

**Universidad Peruana Los Andes**

**Facultad de Ingeniería**

**Escuela profesional de Ingeniería de Sistemas**



**Curso: Base de datos II**

**Docente: Raul Enrique Fernandez Bejarano**

**Estudiante: Sarmiento Mosquera Yeims Abraham**

**Ciclo: V - Código: s03807f**

**Huancayo - 2025**

### Semana 10: Administración esencial

**Objetivo:** Dominar la administración diaria del servidor y la gestión de bases de datos.

#### Temas:

1. Estructura de almacenamiento y archivos de datos.
2. Propiedades y configuraciones de bases de datos.
3. Tipos de recuperación (Simple, Full, Bulk-Logged).
4. Administración de usuarios y roles de seguridad.
5. Asignación de permisos y políticas de acceso.
6. Monitoreo básico con el **SQL Server Activity Monitor**.
7. Introducción al uso de **SQL Server Agent** (tareas automáticas).

#### Prácticas:

- Cambiar la configuración del modelo de recuperación.
- Crear un rol y asignar permisos a un grupo de usuarios.
- Programar un *backup automático diario* con SQL Server Agent.

#### Herramientas:

- SSMS (Object Explorer y Security)
- T-SQL (CREATE LOGIN, ALTER DATABASE, BACKUP DATABASE)

SQL Server almacena datos en archivos físicos:

- Archivos de datos primarios (.mdf)
- Archivos de datos secundarios (.ndf)
- Registro de transacciones (.ldf). El motor gestiona páginas (8 KB) y extensiones (64 KB) para organización física.

Diseño de archivos influye en rendimiento: separar archivos de datos y logs en discos distintos, usar múltiples archivos de datos para cargas altamente concurrentes, y ajustar autogrowth en tamaños razonables (no en % en sistemas OLTP).

Mantenimiento: operaciones de shrink solo cuando es estrictamente necesario; verificar y mantener estadísticas e índices; planificar backups coherentes con el modelo de recuperación.

### EJEMPLO:

Crearemos una base de datos con archivos separados

```
CREATE DATABASE EscuelaLab
ON
( NAME = N'EscuelaLab_Data', FILENAME = N'C:\SQLData\EscuelaLab_Data.mdf', SIZE = 100MB, FILEGROWTH = 50MB ),
( NAME = N'EscuelaLab_Data2', FILENAME = N'C:\SQLData\EscuelaLab_Data2.ndf', SIZE = 50MB, FILEGROWTH = 50MB )
LOG ON
( NAME = N'EscuelaLab_Log', FILENAME = N'C:\SQLLogs\EscuelaLab_Log.ldf', SIZE = 50MB, FILEGROWTH = 20MB );
GO
```

Logical Name	File Type	Filegroup	Size (MB)	Autogrowth / Maxsize	Path	File Name
EscuelaLab...	ROWS...	PRIMARY	100	By 50 MB, Unlimited	C:\SQLData	EscuelaLab_Data.mdf
EscuelaLab...	ROWS...	PRIMARY	50	By 50 MB, Unlimited	C:\SQLData	EscuelaLab_Data2.ndf
EscuelaLab...	LOG	Not Applicable	50	By 20 MB, Limited to 2...	C:\SQLLogs	EscuelaLab_Log.ldf

Propiedades clave de una base de datos: tamaño inicial, crecimiento automático (autogrowth), collation, modelo de recuperación, opciones de compatibilidad (compatibility level), opciones de auto shrink/auto close, y ubicación de archivos.

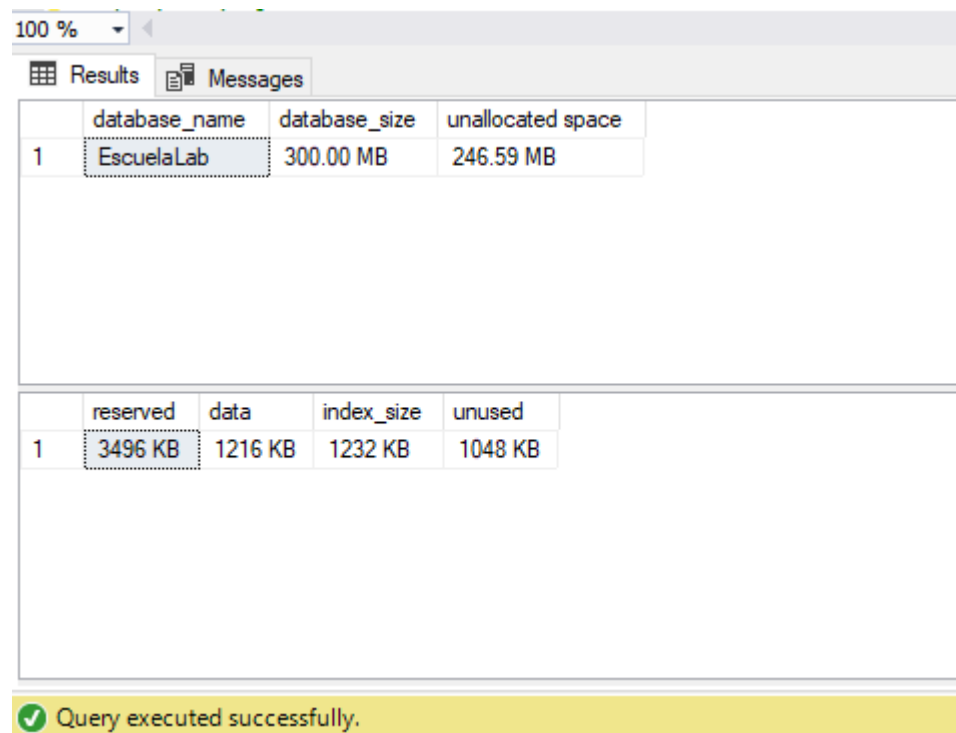
EJEMPLO:

Modificaremos las propiedades y luego comprobaremos el espacio de uso

```
ALTER DATABASE EscuelaLab
MODIFY FILE ( NAME = 'EscuelaLab_Data', SIZE = 200MB, FILEGROWTH = 100MB );
GO

USE EscuelaLab;
EXEC sp_spaceused;
GO
```

EJECUTANDO EL CODIGO:



	database_name	database_size	unallocated space
1	EscuelaLab	300.00 MB	246.59 MB

	reserved	data	index_size	unused
1	3496 KB	1216 KB	1232 KB	1048 KB

Query executed successfully.

Existen 3 tipos de recuperación o Backups:

1. Simple: no preserva el log para recuperaciones punto-en-tiempo; el log se trunca automáticamente. Adecuado para datos regenerables o entornos de desarrollo.
2. Full: registra todas las transacciones; permite recuperación punto-en-tiempo con backups de log; requerido para RPO bajos. Necesita backups regulares de log o crecerá el .ldf.
3. Bulk-Logged: modo intermedio que minimiza tamaño del log durante operaciones bulk (bcp, BULK INSERT, SELECT INTO), sacrificando la capacidad de recuperación punto-en-tiempo para sesiones con operaciones masivas; útil para cargas masivas controladas.

EJEMPLO:

```
--ejemplo 3
--backup actual
SELECT name, recovery_model_desc FROM sys.databases WHERE name = 'EscuelaLab';
GO

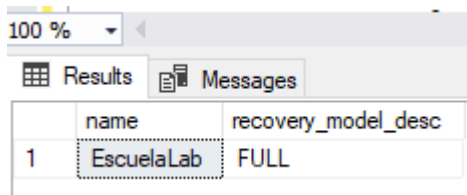
-- backup full
ALTER DATABASE EscuelaLab SET RECOVERY FULL;
GO
BACKUP DATABASE EscuelaLab TO DISK = N'C:\Backups\EscuelaLab_FULL.bak' WITH INIT;
GO

--backup de log
BACKUP LOG EscuelaLab TO DISK = N'C:\Backups\EscuelaLab_Log.trn' WITH INIT;
GO

--backup simple
ALTER DATABASE EscuelaLab SET RECOVERY SIMPLE;
GO
```

RESULTADOS:

Backup actual:



	name	recovery_model_desc
1	EscuelaLab	FULL

La seguridad en SQL Server se compone de logins a nivel de instancia y usuarios a nivel de base de datos. Los logins pueden ser SQL Server o Windows (AD).

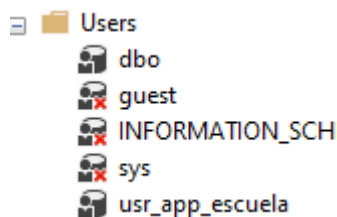
Roles fijos del servidor (sysadmin, serveradmin, etc.) y roles fijos de base de datos (db\_owner, db\_datareader, db\_datawriter, db\_ddladmin) facilitan delegación. También existen roles definidos por el usuario y permisos granulares (SCHEMA, OBJECT, COLUMN, EXECUTE).

## EJEMPLO

```
-- EJEMPLO 4
-- Crear login a nivel de instancia (SQL auth, ejemplo de laboratorio)
CREATE LOGIN app_escuela WITH PASSWORD = 'P@ssw0rdSegura!';
GO

-- Crear usuario en la base de datos y asignarlo a rol personalizado
USE EscuelaLab;
CREATE USER usr_app_escuela FOR LOGIN app_escuela;
GO
```

## RESULTADO DE LOS USUARIOS CREADOS



Principio de menor privilegio: asignar sólo los permisos necesarios. Usar roles de aplicación o grupos AD para administrar conjuntos de permisos. Emplear esquemas para organización y aplicar permisos a nivel de esquema cuando sea apropiado.

Políticas de acceso: en entornos corporativos, integrar con Active Directory, habilitar autenticación y cifrado (TLS), usar Always Encrypted o Dynamic Data Masking para datos sensibles, y auditar accesos.

## EJEMPLO

Para el ejemplo seguiremos usando los datos anteriores

```
-- Asignar permisos mínimos: sólo insertar/select en esquema dbo.Alumnos
CREATE ROLE rol_app_lectura_insercion;
ALTER ROLE rol_app_lectura_insercion ADD MEMBER usr_app_escuela;
GO

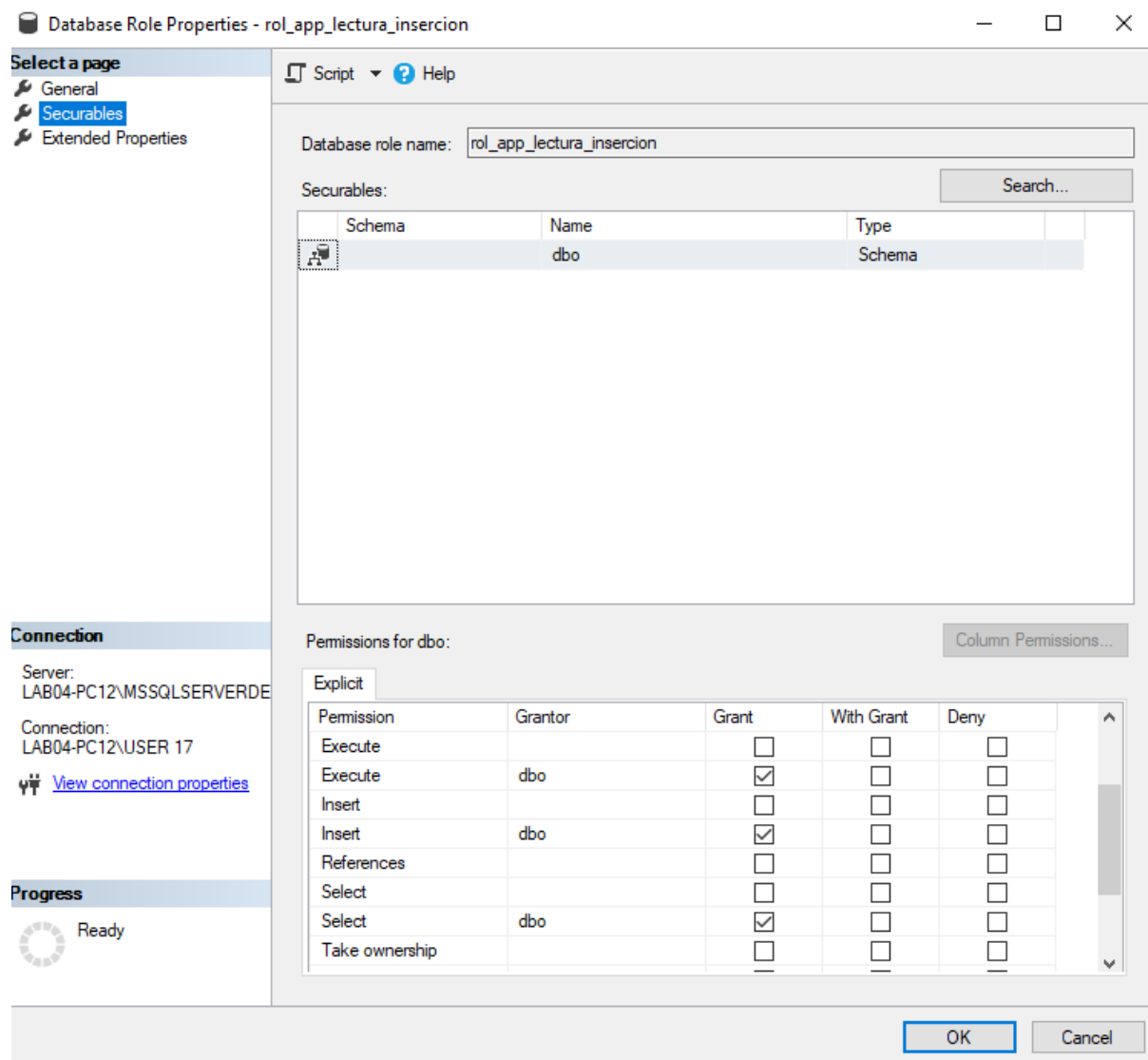
GRANT SELECT, INSERT ON SCHEMA::dbo TO rol_app_lectura_insercion;
GO

--conceder permiso execute
GRANT EXECUTE ON SCHEMA::dbo TO rol_app_lectura_insercion;
GO

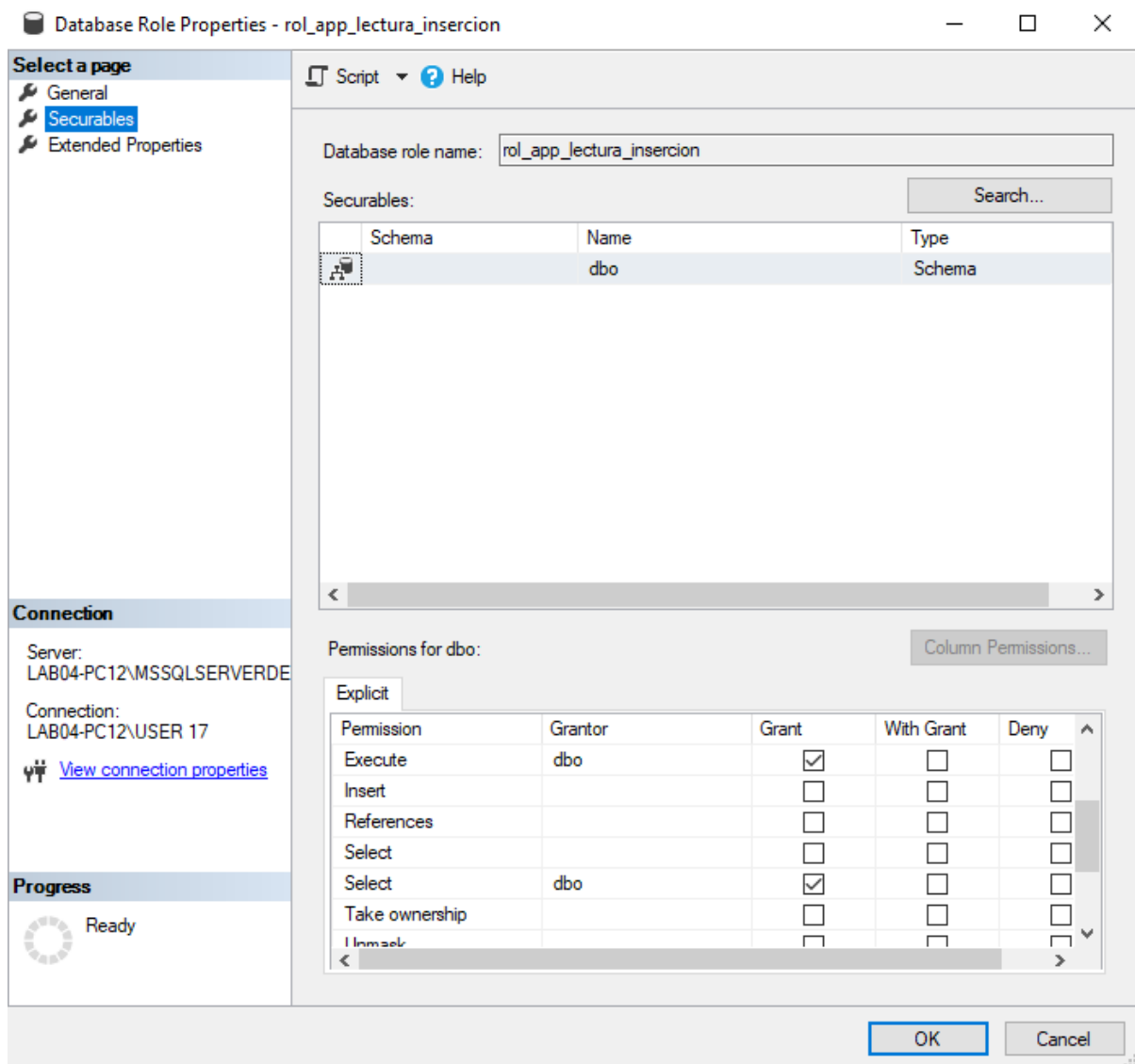
--quitar permiso
REVOKE INSERT ON SCHEMA::dbo FROM rol_app_lectura_insercion;
GO

-- ver permisos de un usuario
SELECT * FROM fn_my_permissions(NULL, 'DATABASE');
-- 0 revisar vistas de catalogo: sys.database_permissions, sys.database_principals
GO
```

## RESULTADO DE LA ASIGNACION DE PERMISOS



## RESULTADO DE REVOCAR LOS PERMISOS



Activity Monitor en SSMS ofrece una vista rápida de la actividad: procesos activos, CPU, I/O, bloqueos, consultas más costosas y espera de recursos. Es útil para diagnóstico inicial y para identificar sesiones bloqueadas o queries de larga duración.

Limitaciones: Activity Monitor es una herramienta GUI de diagnóstico puntual; para monitorización continua y retención histórica conviene usar Query Store, Extended Events, Performance Monitor (PerfMon) o soluciones APM/Monitoring (Azure Monitor, SCOM, etc.).

Métricas clave: CPU, Page Life Expectancy, waits (LCK\_M\_X, PAGEIOLATCH\_XX), Batch Requests/sec, Log Flush Waits, y latencia de I/O.

EJEMPLOS:

```

--ejemplo 6

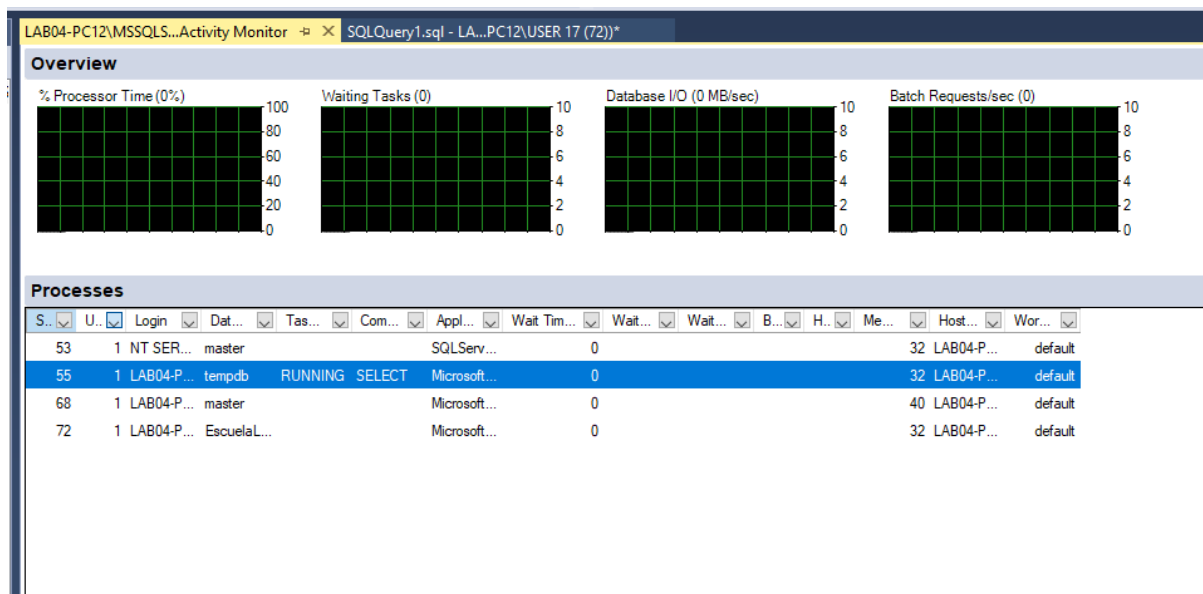
-- Ver detalles de sesión
DBCC INPUTBUFFER(<SPID>);
SELECT * FROM sys.dm_exec_sessions WHERE session_id = <SPID>;
SELECT * FROM sys.dm_exec_requests WHERE session_id = <SPID>;
GO

-- usar query store
-- Habilitar Query Store en la base de datos (si no está)
ALTER DATABASE EscuelaLab SET QUERY_STORE = ON;
GO

-- Consultar top 10 queries por tiempo total
SELECT TOP 10 qsqt.query_id, qsqt.query_sql_text, qsp.avg_duration
FROM sys.query_store_query_text qsqt
JOIN sys.query_store_query qsq ON qsqt.query_text_id = qsq.query_text_id
JOIN sys.query_store_plan qsp ON qsqt.query_id = qsp.query_id
ORDER BY qsp.avg_duration DESC;
GO

```

## RESULTADO



SQL Server Agent es el componente para automatización: jobs (con pasos que ejecutan T-SQL, SSIS, PowerShell), schedules, alerts y operators. Permite programar backups, mantenimiento de índices, cargas ETL y notificaciones.



Estructura: Job → Steps (cada uno con su propio tipo y subsistema) → Schedules → Alerts → Notifications. Jobs pueden ejecutarse bajo cuentas con permisos específicos (proxy) para limitar privilegios.

Buenas prácticas: separar jobs de mantenimiento de jobs de negocio; registrar salida (logging) y alertas; evitar jobs largos en horarios pico; usar retención de historial para no llenar msdb.

## EJEMPLO:

```
--ejemplo 7
USE msdb;
GO
EXEC sp_add_job @job_name = N'Backup_EscuelaLab_Diario';
GO

EXEC sp_add_jobstep
    @job_name = N'Backup_EscuelaLab_Diario',
    @step_name = N'Backup Completo',
    @subsystem = N'TSQL',
    @command = N'BACKUP DATABASE EscuelaLab TO DISK = N'C:\Backups\EscuelaLab_FULL.bak' WITH INIT, COMPRESSION;',
    @retry_attempts = 1,
    @retry_interval = 5;
GO

EXEC sp_add_schedule
    @schedule_name = N'Diario_02AM',
    @freq_type = 4, -- diario
    @freq_interval = 1,
    @active_start_time = 20000; -- 02:00:00
GO

EXEC sp_attach_schedule
    @job_name = N'Backup_EscuelaLab_Diario',
    @schedule_name = N'Diario_02AM';
GO

EXEC sp_add_jobserver @job_name = N'Backup_EscuelaLab_Diario';
GO
```