**Group Title**

**Topic N: SQL Azure Overview**

[](http://msdn.microsoft.com/en-us/rampup/bb352986.aspx)

Ascend to new heights in your career

[**http://msdn.microsoft.com/rampup**](http://msdn.microsoft.com/rampup)

**Contents**

[SQL Azure Features 3](#_Toc244414914)

[Manageability 3](#_Toc244414915)

[Low-Friction Provisioning 3](#_Toc244414916)

[High Availability 3](#_Toc244414917)

[Scalability 4](#_Toc244414918)

[Global Scalability 4](#_Toc244414919)

[Multi-Tenant Support 4](#_Toc244414920)

[Developer Empowerment 4](#_Toc244414921)

[Familiar Client Development Model 4](#_Toc244414922)

[Proven Relational Data Model 4](#_Toc244414923)

[Synchronization and Support for Offline Scenarios 5](#_Toc244414924)

[Architectural Overview 5](#_Toc244414925)

[A Comparison of SQL Azure and SQL Server 5](#_Toc244414926)

[Designing Databases for the Cloud 6](#_Toc244414927)

[Additional Resources 6](#_Toc244414928)

**Introduction**

One of the challenges faced by companies delivering Internet-facing applications is that of data storage and availability. It is expensive to build and maintain a data center, and the risks to uptime posed by the need to maintain and expand server hardware as well as monitoring the operating systems that run the database servers can result in a high degree of management overhead. In addition, the need to manage the allocation of hardware to applications, often makes it difficult to provision and deploy applications quickly.

SQL Azure provides companies with a way to cost-effectively deliver database services without the overhead of maintaining a private data center. Rather than needing to allocate the up-front investments necessary to host even the smallest database, SQL Azure uses a pay-as-you-go pricing model that ensures that you only pay for the services that you need. As storage needs expand, SQL Azure provides the ability to quickly respond to those needs without the overhead of acquiring and configuring additional servers.

Since SQL Azure is based upon Microsoft SQL Server technology, the same management tools and processes that are used in today’s data governance can be applied to the databases in the cloud. All of the data access patterns and security models available to your current in-house SQL databases are applicable to SQL Azure and preserves your investment in your existing IT workflows.

SQL Azure uses the same data definition and programming languages that SQL developers are already familiar with, and provides access using the same server protocols and data-access frameworks. Combined with the ease and speed of provisioning new databases, developers can become more productive and have the freedom to try new designs without having to wait for server availability.

# SQL Azure 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.

# Architectural Overview

For most practical intents and purposes, SQL Azure can be thought of as an instance of SQL Server running in Microsoft’s data centers. In reality, the infrastructure is much more complex. The current architecture consists of multiple logical layers: a services layer, a platform layer, and an infrastructure layer.

The services layer functions as the gateway between the data-access clients and databases. The services layer performs the roles of account setup, tracking and billing of usage, and routing inbound data connections to the correct servers. When you use the SQL Azure portal to create and manage databases, it is with this layer which you are interacting. At runtime, SQL Azure also keeps track of the number of database requests and bandwidth utilization and bills for these services on a per-account basis. Since the platform layer (described below) will be managing multiple instances of SQL, it is the services layer that ensures that incoming connection requests are routed to the correct internal server.

While the services layer exposes what appears to be a single endpoint to clients, the internal topology is that of multiple instances of SQL server, each with their own synchronized copy of the database. Much as the Windows Azure fabric controller manages the lifetime of servers that host Windows Azure roles, the creation and management of these server instances is performed by the SQL Azure fabric controller. The fabric controller constantly monitors the database servers and handles the duties of keeping the database servers’ data synchronized, handles failover, and performs rolling upgrades for deploying patches.

The infrastructure layer consists of the day-to-day operational functions required to keep the datacenters running. It is the duty of this layer to ensure that hardware is maintained or replaced, determining what software upgrades are necessary and should be deployed, and to monitor the overall health of the datacenter. Unlike the services and platform layer, which are completely automated, the infrastructure layer composed of a mix of human IT resources and software tools for performing configuration and monitoring processes.

# A Comparison of SQL Azure and SQL Server

Just as with the standard SQL Server product, a wide range of tools may be used to access SQL Azure. Since SQL Azure supports the Tabular Data Stream (TDS) protocol, virtually any tool that can be used to connect to SQL Server, issue queries, and execute commands is supported to some degree. Tools such as sqlcmd and BCP work in a manner almost identical to that of SQL Server. However, due to the facts that not all TSQL commands are supported and the internal database structure of SQL Azure is slightly different, not all features of tools like SQL Server Management Studio, such as the ability to browse the database structure, are supported yet.

With support of TDS, SQL Azure brings support for a wide range of connectivity options. Access from managed applications using the Microsoft .NET Framework may use the standard ADO.NET libraries. Unmanaged applications may use the ODBC drivers provided for SQL Server (but not OLE DB). Finally, access for PHP applications is provided by Microsoft’s SQL Server driver for PHP.

# Designing Databases for the Cloud

In the current version, SQL Azure provides two “editions” of SQL Azure for hosting your data: a Web Edition, and a Business Edition. The difference between the two are maximum database size, the amount of traffic allowed, and the pricing. The Web Edition database is constrained to a maximum database size of 1GB and is limited to 10 query hours per month for a monthly fee of $9.99. The Business Edition is constrained to a maximum database size of 10GB and is limited to 100 query hours per month for a monthly fee of $99.99. In both cases, the maximum size does not include the size of replicas for failover.

This database size limitation imposes some architectural constraints. Whereas with the installable version of SQL Server, application architects can assume maximum database sizes in the ranges of hundreds of gigabytes, the limitation of 1/10G means that databases must be designed to “scale out” much as applications targeting Windows Azure must be. A principle often used to accomplish this type of data scalability is known as “sharding.” Rather than writing the application against the assumption that all data resides in the same database, the data is partitioned across boundaries such as geography, billing group, and so forth. This partitioning is then used to determine to which database a given query should be directed.

One of the most important principles for designing applications targeting SQL Azure is to design for failure. Since the lifetime of a given instance of a SQL Azure database is being managed by the SQL Azure Fabric, it is possible that a pooled connection to a server may become invalid and will fail to return a result, and this may occur at any point in the application’s code. Therefore, any process that involves writing data to the database must be written so that it is repeatable without any negative side-effects such as duplication. Often this is as simple as writing stored procedures to leverage the “upsert” pattern (insert if a record with a matching key does not exist, update if it does).

# Additional Resources

Additional resources for learning about SQL Azure are listed below:

|  |  |
| --- | --- |
| **Name** | **Location** |
| SQL Azure Developers Portal | <http://msdn.microsoft.com/en-us/sqlserver/dataservices/default.aspx> |
| Windows Azure and ISVs: A Guide for Decision Makers | http://download.microsoft.com/download/E/7/4/E74D55E6-D156-404F-B6C5-A53A9A4B1D42/Windows%20Azure%20for%20ISVs%20v1%2011--Chappell.pdf |