

UNIVERSIDAD PERUANA LOS ANDES

**FACULTAD INGENIERIA INGENIERÍA DE
SISTEMAS Y COMPUTACIÓN**



**MANUAL DOCUMENTA EL DESARROLLO DE DIEZ PROYECTOS
PRÁCTICOS**

ASIGNATURA: BASE DE DATOS II

DOCENTE: FERNÁNDEZ BEJARANO RAUL

ESTUDIANTE: Bonifacio Hilario Erick

CÓDIGO: S01238F

HUANCAYO-2025

1. INTRODUCCIÓN

El presente manual documenta el desarrollo de diez proyectos prácticos realizados en el entorno de Microsoft SQL Server, mediante el lenguaje T-SQL. El objetivo es aplicar comandos de administración, configuración, seguridad, automatización, auditoría y recuperación de la base de datos denominada *QatuPeru*. Cada proyecto está estructurado con enunciado, consulta SQL, explicación técnica y buenas prácticas o conclusiones, para promover un aprendizaje sistemático y profesional.

2. OBJETIVO GENERAL

Aplicar instrucciones T-SQL en la base de datos *QatuPeru* para configurar, mantener y asegurar su correcta operación, así como garantizar su integridad, disponibilidad y rendimiento.

3. DESARROLLO DE ACTIVIDADES

PROYECTO 1: CREACIÓN Y DISTRIBUCIÓN DE ARCHIVOS FÍSICOS

Enunciado: Crear la base de datos *QatuPeru* con sus archivos de datos y registro en rutas definidas además de consultar los archivos físicos.

Consulta SQL:

```
mkdir C:\SQLData
mkdir C:\SQLBackups

USE master;
GO
CREATE DATABASE QatuPeru
ON PRIMARY
(
    NAME = N'QatuPeru_data',
    FILENAME = N'C:\SQLData\QatuPeru_data.mdf',
    SIZE = 10MB,
    FILEGROWTH = 5MB
)
LOG ON
(
    NAME = N'QatuPeru_log',
    FILENAME = N'C:\SQLData\QatuPeru_log.ldf',
```

```
SIZE = 5MB,  
FILEGROWTH = 1MB  
);  
GO
```

The screenshot shows the SQL Server Management Studio interface. On the left, the Object Explorer tree shows the connection to 'localhost\SQLEXPRESS' with various databases listed. The central pane displays a query window titled 'SQLQuery1.sq...N\schat (58)*'. The query itself creates a primary file and a log file for the 'QatuPeru' database:

```
4    ON PRIMARY  
5    ( NAME = N'QatuPeru_data',  
6      FILENAME = N'C:\SQLData\QatuPeru_data.mdf',  
7      SIZE = 10MB,  
8      FILEGROWTH = 5MB  
9    )  
10   LOG ON  
11   ( NAME = N'QatuPeru_log',  
12     FILENAME = N'C:\SQLData\QatuPeru_log.ldf',  
13     SIZE = 5MB,  
14     FILEGROWTH = 1MB  
15   );  
16   GO
```

The status bar at the bottom of the query window indicates 'Línea: 18' (Line: 18), 'Carácter: 3' (Character: 3), and 'TABUL' (TAB). Below the query window, the 'Mensajes' (Messages) pane shows the completion message: 'Los comandos se han completado correctamente.' (The commands completed successfully.) and the end time: 'Hora de finalización: 2025-11-06T18:15:41.6490003-05:00'.

```
USE QatuPeru;  
GO  
SELECT  
    name AS NombreLogico,  
    physical_name AS RutaFisica,  
    type_desc AS TipoArchivo,  
    size * 8 / 1024 AS TamañoMB,  
    CASE  
        WHEN is_percent_growth = 0 THEN CAST(growth * 8 / 1024 AS  
VARCHAR(20)) + ' MB'  
        ELSE CAST(growth AS VARCHAR(20)) + ' %'  
    END AS Crecimiento  
FROM sys.database_files;
```

GO

The screenshot shows a SQL Server Management Studio (SSMS) interface. In the top window, a query named 'SQLQuery1.sq...' is displayed with the following T-SQL code:

```
1 USE QatuPeru;
2 GO
3 SELECT
4     name AS NombreLogico,
5     physical_name AS RutaFisica,
6     type_desc AS TipoArchivo,
7     size * 8 / 1024 AS TamañoMB,
8     CASE
9         WHEN is_percent_growth = 0 THEN CAST(growth * 8 / 1024 AS VARCHAR(20)) + '%'
10        ELSE CAST(growth AS VARCHAR(20)) + '%'
11    END AS Crecimiento
12 FROM sys.database_files;
13 GO
14
```

The bottom section shows the 'Resultados' tab selected, displaying a table with two rows of data:

	NombreLogico	RutaFisica	TipoArchivo	TamañoMB	Crecimiento
1	QatuPeru_data	C:\SQLData\QatuPeru_data.mdf	ROWS	10	5 MB
2	QatuPeru_log	C:\SQLData\QatuPeru_log.ldf	LOG	5	1 MB

Explicación técnica: Se crean dos carpetas para separar datos y respaldos. La instrucción CREATE DATABASE define los archivos físicos con tamaño inicial y crecimiento controlado. La consulta a sys.database_files muestra los detalles de los archivos existentes.

Buenas prácticas / Conclusión: Separar los archivos de datos y de log en discos distintos ayuda en el rendimiento. Establecer crecimiento fijo (no demasiado frecuente) evita fragmentación. Verificar los parámetros ejecutando la consulta.

PROYECTO 2: AJUSTE DE CONFIGURACIÓN Y PROPIEDADES

a) Consultar propiedades y modificar colación

Enunciado: Verificar propiedades de la base QatuPeru (colación, modelo de recuperación, nivel de compatibilidad) y cambiar la colación a Modern_Spanish_CI_AS.

Consulta SQL:

```
USE master;
GO
SELECT
    name AS BaseDatos,
    collation_name AS Colacion,
    recovery_model_desc AS ModeloRecuperacion,
    compatibility_level AS NivelCompatibilidad
FROM sys.databases
WHERE name = 'QatuPeru';
GO
```

```
ALTER DATABASE QatuPeru
COLLATE Modern_Spanish_CI_AS;
GO
```

```
SELECT
    name,
    collation_name
FROM sys.databases
WHERE name = 'QatuPeru';
GO
```

```

8   FROM sys.databases
9   WHERE name = 'QatuPeru';
10  GO
11
12  ALTER DATABASE QatuPeru
13  COLLATE Modern_Spanish_CI_AS;
14  GO
15
16  SELECT
17      name,
18      collation_name
19  FROM sys.databases
20  WHERE name = 'QatuPeru';
21  GO
22

```

Línea: 22 Carácter: 2 TABULACIONES CRLF

	BaseDatos	Colacion	ModeloRecuperacion	NivelCompatibilidad
1	QatuPeru	Modern_Spanish_CI_AS	SIMPLE	160

	name	collation_name
1	QatuPeru	Modern_Spanish_CI_AS

Activar Windows

Explicación técnica: La colación define reglas de comparación y ordenamiento de caracteres. Modern_Spanish_CI_AS admite tildes y ñ adecuadamente. Verificar antes y después asegura que el cambio se aplicó.

Buenas prácticas / Conclusión: Verificar propiedades antes de modificarlas, usar colación alineada al idioma del entorno, documentar cambios.

b) Modificar crecimiento automático del archivo principal a 20 MB

Enunciado: Ajustar FILEGROWTH del archivo de datos principal de QatuPeru a 20 MB y validar el cambio.

Consulta SQL:

```

USE master;
GO
ALTER DATABASE QatuPeru
MODIFY FILE
(

```

```

        NAME = N'QatuPeru_data',
        FILEGROWTH = 20MB
);

GO

USE QatuPeru;
GO
SELECT
    name AS NombreArchivo,
    type_desc AS TipoArchivo,
    size * 8 / 1024 AS TamañoActualMB,
    CASE
        WHEN is_percent_growth = 0 THEN CAST(growth * 8 / 1024 AS
VARCHAR(20)) + ' MB'
        ELSE CAST(growth AS VARCHAR(20)) + ' %'
    END AS Crecimiento
FROM sys.database_files;
GO

```

	NombreArchivo	TipoArchivo	TamañoActualMB	Crecimiento
1	QatuPeru_data	ROWS	10	20 MB
2	QatuPeru_log	LOG	5	1 MB

Explicación técnica: FILEGROWTH establece cómo aumenta el archivo cuando se llena. Un incremento fijo reduce repeticiones de crecimiento que pueden afectar rendimiento. Luego se verifica con consulta al sistema.

Buenas prácticas / Conclusión: Preferir crecimiento fijo cuando se tiene idea del volumen de datos, monitorear crecimiento para evitar saturación del disco.

PROYECTO 3: MODELO DE RECUPERACIÓN Y RESPALDO

a) Cambiar modelo de recuperación y explicar diferencias

Enunciado: Cambiar los modelos de recuperación de QatuPeru (SIMPLE, BULK_LOGGED, FULL) y conocer sus diferencias.

Consulta SQL:

```
USE master;
GO
ALTER DATABASE QatuPeru SET RECOVERY SIMPLE;
GO
SELECT name, recovery_model_desc AS Modelo
FROM sys.databases
WHERE name = 'QatuPeru';
GO
ALTER DATABASE QatuPeru SET RECOVERY BULK_LOGGED;
GO
SELECT name, recovery_model_desc AS Modelo
FROM sys.databases
WHERE name = 'QatuPeru';
GO
ALTER DATABASE QatuPeru SET RECOVERY FULL;
GO
SELECT name, recovery_model_desc AS Modelo
FROM sys.databases
WHERE name = 'QatuPeru';
GO
```

SQLQuery1.sql N\schat (58)* X

```

1   / WHERE name = 'QatuPeru';
2   GO
3   ALTER DATABASE QatuPeru SET RECOVERY BULK_LOGGED;
4   GO
5   SELECT name, recovery_model_desc AS Modelo
6   FROM sys.databases
7   WHERE name = 'QatuPeru';
8   GO
9   ALTER DATABASE QatuPeru SET RECOVERY FULL;
10  GO
11  SELECT name, recovery_model_desc AS Modelo
12  FROM sys.databases
13  WHERE name = 'QatuPeru';
14  GO
15
16
17
18
19
20
21

```

100% 1 A 0 ↑ ↓ Línea: 21 Carácter: 2 TABULACIONES CRLF

Resultados Mensajes

	name	Modelo
1	QatuPeru	SIMPLE

	name	Modelo
1	QatuPeru	BULK_LOGGED

	name	Modelo
1	QatuPeru	FULL

Activar Windows

Explicación técnica: El modelo de recuperación define el registro de transacciones. SIMPLE trunca automáticamente el log, BULK_LOGGED permite operaciones masivas con log reducido, FULL registra todo y permite recuperación punto-en-tiempo.

Buenas prácticas / Conclusión: Seleccionar el modelo adecuado según el tipo de entorno: desarrollo, carga masiva o producción crítica.

b) Realizar respaldo completo

Enunciado: Generar un backup completo de QatuPeru y verificar su integridad.

Consulta SQL:

```

USE master;
GO
BACKUP DATABASE QatuPeru
TO DISK = N'C:\SQLBackups\QatuPeru_Full.bak'
WITH
    FORMAT,
    INIT,

```

```

NAME = N'QatuPeru-Full Database Backup',
DESCRIPTION = N'Respaldo completo de QatuPeru',
STATS = 10;

GO
RESTORE HEADERONLY
FROM DISK = N'C:\SQLBackups\QatuPeru_Full.bak';
GO
RESTORE FILELISTONLY
FROM DISK = N'C:\SQLBackups\QatuPeru_Full.bak';
GO

```

The screenshot shows the SSMS interface with a query window titled "SQLQuery1.sq...N\schat (58)*". The code in the window is:

```

4   TO DISK = N'C:\SQLBackups\QatuPeru_Full.bak'
5   WITH
6     FORMAT,
7     INIT,
8     NAME = N'QatuPeru-Full Database Backup',
9     DESCRIPTION = N'Respaldo completo de QatuPeru',
10    STATS = 10;
11
12   GO
13   RESTORE HEADERONLY
14   FROM DISK = N'C:\SQLBackups\QatuPeru_Full.bak';
15   GO
16   RESTORE FILELISTONLY
17   FROM DISK = N'C:\SQLBackups\QatuPeru_Full.bak';
18   GO

```

The results pane contains two tables:

	BackupName	BackupDescription	BackupType	ExpirationDate	Compressed	P
1	QatuPeru-Full Database Backup	Respaldo completo de QatuPeru	1	NULL	0	1

	LogicalName	PhysicalName	Type	FileGroupName	Size	MaxSize	FileId
1	QatuPeru_data	C:\SQLData\QatuPeru_data.mdf	D	PRIMARY	10485760	35184372080640	1
2	QatuPeru_log	C:\SQLData\QatuPeru_log.ldf	L	NULL	5242880	2199023255552	2

Explicación técnica: El comando BACKUP DATABASE genera la copia. FORMAT e INIT garantizan archivo limpio. RESTORE HEADERONLY y FILELISTONLY permiten inspeccionar sin restaurar.

Buenas prácticas / Conclusión: Establecer política de respaldo, verificar archivo antes de confiar en él, almacenar respaldos fuera del entorno de producción.

PROYECTO 4: ROLES Y USUARIOS

Enunciado: Crear usuarios y asignar roles específicos en QatuPeru para controlar permisos.

Consulta SQL:

```
USE master;
GO
CREATE LOGIN VendedorQatu WITH PASSWORD = 'Vend123*';
GO
USE QatuPeru;
GO
CREATE USER VendedorQatu FOR LOGIN VendedorQatu;
ALTER ROLE db_datawriter ADD MEMBER VendedorQatu;
GO

USE master;
GO
CREATE LOGIN ConsultaCliente WITH PASSWORD = 'Cons123*';
GO
USE QatuPeru;
GO
CREATE USER ConsultaCliente FOR LOGIN ConsultaCliente;
ALTER ROLE db_datareader ADD MEMBER ConsultaCliente;
GO
```

Explicación técnica: Se crean dos logins en el servidor, luego usuarios en la base de datos y se asignan roles: db_datawriter para insertar/actualizar/eliminar y db_datareader sólo lectura.

Buenas prácticas / Conclusión: Seguir el principio de mínimos privilegios, usar contraseñas seguras, documentar usuarios y roles creados.

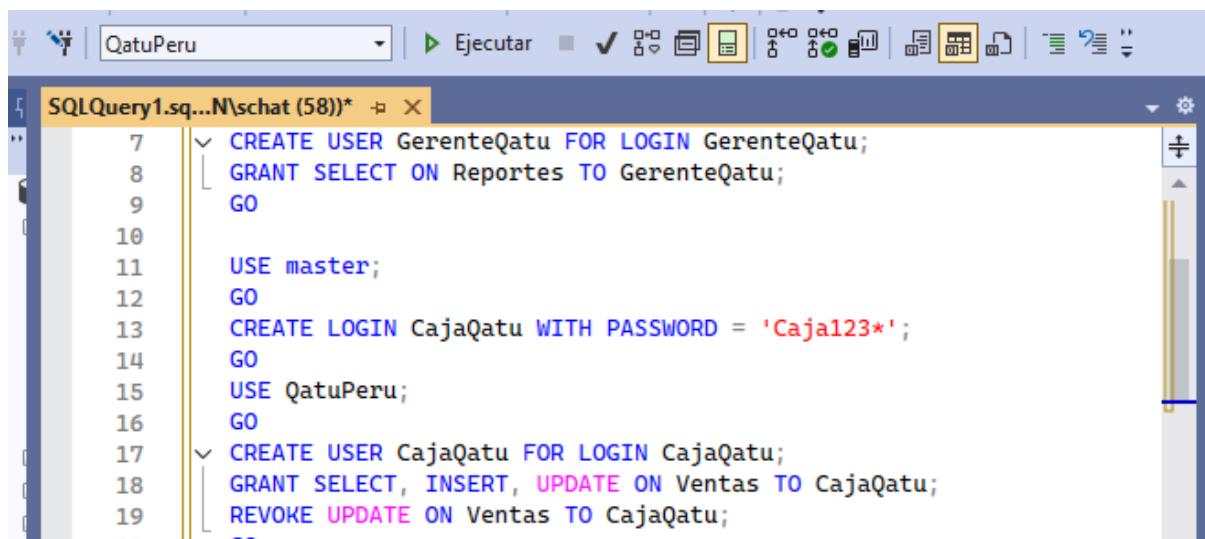
PROYECTO 5: PERMISOS GRANULARES

Enunciado: Otorgar permisos específicos a usuarios y revocar cuando sea necesario.

Consulta SQL:

```
USE master;
GO
CREATE LOGIN GerenteQatu WITH PASSWORD = 'Ger123*';
GO
USE QatuPeru;
GO
CREATE USER GerenteQatu FOR LOGIN GerenteQatu;
GRANT SELECT ON Reportes TO GerenteQatu;
GO

USE master;
GO
CREATE LOGIN CajaQatu WITH PASSWORD = 'Caja123*';
GO
USE QatuPeru;
GO
CREATE USER CajaQatu FOR LOGIN CajaQatu;
GRANT SELECT, INSERT, UPDATE ON Ventas TO CajaQatu;
REVOKE UPDATE ON Ventas TO CajaQatu;
GO
```



```
7   CREATE USER GerenteQatu FOR LOGIN GerenteQatu;
8   GRANT SELECT ON Reportes TO GerenteQatu;
9   GO
10
11  USE master;
12  GO
13  CREATE LOGIN CajaQatu WITH PASSWORD = 'Caja123*';
14  GO
15  USE QatuPeru;
16  GO
17  CREATE USER CajaQatu FOR LOGIN CajaQatu;
18  GRANT SELECT, INSERT, UPDATE ON Ventas TO CajaQatu;
19  REVOKE UPDATE ON Ventas TO CajaQatu;
```

Explicación técnica: GerenteQatu recibe permiso SELECT en Reportes. CajaQatu primero obtiene permisos de INSERT/UPDATE, luego se revoca UPDATE.

Buenas prácticas / Conclusión: Implementar permisos mínimos, revisar periódicamente los permisos otorgados y mantener auditoría.

PROYECTO 6: IDENTIFICACIÓN DE PROCESOS LENTOS

Enunciado: Identificar procesos que consumen mucho CPU o están bloqueados en la base de datos CatuPeru.

Consulta SQL:

```
-- 1. Ver bloqueos actuales
SELECT blocking_session_id AS BloqueadoPor, session_id
AS SesionBloqueada, wait_type AS TipoEspera, wait_time AS TiempoEsperaMiliseg,
wait_resource AS RecursoBloqueado, DB_NAME(database_id) AS BaseDatos
FROM sys.dm_exec_requests WHERE blocking_session_id <> 0; GO

-- 2. Ver procesos activos con uso de CPU (CORREGIDO)
SELECT r.session_id AS Sesion, r.status AS Estado, r.command AS Comando, r.cpu_time AS TiempoCPU,
r.total_elapsed_time AS TiempoTotal, DB_NAME(r.database_id) AS BaseDatos,
s.login_name AS Usuario
FROM sys.dm_exec_requests r
INNER JOIN
sys.dm_exec_sessions s ON r.session_id = s.session_id
WHERE r.status = 'running'
ORDER BY r.cpu_time DESC; GO

-- 3. Ver información de todas las sesiones activas
SELECT s.session_id AS Sesion, s.login_name AS Usuario, s.host_name AS Equipo, s.program_name AS Aplicacion,
r.cpu_time AS CPU, r.status AS Estado, r.command AS Comando
FROM sys.dm_exec_sessions s
LEFT JOIN sys.dm_exec_requests r ON s.session_id = r.session_id
WHERE s.is_user_process = 1
ORDER BY r.cpu_time DESC; GO

-- 4. Ver sesiones con más consumo de recursos
SELECT TOP 10 s.session_id AS Sesion, s.login_name AS Usuario, s.host_name AS Equipo, s.program_name AS Programa,
s.cpu_time AS TiempoCPU, s.memory_usage AS UsoMemoria,
s.total_elapsed_time AS TiempoTotal, s.reads AS Lecturas, s.writes AS Escrituras
FROM sys.dm_exec_sessions s
WHERE s.is_user_process = 1
ORDER BY s.cpu_time
```

DESC; GO

Resultados								
	BloqueadoPor	SesionBloqueada	TipoEspera	TiempoEsperaMiliseg	RecursoBloqueado	BaseDatos		
	Sesion	Estado	Comando	TiempoCPU	TiempoTotal	BaseDatos	Usuario	
1	58	running	SELECT	6	9	QatuPeru	Schatszmon\schat	
	Sesion	Usuario	Equipo	Aplicacion			CPU	Estado
1	53	Schatszmon\schat	SCHATSZMON	Microsoft SQL Server Management Studio - Transa...			127	suspe
2	58	Schatszmon\schat	SCHATSZMON	Microsoft SQL Server Management Studio - Consulta			6	runnin
3	51	NT SERVICE\S...	SCHATSZMON	SQLServerCEIP			N...	NULL
4	52	Schatszmon\schat	SCHATSZMON	SQL Server Management Studio			N...	NULL
	Sesion	Usuario	Equipo	Programa			TiempoCPU	
1	58	Schatszmon\schat	SCHATSZMON	Microsoft SQL Server Management Studio - Consulta			2875	
2	52	Schatszmon\schat	SCHATSZMON	SQL Server Management Studio			172	
3	53	Schatszmon\schat	SCHATSZMON	Microsoft SQL Server Management Studio - Transa...			0	
4	51	NT SERVICE\S...	SCHATSZMON	SQLServerCEIP			0	

Explicación técnica: Las vistas dinámicas sys.dm_exec_requests permiten ver información sobre sesiones, bloqueos y uso de CPU.

Buenas prácticas / Conclusión: Monitorear regularmente, identificar y resolver bloqueos y procesos que ralentizan el sistema.

PROYECTO 7: AUTOMATIZACIÓN CON SQL SERVER AGENT

Enunciado: Crear trabajos (Jobs) para respaldo diario automático y limpieza semanal de sesiones en QatuPeru.

Consulta SQL:

```
USE msdb;
GO
EXEC sp_add_job @job_name = N'RespaldoDiarioQatuPeru';
EXEC sp_add_jobstep
    @job_name = N'RespaldoDiarioQatuPeru',
    @step_name = N'Ejecutar Backup',
    @subsystem = N'TSQL',
    @command = N'BACKUP DATABASE QatuPeru TO DISK =
'C:\SQLBackups\QatuPeru_Diario.bak' WITH INIT;',
    @database_name = N'master';
GO
EXEC sp_add_schedule
    @schedule_name = N'DiarioA2AM',
    @freq_type = 4,
    @freq_interval = 1,
    @active_start_time = 20000;
```

```
GO
EXEC sp_attach_schedule
    @job_name = N'RespaldoDiarioQatuPeru',
    @schedule_name = N'DiarioA2AM';
GO
EXEC sp_add_jobserver
    @job_name = N'RespaldoDiarioQatuPeru';
GO

EXEC sp_add_job @job_name = N'LimpiezaSemanalSesiones';
GO
EXEC sp_add_jobstep
    @job_name = N'LimpiezaSemanalSesiones',
    @step_name = N'Eliminar Sesiones Antiguas',
    @subsystem = N'TSQL',
    @command = N'DELETE FROM Sesiones WHERE FechaInicio <
DATEADD(DAY, -15, GETDATE());',
    @database_name = N'QatuPeru';
GO
EXEC sp_add_schedule
    @schedule_name = N'SemanalDomingo3AM',
    @freq_type = 8,
    @freq_interval = 1,
    @freq_recurrence_factor = 1,
    @active_start_time = 30000;
GO
EXEC sp_attach_schedule
    @job_name = N'LimpiezaSemanalSesiones',
    @schedule_name = N'SemanalDomingo3AM';
GO
EXEC sp_add_jobserver
    @job_name = N'LimpiezaSemanalSesiones';
```

GO

```
msdb Ejecutar ✓ SQLQuery1.sq...N\schat (58)* X
34     EXEC sp_add_schedule
35         @schedule_name = N'SemanalDomingo3AM',
36         @freq_type = 8,
37         @freq_interval = 1,
38         @freq_recurrence_factor = 1,
39         @active_start_time = 30000;
39
40 GO
41 EXEC sp_attach_schedule
42     @job_name = N'LimpiezaSemanalSesiones',
43     @schedule_name = N'SemanalDomingo3AM';
44 GO
45 EXEC sp_add_jobserver
46     @job_name = N'LimpiezaSemanalSesiones';
47 GO
48

100 % ① 0 ↑ ↓ Línea: 48 Carácter: 2 TABULACIONES CRLF
Mensajes
SQLServerAgent is not currently running so it cannot be notified of this action.
SQLServerAgent is not currently running so it cannot be notified of this action.

Hora de finalización: 2025-11-06T18:20:57.3506948-05:00
```

Explicación técnica: SQL Server Agent permite programar tareas automáticas mediante `sp_add_job`, `sp_add_jobstep`, `sp_add_schedule` y `sp_attach_schedule`. El primer trabajo realiza respaldo diario; el segundo limpia semanalmente sesiones antiguas.

Buenas prácticas / Conclusión: Automatizar tareas repetitivas reduce errores humanos. Verificar regularmente que los trabajos se ejecutan correctamente y gestionar sus historiales.

PROYECTO 8: REGISTRO DE ESTADOS EN PEDIDOS

Enunciado: Modificar la estructura de la tabla Pedidos agregando la columna Prioridad y luego eliminar la columna EstadoEnvio.

Consulta SQL:

```
USE QatuPeru;
GO
ALTER TABLE Pedidos ADD Prioridad INT NULL;
GO
```

```
SELECT
    COLUMN_NAME AS Columna,
    DATA_TYPE AS TipoDato,
    IS_NULLABLE AS PermiteNulos
FROM INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_NAME = 'Pedidos';
GO
INSERT INTO Pedidos (ClienteID, Total, Prioridad)
VALUES (1, 1500.00, 1), (1, 2500.00, 2);
GO
SELECT * FROM Pedidos;
GO

ALTER TABLE Pedidos ADD EstadoEnvio VARCHAR(50) NULL;
GO
UPDATE Pedidos SET EstadoEnvio = 'Pendiente';
GO
SELECT * FROM Pedidos;
GO
ALTER TABLE Pedidos DROP COLUMN EstadoEnvio;
GO
SELECT
    COLUMN_NAME AS Columna,
    DATA_TYPE AS TipoDato
FROM INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_NAME = 'Pedidos';
```

GO

The screenshot shows the SSMS interface. In the top window, titled 'SQLQuery1.sq...N\schat (58)*', there is a code editor with the following T-SQL script:

```
26      COLUMN_NAME AS Columna,
27      DATA_TYPE AS TipoDato
28  FROM INFORMATION_SCHEMA.COLUMNS
29  WHERE TABLE_NAME = 'Pedidos';
30  GO
31  USE QatuPeru;
32  GO
33
34
35
```

The results pane below displays three tables of data:

	PedidoID	ClienteID	FechaPedido	Total	Prioridad
1	1	1	2025-11-06 18:23:19.877	1500.00	1
2	2	1	2025-11-06 18:23:19.877	2500.00	2

	PedidoID	ClienteID	FechaPedido	Total	Prioridad	EstadoEnvio
1	1	1	2025-11-06 18:23:19.877	1500.00	1	Pendiente
2	2	1	2025-11-06 18:23:19.877	2500.00	2	Pendiente

	Columna	TipoDato
1	PedidoID	int
2	ClienteID	int
3	FechaP...	datetime
4	Total	decimal

Explicación técnica: Se agrega columna Prioridad para indicar nivel de prioridad en el pedido, luego se añade columna EstadoEnvio sólo para mostrar su eliminación, y finalmente se elimina. Se usan consultas a INFORMATION_SCHEMA para verificar la estructura.

Buenas prácticas / Conclusión: Validar cambios de estructura antes y después de realizarlos, usar migraciones controladas, mantener historial de cambios de esquema.

PROYECTO 9: AUDITORÍA CON TRIGGERS

Enunciado: Crear una tabla de auditoría para la tabla Clientes y un trigger que registre eliminaciones.

Consulta SQL:

```
USE QatuPeru;
GO
```

```

CREATE TABLE AuditoriaClientes
(
    AuditoriaID INT IDENTITY(1,1) PRIMARY KEY,
    ClienteID INT,
    Nombre VARCHAR(100),
    Email VARCHAR(100),
    Telefono VARCHAR(20),
    Accion VARCHAR(20),
    FechaAccion DATETIME DEFAULT GETDATE(),
    Usuario VARCHAR(100) DEFAULT SYSTEM_USER
);
GO
SELECT
    COLUMN_NAME AS Columna,
    DATA_TYPE AS TipoDato
FROM INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_NAME = 'AuditoriaClientes';
GO

CREATE TRIGGER trg_AuditoriaClientes_Delete
ON Clientes
AFTER DELETE
AS
BEGIN
    SET NOCOUNT ON;
    INSERT INTO AuditoriaClientes (ClienteID, Nombre, Email,
    Telefono, Accion)
    SELECT
        ClienteID,
        Nombre,
        Email,
        Telefono,
        'DELETE'
    FROM deleted;
END;
GO

INSERT INTO Clientes (Nombre, Email, Telefono)
VALUES ('Juan Pérez', 'juan@email.com', '123456789');
GO
DELETE FROM Clientes WHERE Nombre = 'Juan Pérez';

```

```
GO
```

```
SELECT AuditoriaID, ClienteID, Nombre, Accion, FechaAccion, Usuario  
FROM AuditoriaClientes;
```

```
GO
```

The screenshot shows a SQL query window titled "SQLQuery1.sql...N\schat (58)*". The code creates the "AuditoriaClientes" table with columns: AuditoriaID (primary key, identity), ClienteID (int), Nombre (varchar(100)), Email (varchar(100)), Telefono (varchar(20)), Accion (varchar(20)), FechaAccion (datetime default GETDATE()), and Usuario (varchar(100) default SYSTEM_USER). A test insert is shown with ClienteID 4, Nombre "Juan Pérez", Accion "DELETE", and Usuario "Schatszmon\schat". The results tab displays the table structure and the inserted row.

```
USE QatuPeru;
GO
CREATE TABLE AuditoriaClientes
(
    AuditoriaID INT IDENTITY(1,1) PRIMARY KEY,
    ClienteID INT,
    Nombre VARCHAR(100),
    Email VARCHAR(100),
    Telefono VARCHAR(20),
    Accion VARCHAR(20),
    FechaAccion DATETIME DEFAULT GETDATE(),
    Usuario VARCHAR(100) DEFAULT SYSTEM_USER
);
GO
SPLIT
```

	Columna	TipoDato
1	AuditoriaID	int
2	ClienteID	int
3	Nombre	varchar
4	Email	varchar
5	Telefono	varchar
6	Accion	varchar
7	FechaAccion	datetime
8	Usuario	varchar

	AuditoriaID	ClienteID	Nombre	Accion	FechaAccion	Usuario
1	1	4	Juan Pérez	DELETE	2025-11-06 18:23:49.407	Schatszmon\schat

Explicación técnica: La tabla AuditoriaClientes almacena el historial de eliminaciones de clientes. El trigger en la tabla Clientes captura el evento AFTER DELETE e inserta la información correspondiente en la tabla de auditoría. Se realiza inserción de prueba y eliminación para comprobar su funcionamiento.

Buenas prácticas / Conclusión: Implementar auditoría para operaciones críticas, usar triggers con cuidado para evitar impacto de rendimiento, documentar triggers y su propósito.

PROYECTO 10: SIMULACIÓN DE RESTAURACIÓN

Enunciado: Simular la pérdida de datos en la tabla Clientes y luego restaurar la base de datos desde un respaldo previo.

Consulta SQL:

```
USE QatuPeru;
GO
INSERT INTO Clientes (Nombre, Email, Telefono)
VALUES
    ('María López', 'maria@email.com', '987654321'),
    ('Carlos Ruiz', 'carlos@email.com', '456789123'),
    ('Ana Torres', 'ana@email.com', '789123456');
GO
SELECT * FROM Clientes;
GO
BACKUP DATABASE QatuPeru
TO DISK = N'C:\SQLBackups\QatuPeru_AntesIncidente.bak'
WITH FORMAT, INIT;
GO
DELETE FROM Clientes;
GO
SELECT * FROM Clientes;
SELECT COUNT(*) AS TotalClientes FROM Clientes;
GO

USE master;
GO
ALTER DATABASE QatuPeru SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
GO
RESTORE DATABASE QatuPeru
FROM DISK = N'C:\SQLBackups\QatuPeru_AntesIncidente.bak'
WITH REPLACE;
GO
ALTER_DATABASE QatuPeru SET MULTI_USER;
GO

USE QatuPeru;
GO
SELECT
    ClienteID,
    Nombre,
    Email,
    Telefono,
```

```

FechaRegistro
FROM Clientes
ORDER BY ClienteID;
GO
SELECT COUNT(*) AS TotalClientesRestaurados FROM Clientes;
GO
SELECT * FROM AuditoriaClientes;
GO

```

SQLQuery1.sql...Ejecutando...* X

```

33 GO
34 ✓ SELECT
35 ClienteID,
36 Nombre,
37 Email,
38 Telefono,
39 FechaRegistro
40 FROM Clientes
41 ORDER BY ClienteID;
42 GO
43 SELECT COUNT(*) AS TotalClientesRestaurados FROM Clientes;
44 GO
45 SELECT * FROM AuditoriaClientes;
46 GO
47

```

100 % 19 0 ↑ ↓ Línea: 47 Carácter: 1 TABULACIONES CRLF

Resultados Mensajes

	ClienteID	Nombre	Email	Telefono	FechaRegistro
1	1	Pedro Martínez	pedro@email.com	111222333	2025-11-06 18:23:14.393
2	2	Laura García	laura@email.com	444555666	2025-11-06 18:23:14.393
3	3	Miguel Sánchez	miguel@email.com	777888999	2025-11-06 18:23:14.393
4	5	María López	maria@email.com	987654321	2025-11-06 18:24:34.133
5	6	Carlos Ruiz	carlos@email.com	456789123	2025-11-06 18:24:34.133
6	7	Ana Torres	ana@email.com	789123456	2025-11-06 18:24:34.133
7	8	María López	maria@email.com	987654321	2025-11-06 18:24:41.797
8	9	Carlos Ruiz	carlos@email.com	456789123	2025-11-06 18:24:41.797

	ClienteID	Nombre	Email	Telefono	FechaRegistro
1	1	Pedro Martínez	pedro@email.com	111222333	2025-11-06 18:23:14.393
2	2	Laura García	laura@email.com	444555666	2025-11-06 18:23:14.393

Ejecutando con | Localhost\SQL EXPRESS (16.0) | SCHATSZMON\schat (581) | Octubre | 00:00:00 | 0 filas

Explicación técnica: Se insertan datos de prueba en Clientes, se hace respaldo, se eliminan los datos, luego se cambia la base a SINGLE_USER para restaurar, se hace la restauración con REPLACE y se vuelve a MULTI_USER. Finalmente se verifican los datos restaurados.

Buenas prácticas / Conclusión: Practicar restauraciones ayuda a estar preparado frente a incidentes. Mantener respaldos recientes, probar restauraciones periódicamente y documentar el proceso.

