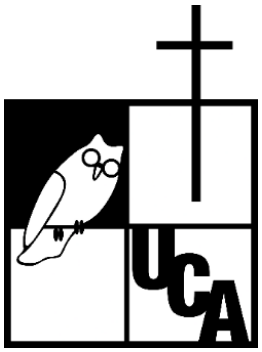
Universidad Centroamericana José Simeón Cañas



Administración de bases de datos

Informe de proyecto final

Integrantes:

Nombre	Carnet	Sección
Vásquez Centeno Alejandro José	00081320	01
Cornejo Sánchez Brandon Josué	00092322	01

26 de noviembre de 2025

PROYECTO FINAL DE ADMINISTRACION DE BASE DE DATOS

1. Descripción del Sistema

El sistema desarrollado corresponde a la gestión integral de un gimnasio, cuyo objetivo principal es administrar de forma eficiente la información relacionada con socios, entrenadores, clases, membresías, horarios y pagos. Este sistema busca centralizar los procesos operativos más importantes del negocio, garantizando control, organización y trazabilidad de todas las actividades que se realizan dentro del gimnasio.

El proyecto se basa en el diseño y construcción de una base de datos relacional implementada en SQL Server, estructurada bajo un enfoque modular utilizando un esquema lógico denominado gym, el cual permite agrupar todos los objetos relacionados al dominio del negocio. La base de datos fue diseñada siguiendo buenas prácticas de normalización, integridad referencial y optimización mediante índices estratégicos.

El sistema contempla las necesidades reales de un gimnasio moderno, incluyendo:

- **Gestión de Membresías**
  Se administran diferentes planes de membresía, cada uno con su precio, duración y descripción. Estos planes se asignan a los socios al momento de su registro y permiten controlar el estado y vigencia de cada miembro del gimnasio.

- **Control de Socios**
  El sistema almacena información general de los socios: datos personales, contacto, estado (activo/inactivo), fecha de alta y tipo de membresía asociada. Esto permite tener un registro histórico del flujo de clientes, facilitando el seguimiento y la administración de sus actividades dentro del gimnasio.

- **Administración de Entrenadores y Clases**
  El sistema gestiona a los entrenadores, sus especialidades y las clases que imparten. Cada clase tiene un nombre, un nivel y un cupo máximo, permitiendo organizar la oferta de actividades.

- **Programación de Horarios**
  Se registran los horarios de cada clase, con fecha y hora de inicio, duración, entrenador asignado y cupo disponible. Esto da estructura a la programación semanal del gimnasio y permite realizar reservas de manera ordenada.

- **Reservas de Socios**
  Los socios pueden reservar cupos en clases específicas mediante la tabla de reservas. El sistema previene conflictos mediante una restricción única que evita que un socio reserve dos veces la misma clase, garantizando integridad en el proceso de inscripción.

- **Registro de Pagos**
  Se controlan todos los pagos realizados por los socios, incluyendo monto, fecha, método de pago y concepto. Con esto se mantiene un historial financiero confiable que facilita auditorías y cálculos de ingresos.

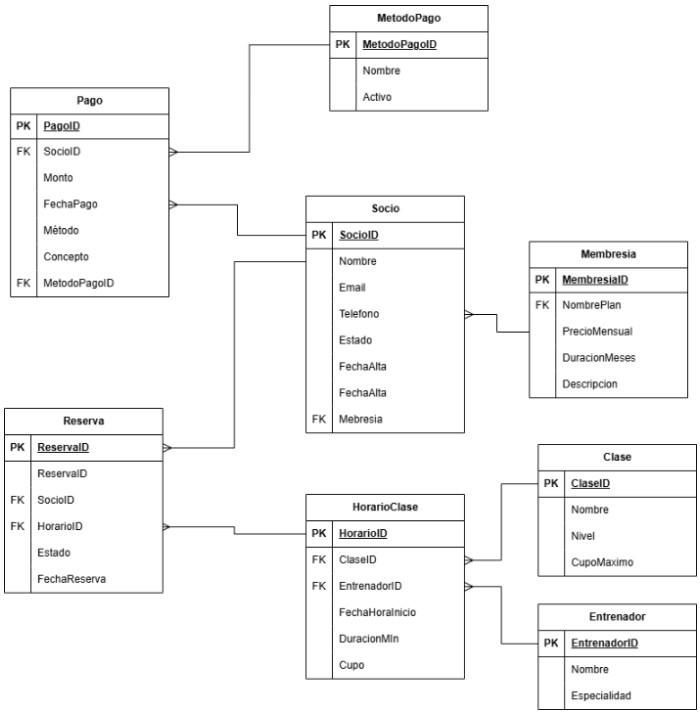
- **Integridad y Optimización**
  Para asegurar la consistencia del sistema, se implementaron diferentes claves foráneas que relacionan de forma correcta las entidades principales. Además, se crearon índices en columnas de uso frecuente, optimizando consultas críticas como reservas, pagos y horarios.

Objetivo del Sistema

El propósito del proyecto es proporcionar una solución robusta que permita gestionar de forma unificada todas las operaciones principales del gimnasio. La base de datos está diseñada para soportar consultas de análisis, dashboards de indicadores y futuras expansiones del sistema, garantizando flexibilidad, rendimiento y confiabilidad.

2. Diagrama relacional

El modelo entidad–relación del sistema de gestión del gimnasio fue diseñado bajo principios de normalización (3FN), garantizando consistencia, integridad referencial y mínima redundancia. El diagrama representa las entidades clave del negocio: socios, entrenadores, clases, horarios, membresías, pagos y reservas. Las relaciones entre entidades reflejan los procesos reales del gimnasio, como la inscripción de socios a clases programadas, la administración de planes de membresía y el registro de pagos. El modelo permite escalar el sistema, realizar análisis completos y asegurar un correcto funcionamiento operativo.



[**Relación**	**Cardinalidad**	**Descripción**
Membresia – Socio	**1 a N**	Una membresía puede estar asignada a muchos socios; cada socio solo tiene una membresía.
Socio – Reserva	**1 a N**	Un socio puede realizar múltiples reservas; cada reserva pertenece a un solo socio.
Socio – Pago	**1 a N**	Un socio puede tener varios pagos registrados; cada pago corresponde a un solo socio.
Clase – HorarioClase	**1 a N**	Una clase (ej. Yoga, Spinning) puede tener muchos horarios programados; cada horario pertenece a una sola clase.
Entrenador – HorarioClase	**1 a N**	Un entrenador puede impartir varios horarios; cada horario tiene un único entrenador asignado.
HorarioClase – Reserva	**1 a N**	Un horario puede tener varias reservas; cada reserva corresponde a un único horario.
MetodoPago – Pago	**1 a N**	Un método de pago (ej. Tarjeta, Efectivo) puede aparecer en múltiples pagos; cada pago tiene un solo método asignado.

Restricción	**Entidad**	**Descripción**
UNIQUE (SocioID, HorarioID)	Reserva	Evita que un socio reserve el mismo horario más de una vez.

3. Diccionario de datos

1. Tabla: gym.Membresia

Campo	**Tipo**	**PK/FK**	**Restricciones**	**Descripción**
MembresiaID	INT IDENTITY	PK	No nulo	Identificador único del plan de membresía.
NombrePlan	NVARCHAR(100)	—	NOT NULL	Nombre del plan de membresía.
PrecioMensual	DECIMAL(10,2)	—	NOT NULL	Precio mensual del plan.
DuracionMeses	INT	—	NOT NULL	Meses que dura la membresía (1, 3, 6, 12…).
Descripcion	NVARCHAR(250)	—	NULL	Detalle opcional del plan.

2. **Tabla gym.MetodoPago**

Campo	Tipo	PK/FK	Restricciones	Descripción
MetodoPagoID	INT IDENTITY	PK	No nulo	Clave primaria.
Nombre	NVARCHAR(50)	—	NOT NULL	Nombre del método (Efectivo, Tarjeta…).
Activo	BIT	—	DEFAULT 1	Indica si el método se encuentra disponible.

3. **Tabla gym.Socio**

Campo	Tipo	PK/FK	Restricciones	Descripción
SocioID	INT IDENTITY	PK	No nulo	Identificador único del socio.
Nombre	NVARCHAR(120)	—	NOT NULL	Nombre completo del socio.
Email	NVARCHAR(120)	—	NOT NULL, UNIQUE	Correo único del socio.
Telefono	NVARCHAR(30)	—	NULL	Teléfono de contacto.
Estado	TINYINT	—	DEFAULT 1	1=Activo, 0=Inactivo.
FechaAlta	DATETIME2	—	DEFAULT SYSDATETIME()	Fecha de registro.
MembresiaID	INT	FK	NULL	Relación con Membresia.

4. **Tabla gym.Entrenador**

Campo	Tipo	PK/FK	Restricciones	Descripción
EntrenadorID	INT IDENTITY	PK	No nulo	Identificador del entrenador.
Nombre	NVARCHAR(120)	—	NOT NULL	Nombre completo.
Especialidad	NVARCHAR(120)	—	NULL	Tipo de entrenamiento que imparte.

5. **Tabla gym.Clase**

Campo	Tipo	PK/FK	Restricciones	Descripción
ClaseID	INT IDENTITY	PK	No nulo	Identificador de clase.
Nombre	NVARCHAR(120)	—	NOT NULL	Nombre de la clase (Yoga, Zumba…).
Nivel	NVARCHAR(50)	—	NULL	Intensidad o nivel (Básico, Intermedio…).
CupoMaximo	INT	—	NOT NULL, CHECK > 0	Cantidad máxima de participantes.

6. Tabla gym.HorarioClase

Campo	Tipo	PK/FK	Restricciones	Descripción
HorarioID	INT IDENTITY	PK	No nulo	Identificador de horario.
ClaseID	INT	FK	NOT NULL	Relación con Clase.
EntrenadorID	INT	FK	NOT NULL	Relación con Entrenador.
FechaHoraInicio	DATETIME2	—	NOT NULL	Fecha y hora en que inicia la clase.
DuracionMin	INT	—	CHECK 15–240	Duración en minutos.
Cupo	INT	—	CHECK > 0	Cupos disponibles en este horario.

7. Tabla gym.Reserva

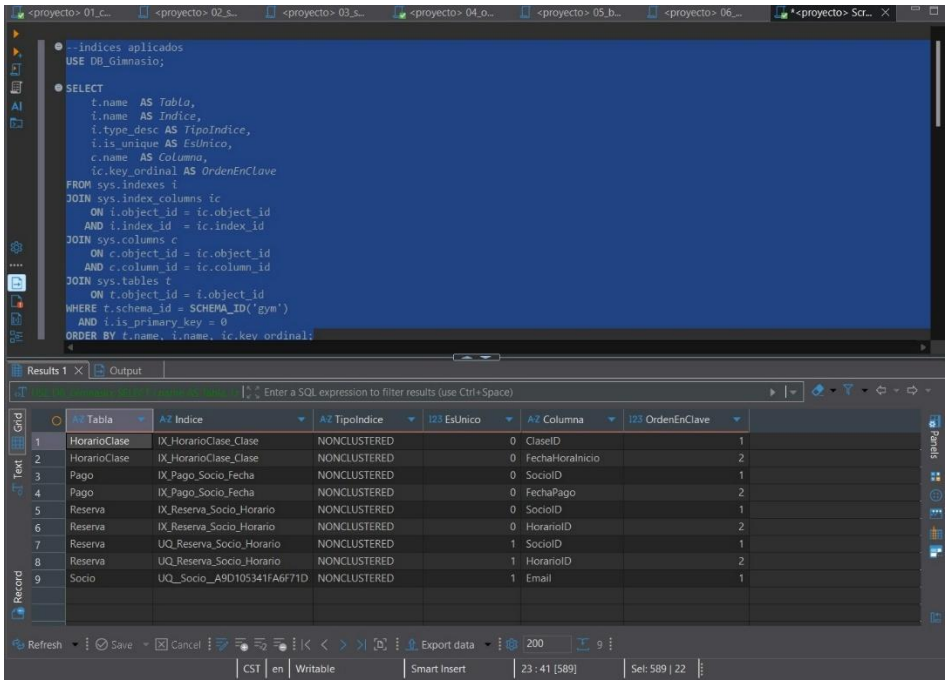
Campo	Tipo	PK/FK	Restricciones	Descripción
ReservaID	INT IDENTITY	PK	No nulo	Identificador de reserva.
SocioID	INT	FK	NOT NULL	Socio que realiza la reserva.
HorarioID	INT	FK	NOT NULL	Horario reservado.
Estado	NVARCHAR(20)	—	DEFAULT 'Confirmada'	Estado de la reserva.
FechaReserva	DATETIME2	—	DEFAULT SYSDATETIME()	Fecha en que se registró la reserva.

8. Tabla gym.Pago

Campo	Tipo	PK/FK	Restricciones	Descripción
PagoID	INT IDENTITY	PK	No nulo	Identificador del pago.
SocioID	INT	FK	NOT NULL	Socio que realiza el pago.
Monto	DECIMAL(12,2)	—	CHECK > 0	Monto pagado.
FechaPago	DATE	—	NOT NULL	Fecha del pago.
Metodo	NVARCHAR(30)	—	NOT NULL	Método escrito (texto).
Concepto	NVARCHAR(120)	—	NULL	Detalle del pago (mensualidad, inscripción…).
MetodoPagoID	INT	FK	NULL	Relación con MetodoPago.

4. Consultas optimizadas e índices aplicados

Consulta con índices aplicados



```sql
--indices aplicados
USE DB_Gimnasio;

SELECT
    t.name  AS Tabla,
    i.name  AS Indice,
    i.type_desc AS TipoIndice,
    i.is_unique AS EsUnico,
    c.name  AS Columna,
    ic.key_ordinal AS OrdenEnClave
FROM sys.indexes i
JOIN sys.index_columns ic
    ON i.object_id = ic.object_id
   AND i.index_id  = ic.index_id
JOIN sys.columns c
    ON c.object_id = ic.object_id
   AND c.column_id = ic.column_id
JOIN sys.tables t
    ON t.object_id = i.object_id
WHERE t.schema_id = SCHEMA_ID('gym')
  AND i.is_primary_key = 0
ORDER BY t.name, i.name, ic.key_ordinal;
```

Results 1 | Output

Enter a SQL expression to filter results (use Ctrl+Space)

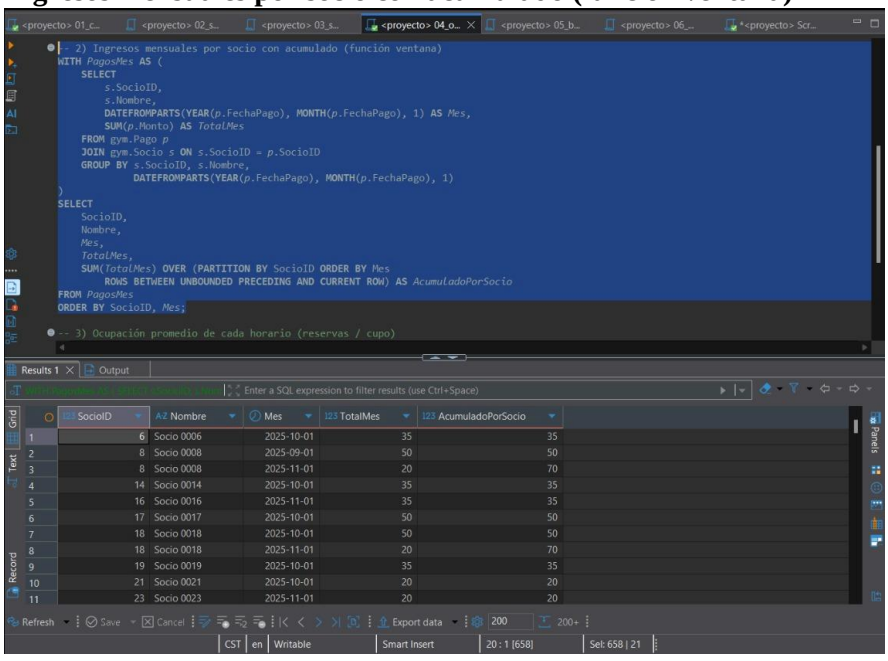
	Tabla	Indice	TipoIndice	EsUnico	Columna	OrdenEnClave
1	HorarioClase	IX_HorarioClase_Clase	NONCLUSTERED	0	ClaseID	1
2	HorarioClase	IX_HorarioClase_Clase	NONCLUSTERED	0	FechaHoraInicio	2
3	Pago	IX_Pago_Socio_Fecha	NONCLUSTERED	0	SocioID	1
4	Pago	IX_Pago_Socio_Fecha	NONCLUSTERED	0	FechaPago	2
5	Reserva	IX_Reserva_Socio_Horario	NONCLUSTERED	0	SocioID	1
6	Reserva	IX_Reserva_Socio_Horario	NONCLUSTERED	0	HorarioID	2
7	Reserva	UQ_Reserva_Socio_Horario	NONCLUSTERED	1	SocioID	1
8	Reserva	UQ_Reserva_Socio_Horario	NONCLUSTERED	1	HorarioID	2
9	Socio	UQ__Socio__A9D105341FA6F71D	NONCLUSTERED	1	Email	1

Refresh | Save | Cancel | Export data | 200 | 9

CST | en | Writable | Smart Insert | 23 : 41 [589] | Sel: 589 | 22

Consultas optimizadas

1. Ingresos mensuales por socio con acumulado (función ventana)



```sql
-- 2) Ingresos mensuales por socio con acumulado (función ventana)
WITH PagosMes AS (
    SELECT
        s.SocioID,
        s.Nombre,
        DATEFROMPARTS(YEAR(p.FechaPago), MONTH(p.FechaPago), 1) AS Mes,
        SUM(p.Monto) AS TotalMes
    FROM gym.Pago p
    JOIN gym.Socio s ON s.SocioID = p.SocioID
    GROUP BY s.SocioID, s.Nombre,
             DATEFROMPARTS(YEAR(p.FechaPago), MONTH(p.FechaPago), 1)
)
SELECT
    SocioID,
    Nombre,
    Mes,
    TotalMes,
    SUM(TotalMes) OVER (PARTITION BY SocioID ORDER BY Mes
        ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS AcumuladoPorSocio
FROM PagosMes
ORDER BY SocioID, Mes;

-- 3) Ocupación promedio de cada horario (reservas / cupo)
```

Results 1 | Output

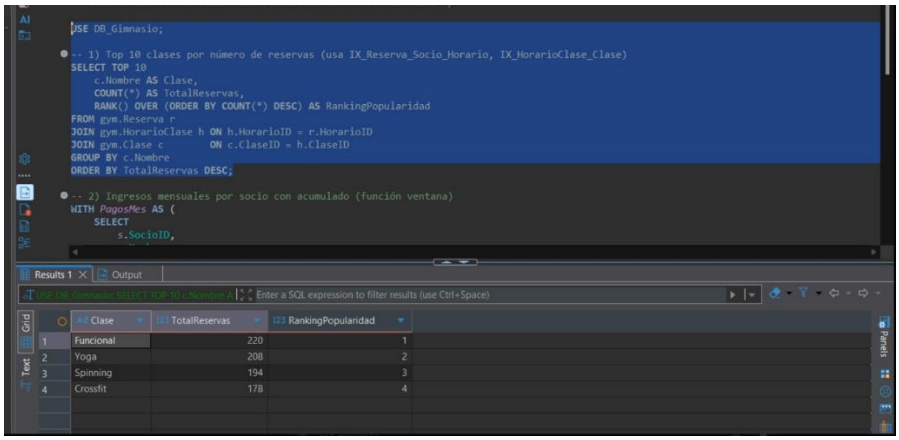
Enter a SQL expression to filter results (use Ctrl+Space)

	SocioID	Nombre	Mes	TotalMes	AcumuladoPorSocio
1	6	Socio 0006	2025-10-01	35	35
2	8	Socio 0008	2025-09-01	50	50
3	8	Socio 0008	2025-11-01	20	70
4	14	Socio 0014	2025-10-01	35	35
5	16	Socio 0016	2025-11-01	35	35
6	17	Socio 0017	2025-10-01	50	50
7	18	Socio 0018	2025-10-01	50	50
8	18	Socio 0018	2025-11-01	20	70
9	19	Socio 0019	2025-10-01	35	35
10	21	Socio 0021	2025-10-01	20	20
11	23	Socio 0023	2025-11-01	20	20

Refresh | Save | Cancel | Export data | 200 | 200+

CST | en | Writable | Smart Insert | 20 : 1 [658] | Sel: 658 | 21

2. Top 10 clases por número de reservas



5. Politicas de seguridad implementadas

La base de datos del proyecto implementa un conjunto de políticas de seguridad enfocadas en el control de acceso, la separación de funciones y la asignación mínima de privilegios necesarios (principio Least Privilege). Estas políticas se materializan mediante la creación de logins, usuarios, roles y permisos a nivel de servidor y base de datos.

1. Control de acceso a nivel de servidor

Se definieron tres logins en SQL Server, cada uno con un propósito específico dentro del sistema:

gym_admin

Usuario con permisos administrativos para tareas de mantenimiento y gestión avanzada.

gym_app_rw

Usuario utilizado por la aplicación para operaciones de lectura y escritura.

gym_app_ro

Usuario destinado a consultas y reportes con permisos únicamente de lectura.

A todos los logins se les desactivó CHECK_POLICY y CHECK_EXPIRATION para evitar bloqueo por políticas locales durante pruebas y desarrollo.

2. Mapeo de logins a usuarios de la base de datos

En la base de datos DB_Gimnasio, los logins anteriores se mapean a usuarios internos del mismo nombre:

CREATE USER gym_admin FOR LOGIN gym_admin;

CREATE USER gym_app_rw FOR LOGIN gym_app_rw;

CREATE USER gym_app_ro FOR LOGIN gym_app_ro;

Esto permite que cada usuario tenga permisos diferenciados dentro del esquema gym.

3. Definición de roles de seguridad

Se aplicó el principio de separación de funciones mediante la creación de tres roles internos:

- *rol_admin*

  Rol con control total sobre el esquema. Capaz de crear, modificar y eliminar objetos.

- *rol_operativo*

  Rol diseñado para la aplicación en producción. Permite realizar operaciones CRUD completas (SELECT, INSERT, UPDATE, DELETE).

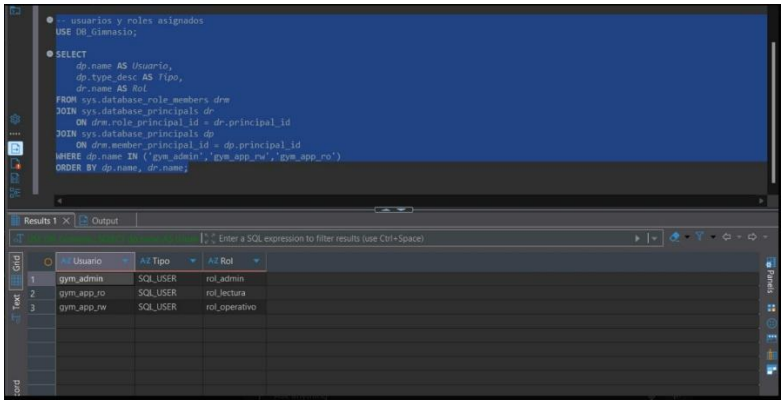
- *rol_lectura*

  Rol restringido a consultas únicamente, ideal para reportes, dashboards o monitoreo.

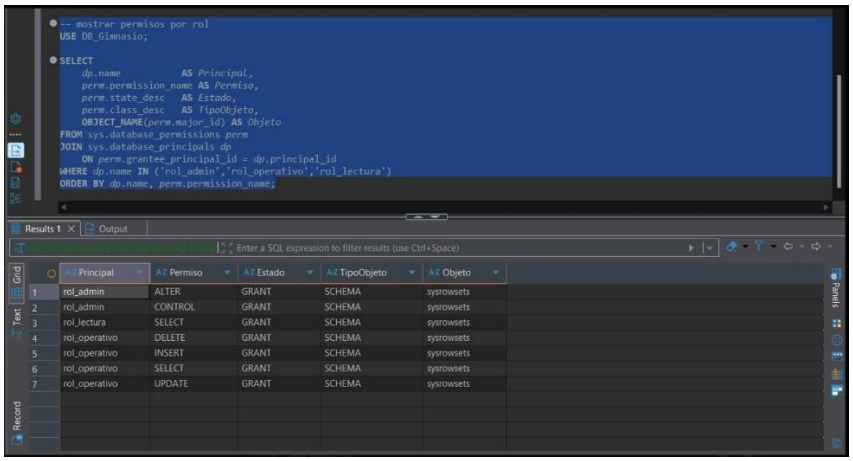
Cada usuario fue asignado a su rol correspondiente mediante el procedimiento *sp_addrolemember*.

Evidencias de los roles

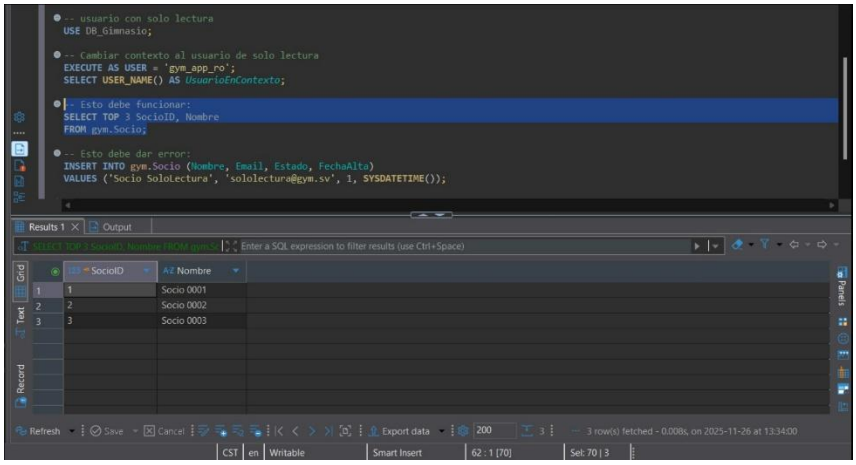
- Usuarios y roles



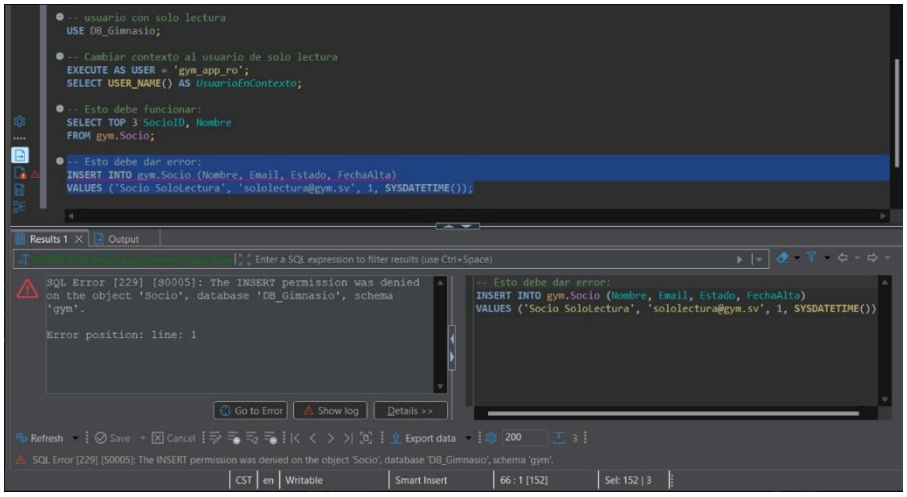
- Permisos de los roles



- Ejemplo de rol con solo lectura



- Comprobando que el usuario lector no puede hacer un insert



6. Estrategia de dimensionamiento, Respaldo y Recuperación

1. Dimensionamiento inicial

Se definieron tamaños iniciales y crecimientos controlados para los archivos de datos y log, con el objetivo de evitar fragmentación y asegurar un rendimiento consistente:

- Archivo de datos: tamaño inicial de 100 MB, crecimiento de 50 MB.
- Archivo de log: tamaño inicial de 50 MB, crecimiento de 25 MB.

Esta configuración permite que la base de datos crezca de manera ordenada sin depender de incrementos muy pequeños que afecten el rendimiento.

2. Estrategia de Respaldo

Se implementó un backup completo periódico de la base de datos, utilizando compresión para optimizar espacio y tiempos de procesamiento.

El respaldo se genera mediante:

- Backup completo almacenado en la ruta estándar del servidor (*/var/opt/mssql/backups/*).
- Uso de parámetros INIT y COMPRESSION para sobrescribir de forma controlada y reducir espacio utilizado.
- Ejecución programable bajo el motor SQL Server.

Este enfoque garantiza la existencia de una copia íntegra de toda la información crítica del gimnasio.

Para asegurar la validez del archivo .bak generado, se ejecuta:

- RESTORE VERIFYONLY: Valida la estructura y consistencia del respaldo sin restaurar la base de datos.

Con esto se confirma que el archivo de respaldo es funcional y puede ser usado en un escenario real de recuperación.

3. Procedimiento de Recuperación

Se definió un procedimiento estándar para restaurar la información en una base de datos paralela denominada DB_Gimnasio_Test, permitiendo:

- Probar la integridad del respaldo.
- Validar el proceso de restauración.

- Contar con un ambiente seguro para pruebas y verificaciones sin afectar la base de datos principal.

La restauración utiliza los comandos RESTORE DATABASE con rutas específicas para los archivos .mdf y .ldf, además de la opción REPLACE para sobrescribir la versión anterior del ambiente de pruebas.