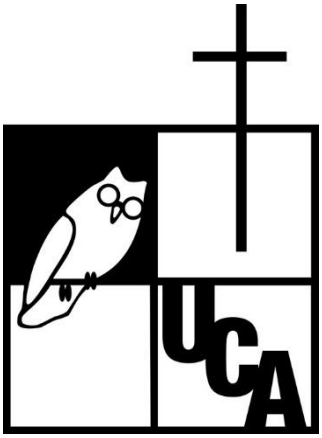
Universidad Centroamericana José Simeón Cañas

Facultad de Ingeniería y Arquitectura - Ingeniería informática

Ing. James Edward Humberstone Morales



Proyecto de Cátedra – Administración de Bases de Datos con SQL Server: “ONG de Donaciones”

Integrantes:

Oscar Eduardo Mercado Guerra 00113124

Rodrigo Bohery Torres Peres 00109524

Daniel Alejandro Murcia Garay 00070523

Cesar Eduardo Navarrete Chavarria 00007023

Fecha: 26 de noviembre de 2025

Modelo Entidad–Relación (ER)

El modelo se compone de las siguientes entidades principales:

País - Representa cualquier país donde opera la ONG.

- Se relaciona con tres entidades:
  - Donante
  - Proyecto
  - Beneficiario

Donante - Contiene información del donante: nombre, contacto, país y fecha de registro. Relaciones:

- Un donante puede realizar muchas donaciones**.**
- Pertenece a un país.

Proyecto - Incluye todos los datos relacionados a un proyecto social. Relaciones:

- Un proyecto puede recibir muchas donaciones**.**
- Cada proyecto está ubicado en un país.

TipoDonacion - Registra los tipos permitidos de donaciones: dinero, alimentos, materiales, servicios, etc. Relaciones:

- Un tipo de donación puede ser usado en muchas donaciones.

Donación - Registra cada aporte realizado por un donante a un proyecto. Incluye:

- Monto donado
- Tipo de donación
- Fecha

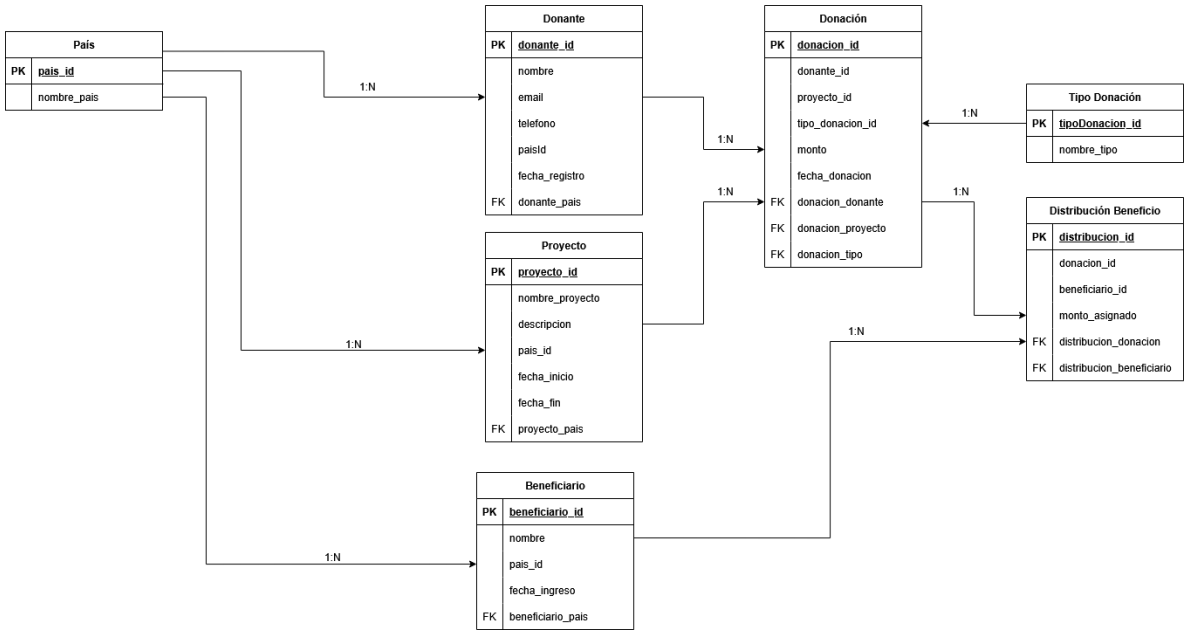
Relaciones:

- Una donación pertenece a:
  - Un donante
  - Un proyecto
  - Un tipo de donación
- Una donación puede distribuirse entre muchos beneficiarios.

Beneficiario - Registra personas que reciben apoyo de la ONG. Relaciones: Un beneficiario puede recibir apoyo desde múltiples donaciones**,** mediante la tabla DistribuciónBeneficio.

DistribuciónBeneficio - Es una entidad puente que permite asignar montos de una donación a distintos beneficiarios. Relaciones:

- Una donación puede tener muchas distribuciones**.**
- Un beneficiario puede recibir aportes de muchas donaciones**.**



Roles de seguridad implementados

La base de datos fue organizada mediante esquemas funcionales, permitiendo separar la información según su propósito y aplicar controles de seguridad específicos para cada conjunto de objetos. Esta división facilita la administración, evita conflictos de nombres y mejora la claridad del modelo, además de permitir respaldos o migraciones selectivas.

Para gestionar los permisos de forma ordenada, se definieron los siguientes roles:

rol_administrador - Posee control total sobre la base de datos. Puede crear, modificar o eliminar objetos, administrar usuarios, roles y aplicar cambios estructurales. Tiene también permisos a nivel servidor para operaciones administrativas avanzadas.

Esquema	Objeto	SELECT	INSERT	UPDATE	DELETE	CREATE	ALTER	DROP	BACKUP	EXECUTE
donaciones	donante	✓	✓	✓	✓	✓	✓	✓		✓
donaciones	beneficiario	✓	✓	✓	✓	✓	✓	✓		✓
proyectos	proyecto	✓	✓	✓	✓	✓	✓	✓		✓
proyectos	asignación	✓	✓	✓	✓	✓	✓	✓		✓
finanzas	pago	✓	✓	✓	✓		✓	✓		✓
finanzas	factura	✓	✓	✓	✓		✓	✓		✓
seguridad	usuarios	✓	✓	✓	✓	✓	✓	✓	✓	✓
seguridad	roles	✓	✓	✓	✓	✓	✓	✓	✓	✓

rol_backup - Su función es exclusivamente la gestión de respaldos y restauraciones. Cuenta con permisos para ejecutar BACKUP DATABASE y RESTORE, garantizando que solo personal autorizado realice tareas de recuperación.

Esquema	Objeto	SELECT	INSERT	UPDATE	DELETE	CREATE	ALTER	DROP	BACKUP	EXECUTE
Todos	Todos los objetos								✓	

rol_coordinador_proyectos - Encargado de supervisar la operación de los proyectos de la ONG. Posee permisos de SELECT sobre todos los esquemas, e INSERT/UPDATE sobre el esquema que contiene los proyectos y asignación de beneficiarios.

Esquema	Objeto	SELECT	INSERT	UPDATE	DELETE	CREATE	ALTER	DROP	BACKUP	EXECUTE
proyectos	proyecto	✓	✓	✓						
proyectos	asignación	✓	✓	✓						
donaciones	donante	✓								
donaciones	beneficiario	✓								

rol_analista_reportes - Enfocado en el análisis y visualización de datos. Cuenta con permisos de lectura sobre todas las tablas y vistas que permiten generar reportes operativos y financieros.

Esquema	Objeto	SELECT	INSERT	UPDATE	DELETE	CREATE	ALTER	DROP	BACKUP	EXECUTE
donaciones	donante	✓								
donaciones	beneficiario	✓								
proyectos	proyecto	✓								
finanzas	factura	✓								
finanzas	pago	✓								
reportes	vista_reportes	✓								

Auditoría SQL

Para asegurar la trazabilidad sobre los datos sensibles de la ONG, se implementó una auditoría específicamente orientada a monitorear cambios en la tabla **Donación**, que contiene montos aportados, método de pago y fechas. Esto permite registrar automáticamente acciones de INSERT, UPDATE y DELETE, manteniendo un historial confiable que facilita revisiones internas o auditorías externas.

Esta auditoría se almacena en una ruta específica en el servidor, lo que permite inspeccionar eventos, identificar accesos indebidos y garantizar la integridad de la información financiera de la ONG.

Script

```
USE master;
GO

CREATE SERVER AUDIT Audit_Donaciones
TO FILE (FILEPATH = 'C:\Auditorias\ONG\', MAXSIZE = 100 MB)
WITH (ON_FAILURE = CONTINUE);
GO

ALTER SERVER AUDIT Audit_Donaciones
WITH (STATE = ON);
GO

USE ONG_Donaciones;
GO

CREATE DATABASE AUDIT SPECIFICATION Audit_ONG
FOR SERVER AUDIT Audit_Donaciones
ADD (UPDATE ON SCHEMA::ong BY PUBLIC),
ADD (INSERT ON SCHEMA::ong BY PUBLIC),
ADD (DELETE ON SCHEMA::ong BY PUBLIC),
ADD (SELECT ON SCHEMA::ong BY PUBLIC);
GO

ALTER DATABASE AUDIT SPECIFICATION Audit_ONG
WITH (STATE = ON);
GO

-- Prueba de Auditoría
INSERT INTO ong.Pais (NombrePais) VALUES ('Belice');
UPDATE ong.Pais SET NombrePais = 'Belice Editado' WHERE NombrePais = 'Belice';
DELETE FROM ong.Pais WHERE NombrePais = 'Belice Editado';

-- Consultas de Log
SELECT * FROM sys.fn_get_audit_file('C:\Auditorias\ONG\*.sqlaudit', DEFAULT, DEFAULT);
GO
```

Se creó un *Server Audit* llamado Audit_Donaciones, configurado para almacenar los eventos auditados en archivos dentro de la ruta *C:\Auditorias\ONG*.
La directiva ON_FAILURE = CONTINUE permite que el servidor siga funcionando aun si ocurre un error en la auditoría, evitando que una falla bloquee operaciones críticas.

Luego se activó el audit con:

```
ALTER SERVER AUDIT Audit_Donaciones WITH (STATE = ON);
```

Esto habilita el componente a nivel de servidor para comenzar a recibir eventos desde cualquier base que lo use.

La *Database Audit Specification* Audit_ONG fue creada dentro de la base *ONG_Donaciones*, enlazándola directamente al *Server Audit*.
Esta configuración permite auditar todas las operaciones críticas en las tablas del esquema ong**,** incluyendo:

- **INSERT:** registra nuevas donaciones, beneficiarios, proyectos, etc.
- **UPDATE:** detecta cambios en datos sensibles, como montos asignados o modificación de beneficiarios.
- **DELETE:** asegura que se registren eliminaciones de registros, permitiendo rastrear posibles pérdidas de información.
- **SELECT:** registra consultas para detectar accesos indebidos a datos sensibles (donaciones, beneficiarios, donantes).

Todo esto se aplica a cualquier usuario del rol PUBLIC, lo que significa que cualquier operación sobre el esquema queda auditada sin excepción**.**

La especificación se activó con:

```
ALTER DATABASE AUDIT SPECIFICATION Audit_ONG WITH (STATE = ON);
```

La implementación de esta auditoría proporciona:

- Trazabilidad completa: cada acción queda registrada con usuario, fecha y operación realizada.
- Seguridad reforzada**:** permite identificar accesos no autorizados o manipulaciones indebidas.
- Cumplimiento normativo: facilita auditorías externas o revisiones internas de transparencia, especialmente importante en organizaciones de carácter social como una ONG.
- Control sobre datos sensibles: montos donados, beneficiarios y proyectos quedan monitoreados.

Con esto la organización mantiene un historial confiable de todas las operaciones ejecutadas en la base de datos.

Estrategia de Backup y Restauración

Para garantizar la disponibilidad, integridad y recuperación completa de la base de datos **ONG_Donaciones**, se implementó un **plan de respaldo híbrido** compuesto por tres tipos de backups: **FULL**, **DIFERENCIAL** y **LOG**.
Este enfoque permite restaurar el sistema incluso ante una falla crítica, minimizando la pérdida de información y reduciendo el tiempo de recuperación (RTO).

1. Configuración del Modo de Recuperación

La base de datos se configuró en modo RECOVERY FULL, lo cual habilita el uso de respaldos de LOG y permite restaurar hasta un punto específico en el tiempo.

```
ALTER DATABASE ONG_Donaciones
SET RECOVERY FULL;
GO
```

2. Backup FULL

El backup completo captura todos los datos de la base de datos en un único archivo. Este tipo de respaldo es la base del resto de restauraciones y sirve como punto inicial sobre el cual se aplican diferenciales y logs.

```
-- FULL backup
BACKUP DATABASE ONG_Donaciones
TO DISK = 'C:\Backups\ONG\ONG_FULL.bak'
WITH FORMAT, INIT, NAME = 'ONG_Donaciones-FULL', SKIP, STATS = 10;
```

La restauración de verificación se realiza con:

```
-- Verificar
RESTORE DATABASE ONG_Donaciones
FROM DISK = 'C:\Backups\ONG\ONG_FULL.bak'
WITH REPLACE,
     NORECOVERY;
```

3. Backup Diferencial

El backup diferencial almacena solo los cambios realizados desde el último backup FULL**.** Esto acelera el proceso de restauración y reduce el tamaño del archivo comparado con un backup completo.

```
-- DIFF backup
BACKUP DATABASE ONG_Donaciones
TO DISK = 'C:\Backups\ONG\ONG_DIFF.bak'
WITH DIFFERENTIAL, INIT, NAME = 'ONG_Donaciones-DIFF', STATS = 10;
```

Se verifica igualmente con:

```
-- Verificar
RESTORE DATABASE ONG_Donaciones
FROM DISK = 'C:\Backups\ONG\ONG_DIFF.bak'
WITH REPLACE,
     NORECOVERY;
```

4. Backup del LOG de Transacciones

El backup de log captura todas las transacciones recientes no incluidas en otros backups. Este respaldo permite una recuperación granular hasta el punto exacto antes de una falla.

```
-- LOG backup
BACKUP LOG ONG_Donaciones
TO DISK = 'C:\Backups\ONG\ONG_LOG.trn'
WITH INIT,
     NAME = 'ONG_Donaciones-LOG',
     SKIP, STATS = 10;
```

Su restauración se ejecuta con:

```
-- Verificar
RESTORE LOG ONG_Donaciones
FROM DISK = 'C:\Backups\ONG\ONG_LOG.trn'
WITH REPLACE,
     RECOVERY;
```

Creación de Jobs de Backup

Con el objetivo de automatizar por completo la estrategia de respaldo definida para la base de datos **se** implementó un conjunto de Jobs de SQL Server Agent**.** Estos trabajos permiten ejecutar los backups de forma programada, validar su integridad y aplicar una política de limpieza automática. Esto garantiza continuidad operativa, disminuye riesgos de pérdida de información y elimina la necesidad de intervención manual.

A continuación se describen los cuatro jobs implementados:

1. Job de Backup FULL Diario - Este job genera un respaldo completo de la base de datos todos los días a las 2:00 AM**,** momento de baja carga. Características principales:

- Construye dinámicamente el nombre del archivo incluyendo timestamp.
- Ejecuta el comando `BACKUP DATABASE` con INIT.
- Ejecuta `RESTORE VERIFYONLY` para validar que el backup no esté corrupto.
- Se programa mediante un schedule diario.

Este respaldo sirve como punto base para restauraciones completas o para aplicar respaldos diferenciales y de log posteriormente.

```
USE msdb;
GO
-- Job FULL
-- Variables:
DECLARE @job_name NVARCHAR(128) = N'ONG_Backup_FULL_Diario';
DECLARE @owner_login NVARCHAR(128) = N'sa';
DECLARE @command NVARCHAR(MAX);
SET @command = N'
DECLARE @BackupFolder NVARCHAR(400) = N''C:\Backups\ONG\'';
DECLARE @dbname NVARCHAR(128) = N''ONG_Donaciones'';
DECLARE @file NVARCHAR(400);
SET @file = @BackupFolder + @dbname + N''_FULL_'' + REPLACE(CONVERT(VARCHAR(19), GETDATE(), 120), '':'' ,'') + ''.bak'';
BACKUP DATABASE [ONG_Donaciones] TO DISK = @file WITH INIT, NAME = N''ONG_Donaciones-FULL'', SKIP, STATS = 10;
RESTORE VERIFYONLY FROM DISK = @file;
';

IF EXISTS (SELECT 1 FROM msdb.dbo.sysjobs WHERE name = @job_name)
  EXEC msdb.dbo.sp_delete_job @job_name = @job_name, @delete_unused_schedule=1;

-- Crear job
EXEC msdb.dbo.sp_add_job @job_name = @job_name, @enabled = 1, @description = 'FULL backup diario ONG_Donaciones', @owner_login_name = @owner_login;

EXEC msdb.dbo.sp_add_jobstep
  @job_name = @job_name,
  @step_name = N'Run Full Backup',
  @subsystem = N'TSQL',
  @command = @command,
  @retry_attempts = 2,
  @retry_interval = 5;

-- Crear horario (diario a las 02:00)
EXEC msdb.dbo.sp_add_schedule @schedule_name = N'ONG_FULL_Diario_Schedule', @freq_type = 4, @freq_interval = 1, @active_start_time = 20000; -- 02:00
-- Attach schedule to job
EXEC msdb.dbo.sp_attach_schedule @job_name = @job_name, @schedule_name = N'ONG_FULL_Diario_Schedule';

-- Añadir Job al servidor
EXEC msdb.dbo.sp_add_jobserver @job_name = @job_name;
GO
```

2. Job de Backup Diferencial cada 6 horas - Este job ejecuta un backup diferencial cada 6 horas con el objetivo de capturar todos los cambios ocurridos desde el último backup FULL.

- El archivo también incluye la fecha y hora para mantener un orden cronológico.
- Incluye validación de integridad con `RESTORE VERIFYONLY`.
- El horario se define usando `freq_subday_type = 8` (cada N horas) con intervalo de 6.

Este tipo de respaldo reduce significativamente el tamaño de los archivos y acelera el proceso de recuperación en comparación con depender únicamente de respaldos de log.

```
-- Job DIFF
USE msdb;
GO
DECLARE @job_name NVARCHAR(128) = N'ONG_Backup_DIFF_6H';
DECLARE @owner_login NVARCHAR(128) = N'sa';
DECLARE @command NVARCHAR(MAX) = N'
DECLARE @BackupFolder NVARCHAR(400) = N''C:\Backups\ONG\'';
DECLARE @dbname NVARCHAR(128) = N''ONG_Donaciones'';
DECLARE @file NVARCHAR(400);
SET @file = @BackupFolder + @dbname + N''_DIFF_'' + REPLACE(CONVERT(VARCHAR(19), GETDATE(), 120), '':'' ,'') + ''.bak'';
BACKUP DATABASE [ONG_Donaciones] TO DISK = @file WITH DIFFERENTIAL, INIT, NAME = N''ONG_Donaciones-DIFF'', SKIP, STATS = 10;
RESTORE VERIFYONLY FROM DISK = @file;
';

IF EXISTS (SELECT 1 FROM msdb.dbo.sysjobs WHERE name = @job_name)
  EXEC msdb.dbo.sp_delete_job @job_name = @job_name, @delete_unused_schedule=1;

EXEC msdb.dbo.sp_add_job @job_name = @job_name, @enabled = 1, @description = 'DIFF backup cada 6 horas ONG_Donaciones', @owner_login_name = @owner_login;

EXEC msdb.dbo.sp_add_jobstep
  @job_name = @job_name,
  @step_name = N'Run Diff Backup',
  @subsystem = N'TSQL',
  @command = @command,
  @retry_attempts = 2,
  @retry_interval = 5;

-- Frecuencia de horario: freq_type=4 (diario); freq_intervalo ignorado cuando freq_subday_type es usado
-- Use freq_subday_type=8 (cada n horas), freq_subday_interval=6
EXEC msdb.dbo.sp_add_schedule @schedule_name = N'ONG_DIFF_6H_Schedule',
    @freq_type = 4,
    @freq_interval = 1,
    @freq_subday_type = 8,
    @freq_subday_interval = 6,
    @active_start_time = 0;
EXEC msdb.dbo.sp_attach_schedule @job_name = @job_name, @schedule_name = N'ONG_DIFF_6H_Schedule';
EXEC msdb.dbo.sp_add_jobserver @job_name = @job_name;
GO
```

3. Job de Backup de Log cada 30 minutos - Dado que se configuró el Recovery Model FULL**,** el respaldo del Log de transacciones es fundamental para tener capacidad de recuperación a un punto específico en el tiempo (PITR).

Este job:

- Se ejecuta cada 30 minutos**.**
- Realiza `BACKUP LOG` hacia un archivo *.trn*.
- Verifica el archivo generado usando `RESTORE VERIFYONLY`.

Con esta frecuencia se garantiza un RPO máximo de 30 minutos**,** adecuado para datos sensibles como donaciones, asignaciones y registros de beneficiarios.

```
-- Job LOG
USE msdb;
GO
DECLARE @job_name NVARCHAR(128) = N'ONG_Backup_LOG_30min';
DECLARE @owner_login NVARCHAR(128) = N'sa';
DECLARE @command NVARCHAR(MAX) = N'
DECLARE @BackupFolder NVARCHAR(400) = N''C:\Backups\ONG\'';
DECLARE @dbname NVARCHAR(128) = N''ONG_Donaciones'';
DECLARE @file NVARCHAR(400);
SET @file = @BackupFolder + @dbname + N''_LOG_'' + REPLACE(CONVERT(VARCHAR(19), GETDATE(), 120), '':'' ,'') + ''.trn'';
BACKUP LOG [ONG_Donaciones] TO DISK = @file WITH INIT, NAME = N''ONG_Donaciones-LOG'', STATS = 5;
RESTORE VERIFYONLY FROM DISK = @file;
';

IF EXISTS (SELECT 1 FROM msdb.dbo.sysjobs WHERE name = @job_name)
  EXEC msdb.dbo.sp_delete_job @job_name = @job_name, @delete_unused_schedule=1;

EXEC msdb.dbo.sp_add_job @job_name = @job_name, @enabled = 1, @description = 'LOG backup cada 30 min ONG_Donaciones', @owner_login_name = @owner_login;

EXEC msdb.dbo.sp_add_jobstep
  @job_name = @job_name,
  @step_name = N'Run Log Backup',
  @subsystem = N'TSQL',
  @command = @command,
  @retry_attempts = 2,
  @retry_interval = 5;

-- Horario cada 30 minutos -> freq_subday_type = 4 (minutos), freq_subday_interval = 30
EXEC msdb.dbo.sp_add_schedule @schedule_name = N'ONG_LOG_30min_Schedule',
    @freq_type = 4,
    @freq_interval = 1,
    @freq_subday_type = 4,
    @freq_subday_interval = 30,
    @active_start_time = 0;
EXEC msdb.dbo.sp_attach_schedule @job_name = @job_name, @schedule_name = N'ONG_LOG_30min_Schedule';
EXEC msdb.dbo.sp_add_jobserver @job_name = @job_name;
GO
```

4. Job de Limpieza Automática de Backups (Retention Policy)

Para evitar saturación del disco, se implementó un job que elimina automáticamente los archivos de respaldo con más de 14 días de antigüedad**.**

- Se ejecuta mediante un paso en PowerShell que:
  - Escanea los archivos *.bak* y *.trn*,
  - Evalúa su fecha de modificación,
  - Elimina los que superan el período de retención.
- Se ejecuta diariamente a las 3:30 AM**.**

Este mecanismo evita acumulación excesiva de datos y mantiene el sistema de respaldo ordenado.

```
-- Job CLEANUP
USE msdb;
GO
DECLARE @job_name NVARCHAR(128) = N'ONG_Backup_Cleanup';
IF EXISTS (SELECT 1 FROM msdb.dbo.sysjobs WHERE name = @job_name)
  EXEC msdb.dbo.sp_delete_job @job_name = @job_name, @delete_unused_schedule=1;

EXEC msdb.dbo.sp_add_job @job_name = @job_name, @enabled = 1, @description = 'Eliminar backups > 14 días', @owner_login_name = N'sa';

-- Comando de PowerShell limpia archivos viejos que hayan pasado los 14 días
DECLARE @ps NVARCHAR(MAX) = N'PowerShell -NoProfile -ExecutionPolicy Bypass -Command "Get-ChildItem -Path ''C:\Backups\ONG\'' -Include ''*.bak'',''*.trn'' -Recurse | Where-Object { $_.LastWriteTime -lt (Get-Date).AddDays(-14) } |

EXEC msdb.dbo.sp_add_jobstep
  @job_name = @job_name,
  @step_name = N'Cleanup Old Backups',
  @subsystem = N'PowerShell',
  @command = @ps,
  @retry_attempts = 1,
  @retry_interval = 5;

-- Horario diario a las 03:30
EXEC msdb.dbo.sp_add_schedule @schedule_name = N'ONG_Cleanup_Diario', @freq_type = 4, @freq_interval = 1, @active_start_time = 33000;
EXEC msdb.dbo.sp_attach_schedule @job_name = @job_name, @schedule_name = N'ONG_Cleanup_Diario';
EXEC msdb.dbo.sp_add_jobserver @job_name = @job_name;
GO

-- "Comprobación"
RESTORE VERIFYONLY FROM DISK = 'C:\Backups\ONG\ONG_Donaciones_FULL_20250725...bak';
RESTORE HEADERONLY FROM DISK = 'C:\Backups\ONG\ONG_Donaciones_FULL_20250725...bak';
```

Realización de consultas

1) Total donado por cada donante y ranking: Esta consulta obtiene el total donado por cada donante y además genera un ranking dentro de cada país según el monto total donado. Para lograrlo, utiliza funciones de ventana (SUM y RANK), las cuales permiten calcular totales y posiciones sin agrupar las filas.

   La consulta une las tablas Donante**,** País **y** Donación**,** suma todas las donaciones de cada donante y luego asigna un puesto (ranking) por país, ordenando de mayor a menor donación. Finalmente, muestra un listado organizado por país y por el lugar que ocupa cada donante dentro de ese país según lo que ha aportado.

```
WITH Totales AS (
    SELECT
        d.DonanteID,
        d.Nombre AS Donante,
        p.NombrePais,
        d.PaisID,
        SUM(o.Monto) AS TotalDonado
    FROM ong.Donante d
    JOIN ong.Pais p ON d.PaisID = p.PaisID
    JOIN ong.Donacion o ON o.DonanteID = d.DonanteID
    GROUP BY d.DonanteID, d.Nombre, p.NombrePais, d.PaisID
)
SELECT
    DonanteID,
    Donante,
    NombrePais,
    TotalDonado,
    RANK() OVER(
        PARTITION BY PaisID
        ORDER BY TotalDonado DESC
    ) AS RankingEnPais
FROM Totales
ORDER BY NombrePais, RankingEnPais;
GO
```

	DonanteID	Donante	NombrePais	TotalDonado	RankingEnPais
1	5	Luis Herrera	Costa Rica	0.00	1
2	3	Fundación Vida	El Salvador	2900.00	1
3	6	ONG Esperanza	El Salvador	1800.00	2
4	1	Carlos Méndez	El Salvador	300.00	3
5	2	Ana Torres	Guatemala	700.00	1
6	4	María Gómez	Honduras	1000.00	1

2) Total mensual acumulado de donaciones: La consulta muestra cada donación y calcula un total acumulado en el tiempo, sumando todas las donaciones anteriores y la actual según la fecha. Usa una función de ventana para obtener este acumulado sin agrupar las filas.

```
SELECT
    DonacionID,
    FechaDonacion,
    Monto,
    SUM(Monto) OVER(
        ORDER BY FechaDonacion
        ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW
    ) AS TotalAcumulado
FROM ong.Donacion
ORDER BY FechaDonacion;
GO
```

	DonacionID	FechaDonacion	Monto	TotalAcumulado
1	1	2025-11-26	150.00	150.00
2	2	2025-11-26	350.00	500.00
3	3	2025-11-26	1200.00	1700.00
4	4	2025-11-26	500.00	2200.00
5	5	2025-11-26	0.00	2200.00
6	6	2025-11-26	900.00	3100.00
7	7	2025-11-26	250.00	3350.00
8	8	2025-11-26	150.00	3500.00
9	9	2025-11-26	350.00	3850.00
10	10	2025-11-26	1200.00	5050.00
11	11	2025-11-26	500.00	5550.00
12	12	2025-11-26	0.00	5550.00
13	13	2025-11-26	900.00	6450.00
14	14	2025-11-26	250.00	6700.00

3) Porcentaje que recibe cada beneficiario dentro de una donación: Esta consulta muestra cómo se distribuye cada donación entre los beneficiarios. Para cada DonacionID calcula el total asignado utilizando una función de ventana y determina el porcentaje que representa el monto asignado a cada beneficiario dentro de esa misma donación. Esto permite identificar qué parte del aporte total recibió cada beneficiario, ordenado de mayor a menor porcentaje.

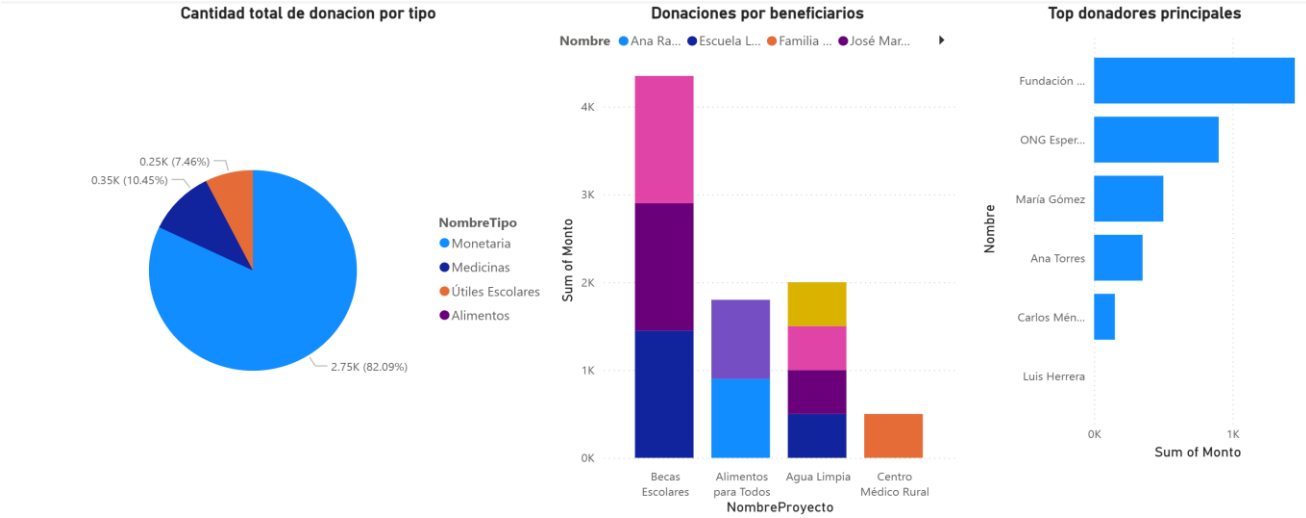
```
SELECT
    db.DonacionID,
    b.Nombre AS Beneficiario,
    db.MontoAsignado,

    SUM(db.MontoAsignado) OVER(
        PARTITION BY db.DonacionID
    ) AS TotalAsignado,

    CAST(db.MontoAsignado * 100.0 /
         SUM(db.MontoAsignado) OVER(PARTITION BY db.DonacionID)
    AS DECIMAL(5,2)) AS Porcentaje
FROM ong.DistribucionBeneficio db
JOIN ong.Beneficiario b ON b.BeneficiarioID = db.BeneficiarioID
ORDER BY db.DonacionID, Porcentaje DESC;
GO
```

	DonacionID	Beneficiario	MontoAsignado	TotalAsignado	Porcentaje
1	1	José Martínez	75.00	300.00	25.00
2	1	Lucía Sánchez	75.00	300.00	25.00
3	1	José Martínez	75.00	300.00	25.00
4	1	Lucía Sánchez	75.00	300.00	25.00
5	2	Pedro Arévalo	200.00	700.00	28.57
6	2	Pedro Arévalo	200.00	700.00	28.57
7	2	Escuela La Esperanza	150.00	700.00	21.43
8	2	Escuela La Esperanza	150.00	700.00	21.43
9	3	Escuela La Esperanza	1200.00	2400.00	50.00
10	3	Escuela La Esperanza	1200.00	2400.00	50.00
11	4	Familia Castillo	500.00	1000.00	50.00
12	4	Familia Castillo	500.00	1000.00	50.00
13	6	Ana Ramos	450.00	1800.00	25.00
14	6	Niño Martínez	450.00	1800.00	25.00
15	6	Ana Ramos	450.00	1800.00	25.00
16	6	Niño Martínez	450.00	1800.00	25.00
17	7	Lucía Sánchez	150.00	500.00	30.00
18	7	Lucía Sánchez	150.00	500.00	30.00
19	7	José Martínez	100.00	500.00	20.00
20	7	José Martínez	100.00	500.00	20.00

Dashboard de Donaciones con PowerBI: Distribución, Proyectos y Donantes



Nuestro Dashboard está basado principalmente en un análisis general de las donaciones registradas por la organización. Como primer punto se muestra la distribución de distintos tipos de donación en forma redonda, donde las aportaciones monetarias representan la mayor porción del total. Además, se permite identificar cuales reciben más apoyo y como se distribuyen los aportes entre los que donan y de qué forma.

También otro punto importante a resaltar es que incluye un ranking de donadores principales, lo que permite conocer los donadores y/o entidades que más contribuyen a los proyectos.

Para la elaboración del tablero, se importaron los datos de donaciones y se hizo la debida limpieza que permitio que se establecieran las relaciones entre las tablas correspondientes. Seguido a esto se crearon los graficos eligiendo los campos relevantes y aplicando los filtros para garantizar una presentación clara y concisa.