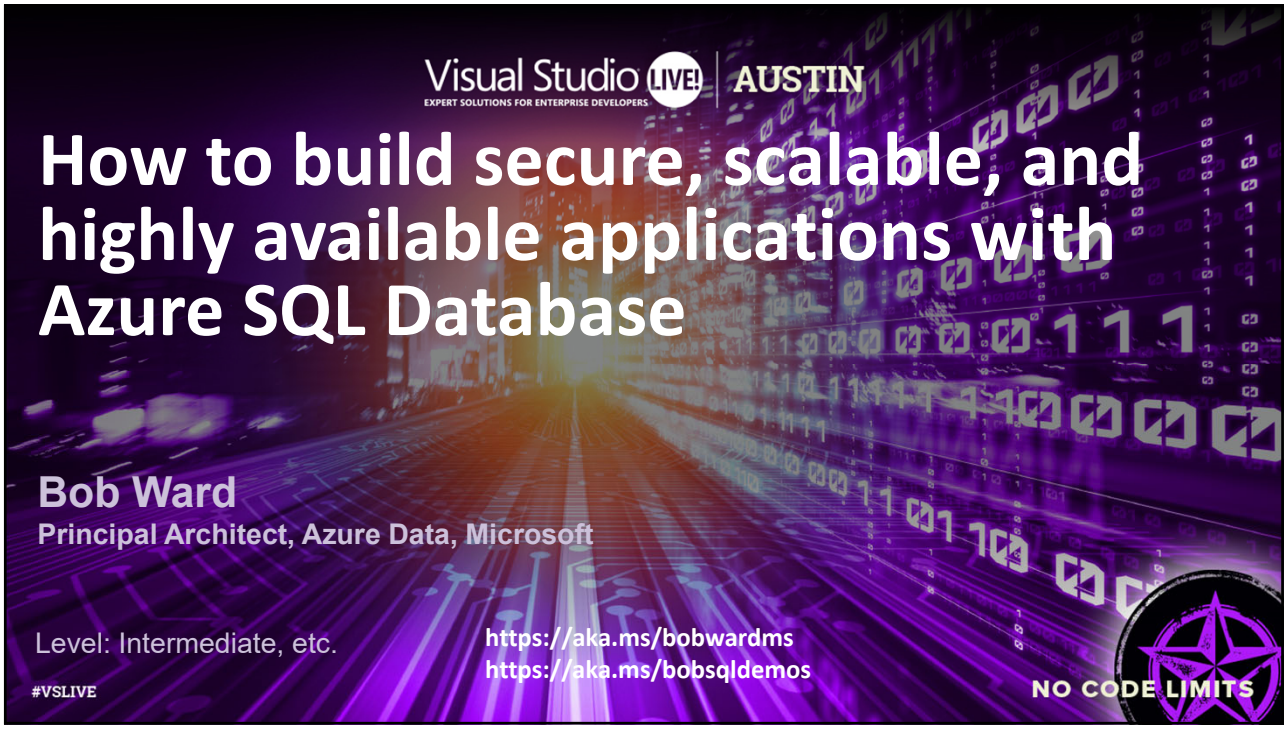


Microsoft

How to build secure, scalable, and highly available applications with Azure SQL Database

Bob Ward, Principal Architect,
Microsoft Azure Data

https://aka.ms/bobwardms
https://aka.ms/bobsqldemos

Building applications with Azure SQL Database

Foundations

Highly available

Secure

Scalable

Azure SQL

The family of SQL cloud databases

SQL Server on Azure Virtual Machines

Best for lift and shift and/or workloads requiring OS-level access

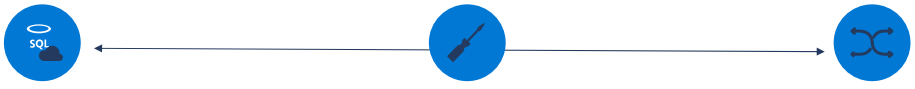
Azure SQL Managed Instance

Best for modernizing existing apps and high compatibility with SQL Server

Azure SQL Database

Best for supporting modern cloud apps with Pre-provisioned or serverless compute and Hyperscale storage

Infrastructure-as-a-Service | **Platform-as-a-Service**



Familiarity | Tools | Flexibility

Azure SQL Database Service tiers

aka.ms/getfreeazuresql

General purpose	Business critical	Hyperscale
Most business workloads	Workloads that require low latency, fast recovery, and a readable secondary	Most business workloads with highly scalable storage and read-scale requirements
Remote storage IOPS $ **Serverless**	Local storage IOPS++ $$$ In-memory	Local + remote storage IOPS+ $$ Unlimited storage

Elastic Pools

Up to 30 named replicas PREVIEW

SQL Driver choices

https://aka.ms/sqldev
https://aka.ms/sqldrivers
Drivers for ORM

ADO.NET
Microsoft.Data.SqlClient

ODBC

OLE-DB

JDBC

Node.js driver (tedious)

PHP driver for SQL

pyodbc

Ruby driver for SQL

New developer experiences

Challenge

Developers need seamless data access to easily integrate with other services and support DevOps

Building a secure application

Network Security
Public vs private endpoint
Firewalls
Virtual Networks
Private link

Transport Layer Security (TLS)
Forced by server
Minimum TLS versions

Authentication
SQL Authentication
Azure Active Directory
Managed Identities

Data Protection
Always encrypted
Dynamic Data Masking
SQL Ledger

Malicious Code
SQL Injection
Microsoft Defender for SQL

A closer look at AAD Authentication

Password	Password in clear text
Integrated	Use existing creds from domain user
Interactive	MFA
Service Principal	Application creds
Device Code Flow	Interactive with other devices
Managed Identity	More secure than with SP
Default	Inherit creds from client
Custom	Custom AAD auth provider

Avoiding SQL Injections (SQLi)

malicious code passed in the form of SQL statements

What does it look like?

Unwarranted access to data
Executing unexpected operations
Executing malicious actions

How does it happen?

Constructing SQL based on user input
Injecting malicious characters in user input
Comments are a weapon

How do you prevent it?

Use parameters
Validate user input
Avoid *dynamic* SQL
It is all about strings

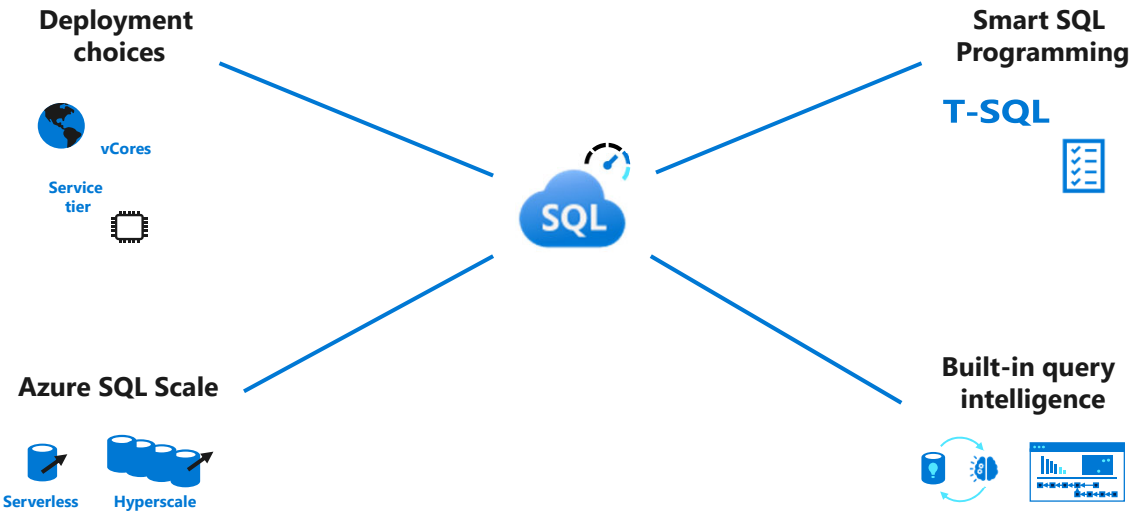
Can it be detected?

Microsoft Defender for SQL

Demo

Securing your Azure SQL Database application

Building a scalable application



Deployment choices

You always connect to a server and a database

$ DTU or vCore Purchasing Model → Choices can impact resources like memory, I/O And use of Azure Hybrid Benefit

Service Tiers

General Purpose	Business Critical	Hyperscale	Resource choices
Provisioned Per hour configured	Fastest I/O	Biggest storage (100TB)	Hardware Generation
Serverless Per second used	Built-in replicas	Best for big read-scale	#vCores
			Max Storage

Smart SQL programming

✓ Use connection pooling

Default with ADO.Net
Do you really need to connect for each query?
Connection pool fragmentation

✓ Server-side programming

T-SQL stored procedures and batches
Take advantage of the T-SQL set based language
T-SQL built-in database functions: JSON, Geospatial, STRING, … **aka.ms/tsqlfunc**

✓ Best practices **aka.ms/sqldbtips**

Process results in timely fashion
Consider async execution with ADO.Net
Use multi-statement transactions
Don't hold transactions open

Choose connection and query timeout settings
Match data types to columns
Statistics up to date
Good index design
Avoid blocking and deadlocking

Azure SQL at scale

Change scale with no migration

Go Serverless for autoscale with auto-pause/resume

Query bottlenecks got you down? Separate read-only

Sharding? Go Hyperscale with named replicas instead

aka.ms/azuresqldbscaleout

Built-in query intelligence

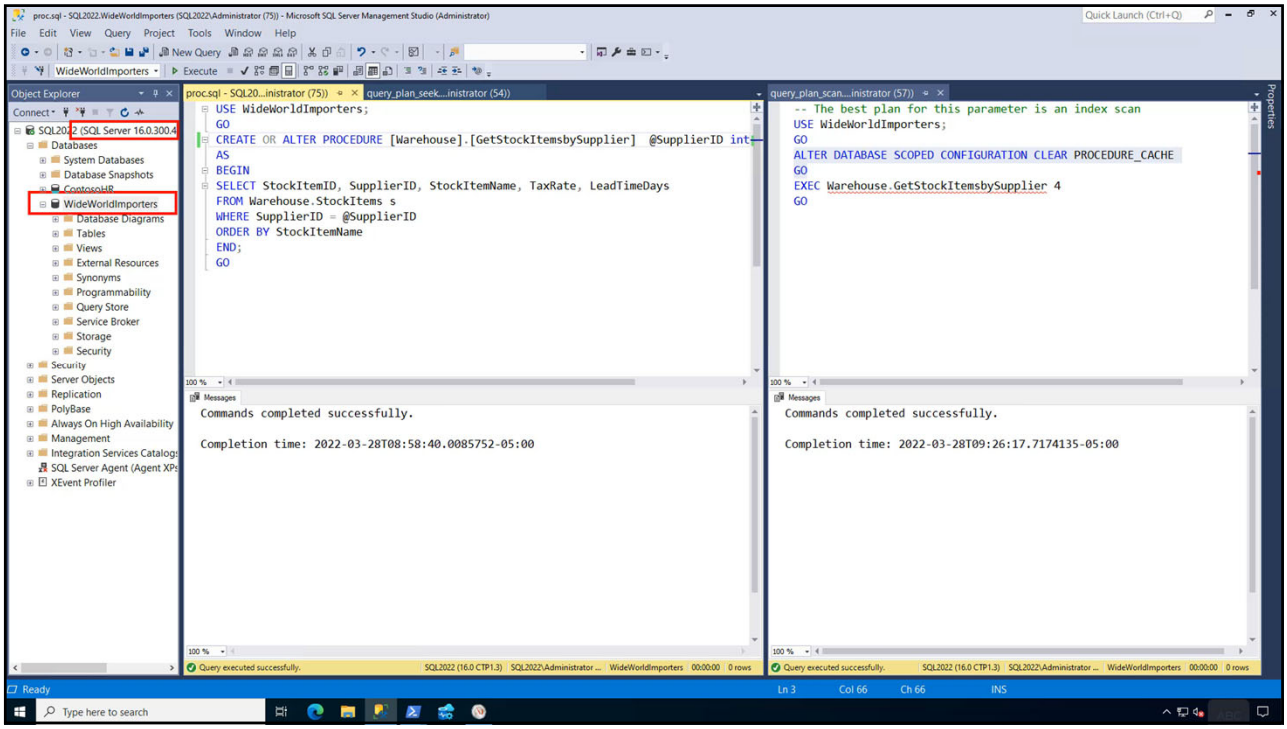
Need built-in tools?
Query Store
Extended Events for tracing

Intelligent Query Processing	Automatic Plan Correction	Automatic Tuning
Built into the engine	Last known good	Hands-free indexing

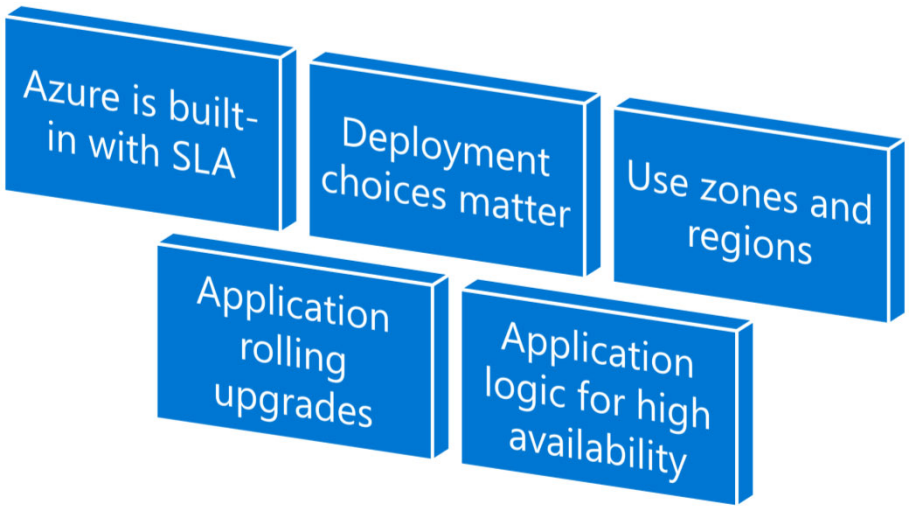
Accelerate performance with no code changes

Demo

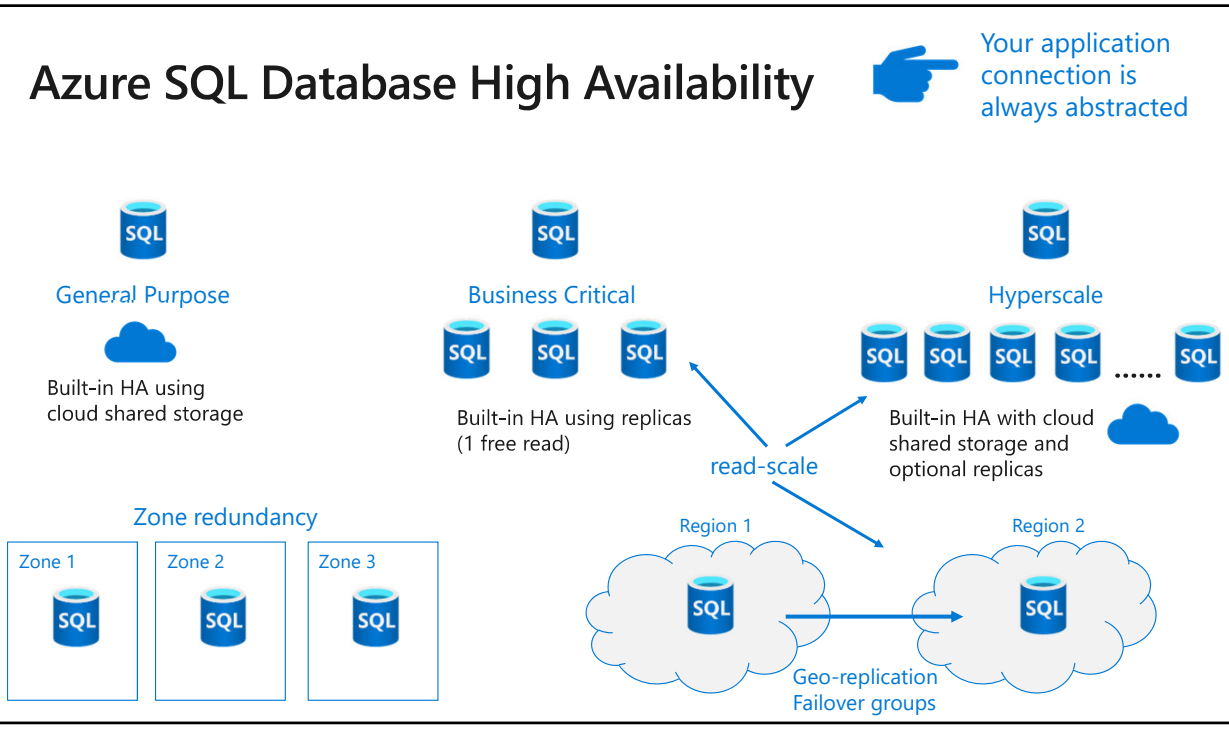
Built-in query intelligence with SQL



Building a highly available application for Azure SQL



Azure SQL Database High Availability



Application logic for high availability

- Direct or listeners
- DNS alias
- ApplicationIntent = readonly
- Built-in provider database retry logic
- Cloud application redundancy
- Azure Traffic Manager for cloud apps

Demo

Making your database application resilient

Summary

- **Focus on the database** and application with Azure SQL Database using the **provider of your choice**
- **Secure your application and data** using network security, authentication, data projection, and avoiding injections.
- **Scale your application** with the right deployment choice, read-scale, smart SQL programming, and best practices
- Ensure your **application is highly available** using tiers, redundancy, and retry-logic
- **Azure SQL Database can meet the needs** of legacy to modern cloud-born applications

Resources

https://aka.ms/bobwardms
https://aka.ms/bobsqldemos

Microsoft Learn: Azure SQL fundamentals learning path
aka.ms/azuresqlfundamentals

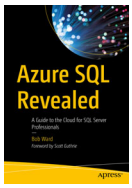
Azure SQL documentation
aka.ms/azuresqldocs

More videos from our team
aka.ms/azuresqlyt

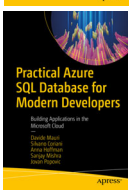
Azure SQL Database tips
aka.ms/sqldbtips

Choose your SQL driver
aka.ms/sqldrivers

Get Azure for free
aka.ms/getfreeazuresql



aka.ms/azuresqlbook



aka.ms/azuresqlfordevelopers