New Features: SQL Server 2025

Presenter: Your Name
October 2025

Agenda

01 Ground to Cloud to Fabric

02 The AI-ready enterprise database

03 Built for Developers

04 The Mission Critical Engine

05 Conclusion

Data is the fuel that powers AI

Microsoft

SQL Server 2025

The AI-ready enterprise database from ground to cloud

Best-in-class security and performance | AI built-In | Made for developers | Cloud agility through Azure

aka.ms/sqlserver2025

Building on a foundation of innovation

SQL Server 2017

SQL Server on Linux

Containers

Adaptive query processing

Automatic Tuning

Graph database

Machine learning services

SQL Server 2019

Data virtualization

Intelligent query processing

Accelerated database recovery

Data classification

SQL Server 2022

Cloud Connected

IQP NextGen

Ledger

Data Lakes

T-SQL enhancements

Microsoft SQL – ground to cloud to fabric

The AI-ready enterprise database from ground to cloud

Develop once deploy anywhere

SQL Server 2025

Azure SQL

SQL database in Fabric

T-SQL

Developers

Engine

Tools

Fabric

AI

Copilots

SQL Server 2025

The AI-ready enterprise database from ground to cloud

AI built-in

Develop modern data applications

Integrate your data with Fabric

Secure by default

Mission critical engine

Connected with Arc

Assisted by Copilots

Benchmark leader

Optimized for latest hardware

Built for all platforms

SSMS 21

Generally Available

aka.ms/ssms21

Based on Visual Studio 2022 (64-bit)

Visual Studio Installer

Copilot Public Preview

Git and TFS

Migration Assistant

Dark theme

New Connection experience

Query Editor improvements

Always Encrypted Assessment

Frequently Asked Questions (FAQ)

When with SQL Server 2025 be Generally Available?
SQL Server 2025 is planned to be Generally Available in CY25H2

Are there any pricing and licensing changes?
We will announce pricing and licensing at General Availability (GA)

What about editions?
We will announce which features will be included in specific editions at GA. We welcome all feedback during our previews[1]

How do I upgrade?
Upgrade methods will be the same as in previous releases. Supported OS and SQL versions for upgrade will be announced at GA

What about deprecated or discontinued features?
Some discontinued features have been announced. Any other deprecated or discontinued features would normally be announced at GA[2]

What is the database option PREVIEW_FEATURES?
Enables features that are in preview for SQL Server 2025 GA and will become GA at a later date

What about "BI Services?"
Power BI Report Server will replace SSRS *for all paid licenses*. SSAS has some new improvements. SSIS is still supported in SQL Server

1 Standard Developer Edition is now available

2 MDS, DQS, Synapse Link, Hot Add CPU, Purview Access Policies

The AI-ready enterprise database

Agenda

01 The AI-ready enterprise database

02 Vector Store & Indexes

03 Vector Search & Embeddings

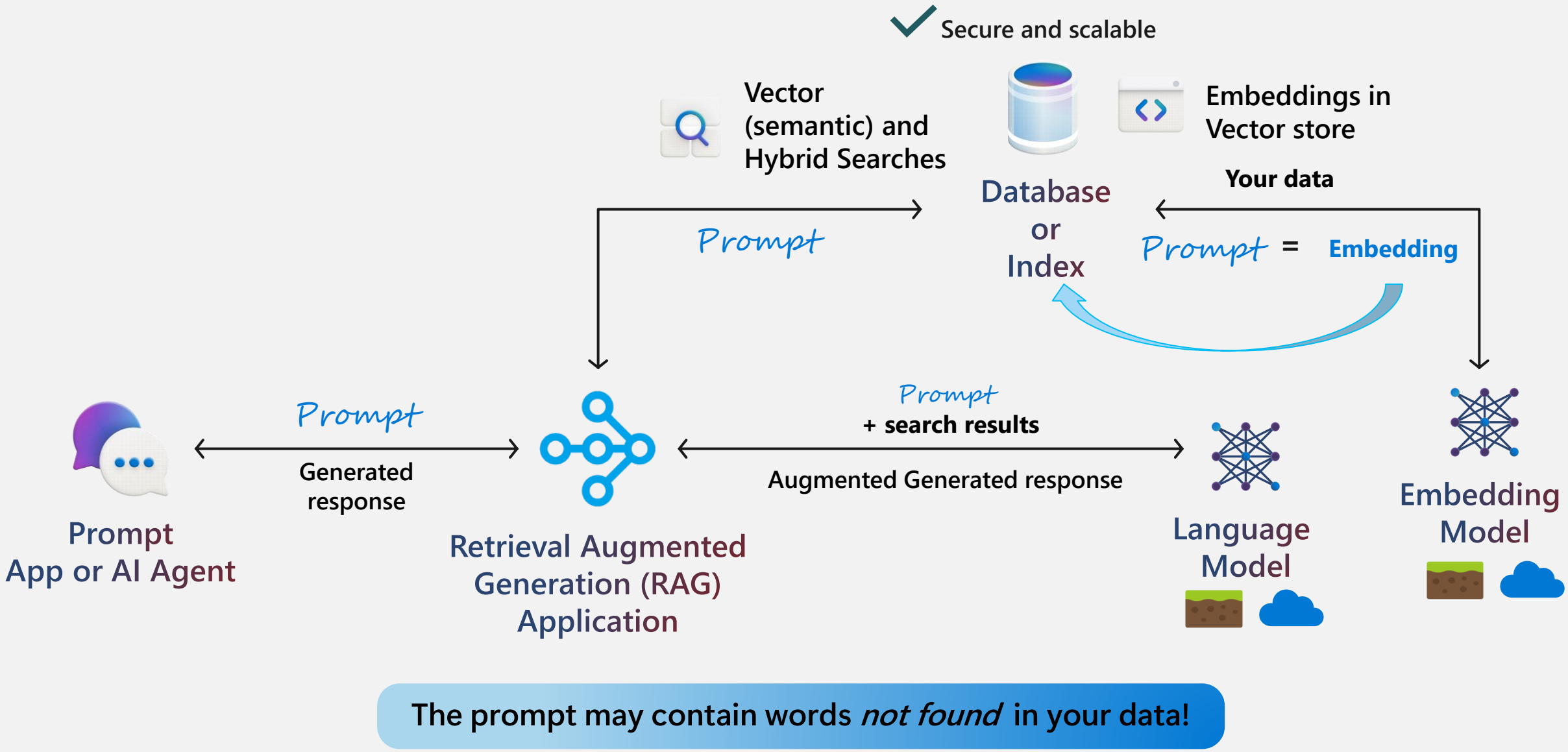
04 Demo: Intelligent Search in SQL Server 2025

05 Conclusion

What problems are we trying to solve with AI?

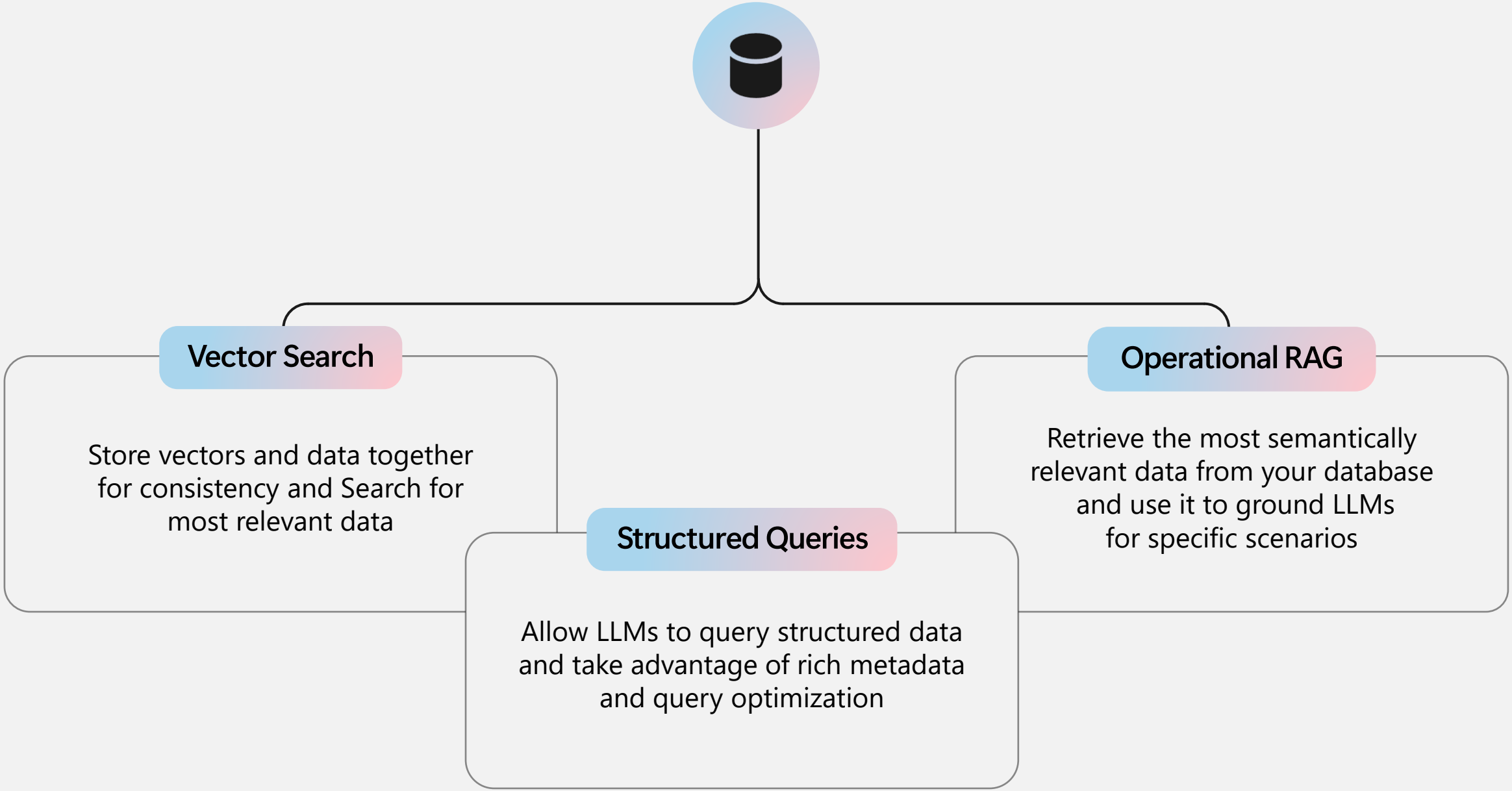
- Smarter searching on your existing text data
- Bring in other documents/text for centralized vector searching
- Provide building blocks - Intelligent assistants, RAG, AI Agents
- Take advantage of AI in a secure and scalable fashion
- Overcome complexity by using the familiar T-SQL language

Vector search with your data



The prompt may contain words *not found* in your data!

Building scalable AI applications: Agentic RAG



Build Enterprise AI-ready applications ground to cloud

Build Agentic RAG patterns *Inside* the engine

Vector Store
Native vector data type and DiskANN index*

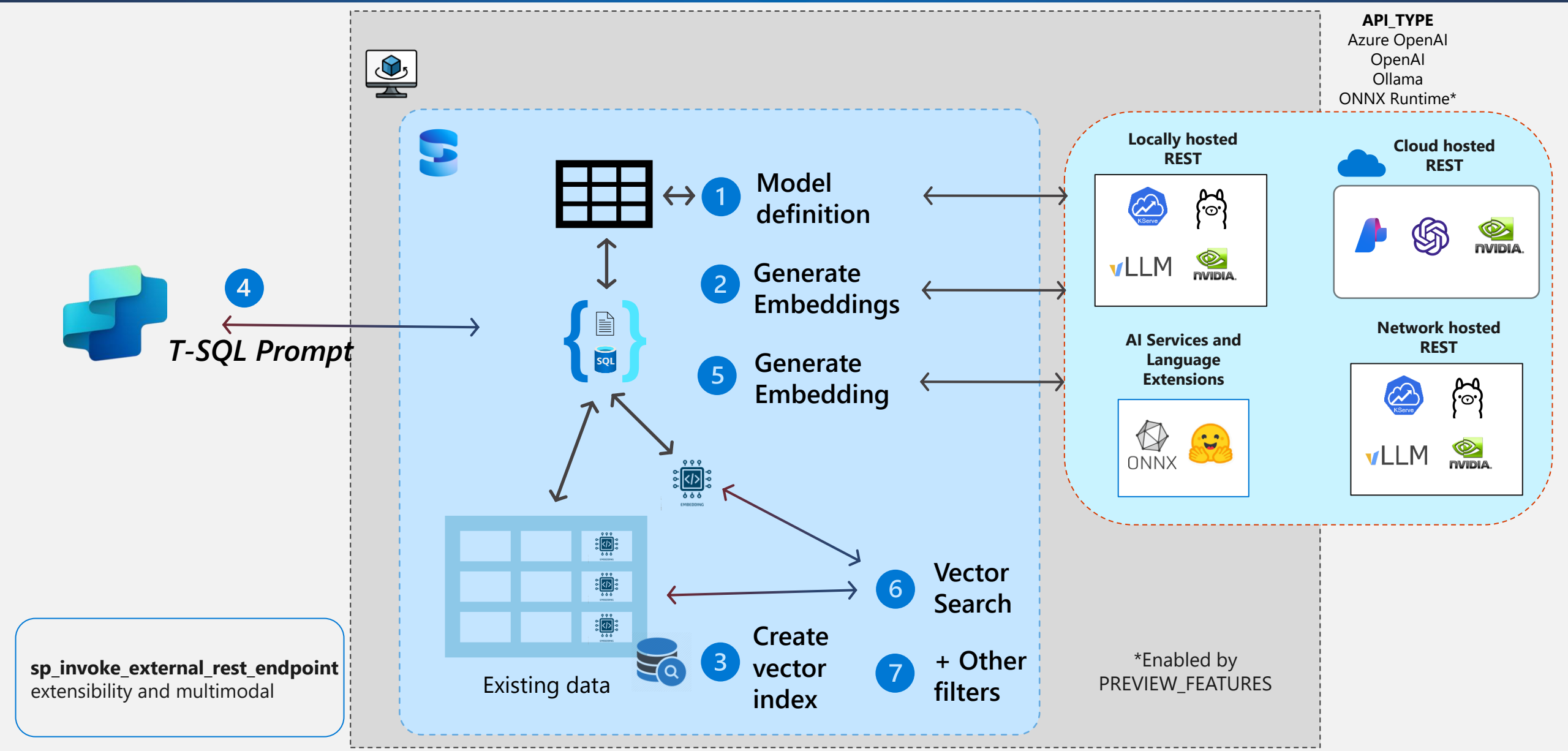
Model Management
Declarative Model definitions ground/cloud

Embeddings built-in
Text Chunking* and built-in multimodal embedding generation

Simple semantic searching
Vector distance (KNN) and Vector search* (ANN)

Framework integration
LangChain, Semantic Kernel, EF Core

SQL Server 2025 Vector Architecture



Security, SQL, and AI

- You control *all access* with SQL security
- You control *which* AI models to use
- AI models ground *and/or* cloud *isolated* from SQL
- Use RLS, TDE, and Dynamic Data Masking
- Track *everything* with SQL Server Auditing
- Ledger for chat history and feedback

What is a Vector Store in SQL Server 2025?

A **Vector store** refers to the ability to store high-dimensional numerical arrays—called vectors—directly in SQL Server tables. These vectors typically represent embeddings generated by AI models from text, images, or other data types.

Data Type: SQL Server 2025 introduces a native VECTOR(n) data type, where n is the number of dimensions (up to 1998). Vectors are stored in an optimized binary format but exposed as JSON arrays for convenience

Use Cases

- Semantic search
- Recommendation systems
- Fraud detection
- Natural Language Processing pipelines
- Machine Learning pipelines
- Retrieval-Augmented Generation (RAG) patterns

What Are Vector Indexes?

Vector indexes are specialized structures that allow fast similarity searches over vector data. SQL Server 2025 supports approximate nearest neighbor (ANN) search using the DiskANN algorithm.

```
CREATE VECTOR INDEX vec_idx ON Articles(embedding)
      WITH (METRIC = 'cosine', TYPE = 'diskann’);
Supported metrics: 'cosine', 'dot', 'euclidean’.
```

Catalog View: You can inspect vector indexes using sys.vector_indexes, which shows metadata like index type and distance metric

Vector Metrics

In SQL Server 2025, when creating a **vector index** for approximate nearest neighbor (ANN) search, you can choose from **three distance metrics** to measure similarity between vectors.

- Cosine Distance
- Euclidean Distance
- Dot Product (Negative)

These metrics define how "closeness" is calculated between the query vector and the stored vectors in your database.

Three Distance Metrics

Cosine Distance

- **Definition**: Measures the cosine of the angle between two vectors.
- **Use Case**: Ideal for **text embeddings** and **semantic similarity** where the magnitude of the vectors is less important than their orientation.
- **Interpretation**: A value closer to 0 means the vectors are more similar.

Euclidean Distance

- **Definition**: Measures the straight-line (L2 norm) distance between two vectors in space.
- **Use Case**: Useful when **absolute differences** in vector values matter, such as in image or sensor data.
- **Interpretation**: Smaller values indicate closer vectors.

Dot Product (Negative)

- **Definition**: Computes the dot product of two vectors, often used in machine learning for similarity scoring.
- **Use Case**: Effective when vectors are normalized or when you want to emphasize **directional alignment**.
- **Interpretation**: Higher dot product values indicate stronger similarity, but SQL Server uses the **negative dot product** as a distance metric, so lower values are better.

How Vector Search Works

SQL Server 2025 supports two types of vector search:

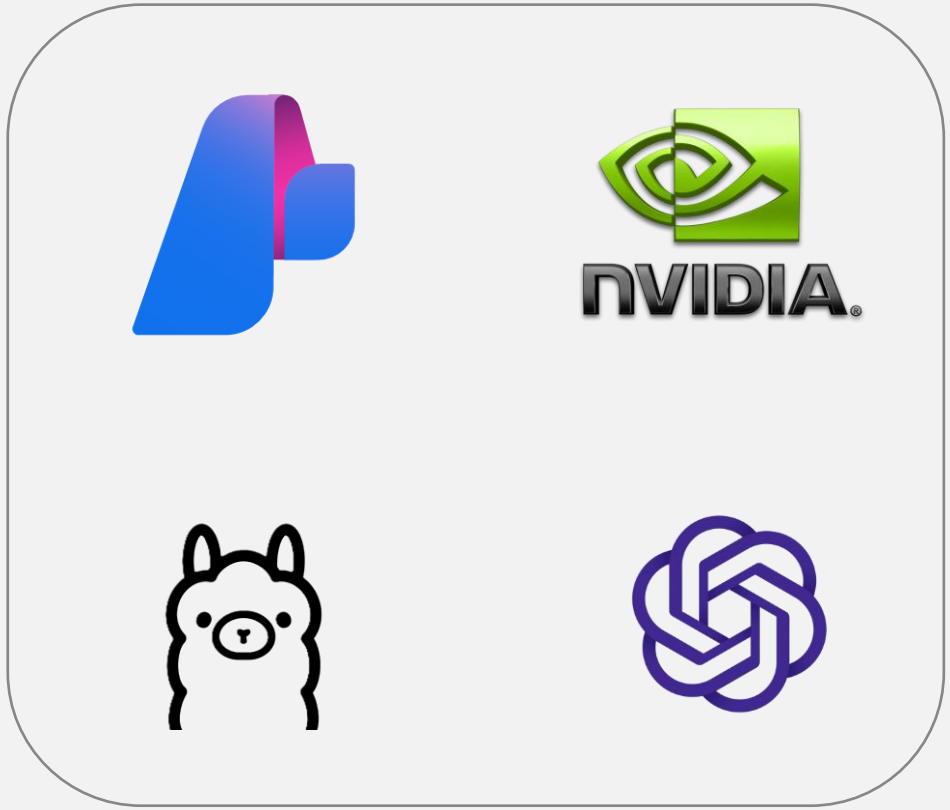
Exact Search (VECTOR_DISTANCE()):	Approximate Search (VECTOR_SEARCH()):
• Calculates distance between a query vector and all stored vectors. Suitable for small datasets (<50,000 vectors). • Resource-intensive.	• Uses vector indexes to find similar vectors quickly. Ideal for large datasets. • Requires enabling **PREVIEW_FEATURES** configuration.

What Is AI Model Management in SQL Server 2025?

SQL Server 2025 introduces native support for managing external AI models directly within the database engine.

This includes:

- Registering models via T-SQL
- Invoking models for inference tasks (e.g., embeddings, classification)
- Versioning and switching providers without changing application code
- Local hosting via ONNX runtime for air-gapped or high-security environments



What Are Embeddings?

In AI, **embeddings** are dense vector representations of data—such as text, images, or audio—that capture semantic meaning.

For example, words with similar meanings will have embeddings that are close together in vector space.

This allows AI systems to perform **semantic search**, **recommendation**, and **classification** tasks more effectively than traditional keyword-based methods.

Embeddings in SQL Server 2025

SQL Server 2025 introduces native support for embeddings through several key features:

AI_GENERATE_EMBEDDINGS Function

- This new T-SQL function allows you to generate embeddings directly within SQL Server using a pre-defined AI model stored in the database.

Vector Data Type

- SQL Server 2025 supports a **native vector data type**, allowing you to store and query embeddings directly in the database. This eliminates the need for external vector databases like Pinecone or FAISS

Practical Use Cases

Semantic Search

- Embed product descriptions or support tickets and use vector similarity to find relevant matches.

AI Feature Storage

- Store embeddings from third-party AI models alongside metadata in SQL Server.

Prototyping

- Experiment with AI use cases without investing in a dedicated vector database.

Agentic RAG Applications

- Build intelligent apps that combine structured SQL queries with LLMs for dynamic responses.

Demo: Intelligent Search in SQL Server 2025

Built for Developers

Agenda

01 New Features for Developers

02 JSON Data Type and Indexes

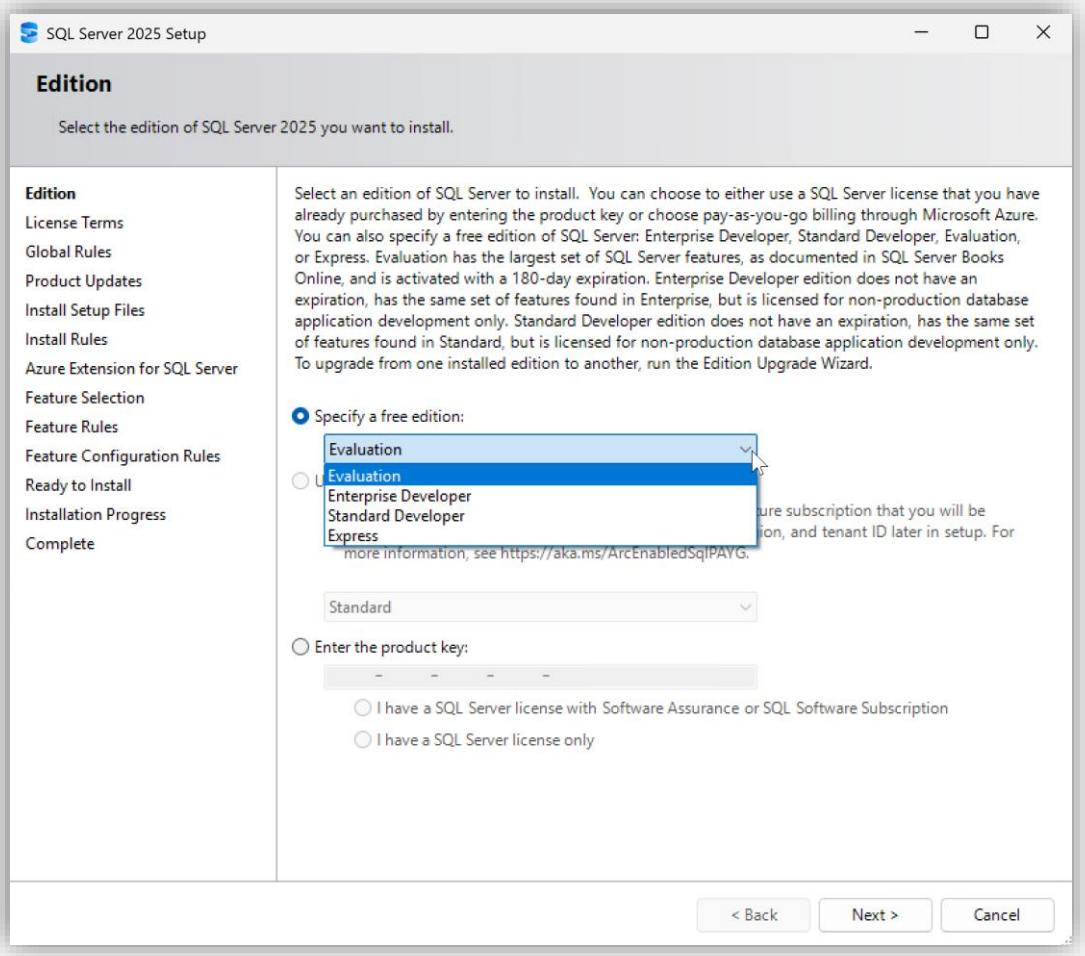
03 New T-SQL Functions

04 Fabric Mirroring

05 Conclusion

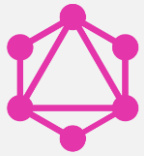
Developer Edition

- ✓ Choose your target production deployment: Standard or Enterprise
- ✓ Develop and test with no license costs
- ✓ Features and limits match target deployment



Built for developers

Develop modern data applications



GraphQL

Use Data API builder
for efficient data applications

JSON

JSON type, index,
and updated T-SQL
functions

T-SQL

RegEx and other
T-SQL functions

Change Event Streaming*

Consume log change
events

REST API

Connect your data
to ***any*** REST interface
including GraphQL

{ REST }

Announcing

GitHub Copilot for SQL

Microsoft Python Driver
for SQL Server

* Enabled by PREVIEW_FEATURES

Demo:
GitHub Copilot for SQL

JSON Data Types and Indexes

Before SQL Server 2025

varchar/nvarchar column

Any index that supports varchar/nvarchar

Various functions for JSON

SQL Server 2025

json data type – binary storage up to 2GB

json index

+ JSON_OBJECTAGG and JSON_ARRAYAGG

+ JSON_CONTAINS

+ JSON_QUERY WITH ARRAY WRAPPER

Save space with compressed storage
Less I/O and page reads

Demo:
Native JSON

File Edit View Query Git Tools Extensions Window Help Search Solution1

New Query Execute Copilot MAIN

orders

Object Explorer

Connect

- . (SQL Server 17.0.600.9 - sqladmin)
  - Databases
    - System Databases
    - Database Snapshots
  - Security
  - Server Objects
  - Replication
  - Always On High Availability
  - Management
  - SQL Server Agent (Agent XPs disabled)
  - XEvent Profiler

json_ty...in (53))*

```
USE master;
GO
DROP DATABASE IF EXISTS orders;
GO
CREATE DATABASE orders;
GO
USE orders;
GO
DROP TABLE IF EXISTS dbo.Orders;
GO
CREATE TABLE dbo.Orders(
order_id int NOT NULL IDENTITY,
order_info json NOT NULL
);
GO

-- INSERT JSON documents directly into the JSON type
```

100 % Ln: 10 Ch: 3 TABS CRLF

Messages

```
Commands completed successfully.

Completion time: 2025-04-21T09:36:50.7194444-05:00
```

100 % Ln: 4 Ch: 1 TABS MIXED

Query executed successfully. (local) (17.0 CTP1.5) sqladmin (53) orders 00:00:06 0 rows

Ready

Search

File Edit View Query Git Tools Extensions Window Help Search Solution1

New Query Execute contactsdb Copilot MAIN

Object Explorer

Connect . (SQL Server 17.0.600.9 - sqladmin)

- Databases
  - System Databases
  - Database Snapshots
  - orders
- Security
- Server Objects
- Replication
- Always On High Availability
- Management
- SQL Server Agent (Agent XPs disabled)
- XEvent Profiler

json_typ...in (53)) json_ind...in (62))

```
USE master;
GO
DROP DATABASE IF EXISTS contactsdb;
GO
CREATE DATABASE contactsdb;
GO
USE contactsdb;
GO

-- Create a new database with a JSON type to hold the document
DROP TABLE IF EXISTS contacts;
GO
CREATE TABLE contacts (id INT IDENTITY PRIMARY KEY, jdoc JSON);
GO

-- Create a JSON index
DROP INDEX IF EXISTS [ji_contacts] ON contacts;
```

100 % Ln: 8 Ch: 3 SPC CRLF

Messages

```
Commands completed successfully.

Completion time: 2025-04-21T09:59:29.4225121-05:00
```

100 % Ln: 4 Ch: 1 TABS MIXED

Query executed successfully. (local) (17.0 CTP1.5) sqladmin (62) contactsdb 00:00:00 0 rows

Ready

Search

New T-SQL Functions

Microsoft

Some T-SQL Love

RegEx
functions

PRODUCT()

BASE64
functions

Fuzzy string match
functions

CURRENT_DATE
returns DATE

SUBSTRING()
optional length

UNISTR()

||
string concat operator

DATEADD()
supports bigint

Native Regex Functions in SQL Server 2025

Function	Description	Example Use Case
REGEXP_LIKE	Returns TRUE/FALSE if a string matches a regex pattern.	Validate email addresses
REGEXP_REPLACE	Replaces occurrences of a regex pattern in a string with a replacement string.	Mask sensitive data
REGEXP_SUBSTR	Extracts substring(s) matching a regex pattern.	Extract domain from email
REGEXP_INSTR	Returns the position of a regex match in a string.	Find where a pattern starts
REGEXP_COUNT	Counts the number of times a regex pattern occurs in a string.	Count digits in a string
REGEXP_MATCHES	Returns a table of all substrings matching a regex pattern.	Get all matches in a text column
REGEXP_SPLIT_TO_TABLE	Splits a string into rows using a regex delimiter.	Split CSV or log line

Performance & SARGability

SARGable Patterns: Some regex functions (like REGEXP_LIKE) are SARGable when the pattern starts with an anchor (^) or uses quantifiers (*, +, {m,n}), allowing index seeks for efficient queries.

RE2 Library: SQL Server 2025 uses the RE2 regex library, known for speed and safety.

Regex Functions Demo

What Is the PRODUCT() Function?

PRODUCT() is a new aggregate function introduced in SQL Server 2025. It calculates the product (multiplication) of all values in a numeric column or expression for a group of rows, similar to how SUM() adds values or AVG() computes the average.

Key Points

Null Handling: If any value in the group is NULL, the result is NULL (unless you use COALESCE or filter out nulls).

Use Cases: Useful for mathematical, statistical, and financial calculations where you need to multiply values across rows (e.g., compound interest, probability calculations, inventory scenarios).

Window Functions: Can be used as a windowed aggregate for running products.

Product Function Demo

What Is Base64 Encoding?

Base64 is a method for converting binary data into a text format using 64 ASCII characters (A–Z, a–z, 0–9, +, /).

It's commonly used for:

- Email encoding
- Embedding data in XML/JSON
- Transmitting binary files over text-only protocols

BASE64_ENCODE Function

BASE64_ENCODE(expression [, url_safe])

expression: Must be of type VARBINARY(n) or VARBINARY(MAX)

url_safe: Optional. If set to a non-zero value, uses a URL-safe alphabet (RFC 4648 Table 2) without padding.

Return Types:
VARCHAR(8000) if input is ≤ 6000 bytes
VARCHAR(MAX) otherwise

BASE64_DECODE Function

BASE64_DECODE (expression)
expression: Must be of type **VARCHAR(n)** or **VARCHAR(MAX)**

Return Types:
VARBINARY(8000) or **VARBINARY(MAX)** depending on input size

BASE64 Functions Demo

Built-in Fuzzy Match Functions in SQL Server 2025

EDIT_DISTANCE
Purpose: Calculates the number of insertions, deletions, substitutions, and transpositions needed to transform one string into another.
Use Case: Detecting how different two strings are.

EDIT_DISTANCE_SIMILARITY
Purpose: Returns a similarity score from 0 to 100, where 100 means identical strings.
Use case: Ranking string similarity for fuzzy searches.

JARO_WINKLER_DISTANCE
Purpose: Measures edit distance with preference for strings that match from the beginning.
Use Case: Useful for short strings like names.

JARO_WINKLER_SIMILARITY
- **Purpose**: Returns a similarity score from **0 to 1**, where 1 means perfect match.
- **Use Case**: Detecting close matches with prefix weighting.

Fuzzy Match Functions Demo

CURRENT_DATE, UNISTR(), SUBSTRING(), DATEADD()

What CURRENT_DATE Does
Returns the **current date** from the operating system where the SQL Server Database Engine is running. Emulates **CAST(GETDATE() AS DATE)**

What UNISTR() Does It interprets Unicode escape sequences (e.g., \XXXX) within a string and returns the corresponding Unicode characters.

SUBSTRING() for SQL Server 2025
Optional Length Parameter Change: The length parameter is now optional.
Behavior: If omitted, the function returns the substring from the specified start position to the end of the string.

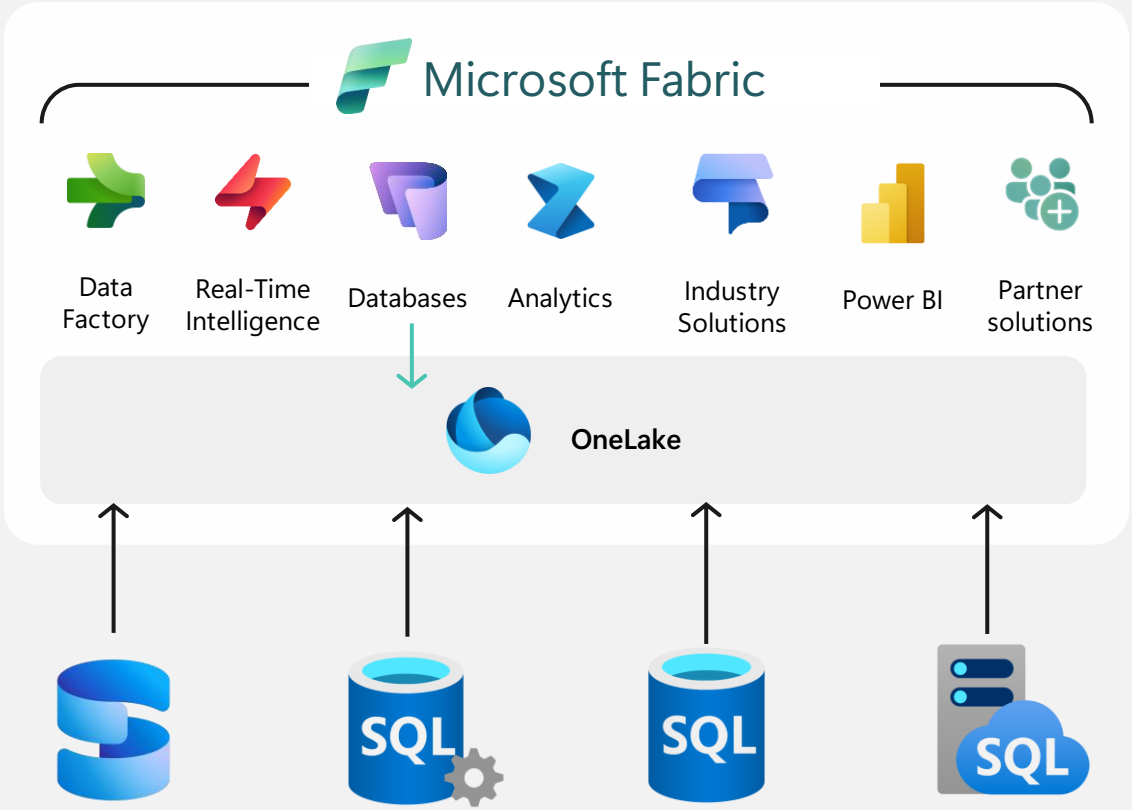
What Does || Do?
The || operator joins two or more strings into a single string value. It is equivalent to using the + operator or the CONCAT() function in previous versions of SQL Server.

Misc functions Demo

Fabric Mirroring

Integrated with Microsoft Fabric

Integrate your operational data with a unified data platform



Mirroring **replicates databases** to **Fabric with zero ETL**

Data is replicated into One Lake and **kept up-to-date in near real-time**

Mirroring **protects operational databases** from **analytical** queries

Compute replication is included with your Fabric capacity **for no cost.**

Free Mirroring Storage for Replicas tiered to Fabric Capacity.

The Mission Critical Engine

Agenda

01 Security Features Overview

02 Performance Features Overview

03 High Availability and Disaster Recovery Features

04 Conclusion

Security Features Overview

Enhancing the industry proven engine

Security

- Security cache improvements
- OAEP support for encryption
- PBKDF password hashing
- Authentication using system-assigned managed identity
- Backup to URL with managed identity
- Managed identity support for EKM
- Managed identity for AI models
- Entra logins with nonunique display names
- Custom password policy on Linux
- TDS 8.0/TLS 1.3 support for tools

Performance

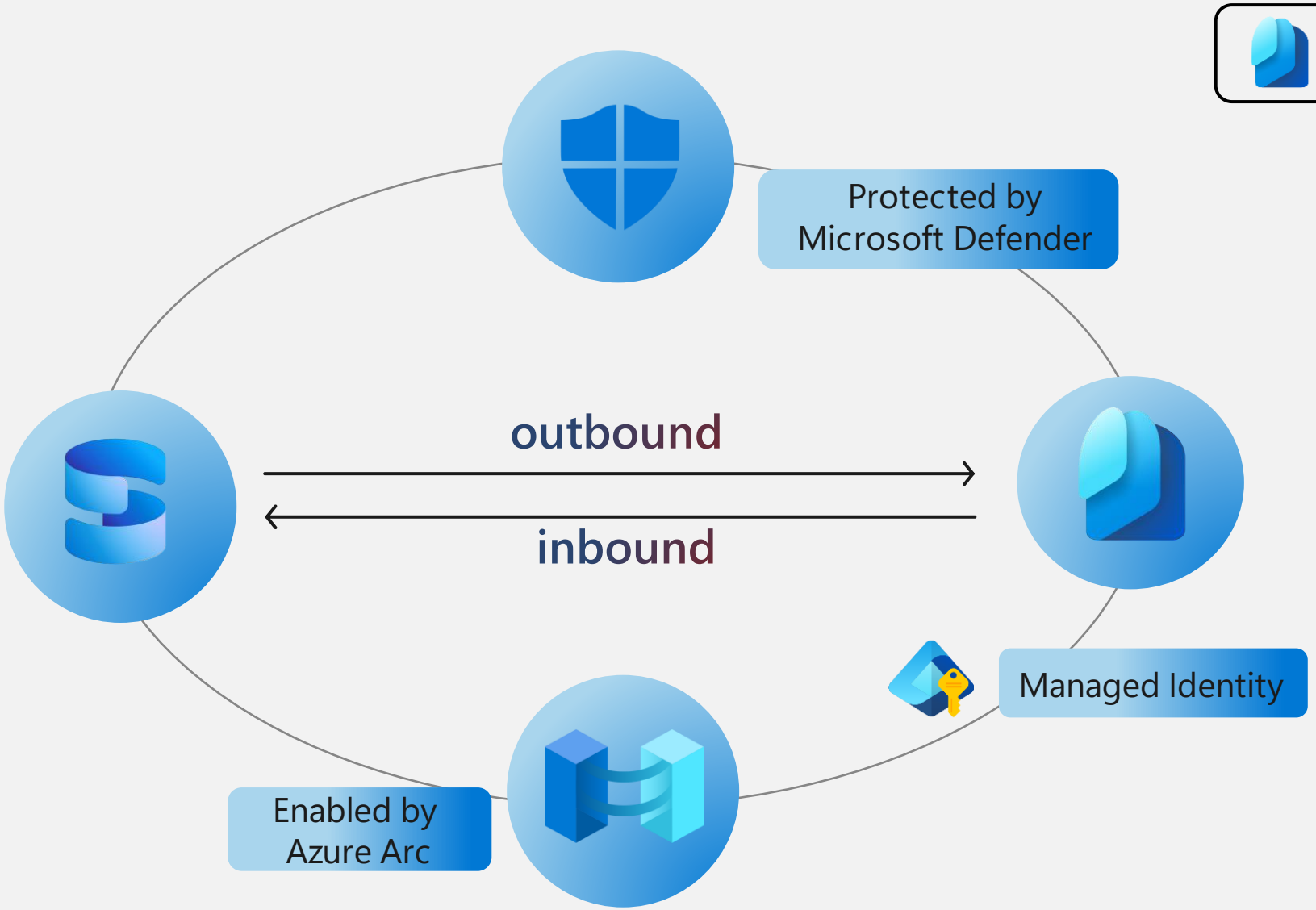
- Optimized Locking
- Tempdb space resource governance
- ADR in tempdb
- Persisted stats for readable secondaries
- Change tracking cleanup
- Columnstore index maintenance
- CE feedback for expressions
- Optional parameter plans optimization
- DOP feedback on by default
- Optimized Halloween protection
- Query store for readable secondaries
- ABORT_QUERY_EXECUTION query hint
- Optimized sp_executesql
- Batch mode optimizations
- Remove In-Memory OLTP from a database
- tmpfs support for tempdb in Linux

HADR

- Fast failover for persistent AG Health
- Async page request dispatching
- Improved health diagnostics
- Communication control flow tuning
- Switching to resolving state
- Remove listener IP address
- NONE for routing
- AG group commit waiting tuning
- Contained AG support for DAG
- DAG sync improvements
- Backups on secondary replicas
- ZSTD Backup compression
- Backup to Azure immutable storage

Secure by default

Provide the latest security capabilities for your data



SQL Server 2025 enabled by Azure Arc

With **SQL Server 2025 enabled by Azure Arc**, managed identity support is available for both inbound and outbound authentication:

Inbound: Users and applications can connect to SQL Server using Microsoft Entra authentication (formerly Azure AD), leveraging managed identities.

Outbound: SQL Server can connect to Azure resources (e.g., backup to URL, PolyBase, Key Vault) using its managed identity, eliminating the need for hard-coded secrets.

How Does Managed Identity Work?

When you connect your SQL Server instance to Azure Arc, a system-assigned managed identity is automatically created for the server.

This identity is associated with the SQL Server instance and the Microsoft Entra tenant ID (Azure AD).

The managed identity is used for authentication to Azure services, and permissions are managed via Azure role assignments (e.g., Storage Blob Data Contributor for Blob Storage).

Only system-assigned managed identities are supported (not user-assigned).

The managed identity is assigned at the Azure Arc server level.

Outbound connections (e.g., backup to URL, PolyBase) require the primary managed identity to be assigned to SQL Server.

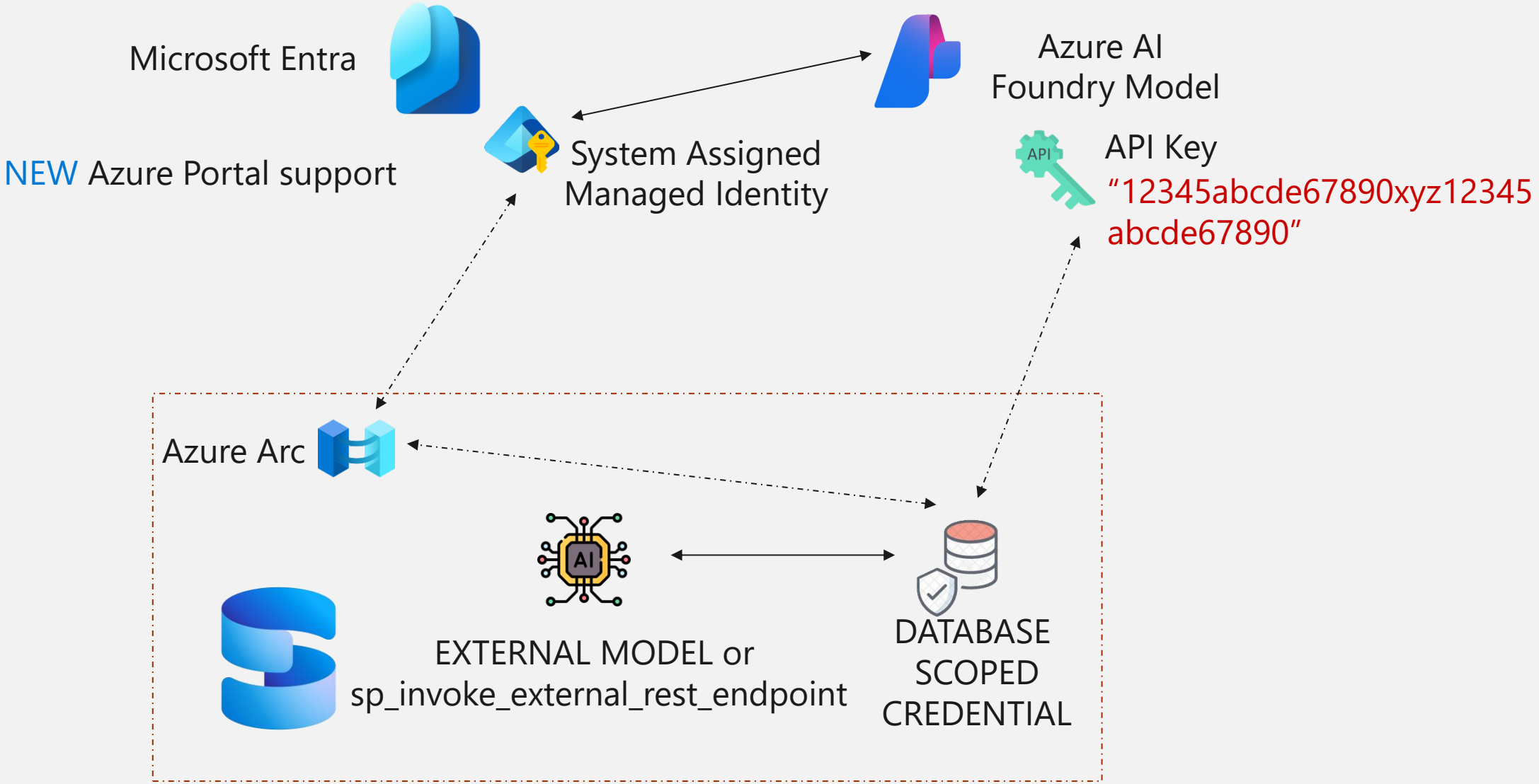
Example Scenarios

Backup to URL
You can back up and restore databases to Azure Blob Storage using managed identity. The SQL Server instance must have the Storage Blob Data Contributor role assigned to its managed identity.

PolyBase
PolyBase can use managed identity to access Azure Blob Storage and Data Lake, provided the SQL Server instance is enabled by Azure Arc and properly configured.

Azure Key Vault
SQL Server can access secrets in Azure Key Vault using its managed identity, simplifying credential management for Extensible Key Management (EKM)

Go *passwordless* with a Managed Identity



Issue: Nonunique Display Names in Microsoft Entra ID

Microsoft Entra ID (formerly Azure Active Directory) allows **duplicate display names** for service principals and groups.

This flexibility, while useful in identity management, causes issues in SQL Server where **login and user names must be unique**.

SQL Server 2025 introduces the WITH OBJECT_ID clause in CREATE LOGIN and CREATE USER statements to resolve this conflict.

```
CREATE LOGIN [login_alias] FROM EXTERNAL PROVIDER WITH OBJECT_ID = 'objectid-guid'
```

```
CREATE USER [user_alias] FROM EXTERNAL PROVIDER WITH OBJECT_ID = 'objectid-guid'
```

Custom Password Policy in SQL Server 2025 on Linux?

Starting with **SQL Server 2025 (17.x)**, administrators can enforce **custom password policies** for SQL Server authentication logins on Linux systems.

This feature brings parity with Windows environments and enhances security by allowing fine-grained control over password rules

You can define the following parameters in the mssql.conf file:

Parameter	Description
passwordpolicy.passwordminimumlength	Minimum number of characters (up to 128)
passwordpolicy.passwordhistorylength	Number of previous passwords remembered
passwordpolicy.passwordminimumage	Minimum time before a password can be changed again
passwordpolicy.passwordmaximumage	Maximum time a password can be used before it must be changed

Improved Security Cache

The **security cache** stores permission data for users and logins across various securable objects (like tables, schemas, and columns). This cache speeds up query execution by avoiding repetitive permission checks.

Login-Specific Cache Invalidation

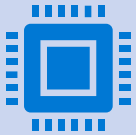
- Previously, any permission change could invalidate the entire security cache, impacting all users and degrading performance.

SQL Server 2025 introduces **targeted cache invalidation**:

- Only the cache entries for the **affected login** are invalidated.
- Other users' tokens remain untouched.
- Applies to CREATE, ALTER, and DROP LOGIN actions, and individual permission changes.
- **Group logins** still trigger server-wide invalidation.

What Is OAEP Support in SQL Server 2025?

OAEP (Optimal Asymmetric Encryption Padding) is a cryptographic padding scheme used with RSA encryption to enhance security.



Starting with **SQL Server 2025**, Microsoft introduced **OAEP-256 support** for RSA-based encryption, replacing the older **PKCS#1 v1.5** padding mode

This change improves resistance against chosen plaintext and ciphertext attacks by adding internal redundancy and hashing between the RSA encryption and padding check. It's a major step forward in securing encrypted data, especially for certificates and asymmetric keys.

Where OAEP-256 Is Applied?

OAEP-256 is now supported in the following encryption contexts:

- **Built-in functions**: EncryptByCert, DecryptByCert, EncryptByAsymKey, DecryptByAsymKey
- **Symmetric key encryption**: When encrypted by a certificate or asymmetric key
- **Database encryption key**
- **Always Encrypted**: OAEP is used for encrypting column encryption keys with column master keys

These enhancements are available **only** when the database is set to **compatibility level 170 or higher**.

OAEP-256 Configuration and Trace Flags

SQL Server 2025 allows fine-grained control over OAEP support via trace flags:

- **TF 15023**: Disable OAEP-256 for built-in functions
- **TF 4669**: Disable OAEP-256 for symmetric key encryption
- **TF 15027**: Disable OAEP-256 for database encryption key

These flags are useful for maintaining backward compatibility with older SQL Server versions while still operating at compatibility level 170.

PBKDF2 Pasword Hashing

PBKDF2 is a cryptographic algorithm designed to make password hashes more resistant to brute-force attacks.

- It works by using a **SHA-512 hash function**.
- Applying **100,000 iterations** of hashing.
- Combining the password with a **unique 32-bit salt**.
- This dramatically increases the time required for attackers to guess passwords, even with powerful hardware

TDS 8.0/TLS 1.3 support for tools

What Are TDS 8.0 and TLS 1.3?

TDS (Tabular Data Stream) 8.0 is the newest version of the protocol used for communication between SQL Server and client tools. It brings enhanced security and performance, especially when paired with **TLS 1.3**.

TLS 1.3 (Transport Layer Security) is the latest encryption protocol, offering faster handshakes, stronger security, and removal of outdated cryptographic algorithms

Key Benefits

End-to-End Encryption: With TDS 8.0, the entire connection sequence—including prelogin, authentication, and data exchange—is encrypted from the start, unlike previous versions where prelogin was in cleartext.

Performance: TLS 1.3 reduces handshake round trips, making connections faster and more secure.

Security: TLS 1.3 removes legacy ciphers (RC4, SHA-1, MD5, etc.), making SQL Server connections more resilient against modern attacks.

Supported Tools and Features in SQL Server 2025

The following tools and features now support TDS 8.0 and TLS 1.3:

SQL Server Agent: Uses the latest ODBC driver (MSODBCSQL v18+) for secure connections. Encryption is mandatory by default, and you can verify encryption status using system views.

Linked Servers: Leverages OLE DB v19 and TDS 8.0 for secure distributed queries. The new Encrypt=Strict connection option enforces TLS 1.3 encryption.

sqlcmd & bcp utilities: Command-line tools now support TDS 8.0/TLS 1.3 for secure scripting and data movement.

Always On Availability Groups & Failover Cluster Instances (FCI): Cluster and replica communication can be encrypted with TLS 1.3 and TDS 8.0.

Transactional, Merge, and Snapshot Replication: Replication traffic is now secured with TLS 1.3/TDS 8.0.

Log Shipping: Inter-server communication for log shipping is encrypted.

PolyBase, SQL VSS Writer, SQL CEIP Service: These services also benefit from the new protocol and encryption support.

How to Enable and Use TDS 8.0 and TLS 1.3

Certificates: You must use a valid CA-signed certificate

Self-signed certificates are not sufficient for strict encryption

The certificate must have the correct Subject Alternative Name for listeners and replicas.

Drivers: Ensure you have ODBC Driver 18 or OLE DB Driver 19 installed.

These are required for strict encryption and TDS 8.0 support.

Connection Strings: Use the Encrypt=Strict parameter in your connection string to enforce TLS 1.3 and TDS 8.0.

Previous options like Encrypt=True/False are replaced with Strict for mandatory encryption.

Additional Notes

Backward Compatibility: TDS 8.0 is compatible with TLS 1.2, but TLS 1.3 is recommended for maximum security.

SSMS 20.0+: The latest SQL Server Management Studio supports strict encryption and TDS 8.0, with improved connection dialogs for security settings.

Linux Support: TLS 1.3 is enabled by default on supported Linux distributions (e.g., Ubuntu 24.04).

Performance Features Overview

Enhancing the industry proven engine

Security

- **Security cache improvements**
- **OAEP support for encryption**
- **PBKDF password hashing**
- **Authentication using system-assigned managed identity**
- **Backup to URL with managed identity**
- **Managed identity support for EKM**
- **Managed identity for AI models**
- **Entra logins with nonunique display names**
- **Custom password policy on Linux**
- **TDS 8.0/TLS 1.3 support for tools**

Performance

- Optimized Locking
- Tempdb space resource governance
- ADR in tempdb
- Persisted stats for readable secondaries
- Change tracking cleanup
- Columnstore index maintenance
- CE feedback for expressions
- Optional parameter plans optimization
- DOP feedback on by default
- Optimized Halloween protection
- Query store for readable secondaries
- ABORT_QUERY_EXECUTION query hint
- Optimized sp_executesql
- Batch mode optimizations
- Remove In-Memory OLTP from a database
- tmpfs support for tempdb in Linux

HADR

- Fast failover for persistent AG Health
- Async page request dispatching
- Improved health diagnostics
- Communication control flow tuning
- Switching to resolving state
- Remove listener IP address
- NONE for routing
- AG group commit waiting tuning
- Contained AG support for DAG
- DAG sync improvements
- Backups on secondary replicas
- ZSTD Backup compression
- Backup to Azure immutable storage

What Is Optimized Locking?

Optimized Locking is a new feature in SQL Server 2025 designed to improve concurrency and reduce memory usage by minimizing the number and duration of locks held during transactions. It is especially beneficial for **Data Modification Language (DML)** operations like **INSERT, UPDATE, DELETE**, and **MERGE**.

- **Transaction ID (TID) Locking**: Each row modified by a transaction is tagged with a unique TID. Instead of holding multiple row-level locks, SQL Server holds a single lock on the TID, which significantly reduces memory consumption and lock contention.

- **Lock After Qualification (LAQ)**: SQL Server evaluates query predicates using the latest committed version of a row without acquiring a lock. Only if the predicate is satisfied does it acquire a lock, and that lock is released immediately after the row is modified.

What Is Optimized Locking?

This behavior reduces the likelihood of **lock escalation** and improves **workload concurrency**.

- **Without Optimized Locking:** Updating 1,000 rows might require 1,000 exclusive locks held until the transaction completes.
- **With Optimized Locking:** Each row lock is released immediately after modification, and only one TID lock is held until the transaction ends.

How to Enable Optimized Locking

To enable optimized locking in SQL Server 2025:

- **Accelerated Database Recovery (ADR)** must be enabled.
- **Read Committed Snapshot Isolation (RCSI)** should be enabled to activate LAQ

```
ALTER DATABASE [YourDatabase] SET ACCELERATED_DATABASE_RECOVERY = ON;
ALTER DATABASE [YourDatabase] SET READ_COMMITTED_SNAPSHOT = ON;
ALTER DATABASE [YourDatabase] SET OPTIMIZED_LOCKING = ON;

SELECT DATABASEPROPERTYEX(DB_NAME(), 'IsOptimizedLockingOn') AS
IsOptimizedLockingOn;
```

Optimized Locking Performance Insights

Lock Memory Reduction: Fewer locks are held, and they are released faster.

Improved Concurrency: Less blocking between concurrent transactions.

Reduced Lock Escalation: Helps avoid performance bottlenecks in high-volume environments.

Isolation Level Matters:
Optimized locking is most effective under READ_COMMITTED.

Under SERIALIZABLE, the benefits diminish due to stricter locking requirements.

Not Applicable to Schema Locks: It does not affect schema-level or object-level locks.

Optimized Locking - SQL Server | Microsoft Learn

Demo: Improving Application Concurrency

TempDB Improvements

Resource Governor

- **Purpose**: Prevent runaway queries or workloads from consuming excessive tempdb space, which historically caused server-wide outages.
- **How it works**: You can now set per-workload group limits on tempdb usage. If a query exceeds its group's limit, it is aborted with error 1138.
- **ALTER WORKLOAD GROUP** [default] WITH (GROUP_MAX_TEMPDB_DATA_MB =17408);

Accelerated Database Recovery

- **Impact**: Enables fast rollback of transactions using temp tables.
- Improves log truncation and recovery speed for tempdb-heavy workloads.
- **Use Case**: Ideal for systems with frequent temp table usage or large reporting queries.

Monitoring Improvements

- **New Error Code**: Error 1138 is triggered when tempdb usage exceeds configured limits.
- **Telemetry Improvements**: Enhanced visibility into tempdb usage patterns and workload behavior.
- **Integration with Extended Events**: Time-bound events help prevent runaway logging from tempdb-related operations.

What Are Persisted Statistics for Readable Secondaries?

Persisted statistics for readable secondaries dramatically improves query performance and plan stability for workloads running against secondary replicas in Always On Availability Groups.

- **Prior to SQL Server 2025:** When running read-only queries on secondary replicas, SQL Server could only create temporary statistics in tempdb. These statistics were lost whenever the instance restarted, causing performance regressions and requiring statistics to be rebuilt from scratch.
- **Problem:** Temporary statistics were not shared across replicas and were not persisted, so query plans could degrade after failover or restart.

Persisted Statistics for Readable Secondary Replicas - SQL Server | Microsoft Learn

Persisted statistics for readable secondaries

Technical Details

- **Enabled by Default:** No configuration is required in SQL Server 2025; the feature is on by default.
- **Query Store Integration:** The feature leverages the infrastructure of Query Store for readable secondaries, *which is now enabled by default in SQL Server 2025*. No trace flags are required.
- **Temporary Statistics** Naming: Temporary statistics created on secondaries are named with the suffix _readonly_database_statistic to distinguish them from permanent statistics.
- **Catalog Views:** New columns in sys.stats help DBAs track where statistics were created and their replica role (primary, secondary, geo-replication, etc.)

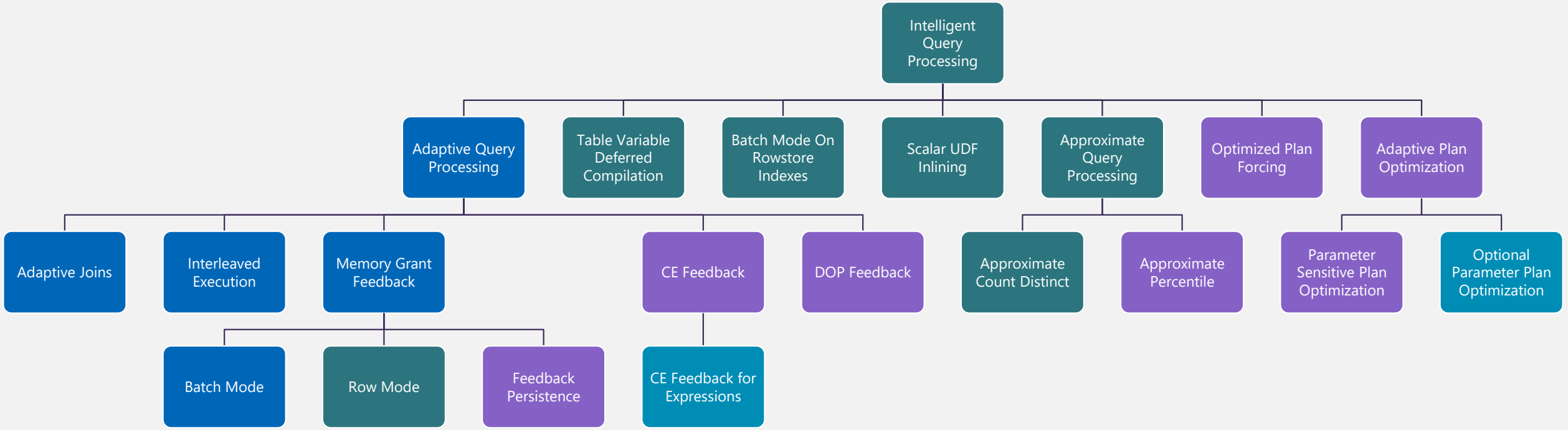
How it Works

- **Automatic Creation and Persistence:** When a secondary replica creates or updates statistics (for example, due to the AUTO_CREATE_STATISTICS option), these statistics are now sent back to the primary replica.
- **Persistence on Primary:** The primary replica persists these statistics in system tables within the user database. This is a logged change, so it is durable and replicated to all secondaries.
- **Availability Across Replicas:** Once persisted, these statistics are available to all readable secondary replicas, ensuring consistent query optimization and plan quality even after restarts or failovers.
- **Monitoring:** Use sys.stats and Query Store views to monitor statistics creation and propagation.

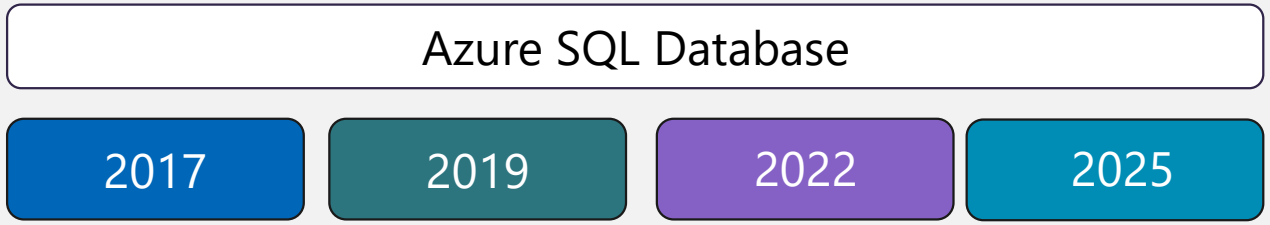
Benefits

- **Performance Stability:** Query plans on readable secondaries remain optimal and stable, even after restarts or failovers.
- **Reduced Overhead:** Eliminates the need to repeatedly recreate statistics, saving time and resources.
- **Consistent Optimization:** All replicas benefit from up-to-date statistics, improving cardinality estimation and query plan quality for offloaded read workloads

Intelligent Query Processing



http://aka.ms/IQP



What is CE Feedback for Expressions?

CE feedback for expressions is a new feature in SQL Server 2025 that enhances the query optimizer's ability to estimate row counts (cardinality) for queries involving complex expressions, such as computed columns, scalar functions, or arithmetic operations in predicates.

Problem: Historically, CE struggled with expressions, especially when statistics were missing or the expressions were complex. This could lead to suboptimal plans and poor performance.

How Does CE Feedback for Expressions Work?

Feedback Loop: When SQL Server executes a query and detects a significant difference between the estimated and actual row counts for expressions, it records this feedback.

Persisted Feedback: The optimizer stores feedback about expressions in system tables, allowing future queries to benefit from improved cardinality estimates.

Automatic Adjustment: On subsequent executions, the optimizer uses this feedback to adjust its estimates, leading to better query plans.

Key Benefits

1

Improved Plan Quality: Queries with complex expressions are more likely to get accurate estimates and efficient plans.

2

Self-Healing: The system automatically learns from previous mistakes and adapts, reducing the need for manual intervention.

3

Reduced Performance Issues: Fewer bad plans due to poor cardinality estimation for expressions.

4

Enabled by Default: CE feedback for expressions is on by default in SQL Server 2025.

What Is Optional Parameter Plans Optimization?

- **Optional parameter plans optimization** is a new feature in SQL Server 2025 that improves how the query optimizer handles queries with optional parameters—parameters that may or may not be provided (i.e., can be NULL or omitted).

- **Parameter Sniffing Problem:** In previous versions, SQL Server would generate a query plan based on the first set of parameter values it encountered. If a query was executed with a parameter as NULL (or not provided), the plan might be suboptimal for subsequent executions with non-NULL values, or vice versa.

How Does the Optimization Work (PSOP v2)

Plan Diversity: SQL Server 2025 can now recognize when a query is executed with optional parameters and generate multiple plans—one for when the parameter is provided, and another for when it is not.

Automatic Plan Selection: The optimizer automatically chooses the most appropriate plan based on whether the parameter is present or NULL.

Reduced Recompilation: This reduces the need for frequent recompilation and avoids performance issues caused by “parameter sniffing.”

Key Benefits

Better Performance: Queries with optional parameters get more optimal plans, improving execution speed and resource usage.

Plan Stability: Reduces the risk of bad plans due to parameter sniffing, especially in procedures or queries with many optional filters.

Less Manual Tuning: DBAs and developers spend less time troubleshooting and forcing recompiles or using hints.

Optional Parameters Optimization Demo

Batch Mode Optimizations

Improved Parallelism: SQL Server 2025 can better distribute work across multiple CPU cores, making batch mode operations faster and more scalable.

Reduced Memory Consumption: Batch mode uses memory more efficiently, allowing larger queries to run without exhausting resources.

Enhanced Execution Plans: The query optimizer generates smarter plans that leverage batch processing, taking full advantage of modern hardware.

Lower CPU Usage: By minimizing the overhead of row-by-row processing, batch mode reduces CPU cycles needed for large analytical queries.

What Is the Halloween Problem?

- The Halloween Problem occurs during update operations when rows that have already been updated are re-evaluated and updated again within the same operation. This can lead to infinite loops or unintended repeated updates, especially when the update criteria depend on indexed columns.

- **Trivia:** This was discovered in Palo Alto in the 1960's on a Halloween night, hence the name.

How Has Halloween Protection Been Optimized?

The query processor is now better at detecting scenarios where the Halloween Problem could occur, especially in complex update statements or bulk operations.

SQL Server uses temporary worktables or intermediate result sets more effectively. This isolates rows being updated from those being read, preventing them from being re-processed during the same operation.

Improved locking ensures that once a row is read for update, it cannot be re-read until the update is complete. This prevents repeated updates and maintains data integrity.

Indexes are used more intelligently during update operations. The optimizer may choose scan or seek strategies that avoid revisiting updated rows, further reducing the risk of the Halloween Problem.

Optimized sp_executesql

- Enhanced Plan Caching and Reuse
- Improved Parameter Sniffing Handling
- Batch Mode Integration
- Reduced Memory Footprint
- Better Feedback for Cardinality Estimation

Optimized sp_executesql Demo

ABORT_QUERY_EXECUTION query hint

Abort future query execution

Stop rogue queries from harming other users

Query Store Hint

ABORT_QUERY_EXECUTION hint Demo

High Availability and Disaster Recovery Features

Enhancing the industry proven engine

Security

- Security cache improvements
- OAEP support for encryption
- PBKDF password hashing
- Authentication using system-assigned managed identity
- Backup to URL with managed identity
- Managed identity support for EKM
- Managed identity for AI models
- Entra logins with nonunique display names
- Custom password policy on Linux
- TDS 8.0/TLS 1.3 support for tools

Performance

- Optimized Locking
- Tempdb space resource governance
- ADR in tempdb
- Persisted stats for readable secondaries
- Change tracking cleanup
- Columnstore index maintenance
- CE feedback for expressions
- Optional parameter plans optimization
- DOP feedback on by default
- Optimized Halloween protection
- Query store for readable secondaries
- ABORT_QUERY_EXECUTION query hint
- Optimized sp_executesql
- Batch mode optimizations
- Remove In-Memory OLTP from a database
- tmpfs support for tempdb in Linux

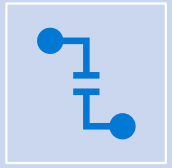
HADR

- Fast failover for persistent AG Health
- Async page request dispatching
- Improved health diagnostics
- Communication control flow tuning
- Switching to resolving state
- Remove listener IP address
- NONE for routing
- AG group commit waiting tuning
- Contained AG support for DAG
- DAG sync improvements
- Backups on secondary replicas
- ZSTD Backup compression
- Backup to Azure immutable storage

What Is Fast Failover for Persistent AG Health?

Fast failover for persistent AG (Availability Group) Health is a new feature in SQL Server 2025 that improves the reliability and responsiveness of Always On Availability Groups by enabling quicker failover when persistent health issues are detected.

Persistent AG Health: SQL Server continuously monitors the health of each replica in an Availability Group. Persistent health means that the system tracks health status over time, not just momentary failures.



Fast Failover: When a replica's health is persistently degraded (not just a transient blip), SQL Server 2025 can trigger a failover much more quickly than in previous versions.

Benefits of Fast Failover

Reduced Downtime: Applications experience less interruption because failover happens faster when a real problem is detected.

Improved Reliability: Persistent health tracking ensures that only genuine, ongoing issues trigger failover, avoiding unnecessary switches due to transient glitches.

Better High Availability: Mission-critical workloads benefit from more robust and responsive failover mechanisms.

How Does Fast Failover Work?

Health Monitoring: SQL Server uses enhanced health checks to monitor the status of each AG replica, including connectivity, synchronization, and data integrity.

Persistent Detection: If a replica is found to be unhealthy for a sustained period (rather than just a brief outage), the system marks it as persistently unhealthy.

Immediate Action: The AG automatically initiates a failover to a healthy secondary replica, minimizing downtime and data loss.

What Is Async Page Request Dispatching?

Async page request dispatching is a new engine-level optimization in SQL Server 2025 that improves how the database engine retrieves data pages from disk or memory. Traditionally, page requests (when SQL Server needs to read a data page) were handled synchronously—each request would wait for the previous one to complete before starting the next

- **Asynchronous Dispatch:** SQL Server 2025 can now issue multiple page requests in parallel, without waiting for each one to finish before starting the next. This is especially beneficial for queries that need to read many pages, such as large scans or analytical workloads.
- **Improved I/O Throughput:** By dispatching page requests asynchronously, SQL Server can better utilize available disk bandwidth and reduce wait times for data retrieval.
- **Reduced Query Latency:** Queries that require many pages can complete faster because the engine doesn't have to wait for each page to be read sequentially.

What Does "Switching to Resolving State" Mean?

In SQL Server Always On Availability Groups, the resolving state is a transitional status that occurs when the cluster detects a problem with the primary replica or the connection between replicas.

The AG resource enters the resolving state while it determines the best course of action—such as failover, recovery, or re-establishing connectivity.

How Is This Improved in SQL Server 2025?

Faster Detection: SQL Server 2025 improves the speed at which AGs detect failures or connectivity issues, allowing the cluster to enter the resolving state more quickly.

Persistent Health Integration: With persistent health monitoring, the AG can more accurately determine whether the issue is transient or persistent, reducing unnecessary transitions.

Optimized Recovery: The engine now uses enhanced diagnostics and telemetry to resolve the state more efficiently, minimizing downtime and improving overall availability.

Clearer Diagnostics: SQL Server 2025 provides better logging and visibility into why the AG entered the resolving state, helping DBAs troubleshoot and respond faster.

What Does "Remove Listener IP Address" Mean?

Removing a listener IP address means deleting or unregistering an IP address that is associated with the AG listener.

This action updates the AG configuration so that the removed IP address is no longer used for client connections.

It is typically done when:

- You are decommissioning a subnet or network.
- You want to restrict access to the AG from certain networks.
- You are reconfiguring your high availability setup.

Removing an IP Listener

The process required manual steps and sometimes service interruptions.

Removing an IP address would update the AG listener configuration, and clients would no longer be able to connect using that IP.

In SQL Server 2025, the process is more integrated and can be automated or scripted more easily.

```
# Remove an IP address from an AG listener
Remove-SqlAvailabilityGroupListenerIPAddress -Path
"SQLSERVER:\SQL\<ServerName>\Default\AvailabilityGroups\<AGName>\Availabilit
yGroupListeners\<ListenerName>\IPAddress\<IPAddress>"
```

What Does "NONE for Routing" Mean?

In SQL Server 2025, NONE for routing indicates that no explicit routing rule or behavior is defined for a particular operation or listener.

- **Default Behavior:** SQL Server will use its default routing logic, rather than applying any custom or advanced routing rules.
- **No Special Routing:** The system will not redirect or route connections in a special way for the specified operation, replica, or listener.
- **Use Case:** This is often used when you want connections to go directly to the primary replica, or when you do not want to configure read-only routing for secondary replicas in an Availability Group.

What Is AG Group Commit Waiting?

What is AG Group Commit Waiting

- In SQL Server Always On Availability Groups (AGs), **group commit** is a mechanism that batches multiple transaction log records together before sending them to secondary replicas. This batching improves throughput and reduces the overhead of synchronizing each transaction individually.
- **Group commit waiting** refers to the time the primary replica waits to accumulate enough transactions to form a batch before sending them to the secondaries. Tuning this waiting period can have a significant impact on both performance and data durability.

New in SQL Server 2025

- **Lower Latency:** Shorter waits mean transactions are sent to secondaries more quickly, reducing commit latency for synchronous AGs.
- **Higher Throughput:** Longer waits allow larger batches, which can improve overall throughput and reduce network overhead.

Key Features of AG Group Commit Waiting

Configurable Wait Time: You can now set the maximum wait time for group commit, tailoring it to your workload's needs.

Dynamic Adjustment: SQL Server can dynamically adjust group commit behavior based on system load and transaction volume.

Monitoring: Enhanced DMVs and telemetry provide visibility into group commit wait times and their impact on AG performance.

Why Is This Important for AG Group Commit Waiting

Performance Tuning: For high-volume OLTP workloads, tuning group commit waiting can reduce commit latency and improve user experience.

Resource Optimization: Efficient batching reduces network and CPU usage, especially in environments with many synchronous replicas.

Consistency: Ensures that data is committed and synchronized according to your business requirements for durability and availability.

Monitor AG performance using DMVs such as sys.dm_hadr_group_commit and related telemetry.

What Is Contained AG Support for DAG?

Contained Availability Groups (AGs): In SQL Server, a contained AG is an Availability Group that includes all database-level objects and users required for operation, making it easier to manage and move AGs between environments without external dependencies.

DAG (Distributed Availability Group): A DAG is a high-availability feature that allows you to link multiple AGs across different clusters or data centers, providing geo-redundancy and disaster recovery.

What's New in SQL Server 2025? (for Availability Groups)

- You can now create DAGs where the member AGs are contained AGs. This means all user objects, logins, jobs, and permissions required for the AG are included within the AG itself, simplifying management and failover across distributed environments.

- **Benefits:**
- **Simplified Management:** No need to manually synchronize logins, jobs, or other server-level objects between clusters.
- **Improved Portability:** Contained AGs can be moved or failed over between clusters with fewer manual steps.
- **Enhanced Security:** All necessary objects are scoped to the AG, reducing risk of missing dependencies during failover or migration.

What Are DAG Sync Improvements in SQL Server 2025

- Faster Synchronization
- Reduced Latency
- Better Error Handling and Recovery
- Improved Monitoring and Telemetry
- Automatic Tuning

Backups on secondary replicas

Expanded Backup Types: You can now perform **differential** backups directly on secondary replicas, in addition to **full and log backups**.

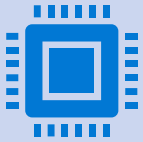
Improved Performance: Backup operations on secondary replicas are further optimized to minimize resource contention and impact on both primary and secondary workloads.

Automated Load Balancing: SQL Server 2025 introduces automatic distribution of backup tasks across multiple secondary replicas, balancing the load and improving overall system efficiency.

Advanced Monitoring & Management: Enhanced tools and dashboards allow for better tracking, alerting, and policy management of backup operations across all replicas.

Seamless Integration: These improvements make it easier to offload backup workloads from primary databases, reducing disruption to mission-critical applications.

What Is ZSTD Backup Compression?



ZSTD (Zstandard) Backup compression is a new backup compression algorithm introduced in SQL Server 2025.

ZSTD is a modern, high-performance compression technology developed by Facebook, designed to provide fast compression and decompression speeds with excellent compression ratios.

Key Features of ZSTD Compression in SQL Server 2025

Higher Compression Ratios: ZSTD typically achieves better compression than legacy algorithms (like LZ or LZ4), resulting in smaller backup files.

Faster Performance: ZSTD is optimized for speed, so backups and restores complete more quickly, especially on modern hardware.

Lower CPU Usage: Efficient compression means less CPU overhead during backup operations, freeing resources for other tasks.

Native Integration: ZSTD is available as a built-in option for backup commands in SQL Server 2025.

How to Use ZSTD Compression

```
BACKUP DATABASE [YourDatabase] TO DISK =
'YourBackupFile.bak'
WITH COMPRESSION_ALGORITHM = 'ZSTD';
```

Why Is This Important?

Storage Savings: Smaller backup files reduce storage costs and make it easier to manage backup retention.

Faster Backups and Restores: Improved speed means less downtime and quicker recovery.

Modern Standard: ZSTD is widely adopted in the industry for its balance of speed and compression efficiency.

What Is Backup to Azure Immutable Storage?

Backup to Azure immutable storage in SQL Server 2025 is a feature that allows you to store your database backups in Azure Blob Storage configured as immutable. This means that once a backup file is written, it cannot be modified or deleted until its retention period expires.

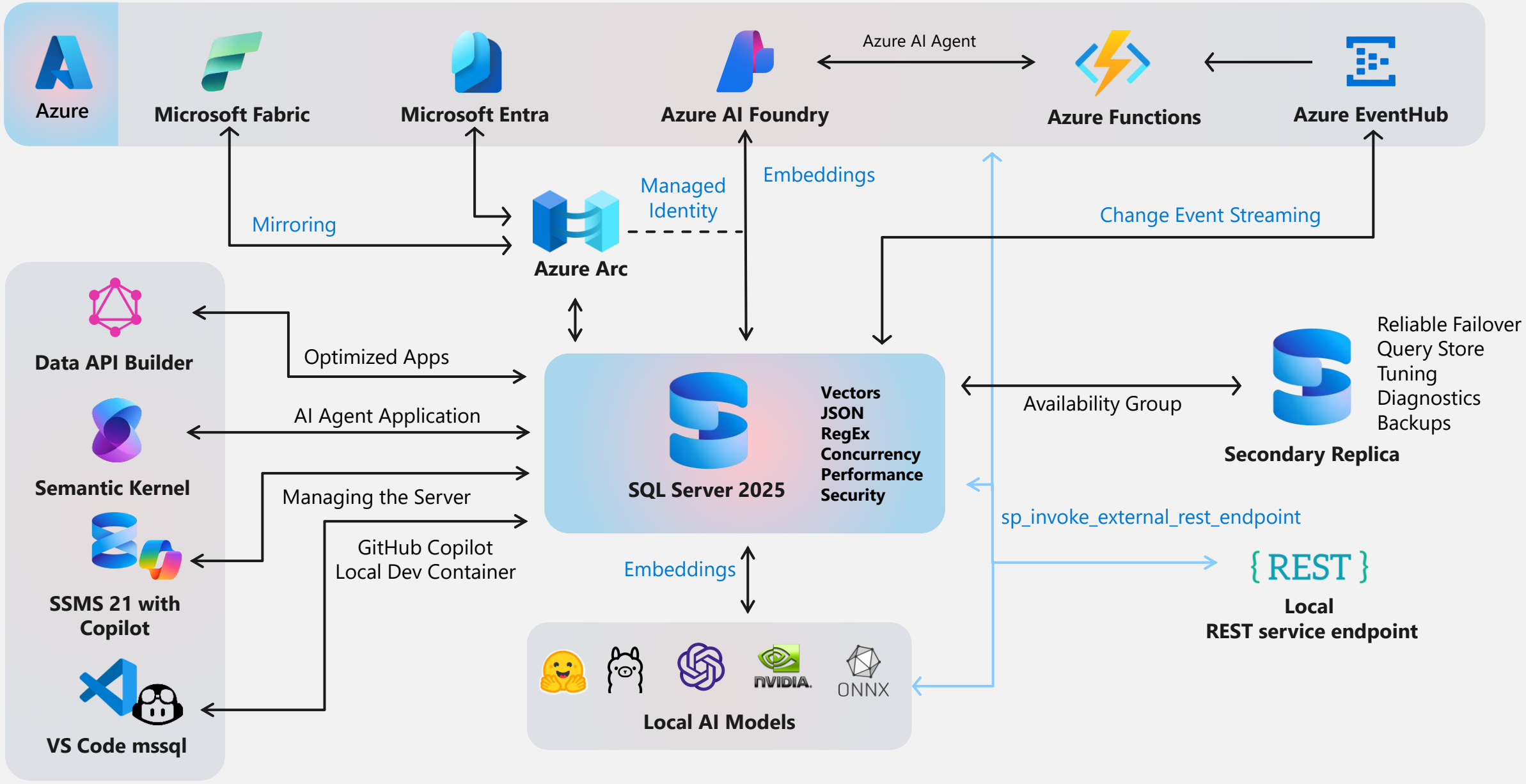
Key Benefits

Ransomware Protection: Immutable backups cannot be altered or deleted, protecting your data from malicious attacks.

Regulatory Compliance: Meets requirements for data retention and integrity, as mandated by many industry regulations (e.g., financial, healthcare).

Data Integrity: Ensures that backup files remain unchanged and available for restore operations throughout their retention period.

The SQL Server 2025 Platform Architecture



Customer experiences

kramer&crew
Der IT-Regisseur.

"With the new semantic search and RAG capabilities in SQL Server 2025, we can empower existing GenAI solutions with data embeddings to create next-generation, more intelligent AI applications."

msc

"Change Event Streaming and Fabric Mirroring for SQL Server 2025 help MSC to build the bridge to bring our operational data into Microsoft Fabric"

SAXO
BE INVESTED

"We are especially looking forward to the performance benefits of optimized locking and the new ZSTD backup compression algorithm; the enhanced reliability of availability groups; and the continued investments in security"


Q
&
A

Thank you!