



Overview of Microsoft SQL Azure Database

Writers

Jason Lee, Graeme Malcolm, and Alistair Matthews (Content Master)

Technical Reviewers

Rick Negrin (Microsoft), Zach Owens (Microsoft), David Robinson (Microsoft)

Published

Sept 2009

Applies to

SQL Azure

Summary

SQL Azure Database is a cloud database service from Microsoft. SQL Azure provides web-facing database functionality as a utility service. Cloud-based database solutions such as SQL Azure can provide many benefits, including rapid provisioning, cost-effective scalability, high availability, and reduced management overhead. This paper provides an architectural overview of SQL Azure Database, and describes how you can use SQL Azure to augment your existing on-premises data infrastructure or as your complete database solution.

Copyright

This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This white paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in, or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2009 Microsoft Corporation. All rights reserved.

Microsoft, ADO.NET Data Services, Cloud Services, Live Services, .NET Services, SharePoint Services, SQL Azure, SQL Azure Database, SQL Server, SQL Server Express, Sync Framework, Visual Studio, Windows Live, and Windows Server are trademarks of the Microsoft group of companies.

All other trademarks are property of their respective owners.

Contents

- Introduction..... 4
- Key Features 4
 - Manageability 5
 - Low-Friction Provisioning..... 5
 - High Availability 5
 - Scalability 5
 - Global Scalability..... 5
 - Multi-Tenant Support..... 5
 - Developer Empowerment 6
 - Familiar Client Development Model 6
 - Proven Relational Data Model..... 6
 - Synchronization and Support for Offline Scenarios 6
- Typical Scenarios..... 7
 - Web Application 7
 - Departmental/Workgroup Application 7
 - Data Hub 8
 - ISV S+S Offering..... 9
- Architectural Overview..... 10
 - Provisioning Model..... 10
 - Windows Azure Platform Accounts 10
 - SQL Azure Servers 10
 - SQL Azure Databases 10
 - Relational Database Model..... 11
 - Data Access Architecture..... 11
 - Security Model..... 12
 - Scaling Out Databases 12
 - Deployment..... 13
- Conclusion and Next Steps 13

Introduction

Companies that provide Internet-based applications are facing many challenges today. Users expect access to ever-increasing amounts of data from anywhere, at any time, and from any device. The size, scale of use, and variety of forms of data are expanding rapidly. Developers must build and deploy applications quickly to keep up with these growing demands. Using the traditional on-premise data management model, meeting these needs demands constant investment in and management of servers, operating systems, storage, and networking. IT and operational staff must constantly monitor the infrastructure to ensure that capacity, performance, and availability are maintained as data volumes and user loads increase.

Cloud database services, such as Microsoft® SQL Azure Database, provide an improved way to respond to these challenges. SQL Azure is built on three key tenets: manageability, scalability, and developer agility.

From a developer's perspective, SQL Azure offers the well-known rich relational programming model, and uses a familiar data access protocol and simple deployment options. SQL Azure simplifies the process of creating, prototyping, and deploying applications that integrate data across the enterprise. SQL Azure removes infrastructure obstacles, thereby giving developers more freedom to innovate and experiment with new ways of sharing data.

From the IT management perspective, SQL Azure offers a systematic and secure cloud-deployed solution that integrates with your on-premise assets and gives the IT organization oversight and control of distributed data assets. SQL Azure is built on the same Microsoft SQL Server® technologies that have already been used and proven in on-premise deployments to provide high availability, reliability, and security.

From the business perspective, SQL Azure offers a cost-effective approach for managing data, with a flexible consumption-based pricing plan, near-zero capital and operational expenditures, and the ability to quickly and easily scale up or down as your needs change.

If you are planning to build applications on large or shared data sets, provide on-demand scalable data storage, or augment your on-premise data infrastructure with low-cost, rapidly provisioned cloud-based storage, SQL Azure can provide a robust and cost-effective solution.

Key Features

SQL Azure is a key component of the Microsoft data platform offering flexibility and scalability; reliability and security; and developer agility. Let's begin by looking at some of these features.

Manageability

SQL Azure Database offers the high availability and functionality of an enterprise data center without the administrative overhead that is associated with an on-premise solution. This self-managing capability enables organizations to provision data services for applications throughout the enterprise without adding to the support burden of the central IT department or distracting technology-savvy employees from their core tasks to maintain a departmental database application.

Low-Friction Provisioning

When you use the traditional on-premise data infrastructure, the time that it takes to deploy and secure servers, network components, and software can slow your ability to prototype or roll out new data-driven solutions. However, by using a cloud based solution such as SQL Azure, you can provision your data-storage needs in minutes and respond rapidly to changes in demand. This reduces the initial costs of data services by enabling you to provision only what you need, secure in the knowledge that you can easily extend your cloud-based data storage if required at a future time.

High Availability

SQL Azure is built on robust and proven Windows Server® and SQL Server technologies, and is flexible enough to cope with any variations in usage and load. The service replicates multiple redundant copies of your data to multiple physical servers to ensure data availability and business continuity. In the case of a disaster, SQL Azure provides automatic failover to ensure maximum availability for your application.

Published service level agreements (SLAs) guarantee a business-ready service. When you move to SQL Azure, you no longer need to back up, store, and protect data yourself.

Scalability

A key advantage of the cloud computing model is the ease with which you can scale your solution. Using SQL Azure, you can create solutions that meet your scalability requirements, whether your application is a small departmental application or the next global Web success story.

Global Scalability

A pay-as-you-grow pricing model allows you to quickly provision new databases as needed or scale down the services without the financial costs associated with unused capacity. With a database scale out strategy your application can utilize the processing power of hundreds of servers and store terabytes of data.

SQL Azure runs in worldwide data centers, so you can reach new markets immediately. If you want to target a specific region, you can deploy your database at the closest data center. You can harness this global scalability to build the next generation of Internet-scale applications that have worldwide reach, but without the infrastructure costs and management overhead.

Multi-Tenant Support

Independent software vendors (ISVs) who develop Software+Services (S+S) offerings must provide adequate isolation for individual customers' data. ISV's must be able to charge each customer the right price for the data storage services that they have consumed. SQL Azure provides the flexibility that ISVs

need to segregate customer data and implement multi-tenant billing, which enables you to build a global S+S solution quickly and easily.

Developer Empowerment

One of the potential obstacles to building great cloud-based applications is the requirement for developers to learn new tools, programming platforms, and data models. However, SQL Azure is built on top of the TSQL language and is designed to be compatible with SQL Server with a few changes, so developers can use their existing knowledge and skills. This reduces the cost and time that is usually associated with creating a cloud-based application.

Familiar Client Development Model

When developers create on-premise applications that use SQL Server as a data store, they employ client libraries that use the Tabular Data Stream (TDS) protocol to communicate between client and server. There is a large global community of developers who are familiar with SQL Server and have experience of using one of the many client access libraries that are available for SQL Server, such as Microsoft ADO.NET, Open Database Connectivity (ODBC), JDBC and the SQL Server driver for PHP. SQL Azure provides the same TDS interface as SQL Server, so developers can use the same tools and libraries to build client applications for data that is in the cloud.

Proven Relational Data Model

SQL Azure data is stored in a way that is very familiar to developers and administrators who use SQL Server. You can create a SQL Azure Server which is a group of databases that are spread across multiple physical machines. This SQL Azure Server is in some ways conceptually analogous to a SQL Server instance and acts as an authorization boundary just as in SQL Server. You can also set geo-location at this level. Windows® Azure™ and SQL Azure data centers are located worldwide; if your application is relevant to a specific region, you can increase performance by geo-locating it there.

Within each server, you can create multiple databases that have tables, views, stored procedures, indices, and other familiar database objects. This data model ensures that your database developers can use their existing relational database design and Transact-SQL programming skills, and easily migrate existing on-premise database applications to the cloud.

SQL Azure servers and databases are logical concepts that do not correspond to physical servers and databases. This abstraction enables the flexible provisioning that was described earlier in this paper. Administrators and developers can concentrate on data model design because SDS insulates them from the physical implementation and management.

Synchronization and Support for Offline Scenarios

SQL Azure is part of the rich Microsoft data platform which integrates with the Microsoft Sync Framework to support occasionally connected synchronization scenarios. For example, by using SQL Azure and the Sync Framework, on-premise applications and client devices can synchronize with each other via a common data hub in the cloud.

Typical Scenarios

To see how these key features of SQL Azure can benefit organizations, let's consider some common business application scenarios.

Web Application

Most Web sites require a database to store user input, e-commerce transactions, and content, or for other purposes. Traditionally, such a data-driven Web site is implemented with a database server in the same data center as the Web server.

Using SQL Azure, Web developers can choose to place data in the cloud where it is highly available and fault tolerant. As with the departmental application scenario, you can host your Web application on your own server, or by using a third-party Web hoster, and access the data in SQL Azure across the Internet. However, to avoid issues of performance and application complexity that latency causes, you should consider hosting the Web site itself in Windows Azure so that it can benefit from co-location with your SQL Azure Database.

Departmental/Workgroup Application

In a large organization, qualified database administrators run the mission-critical, company-wide databases, which receive the benefits of their experience. These databases may also have fault-tolerant configurations that have uninterruptible power supplies, redundant array of inexpensive disk (RAID) storage, and clustered servers.

In contrast, a typical large organization also has many smaller database applications that IT-literate employees in departments and groups across the organization have created. They may have built such applications by using Microsoft Office Access®, Microsoft SQL Server® Express, or third-party software. The databases may be hosted on a single, cheap server, or even on a desktop computer, and they are typically managed by staff whose primary role is not data administration. Although the impact of a database failure is usually limited to the department that uses it, such interruptions can hamper productivity for large groups. It is also difficult to keep track of all such databases throughout your organization.

SQL Azure represents an excellent opportunity to reorganize such disparate database applications. By provisioning SQL Azure Databases for small departmental applications, your users can benefit from the self-management capabilities and fault tolerance that SQL Azure provides, without placing an additional burden on your IT staff. This centralized approach also makes it much easier to audit the databases in your organization. In addition, because SQL Azure has a pay-as-you-grow pricing structure, you can run small database applications very inexpensively.

When you migrate an on-premise client-server application to SQL Azure, you can choose to leave the client application on premise and migrate only the data tier, which you can access by using common data access libraries across the Internet. However, if you use this application design, you must consider the latency issues that are inherent in Internet-based connectivity, which may result in more complex code in the client application. A better solution is to move the data access logic to Windows Azure, so that the same data center hosts both the data access code and the data itself. You can create a Web-based user interface (UI) in Windows Azure to which users connect with a browser. Alternatively, you can create a

service by using ADO.NET Data Services to write code that exposes a SOAP, REST, or JSON interface to a simple desktop UI.

Data Hub

In a data hub scenario, you typically want to enable various mobile and remote users to collaborate by using the same set of data. Consider an insurance company that has a large mobile sales force that consists of more than five thousand people who are scattered across North America. Keeping customer and pricing data synchronized across the entire sales force is a constant problem. The first part of the problem is getting new customer contact information from the sales force into the internal finance systems. The second part is getting new price-list information out to the sales force. The insurance company needs a solution that will:

- ▶ Keep each salesperson's portable computer up to date with the latest pricing information
- ▶ Keep the corporate system up to date with new customer information from the portable
- ▶ computer of each salesperson, without the risk of exposing critical corporate data

Currently, product and customer data is stored in a central SQL Server database in the data center. In addition, employees in the sales force use an application that runs on their portable computers and stores data in SQL Server Express. The IT department does not want to open the firewall to the on-premise data center to provide possibly insecure access from each salesperson's portable computer. The development team can provide a safe and fully synchronized solution that uses SQL Azure, by completing the following three tasks:

1. Create a database in SQL Azure to store product data and customer data.
2. Create a Sync Framework provider for the data center. This Sync Framework provider keeps product and customer data synchronized between the data center and the SQL Azure data hub.
3. Create a second Sync Framework provider for the sales force's portable computers. This Sync Framework provider keeps product and customer data synchronized between the field salespeople and the SQL Azure data hub.

The diagram in Figure 2 illustrates this solution.

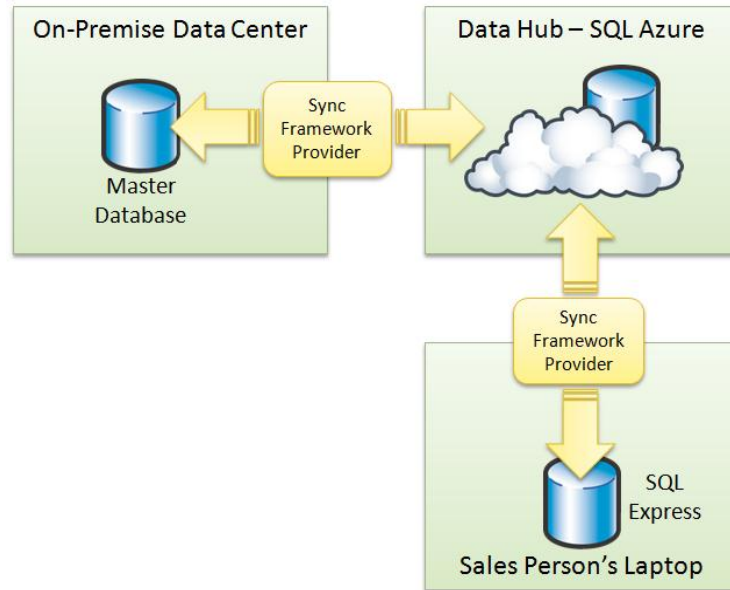


Fig 2: Conceptual overview of a data hub scenario

Product pricing data flows from the enterprise database, through SQL Azure, to more than five thousand salespeople. Customer contact data flows from more than five thousand salespeople, through SQL Azure, to the enterprise database. When a salesperson's portable computer is offline, changes that occur to local data are tracked. When the portable computer's Internet connection is restored, the Sync Framework provider enumerates these changes and sends them to SQL Azure. The safety of the corporate data center is ensured.

ISV S+S Offering

ISVs can use SQL Azure to offer Software+Services solutions without distracting themselves from their core software development competency so that they can develop a hosting infrastructure capability. Windows Azure provides an ideal environment to host software services without the overhead of maintaining the hosting infrastructure at the premises of the ISV or the customer.

Consider an S+S vendor that provides compliance support to businesses, including financial, government, health-care, real estate, and franchising companies. Such organizations need to store historical data, such as financial records, business transactions, or correspondence, either for future reference or for compliance with record-keeping regulations. The S+S vendor employs a document management system for archived data that provides full text search, together with workflow functionality and check processing. The S+S vendor must also track and report access to resources as required for audit purposes. To reduce data storage costs and help to ensure rapid and secure access to records, the vendor wants to migrate its customers' archive data to the cloud.

To achieve this aim, the company can create an account with SQL Azure, together with Windows Azure accounts within the company space for each of its customers. After customers have an account, they can upload any form of document, such as e-mail, scanned checks, and escrow documents. Some of the

documents are stored as binary large objects (BLOBs) in the Windows Azure BLOB store, whereas other documents are stored as structured data in SQL Azure with standardized data fields.

In this scenario, it is essential to ensure that each customer's data is isolated and only available to the right users. The S+S vendor can use SQL Azure to implement this isolation and prevent inappropriate access. SQL Azure also makes it simple to audit usage so that it is possible to bill customers appropriately.

Developers who write this solution for Windows Azure will find that it is tightly integrated with SQL Azure. For example, they can use the familiar SQL Server client libraries that they use on the desktop; they can use one Windows Azure account to authenticate a user across the cloud; and they can use the same geo-location properties for the Windows Azure code and the SQL Azure data.

Architectural Overview

In this section, you will see how SQL Azure uses databases, how it structures relational data, and how you can connect to data.

Provisioning Model

SQL Azure is designed to support extreme scale and low cost while providing a familiar environment to administrators and developers. It has the hierarchical provisioning model that is described below to achieve this.

Windows Azure Platform Accounts

To use SQL Azure, you must begin by creating a Windows Azure platform account. Using this account, you can access all of the facilities within the Windows Azure platform. This account is also used to bill for usage for all Windows Azure platform services.

SQL Azure Servers

Each Windows Azure account can contain multiple SQL Azure servers. These servers are not implemented as SQL Server instances; instead, you can view them as logical concept that is used to provide a central administrative point for multiple SQL Azure servers. Each server includes logins, just as you find in on-premise SQL Server instances, and you can also specify the geographic region your server is located in at this level.

You use the SQL Azure portal to create and manage your database server. This portal provides an easy-to-use interface where you can create logins and provision databases.

SQL Azure Databases

Each SQL Azure Database server can contain multiple databases. A new database server has a master database that is just like an on-premise SQL Server instance. In each database, you can create tables, views, indices, stored procedures, and other familiar database objects. You can use the SQL Azure portal to create a new database. Alternatively, you can use the Transact-SQL `CREATE DATABASE` command.

SQL Azure Databases are implemented as replicated data partitions across multiple physical computers in an SQL Azure data center. This architecture provides automatic failover and load balancing. Customer data is spread across multiple physical servers within the geo-location that is specified for the SQL Azure

Database server that is hosting the database. In this way, SQL Azure Database achieves high availability and stability for all applications from the smallest to the largest without requiring intensive administrative effort.

Relational Database Model

A key design aim for SQL Azure is to provide a familiar environment for database programmers. Therefore, the objects that you can create in a SQL Azure Database are the same as those that are available in a SQL Server database. These include:

- ▶ **Tables:** store data in rows with a consistent and normalized structure.
- ▶ **Indexes:** increase the speed of searches and maximize performance.
- ▶ **Views:** provide alternative ways to look at data in one or more tables.
- ▶ **Stored procedures:** store common Transact-SQL scripts for simple execution.
- ▶ **Triggers:** ensure data integrity by executing checks when data is modified.

Both SQL Server and SQL Azure Database use the Transact-SQL language for database creation and data manipulation. Database developers and administrators can therefore be productive immediately in SQL Azure by using their existing expertise.

For more information about Transact-SQL, see <http://msdn.microsoft.com/en-us/library/ms189826.aspx>

Data Access Architecture

SQL Azure Database exposes a Tabular Data Stream (TDS) endpoint to databases that are hosted in the cloud. TDS is the same network protocol that on premise SQL Server uses, therefore, a desktop client application can connect to SQL Azure Database in the same way it connects to an on-premise SQL Server instance. Such an application runs code that was built by using ADO.NET, ODBC, or whatever technology you prefer to work with. Queries are formulated in the Transact-SQL language. Secure Sockets Layer (SSL) is required when a client application connects to the SQL Azure Database TDS endpoint to ensure security.

In this desktop client application/SQL Azure Database scenario, you must consider the latency that may arise over the cloud and handle it in the client code. The latency is inevitably higher than it would be if the database was on your premises. One way to avoid this latency is to create a Web-based user interface for your database application and host it in Windows Azure. In this scenario, the client code and data is hosted in the same data center, so latency is low. Users connect to such an application by using a Web browser. You can also use this architecture for data-driven Web sites that are hosted in Windows Azure.

In the third architecture that SQL Azure supports, you can create application logic by using ADO.NET and the Entity Framework, and host it in Windows Azure. You can then use ADO.NET Data Services to publish this application as a service that uses a SOAP, REST, or JSON interface, and build a lightweight client application to consume data from the service. In this way, you avoid latency between the application and SQL Azure, but you still provide an Internet-based service interface for your data, which you can then use in rich Internet applications or desktop solutions.

Security Model

Many databases contain sensitive data, so it is essential to carefully control access. This is especially important in a multi-tenant application that involves users from different customers who must be isolated from each other. SQL Azure provides the same set of security principals that are available in SQL Server with SQL Server Authentication. You can use these to authorize access and secure your data:

- ▶ **SQL Server Logins:** Used to authenticate access to SQL Azure at the server level.
- ▶ **Database Users:** Used to grant access to SQL Azure at the database level.
- ▶ **Database Roles:** Used to group users and grant access to SQL Azure at the database level.

Scaling Out Databases

You can store any amount of data, from kilobytes to terabytes, in SQL Azure. However, individual databases are limited to 10 GB in size. To create solutions that store more than 10 GB of data, you must partition large data sets across multiple databases and use parallel queries to access the data.

Data sharding is a technique used by many applications to improve performance, scalability and cost. Some applications are well suited for partitioning because they use data models with natural partitioning boundaries. For example, applications that store and process sales data using date or time predicates. These applications can benefit from processing a subset of the data instead of the entire data set. Data sharding also enables parallel processing of data. Applications can place multiple data partitions on multiple sets of compute resources and processes the data simultaneously.

Although there is ample literature on the benefits of data sharding, little is said about the cost of managing a distributed database on tens or hundreds of servers. SQL Azure provides the infrastructure for applications that require tens or hundreds of databases without the associated administrative cost. An application can partition large data sets into many databases without facing an exponential cost structure and up front capital investment. Provisioning and using three hundred databases is as simple as provisioning three databases. The management burden of keeping hundreds of servers functioning in a synchronized fashion and provide a highly available database is taken on by SQL Azure.

In addition, SQL Azure provides elasticity in the scale out offering as an application can increase the number of databases when needed and decrease when the requirements change. The ability to scale down without cost penalties prevents the customer from being stuck paying for unused resources when they do not need them.

Even applications with small data sets that require large processing capacity (CPU and IO) can also benefit from partitioning by gaining access to parallel resources.

The decision to use a scale out database strategy is something that should be approached carefully as it can have an impact on the complexity of the application code and is not appropriate for every application, although the benefits described above may make it well worth the effort.

Deployment

It is possible to create and populate a database entirely in SQL Azure Database by using Transact-SQL. However, in most cases, developers or administrators will design and populate the database on the premises on a development computer or server. When the application is complete, the database must be deployed to the cloud.

To deploy a database to SQL Azure, you can create a Transact-SQL CREATE DATABASE script in Microsoft SQL Server® Management Studio using the Generate Script Wizard. You can then run the Transact-SQL script in SQL Azure to create the database.

Geo-location for SQL Azure is set at SQL Azure Server level. Therefore, to deploy a database for a specific region, you should create a new SQL Azure Server in that geo-location and connect to it to run the deployment script.

Conclusion and Next Steps

In this paper, we have introduced SQL Azure Database and described its key capabilities and benefits. SQL Azure Database is a cloud-based database service that offers developer agility, application flexibility, and virtually unlimited scalability, with a flexible, cost-effective delivery model. The robust underlying architecture provides reliability, high availability, and security. In addition, support for the most prevalent Internet communication protocols ensures ease of deployment and use. We have examined some scenarios where SQL Azure can offer real business value to customers, such as data hub solutions and archival and compliance systems. We've also taken a look "under the hood" at the architectural and programming models that provide the core functionality of SQL Azure.

For more information, please visit:

- ▶ SQL Azure Portal
<http://www.microsoft.com/azure/sql.mspx>
- ▶ SQL Azure Developer Center
<http://msdn.microsoft.com/en-us/sqlserver/dataservices/default.aspx>
- ▶ SQL Azure Documentation
<http://msdn.microsoft.com/en-us/library/cc512417.aspx>
- ▶ SQL Azure Team Blog
<http://blogs.msdn.com/SDS>
- ▶ Windows Azure Platform Training Kit
<http://www.microsoft.com/downloads/details.aspx?FamilyID=413E88F8-5966-4A83-B309-53B7B77EDF78&displaylang=en>