

Desarrollando con SQL2005

Ing. Jose Mariano Alvarez
Email: jose.mariano.alvarez@SqlTotalConsulting.com

Agenda

- CLR
- XML

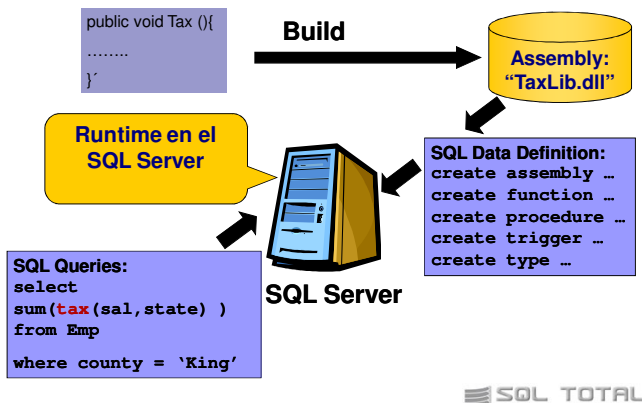
Temario

- Introducción al CLR
- Arquitectura e integración SQL-CLR
- Introducción al Modelo de proceso y de seguridad
- Introducción al desarrollando objetos
- Introducción al desarrollo de CLR stored Procedures
- Introducción al desarrollo de CLR Triggers
- Introducción al desarrollo de CLR User Defined Functions

Objetivos del SQL Server 2005

- Se tomaron en cuenta en el siguiente orden
 - Seguridad
 - Confiabilidad
 - Performance

Como funciona



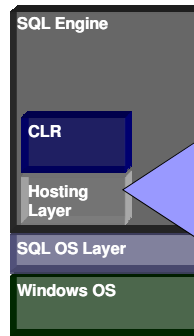
CLR Host y Assembly

- Cualquier proceso que carga el runtime .NET y corre el código en un entorno manejado
 - ASP.NET
 - Otros
 - Ahora el SQL Server
- El Assembly es la unidad de despliegue

Cambios en el CLR

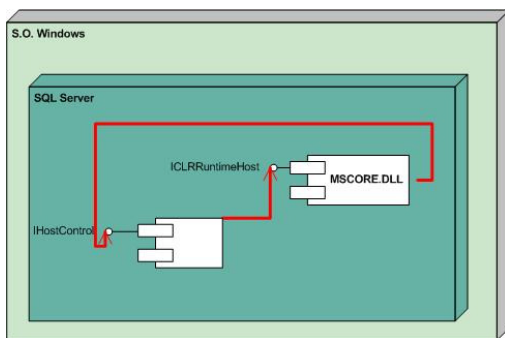
- .NET 1.1
 - Diseñado para IIS
 - Los procesos pueden ser matados (kill)
 - Se puede crear libremente threads (hilos de ejecución) y pedir memoria
- .NET 2.0
 - Diseñado para SQL Server 2005
 - Los procesos NO pueden ser matados
 - El Host puede reusarse a las peticiones de memoria o de nuevos threads

Integración del CLR



- Provee coordinación y control de:
 - Carga de Assemblies
 - Gestión de memoria
 - Modelo de seguridad
 - Confiabilidad
 - Threads y Fibers
 - Detección de Deadlock
 - Contexto de ejecución

Novedades del .NET 2.0



CLR: Gestión de memoria

- Todas las asignaciones de memoria del CLR se realizan a través del SQL Server
- La mayoría de la memoria del CLR (GC Heap) proviene de la memoria multi-page (fuera del Buffer-pool o "MemToLeave")
 - La opción **Max server memory** no cubre la memoria del CLR
- Si hay poca memoria disponible el SQL le pide al CLR que inicie el Garbage Collector

CLR: Gestión de memoria (2)

- DMV (dynamic management view) Para monitorear el uso de memoria del CLR
 - Sys.dm_os_memory_clerks
 - Sys.dm_os_memory_objects
- Perf counters: # GCs, Cantidad de memoria asignada
- SQL-OS gestión de memoria
 - <http://blogs.msdn.com/slavao/archive/2005/02/11/371063.aspx>

CPU: threads y sincronización

- Todos los threads "managed" están mapeados a tareas gestionadas por el scheduler del SQL Server
 - El Schedule del SQL es Cooperativo
 - Excepto: los threads del GC
- El SQL Scheduler detecta las tareas que no liberan (managed o T-SQL) y las castiga
 - Fuerza a la tarea a parar, la pone al final de la cola de proceso y le hace perder algunas vueltas

CPU: threads y sincronización

- Callback al SQL Server en P/Invoke
 - (cuando se llama código nativo)
 - Cuando cambia al modo pre-emptive
- Los pedidos de lock desde el CLR se realizan a través del SQL Server lock manager
 - Hay una detección unificada de deadlock entre los locks del CLR y del SQL nativo

Application Domain

- App-domains
 - Mecanismo del CLR para el aislamiento y la descarga de código
 - SQL Server usa dos clases de app-domains
 - DDL-time: temporario. Usado durante el DDL para verificación
 - Execution time: Cuando se ejecuta el código
- Granularidad
 - Uno por cada dueño de un assembly en cada base de datos
 - Todos los assemblies de un mismo dueño están en el mismo app domain

APPDomains

- No se permite la llamada entre distintos app domain
- DMVs de interés:
 - Sys.dm_clr_appdomains
- Los datos y el código que tienen como dueño a un usuario está aislado de los de los otros usuarios a menos que se habilite el acceso
- Se puede asignar permisos en el create assembly
- El dueño de una assembly puede asignar el permiso de referencia del mismo

Demo



Comparación T-SQL CLR

	T-SQL	CLR
User Defined Functions	X	X
Stored Procedures	X	X
Triggers	X	X
User Defined Types		X
Aggregates		X

Clases del framework disponibles

- La mayoría de las System.* incluyendo:
 - mscorlib.dll
 - system.dll
 - system.data.dll
 - system.xml.dll
 - system.security.dll
- No están disponibles las clases de uso interactivo como por ejemplo:
 - system.windows.forms.dll
 - system.drawing.dll
 - system.web.dll

Código Manajado

- No esta habilitado por default, se debe configurar al servidor para que lo pueda usar
- Se debe usar DML para cargar los assembly y registrar los objetos
- Solo pueden ser DLL
- Se puede cargar desde un archivo o de un bitstream
- Se requiere tener suficientes permisos
- Solo funciona con seguridad integrada

Habilitando el CLR

```
EXEC sp_configure 'show advanced options', 1
GO
RECONFIGURE
GO
sp_configure 'clr enabled', 1
GO
RECONFIGURE
GO
```

Gestión de un Assembly

- Registracion mediante la sentencia CREATE ASSEMBLY:

```
CREATE ASSEMBLY MyCLR
FROM 'C:\MyApp\MyCLR.DLL'
```

- Desregistración mediante la sentencia DROP ASSEMBLY statement:

```
DROP ASSEMBLY MyCLR
```

Metadata

- SYS.ASSEMBLIES
 - Propiedades de los Assembly
- SYS.ASSEMBLY_FILES
 - Binarios, código fuente, PDB, etc de los Assembly
- SYS.ASSEMBLY_REFERENCES
 - Dependencias de los Assembly
- Extensiones a la metadata actual
 - SYS.OBJECTS
 - SYS.ASSEMBLY_MODULES
 - SYS.ASSEMBLY_TYPES

Code Access Security (CAS)

- SAFE
 - Solo se tiene acceso al CLR
 - No se puede acceder a recursos externos, gestión de threads, código unsafe o interop
- EXTERNAL_ACCESS
 - Además se puede acceder a recursos externos al .NET Framework
 - Por ejemplo . EventLog, FileSystem y la red
 - Sin acceso a código unsafe o interop
- UNSAFE
 - Sin restricciones es similar a los extended stored procedures

Ejemplos de CAS

```
CREATE ASSEMBLY ExampleYukon
FROM 'd:\ExampleYukon.dll'
WITH PERMISSION_SET = SAFE
```

```
CREATE ASSEMBLY ExampleYukon
FROM 'd:\ExampleYukon.dll'
WITH PERMISSION_SET = EXTERNAL_ACCESS
```

```
CREATE ASSEMBLY ExampleYukon
FROM 'd:\ExampleYukon.dll'
WITH PERMISSION_SET = UNSAFE
```

Host Protection Attributes (HPAs)

- El CLR proporciona un mecanismo para marcar APIs manejadas que son parte del .NET con ciertas cualidades que puedan ser de interés a un host del CLR.
- Dadas estas cualidades, el host puede especificar una lista de HPAs que se deba deshabilitar en el ambiente. En este caso, el CLR negará los intentos del código del usuario en llamar APIs que son registradas por los HPAs en la lista de prohibidas.

SQL TOTAL

Conjuntos de seguridad

Permission set	Safe	External access	Unsafe
Code Access Security	Execute only	Execute + access to external resources	irrestringido
Programming model restrictions	SI	SI	Sin restricciones
Verifiability requirement	SI	SI	No
Ability to call native code	No	No	SI

SQL TOTAL

Guía de uso del CLR

SQL TOTAL

Cuando usar T-SQL

- Para acceder a datos
 - Es mejor
 - Es generalmente más rápido
 - Requiere escribir menos código
 - No se requiere aprender a programar en un lenguaje del CLR

SQL TOTAL

Cuando usar el CLR

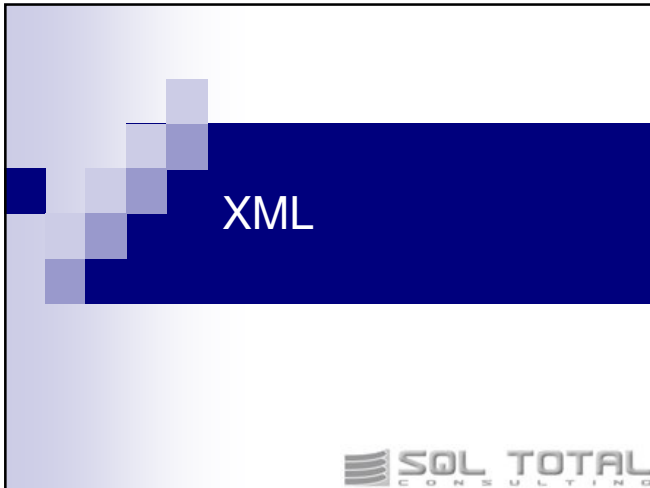
- Cuando hay funciones que requieren mucho cálculo es más rápido
- Para funciones escalares
- Funciones de agregación definidas por el usuario
- Para requerimientos similares a los cursores

SQL TOTAL

Mid Tier vs. Data Tier


- Tener el SQLCLR no significa mover toda la lógica de negocio al SQL Server
- Los mismos compromisos de siempre
 - Reducir el transporte de datos y los round trips vs. agregar lógica y carga al SQL Server
- Buenos candidatos
 - Validación de datos centralizada
 - Round-trips frecuentes
 - Procesar una gran cantidad de datos para obtener solo un resultado pequeño

SQL TOTAL




XML

- XML es un nuevo tipo de dato
- Índices sobre campos XML
- Esquemas XSD
- Consultas XQuery
- Vistas XML (SQLXML)
- Mejoras en FOR XML
- Mejoras en OPENXML



Limitaciones

- Solo los strings soportan cast a XML
- No puede ser usado en
 - GROUP BY
 - Distributed o Materialized Views
 - primary o foreign keys
 - Restricciones Unique
- Máximo 32 columnas XML por tabla
- Máximo 128 niveles de jerarquía




Esquemas

- Los campos XML pueden asociarse con esquemas


```
CREATE TABLE Invoices(
    id INT PRIMARY KEY,
    factura XML(EschemaFactura)
    ...
)

CREATE XML SCHEMA COLLECTION gencoll
'<xs:schema ...
targetNamespace=urn:gen>
...
</xs:schema>'
```



XQUERY


- Standard W3C
 - Con extensiones para actualización
- Basado en XPath
 - Mucha mayor riqueza para búsquedas complejas



XQuery

```
SELECT id, xDoc.query(
'for $s in /doc[id = 13]//sec[@num >= 2]
return <tipo>{data($s/cabeza)}</tipo>')
FROM docs
```

- xml.query: devuelve un tipo XML
- xml.exist: devuelve un booleano si hay resultado
- xml.value: devuelve un escalar
- xml.nodes: devuelve una tabla con una columna
- xml.modify: modifica el XML



Xquery - Accesos

- sql:variable: Variables T-SQL desde Xquery
- sql:column: Ccolumna a la que pertenece el XML

```
select detalleXML.query(  
'for $elem in /resumen/DatosPersonales  
  return  
    <Nombre>  
    { sql:column("Nombre") }  
    </Nombre> '  
) from DatosDetalle
```

Indices XML

- Pueden definirse índices en columnas XML
 - Aceleran las sentencias XQuery
- Varios tipos
 - Atributos
 - Valores
 - XPath