Backup mogelijkheden SQL Azure databases

Kiezen uit verschillende oplossingen

Mohamed Allali

Bij de ontwikkeling van enterprise applicaties dienen we altijd rekening te houden met disaster recovery en backups. Het gaat hierbij enerzijds om het voorkomen van gegevensverlies en anderzijds om het bieden van de mogelijkheid om oude situaties van de gegevensset te kunnen herstellen. Dit artikel geeft een indruk van de (on)mogelijkheden die het Windows Azure Platform ons biedt bij het realiseren van een backup oplossing.

Binnen de Azure omgeving is verlies van data bijna ondenkbaar, maar zeker niet uitgesloten. Het omvallen van het Amazon data center heeft onlangs duidelijk gemaakt dat backups en geo-replicatie nog steeds van belang zijn voor bedrijfskritische systemen in de Cloud. Voor het Azure platform maakt Microsoft gebruik van data replicatie en disaster recovery mechanisme. Daarnaast biedt het platform de mogelijkheid om de data geografisch te verspreiden over verschillende datacenters. Hierbij dient opgemerkt te worden dat het toepassen van geografische replicatie in bepaalde gevallen verboden is door de wetgeving in Europa. Vanuit de business kan het echter nodig zijn om op oude situaties van de database te kunnen terugvallen. Disaster recovery en hardware redundantie van het Microsoft Azure Platform lossen dit probleem niet op. De reden voor deze behoefte kan de wetgeving zijn die de bedrijven en overheidsinstanties verplicht stelt de gegevens voor een bepaalde periode te bewaren of voor het geval een gebruiker per vergissing data heeft verwijderd of gemanipuleerd. In geval van een incident, onderzoek of simpelweg ten behoeve van analyse doeleinden wil men vaak de mogelijkheid hebben om terug te kunnen in de tijd. Daarnaast kan het ook handig zijn voor testdoeleinden gedurende de ontwikkelperiode, waarbij de verschillende gegevenssets eenvoudig kunnen worden teruggeplaatst.
Voor dit soort scenario's biedt Azure op dit moment nog geen volledige Cloud backup en restore service. De mogelijkheid bestaat om een snapshot te maken van de prijzige SQLAzure databases, maar dat zal resulteren in een verdubbeling van de kosten voor een enkele kopie.

Alternatieven

In dit artikel wordt een aantal alternatieven voor de backup van een SQLAzure database verkend en in het belang van de demo worden de volgende (fictieve) requirements aan de oplossing gesteld:

- De backup wordt dagelijks uitgevoerd.
- De backups van de laatste 7 dagen, laatste week en per maand worden bewaard. Oude backups die niet aan deze tijdsorde voldoen moeten worden verwijderd.
- Voor de oplossing is geen lokale infrastructuur beschikbaar.
- De kosten van de opslag moeten zo laag mogelijk worden gehouden.
- De user interface wordt beschikbaar gesteld waarmee backups teruggezet kunnen worden of handmatig aangemaakt worden

Oplossingen

Voor de eerder genoemde scenario's biedt Azure een aantal mogelijke oplossingen, variërend van volledig in de Cloud tot een oplossing op lokatie. Binnen het Microsoft veld zijn de volgende noemenswaardige oplossingen mogelijk:

1. On-premise Microsoft SQL Server in combinatie met Azure Data Sync
2. Azure VM Role met Microsoft SQL Server in combinatie met Azure Data Sync
3. Azure Worker Role met Data-tier Application Framework v2.0

De eerste oplossing lijkt de makkelijkste te zijn. De SQL Azure database wordt in eerste instantie gesynchroniseerd met een SQL Server op locatie. Dit wordt gedaan door gebruik te maken van de SQL Data Sync tool of de nieuwe Azure service genaamd Azure Data Sync. Na de synchronisatie kan de SQL Server vervolgens op de gebruikelijke wijze gebackupt worden naar bijvoorbeeld het filesysteem. De twee grote voordelen van deze optie zijn de snelheid waarmee de oplossing gerealiseerd wordt en de stabiliteit waarmee het functioneert.
Het nadeel is echter het gebruik van de on-premise infrastructuur. Dit betekent namelijk dat de server beheerd en onderhouden moet worden en dat brengt ook weer bepaalde kosten met zich mee. Verder kan men met deze oplossing niet spreken van een volledige Cloud oplossing. Indien men toch besluit tot deze toepassing, dan dient ook rekening te worden gehouden met de kosten voor het datatransport vanuit het Azure data center. Bij zeer

grote hoeveelheden data kan dit tot onaangename verrassingen leiden.
De tweede oplossing is een hybride oplossing en lijkt veel op de vorige oplossing. De VM Role neemt weliswaar de infrastructuur zorgen weg, maar de beheersdruk door het gebruik van de SQL Server blijft gelijk. Een pluspunt hierbij is dat wanneer de VM Role binnen hetzelfde data center draait als de SQL Azure database, er geen additionele kosten zijn tijdens de synchronisatie met de SQL Server.
Het nadeel van deze oplossing is dat de Azure Fabric Controller de VM Role instance op een willekeurig moment kan stoppen om die vervolgens elders weer online te brengen. Dit betekent dat SQL Server op dat moment plotseling wordt afgesloten en is het mogelijk dat de backup snapshots corrupt raken. Verder is het mogelijk dat de Fabric Controller de originele VM terugplaatst, waardoor alle opgeslagen data verdwenen is. Het is dus hier van belang om geen archiefgegevens op te slaan in de VM, maar bijvoorbeeld in een Blob storage. Deze nadelen maken deze tweede oplossing minder stabiel dan de eerste optie. Overigens is een licentie voor het gebruik van de SQL Server Standard of Enterprise een vereiste. Deze oplossing is wellicht het meest kostbare alternatief.

Maatwerk

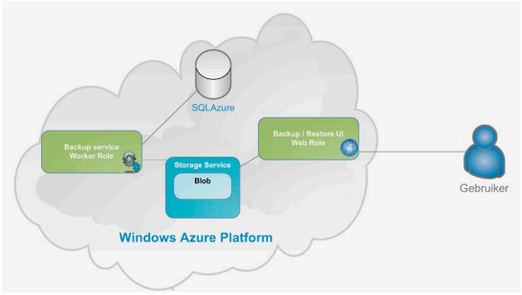
De derde optie is volledig Cloud based, maar ook volledig maatwerk. In gevallen waarbij men zich geen SQL Server op locatie kan permitteren, kan een service worden ontwikkeld voor het exporteren van de SQL Azure database naar de voordeliger Azure Blob Storage. Deze oplossing biedt weliswaar de mogelijkheid om complexe backup logica te implementeren, maar is daarentegen niet even robuust als de on-premise oplossing die hierboven beschreven staat. Het feit dat dit een maatwerk oplossing betreft, maakt het ook niet ideaal voor enterprise gebruik. De Azure kosten bij deze oplossing zijn in verhouding tot de overige twee oplossingen wel het laagst. Dit komt vooral doordat de backupdata het Azure data center niet verlaat en er geen software licenties noodzakelijk zijn. Deze oplossing is eenvoudig te implementeren op basis van het nieuwe Data-tier Application Framework 2.0 (DAC).
We zullen deze oplossing verder uitwerken op basis van de eerder genoemde requirements.

Backup service in Azure

We maken gebruik van de volgende .NET en Azure componenten:

- Azure Web- en Worker role
- SQL Azure en Blob Storage
- Data-tier Application Framework 2.0 (DAC).

De volgende plaat geeft de high-level oplossing weer:



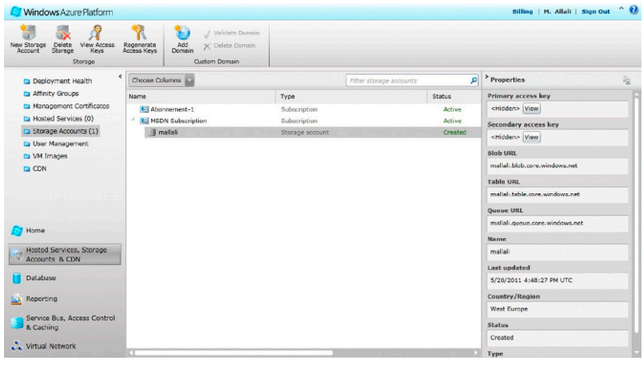
FIGUUR 1: HIGH-LEVEL BACKUP SOLUTION

De Worker Role draait als een service en bestaat uit een scheduler die op gezette tijden de backup uitvoert. De backup logica wordt geïmplementeerd op basis van het command pattern.
De Web Role bestaat uit een MVC.NET applicatie die een overzicht geeft van alle uitgevoerde backups in een Blob Storage en biedt tevens de mogelijkheid om backups te herstellen of handmatig uit te voeren.
Voor het exporteren en herstellen van de data wordt de nieuwe DAC 2.0 API met ondersteuning voor SQL Azure ingezet. Deze API is onderdeel van de SQL Azure Export/Import CTP en is momenteel beschikbaar op de Microsoft SQL Azure Labs.
Deze oplossing maakt verder gebruik van Visual Studio 2010, SQL Server 2008 R2 en een Azure account. Een Azure account is niet noodzakelijk aangezien Visual Studio 2010 een gesimuleerde Azure omgeving heeft voor ontwikkeldoeleinden.

Voor deze opdracht is het volgende stappenplan uitgestippeld:
Stap 1: Setup Azure omgeving
Stap 2: Creëren Visual Studio 2010 Solution
Stap 3: Implementeren backup service in een Worker Role
Stap 4: Implementeren User Interface in een Web Role

Stap 1 : Setup Azure Omgeving
Voor de demonstratie van deze oplossing wordt een Azure omgeving gereed gemaakt. De volgende acties worden ondernomen binnen de Azure Portal:

- Creëren van de database ten behoeve van de demo
- Instellen van een Storage Account voor de opslag van de backups



FIGUUR 2: AZURE PORTAL

Zoals eerder is vermeld, is deze omgeving niet noodzakelijk voor de demo, maar het is wel verstandig om de eindoplossing daar ook te testen.

Stap 2 Creëren Visual Studio 2010 Solution
In Visual Studio wordt gekozen voor een Windows Azure Project en vervolgens worden de benodigde Azure Roles gekozen, namelijk een ASP.NET MVC 2 Web Role en een Worker Role. VS2010 zal daarna op bijna magische wijze de solution inrichten en gereed maken voor de implementatie. De worker role zal draaien als een service en is de host voor onze scheduler en de Web Role zal als host fungeren voor de user interface van de backup service.

Stap 3 Implementeren backup service in een Worker Role
Om de complexiteit van de backup logica enigszins te versimpelen en eventueel te kunnen testen, wordt gebruik gemaakt van encapsulatie. Het backup model is als volgt gestructureerd.

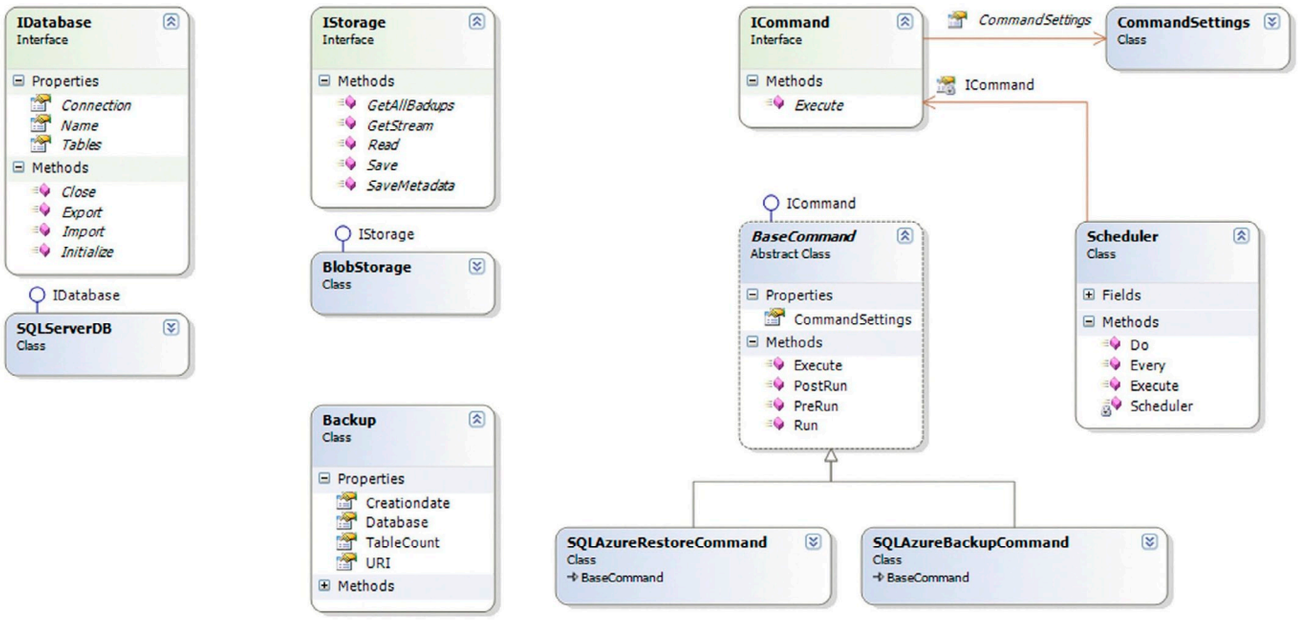


FIGURE 3: MODEL BACKUP SERVICE

Voor deze demo wordt gebruik gemaakt van een timer. Een betere aanpak is het toepassen van de nieuwe Azure queues, waarin standaard scheduling is ingebouwd. Door gebruik te maken van queues worden meerdere instances gelijktijdig ondersteund. De Scheduler in deze demo bestaat uit een timer die op gezette tijden de SQLAzureBackupCommand uitvoert en wordt als volgt in de Worker Role gebruikt:

```
string databaseConnectionString = ConfigurationManager.
ConnectionStrings["Database"].ConnectionString;
string storageConnectionString = "StorageConnectionString";
CommandSettings settings = new CommandSettings(databaseConnection
String, storageConnectionString);
SQLAzureBackupCommand azureBackupCommand =
new SQLAzureBackupCommand(settings); Scheduler.Every(TimeSpan.
FromDays(1)).Do(azureBackupCommand);
```

De Azure database wordt geëncapsuleerd in een SQLServerDatabase class, waarmee het exporteren en importeren door middel van de DAC API vereenvoudigd wordt. De geëxporteerde data in de backup bevat zowel de data als de schema's. Daarnaast worden de aanwezige triggers, stored procedures en andere definities opgenomen in de export. De backups in dit scenario worden uiteindelijk opgeslagen in een Blob Storage. De API voor de Blob Storage wordt verborgen door de class BlobStorage en impementeert de interface IStorage. Door de DAC en Azure API's te verbergen wordt de implementatie van de SQLAzureBackupCommand een stuk eenvoudiger, namelijk:

```
IStorage storage = new BlobStorage(_settings.Storage
ConnectionString);
string directoryName = DateTime.Now.ToString("yyyyMMddHHmmss");
IStorage storage = new BlobStorage(_settings.Storage
ConnectionString);
_database.Export(storage);
```

Voor het terugplaatsen van een backup naar de SQL Azure wordt gebruik gemaakt van de import functionaliteit van DAC 2.0. Hiervoor wordt de volgende code gebruikt:

```
ServerConnection connection = new ServerConnection(this.
Connection);
DacStore store = new DacStore(connection);
DatabaseDeploymentProperties properties =
new DatabaseDeploymentProperties(connection, "MyAzureDB");
properties.AzureEdition =
Microsoft.SqlServer.Management.Dac.AzureEdition.Default;
Stream stream =
storage.GetStream(packageLocation, this.Connection.Database);
 stream.Seek(0, SeekOrigin.Begin);
store.Import(stream, properties);
stream.Close();
```

De import method verwacht hier de naam van de doeldatabase en een stream waarin de backup data is opgeslagen.

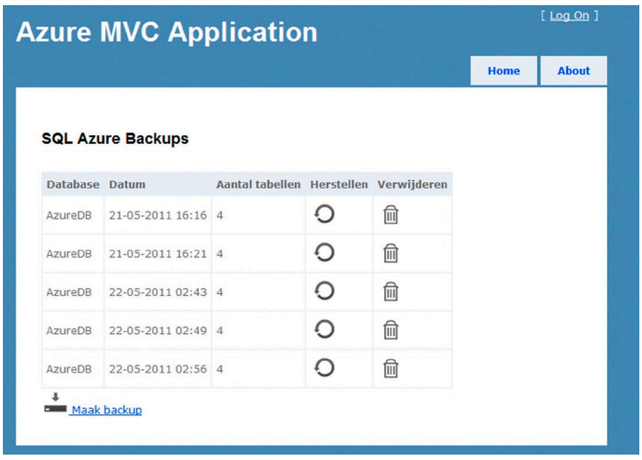
Stap 4 Implementeren User Interface in een Web Role

De opgeslagen backups zullen een keer weer hersteld moeten worden. Een user interface(UI) waarmee met gemak de kopieën teruggezet kunnen worden is hier gewenst. Voor dit demodoeleinde is een MVC.NET applicatie in een Web Role opgenomen, waarmee aan de herstel requirement voldaan wordt. De UI maakt gebruik van het eerder genoemde backupmodel om een geselecteerde backup uit een Blob storage in een SQL Azure database te importeren. Voor een overzicht van alle beschikbare backups wordt de Blob Container doorlopen en wordt de metadata daarvan opgehaald en zichtbaar gemaakt in een MVC .NET View. De volgende plaat geeft de user interface weer zoals deze is ontwikkeld voor deze demo:

Indien deze oplossing gebruikt zou worden in een productieomgeving, dan is het aan te raden om de import en export uit te voeren op een snapshot van de productie database. Hiermee wordt voorkomen dat inconsistente data wordt geëxporteerd. Het is namelijk mogelijk dat de live database tijdens de backup in gebruik is.

Conclusie

In dit artikel is een aantal backupoplossingen voor de SQL Azure databases verkend. De uitgewerkte maatwerk backupoplossing



FIGUUR 4: USER INTERFACE VAN DE BACKUP EN RESTORE SERVICE

biedt een uitweg voor scenario's waarbij de on-premise infrastructuur niet beschikbaar is of een aanvullende backupdienst buiten Azure niet in verhouding is tot de waarde van de data. Op verschillende internet fora wordt de roep harder om een geïntegreerde Azure backup functionaliteit toe te voegen.
Microsoft heeft onlangs een import/export tooling voor SQL Azure als een CTP gelanceerd. Hiermee wordt momenteel nog volop geëxperimenteerd en het is nu nog onduidelijk wanneer deze tooling commercieel beschikbaar zal zijn. Het huidige gat in de Azure dienstverlening is overigens ook opgemerkt door de markt. Een aantal bedrijven heeft onlangs aangekondigd een Azure backup service te gaan bieden voor enterprise gebruik.

De broncode van deze demo is overigens op verzoek te verkrijgen door contact op te nemen met de auteur. De contactgegegevens zijn bij dit artikel vermeld.

Links:

http://www.sqlazurelabs.com/ImportExport.aspx
http://www.microsoft.com/windowsazure/sqlazure/datasync/

Mohamed Allali, is een Microsoft Software Architect bij Betabit BV. Mohamed is bereikbaar via m.allali@betabit.nl