SQL Server 2025 Neuerungen

für Entwickler

Thorsten Kansy (tkansy@dotnetconsulting.eu)

Meine Person- Thorsten Kansy

Freier Consultant, Software Architekt,
Entwickler, Trainer & Fachautor

Microsoft SQL Server

.NET Core

ASP.NET Core

Azure Cosmos DB

DOTNETCONSULTING

Mein Service- Ihr Benefit

- Individuelle Inhouse Trainings
- (Online on-demand) Projektbegleitung
- Beratung
  - Problemanalyse und Lösungen
  - Technologieentscheidungen

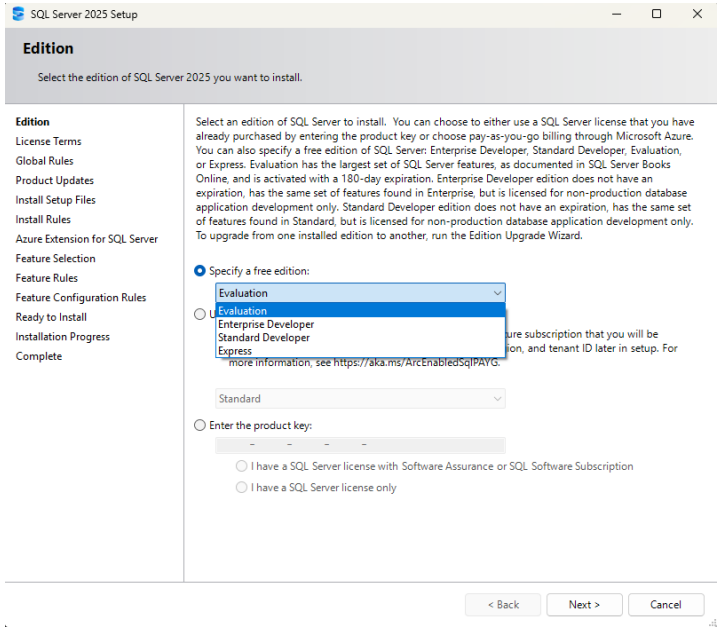
Agenda- Auswahl für Entwickler

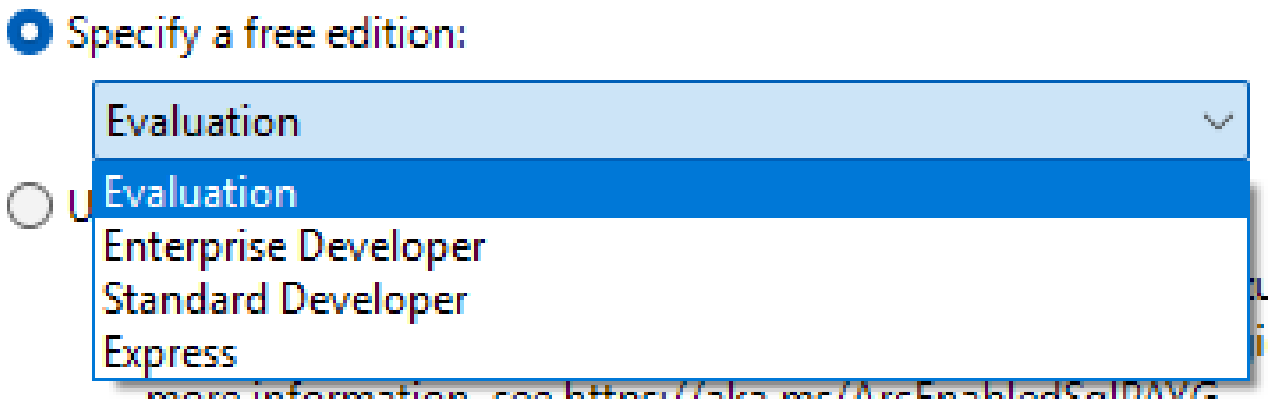
- Developer Editions & Dev Tools
- Reguläre Ausdrücke
- Fuzzy String Matching
- JSON
- REST API-Aufrufe
- Künstliche Intelligenz & Vektoren
- …

Neue Developer Editions

Neue Developer Editions

- Enterprise Developer Edition
  SQL Server 2025 Enterprise Edition Features
- Standard Developer Edition
  SQL Server 2025 Standard Edition Features





Developer Tools

Developer Tools

- SSMS 2021
- GitHub Copilot
- Neuer Python-Treiber (Open-Source, pip-installierbar)

Demo

Reguläre Ausdrücke

Reguläre Ausdrücke

Funktion	Beschreibung
REGEXP_LIKE()	Gibt einen booleschen Wert zurück, der angibt, ob die Texteingabe mit dem Regex-Muster übereinstimmt
REGEXP_REPLACE()	Gibt eine geänderte Quellzeichenkette zurück, die durch eine Ersatzzeichenkette ersetzt wurde, wenn das Regex-Muster gefunden wurde
REGEXP_SUBSTR()	Extrahiert Teile einer Zeichenkette auf der Grundlage eines regulären Ausdrucksmusters. Gibt das N-te Vorkommen einer Teilzeichenkette zurück, die mit dem Regex-Muster übereinstimmt.
REGEXP_INSTR()	Gibt die Anfangs- oder Endposition der übereinstimmenden Teilzeichenkette zurück, abhängig von der angegebenen Option.
REGEXP_COUNT()	Gibt zurück, wie oft das Regex-Muster in einer Zeichenkette vorkommt.
REGEXP_MATCHES()	Gibt eine Tabelle der erfassten Teilzeichenkette(n) zurück, die mit einem Muster eines regulären Ausdrucks für eine Zeichenkette übereinstimmen
REGEXP_SPLIT_INTO_TABLE()	Gibt eine Tabelle mit aufgeteilten Zeichenketten zurück, die durch das Regex-Muster abgegrenzt sind. Wenn es keine Übereinstimmung mit dem Muster gibt, gibt die Funktion die Zeichenkette zurück.

Reguläre Ausdrücke

- Performance (noch) recht ausbaufähig
- Mächtiger als `LIKE` und Fuzzy String Matching
- SQL Server ist keine Suchmaschine
  - Lucene/ Solr/ ElasticSearch

Demo

RegEx.sql

Fuzzy String Matching

Fuzzy String Matching

Funktion	Beschreibung
EDIT_DISTANCE()	Berechnet die Anzahl der Einfügungen, Löschungen, Ersetzungen und Transpositionen, die erforderlich sind, um eine Zeichenkette in eine andere umzuwandeln. Das Verfahren ist auch als Levenshtein-Distanz bekannt.
EDIT_DISTANCE_SIMILARITY()	Berechnet einen Ähnlichkeitswert, der von 0 (keine Übereinstimmung) bis 100 (vollständige Übereinstimmung) reicht.
JARO_WINKLER_DISTANCE()	Berechnet die Editierdistanz zwischen zwei Zeichenfolgen, wobei Zeichenfolgen bevorzugt werden, die von Anfang an für eine bestimmte Präfixlänge übereinstimmen.
JARO_WINKLER_SIMILARITY()	Berechnet einen Ähnlichkeitswert, der von 0 (keine Übereinstimmung) bis 1 (vollständige Übereinstimmung) reicht.

Demo

Fuzzy String Matching.sql

JSON

JSON- Data Type

```sql
DECLARE @jsonNew JSON;
-- UTF-8 Encoding (Latin1_General_100_BIN2_UTF8)

CREATE TABLE dbo.JsonDemo
(
   ID INT IDENTITY PRIMARY KEY,
   Content JSON NOT NULL
);
```

Demo

Json Datatype.sql

JSON-Index

```sql
CREATE JSON INDEX jsonIndex
    ON dbo.JsonDemo(Content);

CREATE JSON INDEX jsonIndex
    ON dbo.JsonDemo(Content) FOR ('$.Customer', '$.Employee', '$.Order')
    WITH (FILLFACTOR = 80);

CREATE JSON INDEX jsonIndex
    ON dbo.JsonDemo(Content) FOR ('$.Customer', '$.Employee')
    /* Fehler in RC1 */
    WHERE JSON_PATH_EXISTS(Info, '$.Order.IsProcessed') = 1
    WITH (FILLFACTOR = 80);
```

Demo

Json Index.sql

REST API-Aufrufe

REST API-Aufrufe

- Daten über eine Azure-Funktion verarbeiten lassen
- Ein Power BI-Dashboard aktualisieren
- Aufrufen eines lokalen, unternehmensinternen REST-Endpunkts
- Sprechen Sie mit Azure OpenAI Services

Aber auch

- Unüberlegtes Aufruf-Chaos
- Undurchsichtiger Infrastruktur-Aufbau

Demo

REST.sql

Vektor-Datentyp

Vektor-Datentyp

```sql
DECLARE @vector1 VECTOR(5) = '[1, 2, 3, 4, 5]';

CREATE TABLE dbo.VectorDemo
(
   ID INT IDENTITY PRIMARY KEY,
   Vector VECTOR(5) NOT NULL
);
```

Demo

Vector Datatype.sql

Künstliche Intelligenz

Künstliche Intelligenz

- Embeddings mittels KI-Modell erzeugen
- Embeddings = Vektoren
- Aufrufe via Rest, Speicherung als `VECTOR`

Demo

AI Generate.sql

Vektor-Funktionen

Vektor-Funktionen

Funktion	Beschreibung
VECTOR_DISTANCE()	Berechnet den Abstand zwischen zwei Vektoren unter Verwendung einer bestimmten Abstandsmetrik.
VECTOR_SEARCH()	Liefert die nächstgelegenen Vektoren zu einem gegebenen Abfragevektor und einer Abstandsmetrik unter Verwendung eines Algorithmus zur approximativen Vektorsuche.
VECTOR_NORM()	Nimmt einen Vektor als Eingabe und gibt die Norm des Vektors (die ein Maß für seine Länge oder Größe ist) in einem bestimmten Normtyp zurück.
VECTOR_NORMALIZE()	Nimmt einen Vektor als Eingabe und gibt den normalisierten Vektor zurück, d.h. einen Vektor, der so skaliert ist, dass er in einem bestimmten Normtyp eine Länge von 1 hat. Passt einen Vektor so an, dass seine Länge nach den Regeln des angegebenen Normtyps normalisiert wird.
VECTORPROPERTY()	Gibt bestimmte Eigenschaften eines gegebenen Vektors zurück.

Demo

Vector Funktions.sql
Vector Search.sql

Vektor-Suche

Vektor Suche

k-Nearest Neighbors (k-NN)

- Berechnung von Distanzen mittels `VECTOR_DISTANCE()`
- Bis ~50.000 Zeilen

Approximate Nearest Neighbors (ANN)

- Verwendet einen Vektor-Index

Demo

Vector Search.sql

Vektor-Index

Vektor-Index

- Keine „gewöhnlichen“ RDBMS-Indizes
- Approximate Nearest Neighbors (ANN)
  - Tauscht Genauigkeit gegen Performance

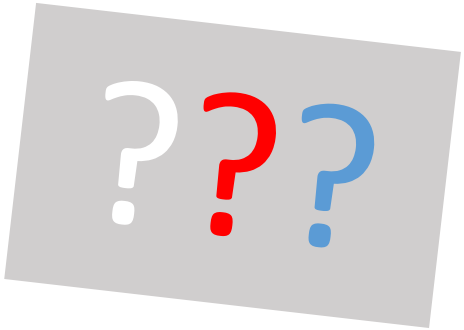
https://www.microsoft.com/en-us/research/project/project-akupara-approximate-nearest-neighbor-search-for-large-scale-semantic-search/

Demo

Vector Index.sql

PRODUCT()

Product()




```
SELECT PRODUCT(/* DISTINCT | ALL */rateOfReturn)
FROM (VALUES (...))
AS _(rateOfReturn, category);

SELECT category,
       PRODUCT(rateOfReturn) OVER (PARTITION BY category)
FROM (VALUES (...))
AS _(rateOfReturn, category);
```

Demo

Product.sql

Und sonst noch?

Und sonst noch?

- Direkter Zugriff auf Formate wie CSV, Parquet und Delta mittels OPENROWSET, ohne PolyBase
- Neue T SQL Funktionen: `DATE_BUCKET()`, `STRING_SPLIT()` mit Positionsparameter, bigint-Support in `DATEADD()`
- Verbesserte Columnstore-Performance
- Query Store auf sekundären Replikaten
- ZSTD-Kompression für Backups
- Tempdb-Optimierung auf Linux durch tmpfs
- Ein intelligenter Security-Cache, der bei vielen gleichzeitigen Logins selektiv aktualisiert wird
- Optimierung bei `sp_executesql`
- Optimiertes Lock-Management, adaptive Memory-Grants, Parallel-Redo und Lock-After-Qualification
- …

Performance Optimierung

Verbesserungen bei älteren Funktionen

- ABS
- ACOS
- ASIN
- ATAN
- ATN2
- CEILING
- COS
- COT
- DEGREES
- EXP
- FLOOR
- LOG
- LOG10
- PI
- POWER
- RADIANS
- RAND
- ROUND
- SIGN
- SIN
- SQRT
- SQUARE
- TAN

syntaxsql Copy

```sql
DATETRUNC ( datepart , date )
```

Fragen? Jetzt oder später!

Kontakt

E-Mail
tkansy@dotnetconsulting.eu

Telefon
+49 (0) 6187 / 2009090

Microsoft Teams
Meet now

LinkedIn
Link me

XING
Xing me

X (Twitter)
@tkansy

DOTNETCONSULTING

IT-Humor mit Kern

IT ist lustig - bis du drinsteckst.

Jetzt lesen und mitlachen

DOTNETCONSULTING

www.dotnetconsulting.eu

SQL Server meets .NET (Core)- professionally!

Ich berate, coache und trainiere im Bereich Entwicklung von .NET (Core) Anwendungen mit Microsoft SQL Server- mit Allem, was dazu gehört- und was man vielleicht weglassen sollte.

DOTNETCONSULTING