SQL Server 2008 の新機能をチェック

マイクロソフト株式会社
ジニアス平井

アジェンダ

- MERGE ステートメント
- ユーザー定義テーブル型
- ロックエスカレーション
- Date & Time 型
- 階層データ型（hierarchyid）
- FILESTREAM ストレージ
- 空間データ型 （geometry, geography）
- JIS 2004 対応

MERGE ステートメント

- **単一ステートメントで INSERT、UPDATE、DELETE 操作を実行**
  - ANSI SQL 2006 準拠
  - ユーザー定義テーブル型と組み合わせれば明細テーブルへの複数行の更新が容易
    - IF NOT EXISTS(SELECT ....) はもういらないかも

```
MERGE Table1
  USING (SELECT 1 AS F1, 'AAA' AS F2) AS Foo
  ON (Table1.F1 = Foo.F1)
WHEN MATCHED AND Table1.F2 = 'XXX' THEN
  DELETE
WHEN MATCHED THEN
  UPDATE F1 = Foo.F1 F2 = 'XXX'
WHEN NOT MATCHED THEN
  INSERT VALUES (Foo.F1, Foo.F2)
```

動き

① 1回目は行が追加される

② 2回目で "AAA"が "XXX" に更新される

③ 3回目は行が削除される

ユーザー定義テーブル型

- **特長**
  - CREATE TYPE ステートメントで定義
  - ストアドプロシージャや関数内でテーブル値の入力パラメータとして使用
    - 配列(コレクション)のようなデータの受け渡しに最適
  - テーブル値は tempdb に置かれる
- **シナリオ**
  - １つの注文ヘッダー情報に対する n 個の注文明細行

```
-- 定義
CREATE TYPE MyMember AS table(
  ID int IDENTITY(1, 1),
  Name nvarchar(20))
GO
--ストアドプロシージャ
CREATE PROC proc1
  @m MyMember READONLY
AS
  SELECT * FROM @m
GO
--実行
DECLARE @tb MyMember
INSERT INTO @tb
  VALUES (N'ジニアス'), (N'平井昌人')
EXECUTE proc1 @tb
```

ユーザー定義テーブル型 + MERGE

- **件数が不特定の明細データなどの CUD に威力を発揮**
  - ユーザーテーブル型をパラメータとして受け取り、MERGEステートメントで一括更新するストアドプロシージャを定義
  - DataTable にレコードをセットしてパラメータに与える

```
CREATE PROCEDURE sp_testCUD
    @tb workTB READONLY
AS
MERGE INTO 明細 AS masterTB
  USING (SELECT * FROM @tb) AS workTB
  ON masterTB.F1 = workTB.F1
-- 両方にあれば更新
WHEN MATCHED THEN
  UPDATE SET F2 = workTB.F2
-- マスターになければ挿入
WHEN NOT MATCHED BY TARGET THEN
  INSERT VALUES (workTB.F1, workTB.F2)
-- ワークTBからなくなっていれば削除
WHEN NOT MATCHED BY SOURCE THEN
  DELETE;
```

```
//明細レコードの取得
var adp = new SqlDataAdapter("SELECT * FROM
明細", cn);
var tb = new DataTable();
adp.Fill(localTable);
this.dataGridView1.DataSource = localTable;

//明細レコードをまとめて更新
var cmd = new SqlCommand("sp_testCUD", cn);
cmd.CommandType =
    CommandType.StoredProcedure;
cmd.Parameters.Add("@tb",
                   SqlDbType.Structured);
cmd.Parameters[0].Value = localTable;
cmd.ExecuteNonQuery();
```

ロックエスカレーションの無効化

- **LOCK_ESCALATION オプション**
  - ロックエスカレーション
    - 行（既定）→ ページ → エクステント → テーブル
    - メモリや範囲、同時トランザクションなどによって変化
  - テーブル単位で設定可能
    - 以前は SQL Server 全体（トレースフラグ 1211）
    - ALTER TABLE [座席予約]
      SET (**LOCK_ESCALATION = DISABLE**)
- **ロックの粒度**

行（RID）	行ロック
キー（KEY）	インデックスの行ロック
ページ（PAG）	8KB のページ
エクステント（EXT）	連続した 8KB ブロック（64KB）
テーブル（TAB）	テーブル全体

- **ロックヒント**
  - 指定した粒度でロックさせる
    - UPDATE 給与マスター **WITH (TABLOCK)** SET 給与 = 給与 * 0.9

Date 型と Time 型

- **ポイント**
  - 時刻と日付の分離
  - 検索条件や比較演算がシンプルになる

データ型	形式	範囲	精度	バイト数
smalldatetime	YYYY-MM-DD hh:mm:ss	1900-01-01 ～ 2079-06-06	1 分	4
datetime	YYYY-MM-DD hh:mm:ss [.nnn]	1753-01-01 ～ 9999-12-31	0.333 秒	8
datetime2	YYYY-MM-DD hh:mm:ss [.nnnnnnn]	0001-01-01 00:00:00.0000000 ～ 9999-12-31 23:59:59.9999999	100 ナノ秒	6 ～ 8
date	YYYY-MM-DD	00001-01-01 ～ 9999-12-31	1 日	3
time	hh:mm:ss [.nnnnnnn]	00:00:00.0000000 ～ 23:59:59.9999999	100 ナノ秒	3 ～ 5
datetimeoffset	YYYY-MM-DD hh:mm:ss [.nnnnnnn] [+/-]hh:mm	00001-01-01 00:00:00.0000000 ～ 9999-12-31 23:59:59.9999999 (UTC)	100 ナノ秒	8 ～ 10

階層データ型（hierarchyid）

- **階層データ構造をエレガントに表現**
  - 可変長のシステム データ型
    - SQL CLR型（.NETの構造体）
  - 階層内の位置を表現する
  - さまざまな階層構造に適用
    - 組織構造
    - 部品展開表
    - コンテンツ管理
    - メーリングリスト・掲示板
  - 任意の挿入・削除をサポート
  - 効率的なストレージ利用
  - ツリーごとまとめて移動も操作できる
  - T-SQL、CLR でのプログラミング
    - 再帰コールをせずに TreeView にセットできる

	階層構造	階層	社員番号	社員名	管理者番号	レベル
1	/	0x	1	木村	NULL	0
2	/2/	0x68	2	亀井	1	1
3	/3/	0x78	3	鈴木	1	1
4	/4/	0x84	4	宮川	1	1
5	/4/10/	0x86A8	10	江口	4	2
6	/3/7/	0x7CE0	7	高橋	3	2
7	/3/8/	0x7D10	8	河合	3	2
8	/3/9/	0x7D30	9	荒木	3	2
9	/3/7/11/	0x7C...	11	中村	7	3
10	/3/7/12/	0x7C...	12	岡野	7	3
11	/3/7/13/	0x7C...	13	西	7	3
12	/3/7/1...	0x7C...	14	飯室	11	4
13	/2/5/	0x6C60	5	今野	2	2
14	/2/6/	0x6CA0	6	上村	2	2

hierarchyid データ型のプログラミング

- **Microsoft.SqlServer.Types 名前空間**
  - ¥MSSQL¥Binn¥Microsoft.SqlServer.Types.dll

メソッド名	機能	T-SQL	CLR
GetAncestor	N 番目の親の hierarchyid を返す	○	○
GetDescendant	子ノードの hierarchyid を返す	○	○
GetLevel	現在のノードの深さ(smailint)を返す	○	○
GetRoot	ルートの hierarychyid を返す静的メソッド	○	○
IsDescendant	現在のノードが子であるか判定する(bool)	○	○
Parse	引数を hierarchyid に変換する静的メソッド	○	○
Read	BinaryStream から SqlHierarcyId に読み込む	×	○
Reparent	現在のノードの親を新しい親ノードに変更する	○	○
ToString	hierarchyid 型から文字列型に変換する	○	○
Write	BinaryStream に SqlHierarchyId のバイナリデータを書き込む	×	○

FILESTREAM ストレージ

- **非構造化データ管理の問題**
  - ドキュメント、ビデオなどのデータは、DB外で管理される
  - データーベース内に置くと機能やパフォーマンスに悪影響
- **FILESTREAM ストレージ**
  - データベース エンジンと NTFS ファイル システムの統合
  - SQL Server の標準機能でバックアップと復元が可能
  - テーブルの varbinary(max) 列に FILESTREAM 属性を追加
  - テーブルには UNIQUEIDENTIFIER 列が必要
- **制約事項**
  - NTFS ファイルシステムでのみサポート
  - varbinary(max) 型のみ FILESTORAGE オプションが使える
  - 透過的な暗号化はサポートしない
  - ログ配布では使用できるが データベース ミラーリングでは使用できない

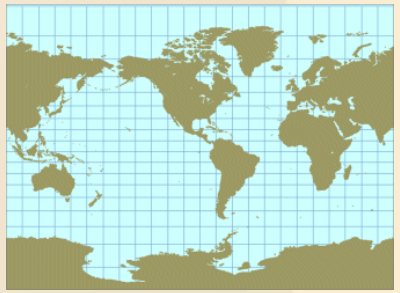
FILESTREAM 使用例

```
-- 有効化
EXEC sp_configure 'filestream_access_level', '2'
reconfigure
-- データベースの作成
CREATE DATABASE BLOB ON PRIMARY ( NAME = BLOB, FILENAME = 'C:¥SQLWork¥BLOB.mdf'),
  FILEGROUP BLOB_STORAGE CONTAINS FILESTREAM(NAME = BLOB_STORAGE,
                                             FILENAME = 'c:¥SQL2008')
GO
-- テーブルの作成
USE BLOG
CREATE TABLE ImageBank(
  ID uniqueidentifier ROWGUIDCOL NOT NULL UNIQUE,
  Picture varbinary(MAX) FILESTREAM NULL)
```

```
Dim cmd As New SqlCommand("INSERT INTO ImageBank VALUES (newid(), @p)", cn)
cmd.Parameters.Add("@p", SqlDbType.VarBinary)
Dim buf As New SqlTypes.SqlBytes()
Dim fs As New FileStream(PictureBox1.ImageLocation, FileMode.Open, FileAccess.Read)
buf.Stream = fs
cmd.Parameters("@p").Value = buf
cmd.ExecuteNonQuery()
```

空間データ型（geometry, geography）

- **特長**
  - 平面モデルと測地モデルに対応
    - Geography Markup Language (GML)を使用した空間情報の交換
  - 空間インデックスによるパフォーマンス最適化
  - .NET アプリケーションやVirtual Earthと連携して視覚化
- **geometry 型**
  - ユークリッド座標系内の幾何データをサポートする平面空間データ型
    - SQL CLR型（.NETの構造体）
    - Open Geospatial Consortium (OGC) Simple Features for SQL Specification version 1.1.0 準拠



- **geography 型**
  - GPS の緯度・経度座標などの楕円体データを格納するデータ型
    - SQL CLR型（.NETの構造体）

空間データの処理

- **空間データ オブジェクト**（インスタンス）
  - Point、LineString、Polygon、MultiLineString、MultiPolygon

```
INSERT INTO SpatialTable
  VALUES (geometry::STGeomFromText('POLYGON(0 0, 2 0, 2 2, 0 2, 0 0)', 0) )
```

- **プロパティとメソッド**
  - OGC 準拠のプロパティとメソッド
  - インスタンスメソッドは[.]を使用
  - 静的メソッドは[::]を使用
  - 大文字小文字が区別される

STGeomFromText（文字列から実体化）
STArea　（多角形の表面積）
STLength（長さ）
STCentroid（中心位置）
STIntersects（重なり部分の位置）
STDistance（距離）
STGeometryN（コレクション）
STPointN（コレクション）

- **Virtual Earth**
  - http://dev.live.com/virtualearth/

空間データの操作

- T-SQL （CLR は Microsoft.SqlServer.Types 名前空間）

```
--- 面積を求める
DECLARE @g geometry
SET @g = geometry::STGeomFromText('POLYGON((0 0, 10 0, 10 10, 0 0))', 0)
SELECT @g.STArea()
--- ２直線の交差する点を求める
DECLARE @g1 geometry, @g2 geometry, @r geometry;
SET @g1 = geometry::STGeomFromText('LINESTRING (0 0, 100 100)', 0)
SET @g2 = geometry::STGeomFromText('LINESTRING (0 100, 100 0 )', 0)
SELECT @r = @g1.STIntersection(@g2)
SELECT @r.STAsText()
--- 各事業所どうしの距離を求める
SELECT A.事業所, B.事業所, A.位置.STDistance(B.位置)
FROM CompanyList A, CompanyList B
WHERE A.事業所 > B.事業所
ORDER BY 3
```

JIS2004 対応

- **Japanese_XJIS_100**
  - JIS2004 に対応した照合順序
  - この照合順序を利用することでサロゲート ペアに対する文字列比較や LIKE 演算が正しく動作
  - ただし文字列操作関数はサロゲートを 4 バイトと判断
  - 旧バージョンとの互換が必要な場合は "Japanese_90"
- **データ型の選択**
  - JIS2004 で拡張されたサロゲート ペアを SQL Server で扱う
  - **n**char、**n**varchar、**n**varchar(max)
- **注意する文字の例**
  - サロゲート（4 バイト文字） ： **鰄**（トビウオ）・**叱**る（シカる）
  - Shift-JIS 非互換：  （ユキ）・  （オンセン）・**倶**楽部（クラブ）
  - Unicode 制御文字： ZWJ（Zero width joiner）, LRM（Left-to-right mark）
  - 結合文字： がぎぐげご（鼻濁音）ゼ ヅ ド （アイヌ語）

Reference

- **SQL Server 2008 Web サイト**
  - 製品の概要はもちろんのこと、データシートなどの各種資料をダウンロードすることができます
  - http://www.microsoft.com/japan/sqlserver/2008/default.mspx
- **SQL Server 2008 日本語評価ガイド「自習書シリーズ」**
  - SQL Server 2008 の新機能を集中的にステップ バイ ステップ形式で習得できるようになっています
  - 非常に出来のいいコンテンツです！
    - SQL Server 2008 の注目の新機能をイチ早く試してみよう!
    - SQL Server 2008 Reporting Services 入門編
    - SQL Server 2008 Analysis Services 入門編
    - SQL Server 2008 Integration Services 入門編
  - http://www.microsoft.com/japan/sqlserver/2008/self-learning/default.mspx

ぜひ、ご評価ください

- **Microsoft Windows Server 2008 Enterprise**
  - 検証には Windows Server 2008 x64 の Hyper-V でゲストに Windows Server 2008 x86 / x64 を入れて利用すると便利です
  - http://msdn.microsoft.com/ja-jp/evalcenter/cc137233.aspx
- **Visual Studio Team System 2008 Team Suite**
  - http://msdn.microsoft.com/ja-jp/evalcenter/bb655862.aspx
- **SQL Server 2008 Developer Edition**
  - x86, x64, IA64 があります
  - SQL Server 2005との共存も可能です（マルチインスタンス）
  - http://msdn.microsoft.com/ja-jp/bb851668.aspx
- **Visual Studio 2008 Service Pack 1**
  - ADO.NET Entity Framework や Visual Studio 2008 で SQL Server 2008 へアクセスするデザイナやウィザードが必要とします
  - http://www.microsoft.com/downloads/details.aspx?FamilyId=FBEE1648-7106-44A7-9649-6D9F6D58056E&displaylang=ja

Thanks for Attending !

お疲れ様でした。

SQL Server 2008 をよろしくお願いします。

Genius Hirai Presents