Synchronisatie van data

IN COMBINATIE MET SQL SERVER 2008 GOEDE STEUN VOOR ONTWIKKELAAR

Om synchroniseren aanmerkelijk eenvoudiger te maken, heeft Microsoft de ADO.NET Sync Services uitgebracht. Hiervan werd een eerste versie geleverd bij Visual Studio 2008 en deze krijgt een update voor SQL Server 2008 in VS2008-SP1. In dit artikel maakt auteur Mark Giesen hiermee een eenvoudige applicatie. Hij laat zien dat synchroniseren van data geen probleem meer hoeft te zijn.

Bedenk eens of je nu kan opstaan en in staat bent om direct een netwerkverbinding naar je bedrijfsnetwerk op te starten. Bijvoorbeeld om de urenregistratie bij te werken of om je e-mail op te halen. Dat gaat dan via de kabel, ADSL, WiFi, WiMax, GPRS of UMTS. Erg handig natuurlijk en we beschouwen het bijna als standaard. Maar als een internetprovider een dag problemen met het netwerk heeft, is dat landelijk nieuws en regent het boze klachten. Valt de verbinding een keertje weg, dan vormt dat ook meteen een gróót probleem. Hele bedrijven komen stil te liggen en niemand kan meer aan de slag. Het enige dat je nog kan doen, is achterstallige mail in Outlook lezen. Dat dit nog wel werkt, komt doordat Outlook gebouwd is als Occasionally Connected System (OCS). Het betekent dat Outlook niet altijd verbonden hoeft te zijn met het netwerk. Erg handig voor die paar momenten zonder netwerk.

Waarom werkt niet méér software op deze manier? De workitem tracking-software bijvoorbeeld, zodat je kan kijken wat er nog te doen valt als het netwerk weg is. Of het incidentregistratiesysteem van de helpdesk, zodat men de netwerkstoring kan registreren. Dit komt omdat het nogal lastig is zo'n systeem te bouwen. Op het moment dat er wel een connectie is, moet de applicatie gesynchroniseerd worden met de server en daar komt veel bij kijken:

- *Nieuwe gegevens*
  Om nieuwe gegevens op de server te achterhalen, moet je van ieder gegeven weten wanneer het is aangemaakt. Ieder record in de database moet dus een AanmaakDatumtijd hebben.
- *Gewijzigde gegevens*
  Om wijzigingen te achterhalen moet je van ieder record ook de MutatieDatumtijd bewaren.
- *Verwijderde gegevens*
  Van verwijderde gegevens kunnen we geen datum bij het record opslaan. Dit record is immers weg! Hiervoor moet je een aparte tabel aanmaken met daarin minimaal de primary key en een VerwijderDatumtijd. Deze tabel wordt een tombstone-tabel genoemd.
- *Wanneer*
  Je moet ook weten wanneer voor het laatst gesynchroniseerd is om te achterhalen vanaf wanneer het systeem de wijzigingen moet ophalen.
- *Triggers*
  Er moeten triggers aangemaakt worden om de Aanmaak- en Mutatiedatumtijd bij te werken en om de tombstone-tabel te vullen.
- *Conflicten*
  Wat als je op de client eenzelfde record hebt toegevoegd als op de server? Hoe detecteren we conflicten en wat doen we er dan mee?

Om synchroniseren aanmerkelijk eenvoudiger te maken, heeft Microsoft de ADO.NET Sync Services uitgebracht. Hiervan werd een eerste versie geleverd bij Visual Studio 2008 en deze wordt geüpdate voor SQL Server 2008 in VS2008-SP1. In dit artikel maken we hiermee een eenvoudige applicatie. Ik raad je aan de stappen te volgen en mee te bouwen. Als je echter lekker op de bank ligt en niet mee wilt bouwen, kun je de Step by Step-hoofdstukken overslaan.

Allereerst zal deze demo worden gemaakt op een SQL 2005-database om te laten zien wat er gebeurt en dat het goed bruikbaar is in bestaande applicaties. Vervolgens maken we dezelfde demo op een SQL 2008-database om de verschillen en voordelen te tonen. Aan de client-kant gebruiken we steeds een SQL Server Compact Edition (SQLCE) 3.5-database. Het gaat om een zeer lightweightversie van SQL Server. Voor deze database hoeft niets extra op de client aanwezig te zijn. Alles wat nodig is, zit binnen het project.

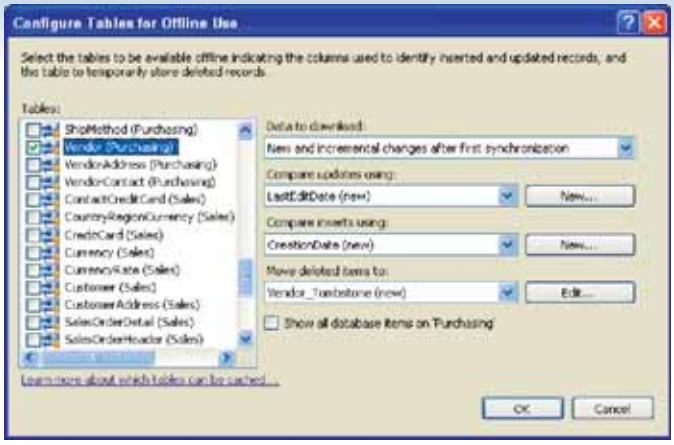
Voor de demo's is VS2008-SP1, SQL 2005 SP2 en SQL 2008 RTM gebruikt. Als database hanteren we de AdventureWorks-database.

Sync met SQL 2005

Step by Step

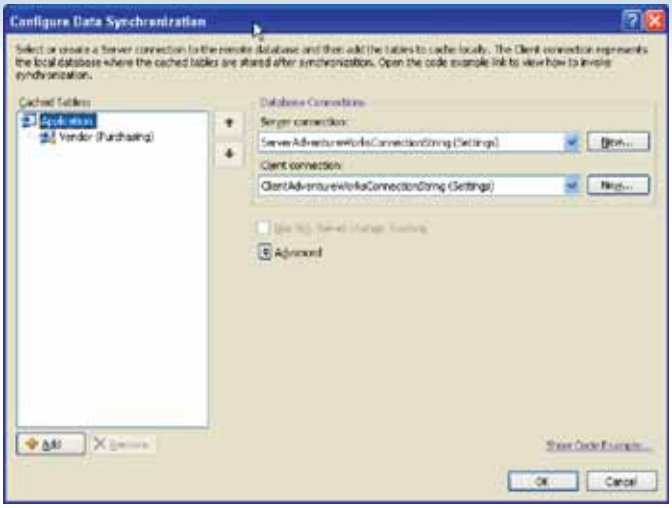
Open een nieuw Windows Form Application Project (demo is in C#). Noem het project Sync2005 en de solution SyncDemos. Rechtsklik op het project en kies Add – New Item. Selecteer Local Database Cache en noem die AW-2005Cache.sync en klik Add. Op dit moment wordt de Data Synchronization wizard gestart. Selecteer (of maak) op het openingsscherm de connectie naar de Server. Op het moment dat je de Server-connectie hebt gekozen zul je zien dat de lijst met Cached tables nu "loading" laat zien. De lijst met tabellen van de server wordt opgehaald. De Client-connectie wordt automatisch gevuld met een SQLCE file die nieuw voor je wordt aangemaakt.

Klik nu op de Add-knop. Je krijgt een lijst met tabellen in de serverdatabase te zien. Het kan zijn dat niet alle tabellen hier zichtbaar zijn. Tabellen zonder primary key, met veldtypes die niet door SQLCE worden ondersteund of met namen langer dan 118 karakters, vallen af. Selecteer hier de Vendor-tabel. Je kunt voor deze tabel nu aangeven welke velden gebruikt moeten worden als AanmaakDatumtijd en MutatieDatumtijd en de naam van de tombstone-tabel. De Vendor-tabel heeft nog geen Datumtijdvelden die automatisch worden bijgehouden en geen tombstone-tabel. Dus kies overal dat die nieuw moeten worden aangemaakt. Laat de bovenste optie staan op "New and incremental changes …" (zie figuur 1).



FIGUUR 1

Klik vervolgens op OK om dit scherm af te sluiten en nogmaals OK om de wizard af te sluiten. Je krijgt nu een waarschuwing die je vertelt dat de wizard van plan is de serverdatabase aan te passen. Laat beide vinkjes aan staan en klik OK. Heb je de rechten straks in de praktijk niet, dan zet je het eerste vinkje uit om de aanpassingen nog niet uit te voeren. VS2008 zal nu automatisch de Data Source Configuration Wizard starten, zodat je ook typed datasets voor je project kan aanmaken. Kies hier de velden VendorID, Name, CreditRating en PurchasingWebServiceURL. Noem de dataset AW2005DataSet en klik op Finish. Als je geen gebruik wil maken van typed datasets, maar bijvoorbeeld het Entity Framework, moet je in deze wizard op cancel klikken en het verder zelf regelen.



FIGUUR 2

Resultaat

Als je op dit moment je project bekijkt, zul je zien dat er veel aan is toegevoegd. Ten eerste referenties. Een aantal daarvan zijn voor synchronisatie specifiek. Er is ook een referentie toegevoegd voor de SQL Server CE-database. Dit is alles wat je nodig hebt om met die database te kunnen werken. Verder vind je twee mapjes met scripts. In de eerste staan de SQL Scripts voor het aanpassen van je database. In de praktijk zul je soms geen rechten hebben om deze aanpassingen uit te voeren en in die gevallen heb je de wizard dit script niet laten uitvoeren. Je kunt dan het aanmaakscript naar je DBA mailen zodat deze het kan uitvoeren. In het tweede mapje vind je scripts om deze aanpassingen weer ongedaan te maken. Van beide scripts vind je er één per tabel. Ook zie je de SQL Server CE-database binnen je project en een sync-bestand dat de metadata bevat over de synchronisatie. Als je hierop dubbelklikt, start de wizard weer en kun je wijzigingen aanbrengen. Vervolgens zie je de dataset die je hebt aangemaakt en tot slot een app.config met daarin ConnectionStrings voor de server en de client. Als je nu de wizard opnieuw start, zie je dat deze connectionstrings gebruikt worden in de wizard (zie figuur 2).

Step by Step

Nu wordt een Winform gemaakt waarop we de data tonen. Open het design window van Form1 en open de Data Sources Explorer (via het Data-menu). Je treft hier nu de dataset. Selecteer Purchasing_Vendor, je ziet dat dit een combo wordt. Open deze en selecteer: "Details". Selecteer zo ook None voor het veld VendorID en NumericUpDown voor CreditRating. Sleep nu Purchasing_Vendor op het Form. De controls voor de velden zullen nu aangemaakt worden, evenals een toolbar voor navigatie.

Start de applicatie en controleer of je een werkende applicatie hebt om door de Vendors te bladeren.

De applicatie synchroniseert op dit moment echter nog niet. De wizard heeft alleen eenmalig de data van de server opgehaald. Om synchronisatie toe te voegen, sluit je de applicatie af en ga je weer naar het Form Design. Klik daar op de toolbar en kies in de verschenen combo voor een Button. Dubbelklik deze button om in de click method van deze button te komen. Hier wordt nu de Synchronisatiecode geplaatst. Ook hier helpt de wizard weer. Open die nog maar eens en zie dat er rechts onderin een linkje staat: "Show Code Example…" (zie ook figuur 2). Klik op dit linkje, vervolgens op "Copy code to the Clipboard" en tot slot op close en cancel. Plak de code van het clipboard nu in de click method. De eerste twee regels code zijn al netjes op maat gemaakt voor je situatie en deze

twee regels zorgen voor de daadwerkelijke synchronisatie tussen beide databases. Wat je nu alleen nog moet doen is na de synchronisatie de data binnen de applicatie verversen. Voeg daartoe de volgende code toe aan de click method, op de plaats waar TODO staat.

```
purchasing_VendorTableAdapter.Fill
    (aW2005DataSet.Purchasing_Vendor);
```

Resultaat

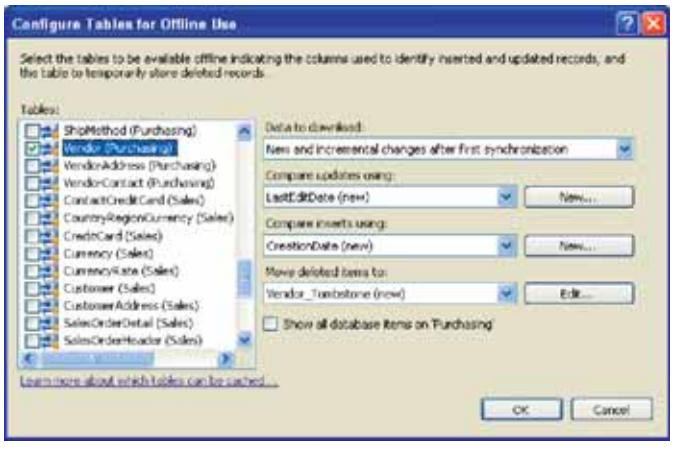
Start de applicatie en start ook de SQL Server Management Studio (SSMS) en open daar de Vendor-tabel. Pas in SSMS de CreditRating van een vendor aan en blader in de applicatie naar die vendor. Je ziet daar nu de oude waarde nog staan. Klik op de sync button en daar verschijnt de nieuwe waarde. Je hebt nu via een wizard, één keer Copy/Paste en één regel code typen, een applicatie gemaakt die zichzelf kan synchroniseren met een server. Er is nog veel meer mogelijk, maar ga eerst deze zelfde oefening met een SQL 2008 database doen.

Sync met SQL 2008

Het lijkt alsof het niet beter kan. SQL 2008 biedt echter een aantal features die het synchroniseren nog beter maken. Zo is het opvragen van de laatst gebruikte timestamp en rowversion aangepast, zodat dit beter overweg kan met transacties. Hierdoor is synchronisatie vooral robuuster geworden. Daarnaast noemen we het nieuwe commando MERGE. Hiermee kun je inserts, updates en deletes in één keer verwerken. Dat scheelt mogelijk roundtrips. De belangrijkste nieuwe feature is echter Change Tracking. Als je dit aanzet, houdt SQL Server 2008 de wijzigingen bij die op de data plaatsvinden. Je kunt vervolgens eenvoudig deze wijzigingen opvragen zonder dat je zelf datumtijd-velden of tombstone-tabellen moet aanmaken en beheren. Ook Triggers voor de datumtijd-velden behoren daarmee tot het verleden. De performance-impact die het aanzetten heeft, komt ongeveer overeen met het op een tabel zetten van een tweede index. Vele malen minder in ieder geval dan drie triggers aan een tabel toevoegen. Het belangrijkste voordeel is misschien wel dat je de tabellen niet hoeft aan te passen. Een DBA is veel eenvoudiger te overtuigen om deze stap te zetten en bij databases die je niet in eigen beheer hebt, kun je nu ook ineens synchronisatie gaan toepassen. Laten we eens kijken wat de wizard in zo'n situatie voor ons aanmaakt.

Step by Step

Rechtsklik op de solution en kies Add > New project… Selecteer weer een Windows Forms Application, noem deze Sync2008 en klik OK.



FIGUUR 3

Voeg opnieuw een Local Database Cache toe en noem die AW2008Cache. Selecteer (of maak) in de wizard nu een connectie naar de SQL 2008 database. Je ziet dat de optie "Use SQL Server Change Tracking" bruikbaar wordt en default aan is gezet. Klik op Add en kies weer de tabel Vendor. De keuzes voor datumtijd-velden en tombstone-tabellen zijn nu geshadowed (zie figuur 3).

Deze velden hebben we niet meer nodig, dus blijven ze hier achterwege. Klik twee keer op OK. Maak vervolgens weer een dataset aan (VenderID heet BusinessEntityID in de AW2008-database) en noem die AW2008DataSet. Vul ten slotte het form op dezelfde manier. Voeg de button toe en Copy/Paste de synchronisatiecode weer in de clickmethod. Voeg dit keer de volgende code toe:

```
purchasing_VendorTableAdapter.Fill
    (aW2008DataSet.Purchasing_Vendor);
```

Start de applicatie en wijzig via SSMS een CreditRating in de SQL 2008-database. Controleer of de applicatie net zo synchroniseert als de 2005-variant.

Change Tracking in SQLCE

Bekijk de SQLCE-database eens in SSMS of in je Server Explorer binnen VS2008. Net als SQL 2008 zijn er in SQL CE geen tombstone-tabel of extra velden toegevoegd. Triggers zijn niet mogelijk, dus ook die ontbreken. Hoe houdt SQL CE dan zijn wijzigingen bij? Net als SQL 2008 met Change Tracking. Op het moment van de eerste synchronisatie wordt Change Tracking aangezet binnen een SQL CE-database, daar hoef je niets voor te doen.

Een stapje verder…

De tot nu toe gemaakte demo is al zeer bruikbaar in de praktijk. Maar er kan nog meer.

- *Opschonen tombstone*
  In het SQLScripts-mapje van Sync2008 vind je twee script files. Eén voor de database en één voor de tabel Vendor. In deze scripts zie je geen schemawijzigingen meer, maar propertywijzigingen. Als je in SSMS naar de properties van de database kijkt, vind je die properties terug op de pagina Change Tracking. Je ziet ook dat je nog meer instellingsmogelijkheden hebt. Je kunt hier aangeven hoe lang je de wijzigingen van de data wil bewaren. Als je dit opschonen ook bij de 2005-variant wil, moet je daarvoor zelf iets bouwen.
- *Bi-Directioneel*
  Van server naar client synchroniseren is erg handig en zal soms genoeg zijn. In veel applicaties is het zinvol om referentiedata op deze manier lokaal te bewaren en bijvoorbeeld tijdens het opstarten te synchroniseren. Hiermee worden de servers ontlast en die doen dan efficiënter hun werk. De snelle clients van tegenwoordig zijn met de meeste applicaties zeker niet druk bezet en deze zijn daarom inzetbaar om de last te delen. Belangrijker is mogelijk dat de applicaties hierdoor sneller kunnen worden.

```
Purchasing_Vendor.SyncDirection =
    Microsoft.Synchronization.Data.SyncDirection.Bidirectional;
```

```
MessageBox.Show("updates naar server:" +
    syncStats.TotalChangesUploaded.ToString() +
    Environment.NewLine + "updates naar client:" +
    syncStats.TotalChangesDownloaded.ToString());
```

Adv

In andere applicaties moet ook van client naar server gesynchroniseerd worden. Iedere tabel heeft een SyncDirection-property. Die kun je door de wizard op DownloadOnly of Snapshot laten zetten via de Data to download combo (zie figuren 2 en 3). Deze kun je ook op Bidirectional zetten, maar (nog) niet via de wizard. Om dit te doen, klik je rechts op AW2008Cache.sync en kiest View Code. Plaats hier de volgende code in de OnInitialized method:

Meer hoef je niet te doen. Ter controle of het ook twee kanten op gaat, voegen we nog een regel toe aan de click method. Voeg daar na de synchronisatie de volgende code toe:

Start de applicatie, pas in de client het eerste Vendor-record aan en klik op save. Pas vervolgens op de server het tweede record aan via SSMS. Klik nu op de synchronize-knop. Nu zie je dat er één record naar de server is gestuurd en één terug is gekomen naar de client.
Dit Bi-Directioneel synchroniseren kun je ook in de SQL 2005-variant inbouwen, de code is gelijk. Het gedrag is echter iets anders. Als je daar alleen aan de client een wijzing doorvoert en dan synchroniseert zul je bij beide in de messagebox toch één record zien staan. Dit komt doordat de client eerst de eigen updates verstuurt en vervolgens de updates van de server ophaalt. Bij het ophalen krijgt de client zijn eigen record weer terug. Die is immers sinds de vorige keer synchroniseren veranderd (namelijk zojuist bij de update). Het werkt, maar is natuurlijk niet zo efficiënt. Hier brengt SQL 2008 nog een voordeel. Deze houdt namelijk bij welke client aan het synchroniseren is en stuurt in diezelfde actie niet dezelfde records terug.

```
using Microsoft.Synchronization.Data;
using System.Windows.Forms;

namespace Sync2008
{
    public partial class AW2008CacheSyncAgent
    {
      partial void OnInitialized()
      {
        Purchasing_Vendor.SyncDirection = SyncDirection.Bidirec-
        tional;
      }
    }

    public partial class AW2008CacheServerSyncProvider
    {
      partial void OnInitialized()
      {
        this.ApplyChangeFailed += new System.EventHandler<ApplyCh
        angeFailedEventArgs>
        (AW2008CacheServerSyncProvider_ApplyChangeFailed);
      }

        private void AW2008CacheServerSyncProvider_
        ApplyChangeFailed(object sender, ApplyChangeFailedEven-
        tArgs e)
      {
        if (e.Conflict.ConflictType == ConflictType.ClientUpdate-
        ServerUpdate)
        {
        // Er is een conflict gevonden waarbij beide kanten een
        update hebben gedaan.
        // We laten hier de Server kant de update geforceerd door
        laten gaan.
        MessageBox.Show("Er is een conflict gedetecteerd, " +
        "de server wijzigingen zijn verloren.");
        e.Action = ApplyAction.RetryWithForceWrite;
        }
      }
    }
}
```

LISTING 1

- *Foutcorrectie*
  Het lastigste stuk van synchroniseren is wat te doen bij conflicten. Ook hierbij helpt de Sync Service bijzonder goed. De architectuur van de Sync Service komt eenvoudig gezegd op het volgende neer. Een SyncAgent coördineert synchroniseren. Deze kwamen we al tegen in de Synchronisatie click method. De SyncAgent stuurt providers

Directioneel synchroniseren kan je ook in de SQL 2005-variant inbouwen

aan die ieder vervolgens hun eigen data bewerken. Van die providers zijn er verschillende beschikbaar Wij gebruiken hier een SQLCE-provider voor de client en de DbServer-provider voor de server. Deze classes zijn door de wizard gegenereerd en staan als partial classes gedefinieerd in AW2008Cache.Designer.cs. Deze kan je eenvoudig uitbreiden door eigen partial classes te schrijven in AW2008Cache.cs. Open die file en zorg dat deze de code uit listing 1 bevat. De server-provider krijgt een ApplyChangeFailed method die wordt gekoppeld aan het ApplyChangeFailed event. In deze methode kan vervolgens op het conflicttype worden geselecteerd. In dit voorbeeld is een conflict gekozen waarbij beide kanten hetzelfde record gewijzigd hebben. De serverprovider meldt het conflict aan de gebruiker en laat de update geforceerd door gaan en daarmee de serverwijziging verloren gaan. Start de applicatie en verander de CreditRating van een Vendor in de Client en druk op save. Verander ook de CreditRating van dezelfde Vendor op de Server in iets anders. Klik nu op de synchronisatiebutton en zie dat de melding verschijnt. De wijziging op de server is nu ongedaan gemaakt en vervangen door de wijziging op de client.

Conclusie

In dit artikel laten we zien dat synchroniseren van data geen probleem meer hoeft te zijn. De tools ondersteunen de ontwikkelaar hierin zeer goed. Vooral in combinatie met SQL Server 2008 bestaan er eigenlijk geen redenen meer waarom je niet aan synchronisatie zou doen. Denk wel goed na of je nu referentietabellen in een cache gaat bewaren of dat je een echte offline werkende applicatie wil bouwen. Maar laat je vooral niet meer weerhouden door de moeilijkheidsgraad. .net

Links

SQL Server Samples (o.a. AdventureWorks): http://www.codeplex.com/SqlServerSamples

Sync Framework Blog:
http://blogs.msdn.com/sync/

Visual Studio:
http://msdn.microsoft.com/nl-nl/vstudio/default(en-us).aspx

SQL 2008:
http://www.microsoft.com/sql/default.mspx

Mark Giesen (mark.giesen@atosorigin.com) is een Technisch projectleider bij Atos Origin Nederland. Hij heeft jarenlange ervaring in het ontwerpen en bouwen van .NET applicaties. Zijn speciale interesse gaat uit naar database-gerelateerde onderwerpen als synchronisatie, Entity Framework, SQL Server en LINQ. Daarnaast hebben Agile-ontwikkelmethodieken zijn bijzondere aandacht.