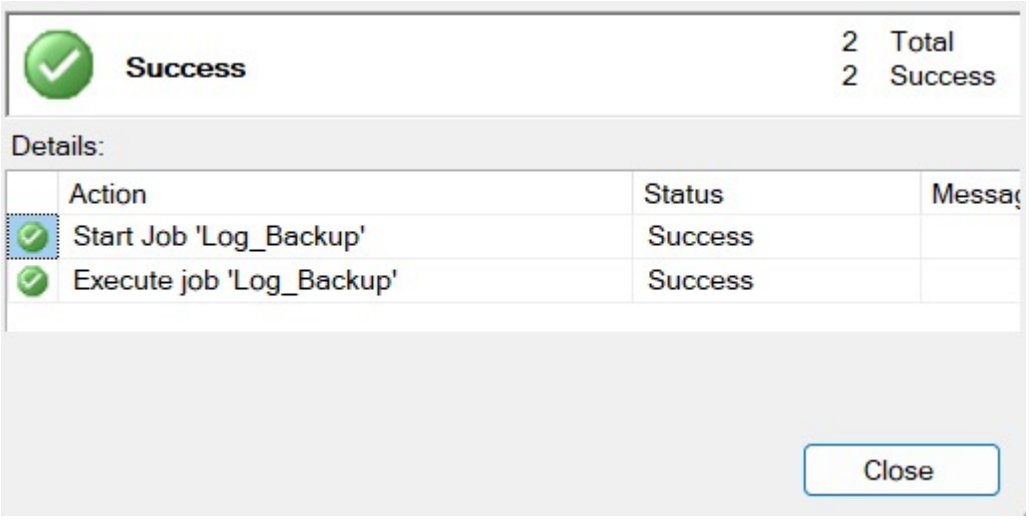
Job	**Tipo de Backup**	**Frecuencia**	**Hora**
Full_Backup	Full o Completo	Semanal (Jueves)	2:00 AM
Diff_Backup	Diferencial	Diario(cada 6 horas)	12:00 AM, 6:00 AM, 12:00 PM, 6:00 PM
Log_Backup	Log	Cada 1h 30 min	1:30 AM a 10:30PM

Capturas de su ejecución:









De igual forma se ha diseñado un calendario de Backups donde se muestra la ejecución de ellos a lo largo de la semana.

Calendario de Banckups							
Hora	Domingo	Lunes	Martes	Miércoles	Jueves	Viernes	Sábado
12:00 AM	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup
1:30 AM	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup
2:00 AM					**Full_Backup**		
3:00 AM	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup
4:30 AM	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup
6:00 AM	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup
7:30 AM	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup
9:00 AM	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup
10:30 AM	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup
12:00 PM	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup
1:30 PM	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup
3:00 PM	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup
4:30 PM	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup
6:00 PM	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup	Diff_Backup
7:30 PM	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup
9:00 PM	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup
10:30 PM	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup	Log_Backup

Simulación de Fallas y Análisis de RTO/RPO:

Para poder responder a la pregunta del RTO (Recovery Time Objective) se realizaron dos simulaciones de falla en la base de datos PHospital, generando dos escenarios distintos:

- RTO mínimo: La falla ocurre un jueves a las 2:30 AM, justo después de un backup completo. El procedimiento de restauración sería: Full_Backup + Tail-Log Backup = 2 pasos.
- RTO máximo: La falla ocurre un miércoles a las 11:00 AM, en medio de la jornada. El procedimiento de restauración sería: Full_Backup + Diff_Backup + Log_Backup + Log_Backup + Log_Backup + Tail-Log Backup = 6 pasos.

En cuanto al RPO (Recovery Point Objective), se estimó una pérdida máxima de información de 1 hora y 30 minutos. Por esta razón, se decidió programar el Backup de Log cada 1 hora y media, con el propósito de reducir al máximo el RPO. Esta decisión se tomó considerando que la base de datos contiene información crítica sobre pacientes, como expedientes, consultas, operaciones y tratamientos, datos que como hospital no se pueden permitir perder ni comprometer en términos de seguridad.