



SQL Anywhere® Server

Unterstützung für räumliche Daten

Version 16.0

Februar 2013

Version 16.0
Februar 2013

© 2013 SAP AG oder ein SAP-Konzernunternehmen. Alle Rechte vorbehalten.

Sie können diese Dokumentation (ganz oder teilweise) unter folgenden Bedingungen benutzen, reproduzieren und verteilen: 1) Sie müssen diese und alle anderen Urheberrechtsvermerke auf allen Kopien oder Auszügen der Dokumentation wiedergeben. 2) Sie dürfen die Dokumentation nicht verändern. 3) Sie dürfen nichts tun, aus dem abgeleitet werden könnte, dass Sie oder jemand anderer als SAP Verfasser oder Quelle der Dokumentation ist. Die hier enthaltenen Informationen können jederzeit ohne vorherigen Hinweis geändert werden.

Einige Softwareprodukte, die von der SAP AG oder einem ihrer Vertriebspartner vermarktet werden, enthalten Softwarekomponenten anderer Softwareanbieter. Die nationalen Produktspezifikationen können unterschiedlich sein.

Diese Dokumentationen werden von der SAP AG und ihren Tochtergesellschaften ("SAP Group") lediglich zu Informationszwecken bereitgestellt, ohne dass eine Gewährleistung oder eine Garantie irgendeiner Art gegeben wird. Die SAP Group übernimmt keine Verantwortung im Hinblick auf Fehler oder Auslassungen in den Dokumentationen. Die einzigen Garantien für Produkte und Dienstleistungen der SAP Group sind diejenigen, die in den mit den Produkten und Dienstleistungen eventuell gelieferten ausdrücklichen Garantieerklärungen enthalten sind. Keine der hier enthaltenen Informationen kann als Gewährung einer weitergehenden Garantie betrachtet werden.

SAP und weitere erwähnte SAP-Produkte und -Dienstleistungen sowie die entsprechenden Logos sind Marken oder eingetragene Marken der SAP AG in Deutschland und anderen Ländern. Weitere Hinweise finden Sie unter <http://www.sap.com/corporate-en/legal/copyright/index.epx#trademark>.

Inhalt

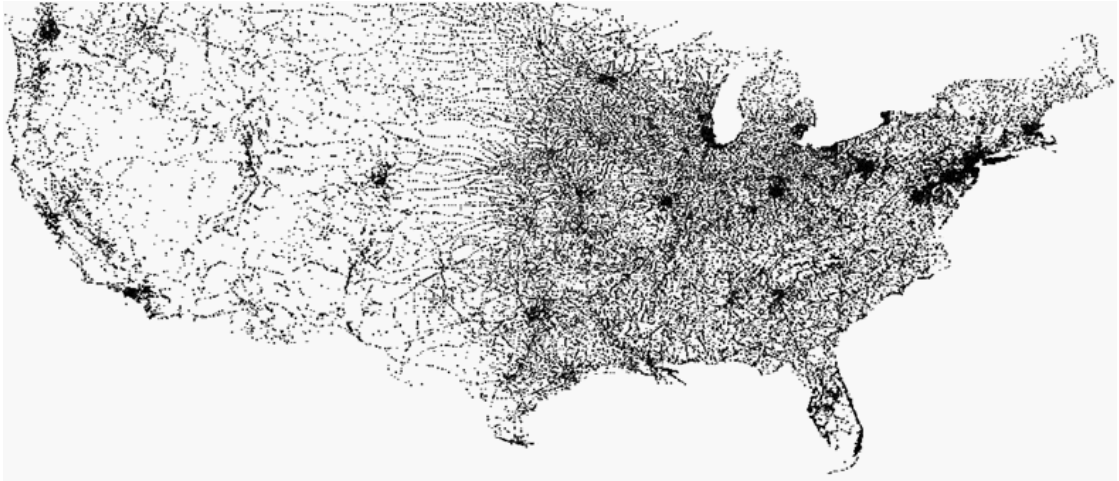
Über diese Dokumentation	v
Räumliche Daten	1
Räumliche Bezugssysteme (SRS) und räumliche Referenz-IDs (SRID)	2
Maßeinheiten	6
SQL Anywhere-Unterstützung für räumliche Daten	7
Empfohlene Lektüre zu räumlichen Themen	16
Erstellen einer räumlichen Spalte (Sybase Central)	17
Räumliche Spalten erstellen (SQL)	20
Indizes für räumliche Spalten	21
Syntax räumlicher Datentypen	22
Geometrien erstellen	25
Anzeigen von räumlichen Daten als Bilder (Interactive SQL)	26
Anzeigen von räumlichen Daten als Bilder (Spatial Viewer)	28
Laden von räumlichen Daten aus einer WKT-Datei (Well Known Text)	31
Maßeinheiten erstellen	37
Räumliche Bezugssysteme erstellen	38
Erweiterte Themen zu räumlichen Daten	42
Praktische Einführung: Mit den räumlichen Funktionen experimentieren ..	57
Zugriff auf räumliche Daten und ihre Verarbeitung	71
ST_CircularString-Datentyp	71
ST_CompoundCurve-Datentyp	79
ST_Curve-Datentyp	85
ST_CurvePolygon-Datentyp	92
ST_GeomCollection-Datentyp	104
ST_Geometry-Datentyp	112
ST_LineString-Datentyp	254
ST_MultiCurve-Datentyp	262
ST_MultiLineString-Datentyp	271
ST_MultiPoint-Datentyp	277

ST_MultiPolygon-Datentyp	283
ST_MultiSurface-Datentyp	291
ST_Point-Datentyp	302
ST_Polygon-Datentyp	319
ST_SpatialRefSys-Datentyp	329
ST_Surface-Datentyp	337
Funktionen der räumlichen Kompatibilität	343
Liste aller unterstützten Methoden	378
Liste aller unterstützten Konstruktoren	390
Liste der statischen Methoden	391
Liste der Aggregatmethoden	395
Liste der Set-Operationsmethoden	396
Liste räumlicher Prädikate	397
 Index	 401

Über diese Dokumentation

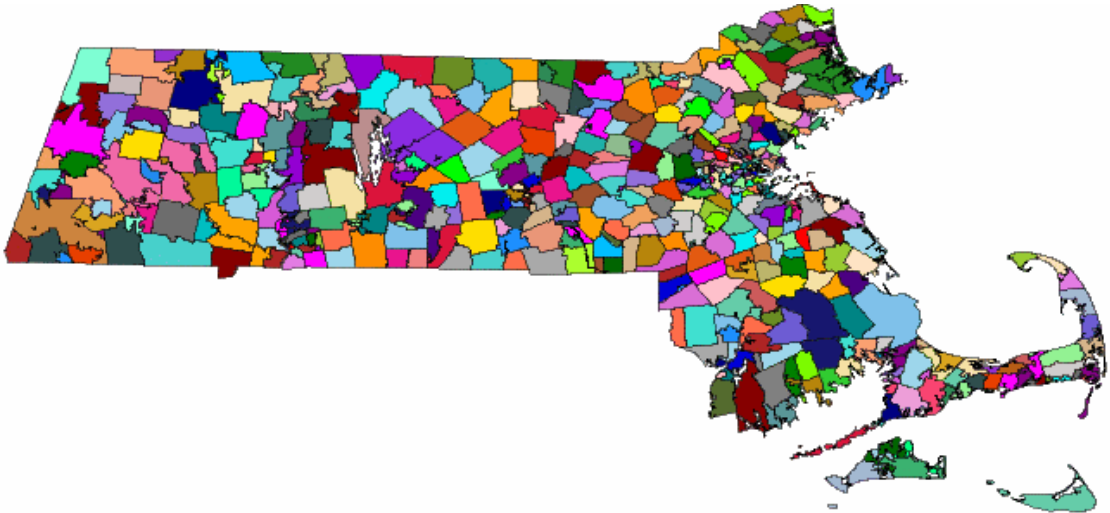
In diesem Handbuch wird beschrieben, wie SQL Anywhere räumliche Daten unterstützt und wie die räumlichen Funktionen verwendet werden können, um räumliche Daten zu erzeugen und zu analysieren.

Das folgende Bild zeigt die Verteilung von großen und kleinen Städten in den USA und ist ein Beispiel für die interessanten Operationen, die Sie mit räumlichen Daten ausführen können.



Räumliche Daten

Räumliche Daten sind Daten, die die Position, Form und Ausrichtung von Objekten in einem definierten Raum beschreiben. Räumliche Daten in SQL Anywhere werden als 2D-Geometrien in Form von Punkten, Kurven (Linienfolgen und Folgen von Kreisbögen) und Polygonen dargestellt. Das folgende Bild zeigt beispielsweise den Staat von Massachusetts, in dem die PLZ-Regionen durch die Verbindung von Polygonen abgebildet werden.



Zwei häufige Vorgänge bei der Bearbeitung räumlicher Daten sind die Berechnung des Abstands zwischen Geometrien und die Bestimmung der Vereinigung oder der Schnittpunkte von mehreren Objekten. Diese Berechnungen werden mithilfe von Prädikaten, wie Überschneidungen, Begrenzungen und Kreuzungspunkten, durchgeführt.

In dieser Dokumentation zu räumlichen Daten wird davon ausgegangen, dass Sie bereits mit räumlichen Bezugssystemen und den räumlichen Daten, die Sie bearbeiten wollen, vertraut sind. Falls dies nicht der Fall ist, finden Sie hier Verknüpfungen zu zusätzlicher Literatur: [„Empfohlene Lektüre zu räumlichen Themen“ auf Seite 16](#).

Hinweis

Die Unterstützung räumlicher Daten für 32-Bit-Windows und 32-Bit Linux erfordert eine CPU, die SSE2-Anweisungen verarbeiten kann. Diese Unterstützung ist bei Prozessoren der Serie Intel Pentium 4 oder höher (ab 2001) und AMD Opteron oder höher (ab 2003) gegeben.

Beispiel der Verwendung von räumlichen Daten

Dank der Unterstützung von räumlichen Daten in SQL Anywhere können Anwendungsentwickler ihren Daten räumliche Informationen zuordnen. Beispiel: Eine Tabelle mit Unternehmen könnte die Standorte der Unternehmen in Form von Punkten oder den Auslieferungsbereich der Unternehmen als Polygone speichern. Dies könnte in SQL wie folgt dargestellt werden:

```
CREATE TABLE Locations(  
    ID INT,  
    ManagerName CHAR(16),  
    StoreName CHAR(16),  
    Address ST_Point,  
    DeliveryArea ST_Polygon )
```

Der räumliche Datentyp `ST_Point` in dem Beispiel stellt einen einzelnen Punkt und `ST_Polygon` ein beliebiges Polygon dar. Mit diesem Schema könnte die Anwendung mit der folgenden Abfrage alle Unternehmenstandorte auf einer Karte anzeigen oder auch ermitteln, ob ein Unternehmen an eine bestimmte Adresse liefert:

```
CREATE VARIABLE @pt ST_Point;  
SET @pt = ST_Geometry::ST_GeomFromText( 'POINT(1 1)' );  
  
SELECT * FROM Locations  
WHERE DeliveryArea.ST_Contains( @pt ) = 1
```

SQL Anywhere stellt Speicher- und Datenverwaltungsfunktionen für räumliche Daten bereit, mit denen Sie Informationen wie geografische Standorte, Routinginformationen und Formdaten speichern können.

Diese Informationen werden als Punkte und verschiedene Formen von Polygonen und Linien in Spalten gespeichert und mit einem entsprechenden **räumlichen Datentyp** definiert (z. B. `ST_Point` und `ST_Polygon`). Hierbei werden Methoden und Konstruktoren verwendet, um auf die räumlichen Daten zuzugreifen und sie zu bearbeiten. SQL Anywhere stellt auch eine Reihe von räumlichen SQL-Funktionen zur Kompatibilität mit anderen Produkten bereit.

Siehe auch

- [„Unterstützte räumliche Datentypen und ihre Hierarchie“ auf Seite 7](#)
- [„Funktionen der räumlichen Kompatibilität“ auf Seite 343](#)
- [„Praktische Einführung: Mit den räumlichen Funktionen experimentieren“ auf Seite 57](#)

Räumliche Bezugssysteme (SRS) und räumliche Referenz-IDs (SRID)

Im Kontext räumlicher Datenbanken wird der definierte Bereich, in dem Geometrien beschrieben werden, als **räumliches Bezugssystem (Spatial Reference System, kurz SRS)** bezeichnet. Ein räumliches Bezugssystem definiert mindestens Folgendes:

- Maßeinheiten des zugrunde liegenden Koordinatensystems (Grad, Meter usw.)
- Maximale und minimale Koordinaten (auch als Grenzwerte bezeichnet)
- Lineare Standardmaßeinheit
- Ob die Daten planar oder sphäroid sind
- Projektionsinformationen für die Transformation der Daten in andere räumliche Bezugssysteme

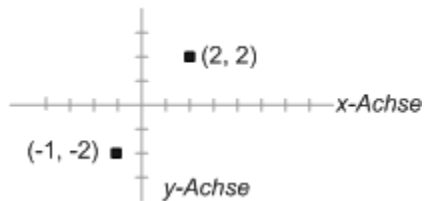
Jedes räumliche Bezugssystem (Spatial Reference System, kurz SRS) hat einen Bezeichner, der **räumliche Referenz-ID (Spatial Reference Identifier, kurz SRID)** genannt wird. Wenn SQL

Anywhere Vorgänge ausführt, z. B. wenn ermittelt werden muss, ob eine Geometrie eine andere Geometrie berührt, wird die SRID dazu verwendet, die Definition des räumlichen Bezugssystems zu suchen, um die Berechnungen für dieses räumliche Bezugssystem richtig ausführen zu können. In einer SQL Anywhere-Datenbank muss jede SRID eindeutig sein.

Standardmäßig fügt SQL Anywhere einer neuen Datenbank folgende räumliche Bezugssysteme hinzu:

- **Standard - SRID 0** Dies ist das standardmäßige räumliche Bezugssystem, das verwendet wird, wenn eine Geometrie erstellt wird und die SRID im SQL-Code nicht angegeben und im eingelesenen Wert nicht vorhanden ist.

Standardmäßig ist dies ein kartesisches räumliches Bezugssystem, das mit Daten auf einer planen, zweidimensionalen Ebene arbeitet. Jeder Punkt auf der Ebene kann mithilfe eines einzelnen Paares einer X- und Y-Koordinate definiert werden, wobei X und Y die Grenzwerte -1.000.000 und 1.000.000 haben. Abstände werden mithilfe der rechtwinkligen Koordinatenachse gemessen. Diesem räumlichen Bezugssystem ist die SRID **0** zugewiesen.



Das kartesische System ist ein planares räumliches Bezugssystem.

- **WGS 84 - SRID 4326** Der WGS 84-Standard stellt eine sphäroide Referenzoberfläche für die Erde bereit. Dies ist das räumliche Bezugssystem, das vom GPS-System (Globales Positionsbestimmungssystem) verwendet wird. Der Koordinatenursprung von WGS 84 ist der Mittelpunkt der Erde und gilt als bis auf ± 1 Meter genau. WGS steht für World Geodetic System (geodätisches Weltsystem).

WGS 84-Koordinaten werden in Grad angegeben, wobei die erste Koordinate der Längengrad mit den Grenzwerten -180 und 180 Grad und die zweite Koordinate der Breitengrad mit den Grenzwerten von -90 und 90 Grad ist.

Die Standardmaßeinheit für WGS 84 ist METER und es ist ein räumliches Bezugssystem vom Typ "gewölbte Erde".

- **WGS 84 (planar) - SRID 1000004326** WGS 84 (planar) unterscheidet sich nur dadurch von WGS 84, dass es die sphärische Projektion verwendet, die Länge, Bereich und andere Berechnungen verzerrt. Beispiel: Am Äquator beträgt 1 Längengrad in den räumlichen Bezugssystemen mit SRID 4326 und 1000004326 etwa 111 km. Bei 80 Grad Nord beträgt 1 Längengrad in der SRID 4326 etwa 19 km, aber die SRID 1000004326 behandelt 1 Längengrad an *allen* Breitengraden als etwa 111 km. Der Umfang der Verzerrung von Längen in der SRID 1000004326 ist beträchtlich (Faktor 10 oder mehr) und der Verzerrungsfaktor variiert außerdem, abhängig von der Position der Geometrien relativ zum Äquator. Aus diesem Grund sollte die SRID 1000004326 nicht für Abstands- und Bereichsberechnungen, sondern ausschließlich für Beziehungsprädikate wie ST_Contains, ST_Touches, ST_Covers usw. verwendet werden.

Die Standardmaßeinheit für WGS 84 (planar) ist GRAD und es ist ein räumliches Bezugssystem vom Typ "plane Erde".

- **sa_planar_unbounded - SRID 2.147.483.646** Wird nur intern verwendet.
- **sa_octahedral_gnomonic - SRID 2.147.483.647** Wird nur intern verwendet.

Da Sie ein räumliches Bezugssystem festlegen und ihm eine beliebige SRID zuordnen können, muss die Definition des räumlichen Bezugssystems (Projektion, Koordinatensystem usw.) die Daten begleiten, wenn sie zwischen Datenbanken verschoben oder in andere räumliche Bezugssysteme konvertiert werden. Beispiel: Wenn Sie räumliche Daten in WKT entladen, ist die Definition für das räumliche Bezugssystem am Anfang der Datei enthalten.

Installieren zusätzlicher räumlicher Bezugssysteme mit der sa_install_feature-Systemprozedur

SQL Anywhere stellt auch Tausende vordefinierter räumlicher Bezugssysteme bereit. Diese räumlichen Bezugssysteme werden nicht standardmäßig in der Datenbank installiert, wenn Sie eine neue Datenbank erstellen. Sie können sie mit der sa_install_feature-Systemprozedur hinzufügen.

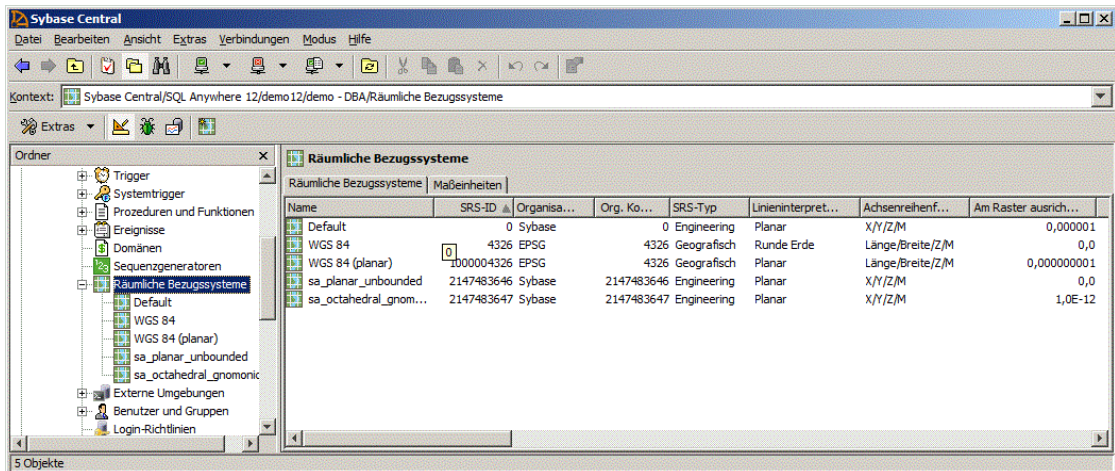
Beschreibungen dieser zusätzlichen räumlichen Bezugssysteme finden Sie unter spatialreference.org und www.epsg-registry.org/.

Liste räumlicher Bezugssysteme festlegen, die sich aktuell in der Datenbank befinden

Die Informationen des räumlichen Bezugssystems sind in der Systemtabelle ISYSPATIALREFERENCESYSTEM gespeichert. Die SRIDs für die räumlichen Bezugssysteme werden in dieser Tabelle als Primärschlüsselwerte verwendet. Der Datenbankserver verwendet SRID-Werte, um die Konfigurationsinformationen für ein räumliches Bezugssystem zu suchen, damit es die ansonsten abstrakten räumlichen Koordinaten als reale Positionen auf der Erde interpretieren kann.

Die Liste räumlicher Bezugssysteme erhalten Sie, indem Sie die konsolidierte Systemansicht ST_SPATIAL_REFERENCE_SYSTEMS abfragen. Jede Zeile in dieser Ansicht definiert ein räumliches Bezugssystem:

Sie können auch im Ordner **Räumliche Bezugssysteme** in Sybase Central die Liste der in der Datenbank installierten räumlichen Bezugssysteme anzeigen.



Kompatibilität mit verbreiteten Zuordnungsanwendungen

Einige verbreitete Kartographie- und Visualisierungsanwendungen im Web wie etwa Google Earth, Bing Maps und ArcGIS Online verwenden ein räumliches Bezugssystem, das auf einem Sphärenmodell der Erddarstellung basiert. Dieses Sphärenmodell ignoriert die Abflachung an den Polen und kann zu Fehlern von bis zu 800 m bei der Position und bis zu 0,7 Prozent beim Maßstab führen. Es ermöglicht aber Anwendungen, Projektionen effizienter auszuführen.

In der Vergangenheit haben kommerzielle Anwendungen diesem räumlichen Bezugssystem die SRID 900913 zugewiesen. Allerdings hat EPSG inzwischen diese Projektion als SRID 3857 freigegeben. Aus Gründen der Kompatibilität mit Anwendungen, die 900913 erfordern, können Sie Folgendes tun:

1. Verwenden Sie die `sa_install_feature`-Systemprozedur, um alle von SQL Anywhere bereitgestellten räumlichen Bezugssysteme (einschließlich SRID 3857) zu installieren.
2. Führen Sie `dbunload -n` aus, um die Definition von SRID 3857 abzurufen.
3. Verwenden Sie Sybase Central, um unter Verwendung der Daten aus der entladenen SRID-Definition ein räumliches Bezugssystem mit SRID 900913 zu erstellen.

Siehe auch

- [Einschränkungen von räumlichen Bezugssystemen mit planer Erddarstellung auf Seite 43](#)
- [„Unterstützte räumliche Prädikate“ auf Seite 10](#)
- [„sa_install_feature-Systemprozedur“ \[*SQL Anywhere Server - SQL-Referenzhandbuch*\]](#)
- [„Räumliche Bezugssysteme erstellen“ auf Seite 38](#)
- [„sa_install_feature-Systemprozedur“ \[*SQL Anywhere Server - SQL-Referenzhandbuch*\]](#)
- [„Konsolidierte Ansicht ST_SPATIAL_REFERENCE_SYSTEMS“ \[*SQL Anywhere Server - SQL-Referenzhandbuch*\]](#)
- [„CREATE SPATIAL REFERENCE SYSTEM-Anweisung“ \[*SQL Anywhere Server - SQL-Referenzhandbuch*\]](#)
- [„Funktionsweise der planaren und gewölbten Erddarstellung“ auf Seite 42](#)

Maßeinheiten

Geografische Objekte können in Breitengrad, Bogenmaß oder anderen Winkelmaßeinheiten gemessen werden. Jedes räumliche Bezugssystem muss den Namen der Einheit, in der geografische Koordinaten gemessen werden, explizit angeben und die Konvertierung von der angegebenen Einheit in ein Bogenmaß umfassen.

Wenn Sie ein projiziertes Koordinatensystem verwenden, stellen die einzelnen Koordinatenwerte einen linearen Abstand entlang der Erde bis zu einem Punkt dar. Koordinatenwerte können in Metern, Fuß, Meilen oder Yard gemessen werden. Das projizierte Koordinatensystem muss die lineare Maßeinheit explizit angeben, in der die Koordinatenwerte ausgedrückt werden.

Die folgenden Maßeinheiten werden automatisch in jeder neuen SQL Anywhere-Datenbank installiert:

- **meter (Meter)** Eine lineare Maßeinheit. Auch bekannt als Urmeter. SI-Standardeinheit. Von ISO 1000 definiert.
- **metre** Eine lineare Maßeinheit. Ein Alias für Meter. SI-Standardeinheit. Von ISO 1000 definiert.
- **radian (Bogenmaß)** Eine Winkelmaßeinheit. SI-Standardeinheit. Von ISO 1000:1992 definiert.
- **degree (Grad)** Eine Winkelmaßeinheit ($\pi()/180,0$ Bogenmaß).
- **planar degree (Planares Grad)** Eine lineare Maßeinheit. Definiert als 60 Seemeilen. Eine lineare Maßeinheit, die für geografische räumliche Bezugssysteme mit PLANARER Linieninterpretation verwendet wird.

Siehe auch

- „Installieren zusätzlicher vordefinierter Maßeinheiten“ auf Seite 6
- „sa_install_feature-Systemprozedur“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SPATIAL REFERENCE SYSTEM-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- LINEAR UNIT OF MEASURE-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- ANGULAR UNIT OF MEASURE-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Konsolidierte Ansicht ST_UNITS_OF_MEASURE“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Installieren zusätzlicher vordefinierter Maßeinheiten

Die sa_install_feature-Systemprozedur fügt zusätzliche vordefinierte Maßeinheiten hinzu, die nicht standardmäßig in einer neuen Datenbank installiert werden.

Voraussetzungen

Keine.

Installieren von zusätzlichen vordefinierten Maßeinheiten

- Führen Sie die folgende Anweisung aus, um alle vordefinierten Maßeinheiten zu installieren:

```
CALL sa_install_feature('st_geometry_predefined_uom');
```

Ergebnisse

Alle vordefinierten Maßeinheiten werden installiert.

Nächste Schritte

Sie können ein räumliches Bezugssystem erstellen, das die Maßeinheit verwendet.

Beschreibungen dieser zusätzlichen Maßeinheiten finden Sie unter www.epsg-registry.org. Geben Sie auf der Webseite den Namen der Maßeinheit im Feld **Name** ein, wählen Sie die Option **Unit of Measure (UOM)** im Feld **Type** und klicken Sie dann auf **Search**.

Siehe auch

- „Maßeinheiten“ auf Seite 6
- „sa_install_feature-Systemprozedur“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SPATIAL REFERENCE SYSTEM-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- LINEAR UNIT OF MEASURE-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- ANGULAR UNIT OF MEASURE-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Konsolidierte Ansicht ST_UNITS_OF_MEASURE“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

SQL Anywhere-Unterstützung für räumliche Daten

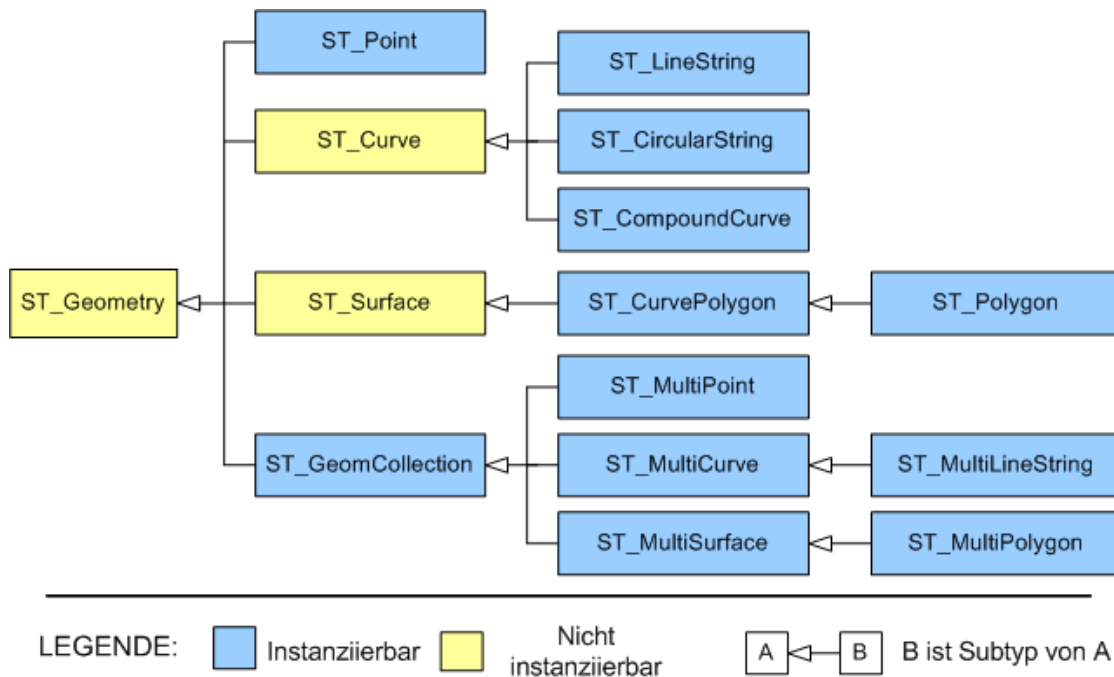
Die folgenden Abschnitte beschreiben die SQL Anywhere-Unterstützung für räumliche Daten.

Unterstützte räumliche Datentypen und ihre Hierarchie

SQL Anywhere folgt dem SQL Multimedia-Standard (SQL/MM) für das Speichern und den Zugriff auf Geodaten. Eine Schlüsselkomponente dieses Standards ist die Verwendung der Hierarchie vom Typ ST_Geometry, mit der festgelegt wird, wie Geodaten erstellt werden. Innerhalb dieser Hierarchie wird das Präfix ST für alle Datentypen verwendet (auch als Klassen oder Typen bezeichnet).

Wenn eine Spalte als spezifischer Typ gekennzeichnet ist, können die Werte des Typs und seiner Unterklassen in der Spalte gespeichert werden. Beispiel: Eine als ST_GeomCollection gekennzeichnete Spalte kann auch ST_MultiPoint-, ST_MultiSurface-, ST_MultiCurve-, ST_MultiPolygon- und ST_MultiLineString-Werte speichern.

Folgendes Diagramm veranschaulicht die Hierarchie der ST_Geometry-Datentypen und ihrer Subtypen:



Die Typen auf der linken Seite sind Obertypen (oder Basistypen) für die Subtypen (oder abgeleitete Typen) auf der rechten Seite.

Beschreibung unterstützter räumlicher Datentypen

SQL Anywhere unterstützt folgende räumliche Datentypen:

- **Punkte** Ein Punkt kennzeichnet eine einzelne Position in einem Raum. Eine Punktgeometrie hat keine Länge und keinen Bereich. Ein Punkt hat immer eine X- und Y-Koordinate.

ST_Dimension gibt für nicht leere Punkte 0 zurück.

In GIS-Daten werden Punkte gewöhnlich verwendet, um Standorte wie Adressen oder geografische Merkmale wie einen Berg darzustellen.

- **Linienfolgen** Eine Linienfolge ist eine Geometrie mit einer bestimmten Länge, aber ohne Bereich. ST_Dimension gibt für nicht leere Linienfolgen 1 zurück. Linienfolgen können dadurch gekennzeichnet werden, dass sie einfach oder nicht einfach, geschlossen oder nicht geschlossen sind. **Einfach** bedeutet, dass eine Linienfolge sich selbst nicht schneidet. **Geschlossen bedeutet, dass eine Linienfolge am gleichen Punkt beginnt und endet.** Ein Beispiel einer einfachen geschlossenen Linienfolge ist ein Kreis.

In GIS-Daten werden Linienfolgen gewöhnlich verwendet, um Flüsse, Straßen oder Zustellungsrouten darzustellen.

- **Polygone** Ein Polygon definiert den Bereich einer Fläche. Ein Polygon besteht aus einem äußeren begrenzenden Ring, der die Außenseite des Bereiches definiert, und null oder mehr inneren Ringen, die Löcher in dem Bereich definieren. Ein Polygon hat einen zugeordneten Bereich, aber keine Länge.

ST_Dimension gibt 2 für nicht-leere Polygone zurück.

In GIS-Daten werden Punkte gewöhnlich verwendet, um Gebiete (Landkreise, Städte, Länder usw.), Seen und große geografische Elemente, wie Parks, darzustellen.

- **Kreisbogenfolgen** Eine Kreisbogenfolge ist eine verbundene Sequenz von Kreisbogensegmenten. Ähnlich wie eine Linienfolge mit Kreisbögen zwischen Punkten.
- **Verbundkurven** Eine Verbundkurve ist eine verbundene Sequenz von Kreisbogenfolgen oder Linienfolgen.
- **Kurvenpolygone** Ein Kurvenpolygon ist ein allgemeineres Polygon, das Kreisbogen-Begrenzungssegmente enthalten kann.
- **Geometrien** Der Begriff Geometrie bezeichnet den übergreifenden Typ von Objekten, wie etwa Punkte, Linienfolgen und Polygone. Der Geometrietyp ist der Obertyp für alle unterstützten räumlichen Datentypen.
- **Geometriegruppen** Eine Geometriegruppe ist eine Gruppe mit einer oder mehreren Geometrien (z.B. Punkte, Linien, Polygone usw.).
- **Mehrpunktangabe** Eine Mehrpunktangabe ist eine Gruppe von Einzelpunkten.

In GIS-Daten werden Mehrpunktangaben gewöhnlich verwendet, um eine Gruppe von Standorten darzustellen.

- **Multipolygone** Ein Multipolygon ist eine Sammlung von null oder mehr Polygonen.

In GIS-Daten werden Multipolygone häufig verwendet, um Gebiete, die aus mehreren Regionen (z.B. ein Staat mit Inseln) bestehen, oder geografische Merkmale (z.B. ein Seensystem) darzustellen.

- **Mehrlinienfolge** Eine Mehrlinienfolge ist eine Sammlung von Linienfolgen.

In GIS-Daten werden Mehrlinienfolgen häufig verwendet, um geografische Merkmale wie Flüsse oder ein Autobahnnetz darzustellen.

- **Mehrfachoberflächen** Eine Mehrfachoberfläche ist eine Sammlung von Kurvenpolygonen.

Objektorientierte Eigenschaften von räumlichen Datentypen

- Ein Subtyp (oder abgeleiteter Typ) ist präziser als der übergeordnete Typ (oder Basistyp). Beispiel: ST_LineString ist ein präziserer Typ von ST_Curve.
- Ein Subtyp erbt alle Methoden von allen Obertypen. Beispiel: ST_Polygon-Werte können Methoden der Obertypen ST_Geometry, ST_Surface und ST_CurvePolygon abrufen.
- Der Wert eines Subtyps kann automatisch in einen seiner Obertypen konvertiert werden. Beispiel: Ein ST_Point-Wert kann verwendet werden, wenn ein ST_Geometry-Parameter erforderlich ist, wie in `point1.ST_Distance(point2)`.
- Eine Spalte oder Variable kann Werte eines beliebigen Subtyps speichern. Beispiel: Eine Spalte vom Typ ST_Geometry(SRID=4326) kann räumliche Werte eines beliebigen Typs speichern.

- Eine Spalte, eine Variable oder ein Ausdruck mit einem deklarierten Typ kann als Subtyp behandelt oder in einen Subtyp umgewandelt werden. Beispiel: Sie können den TREAT-Ausdruck verwenden, um einen ST_Polygon-Wert in einer ST_Geometry-Spalte namens geom als Typ ST_Surface zu deklarieren, sodass Sie die ST_Area-Methode dafür mit `TREAT(geom AS ST_Surface).ST_Area()` aufrufen können.

Siehe auch

- „ST_Point-Datentyp“ auf Seite 302
- „ST_LineString-Datentyp“ auf Seite 254
- „ST_Polygon-Datentyp“ auf Seite 319
- „Funktionsweise der Ausrichtung von Polygonringen“ auf Seite 50
- „ST_CircularString-Datentyp“ auf Seite 71
- „ST_CompoundCurve-Datentyp“ auf Seite 79
- „ST_CurvePolygon-Datentyp“ auf Seite 92
- „ST_Geometry-Datentyp“ auf Seite 112
- „ST_GeomCollection-Datentyp“ auf Seite 104
- „ST_MultiPoint-Datentyp“ auf Seite 277
- „ST_MultiPolygon-Datentyp“ auf Seite 283
- „ST_MultiLineString-Datentyp“ auf Seite 271
- „ST_MultiSurface-Datentyp“ auf Seite 291
- „TREAT-Funktion [Datentypkonvertierung]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CAST-Funktion [Datentypkonvertierung]“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Unterstützte räumliche Prädikate

Ein Prädikat ist ein bedingter Ausdruck der, kombiniert mit den logischen Operatoren AND und OR, die Gruppe der Bedingungen in einer WHERE-, HAVING- oder ON-Klausel oder in einem IF- oder CASE-Ausdruck oder einer CHECK-Integritätsregel bildet. In SQL wird ein Prädikat als TRUE oder FALSE ausgewertet. In vielen Kontexten wird ein Prädikat, das als UNKNOWN aufgelöst wird, als FALSE interpretiert.

Räumliche Prädikate werden als Member-Funktionen implementiert, die 0 oder 1 zurückgeben. Wenn Sie ein räumliches Prädikat testen möchten, sollte Ihre Abfrage das Ergebnis der Funktion mithilfe des Operators = oder <> mit 1 oder 0 vergleichen. Beispiel:

```
SELECT * FROM SpatialShapes WHERE geometry.ST_IsEmpty() = 0;
```

Sie verwenden Prädikate bei der Abfrage von räumlichen Daten, um bestimmte Fragen zu beantworten: Wie nahe zusammen liegen zwei oder mehrere Geometrien? Schneiden oder überlappen sie sich? Befindet sich eine der Geometrien innerhalb einer anderen? Eine Zustellungsfirma könnte z. B. Prädikate verwenden, um festzustellen, ob ein Kunde sich innerhalb eines bestimmten Zustellungsbereichs befindet.

SQL Anywhere unterstützt die räumlichen Prädikate zur Beantwortung von Fragen über die räumlichen Beziehungen zwischen Geometrien.

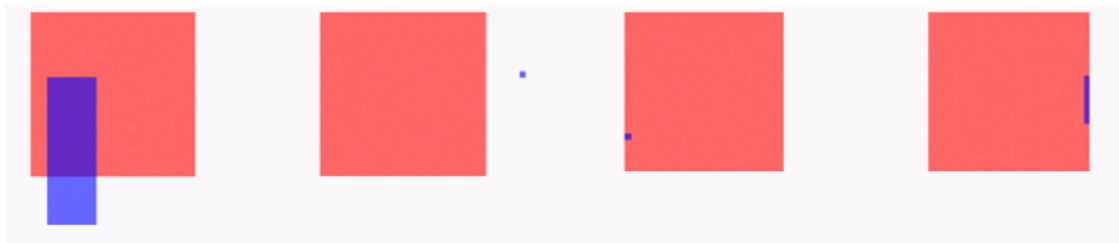
Siehe auch

- [ST_Contains-Methode für den ST_Geometry-Datentyp](#)
- [ST_Covers-Methode für den ST_Geometry-Datentyp](#)
- [ST_CoveredBy-Methode für den ST_Geometry-Datentyp](#)
- [ST_Crosses-Methode für den ST_Geometry-Datentyp](#)
- [ST_Disjoint-Methode für den ST_Geometry-Datentyp](#)
- [ST_IsEmpty-Methode für den ST_Geometry-Datentyp](#)
- [ST_Equals-Methode für den ST_Geometry-Datentyp](#)
- [ST_Intersects-Methode für den ST_Geometry-Datentyp](#)
- [ST_OrderingEquals-Methode für den ST_Geometry-Datentyp](#)
- [ST_Overlaps-Methode für den ST_Geometry-Datentyp](#)
- [ST_Relate-Methode für den ST_Geometry-Datentyp](#)
- [ST_Touches-Methode für den ST_Geometry-Datentyp](#)
- [ST_IsValid-Methode für den ST_Geometry-Datentyp](#)
- [ST_Within-Methode für den ST_Geometry-Datentyp](#)

Intuitivität räumlicher Prädikate

Das Ergebnis eines Prädikats ist manchmal nicht intuitiv. Aus diesem Grund sollten spezielle Fälle getestet werden, um sicherzustellen, dass die gewünschten Ergebnisse geliefert werden. Beispiel: Damit eine Geometrie eine andere Geometrie enthält ($a.ST_Contains(b)=1$) oder eine Geometrie sich innerhalb einer anderen Geometrie befindet ($b.ST_Within(a)=1$), müssen sich die Innenbereiche von a und b schneiden und kein Teil von b darf den Außenbereich von a schneiden. Es gibt jedoch bestimmte Fälle, in denen eine Geometrie als innerhalb einer anderen Geometrie betrachtet werden könnte, dies aber nicht wird.

Folgende Anweisung gibt z. B. für $a.ST_Contains(b)$ und $b.ST_Within(a)$ 0 (a ist rot) zurück:



Fall eins und zwei sind offensichtlich: Die lila Geometrien liegen nicht vollständig innerhalb der roten Quadrate. Fall drei und vier sind allerdings nicht so offensichtlich. In beiden Fällen liegen die lila Geometrien nur auf der Grenze der roten Quadrate. `ST_Contains` betrachtet die lila Geometrien als nicht innerhalb der roten Quadrate liegend, auch wenn sie es anscheinend tun.

`ST_Covers` und `ST_CoveredBy` sind Prädikate ähnlich `ST_Contains` und `ST_Within`. Der Unterschied liegt darin, dass `ST_Covers` und `ST_CoveredBy` nicht erfordern, dass das Innere der beiden Geometrien eine Schnittmenge aufweist. Außerdem haben `ST_Covers` und `ST_CoveredBy` häufig einfacher zu verstehende Ergebnisse als `ST_Contains` und `ST_Within`.

Wenn die Prädikattests nicht das erwartete Ergebnis liefern, sollten Sie die Methode ST_Relate verwenden, um die getestete Beziehung genau festzulegen.

Siehe auch

- „Funktionsweise räumlicher Beziehungen“ auf Seite 52

Konformität mit Standards für räumliche Daten

SQL Anywhere befolgt folgende Standards für räumliche Daten:

- **International Organization for Standardization (ISO)** SQL Anywhere-Geometrien entsprechen den ISO-Standards zur Definition von räumlichen Benutzertypen, Routinen, Schemata sowie zur Verarbeitung räumlicher Daten. SQL Anywhere entspricht den spezifischen Empfehlungen des internationalen Standards ISO/IEC 13249-3:2006. Siehe http://www.iso.org/iso/catalogue_detail.htm?csnumber=38651.
- **OGC-Geometriemodell (Open Geospatial Consortium)** SQL Anywhere-Geometrien entsprechen den vom OGC definierten OpenGIS-Implementierungsspezifikationen für Geoinformationsdaten - Simple Feature Access - Part 2: SQL-Option Version 1.2.0 (OGC 06-104r3). Siehe <http://www.opengeospatial.org/standards/sfs>.

SQL Anywhere verwendet die vom OGC empfohlenen Standards, um sicherzustellen, dass verschiedene Hersteller und Anwendungen gemeinsam auf räumliche Informationen zugreifen können.

Um die Kompatibilität mit räumlichen Geometrien in SQL Anywhere sicherzustellen, wird empfohlen, die vom OGC festgelegten Standards einzuhalten.

- **SQL Multimedia (SQL/MM)** SQL Anywhere folgt dem SQL/MM-Standard und verwendet das Präfix **ST_** für alle Methoden- und Funktionsnamen.

SQL/MM ist ein internationaler Standard, der festlegt, wie räumliche Daten mit SQL gespeichert, abgerufen und verarbeitet werden. Räumliche Datentypenhierarchien wie ST_Geometry gehören zu den Methoden für die Abfrage räumlicher Daten. Die ST_Geometry-Hierarchie umfasst eine Reihe von Subtypen wie ST_Point, ST_Curve und ST_Polygon. Beim SQL/MM-Standard muss jeder räumliche Wert in einer Abfrage im gleichen räumlichen Bezugssystem definiert werden.

Besondere Hinweise zu Unterstützung und Konformität

Dieser Abschnitt beschreibt alle besonderen Hinweise zur Unterstützung von SQL Anywhere für räumliche Daten, einschließlich nicht unterstützter Funktionen und wichtiger Verhaltensunterschiede gegenüber anderen Datenbankprodukten.

- **Geografien und Geometrien** Einige Hersteller unterscheiden räumliche Objekte danach, ob es sich um **Geografien** (sie betreffen Objekte auf einer runden Erde) oder **Geometrien** (sie betreffen Objekte auf einer Ebene oder planen Erde) handelt. In SQL Anywhere werden alle räumlichen Objekte als Geometrien bezeichnet und die SRID des Objekts gibt an, ob es in einem räumlichen Bezugssystem mit gewölbter Erddarstellung oder mit planer Erddarstellung bearbeitet wird.

- **Nicht unterstützte Methoden**

- ST_Buffer-Methode
- ST_LocateAlong-Methode
- ST_LocateBetween-Methode
- ST_Segmentize-Methode
- ST_Simplify-Methode
- ST_Distance_Spheroid-Methode
- ST_Length_Spheroid-Methode

Unterstützte Import- und Exportformate für räumliche Daten

In der folgenden Tabelle sind die von SQL Anywhere für den Import und Export von räumlichen Daten unterstützten Formate aufgelistet:

Datenformat	Importieren	Exportieren	Beschreibung
Well Known Text (WKT)	Yes	Yes	<p>Geografische Daten ausgedrückt in ASCII-Text. Dieses Format wird vom Open Geospatial Consortium (OGC) als Teil der einfachen Funktionen unterstützt, die für die OpenGIS-Implementierungsspezifikation für Geoinformationsdaten definiert sind. Siehe www.opengeospatial.org/standards/sfa.</p> <p>Beispiel der Darstellung eines Punkts in WKT:</p> <pre>'POINT(1 1)'</pre>
Well Known Binary (WKB)	Yes	Yes	<p>Geografische Daten ausgedrückt als Binärdatenstrom. Dieses Format wird vom OGC als Teil der einfachen Funktionen unterstützt, die für die OpenGIS-Implementierungsspezifikation für Geoinformationsdaten definiert sind. Siehe www.opengeospatial.org/standards/sfa.</p> <p>Beispiel der Darstellung eines Punkts in WKB:</p> <pre>'010100000000000000000000F03FF0000000000000F03F'</pre>
Extended Well Known Text (EWKT)	Yes	Yes	<p>WKT-Format, jedoch mit eingebetteten SRID-Informationen. Dieses Format wird als Teil von PostGIS, der räumlichen Datenbankerweiterung von PostgreSQL, verwaltet. Siehe postgis.refractions.net/.</p> <p>Beispiel der Darstellung eines Punkts in EWKT:</p> <pre>'srid=101;POINT(1 1)'</pre>

Datenformat	Importieren	Exportieren	Beschreibung
Extended Well Known Binary (EWKB)	Yes	Yes	<p>WKB-Format, jedoch mit eingebetteten SRID-Informationen. Dieses Format wird als Teil von PostGIS, der räumlichen Datenbankerweiterung von PostgreSQL, verwaltet. Siehe postgis.refractory.net/.</p> <p>Beispiel der Darstellung eines Punkts in EWKB:</p> <pre>'01010000020040000000000000000000F03F000000000000F03F'</pre>
Geographic Markup Language (GML)	No	Yes	<p>XML-Grammatik zur Darstellung geografischer räumlicher Daten. Dieser Standard wird vom Open Geospatial Consortium (OGC) verwaltet und dient zum Austausch von geografischen Daten über das Internet. Siehe www.opengeospatial.org/standards/sfa.</p> <p>Beispiel der Darstellung eines Punkts in GML:</p> <pre><gml:Point> <gml:coordinates>1,1</gml:coordinates> </gml:Point></pre>
KML	No	Yes	<p>Diese ehemals als "Google Keyhole Markup Language" bezeichnete XML-Grammatik wird verwendet, um geografische Daten wie Visualisierungs- und Navigationshilfen darzustellen, und bietet die Möglichkeit, Karten und Bilder mit Anmerkungen zu versehen. Google schlug diesen Standard dem OGC vor. Das OGC hat ihn als offenen Standard akzeptiert, der nun als KML bezeichnet wird. Siehe www.opengeospatial.org/standards/sfa.</p> <p>Beispiel der Darstellung eines Punkts in KML:</p> <pre><Point> <coordinates>1,0</coordinates> </Point></pre>
ESRI-Formdateien	Yes	No	<p>Ein häufig verwendetes Geodaten-Vektorformat zur Darstellung von räumlichen Objekten in Form von Formdateien (mehrere Dateien, die zusammen zur Festlegung der Form verwendet werden). Weitere Hinweise zur Unterstützung von ESRI-Formdateien finden Sie unter „Unterstützung von ESRI-Formdateien“ auf Seite 15.</p>

Datenformat	Importieren	Exportieren	Beschreibung
GeoJSON	No	Yes	<p>Textformat, das Namen-/Wert-Paare, sortierte Wertelisten und Konventionen ähnlich denen in gewöhnlichen Programmiersprachen wie C, C++, C#, Java, JavaScript, Perl und Python verwendet.</p> <p>GeoJSON ist eine Teilmenge des JSON-Standards und wird zur Kodierung von geografischen Informationen verwendet. SQL Anywhere unterstützt den GeoJSON-Standard und stellt die <code>ST_AsGeoJSON</code>-Methode zum Konvertieren einer SQL-Ausgabe in das GeoJSON-Format bereit. Siehe ST_AsGeoJSON-Methode für den ST_Geometry-Datentyp auf Seite 122.</p> <p>Beispiel der Darstellung eines Punkts in GeoJSON:</p> <pre>{"x" : 1, "y" : 1, "spatialReference" : {"wkid" : 4326}}</pre> <p>Weitere Hinweise zur GeoJSON-Spezifikation finden Sie unter geojson.org/geojson-spec.html.</p>
SVG-Dateien (Scalable Vector Graphic)	No	Yes	<p>XML-basiertes Format zur Darstellung zweidimensionaler Geometrien. Das SVG-Format wird vom World Wide Web Consortium (W3C) verwaltet. Siehe www.w3.org/Graphics/SVG/.</p> <p>Beispiel der Darstellung eines Punkts in SVG:</p> <pre><rect width="1" height="1" fill="deepskyblue" stroke="black" stroke-width="1" x="1" y="-1"/></pre>

Unterstützung von ESRI-Formdateien

SQL Anywhere unterstützt das Formdateiformat des Environmental System Research Institute, Inc. (ESRI). ESRI-Formdateien werden verwendet, um geometrische Daten und Attributinformationen für die räumlichen Merkmale in einem Datensatz zu speichern.

Eine ESRI-Formdatei enthält mindestens drei verschiedene Dateien mit den Suffixen *.shp*, *.shx* und *.dbf*. Das Suffix für die Hauptdatei ist *.shp*, das Suffix für die Indexdatei ist *.shx* und das Suffix für die Attributspalten ist *.dbf*. Alle Dateien haben den gleichen Grundnamen und werden häufig in einer einzigen komprimierten Datei kombiniert. SQL Anywhere kann alle ESRI-Formdateien mit allen Formtypen ausgenommen MultiPatch lesen. Dies umfasst Formtypen, die Z- und M-Daten enthalten.

Die Daten in einer ESRI-Formdatei enthalten üblicherweise mehrere Zeilen und Spalten. Beispiel: Die praktische Einführung zu räumlichen Daten lädt eine Formdatei, die Zipcode-Regionen (Zipcodes entsprechen unseren Postleitzahlen) für Massachusetts/USA enthält. Die Formdatei enthält eine Zeile für jede Zipcode-Region, einschließlich der Polygoninformationen für die Region. Sie enthält außerdem zusätzliche Attribute (Spalten) für jede Zipcode-Region, einschließlich den Zipcode-Namen (z.B. die Zeichenfolge '02633') und andere Attribute.

Die einfachste Möglichkeit zum Einlesen einer Formdatei in eine Tabelle bietet der **Import-Assistent** von Interactive SQL oder die `st_geometry_load_shapefile`-Systemprozedur. Beide Tools erstellen eine Tabelle mit den entsprechenden Spalten und lesen die Daten aus der Formdatei ein.

Sie können Formdateien auch mit den Anweisungen `LOAD TABLE` und `INPUT` einlesen. Vor der Durchführung des Einlesevorgangs müssen Sie jedoch bereits die Tabelle mit den entsprechenden Spalten erstellt haben.

Zur Suche der benötigten Spalten beim Einlesen von Daten mit den Anweisungen `LOAD TABLE` oder `INPUT` können Sie die `sa_describe_shapefile`-Systemprozedur verwenden.

Ein Beispiel für das Laden einer Formdatei in eine SQL Anywhere-Datenbank finden Sie unter „Praktische Einführung: Mit den räumlichen Funktionen experimentieren“ auf Seite 57.

Weitere Hinweise zu Abfragen in einer Formdatei finden Sie unter [Openstring-Ausdrücke in einer FROM-Klausel \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Weitere Hinweise zu ESRI-Formdateien finden Sie unter <http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf>.

Siehe auch

- „Importieren von Daten mit dem Import-Assistenten“ [[SQL Anywhere Server - SQL-Benutzerhandbuch](#)]
- „`st_geometry_load_shapefile`-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „`LOAD TABLE`-Anweisung“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „`sa_describe_shapefile`-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „`INPUT`-Anweisung [Interactive SQL]“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]

Empfohlene Lektüre zu räumlichen Themen

- Eine gute Einführung in die verschiedenen Herangehensweisen, die zur Abbildung und Vermessung der Erdoberfläche (Geodäsie) verwendet werden, sowie in die wichtigsten Konzepte in Bezug auf räumliche (oder Koordinaten-)Referenzsysteme finden Sie unter www.epsg.org/guides/index.html, und wählen Sie dann "Geodetic Awareness".
- Von OGC definierte OpenGIS-Implementierungsspezifikation für Geoinformationsdaten - Simple Feature Access: www.opengeospatial.org/standards/sfs
- Internationaler Standard ISO/IEC 13249-3:2006: www.iso.org/iso/catalogue_detail.htm?csnumber=38651
- SVG-Spezifikation (Scalable Vector Graphics 1.1): www.w3.org/Graphics/SVG/
- GML-Spezifikation (Geographic Markup Language): www.opengeospatial.org/standards/gml
- KML-Spezifikation: www.opengeospatial.org/standards/kml
- JavaScript Object Notation (JSON): json.org
- GeoJSON-Spezifikation: geojson.org/geojson-spec.html

Erstellen einer räumlichen Spalte (Sybase Central)

Sie können jeder beliebigen Tabelle räumliche Daten hinzufügen, indem Sie eine Spalte hinzufügen, die räumliche Daten unterstützt.

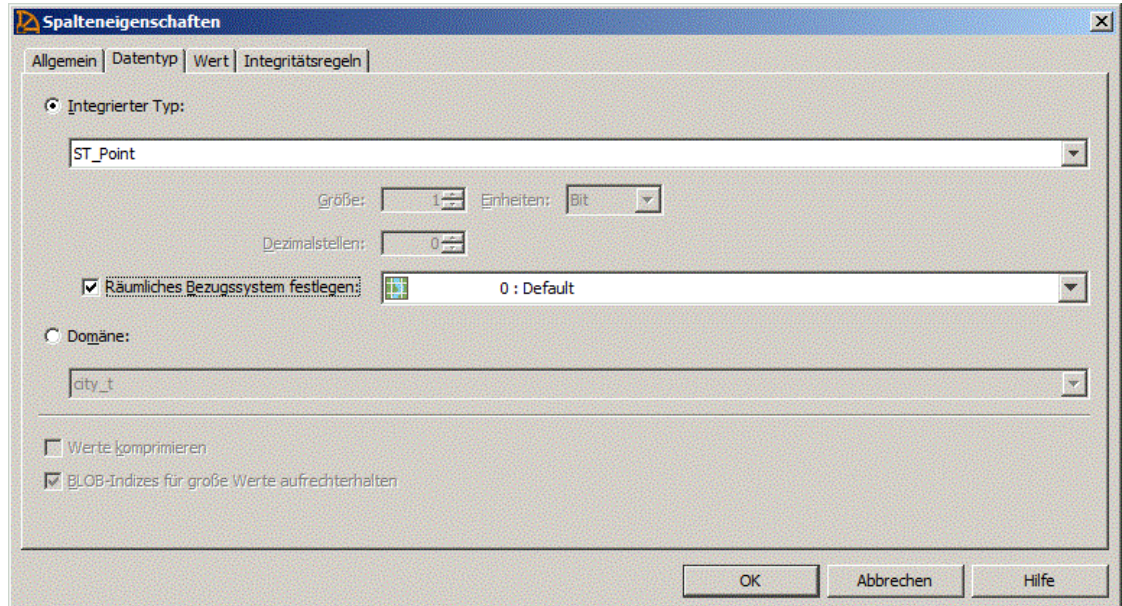
Voraussetzungen

Sie müssen Eigentümer der Tabelle sein, das ALTER-Privileg für die Tabelle haben oder das ALTER ANY TABLE-Systemprivileg oder das ALTER ANY OBJECT-Systemprivileg haben.

Die Tabelle muss einen Primärschlüssel haben. Aktualisierungs- und Löschvorgänge werden für Tabellen, die eine räumliche Spalte enthalten, nur unterstützt, wenn für sie ein Primärschlüssel festgelegt ist.

Erstellen einer räumlichen Spalte (Sybase Central)

1. Stellen Sie eine Verbindung mit der Datenbank her.
2. Erstellen Sie eine neue Spalte:
 - Erweitern Sie im linken Fensterausschnitt die Liste **Tabellen**.
 - Rechtsklicken Sie auf eine Tabelle und klicken Sie dann auf **Neu » Spalte**.
3. Legen Sie den räumlichen Datentyp fest:
 - a. Rechtsklicken Sie auf den Spaltennamen und klicken Sie dann auf **Eigenschaften**.
 - b. Klicken Sie auf die Registerkarte **Datentyp**.
 - c. Klicken Sie auf **Integrierter Typ** und wählen Sie in der Dropdown-Liste einen räumlichen Datentyp.
 - d. Klicken Sie auf **Räumliches Bezugssystem festlegen** und wählen Sie in der Dropdown-Liste ein räumliches Bezugssystem.



e. Klicken Sie auf **OK**.

4. Klicken Sie auf **Datei » Speichern**.

Ergebnisse

Eine räumliche Spalte wird zur vorhandenen Tabelle hinzugefügt.

Nächste Schritte

Sie können die ALTER TABLE-Anweisung verwenden, um SRID-Integritätsregeln für die Spalte festzulegen und dadurch die Werte einzuschränken, die in einer räumlichen Spalte gespeichert werden können. Außerdem können Sie der Spalte räumliche Daten hinzufügen.

Siehe auch

- „Unterstützte räumliche Datentypen und ihre Hierarchie“ auf Seite 7
- „Räumliche Bezugssysteme (SRS) und räumliche Referenz-IDs (SRID)“ auf Seite 2
- „ALTER TABLE-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Hinzufügen von SRID-Spaltenintegritätsregeln“ auf Seite 18

Hinzufügen von SRID-Spaltenintegritätsregeln

SRID-Integritätsregeln ermöglichen es, Einschränkungen für die Werte festzulegen, die in einer räumlichen Spalte gespeichert werden können. Damit eine räumliche Spalte in einen Index einbezogen werden kann, muss sie eine SRID-Integritätsregel besitzen. Diese kann mit den Anweisungen CREATE TABLE und ALTER TABLE hinzugefügt werden.

Voraussetzungen

Um eine Tabelle ändern zu können, müssen Sie Eigentümer der Tabelle sein, das ALTER-Privileg für die Tabelle haben oder das ALTER ANY TABLE-Systemprivileg oder das ALTER ANY OBJECT-Systemprivileg haben.

Wenn Sie einer Tabelle eine räumliche Spalte hinzufügen, sollten Sie sicherstellen, dass für die Tabelle ein Primärschlüssel definiert ist. Aktualisierungs- und Löschvorgänge werden für Tabellen, die eine räumliche Spalte enthalten, nur unterstützt, wenn für sie ein Primärschlüssel definiert ist.

Kontext und Bemerkungen

Spalten mit räumlichen Daten können nicht in einen Primärschlüssel, eindeutigen Index oder eine Eindeutigkeits-Integritätsregel einbezogen werden.

Hinzufügen von SRID-Spaltenintegritätsregeln

- Führen Sie eine CREATE TABLE- oder ALTER TABLE-Anweisung aus, die die SRID-Integritätsregel für die räumliche Spalte enthält.

```
CREATE TABLE Test (  
    ID INTEGER PRIMARY KEY,  
    Geometry_1 ST_Geometry,  
    Geometry_2 ST_Geometry(SRID=4326),  
);
```

Ergebnisse

Die SRID-Integritätsregel wird zu der räumlichen Spalte in der Tabelle hinzugefügt.

Nächste Schritte

Sie können die räumliche Spalte in einem Index einbeziehen.

Beispiel

Beispiel: Führen Sie folgende Anweisung aus, um eine Tabelle namens Test mit einer SRID-Integritätsregel (SRID=4326) in der Spalte Geometry_2 zu erstellen:

```
CREATE TABLE Test (  
    ID INTEGER PRIMARY KEY,  
    Geometry_1 ST_Geometry,  
    Geometry_2 ST_Geometry(SRID=4326),  
);
```

Diese Integritätsregel bedeutet, dass nur Werte in dieser Spalte gespeichert werden können, die SRID 4326 zugeordnet sind.

Die Spalte Geometry_1 hat keine Integritätsregel und kann Werte, die einer beliebigen SRID zugeordnet sind, speichern.

Sie können keinen Index für die Geometry_1-Spalte erstellen. Sie können jedoch einen Index für die Spalte Geometry_2 erstellen.

Wenn Sie eine Tabelle mit einer vorhandenen räumlichen Spalte haben, können Sie dieser Spalte mit der Anweisung `ALTER TABLE` eine SRID hinzufügen. Beispiel: Fügen Sie der Spalte `Geometry_1` in der Tabelle `Test` mit einer Anweisung ähnlich der folgenden eine Integritätsregel hinzu:

```
ALTER TABLE Test
MODIFY Geometry_1 ST_Geometry(SRID=4326);
```

Siehe auch

- „CREATE INDEX-Anweisung“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „CREATE TABLE-Anweisung“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „ALTER TABLE-Anweisung“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „Erstellen einer räumlichen Spalte (Sybase Central)“ auf Seite 17
- „Räumliche Spalten erstellen (SQL)“ auf Seite 20

Räumliche Spalten erstellen (SQL)

Sie können jeder beliebigen Tabelle räumliche Daten hinzufügen, indem Sie eine Spalte hinzufügen, die räumliche Daten unterstützt.

Voraussetzungen

Sie müssen Eigentümer der Tabelle sein, das `ALTER`-Privileg für die Tabelle haben oder das `ALTER ANY TABLE`-Systemprivileg oder das `ALTER ANY OBJECT`-Systemprivileg haben.

Räumliche Spalten erstellen (SQL)

1. Stellen Sie eine Verbindung mit der Datenbank her.
2. Führen Sie die Anweisung `ALTER TABLE` aus.

Ergebnisse

Eine räumliche Spalte wird zur vorhandenen Tabelle hinzugefügt.

Nächste Schritte

Mithilfe von SRID-Integritätsregeln für die Spalte können Sie Einschränkungen für die Werte festlegen, die in einer räumlichen Spalte gespeichert werden können.

Beispiel

Mit der folgenden Anweisung wird der Tabelle `Customers` die Spalte `Location` hinzugefügt. Die neue Spalte hat den räumlichen Datentyp `ST_Point` und ihre deklarierte SRID ist `1000004326`, ein räumliches Bezugssystem mit planer Erddarstellung.

```
ALTER TABLE GROUP0.Customers
ADD Location ST_Point(SRID=1000004326);
```

Siehe auch

- „Unterstützte räumliche Datentypen und ihre Hierarchie“ auf Seite 7
- „Hinzufügen von SRID-Spaltenintegritätsregeln“ auf Seite 18
- „Räumliche Bezugssysteme (SRS) und räumliche Referenz-IDs (SRID)“ auf Seite 2
- „ALTER TABLE-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Indizes für räumliche Spalten

Verwenden Sie bei der Erstellung eines räumlichen Indexes die CREATE INDEX-Anweisung oder den **Assistenten zum Erstellen von Indizes** wie bei der Erstellung eines Indexes für jeden anderen Datentyp. Bei der Erstellung von Indizes für räumliche Daten wird dagegen empfohlen, nicht mehr als eine räumliche Spalte in den Index einzubeziehen, und die räumliche Spalte an das Ende der Indexdefinition zu stellen.

Um eine räumliche Spalte in einen Index einzubeziehen, muss diese Spalte außerdem eine SRID-Integritätsregel aufweisen.

Indizes für räumliche Daten können die Kosten für die Bewertung der Beziehungen zwischen Geometrien verringern. Beispiel: Sie wollen die Grenzen der Vertriebsregionen ändern und möchten die Auswirkungen auf die bestehenden Kunden ermitteln. Um festzustellen, welche Kunden sich in einer bestimmten Vertriebsregion befinden, können Sie die ST_Within-Methode verwenden, um einen Punkt, der eine Kundenadresse darstellt, mit einem Polygon zu vergleichen, das die Vertriebsregion repräsentiert. Ohne jeglichen Index muss der Datenbankserver jeden Adresspunkt in der Kundentabelle anhand der Vertriebsregion überprüfen, um zu ermitteln, ob sie in dem Ergebnis zurückgegeben werden soll. Dies kann kostspielig sein, wenn die Kundentabelle groß ist, und ineffizient, wenn die Vertriebsregion klein ist. Ein Index, der den Adresspunkt jedes Kunden einbezieht, kann dazu beitragen, die Ergebnisse schneller zurückzugeben. Wenn der Abfrage ein Prädikat hinzugefügt werden kann, das die Vertriebsregion den Staaten zuordnet, die sie überlappt, können die Ergebnisse noch schneller mit einem Index ermittelt werden, der sowohl den Code des Staats als auch den Adresspunkt einbezieht.

Abfragen auf räumliche Daten ziehen *möglicherweise* Nutzen aus einem Clustered-Index. Vor der Verwendung eines Clustered-Indexes sollte jedoch die sonstige Nutzung der Tabelle berücksichtigt werden. Sie sollten die Typen von Abfragen, die wahrscheinlich durchgeführt werden, prüfen und testen, ob die Performance durch die Clustered-Indizes verbessert wird.

Es können auch Textindizes für eine räumliche Spalte erstellt werden. Diese bieten jedoch keinen Vorteil gegenüber normalen Indizes, weshalb stattdessen normale Indizes empfohlen werden.

Hinweis

Spalten mit räumlichen Daten können nicht in einen Primärschlüssel, eindeutigen Index oder eine Eindeutigkeits-Integritätsregel einbezogen werden.

Siehe auch

- „Indizes“ [[SQL Anywhere Server - SQL-Benutzerhandbuch](#)]
- „Indizes erstellen“ [[SQL Anywhere Server - SQL-Benutzerhandbuch](#)]
- „CREATE INDEX-Anweisung“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „ALTER INDEX-Anweisung“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- Deklarieren von Clustered-Indizes [[SQL Anywhere Server - SQL-Benutzerhandbuch](#)]
- „Hinzufügen von SRID-Spaltenintegritätsregeln“ auf Seite 18

Syntax räumlicher Datentypen

Der SQL/MM-Standard definiert die Unterstützung für räumliche Daten hinsichtlich benutzerdefinierter erweiterter Typen (UDTs, User-defined Extended Types), die mit der Anweisung ANSI/SQL CREATE TYPE erstellt werden. Wenngleich SQL Anywhere keine benutzerdefinierten Typen unterstützt, ist die Unterstützung für räumliche Daten durch SQL Anywhere so implementiert, als würden sie unterstützt.

Instanziierung eines UDT-Typs

Sie können einen Wert eines benutzerdefinierten Datentyps instanziiieren, indem Sie einen Konstruktor wie folgt aufrufen:

NEW *type-name*(*argument-list*)

Eine Abfrage könnte z. B. Folgendes enthalten, um zwei ST_Point-Werte zu instanziiieren:

```
SELECT NEW ST_Point(), NEW ST_Point(3,4)
```

SQL Anywhere vergleicht die *argument-list* mit den definierten Konstruktoren unter Benutzung der normalen Regeln zur Auflösung von Überladungen. In folgenden Situationen wird ein Fehler zurückgegeben:

- Wenn für einen nicht benutzerdefinierten Typ NEW verwendet wird.
- Wenn der benutzerdefinierte Typ nicht instanziiierbar ist (ST_Geometry ist z. B. ein nicht-instanziiierbarer Typ).
- Wenn keine Überladung vorliegt, die mit den bereitgestellten Argumenttypen übereinstimmt.

Siehe:

- „Zugriff auf räumliche Daten und ihre Verarbeitung“
- „ST_Point-Datentyp“

Instanzmethoden verwenden

Für benutzerdefinierte Typen können Instanzmethoden definiert werden. Instanzmethoden werden wie folgt für einen Wert des Typs aufgerufen:

value-expression.method-name(*argument-list*)

Folgendes fiktive Beispiel wählt die X-Koordinate der Spalte Massdata.CenterPoint aus:

```
SELECT CenterPoint.ST_X() FROM Massdata;
```

Wenn es eine Benutzer-ID namens CenterPoint gibt, betrachtet der Datenbankserver `CenterPoint.ST_X()` als **mehrdeutig**. Der Grund hierfür ist, dass die Anweisung bedeuten könnte: "Aufruf der benutzerdefinierten Funktion `ST_X`, die dem Benutzer CenterPoint gehört" (nicht das gewünschte Ergebnis). Oder sie könnte bedeuten: "Aufruf der `ST_X`-Methode für die Spalte `Massdata.CenterPoint`" (das gewünschte Ergebnis). Der Datenbankserver löst die Mehrdeutigkeit auf, indem er zuerst eine Suche ohne Beachtung der Groß-/Kleinschreibung nach einem Benutzer mit dem Namen CenterPoint durchführt. Wenn einer gefunden wird, verhält sich der Datenbankserver so, als ob eine benutzerdefinierte Funktion namens `ST_X` aufgerufen würde, die dem Benutzer CenterPoint gehört. Wenn kein Benutzer gefunden wird, behandelt der Datenbankserver das Konstrukt als Methodenaufruf und ruft die `ST_X`-Methode für die Spalte `Massdata.CenterPoint` auf.

Der Aufruf einer Instanzmethode gibt in folgenden Fällen einen Fehler zurück:

- Wenn der deklarierte Typ von *value-expression* kein benutzerdefinierter Typ ist.
- Wenn die benannte Methode nicht im deklarierten Typ von *value-expression* oder einem seiner Obertypen definiert ist.
- Wenn *argument-list* mit keiner der definierten Überladungen für die Methode übereinstimmt.

Siehe:

- [„ST_X\(\)-Methode für den ST_Point-Datentyp“](#)

Statische Methoden verwenden

Zusätzlich zu Instanzmethoden erlaubt es der ANSI/SQL-Standard, benutzerdefinierten Typen statische Methoden zuzuordnen. Diese werden mit der folgenden Syntax aufgerufen:

```
type-name::method-name( argument-list )
```

Folgende Anweisung instanziiert z. B. einen `ST_Point` durch die syntaktische Analyse von Text:

```
SELECT ST_Geometry::ST_GeomFromText( 'POINT( 5 6 )' )
```

Der Aufruf einer statischen Methode gibt in folgenden Fällen einen Fehler zurück:

- Wenn der deklarierte Typ von *value-expression* kein benutzerdefinierter Typ ist.
- Wenn die Methode nicht im Typ von *type-value-expression* oder einem seiner Obertypen definiert ist.
- Wenn *argument-list* mit keiner der definierten Überladungen für die benannte Methode übereinstimmt.

Siehe:

- [„ST_Point-Datentyp“](#)
- [ST_GeomFromText-Methode für den ST_Geometry-Datentyp](#)

Statische Aggregatmethoden verwenden (SQL Anywhere-Erweiterung)

Als Erweiterung zu ANSI/SQL unterstützt SQL Anywhere statische Methoden, die benutzerdefinierte Aggregate implementieren. Beispiel:

```
SELECT ST_Geometry::ST_AsSVGAggr(T.geo) FROM table T
```

Alle Überladungen für eine statische Methode müssen Aggregate sein oder keine von ihnen darf ein Aggregat sein.

Der Aufruf einer statischen Aggregatmethode gibt in folgenden Fällen einen Fehler zurück:

- Wenn der Aufruf einer statischen Methode einen Fehler liefern würde
- Wenn eine integrierte Aggregatfunktion einen Fehler liefern würde
- Wenn eine WINDOW-Klausel angegeben wurde

Siehe:

- [ST_AsSVGAggr-Methode für den ST_Geometry-Datentyp](#)

Typenprädikate verwenden

Der ANSI/SQL-Standard definiert Typenprädikate, die es einer Anweisung gestatten, den konkreten Typ (in anderen Sprachen auch als Objekttyp bezeichnet) eines Werts zu überprüfen. Die Syntax lautet wie folgt:

```
value IS [ NOT ] OF ( [ ONLY ] type-name,...)
```

Wenn *value* NULL ist, gibt das Prädikat UNKNOWN zurück. Andernfalls wird der konkrete Typ von *value* mit den einzelnen Elementen in der Liste *type-name* verglichen. Wenn ONLY angegeben wurde, liegt eine Übereinstimmung vor, wenn der konkrete Typ genau dem angegebenen Typ entspricht. Andernfalls liegt eine Übereinstimmung vor, wenn der konkrete Typ der angegebene Typ oder ein abgeleiteter Typ (Subtyp) ist.

Wenn der konkrete Typ von *value* mit einem der Elemente in der Liste übereinstimmt, wird TRUE zurückgegeben, andernfalls FALSE.

Das folgende Beispiel gibt alle Zeilen zurück, in denen der Wert der Shape-Spalte den konkreten Typ ST_Curve oder einen seiner Subtypen (ST_LineString, ST_CircularString oder ST_CompoundCurve) hat:

```
SELECT * FROM SpatialShapes WHERE Shape IS OF ( ST_Curve );
```

Siehe:

- „Suchbedingungen“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „ST_Point-Datentyp“
- [ST_GeomFromText-Methode für den ST_Geometry-Datentyp](#)

Ausdruck TREAT für Subtypen verwenden

Der ANSI/SQL-Standard definiert einen Subtyp-Behandlungsausdruck, der es ermöglicht, den deklarierten Typ eines Ausdrucks effizient von einem Obertyp in einen Subtyp zu ändern. Dieser kann

verwendet werden, wenn Sie wissen, dass der konkrete Typ (in anderen Sprachen auch als Objekttyp bezeichnet) des Ausdrucks der angegebene Subtyp oder ein Subtyp des angegebenen Subtyps ist. Dies ist effizienter als die Verwendung der CAST-Funktion, da diese eine Kopie des Werts erstellt, während TREAT keine Kopie erstellt. Die Syntax lautet wie folgt:

TREAT(*value-expression* AS *target-subtype*)

Wenn keine Fehlerbedingung ausgegeben wird, ist das Ergebnis der *value-expression* mit dem deklarierten *target-subtype*.

Der Ausdruck zur Subtypbehandlung gibt in folgenden Fällen einen Fehler zurück:

- Wenn *value-expression* kein benutzerdefinierter Typ ist
- Wenn *target-subtype* kein Subtyp des deklarierten Typs von *value-expression* ist
- Wenn der dynamische Typ von *value-expression* kein Subtyp von *target-subtype* ist

Im folgenden Beispiel wird der deklarierte Typ ST_Geometry der Shape-Spalte in den ST_Curve-Subtyp geändert, sodass die ST_Length-Methode des Typs ST_Curve aufgerufen werden kann:

```
SELECT ShapeID, TREAT( Shape AS ST_Curve ).ST_Length() FROM SpatialShapes
WHERE Shape IS OF ( ST_Curve );
```

Siehe:

- „TREAT-Funktion [Datentypkonvertierung]“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „ST_Point-Datentyp“
- ST_GeomFromText-Methode für den ST_Geometry-Datentyp

Geometrien erstellen

Es gibt mehrere Methoden zur Erstellung von Geometrien in einer Datenbank:

- **Aus WKT- (Well Known Text) oder WKB-Format (Well Known Binary) laden** Sie können Daten im WKT- oder WKB-Format laden bzw. einfügen. Diese Formate sind vom OGC definiert und werden von allen Herstellern räumlicher Datenbanken unterstützt. SQL Anywhere führt die automatische Konvertierung von diesen Formaten in Geometrietypen aus.
- **Aus ESRI-Formdateien einlesen** Sie können Daten aus ESRI-Formdateien in eine neue oder eine vorhandene Tabelle laden. Es gibt mehrere Möglichkeiten, um diese Aufgabe auszuführen.
- **SELECT...FROM OPENSTRING-Anweisung verwenden** Sie können eine SELECT...FROM OPENSTRING-Anweisung für eine Datei ausführen, die die räumlichen Daten enthält. Beispiel:

```
INSERT INTO world_cities( country, city, point )
SELECT country, city, NEW ST_Point( longitude, latitude, 4326 )
FROM OPENSTRING( FILE 'capitalcities.csv' )
WITH(
    country    CHAR(100),
    city       CHAR(100),
```

```
latitude DOUBLE,  
longitude DOUBLE )
```

- **Koordinatenpunkte durch die Kombination von Breitengrad- und Längengradwerten erstellen** Sie können Breitengrad- und Längengraddaten kombinieren, um eine Koordinate vom räumlichen Datentyp ST_Point zu erstellen. Beispiel: Wenn Sie eine Tabelle haben, die bereits Breitengrad- und Längengrad-Spalten enthält, können Sie eine ST_Point-Spalte erstellen, in der die Werte als Punkte gespeichert werden. Verwenden Sie hierzu folgende Anweisung:

```
ALTER TABLE my_table  
ADD point AS ST_Point(SRID=4326)  
COMPUTE( NEW ST_Point( longitude, latitude, 4326 ) );
```

- **Geometrien mit Konstruktoren und statischen Methoden erstellen** Sie können Geometrien mithilfe von Konstruktoren und statischen Methoden erstellen.

Siehe auch

- [„Laden von räumlichen Daten aus einer WKT-Datei \(Well Known Text\)“ auf Seite 31](#)
- [„Unterstützung von ESRI-Formdateien“ auf Seite 15](#)
- [Openstring-Ausdrücke in einer FROM-Klausel \[*SQL Anywhere Server - SQL-Referenzhandbuch*\]](#)
- [Instanziierung eines UDT-Typs auf Seite 22](#)
- [Statische Methoden verwenden auf Seite 23](#)

Anzeigen von räumlichen Daten als Bilder (Interactive SQL)

In Interactive SQL können Sie mithilfe der Registerkarte **Räumliche Vorschau** eine Geometrie als Bild anzeigen, um zu erkennen, was die Daten in der Datenbank darstellen.

Voraussetzungen

Sie müssen das SELECT-Privileg für die Tabelle haben, aus der Sie auswählen, oder das SELECT ANY TABLE-Systemprivileg.

Kontext und Bemerkungen

Jede Instanz von Interactive SQL ist einer eigenen Verbindung mit einer Datenbank zugeordnet. Wenn Sie eine Instanz des **Spatial Viewers** in Interactive SQL öffnen, bleibt diese Instanz des **Spatial Viewers** weiterhin dieser Instanz von Interactive SQL zugeordnet und teilt die Verbindung zur Datenbank mit ihm.

Ein Fehler wird ausgegeben, wenn Sie beim Ausführen einer Abfrage im **Spatial Viewer** versuchen, eine Abfrage in der zugeordneten Instanz von Interactive SQL auszuführen. Ebenso gilt: Wenn Sie mehrere Instanzen des **Spatial Viewers** öffnen, die von derselben Instanz von Interactive SQL erstellt wurden, kann jeweils nur eine dieser Instanzen eine Abfrage ausführen. Die anderen Instanzen müssen warten, bis die Abfrage abgeschlossen ist.

Hinweis

Standardmäßig kürzt Interactive SQL-Werte im Fensterausschnitt **Ergebnisse** auf 256 Zeichen. Wenn Interactive SQL den Fehler meldet, dass der vollständige Spaltenwert nicht gelesen werden konnte, erhöhen Sie den Kürzungswert. Klicken Sie dazu auf **Extras » Optionen** und anschließend im linken Fensterausschnitt auf **SQL Anywhere**. Ändern Sie auf der Registerkarte **Ergebnisse** die Option **Kürzungslänge** auf einen hohen Wert, z.B. 5000. Klicken Sie auf **OK**, um Ihre Änderungen zu speichern, führen Sie die Abfrage noch einmal aus und doppelklicken Sie anschließend erneut auf die Zeile.

Anzeigen von räumlichen Daten als Bilder (Interactive SQL)

1. Verbinden Sie sich über Interactive SQL mit Ihrer Datenbank.
2. Führen Sie eine Abfrage aus, um räumliche Daten aus einer Tabelle auszuwählen. Beispiel:

```
SELECT * FROM owner.spatial-table;
```

3. Doppelklicken Sie im Fensterausschnitt **Ergebnisse** auf einen beliebigen Wert in der Spalte "Formen", um den Wert im Fenster **Wert** zu öffnen.

Der Wert wird im Fenster **Wert** auf der Registerkarte **Text** angezeigt.

4. Klicken Sie auf die Registerkarte **Räumliche Vorschau**, um die Geometrie als skalierbare Vektorgrafik (SVG) anzuzeigen.

Ergebnisse

Die Geometrie wird als skalierbare Vektorgrafik (SVG) angezeigt.

Nächste Schritte

In der Ansicht der räumlichen Daten als Geometrie können Sie mithilfe der Schaltflächen **Vorherige Zeile** und **Nächste Zeile** andere Zeilen in der Ergebnismenge anzeigen.

Beispiel

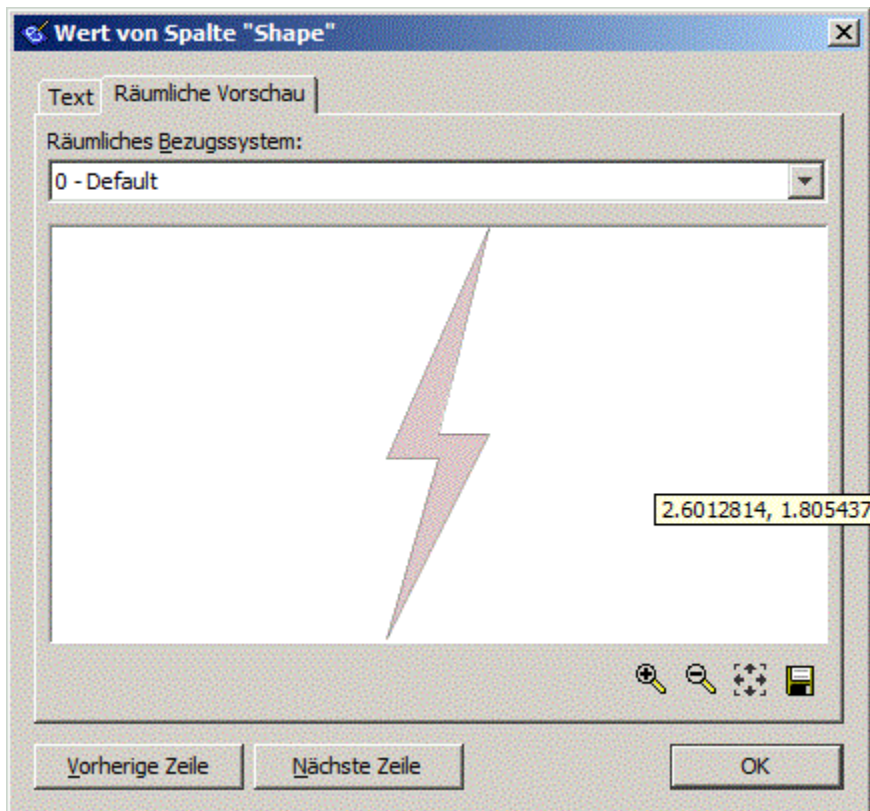
1. Stellen Sie eine Verbindung mit der Beispieldatenbank her und führen Sie die folgende Abfrage aus:

```
SELECT * FROM GROUPO.SpatialShapes;
```

2. Doppelklicken Sie auf einen beliebigen Wert in der Spalte Shapes im Fensterausschnitt **Ergebnisse**, um den Wert im Fenster **Wert** zu öffnen.

Der Wert wird im Fenster **Wert** auf der Registerkarte **Text** angezeigt.

3. Klicken Sie auf die Registerkarte **Räumliche Vorschau**, um die Geometrie als skalierbare Vektorgrafik (SVG) anzuzeigen.



Anzeigen von räumlichen Daten als Bilder (Spatial Viewer)

Im Spatial Viewer können Sie mehrere Geometrien als Bild anzeigen, um zu erkennen, was die Daten in der Datenbank darstellen.

Voraussetzungen

Sie müssen das SELECT-Privileg für die Tabelle haben, aus der Sie auswählen, oder das SELECT ANY TABLE-Systemprivileg.

Kontext und Bemerkungen

Die Reihenfolge der Zahlen in einem Ergebnis beeinflussen, wie das Bild im **Spatial Viewer** angezeigt wird, da das Bild in der Reihenfolge gezeichnet wird, in der die Zeilen verarbeitet werden. Dabei befindet sich die neueste Zeile ganz oben. Formen, die in einer Ergebnismenge weiter unten stehen, können Formen an früheren Stellen in der Ergebnismenge überlagern.

Anzeigen von räumlichen Daten als Bilder (Spatial Viewer)

1. Verbinden Sie sich mit Ihrer Datenbank und klicken Sie in Interactive SQL auf **Extras » Spatial Viewer**.
2. Führen Sie im **Spatial Viewer** eine Abfrage wie die folgende im Fensterausschnitt **SQL** aus und klicken Sie dann auf **Ausführen**:

```
SELECT * FROM GROUPO.SpatialShapes;
```

3. Mit dem Tool **Polygon-Umrisse zeichnen** können Sie die Farbe aus den Polygonen in einer Zeichnung entfernen, um die Umrisse aller Formen sichtbar zu machen. Dieses Tool befindet sich unterhalb des Bilds neben den Steuerelementen zum Speichern, Zoomen und Schwenken.

Ergebnisse

Alle Geometrien in der Ergebnismenge werden im Bereich **Ergebnisse** als ein Bild angezeigt.

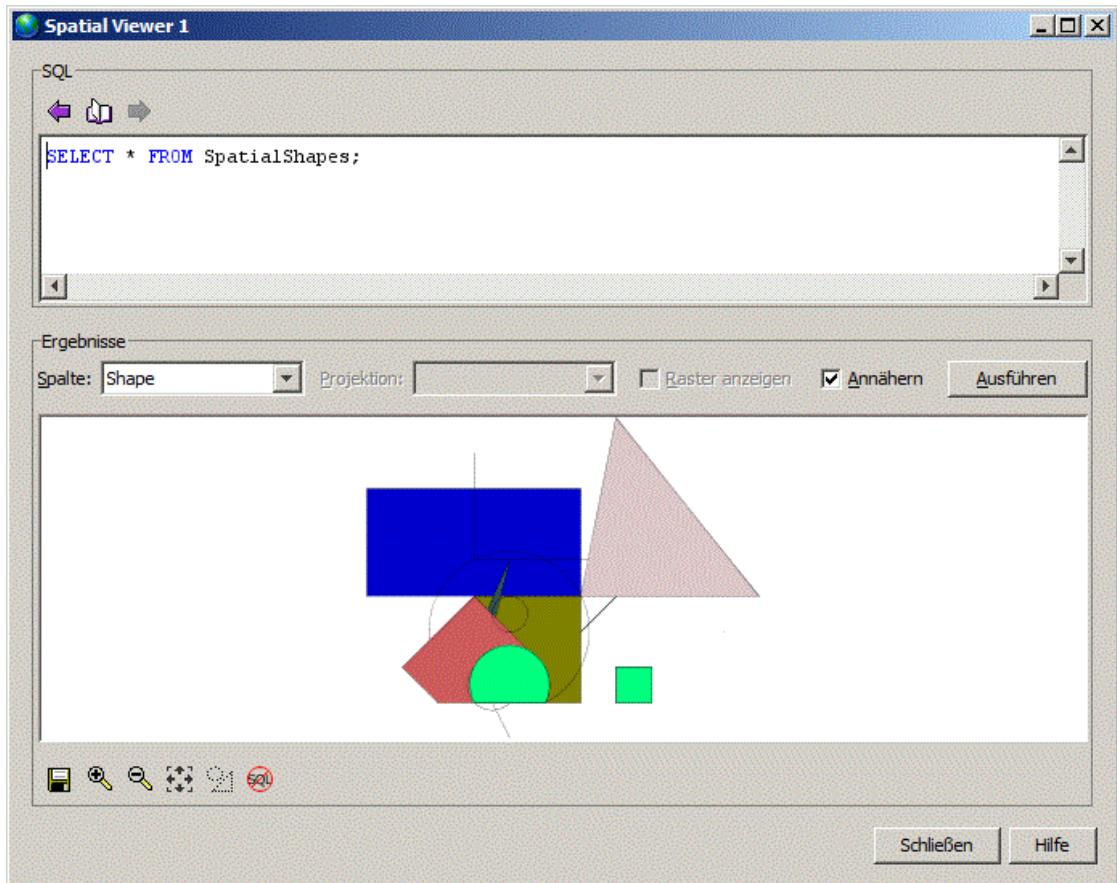
Nächste Schritte

Keine.

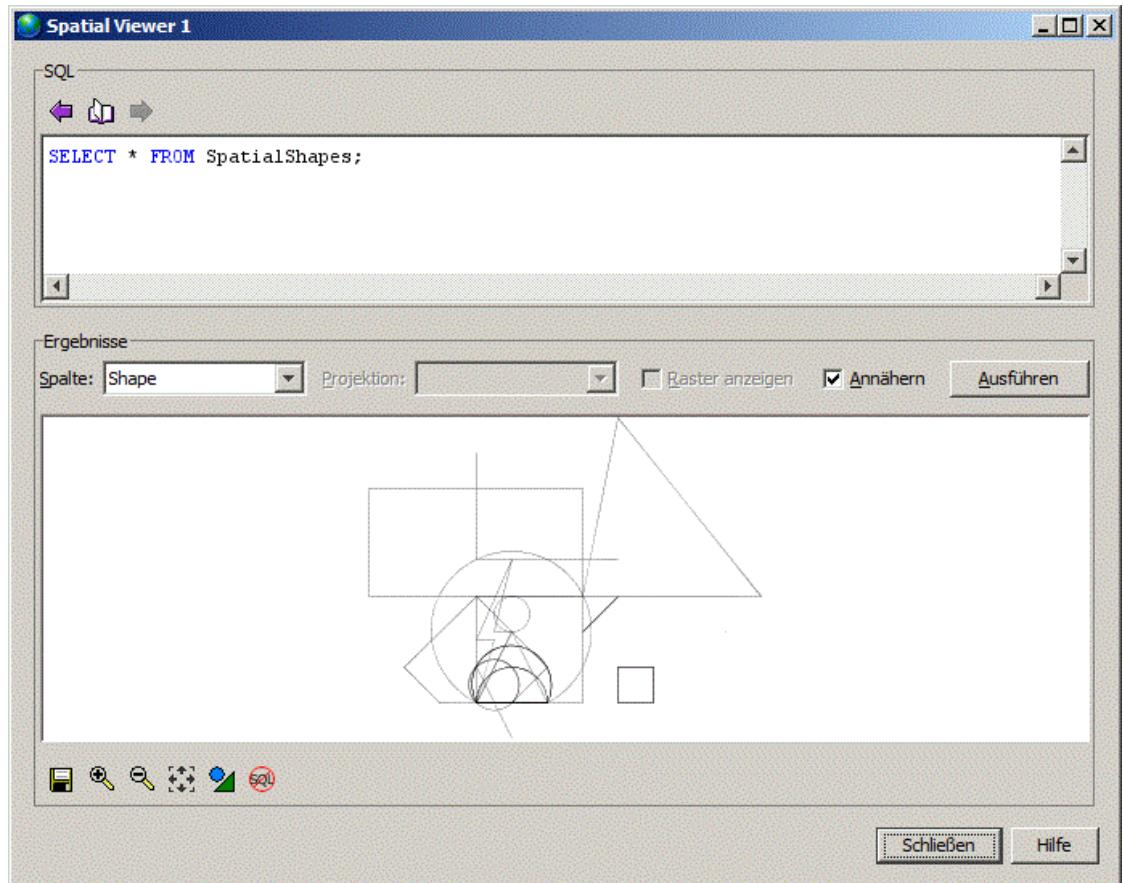
Beispiel

1. Verbinden mit der Beispieldatenbank über Interactive SQL.
2. Klicken Sie auf **Extras » Spatial Viewer**.
3. Führen Sie im **Spatial Viewer** die folgende Abfrage im Fensterausschnitt **SQL** aus:

```
SELECT * FROM GROUPO.SpatialShapes;
```



4. Dieses Beispiel zeigt, wie Umriss im Bild mit dem Tool **Polygon-Umriss zeichnen** angezeigt werden:



Laden von räumlichen Daten aus einer WKT-Datei (Well Known Text)

Sie können zum Hinzufügen räumlicher Daten zu einer Tabelle eine WKT-Datei (Well Known Text) verwenden. Diese enthält Text, mit dessen Hilfe räumliche Daten in eine Datenbank geladen und als Geometrie dargestellt werden können.

Voraussetzungen

Welche Privilegien zum Laden der Daten erforderlich sind, hängt von der Serveroption -gl ab. Wenn -gl auf ALL gesetzt ist, muss eine der folgenden Bedingungen erfüllt sein:

- Sie sind der Eigentümer der Tabelle
- Sie haben das LOAD-Privileg für die Tabelle
- Sie haben das LOAD ANY TABLE-Systemprivileg
- Sie haben das ALTER ANY TABLE-Systemprivileg

Wenn die Option -gl auf DBA gesetzt ist, müssen Sie das LOAD ANY TABLE-Systemprivileg oder das ALTER ANY TABLE-Systemprivileg haben.

Wenn -gl auf NONE gesetzt ist, wird LOAD TABLE nicht zugelassen.

Beim Laden aus einer Datei von einem Clientcomputer gilt Folgendes:

- Das READ CLIENT FILE-Privileg ist ebenfalls erforderlich.
- Leseprivilegien sind in dem Verzeichnis erforderlich, aus dem gelesen werden soll.
- Die Datenbankoption allow_read_client_file muss aktiviert sein.
- Die gesicherte Funktion read_client_file muss aktiviert sein.

Laden von räumlichen Daten aus einer WKT-Datei

1. Erstellen Sie eine Datei mit räumlichen Daten im WKT-Format, die Sie in die Datenbank laden können.

Die Datei kann jedes beliebige Format aufweisen, das durch die LOAD TABLE-Anweisung unterstützt wird.

2. Stellen Sie in Interactive SQL eine Verbindung mit Ihrer Datenbank her.
3. Erstellen Sie eine Tabelle und laden Sie die Daten aus der Datei mit einer Anweisung ähnlich der folgenden:

```
DROP TABLE IF EXISTS SA_WKT;  
CREATE TABLE SA_WKT (  
    description CHAR(24),  
    sample_geometry ST_Geometry(SRID=1000004326)  
);  
  
LOAD TABLE SA_WKT FROM 'C:\\Documents and Settings\\All Users\\Documents\\  
\\SQL Anywhere 16\\Samples\\wktgeometries.csv' DELIMITED BY ',';
```

Die Daten werden in die Tabelle eingelesen.

Ergebnisse

Die räumlichen Daten werden erfolgreich aus der WKT-Datei geladen.

Nächste Schritte

Sie können die Daten in Interactive SQL mit dem **Spatial Viewer** anzeigen.

Beispiel

1. Speichern Sie den folgenden Text in einer Textdatei namens *wktgeometries.csv*.

Der folgende Text enthält eine Gruppe von in WKT festgelegten Geometrien.

```

head,"CircularString(1.1 1.9, 1.1 2.5, 1.1 1.9)"
left iris,"Point(0.96 2.32)"
right iris,"Point(1.24 2.32)"
left eye,"MultiCurve(CircularString(0.9 2.32, 0.95 2.3, 1.0
2.32),CircularString(0.9 2.32, 0.95 2.34, 1.0 2.32))"
right eye,"MultiCurve(CircularString(1.2 2.32, 1.25 2.3, 1.3
2.32),CircularString(1.2 2.32, 1.25 2.34, 1.3 2.32))"
nose,"CircularString(1.1 2.16, 1.1 2.24, 1.1 2.16)"
mouth,"CircularString(0.9 2.10, 1.1 2.00, 1.3 2.10)"
hair,"MultiCurve(CircularString(1.1 2.5, 1.0 2.48, 0.8
2.4),CircularString(1.1 2.5, 1.0 2.52, 0.7 2.5),CircularString(1.1 2.5,
1.0 2.56, 0.9 2.6),CircularString(1.1 2.5, 1.05 2.57, 1.0 2.6))"
neck,"LineString(1.1 1.9, 1.1 1.8)"
clothes and box,"MultiSurface(((1.6 1.9, 1.9 1.9, 1.9 2.2, 1.6 2.2, 1.6
1.9)),((1.1 1.8, 0.7 1.2, 1.5 1.2, 1.1 1.8)))"
holes in box,"MultiPoint((1.65 1.95),(1.75 1.95),(1.85 1.95),(1.65 2.05),
(1.75 2.05),(1.85 2.05),(1.65 2.15),(1.75 2.15),(1.85 2.15))"
arms and legs,"MultiLineString((0.9 1.2, 0.9 0.8),(1.3 1.2, 1.3 0.8),
(0.97 1.6, 1.6 1.9),(1.23 1.6, 1.7 1.9))"
left cart wheel,"CircularString(2.05 0.8, 2.05 0.9, 2.05 0.8)"
right cart wheel,"CircularString(2.95 0.8, 2.95 0.9, 2.95 0.8)"
cart body,"Polygon((1.9 0.9, 1.9 1.0, 3.1 1.0, 3.1 0.9, 1.9 0.9))"
angular shapes on cart,"MultiPolygon(((2.18 1.0, 2.1 1.2, 2.3 1.4, 2.5
1.2, 2.35 1.0, 2.18 1.0)),((2.3 1.4, 2.57 1.6, 2.7 1.3, 2.3 1.4)))"
round shape on cart,"CurvePolygon(CompoundCurve(CircularString(2.6 1.0,
2.7 1.3, 2.8 1.0),(2.8 1.0, 2.6 1.0)))"
cart handle,"GeometryCollection(MultiCurve((2.0 1.0, 2.1
1.0),CircularString(2.0 1.0, 1.98 1.1, 1.9 1.2),CircularString(2.1 1.0,
2.08 1.1, 2.0 1.2),(1.9 1.2, 1.85 1.3),(2.0 1.2, 1.9 1.35),(1.85 1.3,
1.9 1.35)),CircularString(1.85 1.3, 1.835 1.29, 1.825
1.315),CircularString(1.9 1.35, 1.895 1.38, 1.88 1.365),LineString(1.825
1.315, 1.88 1.365))"

```

2. Stellen Sie in Interactive SQL eine Verbindung mit der Beispieldatenbank (*demo.db*) her.
3. Erstellen Sie eine Tabelle namens SA_WKT und laden Sie die Daten aus der Datei *wktgeometries.csv* in die Tabelle. Ersetzen Sie den Pfad zur *.csv*-Datei durch den Pfad, in dem Sie die Datei gespeichert haben:

```

DROP TABLE IF EXISTS SA_WKT;
CREATE TABLE SA_WKT (
    description CHAR(24),
    sample_geometry ST_Geometry(SRID=1000004326)
);

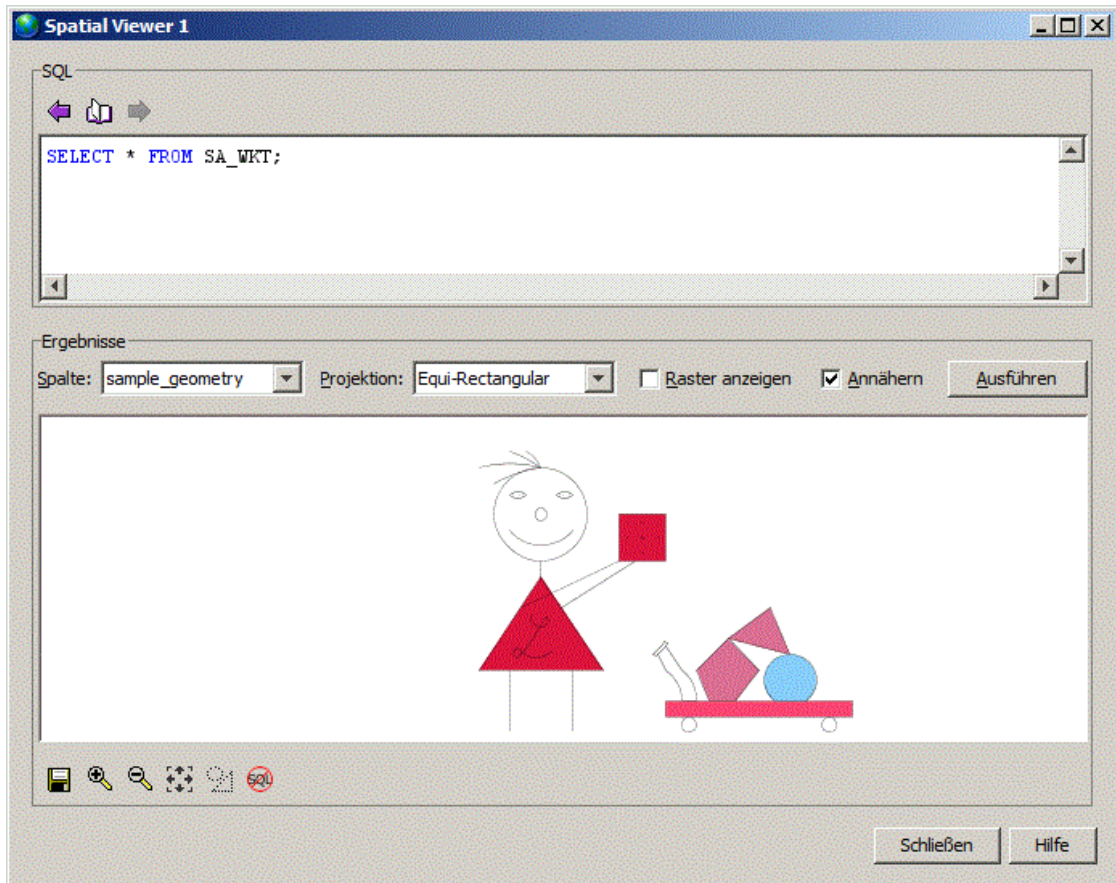
LOAD TABLE SA_WKT FROM 'C:\\Documents and Settings\\All Users\\Documents\\
\\SQL Anywhere 16\\Samples\\wktgeometries.csv' DELIMITED BY ',';

```

Die Daten werden in die Tabelle eingelesen.

4. Klicken Sie in Interactive SQL auf **Extras » Spatial Viewer**.
5. Führen Sie im **Spatial Viewer** die folgende Anweisung aus, um die Geometrien anzuzeigen:

```
SELECT * FROM SA_WKT;
```



6. Ihre Daten können mehrere Spalten räumlicher Daten enthalten. Sie können eine Datei mit WKT-Daten erstellen, die jeweils einen der verschiedenen unterstützten räumlichen Datentypen enthält, gespeichert in einzelnen Spalten.

Kopieren Sie den folgenden Code in Ihren Texteditor und speichern Sie die Datei unter dem Namen *wktgeometries2.csv*:

```
"Point(0 0)",,,,,,,,,,
,"LineString(0 0, 1 1)",,,,,,,,,
,"CircularString(0 0, 1 1, 0 0)",,,,,,,,,
,"CompoundCurve(CircularString(0 0, 1 1, 1 0),(1 0, 0 1))",,,,,,,,,
,"CompoundCurve(CircularString(0 0, 1 1, 1 0),(1 0, 0 1),(0 1, 0
0))",,,,,,,,,
,"Polygon((-1 0, 1 0, 2 1, 0 3, -2 1, -1 0))",,,,,,,,,
,"CurvePolygon(CompoundCurve(CircularString(0 0, 1 1, 1 0),(1 0, 0
0))",,,,,,,,,
,"CurvePolygon(CompoundCurve(CircularString(0 0, 2 1, 2 0),(2 0, 0
0))",,,,,,,,,
,"MultiPoint((2 0),(0 0),(3 0),(1 0))",,,,,,
,"MultiPolygon(((4 0, 4 1, 5 1, 5 0, 4 0)),((-1 0, 1 0, 2 1, 0
3, -2 1, -1 0))",,,,,,
,"MultiSurface(((4 0, 4 1, 5 1, 5 0, 4
0)),CurvePolygon(CompoundCurve(CircularString(0 0, 2 1, 2 0),(2 0, 0
0)))",,,,,,
```

```

,,,,,,,,,"MultiLineString((2 0, 0 0),(3 0, 1 0),(-2 1, 0 4))",,,
,,,,,,,,,"MultiCurve((3 2, 4 3),CircularString(0 0, 1 1, 0 0))",,
,,,,,,,,,"GeometryCollection(MultiPoint((2 0),(0 0),(3 0),(1
0)),MultiSurface(((4 0, 4 1, 5 1, 5 0, 4
0)),CurvePolygon(CompoundCurve(CircularString(0 0, 2 1, 2 0),(2 0, 0
0))),MultiCurve((3 2, 4 3),CircularString(0 0, 1 1, 0 0)))",
,,,,,,,,,"GeometryCollection(Point(0
0),CompoundCurve(CircularString(0 0, 1 1, 1 0),(1 0, 0 1),(0 1, 0
0)),CurvePolygon(CompoundCurve(CircularString(0 0, 2 1, 2 0),(2 0, 0
0))),MultiPoint((2 0),(0 0),(3 0),(1 0)),MultiSurface(((4 0, 4 1, 5 1, 5
0, 4 0)),CurvePolygon(CompoundCurve(CircularString(0 0, 2 1, 2 0),(2 0,
0 0))),MultiCurve((3 2, 4 3),CircularString(0 0, 1 1, 0 0)))"

```

- Erstellen Sie eine Tabelle namens SA_WKT2 und führen Sie die folgenden Anweisungen aus, um die Daten aus der Datei *wktgeometries2.csv* in die Tabelle zu laden. Achten Sie darauf, dass der Pfad zur *.csv*-Datei durch den Pfad ersetzt wird, in dem Sie die Datei gespeichert haben:

```

DROP TABLE IF EXISTS SA_WKT2;
CREATE TABLE SA_WKT2 (
    point          ST_Point,
    line           ST_LineString,
    circle         ST_CircularString,
    compoundcurve  ST_CompoundCurve,
    curve          ST_Curve,
    polygon1       ST_Polygon,
    curvepolygon   ST_CurvePolygon,
    surface        ST_Surface,
    multipoint     ST_MultiPoint,
    multipolygon   ST_MultiPolygon,
    multisurface   ST_MultiSurface,
    multiline      ST_MultiLineString,
    multicurve     ST_MultiCurve,
    geomcollection ST_GeomCollection,
    geometry       ST_Geometry
);

LOAD TABLE SA_WKT2 FROM 'C:\\Documents and Settings\\All Users\\Documents
\\SQL Anywhere 16\\Samples\\wktgeometries2.csv' DELIMITED BY ',';

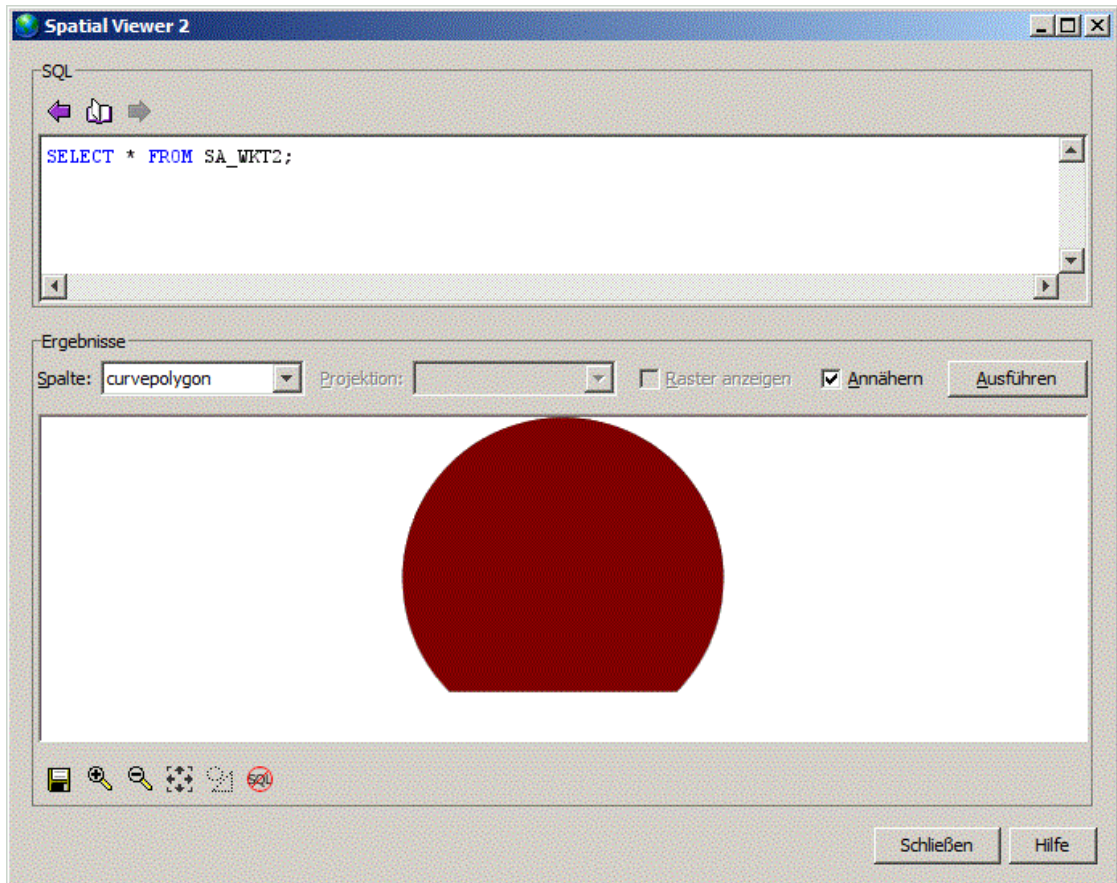
```

Die Daten werden in die Tabelle eingelesen.

- Führen Sie im **Spatial Viewer** die folgende Anweisung aus, um die Geometrien anzuzeigen:

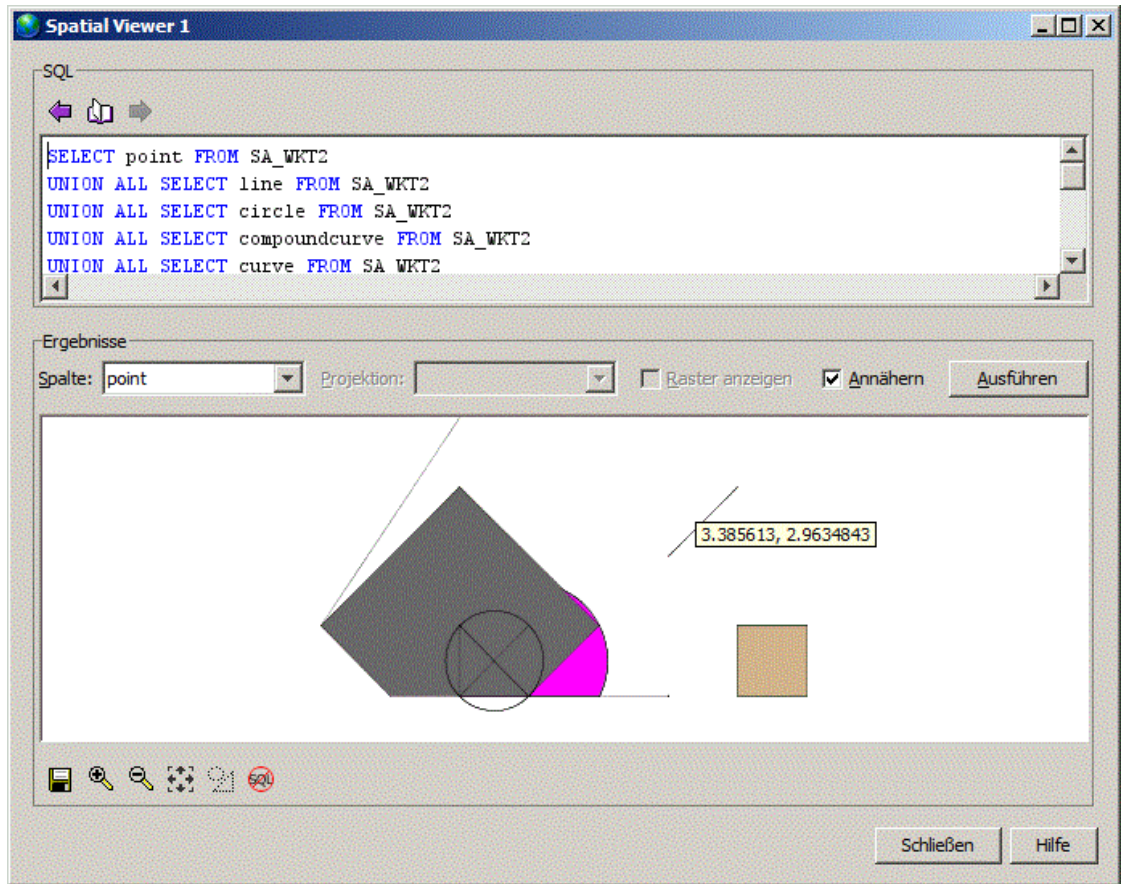
```
SELECT * FROM SA_WKT2;
```

Sie können jeweils nur eine Datenspalte sehen. Um die Geometrien der anderen Spalten anzuzeigen, müssen Sie im Bereich **Ergebnisse** die Dropdown-Liste **Spalte** verwenden. Folgende Ansicht zeigt die Geometrie in der Spalte *curvepolygon*:



9. Um die Geometrien aller Spalten gleichzeitig anzuzeigen, können Sie für jede Spalte eine SELECT-Anweisung ausführen und alle Ergebnisse mit UNION ALL wie folgt vereinigen:

```
SELECT point FROM SA_WKT2
UNION ALL SELECT line FROM SA_WKT2
UNION ALL SELECT circle FROM SA_WKT2
UNION ALL SELECT compoundcurve FROM SA_WKT2
UNION ALL SELECT curve FROM SA_WKT2
UNION ALL SELECT polygon1 FROM SA_WKT2
UNION ALL SELECT curvepolygon FROM SA_WKT2
UNION ALL SELECT surface FROM SA_WKT2
UNION ALL SELECT multipoint FROM SA_WKT2
UNION ALL SELECT multipolygon FROM SA_WKT2
UNION ALL SELECT multisurface FROM SA_WKT2
UNION ALL SELECT multiline FROM SA_WKT2
UNION ALL SELECT multicurve FROM SA_WKT2
UNION ALL SELECT geomcollection FROM SA_WKT2
UNION ALL SELECT geometry FROM SA_WKT2
```



Siehe auch

- „Anzeigen von räumlichen Daten als Bilder (Spatial Viewer)“ auf Seite 28
- „LOAD TABLE-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Maßeinheiten erstellen

Mehrere Maßeinheiten werden mit der Software installiert. Wenn die installierten Maßeinheiten nicht für Ihre Daten geeignet sind, können Sie eigene erstellen.

Voraussetzungen

Sie müssen das `MANAGE ANY SPATIAL OBJECT`-Systemprivileg oder das `CREATE ANY OBJECT`-Systemprivileg haben.

Erstellen einer Maßeinheit (Sybase Central)

1. Stellen Sie eine Verbindung mit der Datenbank her.

2. Klicken Sie im linken Fensterausschnitt auf **Räumliche Bezugssysteme**.
3. Klicken Sie im rechten Fensterausschnitt auf die Registerkarte **Maßeinheit**.
4. Rechtsklicken Sie auf die Registerkarte und klicken Sie auf **Neu » Maßeinheit**.
5. Klicken Sie **Benutzerdefinierte Maßeinheit erstellen** und anschließend auf **Weiter**.
6. Geben Sie im Feld **Wie lautet der Name der neuen Maßeinheit?** einen Namen ein und klicken Sie dann auf **Weiter**.
7. Wählen Sie **Lineare Maßeinheit** im Feld **Welchen Maßeinheitstyp möchten Sie erstellen?**.
8. Befolgen Sie die Anweisungen des Assistenten **Maßeinheit erstellen**.
9. Klicken Sie auf **Fertig stellen**.

Ergebnisse

Eine Maßeinheit wird erstellt.

Nächste Schritte

Sie können Benutzern Privilegien erteilen, damit diese räumliche Bezugssysteme und Maßeinheiten erstellen, ändern oder löschen können.

Siehe auch

- „CREATE SPATIAL UNIT OF MEASURE-Anweisung“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „Maßeinheiten“ auf Seite 6

Räumliche Bezugssysteme erstellen

Mithilfe des **Assistenten zum Erstellen eines räumlichen Bezugssystems** können Sie ein räumliches Bezugssystem (Spatial Reference System, kurz SRS) erstellen und dabei ein vorhandenes als Vorlage verwenden, auf dem Ihre Einstellungen basieren.

Voraussetzungen

Sie müssen das **MANAGE ANY SPATIAL OBJECT**-Systemprivileg oder das **CREATE ANY OBJECT**-Systemprivileg haben.

Die Maßeinheit, die dem räumlichen Bezugssystem zugeordnet werden soll, muss bereits vorhanden sein.

Kontext und Bemerkungen

Wenn Sie ein räumliches Bezugssystem erstellen, verwenden Sie ein vorhandenes Bezugssystem als Vorlage, auf dem Ihre Einstellungen basieren. Sie sollten daher ein räumliches Bezugssystem verwenden,

das dem zu erstellenden räumlichen Bezugssystem ähnlich ist. Anschließend können Sie die Einstellungen bearbeiten.

Erstellen eines räumlichen Bezugssystems (Sybase Central)

1. Stellen Sie in Sybase Central eine Verbindung mit der Datenbank her.
2. Klicken Sie im linken Fensterausschnitt mit der rechten Maustaste auf **Räumliche Bezugssysteme** » **Neu** » **Räumliches Bezugssystem**.
3. Wählen Sie **Auswahl aus einer Liste aller vordefinierten räumlichen Bezugssysteme** und klicken Sie dann auf **Weiter**.
4. Befolgen Sie die Anweisungen des Assistenten.
5. Wenn Sie ein räumliches Bezugssystem basierend auf einem vorhandenen räumlichen Bezugssystem erstellen, setzen Sie den SRID-Wert auf 1000000000 plus dem Well-Known-Wert.

Hinweis

Prüfen Sie bei der Zuweisung einer SRID die Empfehlungen für zu vermeidende Nummernbereiche. Diese Empfehlungen befinden sich in der Klausel IDENTIFIED der Anweisung CREATE SPATIAL REFERENCE SYSTEM.

6. Die Definition für das räumliche Bezugssystem wird angezeigt.
7. Wenn die Definition für das räumliche Bezugssystem richtig ist, klicken Sie auf **Fertig stellen**.

Ergebnisse

Das neue räumliche Bezugssystem wird der Datenbank hinzugefügt.

Nächste Schritte

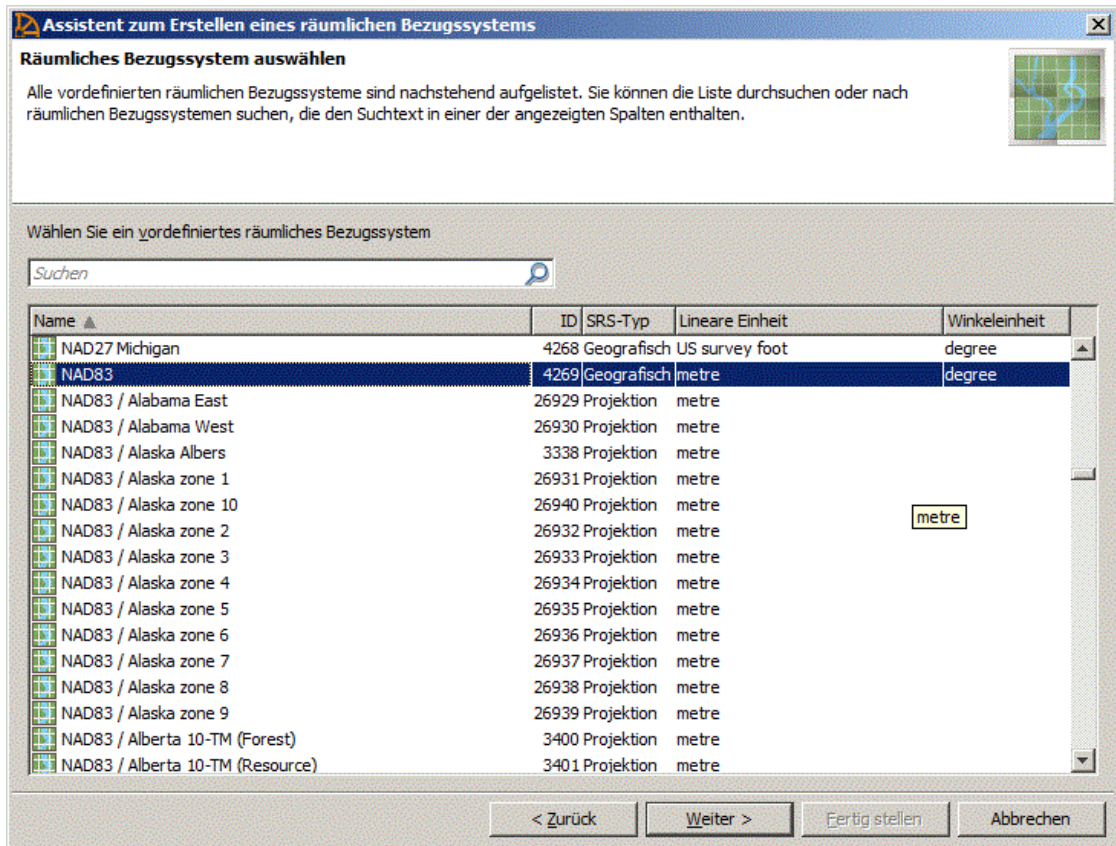
Sie können die Tabellen festlegen, die das räumliche Bezugssystem verwenden.

Beispiel

In diesem Beispiel wird ein räumliches Bezugssystem auf der Basis eines vorhandenen erstellt.

1. Stellen Sie eine Verbindung mit der Datenbank her.
2. Klicken Sie im linken Fensterausschnitt mit der rechten Maustaste auf **Räumliche Bezugssysteme** » **Neu** » **Räumliches Bezugssystem**.
3. Wählen Sie **Auswahl aus einer Liste aller vordefinierten räumlichen Bezugssysteme** und klicken Sie dann auf **Weiter**.

Das Fenster **Räumliches Bezugssystem auswählen** wird geöffnet.



- Um ein räumliches Bezugssystem auf der Basis des räumlichen Bezugssystems NAD83 zu erstellen, geben Sie **NAD83** ein. Während der Eingabe eines Namens oder einer ID im Feld **Wählen Sie ein vordefiniertes räumliches Bezugssystem** ändert sich die Liste der räumlichen Bezugssysteme, sodass das räumliche Bezugssystem angezeigt wird, das Sie als Vorlage verwenden möchten.
- Klicken Sie auf **NAD83** und klicken Sie dann auf **Weiter**.

Das Fenster **Linieninterpretation auswählen** wird geöffnet.

Assistent zum Erstellen eines räumlichen Bezugssystems

Linieninterpretation auswählen

SQL Anywhere unterstützt die gewölbte und die planare Darstellung der Erde. Die Wahl der Darstellung der Erde bestimmt, wie das räumliche Bezugssystem die Linien zwischen den Punkten interpretiert.

Welche Linieninterpretation möchten Sie für dieses räumliche Bezugssystem verwenden?

☒ **Gewölbte Erddarstellung**

Die Erde wird als Ellipsoid mit einer zweidimensionalen Projektion des dreidimensionalen Globusmodells dargestellt. Die Linien zwischen den Punkten werden als große elliptische Bögen interpretiert. Mit zwei gegebenen Punkten auf der Erdoberfläche wird eine Ebene ausgewählt, die die beiden Punkte und den Mittelpunkt der Erde schneidet. Diese Ebene schneidet die Erde, und die Linie zwischen den beiden Punkten ist die kürzeste Distanz entlang dieser Schnittlinie. Mithilfe dieser Linieninterpretation ist es nicht möglich, die geometrischen Daten ohne Verzerrung darzustellen. Je nach dem Standort eines Objekts auf der Erde kann die Fläche, die Form, die Distanz oder die Richtung Verzerrungen unterliegen. Diese Linieninterpretation wird nur für geografische räumliche Bezugssysteme unterstützt.

☐ **Planare Erddarstellung**

Die Erde wird als plane, zweidimensionale Karte dargestellt. Die Linien zwischen den Punkten werden in einer winkeltreuen (equirektangularen) Projektion interpretiert. Diese Linieninterpretation wird für alle räumlichen Bezugssysteme unterstützt und ist die einzige Wahl für nichtgeografische räumliche Bezugssysteme.

Die Standardwerte für den Namen und die SRID eines räumlichen Bezugssystems beruhen auf den vordefinierten Werten und der Auswahl der Linieninterpretation. Für eine planare Interpretation eines geografischen räumlichen Bezugssystems entspricht der SRID-Standardwert 1.000.000.000 plus dem vordefinierten Wert. Sie können diese Standardwerte gegebenenfalls überschreiben.

Name:

ID des räumlichen Bezugssystems:

< Zurück Weiter > Fertig stellen Abbrechen

6. Klicken Sie auf **Gewölbte Erddarstellung**, um diese Linieninterpretation auszuwählen.
7. Geben Sie **NAD83custom** im Feld **Name** ein.
8. Wenn Sie ein räumliches Bezugssystem basierend auf einem vorhandenen räumlichen Bezugssystem erstellen, setzen Sie den SRID-Wert auf 1000000000 plus dem Well-Known-Wert. Ändern Sie z. B. den Wert im Feld **ID des räumlichen Bezugssystems** von 4269 auf **100004269**.

Hinweis

Prüfen Sie bei der Zuweisung einer SRID die Empfehlungen für zu vermeidende Nummernbereiche. Diese Empfehlungen befinden sich in der Klausel IDENTIFIED der Anweisung CREATE SPATIAL REFERENCE SYSTEM.

9. Klicken Sie auf **Weiter**.

Das Fenster **Einen Kommentar angeben** wird geöffnet.

10. Sie können optional eine Beschreibung für das räumliche Bezugssystem eingeben. Klicken Sie dann auf **Weiter**.

11. Klicken Sie auf **Fertig stellen**.

Die Definition für das räumliche Bezugssystem wird angezeigt.

12. Klicken Sie auf **Fertig stellen**.

Siehe auch

- „Maßeinheiten“ auf Seite 6
- „CREATE SPATIAL UNIT OF MEASURE-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- IDENTIFIED BY-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „CREATE SPATIAL REFERENCE SYSTEM-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- „Räumliche Bezugssysteme (SRS) und räumliche Referenz-IDs (SRID)“ auf Seite 2

Erweiterte Themen zu räumlichen Daten

Dieser Abschnitt enthält erweiterte räumliche Themen.

Funktionsweise der planaren und gewölbten Erddarstellung

SQL Anywhere unterstützt sowohl die plane als auch die gewölbte Erddarstellung. Bezugssysteme mit **planarer Erddarstellung** projizieren alle oder einen Teil der Oberfläche der Erde auf eine flache, zweidimensionale Ebene (plan oder planar) und verwenden eine einfache euklidische 2D-Geometrie. Linien zwischen Punkten sind gerade (ausgenommen Kreisfolgen) und Geometrien können sich nicht über die Kante erstrecken (die Datumsgrenze überqueren).

Räumliche Bezugssysteme mit **gewölbter Erddarstellung** verwenden ein Ellipsoid zur Darstellung. Punkte werden dem Ellipsoid für Berechnungen zugeordnet, alle Linien nehmen den kürzesten Weg und sind in Richtung des Pols gewölbt. Geometrien können die Datumsgrenze überqueren.

Die plane und die gewölbte Erddarstellung unterliegen jeweils bestimmten Einschränkungen. Es gibt keine einzige ideale kartografische Projektion, die alle Funktionen der Erde darstellt, und abhängig vom Standort eines Objekts auf der Erde können Verzerrungen seinen Bereich, seine Form, seinen Abstand oder seine Richtung beeinflussen.

Einschränkungen von räumlichen Bezugssystemen mit gewölbter Erddarstellung

Bei der Verwendung eines räumlichen Bezugssystems mit gewölbter Erddarstellung, wie etwa WGS 84, sind zahlreiche Funktionen nicht verfügbar. Das Berechnen der Entfernung ist beispielsweise auf Punkte und Punktesammlungen beschränkt.

Einige Prädikate und Festlegungsvorgänge sind ebenfalls nicht verfügbar.

Kreisbogenfolgen sind in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht zulässig.

Berechnungen in räumlichen Bezugssystemen mit gewölbter Erddarstellung sind kostspieliger als eine entsprechende Berechnung in einem räumlichen Bezugssystem mit planer Erddarstellung.

Einschränkungen von räumlichen Bezugssystemen mit planer Erddarstellung

Ein räumliches Bezugssystem mit planer Erddarstellung ist ein planares räumliches Bezugssystem, für das eine Projektion definiert ist. Die **Projektion** löst Verzerrungsprobleme, die auftreten, wenn ein räumliches Bezugssystem mit planer Erddarstellung verwendet wird, um gewölbte Erddaten zu verarbeiten. Die Verzerrung, die auftritt, wenn keine Projektion verwendet wird, ist am Beispiel der folgenden beiden Bilder zu erkennen, die dieselbe Gruppe von Zipcode-Regionen in Massachusetts zeigen. Das erste Bild zeigt die Daten in einem SRID 3586, einem projizierten planaren räumlichen Bezugssystem speziell für die Daten aus Massachusetts. Das zweite Bild zeigt die Daten in einem planaren räumlichen Bezugssystem ohne Projektion (SRID 1000004326). Die Verzerrung ist auf dem zweiten Bild in Form von übergroßen Abständen, Längen und Bereichen zu sehen, wodurch das Bild horizontal gedehnt erscheint.



Wenngleich mehr Berechnungen mit räumlichen Bezugssystemen mit planer Erddarstellung möglich sind, sind die Berechnungen aufgrund des Projektionseffekts nur für Bereiche mit beschränkter Größe präzise.

Bei der Arbeit mit Abständen von wenigen hundert Kilometern können Daten der gewölbten Erddarstellung in ein räumliches Bezugssystem mit planer Erddarstellung projiziert werden, um präzise

Abstandsberechnungen zu erhalten. Mit der ST_Transform-Methode können Sie die Daten in ein räumliches planares Bezugssystem projizieren.

Siehe auch

- [ST_Transform-Methode für den ST_Geometry-Datentyp auf Seite 237](#)

Wie sich Ausrichten am Raster und Toleranz auf räumliche Berechnungen auswirken

Am Raster ausrichten ist die Maßnahme der Positionierung der Punkte in einer Geometrie, um sie an Schnittpunkten in einem Raster auszurichten. Beim Ausrichten eines Punkt am **Raster** werden die X- und Y-Werte möglicherweise geringfügig verschoben - ähnlich der Rundung. In Bezug auf räumliche Daten ist ein Raster eine Struktur aus Linien, die über eine zweidimensionale Darstellung eines räumlichen Bezugssystems gelegt wird. SQL Anywhere verwendet ein quadratisches Raster.

Einfaches Beispiel für die Ausrichtung am Raster: Wenn die Rastergröße 0,2 ist, wird die Linie von Point(14.2321, 28.3262) zu Point(15.3721, 27.1128) an der Linie von Point(14.2, 28.4) zu Point(15.4, 27.2) ausgerichtet. Die Rastergröße ist normalerweise viel kleiner als in diesem einfachen Beispiel, sodass der Präzisionsverlust wesentlich geringer ist.

Standardmäßig setzt SQL Anywhere automatisch die Rastergröße so fest, dass 12 signifikante Stellen für jeden Punkt innerhalb der X- und Y-Grenzen eines räumlichen Bezugssystems gespeichert werden können. Beispiel: Wenn der Bereich der X-Werte von -180 bis 180 und der Bereich der Y-Werte von -90 bis 90 reicht, legt der Datenbankserver die Rastergröße auf 1e-9 (0.000000001) fest. Das heißt, die Entfernung zwischen horizontalen und vertikalen Rasterlinien beträgt 1e-9. Die Schnittpunkte der Rasterlinie stellen alle Punkte dar, die im räumlichen Bezugssystem dargestellt werden können. Wenn eine Geometrie erstellt oder geladen wird, werden die X- und Y-Koordinaten jedes Punkts an den nächst gelegenen Punkten im Raster ausgerichtet.

Toleranz definiert die Entfernung, innerhalb der zwei Punkte oder Teile von Geometrien als gleichwertig angesehen werden. Dies ist vergleichbar mit der Darstellung von Geometrien, deren Punkte und Linien mit einem dicken Marker nachgezogen werden, wobei die Dicke des Markers der Toleranz entspricht. Alle Teile, die sich bei der Darstellung mit diesem dicken Marker berühren, werden innerhalb der Toleranz als gleich betrachtet. Wenn zwei Punkte genau um den Toleranzwert voneinander getrennt sind, werden sie als nicht als gleichwertig innerhalb der Toleranz betrachtet.

Einfaches Beispiel für Toleranz: Bei einer Toleranz von 0,5 werden Point(14.2, 28.4) und Point(14.4, 28.2) als gleichwertig angesehen. Dies liegt daran, dass die Entfernung zwischen den beiden Punkten (in denselben Einheiten wie X und Y) etwa 0,283 beträgt und somit kleiner ist als die Toleranz. Die Toleranz ist jedoch normalerweise viel kleiner als in diesem einfachen Beispiel.

Die Toleranz kann dazu führen, dass extrem kleine Geometrien ungültig werden. Linien, die kürzer sind als der Toleranzwert, sind ungültig (da die Punkte einander entsprechen), und Polygone, deren Punkte alle innerhalb des Toleranzwerts liegen, sind ebenfalls ungültig.

Die Ausrichtung am Raster und die Toleranz werden im räumlichen Bezugssystem festgelegt und verwenden immer die gleichen Einheiten wie die X- und Y-Koordinaten (oder Längengrad und

Breitengrad). Mithilfe der Anpassung an das Raster und der Toleranz können Probleme mit nicht exakten arithmetischen und ungenauen Daten gelöst werden. Allerdings sollten Sie berücksichtigen, wie sich ihr Verhalten auf die Ergebnisse von Operationen mit räumlichen Daten auswirken kann.

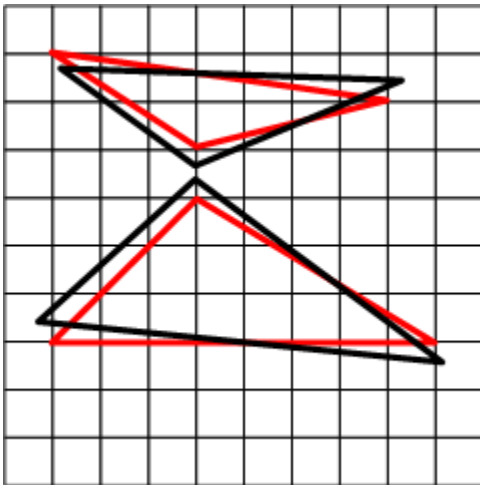
Hinweis

Bei planaren räumlichen Referenzsystemen wird ein Einstellen der Rastergröße auf 0 nicht empfohlen, weil es zu falschen Ergebnissen von räumlichen Vorgängen führen kann. Bei räumlichen Bezugssystemen mit gewölbter Erddarstellung muss die Rastergröße und die Toleranz auf 0 gesetzt sein. SQL Anywhere verwendet eine feste Rastergröße und Toleranz auf einer internen Projektion, wenn Vorgänge mit gewölbter Erddarstellung durchgeführt werden.

Die folgenden Beispiele veranschaulichen die Auswirkungen von Rastergrößen- und Toleranzeinstellungen auf räumliche Berechnungen.

Beispiel 1: Auswirkungen des Ausrichtens am Raster auf die Schnittpunktergebnisse

Zwei Dreiecke (schwarz dargestellt) werden in ein räumliches Bezugssystem geladen, bei dem die Toleranz auf Rastergröße festgelegt ist, und das Raster im Diagramm basiert auf der Rastergröße. Die roten Dreiecke repräsentieren die schwarzen Dreiecke, nachdem die Dreiecksscheitelpunkte am Raster ausgerichtet wurden. Beachten Sie, dass sich die ursprünglichen Dreiecke (schwarz) innerhalb der Toleranz voneinander befinden, während dies bei den ausgerichteten Dreiecken in Rot nicht der Fall ist. ST_Intersects gibt für diese beiden Geometrien 0 zurück. Wenn die Toleranz größer als die Rastergröße ist, würde ST_Intersects 1 für diese beiden Geometrien zurückgeben.

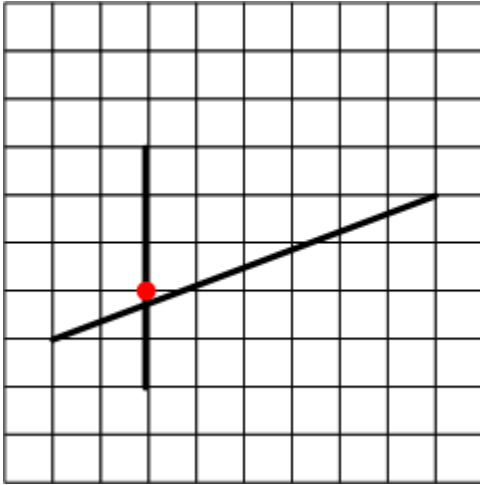


Beispiel 2: Toleranzauswirkungen auf Schnittpunktergebnisse

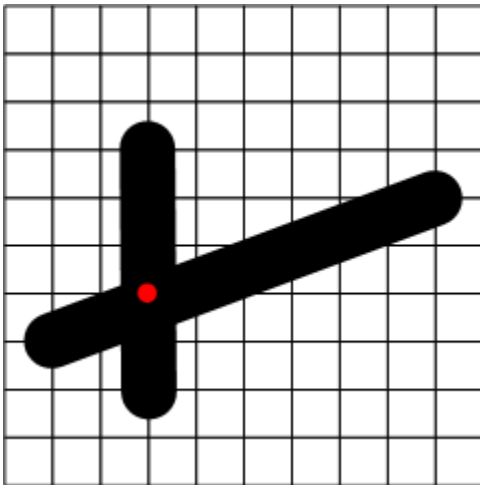
Im folgenden Beispiel befinden sich zwei Linien in einem räumlichen Bezugssystem, in dem die Toleranz auf 0 gesetzt ist. Der Schnittpunkt der zwei Linien ist am nächsten Scheitelpunkt im Raster ausgerichtet. Da die Toleranz auf 0 gesetzt ist, gibt ein Test, um zu ermitteln, ob der Schnittpunkt der zwei Linien die diagonale Linie schneidet, FALSE zurück.

In anderen Worten: Der folgende Ausdruck gibt 0 zurück, wenn die Toleranz 0 ist:

```
vertical_line.ST_Intersection( diagonal_line ).ST_Intersects( diagonal_line )
```

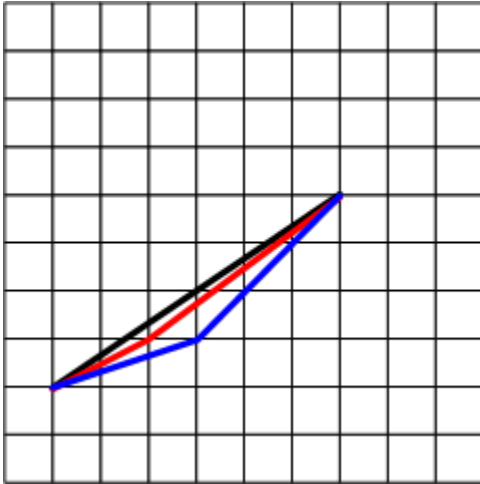


Das Festlegen der Toleranz auf die Rastergröße (der Standardwert) bewirkt jedoch, dass der Schnittpunkt innerhalb der dicken Diagonalen liegt. Daher würde ein Test, ob der Schnittpunkt die Diagonale innerhalb der Toleranzwerte schneidet, positiv ausgehen:



Beispiel 3: Toleranz und Transitivität

Wenn in räumlichen Berechnungen Toleranz verwendet wird, bleibt die Transitivität nicht unbedingt erhalten. Beispiel: Angenommen, Sie haben die folgenden drei Linien in einem räumlichen Bezugssystem, bei dem die Toleranz der Rastergröße entspricht:



Die ST_Equals-Methode betrachtet die schwarzen und die roten Linien Zeilen als gleichwertig innerhalb der Toleranz sowie die roten und die blauen Linien als gleichwertig innerhalb der Toleranz, aber die schwarze und die blaue Linie sind nicht gleichwertig innerhalb der Toleranz. ST_Equals ist nicht transitiv.

ST_OrderingEquals betrachtet jede dieser Linien als unterschiedlich und ST_OrderingEquals ist transitiv.

Beispiel 4: Auswirkungen der Raster- und Toleranzeinstellungen auf ungenaue Daten

Angenommen, Sie haben Daten in einem projizierten planaren räumlichen Bezugssystem, das innerhalb von 10 cm größtenteils genau ist, und innerhalb von 10 m immer genau ist. Es stehen drei Möglichkeiten zur Verfügung:

1. Sie verwenden den von SQL Anywhere ausgewählten Standardwert für Rastergröße und Toleranz, der normalerweise größer als die Präzision Ihrer Daten ist. Auch wenn dies eine maximale Präzision sicherstellt, führen Prädikate wie ST_Intersects, ST_Touches, und ST_Equals möglicherweise bei manchen Geometrien zu Ergebnissen, die sich von den erwarteten unterscheiden, abhängig von der Genauigkeit der Geometriewerte. Beispiel: Zwei nebeneinander liegende Polygone, die eine gemeinsame Kante haben, geben möglicherweise nicht TRUE für ST_Intersects zurück, wenn das linke Polygon Kantendaten ein paar Meter links vom rechten Polygon hat.
2. Sie legen die Rastergröße klein genug fest, um die größte Präzision Ihrer Daten darzustellen (in diesem Fall 10 cm), und zumindest vier Mal kleiner als die Toleranz. Außerdem sollte die Toleranz der Distanz entsprechen, innerhalb der Ihre Daten immer präzise sind (im vorliegenden Beispiel 10 m). Auf diese Weise werden die Daten ohne Verlust der Präzision gespeichert, während Prädikate das erwartete Ergebnis liefern, obwohl die Daten nur innerhalb von 10 m genau sind.
3. Sie legen die Rastergröße und die Toleranz auf die Präzision Ihrer Daten fest (in diesem Fall 10 m). Auf diese Weise werden Ihre Daten im Rahmen der Präzision Ihrer Daten ausgerichtet. Bei Daten, die präziser als 10 Meter sind, geht die zusätzliche Genauigkeit jedoch verloren.

In vielen, jedoch nicht in allen Fällen, liefern Prädikate die erwarteten Ergebnisse. Beispiel: Wenn sich zwei Punkte, die weniger als 10 cm voneinander entfernt sind, nahe der Mitte der Rasterschnittpunkte befinden, wird einer der Punkte auf der einen Seite und der andere Punkt auf der

anderen Seite ausgerichtet. Dies führt dazu, dass die beiden Punkte ca. 10 m voneinander entfernt sind. Aus diesem Grund wird es in diesem Fall nicht empfohlen, die Rastergröße und die Toleranz entsprechend der Genauigkeit Ihrer Daten festzulegen.

Siehe auch

- [SNAP TO GRID-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#)
- [TOLERANCE-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#)
- [ST_Equals-Methode für den ST_Geometry-Datentyp auf Seite 182](#)
- [ST_SnapToGrid-Methode für den ST_Geometry-Datentyp auf Seite 216](#)
- [ST_OrderingEquals-Methode für den ST_Geometry-Datentyp auf Seite 207](#)
- [„Unterstützte räumliche Prädikate“ auf Seite 10](#)
- [„CREATE SPATIAL REFERENCE SYSTEM-Anweisung“ \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#)
- [„ALTER SPATIAL REFERENCE SYSTEM-Anweisung“ \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#)

Auswirkungen der Interpolation auf räumliche Berechnungen

Interpolation ist der Prozess der Verwendung bekannter Punkte in einer Geometrie, um unbekannte Punkte anzunähern. Mehrere räumlichen Methoden und Prädikate verwenden Interpolation, wenn die Berechnungen Kreisbögen umfassen. Die Interpolation verwandelt einen Kreisbogen in eine Sequenz von geraden Linien. Beispiel: Eine Kreisbogenfolge, die einen Viertelbogen darstellt, wird möglicherweise als Linienfolge mit 11 Kontrollpunkten interpoliert.

Interpolationsbeispiel

1. Verbinden Sie sich in Interactive SQL mit der Beispieldatenbank und führen Sie die folgende Anweisung aus, um eine Variable namens arc zu erstellen, in der Sie eine Kreisbogenfolge speichern werden:

```
CREATE VARIABLE arc ST_CircularString;
```

2. Führen Sie die folgende Anweisung aus, um eine Kreisbogenfolge zu erstellen, und speichern Sie sie in der Variablen arc:

```
SET arc = NEW ST_CircularString( 'CircularString( -1 0, -0.707107  
0.707107, 0 1 )' );
```

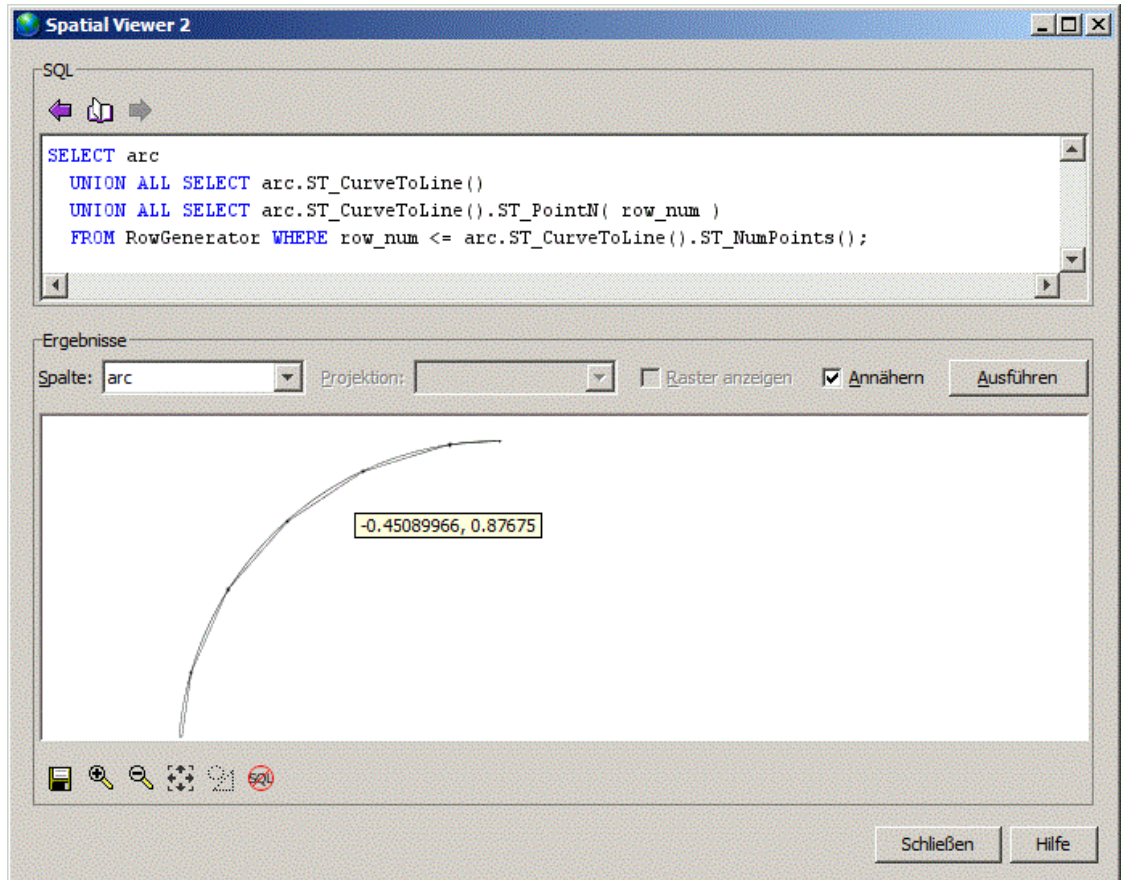
3. Führen Sie die folgende Anweisung aus, um vorübergehend die relative Toleranz mit der st_geometry_interpolation-Option auf 1 % zu setzen.

```
SET TEMPORARY OPTION st_geometry_interpolation = 'relative-tolerance-  
percent=1' ;
```

Das Festlegen der relativen Toleranz auf 1 % ist optional, aber es macht die Auswirkungen der Interpolation in diesem Beispiel deutlicher sichtbar.

4. Öffnen Sie den **Spatial Viewer** (indem Sie in Interactive SQL auf **Extras » Spatial Viewer** klicken) und führen Sie die folgende Abfrage aus, um die Kreisbogenfolge anzuzeigen:

```
SELECT arc
  UNION ALL SELECT arc.ST_CurveToLine()
  UNION ALL SELECT arc.ST_CurveToLine().ST_PointN( row_num )
  FROM RowGenerator WHERE row_num <= arc.ST_CurveToLine().ST_NumPoints();
```



Beachten Sie, dass der Bogen in eine Sequenz von Linienfolgen unterteilt ist. Da die Toleranz auf 1 % gesetzt wurde, wird jedes Liniensegment als Linie angezeigt, die sich im tatsächlichen Bogen beugt. Der maximale Abstand zwischen der interpolierten Linienfolge und dem tatsächlichen Bogen ist 1 % des Bogenradius.

Siehe auch

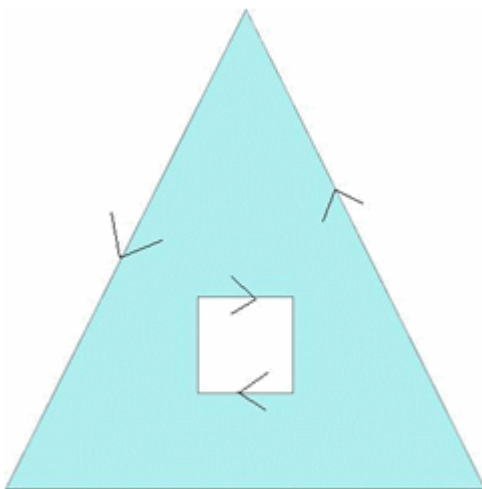
- „st_geometry_interpolation-Option“ [[SQL Anywhere Server - Datenbankadministration](#)]

Funktionsweise der Ausrichtung von Polygonringen

Intern interpretiert SQL Anywhere Polygone anhand der Ausrichtung der Ringe. Wenn Sie einem Ring in der Reihenfolge der definierten Punkte folgen, befindet sich die Innenseite des Polygons auf der linken Seite des Rings. Für räumliche Bezugssysteme mit planer und gewölbter Erddarstellung gelten dieselben Regeln. In den meisten Fällen sind äußere Ringe gegen den Uhrzeigersinn und innere Ringe im Uhrzeigersinn (umgekehrt) ausgerichtet. Eine Ausnahme gilt für Ringe, die den Nord- und den Südpol in der gewölbten Erddarstellung enthalten.

Standardmäßig werden Polygone automatisch neu ausgerichtet, wenn sie mit einer anderen Ringausrichtung erstellt werden als der internen Ringausrichtung in SQL Anywhere. Verwenden Sie die POLYGON FORMAT-Klausel der CREATE SPATIAL REFERENCE SYSTEM-Anweisung, um die Ausrichtung von Polygonringen der Eingabedaten festzulegen. Dies sollte nur durchgeführt werden, wenn alle Eingabedaten für das räumliche Bezugssystem dieselbe Ringausrichtung verwenden. Das Polygonformat kann auch für ein Polygon und Mehrfachoberflächen-Konstrukturen angegeben werden.

Beispiel: Wenn Sie ein Polygon erstellen und die Punkte im Uhrzeigersinn festlegen `Polygon((0 0, 5 10, 10 0, 0 0), (4 2, 4 4, 6 4, 6 2, 4 2))`, ordnet der Datenbankserver die Punkte automatisch wie folgt neu gegen den Uhrzeigersinn an: `Polygon((0 0, 10 0, 5 10, 0 0), (4 2, 4 4, 6 4, 6 2, 4 2))`.



Wenn der innere Ring vor dem äußeren Ring angegeben wurde, erscheint der äußere Ring als erster Ring.

Damit die Neuausrichtung von Polygonen in räumlichen Bezugssystemen mit gewölbter Erddarstellung einwandfrei erfolgen kann, sind Polygone auf einen Durchmesser von 160° begrenzt.

Siehe auch

- [POLYGON FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Funktionsweise von Innen- und Außenbereichen sowie von Begrenzungen von Geometrien

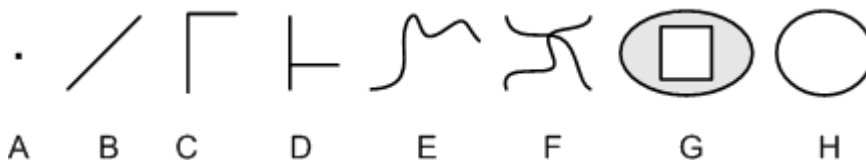
Der **Innenbereich** einer Geometrie umfasst alle Punkte, die zu dieser Geometrie gehören, ausgenommen der Begrenzung.

Der **Außenbereich** einer Geometrie umfasst alle Punkte, die nicht Teil der Geometrie sind. Im Fall eines Polygons mit einem Loch kann hierzu auch der Raum innerhalb eines inneren Rings gehören. Der Raum innerhalb und außerhalb eines Linienfolgenrings wird ebenfalls als Außenbereich betrachtet.

Zur **Begrenzung** einer Geometrie gehört alles, was von der Methode ST_Boundary zurückgegeben wird.

Die Kenntnis der Begrenzung einer Geometrie ist hilfreich, wenn sie mit einer anderen Geometrie verglichen werden soll, um zu ermitteln, in welcher Beziehung sie zueinander stehen. Während alle Geometrien einen Innenbereich und einen Außenbereich haben, besitzen nicht alle Geometrien eine Begrenzung, und ihre Begrenzungen sind auch nicht immer intuitiv.

In folgenden Fällen ist die Begrenzung von Geometrien möglicherweise nicht intuitiv:



- **Punkt** Ein Punkt (z. B. A) hat keine Begrenzung.
- **Linien und Kurven** Die Begrenzung für Linien und Kurven (B, C, D, E, F) sind ihre Endpunkte. Die Geometrien B, C und E haben zwei Endpunkte als Begrenzung. Geometrie D hat drei Endpunkte als Begrenzung und Geometrie F hat vier.
- **Polygon** Die Begrenzung für ein Polygon (z. B. G) ist ihr äußerer Ring und etwaige innere Ringe.
- **Ringe** Ein Ring, also eine Kurve, deren Startpunkt mit dem Endpunkt übereinstimmt und die keinen Schnittpunkt mit sich selbst hat (z. B. H), besitzt keine Begrenzung.

Siehe auch

- [ST_Boundary-Methode für den ST_Geometry-Datentyp auf Seite 160](#)

Funktionsweise räumlicher Vergleiche

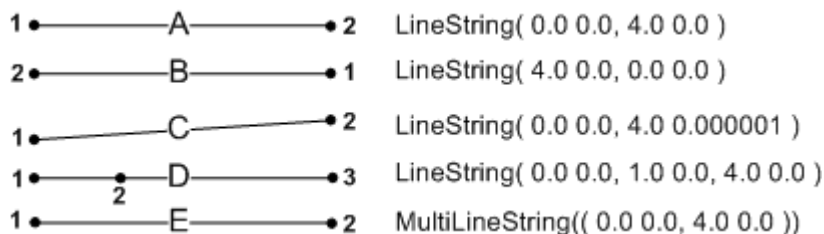
Es sind zwei Methoden verfügbar, um zu testen, ob eine Geometrie mit einer anderen Geometrie übereinstimmt: ST_Equals und ST_OrderingEquals. Diese Methoden führen den Vergleich auf unterschiedliche Weise aus und geben unterschiedliche Ergebnisse zurück.

- **ST_Equals** Die Reihenfolge, in der Punkte angegeben werden, ist unerheblich und der Punktevergleich berücksichtigt die Toleranz. Geometrien werden als gleich betrachtet, wenn sie

innerhalb der Toleranz denselben Raum belegen. Beispiel: Wenn zwei Linienfolgen denselben Raum belegen, werden sie als gleich betrachtet, auch wenn eine von ihnen mit mehr Punkten definiert wurde.

- **ST_OrderingEquals** Bei ST_OrderingEquals müssen die beiden Geometrien dieselbe Hierarchie von Objekten mit genau denselben Punkten in derselben Reihenfolge enthalten, um mit ST_OrderingEquals als gleich eingestuft zu werden. Das bedeutet, dass die beiden Geometrien genau gleich sein müssen.

Die folgenden Linien veranschaulichen die Unterschiede der Ergebnisse bei Vergleichen mithilfe von ST_Equals und ST_OrderingEquals. ST_Equals betrachtet sie alle als gleich (vorausgesetzt, Linie C liegt innerhalb der Toleranz). ST_OrderingEquals betrachtet alle Linien als unterschiedlich.



Vergleichen von Geometrien in SQL Anywhere

Der Datenbankserver verwendet ST_OrderingEquals, um Vorgänge wie GROUP BY und DISTINCT durchzuführen.

Beispiel: Wenn folgende Abfrage verarbeitet wird, behandelt der Server zwei Zeilen als gleich, wenn für die beiden Formausdrücke ST_OrderingEquals() = 1: festgelegt wurde.

```
SELECT DISTINCT Shape FROM GROUPO.SpatialShapes;
```

SQL-Anweisungen können zwei Geometrien mithilfe des Gleichheitsoperators (=) oder des Ungleichheitsoperators (<> oder !=) vergleichen, einschließlich Suchbedingungen mit einer Unterabfrage und dem Schlüsselwort ANY oder ALL. Geometrien können auch in einer IN-Suchbedingung verwendet werden. Zum Beispiel, geom1 IN (geom-expr1, geom-expr2, geom-expr3). Für alle diese Suchbedingungen wird die Übereinstimmung mit der Semantik von ST_OrderingEquals geprüft.

Es ist nicht möglich, andere Vergleichsoperatoren zu verwenden, um zu bestimmen, ob eine Geometrie kleiner oder größer als eine andere ist (Beispiel: geom1 < geom2 wird nicht akzeptiert). Das bedeutet, Geometrieausdrücke können nicht in eine ORDER BY-Klausel einbezogen werden. Sie können jedoch auf die Mitgliedschaft in einer Menge prüfen.

Funktionsweise räumlicher Beziehungen

Um eine optimale Performance zu erzielen, verwenden Sie Methoden wie ST_Within oder ST_Touches, um einzelne spezifische Beziehungen zwischen Geometrien zu testen. Wenn Sie jedoch mehrere Beziehungen testen müssen, empfiehlt sich die Methode ST_Relate, da damit auf mehrere Beziehungen gleichzeitig getestet werden kann. ST_Relate ist ebenfalls geeignet, um auf eine andere Interpretation eines Prädikats zu testen.

ST_Relate wird am häufigsten als Prädikat verwendet, wobei Sie die zu testenden Beziehungen präzise angeben. Sie können ST_Relate jedoch auch verwenden, um alle möglichen Beziehungen zwischen zwei Geometrien zu ermitteln.

Verwendung von ST_Relate als Prädikat

ST_Relate bestimmt anhand von **Schnittpunkttests** ihrer Innenbereiche, Begrenzungen und Außenbereiche, in welcher Beziehung Geometrien zueinander stehen. Die Beziehung zwischen den Geometrien wird dann in einer 9-stelligen Zeichenfolge in DE-9IM-Format (Dimensionally Extended 9 Intersection Model) beschrieben, wobei jedes Zeichen der Zeichenfolge die Dimension des Ergebnisses eines Schnittpunkttests darstellt.

Wenn Sie ST_Relate als Prädikat verwenden, übergeben Sie eine DE-9IM-Zeichenfolge, die die zu testenden Schnittpunktergebnisse widerspiegelt. Wenn die Geometrien die angegebenen Bedingungen in der DE-9IM-Zeichenfolge erfüllen, gibt ST_Relate **1** zurück. Wenn die Bedingungen nicht erfüllt sind, wird **0** zurückgegeben. Wenn eine oder beide Geometrien NULL sind, wird **NULL** zurückgegeben.

Die 9-stellige DE-9IM-Zeichenfolge ist eine abgeflachte Darstellung einer paarweisen Matrix der Schnittpunkttests zwischen Innenbereichen, Begrenzungen und Außenbereichen. Die folgende Tabelle zeigt die 9 Schnittpunkttests in der Reihenfolge, in der sie durchgeführt werden: von links nach rechts und von oben nach unten:

	g2-Innenbereich	g2-Begrenzung	g2-Außenbereich
g1-Innenbereich	$\text{Interior}(g1) \cap \text{Interior}(g2)$	$\text{Interior}(g1) \cap \text{Boundary}(g2)$	$\text{Interior}(g1) \cap \text{Exterior}(g2)$
g1-Begrenzung	$\text{Boundary}(g1) \cap \text{Interior}(g2)$	$\text{Boundary}(g1) \cap \text{Boundary}(g2)$	$\text{Boundary}(g1) \cap \text{Exterior}(g2)$
g1-Außenbereich	$\text{Exterior}(g1) \cap \text{Interior}(g2)$	$\text{Exterior}(g1) \cap \text{Boundary}(g2)$	$\text{Exterior}(g1) \cap \text{Exterior}(g2)$

Wenn Sie die DE-9IM-Zeichenfolge festlegen, können Sie für jedes der 9 Zeichen *, 0, 1, 2, T oder F angeben. Diese Werte beziehen sich auf die Anzahl der Bemessungen der durch den Schnittpunkt entstandenen Geometrie.

Sie geben z. B. Folgendes an:	Der Schnittpunkttest muss Folgendes zurückgeben:
T	Ein Wert von 0, 1, 2 (ein Schnittpunkt einer beliebigen Bemessung)
F	-1
*	-1, 0, 1, 2 (beliebiger Wert)
0	0
1	1

Sie geben z. B. Folgendes an:	Der Schnittpunkttest muss Folgendes zurückgeben:
2	2

Sie möchten z. B. mit ST_Relate und einer benutzerdefinierten DE-9IM-Zeichenfolge für das "innerhalb"-Prädikat testen, ob sich eine Geometrie *innerhalb* einer anderen Geometrie befindet:

```
SELECT new ST_Polygon('Polygon(( 2 3, 8 3, 4 8, 2 3 ))').ST_Relate( new
ST_Polygon('Polygon((-3 3, 3 3, 3 6, -3 6, -3 3))'), 'T*F**F***' );
```

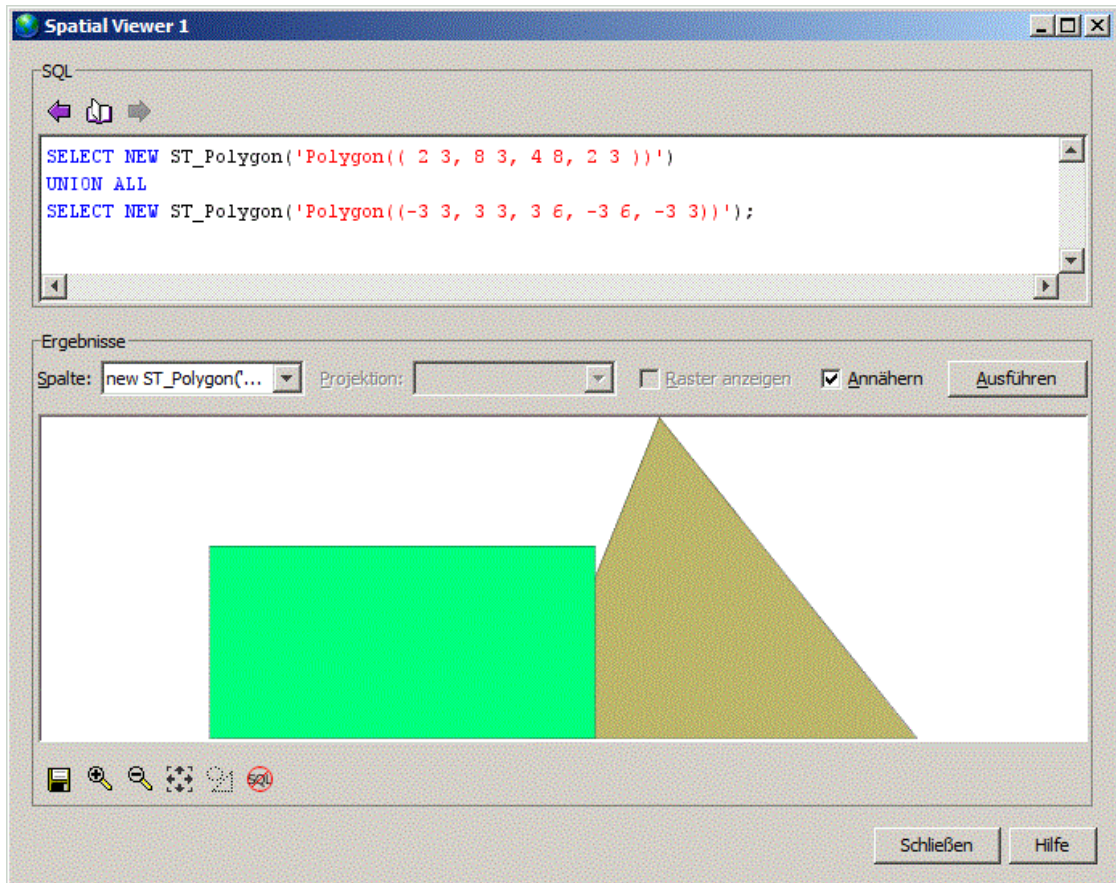
Dies entspricht der Prüfung auf folgende Bedingungen mit ST_Relate, wenn die Schnittpunkttests durchgeführt werden:

	g2-Innenbereich	g2-Begrenzung	g2-Außenbereich
g1-Innenbereich	Ein Wert von 0, 2, -1	Ein Wert von 0, 1, 2, -1	-1
g1-Begrenzung	Ein Wert von 0, 1, 2, -1	Ein Wert von 0, 1, 2, -1	-1
g1-Außenbereich	Ein Wert von 0, 1, 2, -1	Ein Wert von 0, 1, 2, -1	Ein Wert von 0, 1, 2, -1

Wenn Sie die Abfrage ausführen, gibt ST_Relate jedoch 0 zurück. Dadurch wird angezeigt, dass die erste Geometrie sich nicht innerhalb der zweiten Geometrie befindet.

Um die zwei Geometrien anzuzeigen und ihr Erscheinungsbild im Rahmen des Tests zu vergleichen, führen Sie folgende Anweisung im Spatial Viewer von Interactive SQL aus (**Extras » Spatial Viewer**):

```
SELECT NEW ST_Polygon('Polygon(( 2 3, 8 3, 4 8, 2 3 ))')
UNION ALL
SELECT NEW ST_Polygon('Polygon((-3 3, 3 3, 3 6, -3 6, -3 3))');
```



Siehe auch

- „ST_Relate(ST_Geometry,CHAR(9))-Methode für den ST_Geometry-Datentyp“ auf Seite 210

Verwendung von ST_Relate nicht als Prädikat

Wenn ST_Relate nicht als Prädikat verwendet wird, wird die vollständige Beziehung zwischen zwei Geometrien zurückgegeben.

Beispiel: Sie haben die beiden Geometrien, die im vorigen Beispiel verwendet wurden, und möchten wissen, in welcher Beziehung sie miteinander stehen. Sie führen folgende Anweisung in Interactive SQL aus, um die DE-9IM-Zeichenfolge zurückzugeben, die ihre Beziehung definiert:

```
SELECT new ST_Polygon('Polygon(( 2 3, 8 3, 4 8, 2 3 ))').ST_Relate(new
ST_Polygon('Polygon((-3 3, 3 3, 3 6, -3 6, -3 3))');
```

ST_Relate gibt die DE-9IM-Zeichenfolge zurück, 212111212.

Die Matrixansicht dieses Wertes zeigt, dass eine Reihe von Schnittpunkten vorliegen:

	g2-Innenbereich	g2-Begrenzung	g2-Außenbereich
--	-----------------	---------------	-----------------

g1-Innenbereich	2	1	2
g1-Begrenzung	1	1	1
g1-Außenbereich	2	1	2

Siehe auch

- [ST_Intersects-Methode für den ST_Geometry-Datentyp auf Seite 193](#)
- [ST_Overlaps-Methode für den ST_Geometry-Datentyp auf Seite 208](#)
- [ST_Within-Methode für den ST_Geometry-Datentyp auf Seite 241](#)
- [ST_Disjoint-Methode für den ST_Geometry-Datentyp auf Seite 177](#)
- [ST_Touches-Methode für den ST_Geometry-Datentyp auf Seite 236](#)
- [ST_Crosses-Methode für den ST_Geometry-Datentyp auf Seite 173](#)
- [ST_Contains-Methode für den ST_Geometry-Datentyp auf Seite 161](#)
- [ST_Relate-Methode für den ST_Geometry-Datentyp auf Seite 210](#)
- [„ST_Relate\(ST_Geometry\)-Methode für den ST_Geometry-Datentyp“ auf Seite 212](#)

Funktionsweise räumlicher Dimensionen

Jeder Geometrie-Subtyp besitzt eigene Eigenschaften und erbt zudem weitere Eigenschaften vom Obertyp ST_Geometry. Ein Geometrie-Subtyp hat einen der folgenden Bemaßungswerte:

- **-1** Der Wert -1 gibt an, dass die Geometrie leer ist (sie enthält keine Punkte).
- **0** Der Wert 0 gibt an, dass die Geometrie keine Länge und keinen Bereich hat. Die Subtypen ST_Point und ST_MultiPoint haben den Bemaßungswert 0. Ein Punkt stellt ein geometrisches Element dar, das durch ein einzelnes Paar von Koordinaten dargestellt werden kann. Ein Cluster nicht verbundener Punkte ist ein Mehrpunktelement.
- **1** Der Wert 1 gibt an, dass die Geometrie eine Länge hat, aber keinen Bereich besitzt. Die Gruppe von Subtypen mit einem Bemaßungswert 1 sind Subtypen von ST_Curve (ST_LineString, ST_CircularString und ST_CompoundCurve) oder Sammlungstypen, die diese Typen enthalten, aber keine Oberflächen. In GIS-Daten werden diese Geometrien mit dem Bemaßungswert 1 verwendet, um lineare Elemente, wie Bäche, verzweigte Flusssysteme und Straßensegmente zu definieren.
- **2** Der Wert 2 gibt an, dass die Geometrie einen Bereich hat. Die Gruppe von Subtypen mit einem Bemaßungswert 2 sind Subtypen von ST_Surface (ST_Polygon und ST_CurvePolygon) oder Sammlungstypen, die diese Typen enthalten. Polygone und Multipolygone stellen geometrische Elemente mit einem Umfang dar, der einen definierten Bereich, wie etwa Seen oder Parks, umschließt.

Die Dimension einer Geometrie entspricht nicht der Anzahl der Koordinatendimensionen der einzelnen Punkte in der Geometrie.

Ein einzelner ST_GeomCollection-Typ kann Geometrien verschiedener Bemaßungswerte enthalten und die Geometrie mit dem höchsten Bemaßungswert wird zurückgegeben.

Siehe auch

- [ST_CoordDim-Methode für den ST_Geometry-Datentyp auf Seite 167](#)
- [ST_Dimension-Methode für den ST_Geometry-Datentyp auf Seite 176](#)

Praktische Einführung: Mit den räumlichen Funktionen experimentieren

In dieser praktischen Einführung können Sie mit einigen der räumlichen Funktionen in SQL Anywhere experimentieren. Hierzu lesen Sie zunächst eine ESRI-Formdatei in Ihre Beispieldatenbank (demo.db) ein, sodass Sie gültige räumliche Daten für die Übung zur Verfügung haben.

Die praktische Einführung ist in folgende Teile unterteilt:

- [„Lektion 1: Zusätzliche Maßeinheiten und räumliche Bezugssysteme installieren“ auf Seite 57](#)
- [„Lektion 2: Die Daten der ESRI-Formdatei herunterladen“ auf Seite 58](#)
- [„Lektion 3: Die Daten der ESRI-Formdatei einlesen“ auf Seite 59](#)
- [„Lektion 4: Räumliche Daten abfragen“ auf Seite 62](#)
- [„Lektion 5: Ausgabe von räumlichen Daten in SVG“ auf Seite 65](#)
- [„Lektion 6: Räumliche Daten projizieren“ auf Seite 67](#)

Privilegien

Um diese praktischen Übung ausführen zu können, benötigen Sie die folgenden Privilegien:

- `MANAGE ANY SPATIAL OBJECT`-Systemprivileg
- `CREATE TABLE`-Systemprivileg
- `WRITE FILE`-Systemprivileg
- `SELECT`-Privileg für die Tabelle `GROUPO.SpatialContacts`

Lektion 1: Zusätzliche Maßeinheiten und räumliche Bezugssysteme installieren

Diese Lektion zeigt, wie Sie die `sa_describe_shapefile`-Systemprozedur verwenden, um zahlreiche vordefinierte Maßeinheiten und räumliche Bezugssysteme zu installieren, die Sie später in dieser praktischen Einführung benötigen werden.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: [„Praktische Einführung: Mit den räumlichen Funktionen experimentieren“ auf Seite 57](#).

Aufgabe

1. Verwenden Sie Interactive SQL, um die Beispieldatenbank (demo.db) zu starten und eine Verbindung zu ihr herzustellen.

Die Beispieldatenbank befindet sich in `%SQLANYSAMPI6%`.

2. Führen Sie die folgende Anweisung in Interactive SQL aus:

```
CALL sa_install_feature( 'st_geometry_predefined_srs' );
```

Die Anweisung installiert die vordefinierten Maßeinheiten und räumlichen Bezugssysteme.

3. Mit der folgenden Abfrage in Interactive SQL können Sie ermitteln, welche Maßeinheiten in der Datenbank installiert sind:

```
SELECT * FROM ST_UNITS_OF_MEASURE;
```

4. Welche räumlichen Bezugssysteme in der Datenbank installiert sind, können Sie im Ordner **Räumliche Bezugssysteme** in Sybase Central oder mithilfe der folgenden Abfrage in Interactive SQL überprüfen:

```
SELECT * FROM ST_SPATIAL_REFERENCE_SYSTEMS;
```

Ergebnisse

Die Liste der installierten räumlichen Bezugssysteme wird zurückgegeben.

Nächste Schritte

Gehen Sie weiter zu „[Lektion 2: Die Daten der ESRI-Formdatei herunterladen](#)“ auf Seite 58.

Siehe auch

- „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „CREATE SPATIAL REFERENCE SYSTEM-Anweisung“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „Konsolidierte Ansicht ST_UNITS_OF_MEASURE“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „Konsolidierte Ansicht ST_SPATIAL_REFERENCE_SYSTEMS“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]

Lektion 2: Die Daten der ESRI-Formdatei herunterladen

In dieser Lektion laden Sie eine ESRI-Formdatei von der US Census-Website (www2.census.gov) herunter. Die heruntergeladene Formdatei enthält die 5-stelligen PLZ-Informationen, die bei der Volkszählung 2002 tabellarisch gesammelt wurden. Jede PLZ-Region wird als Polygon oder Multipolygon behandelt.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „[Lektion 1: Zusätzliche Maßeinheiten und räumliche Bezugssysteme installieren](#)“ auf Seite 57.

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: „[Praktische Einführung: Mit den räumlichen Funktionen experimentieren](#)“ auf Seite 57.

Aufgabe

1. Erstellen Sie ein lokales Verzeichnis mit dem Namen *c:\temp\massdata*.
2. Gehen Sie zur URL <http://www2.census.gov/cgi-bin/shapefiles2009/national-files>
3. Klicken Sie auf der rechten Seite in der Dropdown-Liste **State- and County-based Shapefiles** (Bundesland/Kanton- und Landkreis-basierte Formdateien) auf **Massachusetts** und anschließend auf **Submit** (Senden).
4. Klicken Sie auf der linken Seite auf **5-Digit ZIP Code Tabulation Area (2002)** (5-stelliger PLZ-Tabellenbereich (2002)) und anschließend auf **Download Selected Files** (Ausgewählte Dateien herunterladen).
5. Wenn Sie dazu aufgefordert werden, speichern Sie die ZIP-Datei *multiple_tiger_files.zip* unter *c:\temp\massdata* und extrahieren Sie ihren Inhalt. Damit erstellen Sie ein Unterverzeichnis namens *25_MASSACHUSETTS*, das eine weitere ZIP-Datei namens *tl_2009_25_zcta5.zip* enthält.
6. Extrahieren Sie den Inhalt von *tl_2009_25_zcta5.zip* nach *C:\temp\massdata*.

Ergebnisse

In dieser Lektion werden fünf Dateien entpackt, einschließlich einer ESRI-Formdatei (.shp), die Sie verwenden können, um die räumlichen Daten in die Datenbank zu laden.

Nächste Schritte

Gehen Sie weiter zu „[Lektion 3: Die Daten der ESRI-Formdatei einlesen](#)“ auf Seite 59.

Lektion 3: Die Daten der ESRI-Formdatei einlesen

Diese Lektion zeigt, wie Sie ermitteln, welche Spalten in der ESRI-Formdatei enthalten sind, und wie Sie diese Informationen verwenden, um eine Tabelle zu erstellen, in die Sie die Daten einlesen.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „[Lektion 1: Zusätzliche Maßeinheiten und räumliche Bezugssysteme installieren](#)“ auf Seite 57.

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: „[Praktische Einführung: Mit den räumlichen Funktionen experimentieren](#)“ auf Seite 57.

Kontext und Bemerkungen

Wenn Sie aufgrund von Privilegproblemen Schwierigkeiten mit der Ausführung der Schritte haben, fragen Sie Ihren Administrator, auf welchen Wert die Datenbankoption -gl gesetzt ist. Ermitteln Sie anschließend im Abschnitt zu den Privilegien für die st_geometry_load_shapefile-Systemprozedur die benötigten Privilegien.

Aufgabe

1. Da die räumlichen Daten einem spezifischen räumlichen Bezugssystem zugeordnet sind, müssen Sie sie beim Einlesen der Daten in die Datenbank in dasselbe räumliche Bezugssystem einlesen oder zumindest in eines mit der gleichen Definition. Um die Informationen des räumlichen Bezugssystems für die ESRI-Formdatei zu ermitteln, öffnen Sie die Projektdatei `c:\temp\massdata\tl_2009_25_zcta5.prj` in einem Texteditor. Diese Datei enthält die benötigten Informationen über das räumliche Bezugssystem.

```
GEOGCS["GCS_North_American_1983", DATUM["D_North_American_1983",  
SPHEROID["GRS_1980",6378137,298.257222101]],  
PRIMEM["Greenwich",0],UNIT["Degree",0.017453292519943295]]
```

Die Zeichenfolge **GCS_North_American_1983** ist der Name des räumlichen Bezugssystems, dem die Daten zugeordnet ist.

2. Eine kurze Abfrage der Ansicht ST_SPATIAL_REFERENCE_SYSTEM, `SELECT * FROM ST_SPATIAL_REFERENCE_SYSTEMS WHERE srs_name= 'GCS_North_American_1983' ;`, zeigt, dass dieser Name nicht in der Liste der vordefinierten räumlichen Bezugssysteme enthalten ist. Sie können jedoch eine Abfrage für ein räumliches Bezugssystem mit derselben Definition erstellen und dieses stattdessen verwenden.

```
SELECT *  
FROM ST_SPATIAL_REFERENCE_SYSTEMS  
WHERE definition LIKE '%1983%'  
AND definition LIKE 'GEOGCS%';
```

Die Abfrage gibt ein einziges räumliches Bezugssystem, NAD83, mit SRID **4269** zurück, das die gleiche Definition hat. Dies ist die SRID, die Sie den Daten zuordnen werden, die Sie aus der Formdatei einlesen.

3. Führen Sie in Interactive SQL folgende Anweisung aus, um eine Tabelle namens Massdata zu erstellen, die Formdatei in die Tabelle einzulesen und den Daten die SRID 4269 zuzuweisen. Der Ladevorgang kann eine Minute dauern.

```
CALL st_geometry_load_shapefile ( 'c:\\temp\\massdata\  
\tl_2009_25_zcta5.shp',  
4269,  
'Massdata' );
```

Hinweis

Der **Import-Assistent** unterstützt auch das Einlesen von Daten aus Formdateien.

4. Fragen Sie in Interactive SQL Abfrage die Tabelle ab, um die Daten anzuzeigen, die in der Formdatei waren:

```
SELECT * FROM Massdata;
```

Jede Zeile in den Ergebnissen gibt Daten für eine PLZ-Region an:

Die Geometrie-Spalte enthält die Forminformationen der PLZ-Region als Polygon (ein Bereich) oder Multipolygon (zwei oder mehr nicht zusammenhängende Bereiche).

- Die Spalte ZCTA5CE enthält Zipcodes (Postleitzahlen). Um die Verwendung dieser Spalte später in der praktischen Einführung zu erleichtern, führen Sie die folgende ALTER TABLE-Anweisung in Interactive SQL aus, um den Spaltennamen auf **ZIP** zu ändern:

```
ALTER TABLE Massdata  
RENAME ZCTA5CE TO ZIP;
```

- Die beiden Spalten INTPTLON und INTPTLAT stellen die X- und Y-Koordinaten für die Mittelpunkte der Zipcode-Regionen dar. Führen Sie die folgende ALTER TABLE-Anweisung in Interactive SQL aus, um eine Spalte namens CenterPoint vom Typ ST_Point erstellen und um den X- und Y-Wert in einen Wert in CenterPoint zu verwandeln.

```
ALTER TABLE Massdata  
ADD CenterPoint AS ST_Point(SRID=4269)  
COMPUTE( new ST_Point( CAST( INTPTLON AS DOUBLE ), CAST( INTPTLAT AS  
DOUBLE ), 4269 ) );
```

Nun stellt jeder ST_Point-Wert in Massdata.CenterPoint den Mittelpunkt der Zipcode-Region dar, die in Massdata.geometry gespeichert ist.

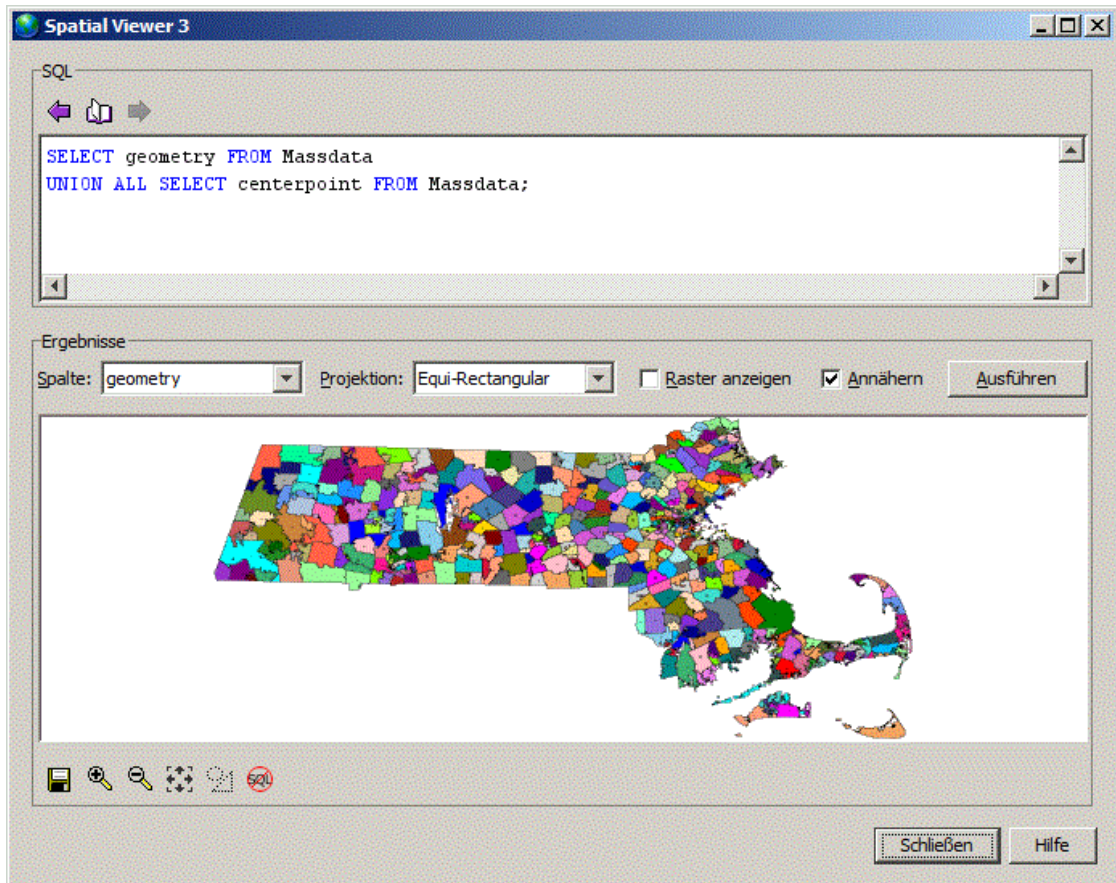
- Um eine einzelne Geometrie (eine PLZ-Region) als Form anzuzeigen, doppelklicken Sie auf einen beliebigen Wert (ausgenommen den ersten) in Massdata.geometry und klicken Sie dann im Fenster **Wert von Spalte** auf die Registerkarte **Räumliche Vorschau**.

Wenn Sie in einer Fehlermeldung darauf hingewiesen werden, dass der Wert zu lang ist oder dass Sie einen Primärschlüssel in die Ergebnisse einbeziehen müssen, liegt das daran, dass der Wert für die Anzeige in Interactive SQL gekürzt wurde. Um dies zu beheben, können Sie die Abfrage ändern und eine Primärschlüsselspalte in die Ergebnisse einbeziehen oder die Einstellung **Kürzungslänge** anpassen. Wenn Sie den Primärschlüssel nicht bei jeder Abfrage einer Geometrie für die Anzeige in Interactive SQL einbeziehen wollen, wird die Einstellung der Kürzungslänge empfohlen.

Zum Ändern der Einstellung **Kürzungslänge** für Interactive SQL klicken Sie auf **Extras » Optionen » SQL Anywhere** und legen Sie **Kürzungslänge** auf einen hohen Wert fest, z. B. auf 100000.

- Um den gesamten Datensatz als eine einzige Form anzuzeigen, klicken Sie auf **Extras » Spatial Viewer**, um den **Spatial Viewer** von SQL Anywhere zu öffnen, und führen Sie folgende Abfrage in Interactive SQL aus:

```
SELECT geometry FROM Massdata  
UNION ALL SELECT CenterPoint FROM Massdata;
```



Ergebnisse

Die Daten der ESRI-Formdatei werden geladen.

Nächste Schritte

Gehen Sie weiter zu „[Lektion 4: Räumliche Daten abfragen](#)“ auf Seite 62.

Siehe auch

- „[st_geometry_load_shapefile-Systemprozedur](#)“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)]
- „[Importieren von Daten mit dem Import-Assistenten](#)“ [[SQL Anywhere Server - SQL-Benutzerhandbuch](#)]

Lektion 4: Räumliche Daten abfragen

Diese Lektion zeigt, wie Sie mit einigen der räumlichen Methoden Daten in einem bestimmten Kontext abfragen können. Es wird auch gezeigt, wie Abstände berechnet werden. Hierfür müssen Sie Maßeinheiten in die Datenbank einfügen.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „[Lektion 1: Zusätzliche Maßeinheiten und räumliche Bezugssysteme installieren](#)“ auf Seite 57.

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: „[Praktische Einführung: Mit den räumlichen Funktionen experimentieren](#)“ auf Seite 57.

Kontext und Bemerkungen

Die Abfragen werden in der Tabelle SpatialContacts oder Massdata oder in beiden Tabellen ausgeführt. Die Tabelle SpatialContacts, die bereits in Ihrer Datenbank vorhanden war, enthält Namen und Kontaktinformationen für Personen, von denen viele in Massachusetts leben.

Abfragen der räumlichen Daten

1. In Interactive SQL erstellen Sie eine Variable namens @Mass_01775, um die zugehörige Geometrie für die PLZ-Region 01775 zu speichern.

```
CREATE VARIABLE @Mass_01775 ST_Geometry;  
SELECT geometry INTO @Mass_01775  
FROM Massdata  
WHERE ZIP = '01775';
```

2. Sie möchten z. B. alle Kontakte in SpatialContacts im PLZ-Bereich 01775 und den umgebenden PLZ-Bereichen suchen. Hierzu können Sie die ST_Intersects-Methode verwenden, die Geometrien zurückgibt, die sich mit der angegebenen Geometrie überschneiden oder mit ihr übereinstimmen. Sie können z.B. die folgende Anweisung in Interactive SQL ausführen:

```
SELECT c.Surname, c.GivenName, c.Street, c.City, c.PostalCode, z.geometry  
FROM Massdata z, GPOUO.SpatialContacts c  
WHERE  
c.PostalCode = z.ZIP  
AND z.geometry.ST_Intersects( @Mass_01775 ) = 1;
```

Siehe [ST_Intersects-Methode für den ST_Geometry-Datentyp auf Seite 193](#).

3. Alle Zeilen in Massdata.geometry sind demselben räumlichen Bezugssystem (SRID 4269) zugeordnet, da der Geometriespalte bei der Erstellung die SRID 4269 zugeordnet wurde und Daten in sie eingelesen wurden.

Es ist jedoch auch möglich, eine **nicht deklarierte** ST_Geometry-Spalte zu erstellen (d. h. ohne Zuweisung einer SRID). Dies kann erforderlich sein, wenn Sie planen, in einer einzelnen Spalte räumliche Daten zu speichern, denen unterschiedliche räumliche Bezugssysteme zugeordnet sind. Wenn diese Werte bearbeitet werden, wird jeweils das räumliche Bezugssystem verwendet, das dem betreffenden Wert zugeordnet ist.

Wenn eine Spalte nicht deklariert wird, besteht die Gefahr, dass der Datenbankserver die Änderung eines räumlichen Bezugssystems zulässt, das den Daten in der nicht deklarierten Spalte zugeordnet ist.

Wenn die Spalte eine deklarierte SRID hat, erlaubt der Datenbankserver es nicht, das räumliche Bezugssystem, das den Daten zugeordnet ist, zu verändern. Sie müssen die Daten zuerst aus der Spalte auslesen und dann kürzen, das räumliche Bezugssystem ändern und dann die Daten erneut einlesen.

Mit der Methode ST_SRID können Sie die SRID ermitteln, die Werten einer Spalte zugeordnet ist, und zwar unabhängig davon, ob diese deklariert ist. Beispiel: Folgende Anweisung zeigt die SRID, die den einzelnen Zeilen in der Spalte Massdata.geometry zugeordnet ist:

```
SELECT geometry.ST_SRID()  
FROM Massdata;
```

4. Mit der ST_CoveredBy-Methode können Sie prüfen, ob eine Geometrie vollständig in einer anderen Geometrie enthalten ist. Beispiel: Massdata.CenterPoint (ST_Point-Typ) enthält die Breitengrad-/Längengrad-Koordinaten des Mittelpunkts des PLZ-Bereichs, während Massdata.geometry das Polygon enthält, das den PLZ-Bereich darstellt. Sie können schnell prüfen, ob ein CenterPoint-Wert außerhalb des PLZ-Bereichs liegt, indem Sie folgende Abfrage in Interactive SQL ausführen:

```
SELECT * FROM Massdata  
WHERE NOT(CenterPoint.ST_CoveredBy(geometry) = 1);
```

Es werden keine Zeilen zurückgegeben. Dies zeigt an, dass alle CenterPoint-Werte innerhalb ihrer in Massdata.geometry zugeordneten Geometrien liegen. Diese Prüfung stellt natürlich nicht sicher, dass es sich um die tatsächlichen Mittelpunkte handelt. Sie müssten die Daten auf ein räumliches Bezugssystem für die plane Erddarstellung projizieren und die CenterPoint-Werte mit der ST_Centroid-Methode prüfen. In dieser praktischen Einführung erhalten Sie später Informationen zur Projektion.

5. Mit der Methode ST_Distance können Sie den Abstand zwischen zwei Mittelpunkten der PLZ-Bereiche messen. Sie möchten z. B. die Liste aller Postleitzahlen innerhalb von 100 Meilen vom PLZ-Bereich 01775. Führen Sie z.B. die folgende Abfrage in Interactive SQL aus:

```
SELECT c.PostalCode, c.City,  
       z.CenterPoint.ST_Distance( ( SELECT CenterPoint  
                                   FROM Massdata WHERE ZIP = '01775' ),  
                                   'Statute mile' ) dist,  
       z.CenterPoint  
FROM Massdata z, GROUPO.SpatialContacts c  
WHERE c.PostalCode = z.ZIP  
      AND dist <= 100  
ORDER BY dist;
```

6. Wenn die Kenntnis der genauen Entfernung nicht von Bedeutung ist, können Sie statt dessen die Abfrage unter Verwendung der ST_WithinDistance-Methode erstellen, die bei bestimmten Datensätzen eine bessere Performance bietet (insbesondere bei großen Geometrien):

```
SELECT c.PostalCode, c.City, z.CenterPoint  
FROM Massdata z, GROUPO.SpatialContacts c  
WHERE c.PostalCode = z.ZIP  
      AND z.CenterPoint.ST_WithinDistance( ( SELECT CenterPoint  
                                              FROM Massdata WHERE ZIP = '01775' ),  
                                              100, 'Statute mile' ) = 1  
ORDER BY c.PostalCode;
```

Ergebnisse

Die Abfragen werden auf die räumlichen Daten durchgeführt.

Nächste Schritte

Gehen Sie weiter zu „[Lektion 5: Ausgabe von räumlichen Daten in SVG](#)“ auf Seite 65.

Siehe auch

- [ST_SRID-Methode für den ST_Geometry-Datentyp auf Seite 214](#)
- [ST_CoveredBy-Methode für den ST_Geometry-Datentyp auf Seite 168](#)
- [ST_Distance-Methode für den ST_Geometry-Datentyp auf Seite 178](#)
- [ST_WithinDistance-Methode für den ST_Geometry-Datentyp auf Seite 242](#)

Lektion 5: Ausgabe von räumlichen Daten in SVG

In dieser Lektion erstellen Sie ein SVG-Dokument, um ein Multipolygon in WKT anzuzeigen. Sie können Geometrien in das SVG-Format zur Anzeige in Interactive SQL oder in einer SVG-kompatiblen Anwendung exportieren.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „[Lektion 1: Zusätzliche Maßeinheiten und räumliche Bezugssysteme installieren](#)“ auf Seite 57.

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: „[Praktische Einführung: Mit den räumlichen Funktionen experimentieren](#)“ auf Seite 57.

Aufgabe

1. Führen Sie folgende Anweisung in Interactive SQL aus, um eine Variable mit einer Beispielgeometrie zu erstellen:

```
CREATE OR REPLACE VARIABLE @svg_geom  
ST_Polygon = (NEW ST_Polygon('Polygon ((1 1, 5 1, 5 5, 1 5, 1 1), (2 2, 2  
3, 3 3, 3 2, 2 2))'));
```

2. Führen Sie folgende SELECT-Anweisung in Interactive SQL aus, um die ST_AsSVG-Methode aufzurufen:

```
SELECT @svg_geom.ST_AsSVG() AS svg;
```

Die Ergebnismenge hat eine einzelne Zeile, die sich in einem SVG-Bild befindet. Sie können das Bild mit der Funktion **SVG-Vorschau** in Interactive SQL anzeigen. Hierzu doppelklicken Sie auf die Ergebniszeile und wählen Sie die Registerkarte **SVG-Vorschau**. Sie sollten eine Quadratgeometrie innerhalb einer anderen Quadratgeometrie sehen.

Hinweis

Standardmäßig kürzt Interactive SQL-Werte im Fensterausschnitt **Ergebnisse** auf 256 Zeichen. Wenn Interactive SQL den Fehler meldet, dass der vollständige Spaltenwert nicht gelesen werden konnte, erhöhen Sie den Kürzungswert. Klicken Sie dazu auf **Extras » Optionen** und anschließend im linken Fensterausschnitt auf **SQL Anywhere**. Ändern Sie auf der Registerkarte **Ergebnisse** die Option **Kürzungslänge** auf einen hohen Wert, z.B. 5000. Klicken Sie auf **OK**, um Ihre Änderungen zu speichern, führen Sie die Abfrage erneut aus und klicken Sie dann noch einmal doppelt auf die Zeile.

- Der letzte Schritt zeigte, wie ein SVG-Bild in Interactive SQL in der Vorschau angezeigt wird. Es kann jedoch sinnvoller sein, das resultierende SVG-Bild in eine Datei zu schreiben, sodass es von einer externen Anwendung gelesen werden kann. Hierzu können Sie die `xp_write_file`-Systemprozedur oder die `WRITE_CLIENT_FILE`-Funktion [Zeichenfolge] verwenden, um in eine Datei des Datenbankservers oder des Clientcomputers zu schreiben. In diesem Beispiel wird die `OUTPUT`-Anweisung [Interactive SQL] verwendet.

Führen Sie in Interactive SQL folgende `SELECT`-Anweisung aus, um die `ST_AsSVG`-Methode aufzurufen und die Geometrie in eine Datei namens `myPolygon.svg` auszugeben.

```
SELECT @svg_geom.ST_AsSVG( );
OUTPUT TO 'c:\\temp\\massdata\\myPolygon.svg'
QUOTE ''
ESCAPES OFF
FORMAT TEXT
```

Sie müssen die Klauseln `QUOTE ''` und `ESCAPES OFF` einbeziehen, da Zeilenvorschubzeichen und einfache Anführungszeichen sonst zur Beibehaltung von Leerzeichen in den XML-Code eingefügt werden, wodurch eine ungültige SVG-Datei ausgegeben würde.

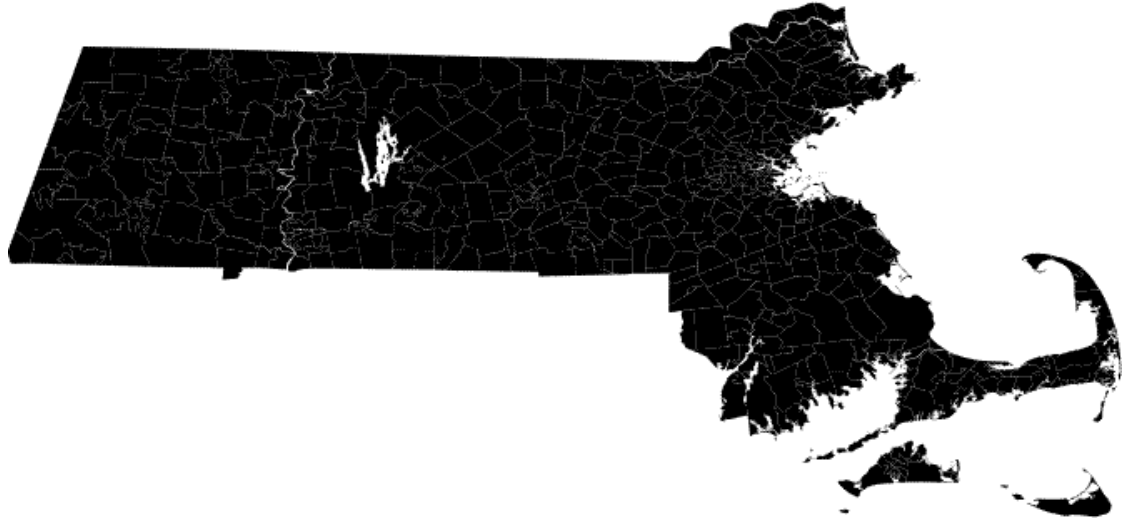
- Öffnen Sie die SVG-Datei in einem Webbrowser oder in einer Anwendung, die die Anzeige von SVG-Bildern unterstützt. Alternativ dazu können Sie die SVG-Datei auch in einem Texteditor öffnen, um den XML-Code für die Geometrie anzuzeigen.
- Die `ST_AsSVG`-Methode generiert aus einer einzelnen Geometrie ein SVG-Bild. In manchen Fällen soll ein SVG-Bild mit allen Formen in einer Gruppe generiert werden. Die `ST_AsSVGAgr`-Methode ist eine Aggregatfunktion, die mehrere Geometrien in einem einzigen SVG-Bild kombiniert. Erstellen Sie zunächst in Interactive SQL eine Variable, um das SVG-Bild zu speichern, und generieren Sie sie mit der `ST_AsSVGAgr`-Methode.

```
CREATE OR REPLACE VARIABLE @svg XML;
SELECT ST_Geometry::ST_AsSVGAgr( geometry, 'attribute=fill="black"' )
INTO @svg
FROM Massdata;
```

Die Variable `@svg` enthält nun ein SVG-Bild, das alle PLZ-Regionen in der `Massdata`-Tabelle darstellt. `'attribute=fill="black"'` gibt die Füllfarbe an, die für das generierte Bild verwendet wird. Wenn keine Farbe angegeben wird, verwendet der Datenbankserver eine willkürliche Füllfarbe. Nachdem die Variable mit dem gewünschten SVG-Bild erstellt wurde, kann sie in eine Datei zur Anzeige in anderen Anwendungen geschrieben werden. Führen Sie folgende Anweisung in Interactive SQL aus, um das SVG-Bild in eine Datei auf dem Datenbankserver zu schreiben.

```
CALL xp_write_file( 'c:\\temp\\Massdata.svg', @svg );
```

Die WRITE_CLIENT_FILE-Funktion könnte auch verwendet werden, um eine Datei relativ zur Clientanwendung zu schreiben. Dabei wären jedoch möglicherweise weitere Schritte erforderlich, um sicherzustellen, dass entsprechende Privilegien aktiviert sind. Wenn Sie das SVG-Bild in einer Anwendung öffnen, das SVG-Daten unterstützt, sollte folgendes Bild angezeigt werden.



Das Bild ist nicht einheitlich schwarz. Es sind kleine Lücken zwischen den Grenzen benachbarter Postleitzahlregionen zu erkennen. Dies sind weiße Linien zwischen den Geometrien, die charakteristisch dafür sind, wie das SVG-Bild gerendert wird. In den Daten sind nicht tatsächlich Lücken vorhanden. Breitere weiße Linien sind Flüsse und Seen.

Ergebnisse

Die Geometrie wurde als SVG-Bild angezeigt.

Nächste Schritte

Gehen Sie weiter zu [„Lektion 6: Räumliche Daten projizieren“](#) auf Seite 67.

Siehe auch

- [„OUTPUT-Anweisung \[Interactive SQL\]“](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- [ST_AsSVG-Methode für den ST_Geometry-Datentyp](#) auf Seite 127
- [ST_AsSVGAgr-Methode für den ST_Geometry-Datentyp](#) auf Seite 131
- [„xp_write_file-Systemprozedur“](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*]
- [„WRITE_CLIENT_FILE-Funktion \[Zeichenfolge\]“](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*]

Lektion 6: Räumliche Daten projizieren

Diese Lektion zeigt, wie Sie Daten in einem räumlichen Bezugssystem mit planer Erddarstellung projizieren, sodass Sie die Bereichs- und Abstandsmessungen berechnen können.

Voraussetzungen

In dieser Lektion wird davon ausgegangen, dass Sie bereits alle vorherigen Lektionen abgeschlossen haben. Siehe „[Lektion 1: Zusätzliche Maßeinheiten und räumliche Bezugssysteme installieren](#)“ auf Seite 57.

In dieser Lektion wird davon ausgegangen, dass Sie die Rollen und Privilegien haben, die im Abschnitt "Privilegien" am Anfang dieser praktischen Einführung aufgeführt sind: „[Praktische Einführung: Mit den räumlichen Funktionen experimentieren](#)“ auf Seite 57.

Kontext und Bemerkungen

Die räumlichen Werte in Massdata wurden SRID 4269 (räumliches Bezugssystem NAD83) zugeordnet, als Sie die Daten aus der ESRI-Formdatei in die Datenbank eingelesen haben. SRID 4269 ist ein räumliches Bezugssystem mit gewölbter Erddarstellung. Bestimmte Berechnungen, wie die des Bereichs von Geometrien und einiger räumlicher Prädikate, werden im Modell der gewölbten Erddarstellung jedoch nicht unterstützt. Wenn Ihre Daten derzeit einem räumlichen Bezugssystem mit gewölbter Erddarstellung zugeordnet sind, können Sie eine neue Spalte für räumliche Daten erstellen, die die Werte in ein räumliches Bezugssystem mit planer Erddarstellung projiziert, und dann die Berechnungen mit dieser Spalte durchführen.

Aufgabe

1. Zur Messung des Bereichs der Polygone, die die PLZ-Bereiche darstellen, müssen Sie die Daten in Massdata.geometry in ein räumliches Bezugssystem mit planer Erddarstellung projizieren.

Zur Auswahl einer geeigneten SRID für die Projektion der Daten in Massdata.geometry fragen Sie die konsolidierte Ansicht ST_SPATIAL_REFERENCE_SYSTEM in Interactive SQL folgendermaßen nach einer SRID ab, die das Wort Massachusetts enthält:

```
SELECT * FROM ST_SPATIAL_REFERENCE_SYSTEMS WHERE srs_name LIKE  
'%massachusetts%';
```

Diese Abfrage gibt mehrere SRIDs zur Verwendung mit den Massachusetts-Daten zurück. Für diese praktische Einführung wird **3586** verwendet.

2. Sie müssen nun die Spalte Massdata.proj_geometry erstellen, in die Sie die Geometrien in 3586 mit der ST_Transform-Methode projizieren werden. Führen Sie hierzu die folgende Anweisung in Interactive SQL aus:

```
ALTER TABLE Massdata  
ADD proj_geometry  
AS ST_Geometry(SRID=3586)  
COMPUTE( geometry.ST_Transform( 3586 ) );
```

3. Sie können den Bereich mit Massdata.proj_geometry berechnen. Führen Sie beispielsweise folgende Anweisung in Interactive SQL aus:

```
SELECT zip, proj_geometry.ST_ToMultiPolygon().ST_Area('Statute Mile') AS  
area  
FROM Massdata  
ORDER BY area DESC;
```

Hinweis

ST_Area wird nur zwischen Punktgeometrien, aber nicht in räumlichen Bezugssystemen mit gewölbter Erddarstellung unterstützt.

- Um die Wirkung zu sehen, die die Projektion in ein anderes räumliches Bezugssystem auf die Abstandsberechnungen hat, können Sie mit folgender Abfrage den Abstand zwischen den Mittelpunkten der Postleitzahlen mit dem Modell mit gewölbter Erddarstellung (präziser) sowie mit dem Modell mit planer Erddarstellung berechnen. Beide Modelle sind für diese Daten brauchbar, da die Projektion für den Datensatz geeignet ist.

```
SELECT M1.zip, M2.zip,
       M1.CenterPoint.ST_Distance( M2.CenterPoint, 'Statute Mile' )
dist_round_earth,

M1.CenterPoint.ST_Transform( 3586 ).ST_Distance( M2.CenterPoint.ST_Transform( 3586 ),
       'Statute Mile' ) dist_flat_earth
FROM Massdata M1, Massdata M2
WHERE M1.ZIP = '01775'
ORDER BY dist_round_earth DESC;
```

- Angenommen, Sie möchten PLZ-Bereiche suchen, die an den PLZ-Bereich von 01775 angrenzen. Hierzu verwenden Sie die ST_Touches-Methode. Die ST_Touches-Methode vergleicht die Geometrien, um festzustellen, ob eine Geometrie eine andere berührt, ohne sie zu überlappen. Die Ergebnisse für ST_Touches enthalten nicht die Zeile für die Postleitzahl 01775 (im Gegensatz zur ST_Intersects-Methode).

```
CREATE OR REPLACE VARIABLE @Mass_01775 ST_Geometry;
SELECT geometry INTO @Mass_01775
FROM Massdata
WHERE ZIP = '01775';

SELECT record_number, proj_geometry
FROM Massdata
WHERE proj_geometry.ST_Touches( @Mass_01775.ST_Transform( 3586 ) ) = 1;
```

- Mit der Methode ST_UnionAggr können Sie eine Geometrie zurückgeben, die die Vereinigung einer Gruppe von PLZ-Bereichen darstellt. Beispiel: Sie möchten eine Geometrie erstellen, die die Vereinigung der benachbarten PLZ-Bereiche des PLZ-Bereichs 01775 darstellt, unter Ausschluss von 01775 selbst.

In Interactive SQL klicken Sie auf **Extras » Spatial Viewer** und führen folgende Abfrage aus:

```
SELECT ST_Geometry::ST_UnionAggr(proj_geometry)
FROM Massdata
WHERE proj_geometry.ST_Touches( @Mass_01775.ST_Transform( 3586 ) ) = 1;
```

Doppelklicken Sie auf das Ergebnis, um es anzuzeigen.

Wenn Sie einen Fehler erhalten, der darauf hinweist, dass die vollständige Spalte nicht aus der Datenbank gelesen werden konnte, erhöhen Sie die Einstellung **Kürzungslänge** für Interactive SQL. Hierzu klicken Sie in Interactive SQL auf **Extras » Optionen » SQL Anywhere** und setzen Sie **Kürzungslänge** auf einen höheren Wert. Führen Sie die Abfrage nochmals aus und sehen Sie sich die Geometrie an.

Ergebnisse

Sie haben die praktische Einführung abgeschlossen.

Nächste Schritte

Stellen Sie den ursprünglichen Zustand der Beispieldatenbank (demo.db) mithilfe der Schritte unter „[Neuerstellung der Beispieldatenbank \(demo.db\)](#)“ [*SQL Anywhere 16 - Einführung*] wieder her.

Siehe auch

- [ST_Transform-Methode für den ST_Geometry-Datentyp auf Seite 237](#)
- [ST_Touches-Methode für den ST_Geometry-Datentyp auf Seite 236](#)
- [ST_UnionAggr-Methode für Typ ST_Geometry auf Seite 240](#)

Zugriff auf räumliche Daten und ihre Verarbeitung

In diesem Abschnitt werden die Typen, Methoden und Konstruktoren beschrieben, die Sie verwenden können, um auf räumliche Daten zuzugreifen, sie zu bearbeiten und zu analysieren. Die räumlichen Datentypen können wie Datentypen oder Klassen gesehen werden. Jeder räumliche Datentyp hat zugeordnete Methoden und Konstruktoren, die Sie für den Zugriff auf die Daten verwenden.

Eine detaillierte Beschreibung der räumlichen Datentypen finden Sie unter „[Unterstützte räumliche Datentypen und ihre Hierarchie](#)“ auf Seite 7.

Aus Gründen der Kompatibilität mit anderen Produkten unterstützt SQL Anywhere auch einige SQL-Funktionen für die Arbeit mit räumlichen Daten. In fast allen Fällen verwenden diese Kompatibilitätsfunktionen eine der räumlichen Methoden für die Durchführung des Vorgangs, daher wird ein Link zur Basismethode bereitgestellt. Siehe „[Funktionen der räumlichen Kompatibilität](#)“ auf Seite 343.

ST_CircularString-Datentyp

Der ST_CircularString-Datentyp ist ein untergeordneter Typ von ST_Curve, der Kreisbogensegmente zwischen Kontrollpunkten verwendet.

Direkt übergeordneter Typ

- „[ST_Curve-Datentyp](#)“

Konstruktor

- „[ST_CircularString-Konstruktor](#)“

Methoden

- Methoden von ST_CircularString:

ST_NumPoints	ST_PointN
------------------------------	---------------------------

- Alle Methoden von „[ST_Curve-Datentyp](#)“ können auch für einen ST_CircularString-Datentyp aufgerufen werden:

ST_CurveToLine	ST_EndPoint	ST_IsClosed	ST_IsRing
ST_Length	ST_StartPoint		

- Alle Methoden von „[ST_Geometry-Datentyp](#)“ können auch für einen ST_CircularString-Datentyp aufgerufen werden:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAgr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr
ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType
ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth
ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBaseType	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine
ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Bemerkungen

Der ST_CircularString-Datentyp ist ein untergeordneter Typ von ST_Curve, der Kreisbogensegmente zwischen Kontrollpunkten verwendet. Die ersten drei Punkte definieren ein Segment wie folgt. Der erste Punkt ist der Startpunkt des Segments. Der zweite Punkt ist ein beliebiger Punkt des Segments, ausgenommen Startpunkt und Endpunkt. Der dritte Punkt ist der Endpunkt des Segments. Nachfolgende Segmente werden nur durch zwei Punkte (Zwischenpunkt und Endpunkt) definiert. Der Startpunkt wird als Endpunkt des vorherigen Segments genommen.

Eine Kreisbogenfolge kann ein kompletter Kreisbogen mit drei Punkten sein, wenn Start- und Endpunkt übereinstimmen. In diesem Fall ist der Zwischenpunkt der Mittelpunkt des Segments.

Wenn Start-, Zwischen- und Endpunkt kollinear sind, ist das Segment ein gerades Liniensegment zwischen dem Start- und Endpunkt.

Eine Kreisbogenfolge mit genau drei Punkten ist ein Kreisbogen. Ein Kreisring ist eine Kreisbogenfolge, die sowohl geschlossen als auch einfach ist.

Kreisbogenfolgen sind in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht zulässig. Der Versuch, eine Kreisbogenfolge für SRID 4326 zu erstellen, führt zu einer Fehlermeldung.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.3

ST_CircularString-Konstruktor

Konstruiert eine Kreisbogenfolge.

Überladungsliste

Name	Beschreibung
„ST_CircularString()-Konstruktor“	Konstruiert eine Kreisbogenfolge, die die leere Menge darstellt.
„ST_CircularString(LONG VARCHAR[, INT])-Konstruktor“	Konstruiert eine Kreisbogenfolge aus einer Textdarstellung.
„ST_CircularString(LONG BINARY[, INT])-Konstruktor“	Konstruiert eine Kreisbogenfolge aus einem Well-Known-Binary-Wert (WKB).
„ST_CircularString(ST_Point,ST_Point,ST_Point,...)-Konstruktor“	Konstruiert einen Kreisbogenfolgenwert aus einer Liste von Punkten in einem angegebenen räumlichen Bezugssystem.

ST_CircularString()-Konstruktor

Konstruiert eine Kreisbogenfolge, die die leere Menge darstellt.

Syntax

NEW ST_CircularString()

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Standardfunktion

Beispiel

Der folgende Code gibt 1 zurück und zeigt damit an, dass der Wert leer ist.

```
SELECT NEW ST_CircularString().ST_IsEmpty()
```

ST_CircularString(LONG VARCHAR[, INT])-Konstruktor

Konstruiert eine Kreisbogenfolge aus einer Textdarstellung.

Syntax

NEW ST_CircularString(*text-representation*[, *srid*])

Parameter

Name	Typ	Beschreibung
text-representation	LONG VARCHAR	Eine Zeichenfolge, die die Textdarstellung einer Kreisbogenfolge enthält. Die Eingabe kann in jedem unterstützten Texteingabeformat erfolgen, einschließlich Well-Known-Text (WKT) oder Extended-Well-Known-Text (EWKT).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert eine Kreisbogenfolge aus einer Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.3.2

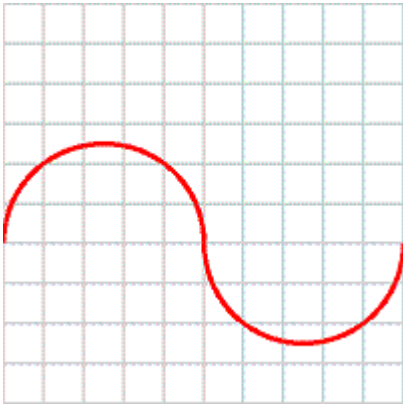
Beispiel

Die folgende Anweisung gibt CircularString (5 10, 10 12, 15 10) zurück.

```
SELECT NEW ST_CircularString('CircularString (5 10, 10 12, 15 10)')
```

Das folgende Beispiel zeigt eine Kreisbogenfolge mit zwei halbkreisförmigen Segmenten.

```
SELECT NEW ST_CircularString('CircularString (0 4, 2.5 6.5, 5 4, 7.5 1.5, 10 4)') CS
```



ST_CircularString(LONG BINARY[, INT])-Konstruktor

Konstruiert eine Kreisbogenfolge aus einem Well-Known-Binary-Wert (WKB).

Syntax

NEW ST_CircularString(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Eine Zeichenfolge mit der binären Darstellung einer Kreisbogenfolge. Die Eingabe kann in jedem unterstützten binären Eingabeformat erfolgen, einschließlich Well-Known-Binary (WKB) oder Extended-Well-Known-Binary (EWKB).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert eine Kreisbogenfolge aus einer binären Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.3.2

Beispiel

Die folgende Anweisung gibt CircularString (5 10, 10 12, 15 10) zurück.

```
SELECT NEW
ST_CircularString(0x010800000003000000000000000000000144000000000000024400000000
00000244000000000000000028400000000000002e400000000000002440)
```

ST_CircularString(ST_Point,ST_Point,ST_Point,...)-Konstruktor

Konstruiert einen Kreisbogenfolgenwert aus einer Liste von Punkten in einem angegebenen räumlichen Bezugssystem.

Syntax

NEW ST_CircularString(*pt1,pt2,pt3[,pt4,...,ptN]*)

Parameter

Name	Typ	Beschreibung
pt1	ST_Point	Der erste Punkt eines Segments.
pt2	ST_Point	Jeder Punkt auf dem Segment zwischen dem ersten und dem letzten Punkt.
pt3	ST_Point	Der letzte Punkt eines Segments.
pt4,...,ptN	ST_Point	Zusätzliche Punkte, die weitere Segmente definieren, die jeweils mit dem vorherigen Endpunkt beginnen, durch den ersten zusätzlichen Punkt gehen und mit dem zweiten zusätzlichen Punkt enden.

Bemerkungen

Konstruiert einen Kreisbogenfolgenwert aus einer Liste von Punkten. Mindestens drei Punkte müssen angegeben werden. Der erste dieser drei Punkte ist der Startpunkt eines Segments, der dritte Punkt ist das Ende des Segments und der zweite Punkt ist ein beliebiger Punkt auf dem Segment zwischen dem ersten und dritten Punkt. Weitere Punkte können in Paaren angegeben werden, um der Kreisbogenfolge weitere Segmente hinzuzufügen. Alle angegebenen Punkte müssen dieselbe SRID aufweisen. Die sich daraus ergebende Kreisbogenfolge wird mit dieser gemeinsamen SRID erstellt. Die angegebenen Punkte dürfen nicht leer sein und müssen dieselbe Antwort für Is3D und IsMeasured aufweisen. Die Kreisbogenfolge ist 3D, wenn alle Punkte 3D sind. Die Kreisbogenfolge wird gemessen, wenn alle Punkte gemessen werden.

Hinweis

Standardmäßig verwendet ST_CircularString das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Der folgende Code gibt einen Fehler zurück: Mindestens drei Punkte müssen angegeben werden.

```
SELECT NEW ST_CircularString( NEW ST_Point( 0, 0 ), NEW ST_Point( 1, 1 ) )
```

Das folgende Beispiel gibt den Wert CircularString (0 0, 1 1, 2 0) zurück.

```
SELECT NEW ST_CircularString( NEW ST_Point( 0, 0 ), NEW ST_Point( 1, 1 ),
NEW ST_Point(2,0) )
```

Der folgende Code gibt einen Fehler zurück: Das erste Segment hat drei Punkte, die nachfolgenden Segmente haben zwei.

```
SELECT NEW ST_CircularString( NEW ST_Point( 0, 0 ), NEW ST_Point( 1, 1 ),
NEW ST_Point(2,0), NEW ST_Point(1,-1) )
```

Das folgende Beispiel gibt den Wert CircularString (0 0, 1 1, 2 0, 1 -1, 0 0) zurück.

```
SELECT NEW ST_CircularString( NEW ST_Point( 0, 0 ), NEW ST_Point( 1, 1 ),
NEW ST_Point(2,0), NEW ST_Point(1,-1), NEW ST_Point( 0, 0 ) )
```

ST_NumPoints-Methode

Gibt die Anzahl der Punkte zurück, die die Kreisbogenfolge definieren.

Hinweis

Standardmäßig verwendet ST_NumPoints das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

circularstring-expression.ST_NumPoints()

Rückgabe

- **INT** Gibt NULL zurück, wenn der Kreisbogenfolgenwert leer ist, sonst die Anzahl der Punkte im Wert.

Siehe auch

- [ST_PointN-Methode](#) für den ST_CircularString-Datentyp
- [ST_NumPoints-Methode](#) für den ST_LineString-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.3.4

Beispiel

Das folgende Beispiel gibt den Wert 5 zurück.

```
SELECT TREAT( Shape AS ST_CircularString ).ST_NumPoints()
FROM SpatialShapes WHERE ShapeID = 18
```

ST_PointN-Methode

Gibt den *n*-ten Punkt in der Kreisbogenfolge zurück.

Hinweis

Standardmäßig verwendet ST_PointN das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

circularstring-expression.ST_PointN(*n*)

Parameter

Name	Typ	Beschreibung
n	INT	Die Position des zurückzugebenden Elements, von 1 bis <i>circularstring-expression</i> .ST_NumPoints().

Rückgabe

- ST_Point** Wenn der Wert des Kreisbogenfolgenausdrucks *circular-expression* eine leere Menge ist, wird NULL zurückgegeben. Wenn die angegebene Position *n* kleiner als 1 oder größer als die Anzahl der Punkte ist, wird NULL zurückgegeben. Andernfalls wird der ST_Point-Wert an der Position *n* zurückgegeben.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *circularstring-expression*.

Siehe auch

- [ST_NumPoints-Methode für den ST_CircularString-Datentyp](#)
- [ST_PointN-Methode für den ST_LineString-Datentyp](#)

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006)** 7.3.5

Beispiel

Das folgende Beispiel gibt den Wert Point (2 0) zurück.

```
SELECT TREAT( Shape AS ST_CircularString ).ST_PointN( 3 )
FROM SpatialShapes WHERE ShapeID = 18
```

Das folgende Beispiel gibt eine Zeile für jeden Punkt in der Geometrie zurück.

```
BEGIN
  DECLARE geom ST_CircularString;
  SET geom = NEW ST_CircularString( 'CircularString( 0 0, 1 1, 2 0 )' );
  SELECT row_num, geom.ST_PointN( row_num )
    FROM sa_rowgenerator( 1, geom.ST_NumPoints() )
    ORDER BY row_num;
END
```

Die Abfrage erzeugt die folgende Ergebnismenge:

row_num	geom.ST_PointN(row_num)
1	Point (0 0)
2	Point (1 1)
3	Point (2 0)

ST_CompoundCurve-Datentyp

Eine Verbundkurve ist eine Sequenz von ST_Curve-Werten, sodass angrenzende Kurven an ihren Endpunkten verknüpft werden. Die daran beteiligten Kurven sind auf ST_LineString und ST_CircularString begrenzt. Der Startpunkt jeder nach der ersten Kurve kommenden Kurve stimmt jeweils mit dem Endpunkt der vorherigen Kurve überein.

Direkt übergeordneter Typ

- „ST_Curve-Datentyp“

Konstruktor

- „ST_CompoundCurve-Konstruktor“

Methoden

- Methoden von ST_CompoundCurve:

ST_CurveN	ST_NumCurves
-----------	--------------

- Alle Methoden von „ST_Curve-Datentyp“ können Sie auch für einen ST_CompoundCurve-Datentyp aufrufen:

ST_CurveToLine	ST_EndPoint	ST_IsClosed	ST_IsRing
ST_Length	ST_StartPoint		

- Alle Methoden von „ST_Geometry-Datentyp“ können Sie auch für einen ST_CompoundCurve-Datentyp aufrufen:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAggr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr

ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType
ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth
ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBase-Type	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine
ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 7.4

ST_CompoundCurve-Konstruktor

Konstruiert eine Verbundkurve.

Überladungsliste

Name	Beschreibung
„ST_CompoundCurve()-Konstruktor“	Konstruiert eine Verbundkurve, die die leere Menge darstellt.
„ST_CompoundCurve(LONG VARCHAR[, INT])-Konstruktor“	Konstruiert eine Verbundkurve aus einer Textdarstellung.
„ST_CompoundCurve(LONG BINARY[, INT])-Konstruktor“	Konstruiert eine Kreisbogenfolge aus einem Well-Known-Binary-Wert (WKB).
„ST_CompoundCurve(ST_Curve,...)-Konstruktor“	Konstruiert eine Verbundkurve aus einer Liste von Kurven.

ST_CompoundCurve()-Konstruktor

Konstruiert eine Verbundkurve, die die leere Menge darstellt.

Syntax

NEW ST_CompoundCurve()

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Standardfunktion

Beispiel

Der folgende Code gibt 1 zurück und zeigt damit an, dass der Wert leer ist.

```
SELECT NEW ST_CompoundCurve().ST_IsEmpty()
```

ST_CompoundCurve(LONG VARCHAR[, INT])-Konstruktor

Konstruiert eine Verbundkurve aus einer Textdarstellung.

Syntax

NEW ST_CompoundCurve(text-representation[, srid])

Parameter

Name	Typ	Beschreibung
text-representation	LONG VARCHAR	Eine Zeichenfolge mit der Textdarstellung einer Verbundkurve. Die Eingabe kann in jedem unterstützten Texteingabeformat erfolgen, einschließlich Well-Known-Text (WKT) oder Extended-Well-Known-Text (EWKT).

Name	Typ	Beschreibung
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert eine Verbundkurve aus einer Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.4.2

Beispiel

Die folgende Anweisung gibt CompoundCurve ((0 0, 5 10), CircularString (5 10, 10 12, 15 10)) zurück.

```
SELECT NEW ST_CompoundCurve('CompoundCurve ((0 0, 5 10), CircularString (5 10, 10 12, 15 10))')
```

ST_CompoundCurve(LONG BINARY[, INT])-Konstruktor

Konstruiert eine Kreisbogenfolge aus einem Well-Known-Binary-Wert (WKB).

Syntax

NEW ST_CompoundCurve(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Eine Zeichenfolge mit der binären Darstellung einer Verbundkurve. Die Eingabe kann in jedem unterstützten binären Eingabeformat erfolgen, einschließlich Well-Known-Binary-Datentypen (WKB) oder Extended-Well-Known-Binary-Datentypen (EWKB).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert eine Verbundkurve aus einer binären Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.4.2

Beispiel

Die folgende Anweisung gibt CompoundCurve ((0 0, 5 10)) zurück.

Konstruiert eine Verbundkurve aus einer Liste von Kurven.

NEW ST_CompoundCurve(*curve1*[,*curve2*,...,*curveN*])

Name	Typ	Beschreibung
------	-----	--------------

Konstruiert eine Verbundkurve aus einer Liste von daran teilnehmenden Kurven. Der Startpunkt jeder

Standardmäßig verwendet ST_CompoundCurve das Originalformat für eine Geometrie, wenn es

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Die folgende Anweisung gibt CompoundCurve ((0 0, 5 10), CircularString (5 10, 10 12, 15 10)) zurück.

```
SELECT NEW ST_CompoundCurve(NEW ST_LineString( 'LineString(0 0 5 10)') NEW
```

Gibt die n -te Kurve in der Verbundkurve zurück.

Hinweis

Standardmäßig verwendet ST_CurveN das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

compoundcurve-expression.ST_CurveN(*n*)

Parameter

Name	Typ	Beschreibung
n	INT	Die Position des zurückzugebenden Elements, von 1 bis <i>compoundcurve-expression</i> .ST_NumInteriorRing().

Rückgabe

- **ST_Curve** Gibt die *n*-te Kurve in der Verbundkurve zurück.
- Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *compoundcurve-expression*.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.4.5

Beispiel

Das folgende Beispiel gibt den Wert CircularString (0 0, 1 1, 2 0) zurück.

```
SELECT TREAT( Shape AS ST_CompoundCurve ).ST_CurveN( 1 )
FROM SpatialShapes WHERE ShapeID = 17
```

ST_NumCurves-Methode

Gibt die Anzahl der Kurven zurück, die die Definition der Verbundkurve definieren.

Hinweis

Standardmäßig verwendet ST_NumCurves das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

compoundcurve-expression.ST_NumCurves()

Rückgabe

- **INT** Gibt die Anzahl der Kurven zurück, die in der Verbundkurve enthalten sind.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.4.4

Beispiel

Das folgende Beispiel gibt den Wert 2 zurück.

```
SELECT TREAT( Shape AS ST_CompoundCurve ).ST_NumCurves()
FROM SpatialShapes WHERE ShapeID = 17
```

ST_Curve-Datentyp

Der ST_Curve-Datentyp ist ein übergeordneter Datentyp für Datentypen, die Linien mit einer Reihe von Punkten darstellen.

Direkt übergeordneter Typ

- „ST_Geometry-Datentyp“

Direkt untergeordnete Typen

- „ST_CircularString-Datentyp“
- „ST_CompoundCurve-Datentyp“
- „ST_LineString-Datentyp“

Methoden

- Methoden von ST_Curve:

ST_CurveToLine	ST_EndPoint	ST_IsClosed	ST_IsRing
ST_Length	ST_StartPoint		

- Alle Methoden von „ST_Geometry-Datentyp“ können Sie auch für einen aufgerufenen ST_Curve-Datentyp verwenden:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAggr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr
ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers

ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType
ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth
ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBaseType	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine
ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Bemerkungen

Der ST_Curve-Datentyp ist ein übergeordneter Datentyp für Datentypen, die Linien mit einer Reihe von Punkten darstellen. Untergeordnete Typen geben an, ob die Kontrollpunkte durch gerade Segmente (ST_LineString), Kreisbogensegmente (ST_CircularString) oder eine Kombination aus beiden (ST_CompoundCurve) verbunden werden.

Der ST_Curve-Datentyp ist nicht instanzierbar.

Ein ST_Curve-Wert ist einfach, wenn er keine Schnittmenge mit sich selbst hat (gegebenenfalls ausgenommen an seinen Endpunkten). Wenn ein ST_Curve-Wert an seinen Endpunkten eine Schnittmenge hat, ist er geschlossen. Ein ST_Curve-Wert, der sowohl einfach als auch geschlossen ist, wird als Ring bezeichnet.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.1

ST_CurveToLine-Methode

Gibt die ST_LineString-Interpolation für einen ST_Curve-Wert zurück.

Syntax

curve-expression.ST_CurveToLine()

Rückgabe

- **ST_LineString** Gibt die ST_LineString-Datentyp-Interpolation von *curve-expression* zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *curve-expression*.

Bemerkungen

Wenn *curve-expression* leer ist, gibt die ST_CurveToLine-Methode eine leere Menge vom Typ ST_LineString zurück. Andernfalls gibt ST_CurveToLine eine Linienfolge zurück, die alle Linienfolgen in *curve-expression* kombiniert mit der Interpolation aller Kreisbogenfolgen in *curve-expression* enthält.

Hinweis

Standardmäßig verwendet ST_CurveToLine das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToLineString-Methode für den ST_Geometry-Datentyp](#)
- [„Auswirkungen der Interpolation auf räumliche Berechnungen“](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.1.7

Beispiel

Das folgende Beispiel gibt das Ergebnis LineString (0 7, 0 4, 4 4) zurück (eine Kopie der ursprünglichen Linienfolge).

```
SELECT TREAT( Shape AS ST_Curve ).ST_CurveToLine()
FROM SpatialShapes WHERE ShapeID = 5
```

Das folgende Beispiel gibt das Ergebnis `LineString (0 0, 5 10)` zurück (die in eine entsprechenden Linienfolge konvertierte Verbundkurve).

```
SELECT NEW ST_CompoundCurve( 'CompoundCurve((0 0, 5 10))' ).ST_CurveToLine()
```

Die folgende Anweisung gibt eine interpolierte Linienfolge zurück, die annähernd der ursprünglichen Kreisbogenfolge entspricht.

```
SELECT TREAT( Shape AS ST_Curve ).ST_CurveToLine()  
FROM SpatialShapes WHERE ShapeID = 19
```

ST_EndPoint-Methode

Gibt einen `ST_Point`-Wert zurück, der der Endpunkt der `ST_Curve`-Kurve ist.

Hinweis

Standardmäßig verwendet `ST_End Point` das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

```
curve-expression.ST_EndPoint()
```

Rückgabe

- **ST_Point** Wenn die Kurve eine leere Menge ist, wird `NULL` zurückgegeben. Andernfalls wird der Endpunkt der Kurve zurückgegeben.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *curve-expression*.

Siehe auch

- [ST_StartPoint-Methode für den ST_Curve-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.1.4

Beispiel

Das folgende Beispiel gibt den Wert `Point (5 10)` zurück.

```
SELECT NEW ST_LineString( 'LineString(0 0, 5 5, 5 10)' ).ST_EndPoint()
```

ST_IsClosed-Methode

Testet, ob der `ST_Curve`-Wert geschlossen ist. Eine Kurve ist geschlossen, wenn der Start- und der Endpunkt zusammenfallen.

Hinweis

Standardmäßig verwendet ST_IsClosed das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

curve-expression.ST_IsClosed()

Rückgabe

- **BIT** Gibt 1 zurück, wenn die Kurve geschlossen (und nicht leer) ist. Andernfalls wird 0 zurückgegeben.

Siehe auch

- [ST_IsClosed-Methode für den ST_MultiCurve-Datentyp](#)
- [ST_IsRing-Methode für den ST_Curve-Datentyp](#)
- [ST_Boundary-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.1.5

Beispiel

Der folgende Code gibt alle Zeilen in SpatialShapes zurück, die geschlossene Kurven enthalten. Der IF-Ausdruck ist erforderlich, um zu gewährleisten, dass die TREAT-Funktion nicht ausgeführt wird, wenn die Form kein untergeordneter Datentyp von ST_Curve ist. Ohne den IF-Ausdruck kann der Server die Bedingungen in der WHERE-Klausel neu sortieren, was zu einem Fehler führen kann.

```
SELECT * FROM SpatialShapes
WHERE IF Shape IS OF ( ST_Curve )
      AND TREAT( Shape AS ST_Curve ).ST_IsClosed() = 1 THEN 1 ENDIF = 1
```

Der folgende Code gibt alle Zeilen in curve_table zurück, die geschlossene Geometrien haben. In diesem Beispiel wird davon ausgegangen, dass die Geometriespalte vom Typ ST_Curve, ST_LineString, ST_CircularString oder ST_CompoundCurve ist.

```
SELECT * FROM curve_table WHERE geometry.ST_IsClosed() = 1
```

ST_IsRing-Methode

Testet, ob der ST_Curve-Wert ein Ring ist. Eine Kurve ist ein Ring, wenn sie geschlossen und einfach ist (keine Schnittmengen mit sich selbst).

Syntax

curve-expression.ST_IsRing()

Rückgabe

- **BIT** Gibt 1 zurück, wenn die Kurve ein Ring (und nicht leer) ist. Andernfalls wird 0 zurückgegeben.

Siehe auch

- [ST_IsClosed-Methode für den ST_Curve-Datentyp](#)
- [ST_IsSimple-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.1.6

Beispiel

Der folgende Code gibt alle Zeilen in SpatialShapes zurück, die Ringe enthalten. Der IF-Ausdruck ist erforderlich, um zu gewährleisten, dass die TREAT-Funktion nicht ausgeführt wird, wenn die Form kein untergeordneter Datentyp von ST_Curve ist. Ohne den IF-Ausdruck kann der Server die Bedingungen in der WHERE-Klausel neu sortieren, was zu einem Fehler führen kann.

```
SELECT * FROM SpatialShapes
WHERE IF Shape IS OF ( ST_Curve )
      AND TREAT( Shape AS ST_Curve ).ST_IsRing() = 1 THEN 1 ENDIF = 1
```

Der folgende Code gibt alle Zeilen in curve_table zurück, die Geometrien mit Ringen enthalten. In diesem Beispiel wird davon ausgegangen, dass die Geometriespalte vom Typ ST_Curve, ST_LineString, ST_CircularString oder ST_CompoundCurve ist.

```
SELECT * FROM curve_table WHERE geometry.ST_IsRing() = 1
```

ST_Length-Methode

Gibt die Längenmessung des ST_Curve-Werts zurück. Das Ergebnis wird mit den Einheiten gemessen, die durch den Einheitname-Parameter (unit-name) festgelegt sind.

Syntax

```
curve-expression.ST_Length([ unit-name])
```

Parameter

Name	Typ	Beschreibung
unit-name	VAR-CHAR(128)	Die Einheiten, in denen die Länge berechnet werden soll. Ist standardmäßig die Einheit des räumlichen Bezugssystems. Der unit-name-Parameter muss mit der UNIT_NAME-Spalte einer Zeile in der ST_UNITS_OF_MEASURE-Ansicht übereinstimmen, wo UNIT_TYPE auf 'LINEAR' eingestellt ist.

Rückgabe

- **DOUBLE** Wenn die Kurve eine leere Menge ist, wird NULL zurückgegeben. Andernfalls wird die Länge der Kurve in den angegebenen Einheiten zurückgegeben.

Bemerkungen

Die ST_Length-Methode gibt die Länge einer Kurve in den Einheiten zurück, die durch den *unit-name*-Parameter festgelegt sind. Wenn die Kurve leer ist, wird NULL zurückgegeben.

Wenn die Kurve Z-Werte enthält, werden diese beim Berechnen der Länge der Geometrie nicht in Betracht gezogen.

Hinweis

Wenn *curve-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_LENGTH das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_Length-Methode für den ST_MultiCurve-Datentyp](#)
- [ST_Area-Methode für den ST_Surface-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.1.2

Beispiel

Das folgende Beispiel gibt den Wert 2 zurück.

```
SELECT NEW ST_LineString( 'LineString(1 0, 1 1, 2 1)' ).ST_Length()
```

Mit dem folgenden Beispiel wird eine Kreisbogenfolge für einen Halbkreis erstellt und ST_Length für die Suche nach der Länge der Geometrie verwendet. Der Wert PI wird zurückgegeben.

```
SELECT NEW ST_CircularString( 'CircularString( 0 0, 1 1, 2 0 )' ).ST_Length()
```

Mit dem folgenden Beispiel wird eine Linienfolge erstellt, die einen Pfad von Halifax nach Waterloo in Kanada darstellt und ST_Length für die Suche nach der Länge des Pfads in Metern verwendet. Als Ergebnis wird 1361967.76789 zurückgegeben

```
SELECT NEW ST_LineString( 'LineString( -63.573566 44.646244, -80.522372  
43.465187 )', 4326 )  
.ST_Length()
```

Das folgende Beispiel gibt die Länge der Kurven in der SpatialShapes-Tabelle zurück. Die Längen werden in kartesischen Einheiten zurückgegeben.

```
SELECT ShapeID, TREAT( Shape AS ST_Curve ).ST_Length()  
FROM SpatialShapes WHERE Shape IS OF ( ST_Curve )
```

Mit dem folgenden Beispiel wird eine Linienfolge und eine Beispielmäßeinheit (example_unit_halfmetre) erstellt. Die ST_Length-Methode findet die Länge des Geometrie in dieser Maßeinheit. Der Wert 4.0 wird zurückgegeben.

```
BEGIN
  DECLARE @curve ST_Curve;
  CREATE SPATIAL UNIT OF MEASURE IF NOT EXISTS "example_unit_halfmetre"
  TYPE LINEAR CONVERT USING .5;
  SET @curve = NEW ST_LineString( 'LineString(1 0, 1 1, 2 1)' );
  SELECT @curve.ST_Length( 'example_unit_halfmetre' );
END
```

ST_StartPoint-Methode

Gibt einen ST_Point-Wert zurück, der der Startpunkt des ST_Curve-Werts ist.

Hinweis

Standardmäßig verwendet ST_StartPoint das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

curve-expression.ST_StartPoint()

Rückgabe

- **ST_Point** Wenn die Kurve eine leere Menge ist, wird NULL zurückgegeben. Andernfalls wird der Startpunkt der Kurve zurückgegeben.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *curve-expression*.

Siehe auch

- [ST_EndPoint-Methode für den ST_Curve-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.1.3

Beispiel

Das folgende Beispiel gibt den Wert Point (0 0) zurück.

```
SELECT NEW ST_LineString( 'LineString(0 0, 5 5, 5 10)' ).ST_StartPoint()
```

ST_CurvePolygon-Datentyp

Ein ST_CurvePolygon steht für eine planare Oberfläche, die durch einen äußeren Ring und null oder mehreren inneren Ringen definiert wird.

Direkt übergeordneter Typ

- „ST_Surface-Datentyp“

Direkt untergeordnete Typen

- „ST_Polygon-Datentyp“

Konstruktor

- „ST_CurvePolygon-Konstruktor“

Methoden

- Methoden von ST_CurvePolygon:

ST_CurvePolyToPoly	ST_ExteriorRing	ST_InteriorRingN	ST_NumInteriorRing
--------------------	-----------------	------------------	--------------------

- Alle Methoden von „ST_Surface-Datentyp“ können Sie auch für einen ST_CurvePolygon-Datentyp aufrufen:

ST_Area	ST_Centroid	ST_IsWorld	ST_Perimeter
ST_PointOnSurface			

- Alle Methoden von „ST_Geometry-Datentyp“ können Sie auch für einen ST_CurvePolygon-Datentyp aufrufen:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAggr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr
ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType
ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth

ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBase-Type	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine
ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Bemerkungen

Der ST_CurvePolygon-Datentyp stellt eine planare Fläche dar, die durch einen äußeren Ring und null oder mehr innere Ringe definiert wird, die Löcher in der Fläche darstellen. Der externe und die internen Ringe eines ST_CurvePolygon-Datentyps können jeden ST_Curve-Wert annehmen. Beispielsweise ist ein Kreis ein ST_CurvePolygon mit einem äußeren ST_CircularString-Ring, der die Begrenzung darstellt. In einem ST_CurvePolygon können zwei Ringe keine Schnittmenge aufweisen, außer möglicherweise an einem einzigen Punkt. Außerdem kann ein ST_CurvePolygon keine abgeschnittenen Linien, Spitzen oder Durchstiche aufweisen.

Das Innere jedes ST_CurvePolygon ist eine Menge aus verbundenen Punkten.

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 8.2

ST_CurvePolygon-Konstruktor

Konstruiert ein Kurvenpolygon.

Überladungsliste

Name	Beschreibung
„ST_CurvePolygon()-Konstruktor“	Konstruiert ein Kurvenpolygon, das eine leere Menge darstellt.
„ST_CurvePolygon(LONG VARCHAR[, INT])-Konstruktor“	Konstruiert ein Kurvenpolygon aus einer Textdarstellung.
„ST_CurvePolygon(LONG BINARY[, INT])-Konstruktor“	Konstruiert ein Kurvenpolygon aus einem Well-Known-Binary-Wert (WKB).
„ST_CurvePolygon(ST_Curve,...)-Konstruktor“	Erstellt ein Kurvenpolygon aus einer Kurve, die den äußeren Ring darstellt, und einer Liste von Kurven, die die inneren Ringe darstellen, alle in einem festgelegten räumlichen Bezugssystem.
„ST_CurvePolygon(ST_MultiCurve[, VARCHAR(128)])-Konstruktor“	Erstellt ein Kurvenpolygon aus einer Mehrfachkurve mit einem äußeren Ring und einer optionalen Liste von inneren Ringen.

ST_CurvePolygon()-Konstruktor

Konstruiert ein Kurvenpolygon, das eine leere Menge darstellt.

Syntax

NEW ST_CurvePolygon()

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Standardfunktion

Beispiel

Der folgende Code gibt 1 zurück und zeigt damit an, dass der Wert leer ist.

```
SELECT NEW ST_CurvePolygon().ST_IsEmpty()
```

ST_CurvePolygon(LONG VARCHAR[, INT])-Konstruktor

Konstruiert ein Kurvenpolygon aus einer Textdarstellung.

Syntax

NEW ST_CurvePolygon(*text-representation*[, *srid*])

Parameter

Name	Typ	Beschreibung
text-representation	LONG VAR- CHAR	Eine Zeichenfolge mit der Textdarstellung eines Kurvenpolygons. Die Eingabe kann in jedem unterstützten Texteingabeformat erfolgen, einschließlich Well-Known-Text-Datentypen (WKT) oder Extended-Well-Known-Text-Datentypen (EWKT).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert ein Kurvenpolygon aus einer Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.2.2

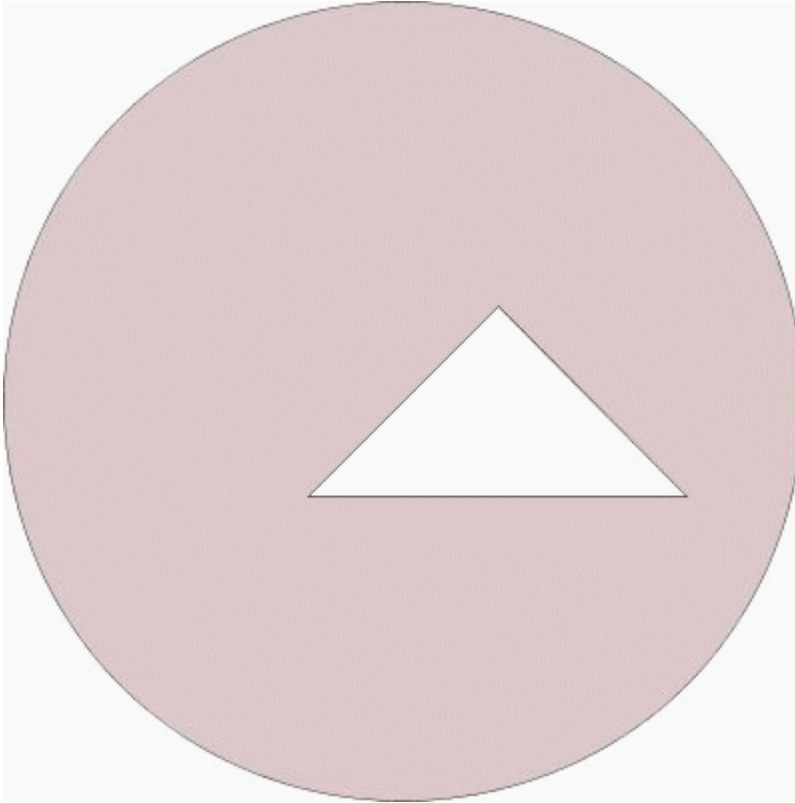
Beispiel

Die folgende Anweisung gibt CurvePolygon (CompoundCurve (CircularString (-5 -5, 0 -5, 5 -5), (5 -5, 0 5, -5 -5))) zurück.

```
SELECT NEW ST_CurvePolygon('CurvePolygon (CompoundCurve (CircularString (-5  
-5, 0 -5, 5 -5), (5 -5, 0 5, -5 -5)))')
```

Das folgende Beispiel zeigt ein Kurvenpolygon mit einem Kreis als äußerem Ring und einem Dreieck als innerem Ring.

```
SELECT NEW ST_CurvePolygon('CurvePolygon ( CircularString (2 0, 5 3, 2 0),  
(3 1, 4 2, 5 1, 3 1) )') cpoly
```



ST_CurvePolygon(LONG BINARY[, INT])-Konstruktor

Konstruiert ein Kurvenpolygon aus einem Well-Known-Binary-Wert (WKB).

Syntax

NEW ST_CurvePolygon(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Eine Zeichenfolge mit der binären Darstellung eines Kurvenpolygons. Die Eingabe kann in jedem unterstützten binären Eingabeformat erfolgen, einschließlich Well-Known-Binary-Datentypen (WKB) oder Extended-Well-Known-Binary-Datentypen (EWKB).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.2.2

Beispiel

Der folgende Code gibt CurvePolygon ((-5 -1, 5 -1, 0 9, -5 -1), CircularString (-2 2, -2 4, 2 4, 2 2, -2 2)) (ein Dreieck mit einem kreisförmigen Loch) zurück.

```
SELECT NEW ST_CurvePolygon(
  NEW ST_LineString ('LineString (-5 -1, 5 -1, 0 9, -5 -1)'),
  NEW ST_CircularString ('CircularString (-2 2, -2 4, 2 4, 2 2, -2 2)'))
```

ST_CurvePolygon(ST_MultiCurve[, VARCHAR(128)])-Konstruktor

Erstellt ein Kurvenpolygon aus einer Mehrfachkurve mit einem äußeren Ring und einer optionalen Liste von inneren Ringen.

Syntax

NEW ST_CurvePolygon(*multi-curve*[, *polygon-format*])

Parameter

Name	Typ	Beschreibung
multi-curve	ST_MultiCurve	Ein Mehrfachkurvenwert mit einen äußeren Ring und (optional) einer Menge von inneren Ringen.
polygon-format	VAR-CHAR(128)	Eine Zeichenfolge mit dem Polygonformat, das beim Interpretieren der angegebenen Kurven verwendet werden soll. Gültige Formate sind 'CounterClockwise', 'Clockwise' und 'EvenOdd'.

Bemerkungen

Erstellt ein Kurvenpolygon aus einer Mehrfachkurve mit einem äußeren Ring und einer optionalen Liste von inneren Ringen.

Wenn dieser Parameter benutzt wird, wählt der *polygon-format*-Parameter den Algorithmus aus, den der Server verwendet, um zu ermitteln, ob ein Ring ein äußerer oder innerer Ring ist. Wenn der Parameter nicht festgelegt ist, wird das Polygonformat des räumlichen Bezugssystems verwendet.

Weitere Hinweise zum *polygon-format* finden Sie unter [POLYGON FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Hinweis

Standardmäßig verwendet ST_CurvePolygon das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Der folgende Code gibt das Ergebnis `CurvePolygon (CircularString (-2 0, 1 -3, 4 0, 1 3, -2 0), (0 0, 1 1, 2 0, 0 0))` (ein Polygon mit einem dreieckigen Loch) zurück.

```
SELECT NEW ST_CurvePolygon( NEW ST_MultiCurve(  
    'MultiCurve(CircularString( -2 0, 4 0, -2 0 ),(0 0, 2 0, 1 1, 0 0 ))' ) )
```

ST_CurvePolyToPoly-Methode

Gibt die Interpolation des Kurvenpolygons als Polygon zurück.

Syntax

curvepolygon-expression.**ST_CurvePolyToPoly()**

Rückgabe

- **ST_Polygon** Gibt die Interpolation von *curvepolygon-expression* als Polygon zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *curvepolygon-expression*.

Bemerkungen

Wenn *curvepolygon-expression* leer ist, gibt die `ST_CurvePolyToPoly`-Methode eine leere Menge vom Typ `ST_Polygon` zurück. Andernfalls gibt `ST_CurvePolyToPoly` ein Polygon zurück, das die linearen Ringe in *curvepolygon-expression* kombiniert mit der Interpolation aller Kreisbogenfolgen oder Verbundkurvenringe in *curvepolygon-expression* enthält.

Hinweis

Standardmäßig verwendet `ST_CurvePolyToPoly` das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Siehe auch

- [ST_ToPolygon-Methode für den ST_Geometry-Datentyp](#)
- [„Auswirkungen der Interpolation auf räumliche Berechnungen“](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.2.7

Beispiel

Das folgende Beispiel gibt das Ergebnis `Polygon ((0 0, 2 0, 1 2, 0 0))` zurück (eine Kopie des ursprünglichen Polygons).

```
SELECT TREAT( Shape AS ST_Polygon ).ST_CurvePolyToPoly()
FROM SpatialShapes WHERE ShapeID = 16
```

Das folgende Beispiel gibt das Ergebnis Polygon ((0 0, 5 0, 5 10, 0 0)) zurück (das Kurvenpolygon konvertiert in ein entsprechendes Polygon).

```
SELECT NEW ST_CurvePolygon( 'CurvePolygon(CompoundCurve((0 0, 5 10, 5 0, 0
0)))' )
        .ST_CurvePolyToPoly()
```

Der folgende Code gibt ein interpoliertes Polygon zurück, das dem ursprünglichen Kurvenpolygon annähernd entspricht.

```
SELECT TREAT( Shape AS ST_CurvePolygon ).ST_CurvePolyToPoly()
FROM SpatialShapes WHERE ShapeId = 24
```

ST_ExteriorRing-Methode

Ruft den äußeren Ring ab oder ändert ihn.

Überladungsliste

Name	Beschreibung
„ST_ExteriorRing()-Methode für den ST_CurvePolygon-Datentyp“	Gibt den äußeren Ring des Kurvenpolygons zurück.
„ST_ExteriorRing(ST_Curve)-Methode für den ST_CurvePolygon-Datentyp“	Ändert den äußeren Ring des Kurvenpolygons.

ST_ExteriorRing()-Methode für den ST_CurvePolygon-Datentyp

Gibt den äußeren Ring des Kurvenpolygons zurück.

Hinweis

Standardmäßig verwendet ST_ExteriorRing das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

curvepolygon-expression.ST_ExteriorRing()

Rückgabe

- **ST_Curve** Gibt den äußeren Ring zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *curvepolygon-expression*.

Siehe auch

- [ST_InteriorRingN-Methode](#) für den [ST_CurvePolygon-Datentyp](#)
- [ST_ExteriorRing-Methode](#) für den [ST_Polygon-Datentyp](#)
- [ST_Boundary-Methode](#) für den [ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.2.3

Beispiel

Das folgende Beispiel gibt den Wert `CircularString (2 0, 5 0, 5 3, 2 3, 2 0)` zurück.

```
SELECT NEW ST_CurvePolygon('CurvePolygon ( CircularString (2 0, 5 3, 2 0),  
(3 1, 4 2, 5 1, 3 1) )')  
      .ST_ExteriorRing()
```

ST_ExteriorRing(ST_Curve)-Methode für den ST_CurvePolygon-Datentyp

Ändert den äußeren Ring des Kurvenpolygons.

Hinweis

Standardmäßig verwendet `ST_ExteriorRing` das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Syntax

```
curvepolygon-expression.ST_ExteriorRing(exterior-ring)
```

Parameter

Name	Typ	Beschreibung
<code>exterior-ring</code>	<code>ST_Curve</code>	Der neue Wert für den äußeren Ring.

Rückgabe

- **ST_CurvePolygon** Gibt eine Kopie des Kurvenpolygonwerts mit dem auf den angegebenen Wert geänderten äußeren Ring zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *curvepolygon-expression*.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.2.3

Beispiel

Das folgende Beispiel gibt den Wert CurvePolygon (CircularString (2 0, 6 1, 5 5, 1 4, 2 0), (3 1, 4 2, 5 1, 3 1)) zurück.

```
SELECT NEW ST_CurvePolygon('CurvePolygon ( CircularString (2 0, 5 3, 2 0),
(3 1, 4 2, 5 1, 3 1) )')
.ST_ExteriorRing( NEW ST_CircularString( 'CircularString (2 0, 5
5, 2 0)' ) )
```

ST_InteriorRingN-Methode

Gibt den n -ten inneren Ring im Kurvenpolygon zurück.

Hinweis

Standardmäßig verwendet ST_InteriorRingN das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

curvepolygon-expression.ST_InteriorRingN(n)

Parameter

Name	Typ	Beschreibung
n	INT	Die Position des zurückzugebenden Elements, von 1 bis <i>curvepolygon-expression</i> .ST_NumInteriorRing().

Rückgabe

- **ST_Curve** Gibt den n -ten inneren Ring im Kurvenpolygon zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *curvepolygon-expression*.

Siehe auch

- [ST_NumInteriorRing-Methode](#) für den ST_CurvePolygon-Datentyp
- [ST_ExteriorRing-Methode](#) für den ST_CurvePolygon-Datentyp
- [ST_InteriorRingN-Methode](#) für den ST_Polygon-Datentyp
- [ST_Boundary-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.2.6

Beispiel

Das folgende Beispiel gibt den Wert LineString (3 1, 4 2, 5 1, 3 1) zurück.

```
SELECT NEW ST_CurvePolygon('CurvePolygon ( CircularString (2 0, 5 3, 2 0),  
(3 1, 4 2, 5 1, 3 1) )')  
      .ST_InteriorRingN( 1 )
```

ST_NumInteriorRing-Methode

Gibt die Anzahl der inneren Ringe im Kurvenpolygon zurück.

Hinweis

Standardmäßig verwendet ST_NumInteriorRing das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Syntax

curvepolygon-expression.ST_NumInteriorRing()

Rückgabe

- **INT** Gibt die Anzahl der inneren Ringe im Kurvenpolygon zurück.

Siehe auch

- [ST_InteriorRingN-Methode für den ST_CurvePolygon-Datentyp](#)
- [ST_InteriorRingN-Methode für den ST_Polygon-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.2.5

Beispiel

Das folgende Beispiel gibt den Wert 1 zurück.

```
SELECT NEW ST_CurvePolygon('CurvePolygon ( CircularString (2 0, 5 3, 2 0),  
(3 1, 4 2, 5 1, 3 1) )')  
      .ST_NumInteriorRing()
```

ST_GeomCollection-Datentyp

ST_GeomCollection ist eine Sammlung von null oder mehr ST_Geometry-Werten.

Direkt übergeordneter Typ

- [„ST_Geometry-Datentyp“](#)

Direkt untergeordnete Typen

- [„ST_MultiCurve-Datentyp“](#)
- [„ST_MultiPoint-Datentyp“](#)
- [„ST_MultiSurface-Datentyp“](#)

Konstruktor

- „ST_GeomCollection-Konstruktor“

Methoden

- Methoden von ST_GeomCollection:

ST_GeomCollectionAggr	ST_GeometryN	ST_NumGeometries
-----------------------	--------------	------------------

- Alle Methoden von „ST_Geometry-Datentyp“ können auch für einen ST_GeomCollection-Datentyp aufgerufen werden:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAggr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr
ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType
ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth
ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBaseType	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly

ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine
ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Bemerkungen

ST_GeomCollection ist eine Sammlung von null oder mehr ST_Geometry-Werten. Alle Werte befinden sich in demselben räumlichen Bezugssystem wie der Sammlungswert. Der ST_GeomCollection-Datentyp kann eine heterogene Sammlung von Objekten (zum Beispiel Punkte, Zeilen und Polygone) enthalten. Untergeordnete Typen von ST_GeomCollection können verwendet werden, um die Sammlung auf bestimmte Geometrietypen zu beschränken.

Die Dimension des Geometriesammlungswerts ist die höchste Dimension seiner Bestandteile.

Eine Geometriesammlung ist einfach, wenn alle ihre Bestandteile einfach sind und keine Bestandteilgeometrien eine Schnittmenge aufweisen, ausgenommen an ihren Begrenzungen.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.1

ST_GeomCollection-Konstruktor

Konstruiert eine Geometriesammlung.

Überladungsliste

Name	Beschreibung
„ST_GeomCollection()-Konstruktor“	Konstruiert eine Geometriesammlung, die die leere Menge darstellt.
„ST_GeomCollection(LONG VARCHAR[, INT])-Konstruktor“	Konstruiert eine Geometriesammlung aus einer Textdarstellung.
„ST_GeomCollection(LONG BINARY[, INT])-Konstruktor“	Konstruiert eine Geometriegruppe aus Well-Known-Binary-Werten (WKB).
„ST_GeomCollection(ST_Geometry,...)-Konstruktor“	Konstruiert eine Geometriesammlung aus einer Liste von Geometriewerten.

ST_GeomCollection()-Konstruktor

Konstruiert eine Geometriesammlung, die die leere Menge darstellt.

Syntax

NEW ST_GeomCollection()

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Standardfunktion

Beispiel

Der folgende Code gibt 1 zurück und zeigt damit an, dass der Wert leer ist.

```
SELECT NEW ST_GeomCollection().ST_IsEmpty()
```

ST_GeomCollection(LONG VARCHAR[, INT])-Konstruktor

Konstruiert eine Geometriesammlung aus einer Textdarstellung.

Syntax

NEW ST_GeomCollection(text-representation[, srid])

Parameter

Name	Typ	Beschreibung
text-representation	LONG VARCHAR	Eine Zeichenfolge mit der Textdarstellung einer Geometriesammlung. Die Eingabe kann in jedem unterstützten Texteingabeformat erfolgen, einschließlich Well-Known-Text (WKT) oder Extended-Well-Known-Text (EWKT).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert eine Geometriesammlung aus einer Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.1.2

Beispiel

Die folgende Anweisung gibt GeometryCollection (CircularString (5 10, 10 12, 15 10), Polygon ((10 -5, 15 5, 5 5, 10 -5))) zurück.

```
SELECT NEW ST_GeomCollection('GeometryCollection (CircularString (5 10, 10 12, 15 10), Polygon ((10 -5, 15 5, 5 5, 10 -5)))')
```

ST_GeomCollection(LONG BINARY[, INT])-Konstruktor

Konstruiert eine Geometriegruppe aus Well-Known-Binary-Werten (WKB).

Syntax

```
NEW ST_GeomCollection(wkb[, srid])
```

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Eine Zeichenfolge mit der binären Darstellung einer Geometriegruppe. Die Eingabe kann in jedem unterstützten binären Eingabeformat erfolgen, einschließlich Well-Known-Binary-Datentypen (WKB) oder Extended-Well-Known-Binary-Datentypen (EWKB).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert eine Geometriesammlung aus einer binären Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 9.1.2

Beispiel

Die folgende Anweisung gibt GeometryCollection (Point (10 20)) zurück.

```
SELECT NEW
ST_GeomCollection(0x010700000001000000010100000000000000000024400000000000003
440)
```

ST_GeomCollection(ST_Geometry,...)-Konstruktor

Konstruiert eine Geometriesammlung aus einer Liste von Geometriewerten.

Syntax

```
NEW ST_GeomCollection(geo1[,geo2,...,geoN])
```

Parameter

Name	Typ	Beschreibung
geo1	ST_Geometry	Der erste Geometriewert der Geometriesammlung.
geo2,...,geoN	ST_Geometry	Zusätzliche Geometriewerte der Geometriesammlung.

Bemerkungen

Konstruiert eine Geometriesammlung aus einer Liste von Geometriewerten. Alle angegebenen Geometriewerte müssen dieselbe SRID aufweisen und die Geometriesammlung wird mit dieser gemeinsamen SRID erstellt.

Alle angegebenen Geometriewerte müssen die gleichen Antworten für Is3D und IsMeasured aufweisen. Die Geometriesammlung ist 3D, wenn alle Geometriewerte 3D sind, und die Geometriesammlung wird gemessen, wenn alle Geometriewerte gemessen werden.

Hinweis

Standardmäßig verwendet ST_GeomCollection das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Der folgende Code gibt eine Geometriesammlung mit dem einzelnen Punkt 'Point (1 2)' zurück.

```
SELECT NEW ST_GeomCollection( NEW ST_Point( 1.0, 2.0 ) )
```

Der folgende Code gibt eine Geometriesammlung mit zwei Punkten 'Point (1 2)' und 'Point (3 4)' zurück.

```
SELECT NEW ST_GeomCollection( NEW ST_Point( 1.0, 2.0 ), NEW ST_Point( 3.0, 4.0 ) )
```

ST_GeomCollectionAggr-Methode

Gibt eine Geometriesammlung zurück, die alle Geometrien in einer Gruppe enthält.

Syntax

```
ST_GeomCollection::ST_GeomCollectionAggr(geometry-column[ ORDER BY order-by-expression [ ASC | DESC ], ... ])
```

Parameter

Name	Typ	Beschreibung
<i>geometry-column</i>	ST_Geometry	Die Geometriewerte zum Generieren der Sammlung. Das ist üblicherweise eine Spalte.

Rückgabe

- **ST_GeomCollection** Gibt eine Geometriesammlung zurück, die alle Geometrien in einer Gruppe enthält.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist derselbe wie der für den ersten Parameter.

Bemerkungen

Die ST_GeomCollectionAggr-Aggregatfunktion kann verwendet werden, um eine Gruppe von Geometrien in einer einzigen Sammlung zu kombinieren. Alle Geometrien, die kombiniert werden sollen, müssen die gleiche SRID und dieselbe Koordinatendimension haben.

Zeilen, in denen das Argument NULL ist, werden nicht eingeschlossen.

Gibt für eine leere Gruppe oder eine Gruppe ohne Nicht-NULL-Werte NULL zurück.

Das sich daraus ergebende ST_GeomCollection-Element hat dieselbe Koordinatendimension wie jede Geometrie.

Die optionale ORDER BY-Klausel kann verwendet werden, um die Elemente in einer bestimmten Reihenfolge anzuordnen, damit ST_GeometryN sie in der gewünschten Reihenfolge zurückgibt. Wenn diese Reihenfolge nicht von Bedeutung ist, sollte aus Effizienzgründen keine Sortierfolge angegeben werden. In diesem Fall hängt die Reihenfolge der Elemente vom Zugriffsplan ab, der vom Abfrageoptimierer ausgewählt wird.

ST_GeomCollectionAggr ist effizienter als ST_UnionAggr, aber ST_GeomCollectionAggr kann eine Sammlung mit mehrfach vorhandenen oder überlappenden Geometrien zurückgeben, wenn sie in der Gruppe von Geometrien vorhanden sind. Insbesondere zurückgegebene Sammlungen, die einander überschneidende Oberflächen enthalten, können unerwartete Ergebnisse hervorrufen, wenn sie als Eingabe für andere räumlichen Methoden verwendet werden. ST_UnionAggr verarbeitet mehrfach vorhandene und überlappende Geometrien.

Hinweis

Standardmäßig verwendet ST_GeomCollectionAggr das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_UnionAggr-Methode für Typ ST_Geometry](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt nur einen einzelnen Wert zurück, der alle Geometrien in der SpatialShapes-Tabelle in einer einzigen Sammlung kombiniert.

```
SELECT ST_GeomCollection::ST_GeomCollectionAggr( Shape ) FROM SpatialShapes
WHERE Shape.ST_Is3D() = 0
```

ST_GeometryN-Methode

Gibt die n -te Geometrie in der Geometriesammlung zurück.

Hinweis

Standardmäßig verwendet ST_GeometryN das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [[SQL Anywhere Server](#) - [SQL-Referenzhandbuch](#)].

Syntax

geomcollection-expression.ST_GeometryN(n)

Parameter

Name	Typ	Beschreibung
n	INT	Die Position des zurückzugebenden Elements, von 1 bis <i>geomcollection-expression</i> .ST_NumInteriorRing().

Rückgabe

- **ST_Geometry** Gibt die n -te Geometrie in der Geometriesammlung zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geomcollection-expression*.

Siehe auch

- [ST_NumGeometries-Methode für den ST_GeomCollection-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.1.5

Beispiel

Das folgende Beispiel gibt den Wert Polygon ((10 -5, 15 5, 5 5, 10 -5)) zurück.

```
SELECT NEW ST_GeomCollection('GeometryCollection (CircularString (5 10, 10
12, 15 10), Polygon ((10 -5, 15 5, 5 5, 10 -5)))')
.ST_GeometryN( 2 )
```

ST_NumGeometries-Methode

Gibt die Anzahl der Geometrien in der Geometriesammlung zurück

Hinweis

Standardmäßig verwendet ST_NumGeometries das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

geomcollection-expression.ST_NumGeometries()

Rückgabe

- **INT** Gibt die Anzahl der in dieser Sammlung gespeicherten Geometrien zurück.

Siehe auch

- [ST_GeometryN-Methode für den ST_GeomCollection-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.1.4

Beispiel

Das folgende Beispiel gibt den Wert 3 zurück.

```
SELECT NEW ST_MultiPoint('MultiPoint ((10 10), (12 12), (14 10))')
       .ST_NumGeometries()
```

ST_Geometry-Datentyp

Der ST_Geometry-Datentyp ist der maximale übergeordnete Datentyp der Geometrie-Datentyp-Hierarchie.

Direkt untergeordnete Typen

- „ST_Curve-Datentyp“
- „ST_GeomCollection-Datentyp“
- „ST_Point-Datentyp“
- „ST_Surface-Datentyp“

Methoden

- Methoden von ST_Geometry:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAgr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr

ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType
ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth
ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBaseType	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine
ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Bemerkungen

Der ST_Geometry-Datentyp ist der maximale übergeordnete Datentyp der Geometrie-Datentyp-Hierarchie. Der Datentyp ST_Geometry unterstützt Methoden, die auf jeden räumlichen Wert angewendet werden können. Der ST_Geometry-Datentyp kann nicht instanziiert werden. Anstelle dessen ist ein untergeordneter Typ zu instanziiieren. Bei der Arbeit mit Originalformaten (WKT oder WKB) können Sie Methoden wie ST_GeomFromText/ST_GeomFromWKB verwenden, um den geeigneten konkreten Typ zu instanziiieren, der den Wert im Originalformat darstellt.

Alle Werte in einem ST_Geometry Wert befinden sich in demselben räumlichen Bezugssystem. Die ST_SRID dieser Methode kann verwendet werden, um das räumliche Bezugssystem abzurufen oder zu ändern, das dem Wert zugeordnet ist.

Spalten vom Datentyp ST_Geometry oder seiner Untertypen können nicht in einen Primärschlüssel, einen eindeutigen Index oder eine Eindeutigkeits-Integritätsregel einbezogen werden.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1

ST_Affine-Methode

Gibt eine neue Geometrie zurück, die das Ergebnis der Anwendung der angegebenen affinen 3-D-Transformation ist.

Syntax

geometry-expression.**ST_Affine**(a00,a01,a02,a10,a11,a12,a20,a21,a22,xoff,yoff,zoff)

Parameter

Name	Typ	Beschreibung
a00	DOUBLE	Das affine Matricelement in Zeile 0, Spalte 0
a01	DOUBLE	Das affine Matricelement in Zeile 0, Spalte 1
a02	DOUBLE	Das affine Matricelement in Zeile 0, Spalte 2
a10	DOUBLE	Das affine Matricelement in Zeile 1, Spalte 0
a11	DOUBLE	Das affine Matricelement in Zeile 1, Spalte 1
a12	DOUBLE	Das affine Matricelement in Zeile 1, Spalte 2
a20	DOUBLE	Das affine Matricelement in Zeile 2, Spalte 0
a21	DOUBLE	Das affine Matricelement in Zeile 2, Spalte 1
a22	DOUBLE	Das affine Matricelement in Zeile 2, Spalte 2
xoff	DOUBLE	Der x-Offset für die Konvertierung
yoff	DOUBLE	Der y-Offset für die Konvertierung
zoff	DOUBLE	Der z-Offset für die Konvertierung

Rückgabe

- **ST_Geometry** Gibt eine neue Geometrie zurück, die das Ergebnis der angegebenen Transformation ist.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Ein affine Transformation kombiniert Rotation, Konvertierung und Skalierung in einem einzigen Methodenaufruf. Die affine Transformation wird mit der Matrix-Multiplikation definiert.

Bei einem Punkt (x,y,z) wird das Ergebnis (x',y',z') wie folgt berechnet:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} a00 & a01 & a02 & xoff \\ a20 & a21 & a22 & zoff \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} yoff \\ w' \end{pmatrix}$$

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Die folgende Anweisung liefert das Ergebnis LineString (5 6, 5 3, 9 3). Die X-Werte werden durch 5 und die Y-Werte durch -1 übersetzt.

```
SELECT Shape.ST_Affine( 1,0,0, 0,1,0, 0,0,1, 5,-1,0 )
FROM SpatialShapes WHERE ShapeID = 5
```

Die folgende Anweisung gibt die Ergebnismenge LineString (.698833 6.965029, .399334 3.980017, 4.379351 3.580683) zurück. Die Form wird mit 0,1 rad (0,1 im Bogenmaß, entspricht 5,7 Grad) um die Z-Achse rotiert.

```
SELECT Shape.ST_Affine( cos(0.1),sin(0.1),0, -sin(0.1),cos(0.1),0, 0,0,1,
0,0,0 )
FROM SpatialShapes WHERE ShapeID = 5
```

ST_AsBinary-Methode

Gibt die WKB-Darstellung eines ST_Geometry Werts zurück.

Syntax

geometry-expression.**ST_AsBinary**([*format*])

Parameter

Name	Typ	Beschreibung
format	VAR-CHAR(128)	Eine Zeichenfolge, die das Ausgabe-Binärformat für die Konvertierung von <i>geometry-expression</i> in eine binäre Darstellung definiert. Wird dies nicht angegeben, kommt der Wert der Option <code>st_geometry_asbinary_format</code> für die Auswahl der binären Darstellung zur Anwendung. Siehe „ st_geometry_asbinary_format-Option “ [<i>SQL Anywhere Server - Datenbank-administration</i>].

Rückgabe

- **LONG BINARY** Gibt die WKB-Darstellung von *geometry-expression* zurück.

Bemerkungen

Die `ST_AsBinary`-Methode gibt eine Binärzeichenfolge zurück, die die Geometrie darstellt. Verschiedene Binärformate werden unterstützt (mit den dazugehörigen Optionen) und das gewünschte Format wird mit dem optionalen *format*-Parameter ausgewählt. Wenn der *format*-Parameter nicht angegeben ist, wird die Option `st_geometry_asbinary_format` zur Auswahl des Ausgabeformats verwendet. Siehe „[st_geometry_asbinary_format-Option](#)“ [*SQL Anywhere Server - Datenbankadministration*].

Die Formatzeichenfolge definiert ein Ausgabeformat und Parameter für das Format. Die Formatzeichenfolge hat eines der folgenden Formate:

```
format-name
format-name(parameter1=value1;parameter2=value2;...)
parameter1=value1;parameter2=value2;...
```

Das erste Format legt den Formatnamen und keine Parameter fest. Alle Formatparameter benutzen ihre Standardwerte. Das zweite Format legt den Formatnamen und eine Liste von Parameterwerten fest. Nicht angegebene Parameter verwenden die Standardwerte. Das letzte Format gibt nur Parameterwerte an, der Formatname verwendet den Standardnamen 'WKB'.

Die folgenden Formatnamen können benutzt werden:

- **WKB** Das Well-Known-Binärformat laut der Definition von SQL/MM und OGC.
- **EWKB** Das erweiterte Well-Known-Binärformat laut der Definition von PostGIS. Dieses Format enthält die SRID der Geometrie und unterscheidet sich vom WKB durch die Darstellung der Z- und M-Werte.

Die folgenden *format*-Parameter können angegeben werden:

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
WKB	Version	1.2	<ul style="list-style-type: none"> 1.1 Das WKB gemäß der OGC SFS 1.1-Definition. Dieses Format enthält keine Z- und M-Werte. Wenn die Geometrie Z- oder M-Werte enthält, werden sie in der Ausgabe entfernt. 1.2 Das WKB gemäß der OGC SFS 1.2-Definition. Dies passt zu Version 1.1 bei 2D-Daten und erweitert das Format zur Unterstützung von Z- und M-Werten. 	Der version-Parameter steuert die Version der benutzten WKB-Spezifikation.

Hinweis

Beim Konvertieren eines Geometriewerts in BINARY verwendet der Server die ST_AsBinary-Methode. Die st_geometry_asbinary_format-Option definiert das Format für die Konvertierung. Siehe „st_geometry_asbinary_format-Option“ [[SQL Anywhere Server - Datenbankadministration](#)].

Hinweis

Standardmäßig verwendet ST_AsBinary das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.37

Beispiel

Wenn die st_geometry_asbinary_format-Option als Standardwert 'WKB' hat, gibt die folgende Anweisung das Ergebnis

```
0x01b90b000000000000000000f03f00000000000000400000000000000840000000000000001040 zurück.
```

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsBinary()
```

Wenn die st_geometry_asbinary_format-Option als Standardwert 'WKB' hat, gibt die folgende Anweisung das Ergebnis

```
0x01b90b000000000000000000f03f00000000000000400000000000000840000000000000001040 zurück. Der Server ruft implizit die ST_AsBinary-Methode auf, wenn Geometrien in BINARY konvertiert werden.
```

```
SELECT CAST( NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ) AS LONG BINARY)
```

Die folgende Anweisung gibt die Ergebnismenge
0x010100000000000000000000f03f0000000000000040 zurück. Die Z- und M-Werte werden
ausgeklammert, weil die Version 1.1 der OGC-Spezifikation für WKB keine Unterstützung dafür bietet.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0,  
4326 ).ST_AsBinary('WKB(Version=1.1;endian=little)')
```

Die folgende Anweisung gibt die Ergebnismenge
0x01010000e0e6100000000000000000f03f000000000000004000000000000084000
00000000001040 zurück. Das erweiterte WKB enthält die SRID.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0,  
4326 ).ST_AsBinary('EWKB(endian=little)')
```

ST_AsGML-Methode

Gibt die GML-Darstellung eines ST_Geometry-Werts zurück.

Syntax

```
geometry-expression.ST_AsGML([ format])
```

Parameter

Name	Typ	Beschreibung
format	VAR-CHAR(128)	Eine Zeichenfolge, die die Parameter für die Konvertierung von <i>geometry-expression</i> in eine GML-Darstellung definiert. Wenn sie nicht festgelegt wird, ist der Standardwert 'GML'.

Rückgabe

- **LONG VARCHAR** Gibt die GML-Darstellung von *geometry-expression* zurück.

Bemerkungen

Die ST_AsGML-Methode gibt eine GML-Zeichenfolge zurück, die die Geometrie darstellt. Verschiedene
Formate werden unterstützt (mit den dazugehörigen Optionen) und das gewünschte Format wird mit dem
optionalen *format*-Parameter ausgewählt. Wenn der *format*-Parameter nicht angegeben wird, ist der
Standardwert 'GML'.

Die Formatzeichenfolge definiert ein Ausgabeformat und Parameter für das Format. Die
Formatzeichenfolge hat eines der folgenden Formate:

```
format-name  
  
format-name(parameter1=value1;parameter2=value2;...)  
  
parameter1=value1;parameter2=value2;...
```

Das erste Format legt den Formatnamen und keine Parameter fest. Alle Formatparameter benutzen ihre
Standardwerte. Das zweite Format legt den Formatnamen und eine Liste von Parameterwerten fest. Nicht

angegebene Parameter verwenden die Standardwerte. Das letzte Format gibt nur Parameterwerte an, der Formatname verwendet den Standardnamen 'GML'.

Die folgenden Formatnamen können benutzt werden:

- **GML** Das "Geography Markup Language"-Format gemäß der Definition in ISO 19136 und OGC.

Die folgenden format-Parameter können angegeben werden:

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
GML	Namespace	none	<ul style="list-style-type: none"> • local Stellt ein Standard-Namespace-Attribut für das angegebene Element (in diesem Fall Point) und seine untergeordneten Elemente bereit. • global Stellt ein dediziertes Präfix ("gml") für das angegebene Element und seine untergeordneten Elemente bereit. Dies ist sinnvoll, wenn die Abfrage innerhalb eines Aggregatvorgangs verwendet wird. Bestimmte Elemente der obersten Ebene definieren den Namespace für das "gml"-Präfix. • none Stellt keinen Namespace oder kein Präfix für das angegebene Element (in diesem Fall Point) und seine untergeordneten Elemente bereit. 	Der namespace-Parameter legt die Ausgabeformat-Konvention für Namespace fest.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
GML	SRSNameFormat	short	<ul style="list-style-type: none"> • short Verwendet ein kurzes Format für das räumliche Bezugssystem, zum Beispiel EPSG:4326 • long Benutzt ein langes Format für den Namen des räumlichen Bezugssystems, zum Beispiel urn:x-ogc:def:crs:EPSG:4326. • none Das Namensattribut für das räumliche Bezugssystem wird für die Geometrie nicht angegeben. 	Der SRSNameFormat-Parameter legt das Format für das srsName-Attribut fest.
GML	SRSDimension	No	Yes oder No	Der SRSDimension-Parameter legt die Anzahl der Koordinatenwerte für die angegebene Geometrie fest. Dies gilt nur für GML(version=3).
GML	SRSFillAll	No	Yes oder No	Der SRSFillAll-Parameter gibt an, ob SRS-Attribute an untergeordnete Geometrieelemente weitergereicht werden sollen oder nicht. Beispiel: MultiGeometry oder MultiPolygon würde die Attribute an die untergeordneten Geometrien weitergeben.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
GML	UseDeprecated	No	Yes oder No	Der UseDeprecated-Parameter gilt nur für GML(version=3). Er wird gegebenenfalls für die Ausgabe von älteren GML-Darstellungen verwendet. Beispiel: Eine Ebene (Surface-Element) kann als Polygon ausgegeben werden, wenn die Geometrie keine CircularStrings enthält.
GML	Attribute	Automatisch generierte optionale Attribute	Ein oder mehrere Attribute können nur für das Geometrieelement der obersten Ebene angegeben werden.	Alle zulässigen XML-Attribute können angegeben werden.
GML	SubElement	Automatisch generierte untergeordnete GML-Elemente	Ein oder mehrere untergeordnete Attribute können nur für das Geometrieelement der obersten Ebene angegeben werden.	Alle zulässigen XML-Elemente können angegeben werden.

Hinweis

Standardmäßig verwendet ST_AsGML das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.39

Beispiel

Das folgende Beispiel gibt den Wert `<Point srsName="EPSG:4326"><pos>1 2 3 4</pos></Point>` zurück.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsGML()
```

Das folgende Beispiel gibt den Wert `<Point srsName="EPSG:4326"><coordinates>1,2</coordinates></Point>` zurück.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsGML('GML(Version=2)')
```

Die folgende Anweisung gibt die Ergebnismenge `<gml:Point srsName="EPSG:4326"><gml:coordinates>1,2</gml:coordinates></gml:Point>` zurück. Der Namespace=global-Parameter stellt ein dediziertes Präfix ("gml") für das angegebene Element und seine untergeordneten Elemente bereit. Dies ist sinnvoll, wenn die Abfrage innerhalb eines Aggregatvorgangs verwendet wird. Bestimmte Elemente der obersten Ebene definieren den Namespace für das "gml"-Präfix.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsGML('GML(Version=2;Namespace=global)')
```

Die folgende Anweisung gibt die Ergebnismenge `<Point srsName="EPSG:4326"><coordinates>1,2</coordinates></Point>` zurück. In der Ausgabe sind keine Namespace-Informationen enthalten.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsGML('GML(Version=2;Namespace=none)')
```

Die folgende Anweisung gibt die Ergebnismenge `<Point srsName="http://www.opengis.net/gml/srs/epsg.xml#4326"><coordinates>1,2</coordinates></Point>` zurück. Das ausführliche Format des srsName-Attributs wird verwendet.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsGML('GML(Version=2;Namespace=none;SRSNameFormat=long)')
```

Die folgende Anweisung gibt die Ergebnismenge `<Point srsName="urn:x-ogc:def:crs:EPSG:4326"><pos>1 2 3 4</pos></Point>` zurück. Das ausführliche Format des srsName-Attributs wird verwendet und das Format unterscheidet sich in Version 3 vom Format der Version 2.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsGML('GML(Version=3;Namespace=none;SRSNameFormat=long)')
```

ST_AsGeoJSON-Methode

Gibt eine Zeichenfolge zurück, die eine Geometrie im JSON-Format darstellt.

Syntax

geometry-expression.**ST_AsGeoJSON**([*format*])

Parameter

Name	Typ	Beschreibung
format	VAR-CHAR(128)	Eine Zeichenfolge, die Parameter definiert, mit denen gesteuert werden kann, wie das GeoJSON-Ergebnis generiert wird. Wenn sie nicht festgelegt wird, ist der Standardwert 'GeoJSON'.

Rückgabe

- **LONG VARCHAR** Gibt die GeoJSON-Darstellung von *geometry-expression* zurück.

Bemerkungen

Der GeoJSON-Standard definiert das Geospatial Interchange Format auf der Basis der JavaScript Object Notation (JSON). Dieses Format ist für webbasierte Anwendungen geeignet und kann ein Format bereitstellen, das klarer und leichter zu interpretieren ist als WKT oder WKB. Siehe <http://geojson.org/geojson-spec.html>.

Die ST_AsGeoJSON-Methode gibt eine Textzeichenfolge zurück, die die Geometrie darstellt. Verschiedene Textformate werden (mit den zugeordneten Optionen) unterstützt und das gewünschte Format wird mit dem optionalen *format*-Parameter ausgewählt. Wenn der *format*-Parameter nicht angegeben wird, ist der Standardwert 'GeoJSON'.

Die Formatzeichenfolge definiert ein Ausgabeformat und Parameter für das Format. Die Formatzeichenfolge hat eines der folgenden Formate:

```
format-name
format-name(parameter1=value1;parameter2=value2;...)
parameter1=value1;parameter2=value2;...
```

Das erste Format legt den Formatnamen und keine Parameter fest. Alle Formatparameter benutzen ihre Standardwerte. Das zweite Format legt den Formatnamen und eine Liste von Parameterwerten fest. Nicht angegebene Parameter verwenden die Standardwerte. Das letzte Format gibt nur Parameterwerte an, der Formatname verwendet den Standardnamen 'GeoJSON'.

Die folgenden Formatnamen können benutzt werden:

- **GeoJSON** Das GeoJSON-Format verwendet JavaScript Object Notation (JSON) gemäß der Definition in <http://geojson.org/geojson-spec.html>.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
GeoJSON	Version	1.0	1.0	Die Version der GeoJSON-Spezifikation, die verwendet werden muss. Derzeit wird nur 1.0 unterstützt.

Hinweis

Standardmäßig verwendet ST_AsGeoJSON das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert `{"type": "Point", "coordinates": [1, 2]}` zurück.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsGeoJSON()
```

ST_AsKML-Methode

Gibt die KML-Darstellung eines ST_Geometry-Werts zurück.

Syntax

geometry-expression.ST_AsKML([*format*])

Parameter

Name	Typ	Beschreibung
format	VAR-CHAR(128)	Eine Zeichenfolge, die die Parameter für die Konvertierung von <i>geometry-expression</i> in eine KML-Darstellung definiert. Wenn sie nicht festgelegt wird, ist der Standardwert 'KML'.

Rückgabe

- **LONG VARCHAR** Gibt die KML-Darstellung von *geometry-expression* zurück.

Bemerkungen

Die ST_AsKML-Methode gibt eine KML Zeichenfolge zurück, die die Geometrie darstellt. Verschiedene Formate werden unterstützt (mit den dazugehörigen Optionen) und das gewünschte Format wird mit dem optionalen *format*-Parameter ausgewählt. Wenn der *format*-Parameter nicht angegeben wird, ist der Standardwert 'KML'.

Die Formatzeichenfolge definiert ein Ausgabeformat und Parameter für das Format. Die Formatzeichenfolge hat eines der folgenden Formate:

```
format-name  
  
format-name(parameter1=value1;parameter2=value2;...)  
  
parameter1=value1;parameter2=value2;...
```

Das erste Format legt den Formatnamen und keine Parameter fest. Alle Formatparameter benutzen ihre Standardwerte. Das zweite Format legt den Formatnamen und eine Liste von Parameterwerten fest. Nicht angegebene Parameter verwenden die Standardwerte. Das letzte Format gibt nur Parameterwerte an, der Formatname verwendet den Standardnamen 'KML'.

Die folgenden Formatnamen können benutzt werden:

- **KML** Das Keyhole Markup Language-Format gemäß Definition von OGC.

Die folgenden format-Parameter können angegeben werden:

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
KML	Version	2	2	KML Version 2.2 wird unterstützt.
KML	Attribute	Automatisch generierte optionale Attribute	Ein oder mehrere Attribute können nur für das Geometrieelement der obersten Ebene angegeben werden.	Alle zulässigen XML-Attribute können angegeben werden.
KML	Namespace	none	<ul style="list-style-type: none"> • local Stellt das Standard Namespace-Attribut http://www.open-gis.net/kml/2.2 für das angegebene Geometrieelement (in diesem Fall Point) und seine untergeordneten Elemente bereit. • global Stellt ein dediziertes Präfix ("kml") für das angegebene Element und seine untergeordneten Elemente bereit. Dies ist sinnvoll, wenn die Abfrage innerhalb eines Aggregatvorgangs verwendet wird. Bestimmte Elemente der obersten Ebene definieren den Namespace für das "kml"-Präfix. • none Stellt keinen Namespace oder kein Präfix für das angegebene Element (in diesem Fall Point) und seine untergeordneten Elemente bereit. 	Der namespace-Parameter legt die Ausgabeformat-Konvention für Namespace fest.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
KML	SubElement	Automatisch generierte untergeordnete KML-Elemente	Ein oder mehrere untergeordnete Attribute können nur für das Geometrieelement der obersten Ebene angegeben werden.	Alle zulässigen XML-Elemente können angegeben werden. Beispiel: extrude, tessellate und altitudeMode-Elemente können angegeben werden.

Hinweis

Standardmäßig verwendet ST_AsKML das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.39

Beispiel

Das folgende Beispiel gibt den Wert `<Point><coordinates>1,2,3,4</coordinates></Point>` zurück.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsKML()
```

Das folgende Beispiel gibt den Wert `<Point><coordinates>1,2,3,4</coordinates></Point>` zurück.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsKML('KML(Version=2)')
```

Die folgende Anweisung gibt die Ergebnismenge `<kml:Point><kml:coordinates>1,2,3,4</kml:coordinates></kml:Point>` zurück. Der Namespace=global-Parameter stellt ein dediziertes Präfix ("kml") für das angegebene Element und seine untergeordneten Elemente bereit. Dies ist sinnvoll, wenn die Abfrage innerhalb eines Aggregatvorgangs verwendet wird. Bestimmte Elemente der obersten Ebene definieren den Namespace für das "kml"-Präfix.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsKML('KML(Version=2;Namespace=global)')
```

Die folgende Anweisung gibt die Ergebnismenge `<Point><coordinates>1,2,3,4</coordinates></Point>` zurück. In der Ausgabe sind keine Namespace-Informationen enthalten.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsKML('KML(Version=2;Namespace=none)')
```

Die folgende Anweisung gibt die Ergebnismenge `<Point xmlns="http://www.opengis.net/kml/2.2"><coordinates>1,2,3,4</coordinates></Point>` zurück. Der xml-Standard-Namespace wird verwendet.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0,
4326 ).ST_AsKML('KML(Version=2;Namespace=default)')
```

Die folgende Anweisung gibt die Ergebnismenge <Point><altitudeMode>absolute</altitudeMode><coordinates>1,2,3,4</coordinates></Point> zurück. Ein untergeordnetes AltitudeMode-Element wird in die Ausgabe einbezogen.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0,
4326 ).ST_AsKML('SubElement=<altitudeMode>absolute</altitudeMode>')
```

ST_AsSVG-Methode

Gibt eine SVG-Abbildung für einen Geometriewert zurück.

Syntax

geometry-expression.ST_AsSVG([*format*])

Parameter

Name	Typ	Beschreibung
format	VAR-CHAR(128)	Eine Zeichenfolge, die die Parameter für die Konvertierung von <i>geometry-expression</i> in eine SVG-Darstellung definiert. Wenn sie nicht festgelegt wird, ist der Standardwert 'SVG'.

Rückgabe

- **LONG VARCHAR** Gibt ein vollständiges oder teilweises SVG-Dokument zurück, das *geometry-expression* darstellt.

Bemerkungen

Die ST_AsSVG-Methode gibt ein vollständiges oder teilweises SVG-Dokument zurück, das für die grafische Anzeige der SVG-Datei verwendet werden kann. Die meisten Webbrowser mit Ausnahme von Microsoft Internet Explorer enthalten integrierte SVG-Anzeigefunktionen.

Verschiedene Optionen werden unterstützt und das gewünschte Format wird mit dem optionalen *format*-Parameter ausgewählt. Wenn der *format*-Parameter nicht angegeben wird, ist der Standardwert 'SVG'.

Die Formatzeichenfolge definiert ein Ausgabeformat und Parameter für das Format. Die Formatzeichenfolge hat eines der folgenden Formate:

```
format-name
```

```
format-name(parameter1=value1;parameter2=value2;...)
```

```
parameter1=value1;parameter2=value2;...
```

Das erste Format legt den Formatnamen und keine Parameter fest. Alle Formatparameter benutzen ihre Standardwerte. Das zweite Format legt den Formatnamen und eine Liste von Parameterwerten fest. Nicht angegebene Parameter verwenden die Standardwerte. Das letzte Format gibt nur Parameterwerte an, der Formatname verwendet den Standardnamen 'SVG'.

Die folgenden Formatnamen können benutzt werden:

- **SVG** Das Scalable Vector Graphics (SVG) 1.1-Format gemäß der Definition des World Wide Web Consortium (W3C).

Die folgenden format-Parameter können angegeben werden:

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
SVG	Approximate	Yes	Yes oder No	Der Parameter "Approximate" gibt an, ob die Größe des ausgegebenen SVG-Dokuments mit einer geringfügigen Verringerung der sichtbaren Details reduziert werden soll oder nicht. Die SVG-Daten werden als Näherungswert ("Approximate") dargestellt, indem Punkte nicht einbezogen werden, die sich innerhalb der Linienbreite des letzten Punkts befinden. Bei Geometrien mit mehreren Megabyte kann dies zu Kompressionsraten von 80 % oder mehr führen.
SVG	Attribute	Automatisch generierte optionale Attribute	Ein oder mehr SVG-Attribute, die an SVG-Formelemente angewendet werden können	Standardmäßig werden optionale SVG-Formattribute wie "fill", "stroke" und "stroke-width" generiert. Wenn der Attribute-Parameter angegeben ist, werden keine optionalen SVG-Formattribute generiert und der Attribute-Wert wird stattdessen verwendet. Wird ignoriert, wenn PathDataOnly=Yes festgelegt ist. Die maximale Länge des Attribute-Werts beträgt etwa 1000 Byte.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
SVG	DecimalDigits	Basierend auf der Anzahl von Dezimalstellen der Rastergröße für die Funktion "Am Raster ausrichten" des räumlichen Bezugssystems. Der maximale Standardwert ist 5 und der Mindestwert ist 0 MB.	integer	Der DecimalDigits-Parameter begrenzt die Anzahl der Ziffern hinter dem Dezimalzeichen für Koordinaten, die in der SVG-Ausgabe generiert werden. Eine negative Stellenanzahl zeigt an, dass die Gesamtstellenzahl der Koordinaten in die SVG-Ausgabe einbezogen werden soll.
SVG	PathDataOnly	No (ein vollständiges SVG-Dokument wird generiert)	Yes oder No	Der PathDataOnly-Parameter legt fest, ob nur die Daten für das SVG-Pfadelement generiert werden sollen. Das nachstehende PathDataOnly-Beispiel zeigt, wie PathDataOnly=Yes verwendet werden kann, um ein vollständiges SVG-Dokument zu erstellen, das angezeigt werden kann. Standardmäßig wird ein vollständiges SVG-Dokument generiert. Die von PathDataOnly=Yes zurückgegebenen Pfaddaten können verwendet werden, um flexiblere SVG-Dokumente zu erstellen, die andere Elemente enthalten, beispielsweise Text.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
SVG	RandomFill	Yes	Yes oder No	Der RandomFill-Parameter gibt an, ob Polygone mit einer zufalls-generierten Farbe gefüllt werden sollen. Die verwendete Farbense-quenz hält sich nicht an eine bestimmte Reihenfolge und ändert sich in der Regel jedes Mal, wenn eine SVG-Ausgabe generiert wird. None legt fest, dass nur ein Umriss jedes Poly-gons ausgegeben wird. Der Rand-omFill-Parameter wird ignoriert, wenn der Attribute-oder PathDa-taOnly=Yes-Parameter angege-ben ist.
SVG	Relative	Yes	Yes oder No	Der Relative-Parameter legt fest, ob Koordinaten im relativen For-mat (Offset) oder absoluten For-mat ausgegeben werden sollen. Relative Koordinatendaten sind in der Regel kompakter als abso-lute Koordinatendaten.

Hinweis

Standardmäßig verwendet ST_AsSVG das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_AsSVGAgr-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Der folgende Code gibt ein vollständiges SVG-Dokument mit Polygonen zurück, die mit zufallsgenerierten Farben gefüllt sind.

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 0 20, 60 10, 0 0 ))' )
       .ST_AsSVG()
```

Der folgende Code gibt ein vollständiges SVG-Dokument mit Umrisspolygonen zurück und beschränkt die Koordinaten auf 3 Zeichen hinter dem Dezimalzeichen.

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 0 20, 60 10, 0 0 ))' )
      .ST_AsSVG( 'RandomFill=No;DecimalDigits=3' )
```

Der folgende Code gibt ein vollständiges SVG-Dokument mit Polygonen zurück, die blau gefüllt sind und Koordinaten mit maximaler Gesamtstellenzahl aufweisen.

```
SELECT Shape.ST_AsSVG( 'Attribute=fill="blue";DecimalDigits=-1' )
FROM SpatialShapes
```

Der folgende Code gibt ein vollständiges SVG-Dokument von SVG-Pfaddaten mit relativen, auf 5 Zeichen hinter dem Dezimalzeichen begrenzten Koordinaten zurück.

```
SELECT '<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
<svg viewBox="-180 -90 360 180" xmlns="http://www.w3.org/2000/svg"
      version="1.1">
  <path fill="lightblue" stroke="black" stroke-width="0.1%" d=" ' ||
NEW ST_Polygon( 'Polygon(( 0 0, 0 20, 60 10, 0 0 ))' )
      .ST_AsSVG( 'PathDataOnly=Yes' ) ||
' "/></svg>'
```

Der folgende Code gibt SVG-Pfaddaten mit absoluten, auf 7 Zeichen hinter dem Dezimalzeichen begrenzten Koordinaten.

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 0 20, 60 10, 0 0 ))' )
      .ST_AsSVG( 'PathDataOnly=Yes;Relative=No;DecimalDigits=7' )
```

ST_AsSVGAgr-Methode

Gibt ein vollständiges oder teilweises SVG-Dokument zurück, das die Geometrien in einer Gruppe darstellt.

Syntax

```
ST_Geometry::ST_AsSVGAgr(geometry-column[ ORDER BY order-by-expression [ ASC | DESC ], ... ][, format])
```

Parameter

Name	Typ	Beschreibung
<i>geometry-column</i>	ST_Geometry	Der Geometriewert für die SVG-Abbildung. Das ist üblicherweise eine Spalte.
<i>format</i>	VAR-CHAR(128)	Eine Zeichenfolge, die die Parameter definiert, die bei der Konvertierung der einzelnen Geometriewerte in eine SVG-Darstellung zu verwenden sind. Wenn sie nicht festgelegt wird, ist der Standardwert 'SVG'.

Rückgabe

- **LONG VARCHAR** Gibt ein vollständiges oder teilweises SVG-Dokument zurück, das die Geometrien in einer Gruppe darstellt.

Bemerkungen

Die `ST_AsSVGAggr`-Methode gibt ein komplettes oder teilweises SVG-Dokument zurück, das verwendet werden kann, um die Vereinigung einer Gruppe von Geometrien mit einer SVG-Anzeigefunktion grafisch darzustellen. Die meisten Webbrowser mit Ausnahme von Microsoft Internet Explorer enthalten integrierte SVG-Anzeigefunktionen.

Verschiedene Optionen werden unterstützt und das gewünschte Format wird mit dem optionalen *format*-Parameter ausgewählt. Wenn der *format*-Parameter nicht angegeben wird, ist der Standardwert 'SVG'.

Die Formatzeichenfolge definiert ein Ausgabeformat und Parameter für das Format. Die Formatzeichenfolge hat eines der folgenden Formate:

```
format-name  
  
format-name(parameter1=value1;parameter2=value2;...)  
  
parameter1=value1;parameter2=value2;...
```

Das erste Format legt den Formatnamen und keine Parameter fest. Alle Formatparameter benutzen ihre Standardwerte. Das zweite Format legt den Formatnamen und eine Liste von Parameterwerten fest. Nicht angegebene Parameter verwenden die Standardwerte. Das letzte Format gibt nur Parameterwerte an, der Formatname verwendet den Standardnamen 'SVG'.

Die folgenden Formatnamen können benutzt werden:

- **SVG** Das Scalable Vector Graphics (SVG) 1.1-Format gemäß der Definition des World Wide Web Consortium (W3C).

Die folgenden format-Parameter können angegeben werden:

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
SVG	Approximate	Yes	Yes oder No	Der Parameter "Approximate" gibt an, ob die Größe des ausgegebenen SVG-Dokuments mit einer geringfügigen Verringerung der sichtbaren Details reduziert werden soll oder nicht. Die SVG-Daten werden als Näherungswert ("Approximate") dargestellt, indem Punkte nicht einbezogen werden, die sich innerhalb der Linienbreite des letzten Punkts befinden. Bei Geometrien mit mehreren Megabyte kann dies zu Kompressionsraten von 80 % oder mehr führen.
SVG	Attribute	Automatisch generierte optionale Attribute	Ein oder mehr SVG-Attribute, die an SVG-Formelemente angewendet werden können	Standardmäßig werden optionale SVG-Formattribute wie "fill", "stroke" und "stroke-width" generiert. Wenn der Attribute-Parameter angegeben ist, werden keine optionalen SVG-Formattribute generiert und der Attribute-Wert wird stattdessen verwendet. Wird ignoriert, wenn PathData-Only=Yes festgelegt ist.
SVG	DecimalDigits	Basierend auf der Anzahl von Dezimalstellen der Rastergröße für die Funktion "Am Raster ausrichten" des räumlichen Bezugssystems. Der maximale Standardwert ist 5 und der Mindestwert ist 0 MB.	integer	Der DecimalDigits-Parameter begrenzt die Anzahl der Ziffern hinter dem Dezimalzeichen für Koordinaten, die in der SVG-Ausgabe generiert werden. Eine negative Stellenanzahl zeigt an, dass die Gesamtstellenzahl der Koordinaten in die SVG-Ausgabe einbezogen werden soll.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
SVG	PathDataOnly	No (ein vollständiges SVG-Dokument wird generiert)	Yes oder No	Der PathDataOnly-Parameter legt fest, ob nur die Daten für das SVG-Pfadelement generiert werden sollen. Das nachstehende PathDataOnly-Beispiel zeigt, wie PathDataOnly=Yes verwendet werden kann, um ein vollständiges SVG-Dokument zu erstellen, das angezeigt werden kann. Standardmäßig wird ein vollständiges SVG-Dokument generiert. Die von PathDataOnly=Yes zurückgegebenen Pfaddaten können verwendet werden, um flexiblere SVG-Dokumente zu erstellen, die andere Elemente enthalten, beispielsweise Text.
SVG	RandomFill	Yes	Yes oder No	Der RandomFill-Parameter gibt an, ob Polygone mit einer zufalls-generierten Farbe gefüllt werden sollen. Die verwendete Farbensequenz hält sich nicht an eine bestimmte Reihenfolge und ändert sich in der Regel jedes Mal, wenn eine SVG-Ausgabe generiert wird. None legt fest, dass nur ein Umriss jedes Polygons ausgegeben wird. Der RandomFill-Parameter wird ignoriert, wenn der Attribute-oder PathDataOnly=Yes-Parameter angegeben ist.
SVG	Relative	Yes	Yes oder No	Der Relative-Parameter legt fest, ob Koordinaten im relativen Format (Offset) oder absoluten Format ausgegeben werden sollen. Relative Koordinatendaten sind in der Regel kompakter als absolute Koordinatendaten.

Die ORDER BY-Klausel kann angegeben werden, um zu steuern, wie überlappende Geometrien dargestellt werden, wobei die Geometrien von hinten nach vorn angezeigt werden. Wenn dieser Parameter

nicht festgelegt wird, werden die Geometrien in einer Reihenfolge angezeigt, die vom Ausführungsplan abhängt, der vom Abfrageoptimierer ausgewählt wird. Dies kann von Ausführung zu Ausführung der Anweisung unterschiedlich sein.

Hinweis

Standardmäßig verwendet ST_AsSVGAggr das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_AsSVG-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Der folgende Code gibt ein vollständiges SVG-Dokument mit Polygonen zurück, die mit zufallsgenerierten Farben gefüllt sind.

```
SELECT ST_Geometry::ST_AsSVGAggr( Shape ) FROM SpatialShapes
```

Der folgende Code gibt ein vollständiges SVG-Dokument von SVG-Pfaddaten mit relativen, auf 5 Zeichen hinter dem Dezimalzeichen begrenzten Koordinaten zurück.

```
SELECT '<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
<svg viewBox="-10 -10 20 12" xmlns="http://www.w3.org/2000/svg"
version="1.1">
  <path fill="lightblue" stroke="black" stroke-width="0.1%" d=" ' ||
    ST_Geometry::ST_AsSVGAggr( Shape, 'PathDataOnly=Yes' ) ||
  '"/></svg>'
FROM SpatialShapes
```

Die folgenden Anweisungen erstellen einen Webdienst, der ein vollständiges SVG-Dokument zurückgibt, das alle Geometrien in der SpatialShapes-Tabelle umfasst. Wenn der Datenbankserver mit der http-Option -xs gestartet wird, können Sie über einen Browser, der das SVG-Format unterstützt, die SVG-Datei anzeigen. Navigieren Sie dafür zur Adresse http://localhost/demo/svg_shapes. Dies funktioniert unter der Voraussetzung, dass sich der Browser und der Datenbankserver auf demselben Computer befinden und die Datenbank "demo" heißt.

```
CREATE SERVICE svg_shapes TYPE 'RAW' USER DBA AUTHORIZATION OFF
AS CALL svg_shapes();

CREATE PROCEDURE svg_shapes()
  RESULT( svg LONG VARCHAR )
BEGIN
  CALL sa_set_http_header( 'Content-type', 'image/svg+xml' );
  SELECT ST_Geometry::ST_AsSVGAggr( Shape ) FROM SpatialShapes;
END;
```

ST_AsText-Methode

Gibt die Textdarstellung eines ST_Geometry-Werts zurück.

Syntax

geometry-expression.**ST_AsText**([*format*])

Parameter

Name	Typ	Beschreibung
format	VAR-CHAR(128)	Eine Zeichenfolge, die das Ausgabetextformat für die Konvertierung von <i>geometry-expression</i> in eine Textdarstellung definiert. Wenn dieser Parameter nicht festgelegt ist, wird die <code>st_geometry_astext_format</code> -Option verwendet, um die Textdarstellung auszuwählen. Siehe „ st_geometry_astext_format-Option “ [<i>SQL Anywhere Server - Datenbankadministration</i>].

Rückgabe

- **LONG VARCHAR** Gibt die Textdarstellung von *geometry-expression* zurück.

Bemerkungen

Die ST_AsText-Methode gibt eine Textzeichenfolge zurück, die die Geometrie darstellt. Verschiedene Textformate werden (mit den zugeordneten Optionen) unterstützt und das gewünschte Format wird mit dem optionalen *format*-Parameter ausgewählt. Wenn der *format*-Parameter nicht angegeben ist, wird die Option `st_geometry_astext_format` zur Auswahl des Ausgabeformats verwendet. Siehe „[st_geometry_astext_format-Option](#)“ [*SQL Anywhere Server - Datenbankadministration*].

Die Formatzeichenfolge definiert ein Ausgabeformat und Parameter für das Format. Die Formatzeichenfolge hat eines der folgenden Formate:

`format-name`

`format-name(parameter1=value1;parameter2=value2;...)`

`parameter1=value1;parameter2=value2;...`

Das erste Format legt den Formatnamen und keine Parameter fest. Alle Formatparameter benutzen ihre Standardwerte. Das zweite Format legt den Formatnamen und eine Liste von Parameterwerten fest. Nicht angegebene Parameter verwenden die Standardwerte. Das letzte Format gibt nur Parameterwerte an, der Formatname verwendet den Standardnamen 'WKT'.

Die folgenden Formatnamen können benutzt werden:

- **WKT** Das Well-Known-Textformat laut der Definition von SQL/MM und OGC.
- **EWKT** Das Extended Well Known-Textformat. Dieses Format enthält die SRID der Geometrie als Präfix.
- **GML** Das "Geography Markup Language"-Format gemäß der Definition in ISO 19136 und OGC.

- **KML** Das Keyhole Markup Language-Format gemäß der OGC-Definition.
- **GeoJSON** Das GeoJSON-Format verwendet JavaScript Object Notation (JSON) gemäß der Definition in <http://geojson.org/geojson-spec.html>.
- **SVG** Das Scalable Vector Graphics (SVG) 1.1-Format gemäß der Definition des World Wide Web Consortium (W3C).

Die folgenden format-Parameter können angegeben werden:

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
WKT	Version	1.2	<ul style="list-style-type: none"> • 1.1 Das WKT gemäß der OGC SFS 1.1-Definition. Dieses Format enthält keine Z- und M-Werte. Wenn die Geometrie Z- oder M-Werte enthält, werden sie in der Ausgabe entfernt. • 1.2 Das WKT gemäß der OGC SFS 1.2-Definition. Dies passt zu Version 1.1 bei 2D-Daten und erweitert das Format zur Unterstützung von Z- und M-Werten. • PostGIS Das WKT-Format in der von anderen Herstellern verwendeten Form. Z- und M-Werte sind in einer Weise enthalten, die nicht zum OGC 1.2 passt. 	Der version-Parameter steuert die Version der benutzten WKT-Spezifikation.
GML	Version	3	<ul style="list-style-type: none"> • 2 Version 2 der GML-Spezifikation. • 3 Version 3.2 der GML-Spezifikation 	Der version-Parameter steuert die Version der benutzten GML-Spezifikation.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
GML	Namespace	none	<ul style="list-style-type: none">• local Stellt ein Standard-Namespace-Attribut für das angegebene Element (in diesem Fall Point) und seine untergeordneten Elemente bereit.• global Stellt ein dediziertes Präfix ("gml") für das angegebene Element und seine untergeordneten Elemente bereit. Dies ist sinnvoll, wenn die Abfrage innerhalb eines Aggregatvorgangs verwendet wird. Bestimmte Elemente der obersten Ebene definieren den Namespace für das "gml"-Präfix.• none Stellt keinen Namespace oder kein Präfix für das angegebene Element (in diesem Fall Point) und seine untergeordneten Elemente bereit.	Der namespace-Parameter legt die Ausgabeformat-Konvention für Namespace fest.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
GML	SRSNameFormat	short	<ul style="list-style-type: none"> • short Verwendet ein kurzes Format für das räumliche Bezugssystem, zum Beispiel EPSG:4326 • long Benutzt ein langes Format für den Namen des räumlichen Bezugssystems, zum Beispiel urn:x-ogc:def:crs:EPSG:4326. • none Das Namensattribut für das räumliche Bezugssystem wird für die Geometrie nicht angegeben. 	Der SRSNameFormat-Parameter legt das Format für das srsName-Attribut fest.
GML	SRSDimension	No	Yes oder No	Der SRSDimension-Parameter legt die Anzahl der Koordinatenwerte für die angegebene Geometrie fest. Dies gilt nur für GML(version=3).
GML	SRSFillAll	No	Yes oder No	Der SRSFillAll-Parameter gibt an, ob SRS-Attribute an untergeordnete Geometrieelemente weitergereicht werden sollen oder nicht. Beispiel: MultiGeometry oder MultiPolygon würde die Attribute an die untergeordneten Geometrien weitergeben.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
GML	UseDeprecated	No	Yes oder No	Der UseDeprecated-Parameter gilt nur für GML(version=3). Er wird gegebenenfalls für die Ausgabe von älteren GML-Darstellungen verwendet. Beispiel: Eine Ebene (Surface-Element) kann als Polygon ausgegeben werden, wenn die Geometrie keine CircularStrings enthält.
GML	Attribute	Automatisch generierte optionale Attribute	Ein oder mehrere Attribute können nur für das Geometrieelement der obersten Ebene angegeben werden.	Alle zulässigen XML-Attribute können angegeben werden.
GML	SubElement	Automatisch generierte untergeordnete GML-Elemente	Ein oder mehrere untergeordnete Attribute können nur für das Geometrieelement der obersten Ebene angegeben werden.	Alle zulässigen XML-Elemente können angegeben werden.
KML	Version	2	2	KML Version 2.2 wird unterstützt.
KML	Attribute	Automatisch generierte optionale Attribute	Ein oder mehrere Attribute können nur für das Geometrieelement der obersten Ebene angegeben werden.	Alle zulässigen XML-Attribute können angegeben werden.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
KML	Namespace	none	<ul style="list-style-type: none"> • local Stellt das Standard Namespace-Attribut http://www.opengis.net/kml/2.2 für das angegebene Geometrieelement (in diesem Fall Point) und seine untergeordneten Elemente bereit. • global Stellt ein dediziertes Präfix ("kml") für das angegebene Element und seine untergeordneten Elemente bereit. Dies ist sinnvoll, wenn die Abfrage innerhalb eines Aggregatvorgangs verwendet wird. Bestimmte Elemente der obersten Ebene definieren den Namespace für das "kml"-Präfix. • none Stellt keinen Namespace oder kein Präfix für das angegebene Element (in diesem Fall Point) und seine untergeordneten Elemente bereit. 	Der namespace-Parameter legt die Ausgabeformat-Konvention für Namespace fest.
KML	SubElement	Automatisch generierte untergeordnete KML-Elemente	Ein oder mehrere untergeordnete Attribute können nur für das Geometrieelement der obersten Ebene angegeben werden.	Alle zulässigen XML-Elemente können angegeben werden. Beispiel: extrude, tessellate und altitudeMode-Elemente können angegeben werden.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
GeoJSON	Version	1	1	Die Version der GeoJSON-Spezifikation, die verwendet werden muss. Derzeit wird nur 1.0 unterstützt.
SVG	Annähern	Yes	Yes oder No	Der Parameter "Approximate" gibt an, ob die Größe des ausgegebenen SVG-Dokuments mit einer geringfügigen Verringerung der sichtbaren Details reduziert werden soll oder nicht. Die SVG-Daten werden als Näherungswert ("Approximate") dargestellt, indem Punkte nicht einbezogen werden, die sich innerhalb der Linienbreite des letzten Punkts befinden. Bei Geometrien mit mehreren Megabyte kann dies zu Kompressionsraten von 80 % oder mehr führen.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
SVG	Attribute	Automatisch generierte optionale Attribute	Ein oder mehr SVG-Attribute, die an SVG-Formelemente angewendet werden können	Standardmäßig werden optionale SVG-Formattribute wie "fill", "stroke" und "stroke-width" generiert. Wenn der Attribute-Parameter angegeben ist, werden keine optionalen SVG-Formattribute generiert und der Attribute-Wert wird stattdessen verwendet. Wird ignoriert, wenn PathDataOnly=Yes festgelegt ist. Die maximale Länge des Attribute-Werts beträgt etwa 1000 Byte.
SVG	DecimalDigits	Basierend auf der Anzahl von Dezimalstellen der Rastergröße für die Funktion "Am Raster ausrichten" des räumlichen Bezugssystems. Der maximale Standardwert ist 5 und der Mindestwert ist 0 MB.	integer	Der DecimalDigits-Parameter begrenzt die Anzahl der Ziffern hinter dem Dezimalzeichen für Koordinaten, die in der SVG-Ausgabe generiert werden. Eine negative Stellenanzahl zeigt an, dass die Gesamtstellenzahl der Koordinaten in die SVG-Ausgabe einbezogen werden soll.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
SVG	PathDataOnly	No (ein vollständiges SVG-Dokument wird generiert)	Yes oder No	Der PathDataOnly-Parameter legt fest, ob nur die Daten für das SVG-Pfadelement generiert werden sollen. Das nachstehende PathDataOnly-Beispiel zeigt, wie PathDataOnly=Yes verwendet werden kann, um ein vollständiges SVG-Dokument zu erstellen, das angezeigt werden kann. Standardmäßig wird ein vollständiges SVG-Dokument generiert. Die von PathDataOnly=Yes zurückgegebenen Pfaddaten können verwendet werden, um flexiblere SVG-Dokumente zu erstellen, die andere Elemente enthalten, beispielsweise Text.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
SVG	RandomFill	Yes	Yes oder No	Der RandomFill-Parameter gibt an, ob Polygone mit einer zufallsgenerierten Farbe gefüllt werden sollen. Die verwendete Farbensequenz hält sich nicht an eine bestimmte Reihenfolge und ändert sich in der Regel jedes Mal, wenn eine SVG-Ausgabe generiert wird. None legt fest, dass nur ein Umriss jedes Polygons ausgegeben wird. Der RandomFill-Parameter wird ignoriert, wenn der Attribute-oder PathDataOnly=Yes-Parameter angegeben ist.
SVG	Relative	Yes	Yes oder No	Der Relative-Parameter legt fest, ob Koordinaten im relativen Format (Offset) oder absoluten Format ausgegeben werden sollen. Relative Koordinatendaten sind in der Regel kompakter als absolute Koordinatendaten.

Hinweis

Beim Konvertieren eines Geometriewerts in VARCHAR oder NVARCHAR verwendet der Server die ST_AsText-Methode. Die st_geometry_astext_format-Option definiert das Format für die Konvertierung. Siehe „st_geometry_astext_format-Option“ [[SQL Anywhere Server - Datenbankadministration](#)].

Hinweis

Standardmäßig verwendet ST_AsText das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_AsGeoJSON-Methode](#) für den ST_Geometry-Datentyp
- [ST_AsGML-Methode](#) für den ST_Geometry-Datentyp
- [ST_AsKML-Methode](#) für den ST_Geometry-Datentyp
- [ST_AsSVG-Methode](#) für den ST_Geometry-Datentyp
- [ST_AsWKT-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.35

Beispiel

Unter der Annahme, dass die st_geometry_astext_format-Option den Wert "WKT" hat, gibt die folgende Anweisung das Ergebnis Point ZM (1 2 3 4) zurück. Siehe „st_geometry_astext_format-Option“ [*SQL Anywhere Server - Datenbankadministration*].

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsText()
```

Unter der Annahme, dass die st_geometry_astext_format-Option den Wert "WKT" hat, gibt die folgende Anweisung das Ergebnis Point ZM (1 2 3 4) zurück. Die ST_AsText-Methode wird implizit aufgerufen, wenn Geometrien in VARCHAR oder NVARCHAR-Datentypen konvertiert werden. Siehe „st_geometry_astext_format-Option“ [*SQL Anywhere Server - Datenbankadministration*].

```
SELECT CAST( NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ) as long varchar)
```

Die folgende Anweisung gibt die Ergebnismenge Point (1 2) zurück. Die Z- und M-Werte werden nicht ausgegeben, weil sie in Version 1.1.0 der OGC-Spezifikation für WKT nicht unterstützt werden.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsText('WKT(Version=1.1)')
```

Die folgende Anweisung liefert das Ergebnis SRID=4326;Point ZM (1 2 3 4). Die SRID ist in der Ergebnismenge als Präfix enthalten.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsText('EWKT')
```

Das folgende Beispiel gibt den Wert <Point srsName="EPSG:4326"><pos>1 2 3 4</pos></Point> zurück.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsText('GML')
```

Die folgende Anweisung gibt '{ "type": "Point", "coordinates": [1,2] } ' zurück.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsText('GeoJSON')
```

Der folgende Code gibt ein vollständiges SVG-Dokument mit Polygonen zurück, die mit zufallsgenerierten Farben gefüllt sind.

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 0 20, 60 10, 0 0 ))' )
      .ST_AsText( 'SVG' )
```

ST_AsWKB-Methode

Gibt die WKB-Darstellung eines ST_Geometry Werts zurück.

Syntax

geometry-expression.**ST_AsWKB**([*format*])

Parameter

Name	Typ	Beschreibung
format	VAR-CHAR(128)	Eine Zeichenfolge, die das WKB-Format für die Konvertierung von <i>geometry-expression</i> in eine binäre Darstellung definiert. Wenn sie nicht festgelegt wird, ist der Standardwert 'WKB'.

Rückgabe

- **LONG BINARY** Gibt die WKB-Darstellung von *geometry-expression* zurück.

Bemerkungen

Die ST_AsWKB-Methode gibt eine Binärzeichenfolge zurück, die die Geometrie im WKB-Format darstellt. Verschiedene Formate werden unterstützt (mit den dazugehörigen Optionen) und das gewünschte Format wird mit dem optionalen *format*-Parameter ausgewählt. Wenn der *format*-Parameter nicht angegeben wird, ist der Standardwert 'WKB'.

Die Formatzeichenfolge definiert ein Ausgabeformat und Parameter für das Format. Die Formatzeichenfolge hat eines der folgenden Formate:

```
format-name
```

```
format-name(parameter1=value1;parameter2=value2;...)
```

```
parameter1=value1;parameter2=value2;...
```

Das erste Format legt den Formatnamen und keine Parameter fest. Alle Formatparameter benutzen ihre Standardwerte. Das zweite Format legt den Formatnamen und eine Liste von Parameterwerten fest. Nicht angegebene Parameter verwenden die Standardwerte. Das letzte Format gibt nur Parameterwerte an, der Formatname verwendet den Standardnamen 'WKB'.

Die folgenden Formatnamen können benutzt werden:

- **WKB** Das Well-Known-Binärformat laut der Definition von SQL/MM und OGC.
- **EWKB** Das erweiterte Well-Known-Binärformat laut der Definition von PostGIS. Dieses Format enthält die SRID der Geometrie und unterscheidet sich vom WKB durch die Darstellung der Z- und M-Werte.

Die folgenden format-Parameter können angegeben werden:

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
WKB	Version	1.2	<ul style="list-style-type: none">• 1.1 Das WKB gemäß der OGC SFS 1.1-Definition. Dieses Format enthält keine Z- und M-Werte. Wenn die Geometrie Z- oder M-Werte enthält, werden sie in der Ausgabe entfernt.• 1.2 Das WKB gemäß der OGC SFS 1.2-Definition. Dies passt zu Version 1.1 bei 2D-Daten und erweitert das Format zur Unterstützung von Z- und M-Werten.	Der version-Parameter steuert die Version der benutzten WKB-Spezifikation.

Hinweis
Standardmäßig verwendet ST_AsWKB das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert
0x01b90b000000000000000000f03f000000000000004000000000000008400000000000
001040 zurück.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsWKB('endian=little')
```

Die folgende Anweisung gibt die Ergebnismenge
0x010100000000000000000000f03f0000000000000040 zurück. Die Z- und M-Werte werden
ausgeklammert, weil die Version 1.1 der OGC-Spezifikation für WKB keine Unterstützung dafür bietet.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0,  
4326 ).ST_AsWKB('WKB(Version=1.1;endian=little)')
```

Die folgende Anweisung gibt die Ergebnismenge
0x01010000e0e61000000000000000f03f00000000000000400000000000084000
00000000001040 zurück.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0,  
4326 ).ST_AsWKB('EWKB(endian=little)')
```

ST_AsWKT-Methode

Gibt die WKT-Darstellung eines ST_Geometry-Werts zurück.

Syntax

geometry-expression.**ST_AsWKT**([*format*])

Parameter

Name	Typ	Beschreibung
format	VAR-CHAR(128)	Eine Zeichenfolge, die das Ausgabeformat für die Konvertierung von <i>geometry-expression</i> in WKT definiert. Wenn sie nicht angegeben wird, ist die Formatzeichenfolge standardmäßig 'WKT'.

Rückgabe

- **LONG VARCHAR** Gibt die WKT-Darstellung von *geometry-expression* zurück.

Bemerkungen

Die ST_AsWKT-Methode gibt eine Textzeichenfolge für die Geometrie zurück. Verschiedene Textformate werden (mit den zugeordneten Optionen) unterstützt und das gewünschte Format wird mit dem optionalen *format*-Parameter ausgewählt.

Die Formatzeichenfolge definiert ein Ausgabeformat und Parameter für das Format. Die Formatzeichenfolge hat eines der folgenden Formate:

```
format-name
```

```
format-name(parameter1=value1;parameter2=value2;...)
```

```
parameter1=value1;parameter2=value2;...
```

Das erste Format legt den Formatnamen und keine Parameter fest. Alle Formatparameter benutzen ihre Standardwerte. Das zweite Format legt den Formatnamen und eine Liste von Parameterwerten fest. Nicht angegebene Parameter verwenden die Standardwerte. Das letzte Format gibt nur Parameterwerte an, der Formatname verwendet den Standardnamen 'WKT'.

Die folgenden Formatnamen können benutzt werden:

- **WKT** Das Well-Known-Textformat laut der Definition von SQL/MM und OGC.
- **EWKT** Das Extended Well-Known-Textformat gemäß der Definition in PostGIS. Dieses Format enthält die SRID der Geometrie und unterscheidet sich vom WKT durch die Darstellung der Z- und M-Werte.

Die folgenden format-Parameter können angegeben werden:

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
WKT	Version	1.2	<ul style="list-style-type: none">• 1.1 Das WKT gemäß der OGC SFS 1.1-Definition. Dieses Format enthält keine Z- und M-Werte. Wenn die Geometrie Z- oder M-Werte enthält, werden sie in der Ausgabe entfernt.• 1.2 Das WKT gemäß der OGC SFS 1.2-Definition. Dies passt zu Version 1.1 bei 2D-Daten und erweitert das Format zur Unterstützung von Z- und M-Werten.• PostGIS Das WKT-Format in der von anderen Herstellern verwendeten Form. Z- und M-Werte sind in einer Weise enthalten, die nicht zum OGC 1.2 passt.	Der version-Parameter steuert die Version der benutzten WKT-Spezifikation.

Hinweis
Standardmäßig verwendet ST_AsWKT das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert SRID=0;Polygon ((3 3, 8 3, 4 8, 3 3)) zurück.

```
SELECT Shape.ST_AsWKT( 'EWKT' ) FROM SpatialShapes WHERE ShapeID = 22
```

ST_AsXML-Methode

Gibt die XML-Darstellung eines ST_Geometry-Werts zurück.

Syntax

geometry-expression.**ST_AsXML**([*format*])

Parameter

Name	Typ	Beschreibung
format	VAR-CHAR(128)	Eine Zeichenfolge, die das Ausgabetextformat für die Konvertierung von <i>geometry-expression</i> in eine XML-Textdarstellung definiert. Wenn dieser Parameter nicht festgelegt ist, wird die <code>st_geometry_asxml_format</code> -Option verwendet, um die XML-Darstellung auszuwählen. Siehe „ st_geometry_asxml_format-Option “ [<i>SQL Anywhere Server - Datenbankadministration</i>].

Rückgabe

- **LONG VARCHAR** Gibt die XML-Darstellung von *geometry-expression* zurück.

Bemerkungen

Die `ST_AsXML`-Methode gibt eine XML-Zeichenfolge für die Geometrie zurück. GML, KML und SVG sind die unterstützten XML-Formate. Der *format*-Parameter legt Parameter für die Steuerung der Konvertierung in XML fest. Wenn *format* nicht angegeben ist, wird der Wert der `st_geometry_asxml_format`-Option verwendet, um das Ausgabeformat auszuwählen. Siehe „[st_geometry_asxml_format-Option](#)“ [*SQL Anywhere Server - Datenbankadministration*].

Die Formatzeichenfolge definiert ein Ausgabeformat und Parameter für das Format. Die Formatzeichenfolge hat eines der folgenden Formate:

`format-name`

`format-name(parameter1=value1;parameter2=value2;...)`

`parameter1=value1;parameter2=value2;...`

Das erste Format legt den Formatnamen und keine Parameter fest. Alle Formatparameter benutzen ihre Standardwerte. Das zweite Format legt den Formatnamen und eine Liste von Parameterwerten fest. Nicht angegebene Parameter verwenden die Standardwerte. Das letzte Format gibt nur Parameterwerte an, der Formatname verwendet den Standardnamen 'GML'.

Die folgenden Formatnamen können benutzt werden:

- **GML** Das "Geography Markup Language"-Format gemäß der Definition in ISO 19136 und OGC.
- **KML** Das Keyhole Markup Language-Format gemäß Definition von OGC.
- **SVG** Das Scalable Vector Graphics (SVG) 1.1-Format gemäß der Definition des World Wide Web Consortium (W3C).

Die folgenden format-Parameter können angegeben werden:

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
GML	Version	3	<ul style="list-style-type: none"> • 2 Version 2 der GML-Spezifikation. • 3 Version 3.2 der GML-Spezifikation 	Der version-Parameter steuert die Version der benutzten GML-Spezifikation.
GML	Namespace	none	<ul style="list-style-type: none"> • local Stellt ein Standard-Namespace-Attribut für das angegebene Element (in diesem Fall Point) und seine untergeordneten Elemente bereit. • global Stellt ein dediziertes Präfix ("gml") für das angegebene Element und seine untergeordneten Elemente bereit. Dies ist sinnvoll, wenn die Abfrage innerhalb eines Aggregatvorgangs verwendet wird. Bestimmte Elemente der obersten Ebene definieren den Namespace für das "gml"-Präfix. • none Stellt keinen Namespace oder kein Präfix für das angegebene Element (in diesem Fall Point) und seine untergeordneten Elemente bereit. 	Der namespace-Parameter legt die Ausgabeformat-Konvention für Namespace fest.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
GML	SRSNameFormat	short	<ul style="list-style-type: none"> • short Verwendet ein kurzes Format für das räumliche Bezugssystem, zum Beispiel EPSG:4326 • long Benutzt ein langes Format für den Namen des räumlichen Bezugssystems, zum Beispiel urn:x-ogc:def:crs:EPSG:4326. • none Das Namensattribut für das räumliche Bezugssystem wird für die Geometrie nicht angegeben. 	Der SRSNameFormat-Parameter legt das Format für das srsName-Attribut fest.
GML	SRSDimension	No	Yes oder No	Der SRSDimension-Parameter legt die Anzahl der Koordinatenwerte für die angegebene Geometrie fest. Dies gilt nur für GML(version=3).
GML	SRSFillAll	No	Yes oder No	Der SRSFillAll-Parameter gibt an, ob SRS-Attribute an untergeordnete Geometrieelemente weitergereicht werden sollen oder nicht. Beispiel: MultiGeometry oder MultiPolygon würde die Attribute an die untergeordneten Geometrien weitergeben.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
GML	UseDeprecated	No	Yes oder No	Der UseDeprecated-Parameter gilt nur für GML(version=3). Er wird gegebenenfalls für die Ausgabe von älteren GML-Darstellungen verwendet. Beispiel: Eine Ebene (Surface-Element) kann als Polygon ausgegeben werden, wenn die Geometrie keine CircularStrings enthält.
GML	Attribute	Automatisch generierte optionale Attribute	Ein oder mehrere Attribute können nur für das Geometrieelement der obersten Ebene angegeben werden.	Alle zulässigen XML-Attribute können angegeben werden.
GML	SubElement	Automatisch generierte untergeordnete GML-Elemente	Ein oder mehrere untergeordnete Attribute können nur für das Geometrieelement der obersten Ebene angegeben werden.	Alle zulässigen XML-Elemente können angegeben werden.
KML	Version	2	2	KML Version 2.2 wird unterstützt.
KML	Attribute	Automatisch generierte optionale Attribute	Ein oder mehrere Attribute können nur für das Geometrieelement der obersten Ebene angegeben werden.	Alle zulässigen XML-Attribute können angegeben werden.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
KML	Namespace	none	<ul style="list-style-type: none"> • local Stellt das Standard Namespace-Attribut http://www.opengis.net/kml/2.2 für das angegebene Geometrieelement (in diesem Fall Point) und seine untergeordneten Elemente bereit. • global Stellt ein dediziertes Präfix ("kml") für das angegebene Element und seine untergeordneten Elemente bereit. Dies ist sinnvoll, wenn die Abfrage innerhalb eines Aggregatvorgangs verwendet wird. Bestimmte Elemente der obersten Ebene definieren den Namespace für das "kml"-Präfix. • none Stellt keinen Namespace oder kein Präfix für das angegebene Element (in diesem Fall Point) und seine untergeordneten Elemente bereit. 	Der namespace-Parameter legt die Ausgabeformat-Konvention für Namespace fest.
KML	SubElement	Automatisch generierte untergeordnete KML-Elemente	Ein oder mehrere untergeordnete Attribute können nur für das Geometrieelement der obersten Ebene angegeben werden.	Alle zulässigen XML-Elemente können angegeben werden. Beispiel: extrude, tessellate und altitudeMode-Elemente können angegeben werden.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
SVG	Annähern	Yes	Yes oder No	Der Parameter "Approximate" gibt an, ob die Größe des ausgegebenen SVG-Dokuments mit einer geringfügigen Verringerung der sichtbaren Details reduziert werden soll oder nicht. Die SVG-Daten werden als Näherungswert ("Approximate") dargestellt, indem Punkte nicht einbezogen werden, die sich innerhalb der Linienbreite des letzten Punkts befinden. Bei Geometrien mit mehreren Megabyte kann dies zu Kompressionsraten von 80 % oder mehr führen.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
SVG	Attribute	Automatisch generierte optionale Attribute	Ein oder mehr SVG-Attribute, die an SVG-Formelemente angewendet werden können	Standardmäßig werden optionale SVG-Formattribute wie "fill", "stroke" und "stroke-width" generiert. Wenn der Attribute-Parameter angegeben ist, werden keine optionalen SVG-Formattribute generiert und der Attribute-Wert wird stattdessen verwendet. Wird ignoriert, wenn PathDataOnly=Yes festgelegt ist. Die maximale Länge des Attribute-Werts beträgt etwa 1000 Byte.
SVG	DecimalDigits	Basierend auf der Anzahl von Dezimalstellen der Rastergröße für die Funktion "Am Raster ausrichten" des räumlichen Bezugssystems. Der maximale Standardwert ist 5 und der Mindestwert ist 0 MB.	integer	Der DecimalDigits-Parameter begrenzt die Anzahl der Ziffern hinter dem Dezimalzeichen für Koordinaten, die in der SVG-Ausgabe generiert werden. Eine negative Stellenanzahl zeigt an, dass die Gesamtstellenzahl der Koordinaten in die SVG-Ausgabe einbezogen werden soll.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
SVG	PathDataOnly	No (ein vollständiges SVG-Dokument wird generiert)	Yes oder No	Der PathDataOnly-Parameter legt fest, ob nur die Daten für das SVG-Pfadelement generiert werden sollen. Das nachstehende PathDataOnly-Beispiel zeigt, wie PathDataOnly=Yes verwendet werden kann, um ein vollständiges SVG-Dokument zu erstellen, das angezeigt werden kann. Standardmäßig wird ein vollständiges SVG-Dokument generiert. Die von PathDataOnly=Yes zurückgegebenen Pfaddaten können verwendet werden, um flexiblere SVG-Dokumente zu erstellen, die andere Elemente enthalten, beispielsweise Text.

Formatname	Parametername	Standardwert	Zulässige Werte	Beschreibung
SVG	RandomFill	Yes	Yes oder No	Der RandomFill-Parameter gibt an, ob Polygone mit einer zufallsgenerierten Farbe gefüllt werden sollen. Die verwendete Farbensequenz hält sich nicht an eine bestimmte Reihenfolge und ändert sich in der Regel jedes Mal, wenn eine SVG-Ausgabe generiert wird. None legt fest, dass nur ein Umriss jedes Polygons ausgegeben wird. Der Random-Fill-Parameter wird ignoriert, wenn der Attribute-oder Path-DataOnly=Yes-Parameter angegeben ist.
SVG	Relative	Yes	Yes oder No	Der Relative-Parameter legt fest, ob Koordinaten im relativen Format (Offset) oder absoluten Format ausgegeben werden sollen. Relative Koordinatendaten sind in der Regel kompakter als absolute Koordinatendaten.

Hinweis

Beim Konvertieren eines Geometriewerts in XML verwendet der Server die ST_AsXML-Methode. Die st_geometry_asxml_format-Option definiert das Format für die Konvertierung. Siehe „st_geometry_asxml_format-Option“ [[SQL Anywhere Server - Datenbankadministration](#)].

Hinweis

Standardmäßig verwendet ST_AsXML das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_AsGML-Methode](#) für den ST_Geometry-Datentyp
- [ST_AsKML-Methode](#) für den ST_Geometry-Datentyp
- [ST_AsSVG-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Wenn die `st_geometry_asxml_format`-Option als Standardwert 'GML' hat, gibt die folgende Anweisung das Ergebnis `<Point srsName="EPSG:4326"><pos>1 2 3 4</pos></Point>` zurück.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsXML()
```

Wenn die `st_geometry_asxml_format`-Option als Standardwert 'GML' hat, gibt die folgende Anweisung das Ergebnis `<Point srsName="EPSG:4326"><pos>1 2 3 4</pos></Point>` zurück.

```
SELECT CAST( NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ) AS XML)
```

Das folgende Beispiel gibt den Wert `<Point srsName="EPSG:4326"><coordinates>1,2</coordinates></Point>` zurück.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsXML('GML(Version=2)')
```

Der folgende Code gibt ein vollständiges SVG-Dokument mit Polygonen zurück, die mit zufallsgenerierten Farben gefüllt sind.

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 0 20, 60 10, 0 0 ))' )  
      .ST_AsXML( 'SVG' )
```

ST_Boundary-Methode

Gibt die Begrenzung des Domänenwerts zurück

Syntax

geometry-expression.**ST_Boundary**()

Rückgabe

- **ST_Geometry** Gibt einen Geometriewert zurück, der die Begrenzung von *geometry-expression* darstellt.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Die ST_Boundary-Methode gibt die räumliche Begrenzung von *geometry-expression* zurück. Geometrien werden durch den Innenbereich, die Begrenzung und den Außenbereich gekennzeichnet. Alle Geometriewerte werden als topologisch geschlossen definiert, das bedeutet, dass die Begrenzung Teil der Geometrie ist.

Punktgeometrien haben eine leere Begrenzung. Kurvengeometrien können geschlossen sein und haben in diesem Fall eine leere Begrenzung. Wenn eine Kurve nicht geschlossen ist, bilden der Start- und Endpunkt die Begrenzung. Bei einer Flächengeometrie ist die Begrenzung die Menge von Kurven, die den Rand der Fläche darstellen. Beispiel: Für ein Polygon besteht die Begrenzung der Geometrie aus dem äußeren Ring und inneren Ringen.

Siehe auch: „[Funktionsweise von Innen- und Außenbereichen sowie von Begrenzungen von Geometrien](#)“ auf Seite 51.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.14

Beispiel

Im folgenden Beispiel wird zunächst eine Geometriesammlung erstellt, die aus einem Polygon und einer Linienfolge besteht, und dann die Begrenzung für die Sammlung zurückgegeben. Die zurückgegebene Begrenzung ist eine Sammlung, die den äußeren Ring des Polygons und die beiden Endpunkte der Linienfolge enthält. Sie ist äquivalent zu der folgenden Sammlung: 'GeometryCollection (LineString (0 0, 3 0, 3 3, 0 3, 0 0), MultiPoint ((0 7), (4 4)))'

```
SELECT NEW ST_GeomCollection('GeometryCollection (Polygon ((0 0, 3 0, 3 3, 0 3, 0 0)), LineString (0 7, 0 4, 4 4))').ST_Boundary()
```

ST_Contains-Methode

Testet, ob ein räumlicher Geometriewert einen anderen räumlichen Geometriewert enthält.

Syntax

geometry-expression.ST_Contains(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* den Wert *geo2* enthält, sonst 0.

Bemerkungen

Die ST_Contains-Methode prüft, ob *geometry-expression* den Wert *geo2* vollständig enthält und mindestens ein innerer Punkt von *geo2* vorhanden ist, der im Inneren von *geometry-expression* liegt.

geometry-expression.ST_Contains(*geo2*) ist gleichwertig mit *geo2*.ST_Within(*geometry-expression*).

Die ST_Contains- und ST_Covers-Methoden sind ähnlich. Der Unterschied liegt darin, dass ST_Covers innere Punkte nicht schneiden muss.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

- [ST_Within-Methode für den ST_Geometry-Datentyp](#)
- [ST_Covers-Methode für den ST_Geometry-Datentyp](#)
- [ST_Intersects-Methode für den ST_Geometry-Datentyp](#)
- [ST_ContainsFilter-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.31

Beispiel

Mit dem folgenden Beispiel wird getestet, ob ein Polygon einen Punkt enthält. Das Polygon enthält den Punkt vollständig und das Innere des Punkts (der Punkt selbst) schneidet den Innenbereich des Polygons, daher gibt das Beispiel 1 zurück.

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' )
.ST_Contains( NEW ST_Point( 1, 1 ) )
```

Mit dem folgenden Beispiel wird getestet, ob ein Polygon eine Linie enthält. Das Polygon enthält die Linie vollständig, aber das Innere der Linie und das Innere des Polygons haben keine Schnittmenge (die Linie schneidet das Polygon nur an der Begrenzung des Polygons und die Begrenzung ist nicht Teil des

Inneren), daher gibt das Beispiel 0 zurück. Wenn ST_Covers anstelle von ST_Contains verwendet wurde, gibt ST_Covers 1 zurück.

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' )
       .ST_Contains( NEW ST_LineString( 'LineString( 0 0, 1 0 )' ) )
```

Das folgende Beispiel enthält eine Liste der ShapeIDs, in denen das gegebene Polygon alle Shape-Geometrien enthält. Die Anweisung in diesem Beispiel gibt das Ergebnis 16,17,19 zurück. ShapeID 1 ist nicht aufgelistet, weil das Polygon den Shape-Punkt dieser Zeile an der Begrenzung des Polygons schneidet.

```
SELECT LIST( ShapeID ORDER BY ShapeID )
FROM SpatialShapes
WHERE NEW ST_Polygon( NEW ST_Point( 0, 0 ),
                      NEW ST_Point( 8, 2 ) ).ST_Contains( Shape ) = 1
```

ST_ContainsFilter-Methode

Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie eine andere enthält.

Syntax

geometry-expression.**ST_ContainsFilter**(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* den Wert *geo2* enthalten könnte, sonst 0.

Bemerkungen

Die ST_ContainsFilter-Methode bietet einen effizienten Test, um zu ermitteln, ob eine Geometrie eine andere enthalten kann. Gibt 1 zurück, wenn *geometry-expression* den Wert *geo2* enthalten könnte, sonst 0.

Dieser Test ist weniger kostenträchtig als ST_Contains, kann aber in manchen Fällen 1 zurückgeben, in denen *geometry-expression* den Wert *geo2* nicht enthält.

Aus diesem Grund kann diese Methode als Primärfilter nützlich sein, wenn die weitere Verarbeitung erkennen lässt, ob Geometrien auf gewünschte Art und Weise interagieren.

Die Implementierung von ST_ContainsFilter beruht auf Metadaten, die den gespeicherten Geometrien zugeordnet sind. Da sich die verfügbaren Metadaten je nachdem, wie Daten geladen werden oder wo ST_ContainsFilter in einer Abfrage verwendet wird, von Serverversion zu Serverversion verändern können, kann der Ausdruck *geometry-expression*.ST_ContainsFilter(*geo2*) unterschiedliche Ergebnisse zurückgeben, wenn *geometry-expression* nicht den Wert *geo2* enthält. Wenn *geometry-expression* den Wert *geo2* enthält, gibt ST_ContainsFilter immer 1 zurück.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

- [ST_Contains-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

ST_ConvexHull-Methode

Gibt die konvexe Hülle des Geometriewerts zurück.

Syntax

geometry-expression.**ST_ConvexHull()**

Rückgabe

- **ST_Geometry** Wenn der Geometriewert NULL oder ein leerer Wert ist, wird NULL zurückgegeben. Andernfalls wird die konvexe Hülle des Geometriewerts zurückgegeben.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Die konvexe Hülle einer Geometrie ist die kleinste konvexe Geometrie, die alle Punkte in der Geometrie enthält.

Die konvexe Hülle kann man sich als Gummiband vorstellen, das gedehnt wird, um alle Punkte in der Geometrie zu umspannen. Wenn das Gummiband losgelassen wird, nimmt es die Form der konvexen Hülle an.

Wenn die Geometrie aus einem einzelnen Punkt besteht, wird der Punkt zurückgegeben. Wenn alle Punkte der Geometrie in einem einzigen geraden Liniensegment liegen, wird eine Linienfolge zurückgegeben. Andernfalls wird ein konvexes Polygon zurückgegeben.

Die konvexe Hülle kann als Näherungswert der ursprünglichen Geometrie dienen. Beim Testen einer räumlichen Beziehung kann die konvexe Hülle als schneller Vorfilter herangezogen werden: Wenn keine räumliche Schnittmenge mit der konvexen Hülle vorliegt, kann es keine Schnittmenge mit der ursprünglichen Geometrie geben.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

ST_ConvexHull wird nicht für Geometrien unterstützt, die Kreisbogenfolgen enthalten.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

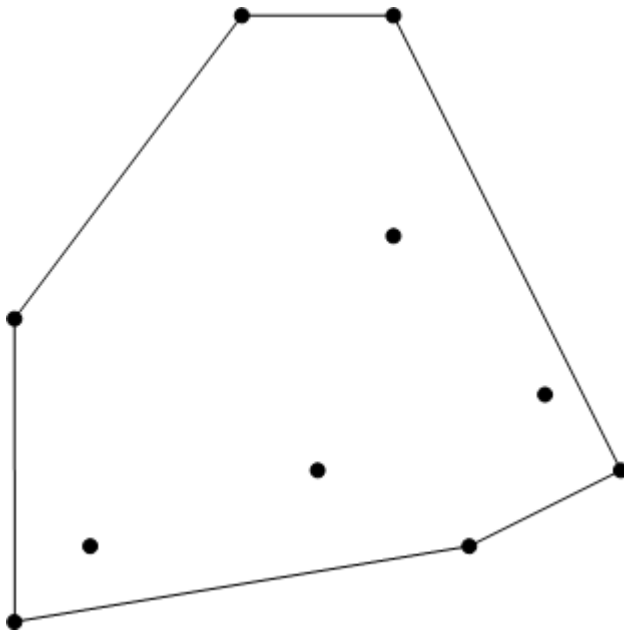
- [ST_ConvexHullAggr-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.16

Beispiel

Im folgenden Beispiel wird die konvexe Hülle gezeigt, die aus 10 Punkten berechnet wird. Die sich daraus ergebende Hülle ist das Ergebnis `Polygon ((1 1, 7 2, 9 3, 6 9, 4 9, 1 5, 1 1))`.



```
SELECT NEW ST_MultiPoint('MultiPoint( (1 1), (2 2), (5 3), (7 2), (9 3), (8 4), (6 6), (6 9), (4 9), (1 5) )').ST_ConvexHull()
```

Mit dem folgenden Beispiel wird der einzelne Punkt (0,0) zurückgegeben. Die konvexe Hülle eines einzelnen Punkts ist ein Punkt.

```
SELECT NEW ST_Point(0,0).ST_ConvexHull()
```

Das folgende Beispiel gibt den Wert `LineString (0 0, 3 3)` zurück. Die konvexe Hülle einer einzelnen geraden Linie ist eine Linienfolge mit einem einzelnen Segment.

```
SELECT NEW ST_LineString('LineString(0 0,1 1,2 2,3 3)').ST_ConvexHull()
```

ST_ConvexHullAggr-Methode

Gibt die konvexe Hülle für alle Geometrien in einer Gruppe zurück

Syntax

ST_Geometry::ST_ConvexHullAggr(*geometry-column*)

Parameter

Name	Typ	Beschreibung
geometry-column	ST_Geometry	Die Geometriewerte zur Generierung der konvexen Hülle. Das ist üblicherweise eine Spalte.

Rückgabe

- **ST_Geometry** Gibt die konvexe Hülle für alle Geometrien in einer Gruppe zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist derselbe wie der für den ersten Parameter.

Bemerkungen

Die `ST_ConvexHullAggr`-Methode berücksichtigt alle Punkte in der Gruppe von Geometrien, aus der die Berechnung erfolgt, und gibt die konvexe Hülle aller dieser Punkte zurück. Die konvexe Hülle einer Geometrie ist die kleinste konvexe Geometrie, die alle Punkte in der Geometrie enthält.

Die konvexe Hülle kann man sich als Gummiband vorstellen, das gedehnt wird, um alle Punkte in der Geometrie zu umspannen. Wenn das Gummiband losgelassen wird, nimmt es die Form der konvexen Hülle an.

Wenn die Geometrien in der Gruppe nur aus einem einzigen Punkt bestehen, wird der Punkt zurückgegeben. Wenn alle Punkte einer Gruppe von Geometrien in einem einzigen geraden Liniensegment liegen, wird eine Linienfolge zurückgegeben. Andernfalls wird ein konvexes Polygon zurückgegeben.

Die konvexe Hülle kann als Näherungswert der ursprünglichen Geometrie dienen. Beim Testen einer räumlichen Beziehung kann die konvexe Hülle als schneller Vorfilter herangezogen werden: Wenn keine räumliche Schnittmenge mit der konvexen Hülle vorliegt, kann es keine Schnittmenge mit der ursprünglichen Geometrie geben.

Hinweis

`ST_ConvexHullAggr` wird nicht für Geometrien unterstützt, die Kreisbogenfolgen enthalten.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

- [ST_ConvexHull-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert Polygon ((3 0, 7 2, 3 6, 0 7, -3 6, -3 3, 0 0, 3 0)) zurück.

```
SELECT ST_Geometry::ST_ConvexHullAggr( Shape )
FROM SpatialShapes WHERE ShapeID <= 16
```

ST_CoordDim-Methode

Gibt die Anzahl der Koordinatendimensionen zurück, die für jeden Punkt des ST_Geometry-Werts gespeichert sind.

Syntax

geometry-expression.**ST_CoordDim()**

Rückgabe

- **SMALLINT** Gibt einen Wert zwischen 2 und 4 zurück, der die Anzahl der Koordinatendimensionen angibt, die für jeden Punkt des ST_Geometry-Werts gespeichert sind.

Bemerkungen

Die ST_CoordDim-Methode gibt die Anzahl der Koordinaten zurück, die für jeden Punkt der Geometrie gespeichert sind. Alle Geometrien haben mindestens zwei Koordinatendimensionen. Für geografische räumliche Bezugssysteme sind dies der Breitengrad und Längengrad des Punkts. Bei anderen räumlichen Bezugssystemen sind diese Koordinaten die X- und Y-Positionen des Punkts.

Den Geometrien können für jeden Punkt in der Geometrie optional auch Z- und M-Werte zugewiesen werden. Diese zusätzlichen Koordinatenwerte werden nicht berücksichtigt, wenn räumliche Bezüge oder Set-Operationen berechnet werden, können aber verwendet werden, um zusätzliche Informationen aufzuzeichnen. Beispiel: Der Messwert (M) kann verwendet werden, um die Verschmutzung an verschiedenen Punkten in einer Geometrie zu messen. Der Z-Wert wird in der Regel verwendet, um eine Erhebung anzuzeigen, aber diese Interpretation wird vom Datenbankserver nicht erzwungen.

Die folgenden Werte können von der ST_CoordDim-Methode zurückgegeben werden:

- **2** Die Geometrie enthält nur zwei Koordinaten (Breitengrad/Längengrad oder X/Y).

- **3** Die Geometrie enthält eine zusätzliche Koordinate (Z oder M) für jeden Punkt.
- **4** Die Geometrie enthält zwei weitere Koordinaten (Z und M) für jeden Punkt.

Hinweis

Räumliche Operationen, die mit Set-Operationen Geometrien kombinieren, behalten keine Z- oder M-Werte, die mit den Punkten der Geometrie verbunden sind.

Hinweis

Standardmäßig verwendet ST_CoordDim das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Siehe auch

- [ST_Is3D-Methode für den ST_Geometry-Datentyp](#)
- [ST_IsMeasured-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.3

Beispiel

Das folgende Beispiel gibt den Wert 2 zurück.

```
SELECT NEW ST_Point(1.0, 1.0).ST_CoordDim()
```

Das folgende Beispiel gibt den Wert 3 zurück.

```
SELECT NEW ST_Point(1.0, 1.0, 1.0, 0).ST_CoordDim()
```

Das folgende Beispiel gibt den Wert 3 zurück.

```
SELECT NEW ST_Point('Point M (1 1 1)' ).ST_CoordDim()
```

Das folgende Beispiel gibt den Wert 4 zurück.

```
SELECT NEW ST_Point('Point ZM (1 1 1 1)' ).ST_CoordDim()
```

ST_CoveredBy-Methode

Testet, ob ein Geometriewert räumlich von einem anderen Geometriewert abgedeckt wird.

Syntax

geometry-expression.**ST_CoveredBy**(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* den Wert *geo2* abdeckt, sonst 0.

Bemerkungen

Die ST_CoveredBy-Methode testet, ob *geometry-expression* durch *geo2* vollständig abgedeckt ist.

geometry-expression.ST_CoveredBy(*geo2*) ist gleichwertig mit *geo2*.ST_Covers(*geometry-expression*).

Dieses Prädikat ist mit einem geringen Unterschied ähnlich wie ST_Within. Das ST_Within-Prädikat erfordert, dass mindestens ein innerer Punkt von *geometry-expression* im Inneren von *geo2* liegt. Bei ST_CoveredBy() gibt die Methode 1 zurück, wenn kein Punkt von *geometry-expression* außerhalb von *geo2* liegt, unabhängig davon, ob innere Punkte der beiden Geometrien eine Schnittmenge aufweisen. ST_CoveredBy kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung verwendet werden.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Siehe auch

- [ST_Covers-Methode für den ST_Geometry-Datentyp](#)
- [ST_Within-Methode für den ST_Geometry-Datentyp](#)
- [ST_Intersects-Methode für den ST_Geometry-Datentyp](#)
- [ST_CoveredByFilter-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Mit dem folgenden Beispiel wird getestet, ob ein Punkt von einem Polygon abgedeckt ist. Der Punkt ist vollständig abgedeckt vom Polygon, daher gibt das Beispiel 1 zurück.

```
SELECT NEW ST_Point( 1, 1 )
       .ST_CoveredBy( NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' ) )
```

Mit dem folgenden Beispiel wird getestet, ob eine Linie von einem Polygon abgedeckt ist. Die Linie wird vom Polygon komplett abgedeckt, daher gibt das Beispiel 1 zurück. Wenn ST_Within anstelle von ST_CoveredBy verwendet wird, gibt ST_Within 0 zurück.

```
SELECT NEW ST_LineString( 'LineString( 0 0, 1 0 )' )
       .ST_CoveredBy( NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' ) )
```

Das folgende Beispiel enthält eine Liste der ShapeIDs, bei denen der angegebene Punkt in der Shape-Geometrie liegt. Die Anweisung in diesem Beispiel gibt das Ergebnis 3 , 5 , 6 zurück. Beachten Sie, dass ShapeID 6 aufgelistet ist, auch wenn der Punkt das Shape-Polygon dieser Zeile nur an der Polygonbegrenzung schneidet.

```
SELECT LIST( ShapeID ORDER BY ShapeID )  
FROM SpatialShapes  
WHERE NEW ST_Point( 1, 4 ).ST_CoveredBy( Shape ) = 1
```

ST_CoveredByFilter-Methode

Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie von einer anderen abgedeckt wird.

Syntax

geometry-expression.**ST_CoveredByFilter**(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* von *geo2* abgedeckt werden könnte, sonst 0.

Bemerkungen

Die ST_CoveredByFilter-Methode bietet einen effizienten Test, um zu ermitteln, ob eine Geometrie von einer anderen abgedeckt werden könnte.

Dieser Test ist weniger kostenträchtig als ST_CoveredBy, kann aber in manchen Fällen 1 zurückgeben, in denen *geometry-expression* nicht von *geo2* abgedeckt wird.

Aus diesem Grund kann diese Methode als Primärfilter nützlich sein, wenn die weitere Verarbeitung erkennen lässt, ob Geometrien auf gewünschte Art und Weise interagieren.

Die Implementierung von ST_CoveredByFilter beruht auf Metadaten, die den gespeicherten Geometrien zugeordnet sind. Da sich die verfügbaren Metadaten je nachdem, wie Daten geladen werden oder wo ST_CoveredByFilter in einer Abfrage verwendet wird, von Serverversion zu Serverversion verändern können, kann der Ausdruck *geometry-expression*.ST_CoveredByFilter(*geo2*) unterschiedliche Ergebnisse zurückgeben, wenn *geometry-expression* nicht von *geo2* abgedeckt wird. Wenn *geometry-expression* von *geo2* abgedeckt wird, gibt ST_CoveredByFilter 1 zurück.

Siehe auch

- [ST_CoveredBy-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

ST_Covers-Methode

Testet, ob ein Geometriewert räumlich einen anderen Geometriewert abdeckt.

Syntax

geometry-expression.**ST_Covers**(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* den Wert *geo2* abdeckt, sonst 0.

Bemerkungen

Die ST_Covers-Methode testet, ob den *geometry-expression* den Wert *geo2* vollständig abdeckt. *geometry-expression*.ST_Covers(*geo2*) ist gleichwertig mit *geo2*.ST_CoveredBy(*geometry-expression*).

Dieses Prädikat ist mit einem geringen Unterschied ähnlich wie ST_Contains. Das ST_Contains-Prädikat erfordert, dass mindestens ein innerer Punkt von *geo2* im Inneren von *geometry-expression* liegt. Für ST_Covers() gibt die Methode 1 zurück, wenn kein Punkt von *geo2* außerhalb von *geometry-expression* liegt. ST_Covers kann auch mit Geometrien in einem räumlichen Bezugssystem mit gewölbter Erddarstellung verwendet werden, ST_Contains hingegen nicht.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Siehe auch

- [ST_CoveredBy-Methode für den ST_Geometry-Datentyp](#)
- [ST_Contains-Methode für den ST_Geometry-Datentyp](#)
- [ST_Intersects-Methode für den ST_Geometry-Datentyp](#)
- [ST_CoversFilter-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Mit dem folgenden Beispiel wird getestet, ob ein Polygon einen Punkt abdeckt. Das Polygon deckt den Punkt vollständig ab, daher wird in diesem Beispiel 1 zurückgegeben.

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' )
.ST_Covers( NEW ST_Point( 1, 1 ) )
```

Mit dem folgenden Beispiel wird getestet, ob ein Polygon eine Linie abdeckt. Das Polygon deckt die Linie vollständig ab, daher wird in diesem Beispiel 1 zurückgegeben. Wenn ST_Contains anstelle von ST_Covers verwendet wurde, gibt ST_Contains 0 zurück.

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' )
.ST_Covers( NEW ST_LineString( 'LineString( 0 0, 1 0 )' ) )
```

Das folgende Beispiel enthält eine Liste der ShapeIDs, in denen das angegebene Polygon jede Shape-Geometrie abdeckt. Die Anweisung in diesem Beispiel gibt das Ergebnis 1,16,17,19,26 zurück. Beachten Sie, dass ShapeID 1 aufgelistet ist, obwohl das Polygon den Shape-Punkt dieser Zeile nur an der Begrenzung des Polygons schneidet.

```
SELECT LIST( ShapeID ORDER BY ShapeID )
FROM SpatialShapes
WHERE NEW ST_Polygon( NEW ST_Point( 0, 0 ),
NEW ST_Point( 8, 2 ) ).ST_Covers( Shape ) = 1
```

ST_CoversFilter-Methode

Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie eine andere abdeckt.

Syntax

```
geometry-expression.ST_CoversFilter(geo2)
```

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* den Wert *geo2* abdecken könnte, sonst 0.

Bemerkungen

Die ST_CoversFilter-Methode bietet einen effizienten Test, um zu ermitteln, ob eine Geometrie eine andere abdecken kann. Gibt 1 zurück, wenn *geometry-expression* den Wert *geo2* abdecken könnte, sonst 0.

Dieser Test ist kostengünstiger als ST_Covers, kann aber in manchen Fällen 1 zurückgeben, in denen *geometry-expression* den Wert *geo2* nicht abdeckt.

Aus diesem Grund kann diese Methode als Primärfilter nützlich sein, wenn die weitere Verarbeitung erkennen lässt, ob Geometrien auf gewünschte Art und Weise interagieren.

Die Implementierung von ST_CoversFilter beruht auf Metadaten, die den gespeicherten Geometrien zugeordnet sind. Da sich die verfügbaren Metadaten je nachdem, wie Daten geladen werden oder wo ST_CoversFilter in einer Abfrage verwendet wird, von Serverversion zu Serverversion verändern können, kann der Ausdruck *geometry-expression.ST_CoversFilter (geo2)* unterschiedliche Ergebnisse zurückgeben, wenn *geometry-expression* nicht den Wert *geo2* abdeckt. Wenn *geometry-expression* den Wert *geo2* abdeckt, gibt ST_CoversFilter immer 1 zurück.

Siehe auch

- [ST_Covers-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

ST_Crosses-Methode

Testet, ob ein Geometriewert einen anderen Geometriewert kreuzt.

Syntax

geometry-expression.ST_Crosses(geo2)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* den Wert *geo2* kreuzt, sonst 0. Gibt NULL zurück, wenn *geometry-expression* eine Fläche oder Mehrfachoberfläche oder *geo2* ein Punkt oder ein Mehrfachpunkt ist.

Bemerkungen

Testet, ob ein Geometriewert einen anderen Geometriewert kreuzt.

Wenn sowohl *geometry-expression* als auch *geo2* Kurven oder Mehrfachkurven sind, kreuzen sie einander, wenn sich ihre Innenbereiche an mindestens einem Punkt kreuzen. Wenn die Schnittmenge eine Kurve oder Mehrfachkurve ergibt, kreuzen sich die Geometrien nicht. Wenn alle Schnittpunkte Begrenzungspunkte sind, kreuzen sich die Geometrien nicht.

Wenn *geometry-expression* eine niedrigere Dimension hat als *geo2*, kreuzt *geometry-expression* dann *geo2*, wenn ein Teil von *geometry-expression* im Inneren von *geo2* liegt und ein Teil von *geometry-expression* im Außenbereich von *geo2* liegt.

Genauer gesagt gibt *geometry-expression.ST_Crosses(geo2)* 1 zurück, wenn Folgendes TRUE ist:

```
( geometry-expression.ST_Dimension() = 1 AND geo2.ST_Dimension() = 1 AND
geometry-expression.ST_Relate( geo2, '0*****' ) = 1 ) OR( geometry-
expression.ST_Dimension() < geo2.ST_Dimension() AND geometry-
expression.ST_Relate( geo2, 'T*T*****' ) = 1 )
```

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

- [ST_Intersects-Methode](#) für den ST_Geometry-Datentyp
- [ST_Dimension-Methode](#) für den ST_Geometry-Datentyp
- [ST_Relate-Methode](#) für den ST_Geometry-Datentyp
- [ST_Overlaps-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.29

Beispiel

Das folgende Beispiel gibt den Wert 1 zurück.

```
SELECT NEW ST_LineString( 'LineString( 0 0, 2 2 )' )
.ST_Crosses( NEW ST_LineString( 'LineString( 0 2, 2 0 )' ) )
```

Das folgende Beispiel gibt die Ergebnismenge 0 zurück, da die Innenbereiche der beiden Linien keine Schnittmenge aufweisen (die einzige Schnittmenge befindet sich an der ersten Begrenzung der Linienfolge).

```
SELECT NEW ST_LineString( 'LineString( 0 1, 2 1 )' )
.ST_Crosses( NEW ST_LineString( 'LineString( 0 0, 2 0 )' ) )
```

Das folgende Beispiel gibt NULL zurück, da die erste Geometrie eine Fläche ist.

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 0 1, 1 0, 0 0))' )
.ST_Crosses( NEW ST_LineString( 'LineString( 0 0, 2 0 )' ) )
```

ST_Difference-Methode

Gibt den Geometriewert zurück, der die Punktmengendifferenz von zwei Geometrien darstellt.

Syntax

geometry-expression.ST_Difference(geo2)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der von <i>geometry-expression</i> subtrahiert werden soll.

Rückgabe

- **ST_Geometry** Gibt den Geometriewert zurück, der die Punktmengendifferenz von zwei Geometrien darstellt.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Die ST_Difference-Methode findet die räumliche Differenz zweier Geometrien. Ein Punkt ist in der Ergebnismenge enthalten, wenn er in *geometry-expression*, nicht aber in *geo2* vorhanden ist.

Im Unterschied zu anderen räumlichen Set-Operationen (ST_Union, ST_Intersection, ST_SymDifference) ist die ST_Difference-Methode nicht symmetrisch. Die Methode kann für `A.ST_Difference(B)` und `B.ST_Difference(A)` unterschiedliche Ergebnisse liefern.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Siehe auch

- [ST_Intersection-Methode für den ST_Geometry-Datentyp](#)
- [ST_SymDifference-Methode für den ST_Geometry-Datentyp](#)
- [ST_Union-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

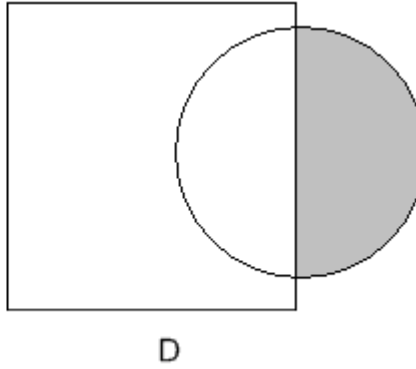
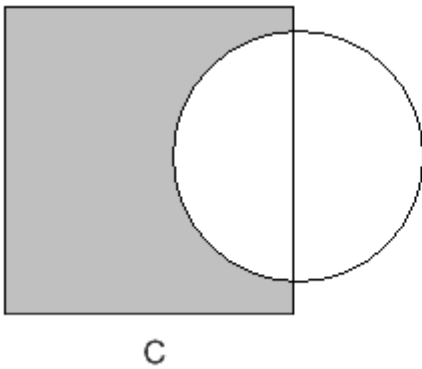
- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.20

Beispiel

Das folgende Beispiel zeigt die Differenz (C) eines Quadrats(A) mit einem entfernten Kreis (B) und die Differenz (D) eines Kreises (B) mit einem entfernten Quadrat (A).

```
SELECT NEW ST_Polygon( 'Polygon( (-1 -0.25, 1 -0.25, 1 2.25, -1 2.25, -1
-0.25) )' ) AS A
, NEW ST_CurvePolygon( 'CurvePolygon( CircularString( 0 1, 1 2, 2 1, 1
0, 0 1 ) )' ) AS B
, A.ST_Difference( B ) AS C
, B.ST_Difference( A ) AS D
```

Das folgende Bild zeigt die Differenz $C=A-B$ und $D=B-A$ als schattierten Teil des Bilds. Jede Differenz besteht aus einer einzelnen Fläche, die alle Punkte enthält, die sich in der Geometrie auf der linken Seite der Differenz und nicht in der Geometrie auf der rechten Seite befinden.



ST_Dimension-Methode

Gibt die Dimension des ST_Geometry-Werts zurück. Die Punkte haben die Dimension 0, Linien haben die Dimension 1 und Flächen haben die Dimension 2. Eine leere Geometrie hat die Dimension -1.

Syntax

geometry-expression.ST_Dimension()

Rückgabe

- **SMALLINT** Gibt die Dimension von *geometry-expression* als SMALLINT zwischen -1 und 2 zurück.

Bemerkungen

Die ST_Dimension-Methode gibt die räumliche Dimension zurück, die durch eine Geometrie belegt wird. Die folgenden Werte können zurückgegeben werden:

- **-1** Die Geometrie entspricht der leeren Menge.
- **0** Die Geometrie besteht nur aus einzelnen Punkten (zum Beispiel ST_Point oder ST_MultiPoint).
- **1** Die Geometrie enthält mindestens eine Kurve und keine Fläche (zum Beispiel ST_LineString oder ST_MultiCurve).
- **2** Die Geometrie besteht aus mindestens einer Fläche (zum Beispiel ST_Polygon oder ST_MultiPolygon).

Bei der Berechnung der Dimension einer Sammlung wird die größte Dimension jedes Elements zurückgegeben. Beispiel: Wenn eine Geometriesammlung eine Kurve und einen Punkt enthält, gibt ST_Dimension für die Sammlung 1 zurück.

Siehe „Funktionsweise räumlicher Dimensionen“ auf Seite 56.

Hinweis

Standardmäßig verwendet ST_Dimension das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_CoordDim-Methode](#) für den ST_Geometry-Datentyp
- [ST_Relate-Methode](#) für den ST_Geometry-Datentyp
- [ST_Relate-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.2

Beispiel

Das folgende Beispiel gibt den Wert 0 zurück.

```
SELECT NEW ST_Point(1.0,1.0).ST_Dimension()
```

Das folgende Beispiel gibt den Wert 1 zurück.

```
SELECT NEW ST_LineString('LineString( 0 0, 1 1)' ).ST_Dimension()
```

ST_Disjoint-Methode

Testet, ob ein Geometriewert von einem anderen Wert räumlich unabhängig ist.

Syntax

geometry-expression.**ST_Disjoint**(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* von *geo2* räumlich unabhängig ist, sonst 0.

Bemerkungen

Testet, ob ein Geometriewert von einem anderen Wert räumlich unabhängig ist. Zwei Geometrien sind voneinander unabhängig, wenn ihre Schnittmenge leer ist. Anders gesagt: Sie sind unabhängig, wenn im *geometry-expression* nirgendwo ein Punkt liegt, der auch in *geo2* liegt.

geometry-expression.ST_Disjoint(*geo2*) = 1 ist gleichwertig mit *geometry-expression*.ST_Intersects(*geo2*) = 0.

Hinweis
Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Siehe auch

- [ST_Intersects-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.26

Beispiel

Im folgenden Beispiel wird eine Ergebnismenge mit einer Zeile für jede Form zurückgegeben, die keinen gemeinsamen Punkt mit dem angegebenen Dreieck hat.

```
SELECT ShapeID, "Description"
FROM SpatialShapes
WHERE NEW ST_Polygon( 'Polygon((0 0, 5 0, 0 5, 0 0))' ).ST_Disjoint( Shape )
= 1
ORDER BY ShapeID
```

Die Abfrage erzeugt die folgende Ergebnismenge:

ShapeID	Description
1	Point
22	Triangle

ST_Distance-Methode

Gibt die kleinste Entfernung zwischen *geometry-expression* und dem angegebenen Geometriewert zurück.

Syntax

```
geometry-expression.ST_Distance(geo2[, unit-name])
```

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, dessen Entfernung von <i>geometry-expression</i> gemessen werden soll.

Name	Typ	Beschreibung
unit-name	VAR-CHAR(128)	Die Einheiten, in denen die Entfernung gemessen werden soll. Ist standardmäßig die Einheit des räumlichen Bezugssystems. Der unit-name-Parameter muss mit der UNIT_NAME-Spalte einer Zeile in der ST_UNITS_OF_MEASURE-Ansicht übereinstimmen, wo UNIT_TYPE auf 'LINEAR' eingestellt ist.

Rückgabe

- **DOUBLE** Gibt die kleinste Entfernung zwischen *geometry-expression* und *geo2* in den angegebenen linearen Maßeinheiten zurück. Wenn *geometry-expression* oder *geo2* leer ist, wird NULL zurückgegeben.

Bemerkungen

Die ST_Distance-Methode berechnet die kürzeste Entfernung zwischen zwei Geometrien. Für räumliche Bezugssysteme mit planer Erddarstellung wird die Entfernung als kartesische Entfernung in der Ebene gemessen und in den linearen Maßeinheiten des damit verbundenen räumlichen Bezugssystems berechnet. Für räumliche Bezugssysteme mit gewölbter Erddarstellung wird die Entfernung berechnet, indem die Krümmung der Erdoberfläche mithilfe von Ellipsoidparametern in der Definition des räumlichen Bezugssystems berücksichtigt wird.

Hinweis

Für räumliche Bezugssysteme mit gewölbter Erddarstellung wird die ST_Distance-Methode nur unterstützt, wenn *geometry-expression* und *geo2* nur Punkte enthalten.

Hinweis

Standardmäßig verwendet ST_Distance das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_Area-Methode für den ST_Surface-Datentyp](#)
- [ST_Length-Methode für den ST_Curve-Datentyp](#)
- [ST_Perimeter-Methode für den ST_Surface-Datentyp](#)
- [ST_WithinDistance-Methode für den ST_Geometry-Datentyp](#)
- [ST_WithinDistanceFilter-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.23

Beispiel

Das folgende Beispiel gibt eine geordnete Ergebnismenge mit einer Zeile für jede Form und die entsprechende Entfernung vom Point (2,3) zurück.

```
SELECT ShapeID, ROUND( Shape.ST_Distance( NEW ST_Point( 2, 3 ) ), 2 ) AS dist
FROM SpatialShapes
WHERE ShapeID < 17
ORDER BY dist
```

Die Abfrage erzeugt die folgende Ergebnismenge:

ShapeID	DIST
2	0.0
3	0.0
5	1.0
6	1.21
16	1.41
1	5.1

Im folgenden Beispiel werden Punkte erstellt, die Halifax, NS und Waterloo, ON, in Kanada darstellen. ST_Distance wird verwendet, um die Entfernung zwischen den beiden Punkten in Meilen zu ermitteln. Als Ergebnis wird 846 zurückgegeben. In diesem Beispiel wird vorausgesetzt, dass die st_geometry_predefined_uom-Funktion von der sa_install_feature-Systemprozedur installiert wurde. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

```
SELECT ROUND( NEW ST_Point( -63.573566, 44.646244, 4326 )
               .ST_Distance( NEW ST_Point( -80.522372, 43.465187, 4326 )
                             , 'Statute mile' ), 0 )
```

ST_Envelope-Methode

Gibt das begrenzende Rechteck für den Geometriewert zurück.

Syntax

geometry-expression.ST_Envelope()

Rückgabe

- **ST_Polygon** Gibt ein Polygon zurück, das das begrenzende Rechteck für *geometry-expression* darstellt.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Die ST_Envelope-Methode erstellt ein Polygon, das ein achsenangepasstes begrenzendes Rechteck für *geometry-expression* darstellt. Der Rahmen deckt die komplette Geometrie ab und kann als einfacher Näherungswert für die Geometrie verwendet werden.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

- [ST_EnvelopeAggr-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.15

Beispiel

Das folgende Beispiel gibt den Wert Polygon ((0 0, 1 0, 1 4, 0 4, 0 0)) zurück.

```
SELECT Shape.ST_Envelope()  
FROM SpatialShapes WHERE ShapeID = 6
```

ST_EnvelopeAggr-Methode

Gibt das begrenzende Rechteck für alle Geometrien in einer Gruppe zurück

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Syntax

ST_Geometry::ST_EnvelopeAggr(*geometry-column*)

Parameter

Name	Typ	Beschreibung
geometry-column	ST_Geometry	Die Geometriewerte zum Generieren des begrenzenden Rechtecks. Das ist üblicherweise eine Spalte.

Rückgabe

- **ST_Polygon** Gibt ein Polygon zurück, das das begrenzende Rechteck für alle Geometrien in einer Gruppe darstellt.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist derselbe wie der für den ersten Parameter.

Siehe auch

- [ST_Envelope-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert Polygon ((-3 -1, 8 -1, 8 8, -3 8, -3 -1)) zurück.

```
SELECT ST_Geometry::ST_EnvelopeAggr( Shape ) FROM SpatialShapes
```

ST_Equals-Methode

Testet, ob ein ST_Geometry-Wert räumlich gleich einem anderen ST_Geometry-Wert ist.

Syntax

```
geometry-expression.ST_Equals(geo2)
```

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn die beiden Geometriewerte räumlich gleich sind, sonst 0.

Bemerkungen

Testet, ob ein ST_Geometry-Wert gleich einem anderen ST_Geometry-Wert ist.

Der Test auf räumliche Gleichheit erfolgt durch den Vergleich der begrenzenden Rechtecke der beiden Geometrien. Wenn sie innerhalb der Toleranz nicht gleich sind, werden die beiden Geometrien als ungleich angesehen und 0 wird zurückgegeben. Andernfalls gibt der Datenbankserver 1 zurück, wenn *geometry-expression*.ST_SymDifference(*geo2*) eine leere Menge ist, sonst 0.

Der SQL/MM-Standard definiert ST_Equals nur für ST_SymDifference, ohne zusätzlichen Vergleich der begrenzenden Rechtecke. Es gibt einige Geometrien, die mit ST_SymDifference eine leere Ergebnismenge zurückgeben, obwohl ihre begrenzenden Rechtecke nicht gleich sind. Diese Geometrien würden vom SQL/MM-Standard als gleich angesehen, nicht aber von SQL Anywhere. Dieser Unterschied kann auftreten, wenn eine oder beide Geometrien Spitzen oder Punktierungen enthalten.

Zwei Geometriewerte können als gleichwertig angesehen werden, selbst wenn sie unterschiedliche Darstellungen haben. Beispiel: Zwei Linienfolgen können unterschiedliche Ausrichtungen haben, aber dieselbe Gruppe von Punkten im Raum enthalten. Diese beiden Linienfolgen werden von ST_Equals als

gleich angesehen, nicht aber von ST_OrderingEquals. Siehe „[Funktionsweise räumlicher Vergleiche](#)“ auf Seite 51.

ST_Equals kann durch die Auflösung des räumlichen Bezugssystem oder die Präzision der Daten begrenzt sein.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Siehe auch

- [ST_OrderingEquals-Methode](#) für den ST_Geometry-Datentyp
- [ST_EqualsFilter-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.24

Beispiel

Das folgende Beispiel gibt den Wert 16 zurück. Für die ShapeID entsprechende Form enthält das Ergebnis 16 die gleichen Punkte, allerdings in anderer Reihenfolge als das angegebene Polygon.

```
SELECT ShapeID FROM SpatialShapes
WHERE Shape.ST_Equals( NEW ST_Polygon( 'Polygon ((2 0, 1 2, 0 0, 2 0))' ) )
= 1
```

Im folgenden Beispiel wird das Ergebnis 1 zurückgegeben. Es zeigt, dass die zwei Linienfolgen gleich sind, obwohl sie eine unterschiedliche Anzahl von Stellen in einer anderen Reihenfolge enthalten und der Zwischenpunkt nicht genau auf der Linie liegt. Der Zwischenpunkt ist rund 3.33e-7 entfernt von der Linie mit nur zwei Punkten, aber diese Entfernung ist geringer als die Toleranz 1e-6 für das räumliche Standard-Bezugssystem (SRID 0).

```
SELECT NEW ST_LineString( 'LineString( 0 0, 0.333333 1, 1 3 )' )
.ST_Equals( NEW ST_LineString( 'LineString( 1 3, 0 0 )' ) )
```

ST_EqualsFilter-Methode

Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie gleich einer anderen ist.

Syntax

geometry-expression.**ST_EqualsFilter**(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn das begrenzende Rechteck für *geometry-expression* innerhalb der Toleranzgrenzen gleich dem begrenzenden Rechteck für *geo2* ist, sonst 0.

Bemerkungen

Die ST_EqualsFilter-Methode bietet einen effizienten Test, um zu ermitteln, ob eine Geometrie gleich einer anderen sein könnte. ST_EqualsFilter gibt 1 zurück, wenn *geometry-expression* gleich *geo2* sein könnte, sonst 0.

Dieser Test ist weniger kostenträchtig als ST_Equals, kann aber in manchen Fällen 1 zurückgeben, in denen *geometry-expression* nicht gleich *geo2* ist.

Aus diesem Grund kann diese Methode als Primärfilter nützlich sein, wenn die weitere Verarbeitung erkennen lässt, ob Geometrien auf gewünschte Art und Weise interagieren.

Die Implementierung von ST_EqualsFilter beruht auf Metadaten, die den gespeicherten Geometrien zugeordnet sind. Da sich die verfügbaren Metadaten je nachdem, wie Daten geladen werden oder wo ST_EqualsFilter in einer Abfrage verwendet wird, von Serverversion zu Serverversion verändern können, kann der Ausdruck *geometry-expression*.ST_EqualsFilter (*geo2*) unterschiedliche Ergebnisse zurückgeben, wenn *geometry-expression* nicht gleich *geo2* ist. Wenn *geometry-expression* gleich *geo2* ist, gibt ST_EqualsFilter immer 1 zurück.

Siehe auch

- [ST_Equals-Methode für den ST_Geometry-Datentyp](#)
- [ST_OrderingEquals-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

ST_GeomFromBinary-Methode

Konstruiert eine Geometrie aus einer binären Zeichenfolgendarstellung.

Syntax

ST_Geometry::ST_GeomFromBinary(*binary-string*[, *srid*])

Parameter

Name	Typ	Beschreibung
binary-string	LONG BINARY	Eine Zeichenfolge, die die binäre Darstellung einer Geometrie enthält. Die Eingabedaten können in jedem unterstützten Binärformat sein, einschließlich WKB oder EWKB.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird und die Eingabezeichenfolge keine SRID übergibt, ist der Standardwert 0.

Rückgabe

- **ST_Geometry** Gibt einen Geometriewert des geeigneten Typs basierend auf der Quellzeichenfolge zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Führt die syntaktische Analyse einer Zeichenfolge mit einem der unterstützten Formate durch und erstellt einen Geometriewert des entsprechenden Datentyps. Diese Methode wird vom Server benutzt, wenn eine Konvertierung von einer Binärzeichenfolge in einen geometrischen Typ ausgewertet wird.

Einige Eingabeformate enthalten eine SRID-Definition. Wenn er übergeben wird, muss der *srid*-Parameter zu jedem Wert passen, der der Eingabezeichenfolge entnommen wird.

Siehe auch

- [ST_GeomFromWKB-Methode für den ST_Geometry-Datentyp](#)
- [ST_GeomFromText-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert `Point (10 20)` zurück.

```
SELECT
ST_Geometry::ST_GeomFromBinary( 0x010100000000000000000000024400000000000003440
)
```

ST_GeomFromShape-Methode

Führt die syntaktische Analyse einer Zeichenfolge mit einem ESRI-Formdatensatz durch und erstellt einen Geometriewert des entsprechenden Datentyps.

Syntax

ST_Geometry::ST_GeomFromShape(*shape*[, *srid*])

Parameter

Name	Typ	Beschreibung
shape	LONG BINARY	Eine Zeichenfolge, die eine Geometrie im ESRI-Formformat enthält.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_Geometry** Gibt einen Geometriewert des geeigneten Typs basierend auf der Quellzeichenfolge zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Führt die syntaktische Analyse einer einzelnen ESRI-Form durch und erstellt einen Geometriewert des entsprechenden Datentyps. Der Datensatz ist ein einzelner Datensatz aus der *.shp*-Datei einer ESRI-Formdatei oder kann ein einzelner Zeichenfolgewert aus einer Geodatenbank sein.

Die Formdarstellung wird oft verwendet, um räumliche Daten darzustellen. Eine vollständige Beschreibung der Formdefinition finden Sie auf der ESRI-Website im Dokument <http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf>.

In den meisten Fällen ist es einfacher, eine ESRI-Formdatei mit dem SHAPEFILE-Format mithilfe der FORMAT-Klausel der LOAD TABLE-Anweisung oder einem OPENSTRING-Ausdruck in einer FROM-Klausel zu laden anstatt die ST_GeomFromShape-Methode zu verwenden. Siehe „LOAD TABLE-Anweisung“ [*SQL Anywhere Server - SQL-Referenzhandbuch*] und „FROM-Klausel“ [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

ST_GeomFromText-Methode

Konstruiert eine Geometrie aus einer Zeichenfolgendarstellung.

Syntax

ST_Geometry::ST_GeomFromText(*character-string*[, *srid*])

Parameter

Name	Typ	Beschreibung
character-string	LONG VAR- CHAR	Eine Zeichenfolge mit der Textdarstellung einer Geometrie. Die Eingabe kann in jedem unterstützten Texteingabeformat erfolgen, einschließlich Well-Known-Text (WKT) oder Extended-Well-Known-Text (EWKT).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird und die Eingabezeichenfolge keine SRID enthält, ist der Standardwert 0.

Rückgabe

- **ST_Geometry** Gibt einen Geometriewert des geeigneten Typs basierend auf der Quellzeichenfolge zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Führt die syntaktische Analyse einer Textzeichenfolge durch, die eine Geometrie darstellt, und erstellt einen Geometriewert des entsprechenden Datentyps. Diese Methode wird vom Server benutzt, wenn eine Konvertierung von einer Textzeichenfolge in einen geometrischen Typ ausgewertet wird.

Der Server erkennt das Format der Eingabezeichenfolge. Einige Eingabeformate enthalten eine SRID-Definition. Wenn er übergeben wird, muss der *srid*-Parameter zu jedem Wert passen, der der Eingabezeichenfolge entnommen wird.

Siehe auch

- [ST_GeomFromBinary-Methode für den ST_Geometry-Datentyp](#)
- [ST_GeomFromWKT-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.40

Beispiel

Das folgende Beispiel gibt den Wert `LineString (1 2, 5 7)` zurück.

```
SELECT ST_Geometry::ST_GeomFromText( 'LineString( 1 2, 5 7 )', 4326 )
```

ST_GeomFromWKB-Methode

Führt die syntaktische Analyse einer Zeichenfolge durch, die eine WKB- oder EWKB-Darstellung einer Geometrie enthält und erstellt einen Geometriewert des entsprechenden Typs.

Syntax

```
ST_Geometry::ST_GeomFromWKB(wkb [, srid])
```

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Eine Zeichenfolge, die die WKB- oder EWKB-Darstellung eines Geometriewerts enthält.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_Geometry** Gibt einen Geometriewert des geeigneten Typs basierend auf der Quellzeichenfolge zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Führt die syntaktische Analyse einer Zeichenfolge durch, die die WKB- oder EWKB-Darstellung einer Geometrie enthält und einen Geometriewert des entsprechenden Typs erstellt.

Siehe auch

- [ST_GeomFromBinary-Methode](#) für den ST_Geometry-Datentyp
- [ST_GeomFromWKT-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.41

ST_GeomFromWKT-Methode

Führt die syntaktische Analyse einer Zeichenfolge durch, die die WKT- oder EWKT-Darstellung eines Geometriewerts enthält und erstellt einen Geometriewert des entsprechenden Typs.

Syntax

ST_Geometry::ST_GeomFromWKT(*wkt*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkt	LONG VAR-CHAR	Eine Zeichenfolge mit der WKT- oder EWKT-Darstellung eines Geometriewerts.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_Geometry** Gibt einen Geometriewert des geeigneten Typs basierend auf der Quellzeichenfolge zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Führt die syntaktische Analyse einer Zeichenfolge durch, die die WKT- oder EWKT-Darstellung eines Geometriewerts enthält und einen Geometriewert des entsprechenden Typs erstellt.

Siehe auch

- [ST_GeomFromText-Methode](#) für den ST_Geometry-Datentyp
- [ST_GeomFromWKB-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

ST_GeometryType-Methode

Gibt den Namen des Typs des ST_Geometry-Werts zurück.

Syntax

geometry-expression.**ST_GeometryType**()

Rückgabe

- **VARCHAR(128)** Gibt den Datentyp des Geometriewerts als Textzeichenfolge zurück. Mit dieser Methode können Sie den dynamischen Typ eines Werts ermitteln.

Bemerkungen

Die ST_GeometryType-Methode gibt eine Zeichenfolge zurück, die den spezifischen Typnamen von *geometry-expression* enthält.

Die Syntax "value IS OF(type)" kann auch verwendet werden, um den spezifischen Typ eines Werts festzulegen.

Hinweis

Standardmäßig verwendet ST_GeometryType das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.4

Beispiel

Der folgende Code gibt das Ergebnis 2, 3, 6, 16, 22, 24, 25 zurück, das die Liste der ShapeIDs enthält, deren entsprechende Form einer der angegebenen Typen ist.

```
SELECT LIST( ShapeID ORDER BY ShapeID )
FROM SpatialShapes
WHERE Shape.ST_GeometryType() IN( 'ST_Polygon', 'ST_CurvePolygon' )
```

ST_GeometryTypeFromBaseType-Methode

Führt die syntaktische Analyse einer Zeichenfolge durch, die die Typzeichenfolge definiert.

Syntax

ST_Geometry::ST_GeometryTypeFromBaseType(*base-type-str*)

Parameter

Name	Typ	Beschreibung
base-type-str	VARCHAR(128)	Eine Zeichenfolge, die die Basistypzeichenfolge enthält

Rückgabe

- **VARCHAR(128)** Gibt den Geometrietyp aus einer Basistypzeichenfolge zurück (die eine SRID-Definition enthalten kann). Wenn die Typzeichenfolge keine gültige Geometrietypzeichenfolge ist, wird ein Fehler zurückgegeben.

Bemerkungen

Die *ST_Geometry::ST_GeometryTypeFromBaseType*-Methode können Sie verwenden, um eine syntaktische Analyse des Geometrietypnamens aus einer Typenzeichenfolgendefinition durchzuführen.

Siehe auch

- [ST_SRIDFromBaseType-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert ST_Geometry zurück.

```
SELECT ST_Geometry::ST_GeometryTypeFromBaseType('ST_Geometry')
```

Das folgende Beispiel gibt den Wert ST_Point zurück.

```
SELECT ST_Geometry::ST_GeometryTypeFromBaseType('ST_Point(SRID=4326)')
```

Im folgenden Beispiel wird der Geometrietyp (ST_Point) gefunden, der von einem Parameter einer gespeicherten Prozedur akzeptiert wird.

```
CREATE PROCEDURE myprocedure( parm1 ST_Point(SRID=0) )
BEGIN
    -- ...
END;

SELECT    parm_name nm, base_type_str,
ST_Geometry::ST_GeometryTypeFromBaseType(base_type_str) geom_type
FROM      sysprocedure KEY JOIN sysprocparm
WHERE     proc_name='myprocedure' and parm_name='parm1'
```

ST_Intersection-Methode

Gibt den Geometriewert zurück, der die Punktmengenschnittmenge von zwei Geometrien darstellt.

Syntax

geometry-expression.**ST_Intersection**(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> geschnitten werden soll.

Rückgabe

- **ST_Geometry** Gibt den Geometriewert zurück, der die Punktmengenschnittmenge von zwei Geometrien darstellt.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Die ST_Intersection-Methode findet die räumliche Schnittmenge von zwei Geometrien. Ein Punkt befindet sich in der Schnittmenge, wenn er in beiden eingegebenen Geometrien vorhanden ist. Wenn die beiden Geometrien keine gemeinsamen Punkt mehr haben, ist das Ergebnis eine leere Geometrie.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Siehe auch

- [ST_Difference-Methode für den ST_Geometry-Datentyp](#)
- [ST_IntersectionAggr-Methode für den ST_Geometry-Datentyp](#)
- [ST_SymDifference-Methode für den ST_Geometry-Datentyp](#)
- [ST_Union-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

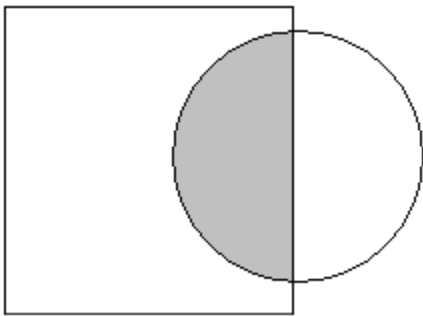
- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.18

Beispiel

Das folgende Beispiel zeigt die Schnittmenge (C) eines Quadrats (A) und eines Kreises (B).

```
SELECT NEW ST_Polygon( 'Polygon( (-1 -0.25, 1 -0.25, 1 2.25, -1 2.25, -1
-0.25) )' ) AS A
      , NEW ST_CurvePolygon( 'CurvePolygon( CircularString( 0 1, 1 2, 2 1, 1
0, 0 1 ) )' ) AS B
      , A.ST_Intersection( B ) AS C
```

Die Schnittmenge wird im folgenden Bild schattiert dargestellt. Es handelt sich um eine einzelne Fläche, die alle Punkte enthält, die sich im Quadrat und auch im Kreis befinden.



ST_IntersectionAggr-Methode

Gibt die räumliche Schnittmenge aller Geometrien in einer Gruppe zurück

Syntax

`ST_Geometry::ST_IntersectionAggr(geometry-column)`

Parameter

Name	Typ	Beschreibung
geometry-column	ST_Geometry	Die Geometriewerte zur Generierung der räumlichen Schnittmenge. Das ist üblicherweise eine Spalte.

Rückgabe

- **ST_Geometry** Gibt eine Geometrie zurück, die die räumliche Schnittmenge für alle Geometrien in einer Gruppe ist.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist derselbe wie der für den ersten Parameter.

Bemerkungen

Zeilen, in denen das Argument NULL ist, werden nicht eingeschlossen.

Gibt für eine leere Gruppe oder eine Gruppe ohne Nicht-NULL-Werte NULL zurück.

Wenn die Gruppe nur eine Nicht-NULL-Geometrie enthält, wird sie zurückgegeben. Sonst wird die Schnittmenge logisch berechnet, indem wiederholt die ST_Intersection-Methode angewendet wird, um die beiden Geometrien gleichzeitig zu kombinieren. Siehe [ST_Intersection-Methode für den ST_Geometry-Datentyp auf Seite 190](#).

Siehe auch

- [ST_Intersection-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert Polygon ((0 0, 1 2, .5 2, .75 3, .555555 3, 0 1.75, .5 1.75, 0 0)) zurück.

```
SELECT ST_Geometry::ST_IntersectionAggr( Shape )
FROM SpatialShapes WHERE ShapeID IN ( 2, 6 )
```

ST_Intersects-Methode

Testet, ob ein Geometriewert räumlich eine Schnittmenge mit einem anderen Wert hat.

Syntax

geometry-expression.**ST_Intersects**(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* eine räumliche Schnittmenge mit *geo2* hat, sonst 0.

Bemerkungen

Testet, ob ein Geometriewert räumlich eine Schnittmenge mit einem anderen Wert hat. Zwei Geometrien haben eine Schnittmenge, wenn Sie mindestens einen Punkt gemeinsam haben.

geometry-expression.ST_Intersects(*geo2*) = 1 ist gleichwertig mit *geometry-expression*.ST_Disjoint(*geo2*) = 0.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Siehe auch

- [ST_IntersectsRect-Methode](#) für den ST_Geometry-Datentyp
- [ST_Disjoint-Methode](#) für den ST_Geometry-Datentyp
- [ST_IntersectsFilter-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.27

Beispiel

Das folgende Beispiel gibt eine Ergebnismenge mit einer Zeile für jede Form zurück, die eine Schnittmenge mit der angegebenen Linie hat.

```
SELECT ShapeID, "Description"
FROM SpatialShapes
WHERE NEW ST_LineString( 'LineString( 2 2, 4 4 )' ).ST_Intersects( Shape ) =
1
ORDER BY ShapeID
```

Die Abfrage erzeugt die folgende Ergebnismenge:

ShapeID	Description
2	Square
3	Rectangle
5	L shape line
18	CircularString
22	Triangle

Um anzuzeigen, wie die Geometrien in der SpatialShapes-Tabelle Schnittmengen mit der Linie im oben gezeigten Beispiel haben, führen Sie die folgende Abfrage im Interactive SQL Spatial Viewer aus.

```
SELECT Shape
FROM SpatialShapes
WHERE NEW ST_LineString( 'LineString( 2 2, 4 4 )' ).ST_Intersects( Shape ) =
1
UNION ALL SELECT NEW ST_LineString( 'LineString( 2 2, 4 4 )' )
```

ST_IntersectsFilter-Methode

Ein kostengünstiger Test, um zu prüfen, ob sich die beiden Geometrien schneiden.

Syntax

```
geometry-expression.ST_IntersectsFilter(geo2)
```

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* eine Schnittmenge mit *geo2* haben könnte, sonst 0.

Bemerkungen

Die ST_IntersectsFilter-Methode bietet einen effizienten Test, um zu ermitteln, ob zwei Geometrien eine Schnittmenge haben. Gibt 1 zurück, wenn *geometry-expression* eine Schnittmenge mit *geo2* haben könnte, sonst 0.

Dieser Test ist weniger kostenträchtig als ST_Intersects, kann in manchen Fällen aber 1 zurückgeben, obwohl die Geometrien eigentlich keine Schnittmengen haben. Aus diesem Grund kann diese Methode als Primärfilter nützlich sein, wenn die weitere Verarbeitung erkennen lässt, ob Geometrien in der Tat eine Schnittmenge aufweisen.

Die Implementierung von ST_IntersectsFilter beruht auf Metadaten, die den gespeicherten Geometrien zugeordnet sind. Da sich die verfügbaren Metadaten je nachdem, wie Daten geladen werden oder wo ST_IntersectsFilter in einer Abfrage verwendet wird, von Serverversion zu Serverversion verändern können, kann der Ausdruck *geometry-expression.ST_IntersectsFilter (geo2)* unterschiedliche Ergebnisse zurückgeben, wenn *geometry-expression* keine Schnittmenge mit *geo2* hat. Wenn *geometry-expression* eine Schnittmenge mit *geo2* hat, gibt ST_IntersectsFilter immer 1 zurück.

Siehe auch

- [ST_Intersects-Methode für den ST_Geometry-Datentyp](#)
- [ST_IntersectsRect-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

ST_IntersectsRect-Methode

Testet, ob eine Geometrie eine Schnittmenge mit einem Rechteck hat.

Syntax

geometry-expression.ST_IntersectsRect(pmin,pmax)

Parameter

Name	Typ	Beschreibung
pmin	ST_Point	Der kleinste Punktwert, der mit <i>geometry-expression</i> verglichen werden soll.
pmax	ST_Point	Der größte Punktwert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt den Wert 1 zurück, wenn *geometry-expression* eine Schnittmenge mit dem angegebenen Rechteck hat, sonst 0.

Bemerkungen

Die ST_IntersectsRect-Methode testet, ob eine Geometrie eine Schnittmenge mit einem bestimmten achsenangepassten begrenzenden Rechteck aufweist.

Die Methode ist mit Folgendem gleichwertig: *geometry-expression.ST_Intersects(NEW ST_Polygon(pmin, pmax))*

Aus diesem Grund kann diese Methode zum Schreiben von Fensterabfragen nützlich sein, um alle Geometrien zu finden, die eine Schnittmenge mit einem gegebenen achsenangepassten Rechteck aufweisen.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Siehe auch

- [ST_Intersects-Methode für den ST_Geometry-Datentyp](#)
- [ST_IntersectsFilter-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel enthält eine Liste der ShapeIDs, in denen das vom Rahmen der beiden Punkte angegebene Rechteck eine Schnittmenge mit der entsprechenden Shape-Geometrie aufweist. Die Anweisung in diesem Beispiel gibt das folgende Ergebnis 3 , 5 , 6 , 18 zurück.

```
SELECT LIST( ShapeID ORDER BY ShapeID )
FROM SpatialShapes
WHERE Shape.ST_IntersectsRect( NEW ST_Point( 0, 4 ), NEW ST_Point( 2, 5 ) )
= 1
```

Mit dem folgenden Beispiel wird getestet, ob eine Linienfolge eine Schnittmenge mit einem Rechteck aufweist. Die bereitgestellte Linienfolge hat keine Schnittmenge mit dem Rechteck, das durch die beiden Punkte definiert wird (auch wenn der Rahmen der Linienfolge eine Schnittmenge mit dem Rahmen der beiden Punkte hat).

```
SELECT NEW ST_LineString( 'LineString( 0 0, 10 0, 10 10 )' )
.ST_IntersectsRect( NEW ST_Point( 4, 4 ) , NEW ST_Point( 6, 6 ) )
```

ST_Is3D-Methode

Ermittelt, ob der Geometriewert Z-Koordinatenwerte hat.

Hinweis

Standardmäßig verwendet ST_Is3D das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

geometry-expression.**ST_Is3D**()

Rückgabe

- **BIT** Gibt 1 zurück, wenn der Geometriewert Z-Koordinatenwerte hat, sonst 0.

Siehe auch

- [ST_CoordDim-Methode](#) für den ST_Geometry-Datentyp
- [ST_IsMeasured-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.10

Beispiel

Das folgende Beispiel gibt den Wert 1 zurück.

```
SELECT ShapeID FROM SpatialShapes WHERE Shape.ST_Is3D() = 1
```

ST_IsEmpty-Methode

Ermittelt, ob der Geometriewert für eine leere Menge steht.

Syntax

geometry-expression.**ST_IsEmpty()**

Rückgabe

- **BIT** Gibt 1 zurück, wenn der Geometriewert leer ist, sonst 0.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.7

Beispiel

Das folgende Beispiel gibt den Wert 1 zurück.

```
SELECT NEW ST_LineString().ST_IsEmpty()
```

ST_IsMeasured-Methode

Ermittelt, ob der Geometriewert zugeordnete Messwerte hat.

Hinweis

Standardmäßig verwendet ST_IsMeasured das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

geometry-expression.**ST_IsMeasured()**

Rückgabe

- **BIT** Gibt 1 zurück, wenn der Geometriewert Messwerte hat, sonst 0.

Siehe auch

- [ST_CoordDim-Methode](#) für den [ST_Geometry-Datentyp](#)
- [ST_Is3D-Methode](#) für den [ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.11

Beispiel

Das folgende Beispiel gibt den Wert 1 zurück.

```
SELECT ST_Geometry::ST_GeomFromText( 'LineString M( 1 2 4, 5 7  
3 )' ).ST_IsMeasured()
```

Das folgende Beispiel gibt den Wert 0 zurück.

```
SELECT count(*) FROM SpatialShapes WHERE Shape.ST_IsMeasured() = 1
```

ST_IsSimple-Methode

Legt fest, ob der Geometriewert einfach (keine Eigenschnittmengen oder andere Unregelmäßigkeiten) ist.

Syntax

geometry-expression.**ST_IsSimple()**

Rückgabe

- **BIT** Gibt 1 zurück, wenn der Geometriewert einfach ist, sonst 0.

Siehe auch

- [ST_IsValid-Methode](#) für den [ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.8

Beispiel

Der folgende Code gibt das Ergebnis 29 zurück, weil die entsprechende Mehrfachlinienfolge zwei Linien enthält, die einander kreuzen.

```
SELECT ShapeID FROM SpatialShapes WHERE Shape.ST_IsSimple() = 0
```

ST_IsValid-Methode

Ermittelt, ob die Geometrie ein gültiges räumliches Objekt ist.

Syntax

geometry-expression.**ST_IsValid()**

Rückgabe

- **BIT** Gibt 1 zurück, wenn der Geometriewert gültig ist, sonst 0.

Bemerkungen

Standardmäßig validiert der Server räumliche Daten nicht, wenn sie erstellt oder aus anderen Formaten importiert werden. Die ST_IsValid-Methode kann verwendet werden, um zu überprüfen, ob die importierten Daten eine gültige Geometrie darstellen. Vorgänge mit ungültigen Geometrien geben undefinierte Ergebnisse zurück.

Siehe auch

- [ST_IsSimple-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.9

Beispiel

Der folgende Code gibt die Ergebnismenge 0 zurück, weil das Polygon eine liegende Acht enthält (der Ring hat eine Schnittstelle mit sich selbst).

```
SELECT ST_Geometry::ST_GeomFromText( 'Polygon(( 0 0, 4 0, 4 5, 0 -1, 0
0 ))' )
.ST_IsValid()
```

Der folgende Code gibt die Ergebnismenge 0 zurück, weil die Polygone innerhalb der Geometrie eine Schnittmenge mit sich selbst aufweisen. Eigenschnittmengen einer Geometriegruppe mit endlicher Anzahl von Punkten werden als gültig angesehen.

```
SELECT ST_Geometry::ST_GeomFromText(
'MultiPolygon((( 0 0, 2 0, 1 2, 0 0 )),((0 2, 1 0, 2 2, 0 2)))' )
.ST_IsValid()
```

ST_LatNorth-Methode

Ruft den nördlichsten Breitengrad einer Geometrie ab.

Syntax

geometry-expression.**ST_LatNorth()**

Rückgabe

- **DOUBLE** Gibt den nördlichsten Breitengrad von *geometry-expression* zurück.

Bemerkungen

Gibt den nördlichsten Breitengradwert von *geometry-expression* zurück. Im Modell der gewölbten Erddarstellung entspricht der nördlichste Breitengrad möglicherweise nicht dem Breitengrad eines der Punkte, die die Geometrie definieren.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_LatNorth das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_LatSouth-Methode](#) für den ST_Geometry-Datentyp
- [ST_LongEast-Methode](#) für den ST_Geometry-Datentyp
- [ST_LongWest-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMax-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert 49.74 zurück.

```
SELECT ROUND( NEW ST_LineString( 'LineString( -122 49, -96 49 )', 4326 )
              .ST_LatNorth(), 2 )
```

ST_LatSouth-Methode

Ruft den südlichsten Breitengrad einer Geometrie ab.

Syntax

geometry-expression.ST_LatSouth()

Rückgabe

- **DOUBLE** Gibt den südlichsten Breitengrad von *geometry-expression* zurück.

Bemerkungen

Gibt den südlichsten Breitengradwert von *geometry-expression* zurück. Im Modell der gewölbten Erddarstellung entspricht der südlichste Breitengrad möglicherweise nicht dem Breitengrad eines der Punkte, die die Geometrie definieren.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_LatSouth das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_LatNorth-Methode](#) für den ST_Geometry-Datentyp
- [ST_LongEast-Methode](#) für den ST_Geometry-Datentyp
- [ST_LongWest-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMin-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert 49 zurück.

```
SELECT ROUND( NEW ST_LineString( 'LineString( -122 49, -96 49 )', 4326 )
              .ST_LatSouth(), 2 )
```

ST_LinearHash-Methode

Gibt eine Binärzeichenfolge zurück, die einen linearen Hash der Geometrie darstellt.

Syntax

geometry-expression.ST_LinearHash()

Rückgabe

- **BINARY(32)** Gibt eine Binärzeichenfolge zurück, die einen linearen Hash der Geometrie darstellt.

Bemerkungen

Der räumliche Index unterstützt einen linearen Hash für Geometrien, die die Geometrien in einer Tabelle einer linearen Reihenfolge in einem B-Baumindex zuordnen. Die ST_LinearHash-Methode zeigt diese Zuordnung, indem eine Binärzeichenfolge zurückgegeben wird, die die Reihenfolge der Zeilen im B-Baumindex angeben. Die Hash-Zeichenfolge stellt folgende Eigenschaft bereit: Wenn Geometrie A die Geometrie B abdeckt, dann gilt A.ST_LinearHash() >= B.ST_LinearHash().

Der lineare Hash kann in einer ORDER BY-Klausel verwendet werden. Beispiel: Beim Entladen von Daten mit einer SELECT-Anweisung kann ST_LinearHash verwendet werden, um eine Datendatei zu erstellen, die die Gruppierung eines räumlichen Indexes nachbildet.

Siehe auch

- [ST_LinearUnHash-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

ST_LinearUnHash-Methode

Gibt eine Geometrie zurück, die den Index-Hash darstellt.

Syntax

ST_Geometry::ST_LinearUnHash(*index-hash*[, *srid*])

Parameter

Name	Typ	Beschreibung
Index-Hash	BINARY(32)	Die Index-Hash-Zeichenfolge.
srid	INT	Die SRID des Index-Hashs. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_Geometry** Gibt eine repräsentative Geometrie für den angegebenen linearen Hash zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_LinearUnHash-Methode generiert eine repräsentative Geometrie für eine lineare Hash-Zeichenfolge, die von ST_LinearHash() generiert wird. Der Server ordnet Geometrien einer linearen Reihenfolge für räumliche Indizes zu und die ST_LinearHash-Methode ergibt eine Binärzeichenfolge, die die lineare Reihenfolge definiert. Die ST_LinearUnHash-Methode kehrt diesen Vorgang um, um eine Geometrie zu erzeugen, die eine bestimmte Hash-Zeichenfolge darstellt. Der Hash-Vorgang ist verlustreich, weil mehrere getrennte Geometrien in der gleichen Binärzeichenfolge Hash-Einträge vornehmen können. Die ST_LinearUnHash-Methode gibt eine Geometrie zurück, die jede Geometrie abdeckt, die dem gegebenen linearen Hash zugeordnet ist.

Der grafische Plan für eine Abfrage, die einen räumlichen Index verwendet, zeigt die linearen Hash-Werte, die zum Absuchen des räumlichen Indexes verwendet werden. Die ST_LinearUnHash-Methode kann verwendet werden, um eine Geometrie zu erstellen, die diese Hashes darstellt.

Siehe auch

- [ST_LinearHash-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

ST_LoadConfigurationData-Methode

Gibt binäre Konfigurationsdaten zurück. Wird nur intern verwendet.

Syntax

ST_Geometry::ST_LoadConfigurationData(*configuration-name*)

Parameter

Name	Typ	Beschreibung
configuration-name	VARCHAR(128)	Der Name des zu ladenden Konfigurationsdatenelements.

Rückgabe

- **LONG BINARY** Gibt binäre Konfigurationsdaten zurück. Wird nur intern verwendet.

Bemerkungen

Diese Methode wird vom Server verwendet, um Konfigurationsdaten aus installierten Dateien zu laden. Wenn die Konfigurationsdateien nicht mit dem Server installiert sind, wird NULL zurückgegeben. Wird nur intern verwendet.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

ST_LongEast-Methode

Ruft den Längengrad der östlichen Begrenzung einer Geometrie ab.

Syntax

geometry-expression.**ST_LongEast**()

Rückgabe

- **DOUBLE** Ruft den Längengrad der östlichen Begrenzung von *geometry-expression* ab.

Bemerkungen

Gibt den Längengrad der östlichen Begrenzung von *geometry-expression* zurück. Wenn die Geometrie im Modell der gewölbten Erddarstellung die Datumsgrenze überschreitet, ist ST_LongWest größer als der ST_LongEast-Wert.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_LongEast das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_LongWest-Methode](#) für den ST_Geometry-Datentyp
- [ST_LatNorth-Methode](#) für den ST_Geometry-Datentyp
- [ST_LatSouth-Methode](#) für den ST_Geometry-Datentyp
- [ST_XMax-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert -157.8 zurück.

```
SELECT NEW ST_LineString( 'LineString( -157.8 21.3, 144.5 13 )', 4326 )
      .ST_LongEast()
```

ST_LongWest-Methode

Ruft den Längengrad der westlichen Begrenzung einer Geometrie ab.

Syntax

geometry-expression.ST_LongWest()

Rückgabe

- **DOUBLE** Ruft den Längengrad der westlichen Begrenzung von *geometry-expression* ab.

Bemerkungen

Gibt den Längengrad der westlichen Begrenzung von *geometry-expression* zurück. Wenn die Geometrie im Modell der gewölbten Erddarstellung die Datumsgrenze überschreitet, ist ST_LongWest größer als der ST_LongEast-Wert.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_LongWest das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_LongEast-Methode](#) für den ST_Geometry-Datentyp
- [ST_LatNorth-Methode](#) für den ST_Geometry-Datentyp
- [ST_LatSouth-Methode](#) für den ST_Geometry-Datentyp
- [ST_XMin-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert 144.5 zurück.

```
SELECT NEW ST_LineString( 'LineString( -157.8 21.3, 144.5 13 )', 4326 )
      .ST_LongWest()
```

ST_MMax-Methode

Ruft den maximalen M-Koordinatenwert einer Geometrie ab.

Syntax

geometry-expression.ST_MMax()

Rückgabe

- **DOUBLE** Gibt den maximalen M-Koordinatenwert von *geometry-expression* zurück.

Bemerkungen

Gibt den maximalen M-Koordinatenwert von *geometry-expression* zurück. Dies wird durch einen Vergleich des M-Attributs aller Punkte in der Geometrie berechnet.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_MMax das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_XMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_XMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_ZMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_ZMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_MMin-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert 8 zurück.

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_MMax()
```

ST_MMin-Methode

Ruft den minimalen M-Koordinatenwert einer Geometrie ab.

Syntax

geometry-expression.**ST_MMin**()

Rückgabe

- **DOUBLE** Gibt den minimalen M-Koordinatenwert von *geometry-expression* zurück.

Bemerkungen

Gibt den minimalen M-Koordinatenwert von *geometry-expression* zurück. Dies wird durch einen Vergleich des M-Attributs aller Punkte in der Geometrie berechnet.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_MMin das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_XMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_XMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_ZMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_ZMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_MMax-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert 4 zurück.

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_MMin()
```

ST_OrderingEquals-Methode

Testet, ob eine Geometrie identisch mit einer anderen Geometrie ist.

Syntax

geometry-expression.**ST_OrderingEquals**(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn die beiden Geometriewerte genau gleich sind, sonst 0.

Bemerkungen

Testet, ob ein ST_Geometry-Wert mit einem anderen ST_Geometry-Wert identisch ist. Die beiden Geometrien müssen dieselbe Hierarchie von Objekten mit genau denselben Punkten in derselben Reihenfolge enthalten, um mit ST_OrderingEquals als gleich eingestuft zu werden.

Die ST_OrderingEquals-Methode unterscheidet sich von ST_Equals dadurch, dass sie die Orientierung von Kurven berücksichtigt. Zwei Kurven können die gleichen Punkte enthalten, aber in umgekehrter Reihenfolge. Diese beiden Kurven sind mit ST_Equals gleich, aber mit ST_OrderingEquals ungleich. Außerdem erfordert ST_OrderingEquals, dass jeder Punkt beider Geometrien genau gleich ist, nicht nur innerhalb der Toleranz/Entfernung, die im räumlichen Bezugssystem festgelegt ist.

Die ST_OrderingEquals-Methode definiert die Semantik für die Vergleichsprädikate (= und <>), IN-Listenprädikate, DISTINCT und GROUP BY. Wenn Sie vergleichen möchten, ob zwei räumliche Werte

räumlich gleich sind (die gleiche Menge von Punkten in der Menge enthalten), können Sie die ST_Equals-Methode verwenden.

Weitere Hinweise finden Sie unter „[Funktionsweise räumlicher Vergleiche](#)“ auf Seite 51.

Hinweis

Der SQL/MM-Standard definiert, dass ST_OrderingEquals eine relative Reihenfolge zurückgeben muss, wobei 0 zurückgegeben wird, wenn zwei Geometrien räumlich gleich sind (gemäß ST_Equals), und 1, wenn sie nicht gleich sind. Die SQL Anywhere-Implementierung folgt der branchenüblichen Handhabung und unterscheidet sich insofern von SQL/MM, als sie einen booleschen Wert zurückgibt, bei dem 1 anzeigt, dass die Geometrien gleich sind, und 0, um anzuzeigen, dass sie unterschiedlich sind. Außerdem unterscheidet sich die ST_OrderingEquals-Implementierung von SQL/MM, weil sie testet, ob die Werte identisch sind (dieselbe Objekthierarchie in derselben Reihenfolge) und nicht räumlich gleich (gleiche Punktmenge im Raum).

Siehe auch

- [ST_Equals-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.43

Beispiel

Das folgende Beispiel gibt den Wert 16 zurück. Wenn Shape der ShapeID entspricht, enthält das Ergebnis 16 genau dieselben Punkte in genau derselben Reihenfolge wie das Polygon.

```
SELECT ShapeID FROM SpatialShapes
WHERE Shape.ST_OrderingEquals( NEW ST_Polygon( 'Polygon ((0 0, 2 0, 1 2, 0
0))' ) ) = 1
```

ST_Overlaps-Methode

Testet, ob ein Geometriewert einen anderen Geometriewert überlappt.

Syntax

geometry-expression.ST_Overlaps(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* den Wert *geo2* überlappt, sonst 0. Gibt Null zurück, wenn *geometry-expression* und *geo2* verschiedene Dimensionen haben.

Bemerkungen

Zwei Geometrien überlappen einander, wenn die folgenden Bedingungen gelten:

- Beide Geometrien haben dieselbe Dimension.
- Die Schnittmenge von *geometry-expression* und *geo2* hat dieselbe Dimension wie *geometry-expression*.
- Keine der ursprünglichen Geometrien ist eine Teilmenge der anderen.

Genauer gesagt gibt *geometry-expression.ST_Overlaps(geo2)* 1 zurück, wenn Folgendes TRUE ist:

```
geometry-expression.ST_Dimension() = geo2.ST_Dimension() AND geometry-
expression.ST_Intersection( geo2 ).ST_Dimension() = geometry-
expression.ST_Dimension() AND geometry-expression.ST_Covers( geo2 ) = 0 AND
geo2.ST_Covers( geometry-expression ) = 0
```

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

- [ST_Dimension-Methode für den ST_Geometry-Datentyp](#)
- [ST_Intersects-Methode für den ST_Geometry-Datentyp](#)
- [ST_Covers-Methode für den ST_Geometry-Datentyp](#)
- [ST_Crosses-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.32

Beispiel

Der folgende Code gibt das Ergebnis 1 zurück, da die Schnittmenge der beiden Linienfolgen ebenfalls eine Linienfolge ist und keine der beiden Geometrien eine Untermenge der anderen ist.

```
SELECT NEW ST_LineString( 'LineString( 0 0, 5 0 )' )
.ST_Overlaps( NEW ST_LineString( 'LineString( 2 0, 3 0, 3 3 )' ) )
```

Der folgende Code gibt das Ergebnis NULL zurück, da die Linienfolge und der Punkt unterschiedliche Dimensionen haben.

```
SELECT NEW ST_LineString( 'LineString( 0 0, 5 0 )' )
.ST_Overlaps( NEW ST_Point( 1, 0 ) )
```

Der folgende Code gibt das Ergebnis 0 zurück, da der Punkt eine Teilmenge des Mehrfachpunkts ist.

```
SELECT NEW ST_MultiPoint( 'MultiPoint(( 2 3 ), ( 1 0 ))' )
.ST_Overlaps( NEW ST_Point( 1, 0 ) )
```

Der folgende Code gibt das Ergebnis 24 , 25 , 28 , 31 zurück, bei dem es sich um die Liste der ShapeIDs handelt, die das angegebene Polygon überlappen.

```
SELECT LIST( ShapeID ORDER BY ShapeID ) FROM SpatialShapes
WHERE Shape.ST_Overlaps( NEW ST_Polygon( 'Polygon(( -1 0, 0 0, 0 1, -1 1, -1
0 ))' )
                        ) = 1
```

ST_Relate-Methode

Testet, ob ein Geometriewert räumlich mit einem anderen, in der Schnittmatrix angegebenen Geometriewert in Beziehung steht. Die ST_Relate-Methode verwendet eine aus 9 Stellen bestehende Zeichenfolge aus dem Dimensionally Extended 9 Intersection Model (DE-9IM), um die paarweise Beziehung zwischen zwei räumlichen Datenelementen zu beschreiben. Die ST_Relate-Methode ermittelt beispielsweise, ob eine Schnittmenge zwischen den Geometrien vorliegt und gegebenenfalls die Geometrie der resultierenden Schnittmenge. Siehe auch: „Funktionsweise räumlicher Beziehungen“ auf Seite 52.

Überladungsliste

Name	Beschreibung
„ST_Relate(ST_Geometry,CHAR(9))-Methode für den ST_Geometry-Datentyp“	Testet, ob ein Geometriewert räumlich mit einem anderen, in der Schnittmatrix angegebenen Geometriewert in Beziehung steht. Die ST_Relate-Methode verwendet eine aus 9 Stellen bestehende Zeichenfolge aus dem Dimensionally Extended 9 Intersection Model (DE-9IM), um die paarweise Beziehung zwischen zwei räumlichen Datenelementen zu beschreiben. Die ST_Relate-Methode ermittelt beispielsweise, ob eine Schnittmenge zwischen den Geometrien vorliegt und gegebenenfalls die Geometrie der resultierenden Schnittmenge.
„ST_Relate(ST_Geometry)-Methode für den ST_Geometry-Datentyp“	Legt durch die Rückgabe einer Schnittmatrix fest, wie ein Geometriewert mit einem anderen Geometriewert in Beziehung steht. Die ST_Relate-Methode gibt eine aus 9 Stellen bestehende Zeichenfolge aus dem Dimensionally Extended 9 Intersection Model (DE-9IM) zurück, um die paarweise Beziehung zwischen zwei räumlichen Datenelementen zu beschreiben. Die ST_Relate-Methode ermittelt beispielsweise, ob eine Schnittmenge zwischen den Geometrien vorliegt und gegebenenfalls die Geometrie der resultierenden Schnittmenge.

ST_Relate(ST_Geometry,CHAR(9))-Methode für den ST_Geometry-Datentyp

Testet, ob ein Geometriewert räumlich mit einem anderen, in der Schnittmatrix angegebenen Geometriewert in Beziehung steht. Die ST_Relate-Methode verwendet eine aus 9 Stellen bestehende Zeichenfolge aus dem Dimensionally Extended 9 Intersection Model (DE-9IM), um die paarweise Beziehung zwischen zwei räumlichen Datenelementen zu beschreiben. Die ST_Relate-Methode ermittelt beispielsweise, ob eine Schnittmenge zwischen den Geometrien vorliegt und gegebenenfalls die Geometrie der resultierenden Schnittmenge.

Syntax

geometry-expression.**ST_Relate**(*geo2*,*relate-matrix*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der zweite Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.
relate-matrix	CHAR(9)	Eine aus neun Zeichen bestehende Zeichenfolge, die eine Matrix im Dimensionally Extended 9 Intersection Model darstellt. Jedes in der neunstelligen Zeichenfolge definierte Zeichen stellt den Typ der Schnittmenge dar, der an einer der neun möglichen Schnittstellen zwischen dem Innenbereich, der Begrenzung und dem Außenbereich der beiden Geometrien zulässig ist.

Rückgabe

- **BIT** Gibt 1 zurück, wenn die beiden Geometrien die angegebene Beziehung haben, sonst 0.

Bemerkungen

Testet, ob ein Geometriewert räumlich mit einem anderen Geometriewert in Beziehung steht, indem geprüft wird, ob eine Schnittmenge zwischen dem Innenbereich, der Begrenzung und dem Außenbereich zweier Geometrien gemäß der Angabe in der Schnittmatrix vorliegt. Siehe auch: „[Funktionsweise räumlicher Beziehungen](#)“ auf Seite 52.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.25

Beispiel

Das folgende Beispiel gibt eine Ergebnismenge mit einer Zeile für jede Form zurück, die die Beziehung '0F***T***' mit der angegebenen Linie enthält. '0' bedeutet, dass das Innere der beiden Geometrien eine Schnittmenge haben muss und einen Punkt oder einen Mehrfachpunkt ergibt. 'F' bedeutet, dass das Innere der Linie und die Begrenzung der Form keine Schnittmenge haben dürfen. 'T' bedeutet, dass das Äußere der Linie und das Innere der Form eine Schnittmenge haben müssen.

```
SELECT ShapeID, "Description" From SpatialShapes
WHERE NEW ST_LineString( 'LineString( 0 0, 10 0 )' )
.ST_Relate( Shape, '0F***T***' ) = 1
ORDER BY ShapeID
```

Die Abfrage erzeugt die folgende Ergebnismenge:

ShapeID	Description
18	CircularString
30	Multicurve

ST_Relate(ST_Geometry)-Methode für den ST_Geometry-Datentyp

Legt durch die Rückgabe einer Schnittmatrix fest, wie ein Geometriewert mit einem anderen Geometriewert in Beziehung steht. Die ST_Relate-Methode gibt eine aus 9 Stellen bestehende Zeichenfolge aus dem Dimensionally Extended 9 Intersection Model (DE-9IM) zurück, um die paarweise Beziehung zwischen zwei räumlichen Datenelementen zu beschreiben. Die ST_Relate-Methode ermittelt beispielsweise, ob eine Schnittmenge zwischen den Geometrien vorliegt und gegebenenfalls die Geometrie der resultierenden Schnittmenge.

Syntax

geometry-expression.ST_Relate(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der zweite Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **CHAR(9)** Gibt eine aus neun Stellen bestehende Zeichenfolge zurück, die eine Matrix im Dimensionally Extended 9 Intersection Model (DE-9IM) darstellt. Jedes Zeichen in der neunstelligen Zeichenfolge stellt den Typ der Schnittmenge an einer der neun möglichen Schnittstellen zwischen dem Innenbereich, der Begrenzung und dem Außenbereich der beiden Geometrien dar.

Bemerkungen

Testet, ob ein Geometriewert räumlich mit einem anderen Geometriewert in Beziehung steht, indem geprüft wird, ob eine Schnittmenge zwischen dem Innenbereich, der Begrenzung und dem Außenbereich zweier Geometrien gemäß der Angabe in der Schnittmatrix vorliegt.

Die Schnittmatrix wird als Zeichenfolge zurückgegeben. Es ist zwar möglich, diese Methodenvariante zu verwenden, um eine räumliche Beziehung zu testen, indem die zurückgegebene Zeichenfolge untersucht wird, dennoch ist es effizienter, die Beziehungen zu testen, indem eine Musterzeichenfolge als zweiter Parameter übergeben wird oder indem spezifische räumliche Prädikate wie ST_Contains oder ST_Intersects verwendet werden. Siehe auch: „[Funktionsweise räumlicher Beziehungen](#)“ auf Seite 52.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert 1F2001102 zurück.

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 0 2, 0 0 ))' )
.ST_Relate( NEW ST_LineString( 'LineString( 0 1, 5 1 )' ) )
```

ST_Reverse-Methode

Gibt die Geometrie mit einer umgekehrten Elementreihenfolge zurück.

Syntax

geometry-expression.**ST_Reverse()**

Rückgabe

- **ST_Geometry** Gibt die Geometrie mit einer umgekehrten Elementreihenfolge zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Gibt die Geometrie mit umgekehrter Reihenfolge der Elemente zurück. Bei einer Kurve wird eine Kurve mit umgekehrten Scheiteln zurückgegeben. Bei einer Sammlung wird eine Sammlung mit untergeordneten Geometrien in umgekehrter Reihenfolge zurückgegeben.

Hinweis

Standardmäßig verwendet ST_Reverse das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert LineString (3 4, 1 2) zurück. Es zeigt, wie die Reihenfolge von Punkten in einer Linienfolge durch ST_Reverse umgekehrt wird.

```
SELECT NEW ST_LineString( NEW ST_Point(1,2), NEW ST_Point(3,4) ).ST_Reverse()
```

ST_SRID-Methode

Ruft das mit dem Geometriewert verbundene räumliche Bezugssystem ab oder verändert es.

Überladungsliste

Name	Beschreibung
„ST_SRID()-Methode für den ST_Geometry-Datentyp“	Gibt die SRID der Geometrie zurück.
„ST_SRID(INT)-Methode für den ST_Geometry-Datentyp“	Ändert das mit der Geometrie verbundene räumliche Bezugssystem, ohne einen der Werte zu verändern.

ST_SRID()-Methode für den ST_Geometry-Datentyp

Gibt die SRID der Geometrie zurück.

Syntax

geometry-expression.ST_SRID()

Rückgabe

- **INT** Gibt die SRID der Geometrie zurück.

Bemerkungen

Gibt die SRID der Geometrie zurück. Jede Geometrie ist mit einem räumlichen Bezugssystem verbunden.

Hinweis

Standardmäßig verwendet ST_SRID das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.5

Beispiel

Der folgende Code gibt das Ergebnis 0 zurück, das anzeigt, dass alle Formen in der Tabelle die SRID 0 haben, was dem räumlichen Standard-Bezugssystem ('Default') entspricht.

```
SELECT DISTINCT Shape.ST_SRID() FROM SpatialShapes
```

ST_SRID(INT)-Methode für den ST_Geometry-Datentyp

Ändert das mit der Geometrie verbundene räumliche Bezugssystem, ohne einen der Werte zu verändern.

Syntax

geometry-expression.**ST_SRID**(*srid*)

Parameter

Name	Typ	Beschreibung
srid	INT	Die für das Ergebnis zu verwendende SRID.

Rückgabe

- **ST_Geometry** Gibt eine Kopie des Geometriewerts mit dem angegebenen räumlichen Bezugssystem zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_SRID-Methode erstellt eine Kopie von *geometry-expression* mit der durch den srid-Parameter angegebenen SRID. ST_SRID kann verwendet werden, wenn das räumliche Quellbezugssystem und das räumliche Zielbezugssystem dasselbe Koordinatensystem haben.

Wenn Sie eine Transformation einer Geometrie zwischen zwei räumlichen Bezugssystemen vornehmen, die verschiedene Koordinatensysteme haben, müssen Sie die ST_Transform-Methode verwenden.

Hinweis

Standardmäßig verwendet ST_SRID das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_Transform-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.5

Beispiel

Das folgende Beispiel gibt den Wert SRID=1000004326;Point (-118 34) zurück.

```
SELECT NEW ST_Point( -118, 34,
4326 ).ST_SRID( 1000004326 ).ST_AsText( 'EWKT' )
```

ST_SRIDFromBaseType-Methode

Führt die syntaktische Analyse einer Zeichenfolge durch, die die Typzeichenfolge definiert.

Syntax

```
ST_Geometry::ST_SRIDFromBaseType(base-type-str)
```

Parameter

Name	Typ	Beschreibung
base-type-str	VARCHAR(128)	Eine Zeichenfolge, die die Basistypzeichenfolge enthält

Rückgabe

- **INT** Gibt die SRID aus einer Typzeichenfolge zurück. Wenn in der Zeichenfolge keine SRID angegeben ist, wird NULL zurückgegeben. Wenn die Typzeichenfolge keine gültige Geometrietypzeichenfolge ist, wird ein Fehler zurückgegeben.

Bemerkungen

Die *ST_Geometry::ST_SRIDFromBaseType*-Methode kann verwendet werden, um die SRID syntaktisch aus einer Typenzeichenfolgendefinition herauszuanalysieren.

Siehe auch

- [ST_GeometryTypeFromBaseType-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert NULL zurück.

```
SELECT ST_Geometry::ST_SRIDFromBaseType('ST_Geometry')
```

Das folgende Beispiel gibt den Wert 4326 zurück.

```
SELECT ST_Geometry::ST_SRIDFromBaseType('ST_Geometry(SRID=4326)')
```

ST_SnapToGrid-Methode

Gibt eine Kopie der Geometrie zurück, in der alle Punkte am angegebenen Raster ausgerichtet sind.

Überladungsliste

Name	Beschreibung
„ST_SnapToGrid(DOUBLE)-Methode für den ST_Geometry-Datentyp“	Gibt eine Kopie der Geometrie zurück, in der alle Punkte am angegebenen Raster ausgerichtet sind.

Name	Beschreibung
„ST_SnapToGrid(ST_Point,DOUBLE,DOUBLE,DOUBLE,DOUBLE)-Methode für den ST_Geometry-Datentyp“	Gibt eine Kopie der Geometrie zurück, in der alle Punkte am angegebenen Raster ausgerichtet sind.

ST_SnapToGrid(DOUBLE)-Methode für den ST_Geometry-Datentyp

Gibt eine Kopie der Geometrie zurück, in der alle Punkte am angegebenen Raster ausgerichtet sind.

Syntax

geometry-expression.**ST_SnapToGrid**(*cell-size*)

Parameter

Name	Typ	Beschreibung
cell-size	DOUBLE	Die Zellengröße für das Raster.

Rückgabe

- **ST_Geometry** Gibt die Geometrie zurück, in der alle Punkte am angegebenen Raster ausgerichtet sind.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Die ST_SnapToGrid-Methode kann verwendet werden, um die Präzision der Daten zu reduzieren, indem alle Punkte in einer Geometrie an einem Raster ausgerichtet werden, das nach Ursprung und Zellengröße definiert wird.

Die X- und Y-Koordinaten werden am Raster ausgerichtet. Z- und M-Werte werden nicht geändert.

Hinweis

Das Reduzieren der Präzision kann dazu führen, dass die sich daraus ergebende Geometrie andere Eigenschaften hat. Zum Beispiel kann dies dazu führen, dass eine einfache Linienfolge sich selbst schneidet, oder es kann eine ungültige Geometrie erzeugt werden.

Hinweis

Jedes räumliche Bezugssystem definiert ein Raster, an dem alle Geometrien automatisch ausgerichtet werden. Sie können keine höhere Präzision speichern als durch dieses vordefinierte Raster vorgegeben.

Hinweis

Standardmäßig verwendet ST_SnapToGrid das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [„Wie sich Ausrichten am Raster und Toleranz auf räumliche Berechnungen auswirken“](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert LineString (1.536133 6.439453, 2.173828 6.100586) zurück.

```
SELECT NEW ST_LineString( 'LineString( 1.5358 6.4391, 2.17401 6.10018 )' )
.ST_SnapToGrid( 0.001 )
```

Jede X- und Y-Koordinate wird unter Verwendung einer Rastergröße von etwa 0,001 zum nächst gelegenen Rasterpunkt verschoben. Die tatsächlich verwendete Rastergröße entspricht nicht genau der angegebenen Rastergröße.

**ST_SnapToGrid(ST_Point,DOUBLE,DOUBLE,DOUBLE,DOUBLE)-
Methode für den ST_Geometry-Datentyp**

Gibt eine Kopie der Geometrie zurück, in der alle Punkte am angegebenen Raster ausgerichtet sind.

Syntax

geometry-expression.**ST_SnapToGrid**(*origin,cell-size-x,cell-size-y,cell-size-z,cell-size-m*)

Parameter

Name	Typ	Beschreibung
origin	ST_Point	Der Ursprung des Rasters.
cell-size-x	DOUBLE	Die Zellengröße für das Raster in der X-Dimension.
cell-size-y	DOUBLE	Die Zellengröße für das Raster in der Y-Dimension.
cell-size-z	DOUBLE	Die Zellengröße für das Raster in der Z-Dimension.
cell-size-m	DOUBLE	Die Zellengröße für das Raster in der M-Dimension.

Rückgabe

- **ST_Geometry** Gibt die Geometrie zurück, in der alle Punkte am angegebenen Raster ausgerichtet sind.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Die ST_SnapToGrid-Methode kann verwendet werden, um die Präzision von Daten zu reduzieren, indem alle Punkte in einer Geometrie an einem Raster ausgerichtet werden, das durch Ursprung und Zellegröße definiert ist.

Sie können für jede Dimension eine andere Zellengröße definieren. Wenn die Punkte in einer Dimension nicht ausgerichtet werden sollen, können Sie eine Zellengröße null verwenden.

Hinweis

Das Reduzieren der Präzision kann dazu führen, dass die sich daraus ergebende Geometrie andere Eigenschaften hat. Zum Beispiel kann dies dazu führen, dass eine einfache Linienfolge sich selbst schneidet, oder es kann eine ungültige Geometrie erzeugt werden.

Hinweis

Jedes räumliche Bezugssystem definiert ein Raster, an dem alle Geometrien automatisch ausgerichtet werden. Sie können keine höhere Präzision speichern als durch dieses vordefinierte Raster vorgegeben.

Hinweis

Standardmäßig verwendet ST_SnapToGrid das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Siehe auch

- [„Wie sich Ausrichten am Raster und Toleranz auf räumliche Berechnungen auswirken“](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert LineString (1.010101 20.20202, 1.015625 20.203125, 1.01 20.2) zurück.

```
SELECT NEW ST_LineString(
  NEW ST_Point( 1.010101, 20.202020 ),
  TREAT( NEW ST_Point( 1.010101, 20.202020 ).ST_SnapToGrid( NEW
ST_Point( 0.0, 0.0 ), POWER( 2, -6 ), POWER( 2, -7 ), 0.0, 0.0 ) AS
ST_Point ),
  TREAT( NEW ST_Point( 1.010101, 20.202020 ).ST_SnapToGrid( NEW
ST_Point( 1.01, 20.2 ), POWER( 2, -6 ), POWER( 2, -7 ), 0.0, 0.0 ) AS
ST_Point ) )
```

Der erste Punkt der Linienfolge ist `ST_Point(1.010101, 20.202020)`, ausgerichtet am für SRID 0 definierten Raster.

Der zweite Punkt der Linienfolge ist derselbe Punkt, ausgerichtet an einem Raster, das mit seinem Ursprung an Punkt (0.0 0.0) definiert ist, wobei die Zellengröße x `POWER(2, -6)` und die Zellengröße y `POWER(2, -7)` ist.

Der dritte Punkt der Linienfolge ist derselbe Punkt, ausgerichtet an einem Raster, das mit seinem Ursprung am Punkt (1.01, 20.2) definiert ist, wobei die Zellengröße x `POWER(2, -6)` und die Zellengröße y `POWER(2, -7)` ist.

ST_SymDifference-Methode

Gibt den Geometriewert zurück, der die symmetrische Differenz der Punktmenge zweier Geometrien darstellt.

Syntax

geometry-expression.**ST_SymDifference**(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der von <i>geometry-expression</i> subtrahiert wird, um die symmetrische Differenz zu finden.

Rückgabe

- **ST_Geometry** Gibt den Geometriewert zurück, der die symmetrische Differenz der Punktmenge zweier Geometrien darstellt.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Die ST_SymDifference-Methode findet die symmetrische Differenz zweier Geometrien. Die symmetrische Differenz besteht aus allen Punkten, die sich nur in einer der beiden Geometrien befinden. Wenn die beiden Geometriewerte aus denselben Punkten bestehen, gibt die ST_SymDifference-Methode eine leere Geometrie zurück.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Siehe auch

- [ST_Difference-Methode für den ST_Geometry-Datentyp](#)
- [ST_Intersection-Methode für den ST_Geometry-Datentyp](#)
- [ST_Union-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

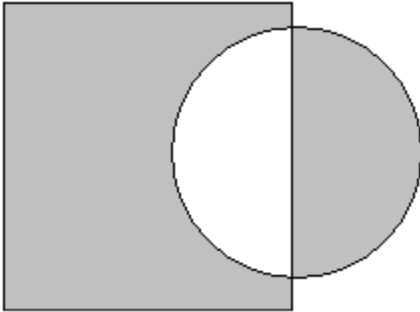
- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.21

Beispiel

Das folgende Beispiel zeigt den symmetrischen Unterschied (C) eines Rechtecks (A) und eines Kreises (B).

```
SELECT NEW ST_Polygon( 'Polygon( (-1 -0.25, 1 -0.25, 1 2.25, -1 2.25, -1
-0.25) )' ) AS A
, NEW ST_CurvePolygon( 'CurvePolygon( CircularString( 0 1, 1 2, 2 1, 1
0, 0 1 ) )' ) AS B
, A.ST_SymDifference( B ) AS C
```

Das folgende Bild zeigt das Ergebnis der symmetrischen Differenz als schattierten Teil des Bilds. Die symmetrische Differenz ist eine Mehrfachoberfläche, die zwei Flächen umfasst: Eine Fläche enthält alle Punkte des Quadrats, die sich nicht im Kreis befinden und die andere Fläche enthält alle Punkte des Kreises, die sich nicht im Rechteck befinden.



ST_ToCircular-Methode

Konvertiert die Geometrie in eine Kreisbogenfolge.

Syntax

geometry-expression.**ST_ToCircular**()

Rückgabe

- **ST_CircularString** Wenn *geometry-expression* vom Typ ST_CircularString ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* vom Typ ST_CompoundCurve mit einem einzelnen Element vom Datentyp ST_CircularString ist, wird dieses Element zurückgegeben. Wenn *geometry-expression* eine Geometriegruppe mit einem einzelnen Element vom Typ ST_CircularString ist, wird dieses Element zurückgegeben. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ ST_CircularString zurückgegeben. Andernfalls wird eine Ausnahmerebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Konvertiert diese Geometrie in eine Kreisbogenfolge. Die Logik ist äquivalent zu derjenigen, die für `CAST(geometry-expression AS ST_CircularString)` verwendet wird.

Wenn *geometry-expression* bereits als `ST_CircularString`-Wert bekannt ist, sollte für höhere Effizienz `TREAT(geometry-expression AS ST_CircularString)` und nicht die `ST_ToCircular`-Methode verwendet werden.

Hinweis

Standardmäßig verwendet `ST_ToCircular` das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToCompound-Methode](#) für den `ST_Geometry`-Datentyp
- [ST_ToCurve-Methode](#) für den `ST_Geometry`-Datentyp
- [ST_ToLineString-Methode](#) für den `ST_Geometry`-Datentyp
- [ST_ToMultiCurve-Methode](#) für den `ST_Geometry`-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

Beispiel

Das folgende Beispiel gibt den Wert `CircularString (0 0, 1 1, 2 0)` zurück.

```
SELECT NEW ST_CompoundCurve( 'CompoundCurve(CircularString( 0 0, 1 1, 2 0 ))' ).ST_ToCircular()
```

ST_ToCompound-Methode

Konvertiert die Geometrie in eine Verbundkurve.

Syntax

geometry-expression.**ST_ToCompound()**

Rückgabe

- **ST_CompoundCurve** Wenn *geometry-expression* vom Typ `ST_CompoundCurve` ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* vom Typ `ST_LineString` oder `ST_CircularString` ist, wird eine Verbundkurve mit einem Element zurückgegeben, nämlich *geometry-expression*. Wenn *geometry-expression* eine Geometriegruppe mit einem einzelnen Element vom Typ `ST_Curve` ist, wird dieses Element in `ST_CompoundCurve` umgewandelt. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ `ST_CompoundCurve` zurückgegeben. Andernfalls wird eine Ausnahmebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Konvertiert die Geometrie in eine Kreisbogenfolge. Die Logik ist äquivalent zu derjenigen, die für `CAST(geometry-expression AS ST_CompoundCurve)` verwendet wird.

Wenn *geometry-expression* bereits als ST_CompoundCurve-Wert bekannt ist, sollte für höhere Effizienz `TREAT(geometry-expression AS ST_CompoundCurve)` und nicht die ST_ToCompound-Methode verwendet werden.

Hinweis

Standardmäßig verwendet ST_ToCompound das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToCircular-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToCurve-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToLineString-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToMultiCurve-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

Beispiel

Das folgende Beispiel gibt den Wert CompoundCurve ((0 0, 2 1)) zurück.

```
SELECT NEW ST_LineString( 'LineString( 0 0, 2 1 )' ).ST_ToCompound()
```

ST_ToCurve-Methode

Konvertiert die Geometrie in eine Kurve.

Syntax

geometry-expression.**ST_ToCurve()**

Rückgabe

- **ST_Curve** Wenn *geometry-expression* vom Typ ST_Curve ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* eine Geometriegruppe mit einem einzelnen Element vom Typ ST_Curve ist, wird dieses Element zurückgegeben. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ ST_LineString zurückgegeben. Andernfalls wird eine Ausnahmebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Konvertiert die Geometrie in eine Kurve. Die Logik ist äquivalent zu derjenigen, die für **CAST**(*geometry-expression* AS **ST_Curve**) verwendet wird.

Wenn *geometry-expression* bereits als ST_Curve-Wert bekannt ist, sollte für höhere Effizienz **TREAT**(*geometry-expression* AS **ST_Curve**) und nicht die ST_ToCurve-Methode verwendet werden.

Hinweis

Standardmäßig verwendet ST_ToCurve das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToCircular-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToCompound-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToLineString-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToMultiCurve-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert LineString (0 0, 1 1, 2 0) zurück.

```
SELECT NEW ST_GeomCollection( 'GeometryCollection(LineString(0 0, 1 1, 2 0))' ).ST_ToCurve()
```

ST_ToCurvePoly-Methode

Konvertiert die Geometrie in ein Kurvenpolygon.

Syntax

geometry-expression.**ST_ToCurvePoly**()

Rückgabe

- **ST_CurvePolygon** Wenn *geometry-expression* vom Typ ST_CurvePolygon ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* eine Geometriegruppe mit einem einzelnen Element vom Typ ST_CurvePolygon ist, wird dieses Element zurückgegeben. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ ST_CurvePolygon zurückgegeben. Andernfalls wird eine Ausnahmebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Wenn *geometry-expression* bereits als ST_CurvePolygon-Wert bekannt ist, sollte für höhere Effizienz `TREAT(geometry-expression AS ST_CurvePolygon)` und nicht die ST_ToCurvePoly-Methode verwendet werden.

Hinweis

Standardmäßig verwendet ST_ToCurvePoly das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToPolygon-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToSurface-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToMultiSurface-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

Beispiel

Das folgende Beispiel gibt den Wert Polygon ((0 0, 2 0, 1 2, 0 0)) zurück.

```
SELECT NEW ST_MultiPolygon('MultiPolygon(((0 0, 2 0, 1 2, 0
0)))' ).ST_ToCurvePoly()
```

ST_ToGeomColl-Methode

Konvertiert die Geometrie in eine Geometriegruppe.

Syntax

geometry-expression.**ST_ToGeomColl**()

Rückgabe

- **ST_GeomCollection** Wenn *geometry-expression* vom Typ ST_GeomCollection ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* vom Typ ST_Point, ST_Curve oder ST_Surface ist, wird eine Geometriegruppe mit einem Element zurückgegeben, nämlich *geometry-expression*. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ ST_GeomCollection zurückgegeben. Andernfalls wird eine Ausnahmebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Wenn *geometry-expression* bereits als ST_GeomCollection-Wert bekannt ist, sollte für höhere Effizienz `TREAT(geometry-expression AS ST_CurvePolygon)` und nicht die ST_ToCurvePoly-Methode verwendet werden.

Hinweis

Standardmäßig verwendet ST_ToGeomColl das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToMultiCurve-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToMultiPoint-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToMultiSurface-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

Beispiel

Das folgende Beispiel gibt den Wert `GeometryCollection (Point (0 1))` zurück.

```
SELECT NEW ST_Point( 0, 1 ).ST_ToGeomColl()
```

ST_ToLineString-Methode

Konvertiert die Geometrie in eine Linienfolge.

Syntax

geometry-expression.**ST_ToLineString()**

Rückgabe

- **ST_LineString** Wenn *geometry-expression* vom Typ ST_LineString ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* vom Typ ST_CircularString oder ST_CompoundCurve ist, wird *geometry-expression*.ST_CurveToLine() zurückgegeben. Wenn *geometry-expression* eine Geometriegruppe mit einem einzelnen Element vom Typ ST_Curve ist, wird dieses Element in ST_LineString umgewandelt. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ ST_LineString zurückgegeben. Andernfalls wird eine Ausnahmebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Konvertiert die Geometrie in eine Linienfolge. Die Logik ist äquivalent zu derjenigen, die für `CAST(geometry-expression AS ST_LineString)` verwendet wird. Wenn *geometry-expression* eine Kreisbogenfolge oder eine Verbundkurve ist, wird sie mit ST_CurveToLine() interpoliert.

Wenn *geometry-expression* bereits als ST_LineString-Wert bekannt ist, sollte für höhere Effizienz `TREAT(geometry-expression AS ST_LineString)` und nicht die ST_ToLineString-Methode verwendet werden.

Hinweis

Standardmäßig verwendet ST_ToLineString das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToMultiLine-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToCircular-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToCompound-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToCurve-Methode](#) für den ST_Geometry-Datentyp
- [ST_CurveToLine-Methode](#) für den ST_Curve-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

Beispiel

Der folgende Code gibt einen Fehler zurück, weil die Shape-Spalte vom Typ ST_Geometry ist und ST_Geometry die ST_Length-Methode nicht unterstützt.

```
SELECT Shape.ST_Length()
FROM SpatialShapes WHERE ShapeID = 5
```

Der folgende Code verwendet ST_ToLineString, um den Typ des Shape-Spaltenausdrucks auf ST_LineString zu ändern. ST_Length gibt die Ergebnismenge 7 zurück.

```
SELECT Shape.ST_ToLineString().ST_Length()
FROM SpatialShapes WHERE ShapeID = 5
```

In diesem Fall ist der Wert der Shape-Spalte als ST_LineString-Typ bekannt, daher kann TREAT verwendet werden, um den Ausdruckstyp effizient zu ändern. ST_Length gibt die Ergebnismenge 7 zurück.

```
SELECT TREAT( Shape AS ST_LineString ).ST_Length()
FROM SpatialShapes WHERE ShapeID = 5
```

ST_ToMultiCurve-Methode

Konvertiert die Geometrie in einen Mehrfachkurvenwert.

Syntax

```
geometry-expression.ST_ToMultiCurve()
```

Rückgabe

- **ST_MultiCurve** Wenn *geometry-expression* vom Typ ST_MultiCurve ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* eine Geometriegruppe mit nur Kurven ist, wird ein Mehrfachkurvenobjekt zurückgegeben, das die Elemente von *geometry-expression* enthält. Wenn *geometry-expression* vom Typ ST_Curve ist, wird ein Mehrfachkurvenausdruck zurückgegeben, der ein Element enthält, nämlich *geometry-expression*. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ ST_MultiCurve zurückgegeben. Andernfalls wird eine Ausnahmebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Wenn *geometry-expression* bereits als ST_MultiCurve-Wert bekannt ist, sollte für höhere Effizienz `TREAT(geometry-expression AS ST_MultiCurve)` und nicht die ST_ToMultiCurve-Methode verwendet werden.

Hinweis

Standardmäßig verwendet ST_ToMultiCurve das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToMultiLine-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToGeomColl-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToCurve-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

Beispiel

Das folgende Beispiel gibt den Wert MultiCurve ((0 7, 0 4, 4 4)) zurück.

```
SELECT Shape.ST_ToMultiCurve()  
FROM SpatialShapes WHERE ShapeID = 5
```

ST_ToMultiLine-Methode

Konvertiert die Geometrie in einen Mehrlinienfolgenwert.

Syntax

geometry-expression.**ST_ToMultiLine**()

Rückgabe

- **ST_MultiLineString** Wenn *geometry-expression* vom Typ ST_MultiLineString ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* eine Geometriegruppe mit nur Linien ist, wird ein Mehrfachlinienfolgenobjekt zurückgegeben, das die Elemente von *geometry-expression* enthält. Wenn *geometry-expression* vom Typ ST_Curve ist, wird ein Mehrfachlinienfolgenausdruck zurückgegeben, der ein Element enthält, nämlich *geometry-expression*. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ ST_MultiCurve zurückgegeben. Andernfalls wird eine Ausnahmebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Wenn *geometry-expression* bereits als ST_MultiLineString-Wert bekannt ist, sollte für höhere Effizienz `TREAT(geometry-expression AS ST_MultiLineString)` und nicht die ST_ToMultiLine-Methode verwendet werden.

Hinweis

Standardmäßig verwendet ST_ToMultiLine das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToMultiCurve-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToGeomColl-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToLineString-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

Beispiel

Der folgende Code gibt einen Fehler zurück, weil die Shape-Spalte vom Typ ST_Geometry ist und ST_Geometry die ST_Length-Methode nicht unterstützt.

```
SELECT Shape.ST_Length()
FROM SpatialShapes WHERE ShapeID = 29
```

Der folgende Code verwendet ST_ToMultiLine, um den Typ des Shape-Spaltenausdrucks auf ST_MultiLineString zu ändern. Dieses Beispiel funktioniert auch mit ShapeID 5, wobei der Shape-Wert vom Typ ST_LineString ist. ST_Length gibt die Ergebnismenge 4.236068 zurück.

```
SELECT Shape.ST_ToMultiLine().ST_Length()
FROM SpatialShapes WHERE ShapeID = 29
```

In diesem Fall ist der Wert der Shape-Spalte als ST_MultiLineString-Typ bekannt, daher kann TREAT verwendet werden, um den Ausdruckstyp effizient zu ändern. Dieses Beispiel würde mit ShapeID 5 **nicht**

funktionieren, wo der Shape-Wert vom Typ ST_LineString ist. ST_Length gibt die Ergebnismenge 4.236068 zurück.

```
SELECT TREAT( Shape AS ST_MultiLineString ).ST_Length()  
FROM SpatialShapes WHERE ShapeID = 29
```

ST_ToMultiPoint-Methode

Konvertiert die Geometrie in einen Mehrpunktwert.

Syntax

geometry-expression.ST_ToMultiPoint()

Rückgabe

- **ST_MultiPoint** Wenn *geometry-expression* vom Typ ST_MultiPoint ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* eine Geometriegruppe mit nur Linien ist, wird ein Mehrpunktobjekt zurückgegeben, das die Elemente von *geometry-expression* enthält. Wenn *geometry-expression* vom Typ ST_Point ist, wird ein Mehrfachlinienfolgenausdruck zurückgegeben, der ein Element enthält, nämlich *geometry-expression*. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ ST_MultiPoint zurückgegeben. Andernfalls wird eine Ausnahmebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Wenn *geometry-expression* bereits als ST_MultiPoint-Wert bekannt ist, sollte für höhere Effizienz TREAT(*geometry-expression* AS ST_MultiPoint) und nicht die ST_ToMultiPoint-Methode verwendet werden.

Hinweis

Standardmäßig verwendet ST_ToMultiPoint das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [SQL Anywhere Server - SQL-Referenzhandbuch].

Siehe auch

- [ST_ToGeomColl-Methode für den ST_Geometry-Datentyp](#)
- [ST_ToPoint-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

Beispiel

Das folgende Beispiel gibt den Wert MultiPoint EMPTY zurück.

```
SELECT NEW ST_GeomCollection().ST_ToMultiPoint()
```

ST_ToMultiPolygon-Methode

Konvertiert die Geometrie in einen Multipolygonwert.

Syntax

geometry-expression.ST_ToMultiPolygon()

Rückgabe

- **ST_MultiPolygon** Wenn *geometry-expression* vom Typ ST_MultiPolygon ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* eine Geometriegruppe mit nur Polygonen ist, wird ein Multipolygonobjekt zurückgegeben, das die Elemente von *geometry-expression* enthält. Wenn *geometry-expression* vom Typ ST_Polygon ist, wird ein Multipolygonausdruck zurückgegeben, der ein Element enthält, nämlich *geometry-expression*. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ ST_MultiSurface zurückgegeben. Andernfalls wird eine Ausnahmebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Wenn *geometry-expression* bereits als ST_MultiPolygon-Wert bekannt ist, sollte für höhere Effizienz `TREAT(geometry-expression AS ST_MultiPolygon)` und nicht die ST_ToMultiPolygon-Methode verwendet werden.

Hinweis

Standardmäßig verwendet ST_ToMultiPolygon das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToMultiSurface-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToGeomColl-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToPolygon-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

Beispiel

Das folgende Beispiel gibt den Wert MultiPolygon EMPTY zurück.

```
SELECT NEW ST_GeomCollection().ST_ToMultiPolygon()
```

Der folgende Code gibt einen Fehler zurück, weil die Shape-Spalte vom Typ ST_Geometry ist und ST_Geometry die ST_Area-Methode nicht unterstützt.

```
SELECT Shape.ST_Area()
FROM SpatialShapes WHERE ShapeID = 27
```

Der folgende Code verwendet `ST_ToMultiPolygon`, um den Typ des Shape-Spaltenausdrucks auf `ST_ToMultiPolygon` zu ändern. Dieses Beispiel funktioniert auch mit ShapeID 22, wobei der Shape-Wert vom Typ `ST_LineString` ist. `ST_Area` gibt das Ergebnis 8 zurück.

```
SELECT Shape.ST_ToMultiPolygon().ST_Area()  
FROM SpatialShapes WHERE ShapeID = 27
```

In diesem Fall ist der Wert der Shape-Spalte als `ST_MultiPolygon`-Typ bekannt, daher kann `TREAT` verwendet werden, um den Ausdruckstyp effizient zu ändern. Dieses Beispiel würde mit ShapeID 22 **nicht** funktionieren, wo der Shape-Wert vom Typ `ST_Polygon` ist. `ST_Area` gibt das Ergebnis 8 zurück.

```
SELECT TREAT( Shape AS ST_MultiPolygon ).ST_Area()  
FROM SpatialShapes WHERE ShapeID = 27
```

ST_ToMultiSurface-Methode

Konvertiert die Geometrie in einen Mehrfachoberflächenwert.

Syntax

geometry-expression.**ST_ToMultiSurface**()

Rückgabe

- **ST_MultiSurface** Wenn *geometry-expression* vom Typ `ST_MultiSurface` ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* eine Geometriegruppe mit nur Flächen ist, wird ein Mehrfachoberflächenobjekt zurückgegeben, das die Elemente von *geometry-expression* enthält. Wenn *geometry-expression* vom Typ `ST_Surface` ist, wird ein Mehrfachoberflächenausdruck zurückgegeben, der ein Element enthält, nämlich *geometry-expression*. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ `ST_MultiSurface` zurückgegeben. Andernfalls wird eine Ausnahmebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Wenn *geometry-expression* bereits als `ST_MultiSurface`-Wert bekannt ist, sollte für höhere Effizienz `TREAT(geometry-expression AS ST_MultiSurface)` und nicht die `ST_ToMultiSurface`-Methode verwendet werden.

Hinweis

Standardmäßig verwendet `ST_ToMultiSurface` das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToMultiPolygon-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToGeomColl-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToSurface-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

Beispiel

Das folgende Beispiel gibt den Wert `MultiSurface EMPTY` zurück.

```
SELECT NEW ST_GeomCollection().ST_ToMultiSurface()
```

Das folgende Beispiel gibt den Wert `MultiSurface (((3 3, 8 3, 4 8, 3 3)))` zurück.

```
SELECT Shape.ST_ToMultiSurface()
FROM SpatialShapes WHERE ShapeID = 22
```

ST_ToPoint-Methode

Konvertiert die Geometrie in einen Punkt.

Syntax

geometry-expression.**ST_ToPoint()**

Rückgabe

- **ST_Point** Wenn *geometry-expression* vom Typ `ST_Point` ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* eine Geometriegruppe mit einem einzelnen Element vom Typ `ST_Point` ist, wird dieses Element zurückgegeben. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ `ST_Point` zurückgegeben. Andernfalls wird eine Ausnahmebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Konvertiert die Geometrie in einen Punkt. Die Logik ist äquivalent zu derjenigen, die für **CAST(*geometry-expression* AS `ST_Point`)** verwendet wird.

Wenn *geometry-expression* bereits als `ST_Point`-Wert bekannt ist, sollte für höhere Effizienz **TREAT(*geometry-expression* AS `ST_Point`)** und nicht die `ST_ToPoint`-Methode verwendet werden.

Hinweis

Standardmäßig verwendet `ST_ToPoint` das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToMultiPoint-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

Beispiel

Das folgende Beispiel gibt den Wert `Point (1 2)` zurück.

```
SELECT NEW ST_GeomCollection( NEW ST_Point(1,2) ).ST_ToPoint()
```

ST_ToPolygon-Methode

Konvertiert die Geometrie in ein Polygon.

Syntax

geometry-expression.**ST_ToPolygon()**

Rückgabe

- **ST_Polygon** Wenn *geometry-expression* vom Typ `ST_Polygon` ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* vom Typ `ST_CurvePolygon` ist, wird *geometry-expression*.`ST_CurvePolyToPoly()` zurückgegeben. Wenn *geometry-expression* eine Geometriegruppe mit einem einzelnen Element vom Typ `ST_CurvePolygon` ist, wird dieses Element zurückgegeben. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ `ST_Polygon` zurückgegeben. Andernfalls wird eine Ausnahmebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Konvertiert die Geometrie in ein Polygon. Die Logik ist äquivalent zu derjenigen, die für **CAST(*geometry-expression* AS `ST_Polygon`)** verwendet wird. Wenn *geometry-expression* ein Kurvenpolygon ist, wird es mit `ST_CurvePolyToPoly()` interpoliert.

Wenn *geometry-expression* bereits als `ST_Polygon`-Wert bekannt ist, sollte für höhere Effizienz **TREAT(*geometry-expression* AS `ST_Polygon`)** und nicht die `ST_ToPolygon`-Methode verwendet werden.

Hinweis

Standardmäßig verwendet `ST_ToPolygon` das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToCurvePoly-Methode für den ST_Geometry-Datentyp](#)
- [ST_ToSurface-Methode für den ST_Geometry-Datentyp](#)
- [ST_ToMultiPolygon-Methode für den ST_Geometry-Datentyp](#)
- [ST_CurvePolyToPoly-Methode für den ST_CurvePolygon-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

Beispiel

Das folgende Beispiel gibt den Wert Polygon EMPTY zurück.

```
SELECT NEW ST_GeomCollection().ST_ToPolygon()
```

Der folgende Code gibt einen Fehler zurück, weil die Shape-Spalte vom Typ ST_Geometry ist und ST_Geometry die ST_Area-Methode nicht unterstützt.

```
SELECT Shape.ST_Area()  
FROM SpatialShapes WHERE ShapeID = 22
```

Der folgende Code verwendet ST_ToPolygon, um den Typ des Shape-Spaltenausdrucks auf ST_Polygon zu ändern. ST_Area gibt das Ergebnis 12.5 zurück.

```
SELECT Shape.ST_ToPolygon().ST_Area()  
FROM SpatialShapes WHERE ShapeID = 22
```

In diesem Fall ist der Wert der Shape-Spalte als ST_Polygon-Typ bekannt, daher kann TREAT verwendet werden, um den Ausdruckstyp effizient zu ändern. ST_Area gibt das Ergebnis 12.5 zurück.

```
SELECT TREAT( Shape AS ST_Polygon ).ST_Area()  
FROM SpatialShapes WHERE ShapeID = 22
```

ST_ToSurface-Methode

Konvertiert die Geometrie in eine Fläche.

Syntax

geometry-expression.**ST_ToSurface()**

Rückgabe

- **ST_Surface** Wenn *geometry-expression* vom Typ ST_Surface ist, wird *geometry-expression* zurückgegeben. Wenn *geometry-expression* eine Geometriegruppe mit einem einzelnen Element vom Typ ST_Surface ist, wird dieses Element zurückgegeben. Wenn *geometry-expression* die leere Menge ist, wird eine leere Menge vom Typ ST_Polygon zurückgegeben. Andernfalls wird eine Ausnahmebedingung generiert.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Konvertiert die Geometrie in eine Fläche. Die Logik ist äquivalent zu derjenigen, die für **CAST(*geometry-expression* AS **ST_Surface**)** verwendet wird.

Wenn *geometry-expression* bereits als ST_Surface-Wert bekannt ist, sollte für höhere Effizienz **TREAT(*geometry-expression* AS ST_Surface)** und nicht die ST_ToSurface-Methode verwendet werden.

Hinweis

Standardmäßig verwendet ST_ToSurface das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_ToCurvePoly-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToPolygon-Methode](#) für den ST_Geometry-Datentyp
- [ST_ToMultiSurface-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert Polygon EMPTY zurück.

```
SELECT NEW ST_GeomCollection().ST_ToSurface()
```

ST_Touches-Methode

Testet, ob ein Geometriewert räumlich einen anderen Geometriewert berührt.

Syntax

```
geometry-expression.ST_Touches(geo2)
```

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* den Wert *geo2* berührt, sonst 0. Gibt Null zurück, wenn *geometry-expression* und *geo2* die Dimension 0 haben.

Bemerkungen

Testet, ob ein Geometriewert räumlich einen anderen Geometriewert berührt. Zwei Geometrien berühren einander räumlich, wenn ihre Innenbereiche keine Schnittmengen aufweisen, aber eine oder mehrere Begrenzungspunkte eines Werts eine Schnittmenge mit des Innenbereichs oder der Begrenzung des anderen Werts haben.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

- [ST_Intersects-Methode](#) für den ST_Geometry-Datentyp
- [ST_Boundary-Methode](#) für den ST_Geometry-Datentyp
- [ST_Dimension-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.28

Beispiel

Das folgende Beispiel gibt NULL zurück, da beide Eingaben Punkte sind und keine Begrenzung haben.

```
SELECT NEW ST_Point(1,1).ST_Touches( NEW ST_Point( 1,1 ) )
```

Das folgende Beispiel enthält eine Liste der ShapeIDs der Geometrien, die die "Lightning Bolt"-Form ("Blitz") mit der ShapeID 6 berühren. Die Anweisung in diesem Beispiel gibt das Ergebnis 5 , 16 , 26 zurück. Jede der drei einander berührenden Geometrien hat nur an der Begrenzung eine Schnittmenge mit Lightning Bolt.

```
SELECT List( ShapeID ORDER BY ShapeID )
FROM SpatialShapes
WHERE Shape.ST_Touches( ( SELECT Shape FROM SpatialShapes WHERE ShapeID =
6 ) ) = 1
```

ST_Transform-Methode

Erstellt eine Kopie des in das angegebene räumliche Bezugssystem konvertierten Geometriewerts.

Syntax

geometry-expression.**ST_Transform**(*srid*)

Parameter

Name	Typ	Beschreibung
srid	INT	Die SRID des Ergebnisses.

Rückgabe

- **ST_Geometry** Gibt eine Kopie des in das angegebene räumliche Bezugssystem konvertierten Geometriewerts zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_Transform-Methode wandelt *geometry-expression* aus seinem räumlichen Bezugssystem in das angegebene räumliche Bezugssystem um und verwendet dabei die Transformationsdefinition der beiden räumlichen Bezugssysteme. Die Umwandlung erfolgt mit der Bibliothek PROJ.4

ST_Transform ist zum Verschieben zwischen verschiedenen Koordinatensystemen erforderlich. Sie können beispielsweise ST_Transform verwenden, um eine Geometrie, die Breitengrad und Längengrad verwendet, in eine Geometrie mit SRID 3310 "NAD83 / California Albers" umzuwandeln. Das räumliche Bezugssystem "NAD83 / California Albers" ist eine flache Projektion von Kaliforniendaten, die den Albers-Projektionsalgorithmus und Meter als lineare Maßeinheit verwendet.

Umwandlungen aus einem Längengrad-/Breitengradssystem in ein kartesisches System können bei Polarpunkten problematisch sein. Wenn der Datenbankserver einen Punkt nahe dem Nord- oder Südpol nicht umwandeln kann, verschiebt sich der Breitengrad des Punktes geringfügig (kaum mehr als 1e-10 rad (Bogenmaß) weg vom Pol und entlang desselben Längengrads, damit die Umwandlung erfolgreich verläuft.

Wenn Sie eine Geometrie zwischen zwei räumlichen Bezugssystemen mit demselben Koordinatensystem umwandeln, können Sie anstelle von ST_Transform die ST_SRID-Methode verwenden.

Die praktische Einführung zu räumlichen Bezugssystemen zeigt, wie Sie Daten zwischen räumlichen Bezugssystemen umwandeln. Siehe „[Praktische Einführung: Mit den räumlichen Funktionen experimentieren](#)“ auf Seite 57.

Siehe auch

- [ST_SRID-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.6

Beispiel

Das folgende Beispiel gibt den Wert `Point (184755.86861 -444218.175691)` zurück. Damit erfolgt die Umwandlung eines Punkts in Los Angeles, der mit Längengrad und Breitengrad angegeben ist, in die projizierte, flache SRID 3310 ("NAD83 / California Albers"). In diesem Beispiel wird vorausgesetzt, dass die `st_geometry_predefined_srs`-Funktion von der `sa_install_feature`-Systemprozedur

installiert wurde. Siehe „[sa_install_feature-Systemprozedur](#)“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

```
SELECT NEW ST_Point( -118, 34, 4326 ).ST_Transform( 3310 )
```

ST_Union-Methode

Gibt den Geometriewert zurück, der die Punktmengenvereinigung von zwei Geometrien darstellt.

Syntax

geometry-expression.**ST_Union**(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> vereint werden soll.

Rückgabe

- **ST_Geometry** Gibt den Geometriewert zurück, der die Punktmengenvereinigung von zwei Geometrien darstellt.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *geometry-expression*.

Bemerkungen

Die ST_Union-Methode findet die räumliche Vereinigung von zwei Geometrien. Ein Punkt ist in der Vereinigung enthalten, wenn er in beiden Eingabegeometrien vorhanden ist.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Siehe auch

- [ST_Difference-Methode](#) für den ST_Geometry-Datentyp
- [ST_Intersection-Methode](#) für den ST_Geometry-Datentyp
- [ST_SymDifference-Methode](#) für den ST_Geometry-Datentyp
- [ST_UnionAggr-Methode](#) für Typ ST_Geometry

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.19

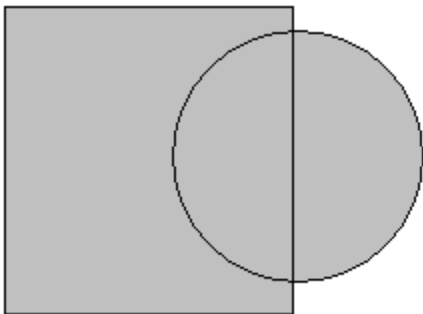
Beispiel

Das folgende Beispiel zeigt die Vereinigung (C) eines Quadrats (A) und eines Kreises (B).

```
SELECT NEW ST_Polygon( 'Polygon( (-1 -0.25, 1 -0.25, 1 2.25, -1 2.25, -1
-0.25) )' ) AS A
, NEW ST_CurvePolygon( 'CurvePolygon( CircularString( 0 1, 1 2, 2 1, 1
```

```
0, 0 1 ) )' ) AS B
, A.ST_Union( B ) AS C
```

Die Vereinigung wird im folgenden Bild schattiert dargestellt. Die Vereinigung besteht aus einer einzelnen Fläche, die alle Punkte in A oder B enthält.



ST_UnionAggr-Methode

Gibt die räumliche Vereinigung aller Geometrien in einer Gruppe zurück

Syntax

```
ST_Geometry::ST_UnionAggr(geometry-column)
```

Parameter

Name	Typ	Beschreibung
geometry-column	ST_Geometry	Die Geometriewerte, aus denen die räumliche Vereinigung generiert wird. Das ist üblicherweise eine Spalte.

Rückgabe

- **ST_Geometry** Gibt eine Geometrie zurück, die die räumliche Vereinigung für alle Geometrien in einer Gruppe darstellt.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist derselbe wie der für den ersten Parameter.

Bemerkungen

Zeilen, in denen das Argument NULL ist, werden nicht eingeschlossen.

Gibt für eine leere Gruppe oder eine Gruppe ohne Nicht-NULL-Werte NULL zurück.

Wenn die Gruppe nur eine Nicht-NULL-Geometrie enthält, wird sie zurückgegeben. Sonst wird die Vereinigung logisch berechnet, indem wiederholt die ST_Union-Methode angewendet wird, um die beiden Geometrien gleichzeitig zu kombinieren. Siehe [ST_Union-Methode für den ST_Geometry-Datentyp auf Seite 239](#).

Siehe auch

- [ST_Union-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert Polygon ((.555555 3, 0 3, 0 1.75, 0 0, 3 0, 3 3, .75 3, 1 4, .555555 3)) zurück.

```
SELECT ST_Geometry::ST_UnionAggr( Shape )
FROM SpatialShapes WHERE ShapeID IN ( 2, 6 )
```

ST_Within-Methode

Testet, ob ein Geometriewert räumlich in einem anderen Geometriewert enthalten ist.

Syntax

geometry-expression.**ST_Within**(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* in *geo2* liegt, sonst 0.

Bemerkungen

Die ST_Within-Methode prüft, ob *geometry-expression* vollständig in *geo2* liegt und mindestens ein innerer Punkt von *geo2* vorhanden ist, der im Inneren von *geometry-expression* liegt.

geometry-expression.ST_Within(*geo2*) ist gleichwertig mit *geo2*.ST_Contains(*geometry-expression*).

Die ST_Within- und ST_CoveredBy-Methoden sind ähnlich. Der Unterschied liegt darin, dass ST_CoveredBy keine Schnittmenge mit Innenpunkten erfordert.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

- [ST_Contains-Methode für den ST_Geometry-Datentyp](#)
- [ST_CoveredBy-Methode für den ST_Geometry-Datentyp](#)
- [ST_Intersects-Methode für den ST_Geometry-Datentyp](#)
- [ST_WithinFilter-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.30

Beispiel

Mit dem folgenden Beispiel wird getestet, ob ein Punkt innerhalb eines Polygons liegt. Der Punkt liegt vollständig im Polygon und der Innenbereich des Punkts (der Punkt selbst) hat eine Schnittmenge mit dem Innenbereich des Polygons, daher gibt das Beispiel 1 zurück.

```
SELECT NEW ST_Point( 1, 1 )
.ST_Within( NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' ) )
```

Mit dem folgenden Beispiel wird getestet, ob eine Linie in einem Polygon liegt. Die Linie liegt vollständig im Polygon, aber das Innere der Linie und das Innere des Polygons haben keine Schnittmenge (die Linie schneidet das Polygon nur an der Begrenzung des Polygons und die Begrenzung ist nicht Teil des Inneren), daher gibt das Beispiel 0 zurück. Wenn ST_CoveredBy anstelle von ST_Within verwendet wurde, gibt ST_CoveredBy 1 zurück.

```
SELECT NEW ST_LineString( 'LineString( 0 0, 1 0 )' )
.ST_Within( NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' ) )
```

Das folgende Beispiel enthält eine Liste der ShapeIDs, bei denen der angegebene Punkt in der Shape-Geometrie liegt. Die Anweisung in diesem Beispiel gibt das Ergebnis 3, 5 zurück. ShapeID 6 ist nicht aufgelistet, weil das Polygon das Shape-Polygon dieser Zeile an der Begrenzung des Polygons schneidet.

```
SELECT LIST( ShapeID ORDER BY ShapeID )
FROM SpatialShapes
WHERE NEW ST_Point( 1, 4 ).ST_Within( Shape ) = 1
```

ST_WithinDistance-Methode

Testet, ob sich zwei Geometrien innerhalb eines angegebenen Abstands voneinander befinden.

Syntax

```
geometry-expression.ST_WithinDistance(geo2,distance[, unit-name])
```

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, dessen Entfernung von <i>geometry-expression</i> gemessen werden soll.
distance	DOUBLE	Der maximale Abstand zwischen den beiden Geometrien.

Name	Typ	Beschreibung
unit-name	VAR-CHAR(128)	Die Maßeinheit, in der der distance-Parameter angegeben wird. Ist standardmäßig die Einheit des räumlichen Bezugssystems. Der unit-name-Parameter muss mit der UNIT_NAME-Spalte einer Zeile in der ST_UNITS_OF_MEASURE-Ansicht übereinstimmen, wo UNIT_TYPE auf 'LINEAR' eingestellt ist.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* und *geo2* innerhalb des angegebenen Abstands voneinander liegen, sonst 0.

Bemerkungen

Die ST_WithinDistance-Methode testet, ob der kleinste Abstand zwischen zwei Geometrien eine angegebene Entfernung nicht überschreitet, wobei eine Toleranz berücksichtigt wird.

Genauer gesagt gibt *d* den kleinsten Abstand zwischen *geometry-expression* und *geo2* an. Der Ausdruck *geometry-expression*.ST_WithinDistance(*geo2*, *distance*[, *unit_name*]) wird mit 1 ausgewertet, wenn *d* <= *distance* ist oder *d* größer ist als *distance*, aber noch innerhalb der Toleranz des zugeordneten räumlichen Bezugssystems liegt.

Für räumliche Bezugssysteme mit planer Erddarstellung wird die Entfernung als kartesische Entfernung in der Ebene gemessen und in den linearen Maßeinheiten des damit verbundenen räumlichen Bezugssystems berechnet. Für räumliche Bezugssysteme mit gewölbter Erddarstellung wird die Entfernung berechnet, indem die Krümmung der Erdoberfläche mithilfe von Ellipsoidparametern in der Definition des räumlichen Bezugssystems berücksichtigt wird.

Hinweis

Wenn *geometry-expression* Kreisbogenfolgen enthält, werden diese zu Linienfolgen interpoliert.

Hinweis

Für räumliche Bezugssysteme mit gewölbter Erddarstellung wird die ST_WithinDistance-Methode nur unterstützt, wenn *geometry-expression* und *geo2* nur Punkte enthalten.

Siehe auch

- [ST_Distance-Methode für den ST_Geometry-Datentyp](#)
- [ST_WithinDistanceFilter-Methode für den ST_Geometry-Datentyp](#)
- [ST_Intersects-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Im folgenden Beispiel wird eine geordnete Ergebnismenge mit einer Zeile für jede Form zurückgegeben, die innerhalb des Abstands 1.4 des Punkts (2,3) liegt.

```
SELECT ShapeID, ROUND( Shape.ST_Distance( NEW ST_Point( 2, 3 ) ), 2 ) AS dist
FROM SpatialShapes
WHERE ShapeID < 17
AND Shape.ST_WithinDistance( NEW ST_Point( 2, 3 ), 1.4 ) = 1
ORDER BY dist
```

Die Abfrage erzeugt die folgende Ergebnismenge:

ShapeID	DIST
2	0.0
3	0.0
5	1.0
6	1.21

Im folgenden Beispiel werden Punkte erstellt, die Halifax, NS und Waterloo, ON, in Kanada darstellen. ST_WithinDistance wird verwendet, um zu zeigen, dass der Abstand zwischen zwei Punkten geringer als 850 Meilen, aber nicht geringer als 840 Meilen ist. In diesem Beispiel wird vorausgesetzt, dass die st_geometry_predefined_uom-Funktion von der sa_install_feature-Systemprozedur installiert wurde. Siehe „sa_install_feature-Systemprozedur“ [SQL Anywhere Server - SQL-Referenzhandbuch].

```
SELECT NEW ST_Point( -63.573566, 44.646244, 4326 )
.ST_WithinDistance( NEW ST_Point( -80.522372, 43.465187, 4326 )
, 850, 'Statute mile' ) within850,
NEW ST_Point( -63.573566, 44.646244, 4326 )
.ST_WithinDistance( NEW ST_Point( -80.522372, 43.465187, 4326 )
, 840, 'Statute mile' ) within840
```

Die Abfrage erzeugt die folgende Ergebnismenge:

within850	within840
1	0

ST_WithinDistanceFilter-Methode

Ein kostengünstiger Test, ob zwei Geometrien in einem angegebenen Abstand voneinander liegen.

Syntax

```
geometry-expression.ST_WithinDistanceFilter(geo2,distance[, unit-name])
```

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, dessen Entfernung von <i>geometry-expression</i> gemessen werden soll.

Name	Typ	Beschreibung
distance	DOUBLE	Der maximale Abstand zwischen den beiden Geometrien.
unit-name	VAR-CHAR(128)	Die Maßeinheit, in der der distance-Parameter angegeben wird. Dies ist standardmäßig die Einheit des räumlichen Bezugssystems. Der unit-name-Parameter muss mit der UNIT_NAME-Spalte einer Zeile in der ST_UNITS_OF_MEASURE-Ansicht übereinstimmen, wo UNIT_TYPE auf 'LINEAR' eingestellt ist.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* und *geo2* innerhalb des angegebenen Abstands voneinander liegen könnten, sonst 0.

Bemerkungen

Die ST_WithinDistanceFilter-Methode bietet einen effizienten Test, um zu ermitteln, ob zwei Geometrien in einem angegebenen Abstand voneinander liegen (wie von der ST_WithinDistance-Methode festgelegt). Gibt 1 zurück, wenn *geometry-expression* innerhalb des angegebenen Abstands von *geo2* liegen könnte, sonst 0.

Dieser Test ist weniger kostenträchtig als ST_WithinDistance, kann aber in einigen Fällen, in denen der kleinste Abstand zwischen den beiden Geometrien größer als der angegebene Abstand ist, 1 zurückgeben. Aus diesem Grund kann diese Methode als Primärfilter nützlich sein, wenn die weitere Verarbeitung den wahren Abstand zwischen den Geometrien ermittelt.

Die Implementierung von ST_WithinDistanceFilter beruht auf Metadaten, die den gespeicherten Geometrien zugeordnet sind. Da sich die verfügbaren Metadaten je nachdem, wie Daten geladen werden oder wo ST_WithinDistanceFilter in einer Abfrage verwendet wird, von Serverversion zu Serverversion verändern können, kann der Ausdruck *geometry-expression*.ST_WithinDistanceFilter(*geo2*, *distance* [, *unit_name*]) unterschiedliche Ergebnisse zurückgeben, wenn *geometry-expression* nicht innerhalb des angegebenen Abstands von *geo2* liegt. Wenn *geometry-expression* innerhalb des angegebenen Abstands von *geo2* liegt, gibt ST_WithinDistanceFilter immer 1 zurück.

Hinweis

Standardmäßig verwendet ST_WithinDistanceFilter das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_Distance-Methode für den ST_Geometry-Datentyp](#)
- [ST_WithinDistance-Methode für den ST_Geometry-Datentyp](#)
- [ST_IntersectsFilter-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Im folgenden Beispiel wird eine geordnete Ergebnismenge mit einer Zeile für jede Form zurückgegeben, die innerhalb des Abstands von 1,4 des Punkts (2,3) liegt. Das Ergebnis enthält eine Form, die nicht innerhalb des angegebenen Abstands liegt.

```
SELECT ShapeID, ROUND( Shape.ST_Distance( NEW ST_Point( 2, 3 ) ), 2 ) AS dist
FROM SpatialShapes
WHERE ShapeID < 17
AND Shape.ST_WithinDistanceFilter( NEW ST_Point( 2, 3 ), 1.4 ) = 1
ORDER BY dist
```

Die Abfrage erzeugt die folgende Ergebnismenge:

ShapeID	DIST
2	0.0
3	0.0
5	1.0
6	1.21
16	1.41

Im folgenden Beispiel werden Punkte erstellt, die Halifax, NS und Waterloo, ON, in Kanada darstellen. ST_WithinDistanceFilter wird verwendet, um zu zeigen, dass der Abstand zwischen zwei Punkten geringer als 850 Meilen sein könnte, aber sicher nicht geringer als 750 Meilen ist. In diesem Beispiel wird vorausgesetzt, dass die st_geometry_predefined_uom-Funktion von der sa_install_feature-Systemprozedur installiert wurde. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

```
SELECT NEW ST_Point( -63.573566, 44.646244, 4326 )
.ST_WithinDistanceFilter( NEW ST_Point( -80.522372, 43.465187, 4326 )
, 850, 'Statute mile' ) within850,
NEW ST_Point( -63.573566, 44.646244, 4326 )
.ST_WithinDistanceFilter( NEW ST_Point( -80.522372, 43.465187, 4326 )
, 750, 'Statute mile' ) within750
```

Die Abfrage erzeugt die folgende Ergebnismenge:

within850	within750
1	0

ST_WithinFilter-Methode

Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie innerhalb einer anderen liegt.

Syntax

geometry-expression.ST_WithinFilter(*geo2*)

Parameter

Name	Typ	Beschreibung
geo2	ST_Geometry	Der andere Geometriewert, der mit <i>geometry-expression</i> verglichen werden soll.

Rückgabe

- **BIT** Gibt 1 zurück, wenn *geometry-expression* innerhalb von *geo2* liegen könnte, sonst 0.

Bemerkungen

Die ST_WithinFilter-Methode bietet einen effizienten Test, um zu ermitteln, ob eine Geometrie in einer anderen liegen könnte. Gibt 1 zurück, wenn *geometry-expression* innerhalb von *geo2* liegen könnte, sonst 0.

Dieser Test ist weniger kostenträchtig als ST_Within, kann aber in manchen Fällen 1 zurückgeben, in denen *geometry-expression* räumlich nicht innerhalb von *geo2* liegt.

Aus diesem Grund kann diese Methode als Primärfilter nützlich sein, wenn die weitere Verarbeitung erkennen lässt, ob Geometrien auf gewünschte Art und Weise interagieren.

Die Implementierung von ST_WithinFilter beruht auf Metadaten, die den gespeicherten Geometrien zugeordnet sind. Da sich die verfügbaren Metadaten je nachdem, wie Daten geladen werden oder wo ST_WithinFilter in einer Abfrage verwendet wird, von Serverversion zu Serverversion verändern können, kann der Ausdruck *geometry-expression*.ST_WithinFilter(*geo2*) unterschiedliche Ergebnisse zurückgeben, wenn *geometry-expression* nicht innerhalb von *geo2* liegt. Wenn *geometry-expression* innerhalb von *geo2* liegt, gibt ST_WithinFilter immer 1 zurück.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

- [ST_Within-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

ST_XMax-Methode

Ruft den maximalen X-Koordinatenwert einer Geometrie ab.

Syntax

geometry-expression.ST_XMax()

Rückgabe

- **DOUBLE** Gibt den maximalen X-Koordinatenwert von *geometry-expression* zurück.

Bemerkungen

Gibt den maximalen X-Koordinatenwert von *geometry-expression* zurück. Dies wird durch einen Vergleich des X-Attributs aller Punkte in der Geometrie berechnet.

Bei geografischen räumlichen Bezugssystemen entspricht der zurückgegebene Wert der ersten Koordinate in der Achsenreihenfolge. Wenn die Achsenreihenfolge Breitengrad/Längengrad/a/m ist, entspricht das Minimum der westlichen Begrenzung von *geometry-expression*, wie von ST_LongWest zurückgegeben, und das Maximum entspricht der östlichen Begrenzung von *geometry-expression*, wie von ST_LongEast zurückgegeben. Im Modell mit gewölbter Erddarstellung bedeutet dies: Wenn *geometry-expression* die Datumsgrenze überschreitet, ist das Minimum größer als das Maximum. Wenn die Achsenreihenfolge Längengrad/Breitengrad/z/m ist, entspricht das Minimum dem südlichsten Punkt von *geometry-expression*, wie von ST_LatSouth zurückgegeben, und das Maximum dem nördlichsten Punkt von *geometry-expression*, wie von ST_LatNorth zurückgegeben.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_XMax das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_XMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_ZMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_ZMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_MMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_MMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_LongEast-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert 5 zurück.

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_XMax()
```

ST_XMin-Methode

Ruft den minimalen X-Koordinatenwert einer Geometrie ab.

Syntax

geometry-expression.ST_XMin()

Rückgabe

- **DOUBLE** Gibt den minimalen X-Koordinatenwert von *geometry-expression* zurück.

Bemerkungen

Gibt den minimalen X-Koordinatenwert von *geometry-expression* zurück. Dies wird durch einen Vergleich des X-Attributs aller Punkte in der Geometrie berechnet.

Für geografische räumliche Bezugssysteme entspricht der zurückgegebene Wert der ersten Koordinate in der Achsenreihenfolge. Wenn die Achsenreihenfolge Breitengrad/Längengrad/a/m ist, entspricht das Minimum der westlichen Begrenzung von *geometry-expression*, wie von ST_LongWest zurückgegeben, und das Maximum entspricht der östlichen Begrenzung von *geometry-expression*, wie von ST_LongEast zurückgegeben. Im Modell mit gewölbter Erddarstellung bedeutet dies: Wenn *geometry-expression* die Datumsgrenze überschreitet, ist das Minimum größer als das Maximum. Wenn die Achsenreihenfolge Längengrad/Breitengrad/z/m ist, entspricht das Minimum dem südlichsten Punkt von *geometry-expression*, wie von ST_LatSouth zurückgegeben, und das Maximum dem nördlichsten Punkt von *geometry-expression*, wie von ST_LatNorth zurückgegeben.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_XMin das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_XMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_ZMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_ZMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_MMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_MMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_LongWest-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert 1 zurück.

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_XMin()
```

ST_YMax-Methode

Ruft den maximalen Y-Koordinatenwert einer Geometrie ab.

Syntax

geometry-expression.ST_YMax()

Rückgabe

- **DOUBLE** Gibt den maximalen Y-Koordinatenwert von *geometry-expression* zurück.

Bemerkungen

Gibt den maximalen Y-Koordinatenwert von *geometry-expression* zurück. Dies wird durch einen Vergleich des Y-Attributs aller Punkte in der Geometrie berechnet.

Für geografische räumliche Bezugssysteme entspricht der zurückgegebene Wert der ersten Koordinate in der Achsenreihenfolge. Wenn die Achsenreihenfolge Längengrad/Breitengrad/z/m ist, entspricht das Minimum dem südlichsten Punkt von *geometry-expression*, wie von ST_LatSouth zurückgegeben, und das Maximum dem nördlichsten Punkt von *geometry-expression*, wie von ST_LatNorth zurückgegeben. Wenn die Achsenreihenfolge Breitengrad/Längengrad/a/m ist, entspricht das Minimum der westlichen Begrenzung von *geometry-expression*, wie von ST_LongWest zurückgegeben, und das Maximum entspricht der östlichen Begrenzung von *geometry-expression*, wie von ST_LongEast zurückgegeben. Im Modell mit gewölbter Erddarstellung bedeutet dies: Wenn *geometry-expression* die Datumsgrenze überschreitet, ist das Minimum größer als das Maximum.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_YMax das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_XMin-Methode für den ST_Geometry-Datentyp](#)
- [ST_XMax-Methode für den ST_Geometry-Datentyp](#)
- [ST_YMin-Methode für den ST_Geometry-Datentyp](#)
- [ST_ZMin-Methode für den ST_Geometry-Datentyp](#)
- [ST_ZMax-Methode für den ST_Geometry-Datentyp](#)
- [ST_MMin-Methode für den ST_Geometry-Datentyp](#)
- [ST_MMax-Methode für den ST_Geometry-Datentyp](#)
- [ST_LatNorth-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert 6 zurück.

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_YMax()
```

ST_YMin-Methode

Ruft den minimalen Y-Koordinatenwert einer Geometrie ab.

Syntax

geometry-expression.**ST_YMin**()

Rückgabe

- **DOUBLE** Gibt den minimalen Y-Koordinatenwert von *geometry-expression* zurück.

Bemerkungen

Gibt den minimalen Y-Koordinatenwert von *geometry-expression* zurück. Dies wird durch einen Vergleich des Y-Attributs aller Punkte in der Geometrie berechnet.

Für geografische räumliche Bezugssysteme entspricht der zurückgegebene Wert der ersten Koordinate in der Achsenreihenfolge. Wenn die Achsenreihenfolge Längengrad/Breitengrad/z/m ist, entspricht das Minimum dem südlichsten Punkt von *geometry-expression*, wie von ST_LatSouth zurückgegeben, und das Maximum dem nördlichsten Punkt von *geometry-expression*, wie von ST_LatNorth zurückgegeben. Wenn die Achsenreihenfolge Breitengrad/Längengrad/a/m ist, entspricht das Minimum der westlichen Begrenzung von *geometry-expression*, wie von ST_LongWest zurückgegeben, und das Maximum entspricht der östlichen Begrenzung von *geometry-expression*, wie von ST_LongEast zurückgegeben. Im Modell mit gewölbter Erddarstellung bedeutet dies: Wenn *geometry-expression* die Datumsgrenze überschreitet, ist das Minimum größer als das Maximum.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_YMin das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_XMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_XMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_ZMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_ZMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_MMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_MMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_LatSouth-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert 2 zurück.

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_YMin()
```

ST_ZMax-Methode

Ruft den maximalen Z-Koordinatenwert einer Geometrie ab.

Syntax

geometry-expression.**ST_ZMax**()

Rückgabe

- **DOUBLE** Gibt den maximalen Z-Koordinatenwert von *geometry-expression* zurück.

Bemerkungen

Gibt den maximalen Z-Koordinatenwert von *geometry-expression* zurück. Dies wird durch einen Vergleich des Z-Attributs aller Punkte in der Geometrie berechnet.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_ZMax das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_XMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_XMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_ZMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_MMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_MMax-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert 7 zurück.

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_ZMax()
```

ST_ZMin-Methode

Ruft den minimalen Z-Koordinatenwert einer Geometrie ab.

Syntax

geometry-expression.**ST_ZMin**()

Rückgabe

- **DOUBLE** Gibt den minimalen Z-Koordinatenwert von *geometry-expression* zurück.

Bemerkungen

Gibt den minimalen Z-Koordinatenwert von *geometry-expression* zurück. Dies wird durch einen Vergleich des Z-Attributs aller Punkte in der Geometrie berechnet.

Hinweis

Wenn *geometry-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_ZMin das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_XMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_XMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_YMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_ZMax-Methode](#) für den ST_Geometry-Datentyp
- [ST_MMin-Methode](#) für den ST_Geometry-Datentyp
- [ST_MMax-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert 3 zurück.

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_ZMin()
```

ST_LineString-Datentyp

Der ST_LineString-Datentyp ist ein untergeordneter Datentyp von ST_Curve, der gerade Liniensegmente zwischen Kontrollpunkten verwendet.

Direkt übergeordneter Typ

- „ST_Curve-Datentyp“

Konstruktor

- „ST_LineString-Konstruktor“

Methoden

- Methoden von ST_LineString:

ST_LineStringAggr	ST_NumPoints	ST_PointN
-----------------------------------	------------------------------	---------------------------

- Alle Methoden von „ST_Curve-Datentyp“ können auch für einen ST_LineString-Datentyp aufgerufen werden:

ST_CurveToLine	ST_EndPoint	ST_IsClosed	ST_IsRing
--------------------------------	-----------------------------	-----------------------------	---------------------------

ST_Length	ST_StartPoint		
-----------	---------------	--	--

- Alle Methoden von „ST_Geometry-Datentyp“ können auch für einen ST_LineString-Datentyp aufgerufen werden:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAggr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr
ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType
ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth
ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBaseType	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine
ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance

ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Bemerkungen

Der ST_LineString-Datentyp ist ein untergeordneter Datentyp von ST_Curve, der gerade Liniensegmente zwischen Kontrollpunkten verwendet. Jedes aufeinander folgende Paar von Punkten wird mit einem geraden Liniensegment verknüpft.

Eine Linie ist ein ST_LineString-Wert mit genau zwei Punkten. Ein linearer Ring ist ein ST_LineString-Wert, der geschlossen und einfach ist.

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 7.2

ST_LineString-Konstruktor

Konstruiert eine Linienfolge.

Überladungsliste

Name	Beschreibung
„ST_LineString()-Konstruktor“	Konstruiert eine Linienfolge, die eine leere Menge darstellt.
„ST_LineString(LONG VARCHAR[, INT])-Konstruktor“	Konstruiert eine Linienfolge aus einer Textdarstellung.
„ST_LineString(LONG BINARY[, INT])-Konstruktor“	Konstruiert eine Linienfolge aus einem Well-Known-Binary-Wert (WKB).
„ST_LineString(ST_Point,ST_Point,...)-Konstruktor“	Konstruiert einen Linienfolgenwert aus einer Liste von Punkten in einem angegebenen räumlichen Bezugssystem.

ST_LineString()-Konstruktor

Konstruiert eine Linienfolge, die eine leere Menge darstellt.

Syntax

NEW ST_LineString()

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Standardfunktion

Beispiel

Der folgende Code gibt 1 zurück und zeigt damit an, dass der Wert leer ist.

```
SELECT NEW ST_LineString().ST_IsEmpty()
```

ST_LineString(LONG VARCHAR[, INT])-Konstruktor

Konstruiert eine Linienfolge aus einer Textdarstellung.

Syntax

```
NEW ST_LineString(text-representation[, srid])
```

Parameter

Name	Typ	Beschreibung
text-representation	LONG VARCHAR	Eine Zeichenfolge, die die Textdarstellung einer Linienfolge enthält. Die Eingabe kann in jedem unterstützten Texteingabeformat erfolgen, einschließlich Well-Known-Text (WKT) oder Extended-Well-Known-Text (EWKT).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert eine Linienfolge aus einer Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.2.2

Beispiel

Die folgende Anweisung gibt LineString (0 0, 5 10) zurück.

```
SELECT NEW ST_LineString('LineString (0 0, 5 10)')
```

ST_LineString(LONG BINARY[, INT])-Konstruktor

Konstruiert eine Linienfolge aus einem Well-Known-Binary-Wert (WKB).

Syntax

```
NEW ST_LineString(wkb[, srid])
```


angegebenen Punkte dürfen nicht leer sein und müssen dieselbe Antwort für Is3D und IsMeasured aufweisen. Die Linienfolge ist 3D, wenn alle Punkte 3D sind, und die Linienfolge wird gemessen, wenn alle Punkte gemessen werden.

Hinweis

Standardmäßig verwendet ST_LineString das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert LineString (0 0, 1 1) zurück.

```
SELECT NEW ST_LineString( NEW ST_Point( 0, 0 ), NEW ST_Point( 1, 1 ) )
```

Das folgende Beispiel gibt den Wert LineString (0 0, 1 1, 2 0) zurück.

```
SELECT NEW ST_LineString( NEW ST_Point( 0, 0 ), NEW ST_Point( 1, 1 ), NEW  
ST_Point(2,0) )
```

ST_LineStringAggr-Methode

Gibt eine Linienfolge zurück, die aus den sortierten Punkten in einer Gruppe gebildet wird.

Syntax

```
ST_LineString::ST_LineStringAggr(point[ ORDER BY order-by-expression [ ASC | DESC ], ... ] )
```

Parameter

Name	Typ	Beschreibung
point	ST_Point	Die Punkte zur Generierung der Linienfolge. Das ist üblicherweise eine Spalte.

Rückgabe

- **ST_LineString** Gibt eine Linienfolge zurück, die aus den Punkten in einer Gruppe gebildet wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist derselbe wie der für den ersten Parameter.

Bemerkungen

Die ST_LineStringAggr-Aggregatmethode kann verwendet werden, um eine Linienfolge aus einer Gruppe von geordneten Punkten zu erstellen. Alle Geometriespalten, die kombiniert werden sollen, müssen die gleiche SRID aufweisen. Punkte, die kombiniert werden sollen, dürfen nicht leer sein und müssen dieselbe Koordinatendimension aufweisen.

Zeilen, in denen der Punkt NULL ist, werden nicht eingeschlossen.

Gibt für eine leere Gruppe oder eine Gruppe ohne Nicht-NULL-Werte NULL zurück.

Die sich daraus ergebende Linienfolge hat dieselbe Koordinatendimension wie jeder Punkt.

Hinweis

Die ORDER BY-Klausel sollte angegeben werden, um die Reihenfolge von Punkten innerhalb der Linienfolge zu steuern. Wenn sie nicht angegeben wird, ist die Reihenfolge der Punkte in der Linienfolge je nach Zugriffsplan, der vom Abfrageoptimierer ausgewählt wird, verschieden.

Hinweis

Standardmäßig verwendet ST_LineStringAggr das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert LineString (0 0, 2 0, 1 1) zurück.

```
BEGIN
  DECLARE LOCAL TEMPORARY TABLE t_points( pk INT PRIMARY KEY,
                                             p ST_Point );
  INSERT INTO t_points VALUES( 1, 'Point( 0 0 )' );
  INSERT INTO t_points VALUES( 2, 'Point( 2 0 )' );
  INSERT INTO t_points VALUES( 3, 'Point( 1 1 )' );

  SELECT ST_LineString::ST_LineStringAggr( p ORDER BY pk )
  FROM t_points;
END
```

ST_NumPoints-Methode

Gibt die Anzahl der Punkte zurück, die die Linienfolge definieren.

Hinweis

Standardmäßig verwendet ST_NumPoints das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

linestring-expression.ST_NumPoints()

Rückgabe

- **INT** Gibt NULL zurück, wenn der Linienfolgenwert leer ist, sonst die Anzahl der Punkte im Wert.

Siehe auch

- [ST_PointN-Methode](#) für den ST_LineString-Datentyp
- [ST_NumPoints-Methode](#) für den ST_CircularString-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.2.4

Beispiel

Das folgende Beispiel gibt den Wert 3 zurück.

```
SELECT TREAT( Shape AS ST_LineString ).ST_NumPoints()
FROM SpatialShapes WHERE ShapeID = 5
```

ST_PointN-Methode

Gibt den *n*-ten Punkt in der Linienfolge zurück.

Hinweis

Standardmäßig verwendet ST_PointN das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

linestring-expression.**ST_PointN**(*n*)

Parameter

Name	Typ	Beschreibung
n	INT	Die Position des zurückzugebenden Elements, von 1 bis <i>linestring-expression</i> .ST_NumPoints().

Rückgabe

- **ST_Point** Wenn der Wert von *linestring-expression* eine leere Menge ist, wird NULL zurückgegeben. Wenn die angegebene Position *n* kleiner als 1 oder größer als die Anzahl der Punkte ist, wird NULL zurückgegeben. Andernfalls wird der ST_Point-Wert an der Position *n* zurückgegeben.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *linestring-expression*.

Siehe auch

- [ST_NumPoints-Methode für den ST_LineString-Datentyp](#)
- [ST_PointN-Methode für den ST_CircularString-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.2.5

Beispiel

Das folgende Beispiel gibt den Wert `Point (0 4)` zurück.

```
SELECT TREAT( Shape AS ST_LineString ).ST_PointN( 2 )
FROM SpatialShapes WHERE ShapeID = 5
```

Das folgende Beispiel gibt eine Zeile für jeden Punkt in der Geometrie zurück.

```
BEGIN
  DECLARE geom ST_LineString;
  SET geom = NEW ST_LineString( 'LineString( 0 0, 1 0 )' );
  SELECT row_num, geom.ST_PointN( row_num )
    FROM sa_rowgenerator( 1, geom.ST_NumPoints() )
   ORDER BY row_num;
END
```

Die Abfrage erzeugt die folgende Ergebnismenge:

row_num	geom.ST_PointN(row_num)
1	Point (0 0)
2	Point (1 0)

ST_MultiCurve-Datentyp

Eine `ST_MultiCurve` ist eine Sammlung von null oder mehr `ST_Curve`-Werten, wobei sich alle Kurven im räumlichen Bezugssystem befinden.

Direkt übergeordneter Typ

- [„ST_GeomCollection-Datentyp“](#)

Direkt untergeordnete Typen

- [„ST_MultiLineString-Datentyp“](#)

Konstruktor

- [„ST_MultiCurve-Konstruktor“](#)

Methoden

- Methoden von `ST_MultiCurve`:

ST_IsClosed	ST_Length	ST_MultiCurveAggr
-------------	-----------	-------------------

- Alle Methoden von „ST_GeomCollection-Datentyp“ können auch für einen ST_MultiCurve-Datentyp aufgerufen werden:

ST_GeomCollectionAggr	ST_GeometryN	ST_NumGeometries
-----------------------	--------------	------------------

- Alle Methoden von „ST_Geometry-Datentyp“ können auch für einen ST_MultiCurve-Datentyp aufgerufen werden:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAggr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr
ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType
ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth
ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBaseType	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine

ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.3

ST_MultiCurve-Konstruktor

Konstruiert eine Mehrfachkurve.

Überladungsliste

Name	Beschreibung
„ ST_MultiCurve() -Konstruktor“	Konstruiert eine Mehrfachkurve, die die leere Menge darstellt.
„ ST_MultiCurve(LONG VARCHAR[, INT]) -Konstruktor“	Konstruiert eine Mehrfachkurve aus einer Textdarstellung.
„ ST_MultiCurve(LONG BINARY[, INT]) -Konstruktor“	Konstruiert eine Mehrfachkurve aus Well-Known-Binary-Werten (WKB).
„ ST_MultiCurve(ST_Curve,...) -Konstruktor“	Konstruiert eine Mehrfachkurve aus einer Liste von Kurvenwerten.

ST_MultiCurve()-Konstruktor

Konstruiert eine Mehrfachkurve, die die leere Menge darstellt.

Syntax

NEW ST_MultiCurve()

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Standardfunktion

Beispiel

Der folgende Code gibt 1 zurück und zeigt damit an, dass der Wert leer ist.

```
SELECT NEW ST_MultiCurve().ST_IsEmpty()
```

ST_MultiCurve(LONG VARCHAR[, INT])-Konstruktor

Konstruiert eine Mehrfachkurve aus einer Textdarstellung.

Syntax

```
NEW ST_MultiCurve(text-representation[, srid])
```

Parameter

Name	Typ	Beschreibung
text-representation	LONG VAR- CHAR	Eine Zeichenfolge, die die Textdarstellung einer Mehrfachkurve enthält. Die Eingabe kann in jedem unterstützten Texteingabeformat erfolgen, einschließlich Well-Known-Text (WKT) oder Extended-Well-Known-Text (EWKT).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert eine Mehrfachkurve aus einer Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 9.3.2

Beispiel

Die folgende Anweisung gibt MultiCurve ((10 10, 12 12), CircularString (5 10, 10 12, 15 10)) zurück.

```
SELECT NEW ST_MultiCurve('MultiCurve ((10 10, 12 12), CircularString (5 10, 10 12, 15 10))')
```

ST_MultiCurve(LONG BINARY[, INT])-Konstruktor

Konstruiert eine Mehrfachkurve aus Well-Known-Binary-Werten (WKB).

Syntax

```
NEW ST_MultiCurve(wkb[, srid])
```

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Eine Zeichenfolge mit der binären Darstellung einer Mehrfachkurve. Die Eingabe kann in jedem unterstützten binären Eingabeformat erfolgen, einschließlich Well-Known-Binary-Datentypen (WKB) oder Extended-Well-Known-Binary-Datentypen (EWKB).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert eine Mehrfachkurve aus einer binären Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.3.2

Beispiel

Die folgende Anweisung gibt MultiCurve (CircularString (5 10, 10 12, 15 10)) zurück.

```
SELECT NEW
ST_MultiCurve(0x010b00000001000000010800000003000000000000000001440000000000
000244000000000000002440000000000002840000000000002e400000000000002440)
```

ST_MultiCurve(ST_Curve,...)-Konstruktor

Konstruiert eine Mehrfachkurve aus einer Liste von Kurvenwerten.

Syntax

NEW ST_MultiCurve(*curve1*[,*curve2*,...,*curveN*])

Parameter

Name	Typ	Beschreibung
curve1	ST_Curve	Der erste Kurvenwert der Mehrfachkurve.
curve2,...,curveN	ST_Curve	Zusätzliche Kurvenwerte der Mehrfachkurve.

Bemerkungen

Konstruiert eine Mehrfachkurve aus einer Liste von Kurvenwerten. Alle angegebenen Kurvenwerte müssen dieselbe SRID haben und die Mehrfachkurve wird mit dieser gemeinsamen SRID konstruiert.

Alle angegebenen Kurvenwerte müssen die gleichen Antworten für Is3D und IsMeasured aufweisen. Die Mehrfachkurve ist 3D, wenn alle Kurvenwerte 3D sind, und die Mehrfachkurve wird gemessen, wenn alle Kurvenwerte gemessen werden.

Hinweis

Standardmäßig verwendet ST_MultiCurve das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert MultiCurve ((0 0, 1 1)) zurück.

```
SELECT NEW ST_MultiCurve( NEW ST_LineString('LineString (0 0, 1 1)' ) )
```

Das folgende Beispiel gibt den Wert MultiCurve ((0 0, 1 1), CircularString (0 0, 1 1, 2 0)) zurück.

```
SELECT NEW ST_MultiCurve(
    NEW ST_LineString('LineString (0 0, 1 1)' ),
    NEW ST_CircularString( 'CircularString( 0 0, 1 1, 2 0)' ) )
```

ST_IsClosed-Methode

Testet, ob der ST_MultiCurve-Wert geschlossen ist. Eine Kurve ist geschlossen, wenn der Start- und der Endpunkt zusammenfallen. Eine Mehrfachkurve ist geschlossen, wenn sie nicht leer ist und eine leere Begrenzung hat.

Hinweis

Standardmäßig verwendet ST_IsClosed das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

multicurve-expression.**ST_IsClosed()**

Rückgabe

- **BIT** Gibt 1 zurück, wenn die Mehrfachkurve geschlossen ist, sonst 0.

Siehe auch

- [ST_IsClosed-Methode für den ST_Curve-Datentyp](#)
- [ST_Boundary-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.3.3

Beispiel

Der folgende Code gibt das Ergebnis 0 zurück, weil die Begrenzung der Mehrfachkurve zwei Punkte enthält.

```
SELECT NEW ST_MultiCurve( 'MultiCurve((0 0, 1 1))' ).ST_IsClosed()
```

Der folgende Code gibt alle Zeilen in multicurve_table zurück, die geschlossene Geometrien enthalten. In diesem Beispiel wird davon ausgegangen, dass die Geometriespalte vom Typ ST_MultiCurve oder ST_MultiLineString ist.

```
SELECT * FROM multicurve_table WHERE geometry.ST_IsClosed() = 1
```

ST_Length-Methode

Gibt die Längenmessung des ST_MultiCurve-Werts zurück. Das Ergebnis wird mit den Einheiten gemessen, die durch den Parameter festgelegt sind.

Syntax

```
multicurve-expression.ST_Length([ unit-name])
```

Parameter

Name	Typ	Beschreibung
unit-name	VAR-CHAR(128)	Die Einheiten, in denen die Länge berechnet werden soll. Ist standardmäßig die Einheit des räumlichen Bezugssystems. Der unit-name-Parameter muss mit der UNIT_NAME-Spalte einer Zeile in der ST_UNITS_OF_MEASURE-Ansicht übereinstimmen, wo UNIT_TYPE auf 'LINEAR' eingestellt ist.

Rückgabe

- **DOUBLE** Gibt die Längenmessung des ST_MultiCurve-Werts zurück.

Bemerkungen

Die ST_Length-Methode gibt die Länge einer Mehrfachkurve in den Einheiten zurück, die durch den *unit_name*-Parameter festgelegt sind. Die Länge einer Mehrfachkurve ist die Summe der Längen der enthaltenen Kurven. Wenn die Kurve leer ist, wird NULL zurückgegeben.

Wenn die Kurve Z-Werte enthält, werden diese beim Berechnen der Länge der Geometrie nicht in Betracht gezogen.

Hinweis

Wenn *multicurve-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_LENGTH das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_Length-Methode für den ST_Curve-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.3.4

Beispiel

Mit dem folgenden Beispiel wird eine Mehrfachkurve erstellt und ST_Length für die Suche nach der Länge der Geometrie verwendet. Der Wert PI+1 wird zurückgegeben.

```
SELECT NEW ST_MultiCurve(  
    NEW ST_LineString('LineString (0 0, 1 0)' ),  
    NEW ST_CircularString( 'CircularString( 0 0, 1 1, 2 0)' ) )  
.ST_Length()
```

Das folgende Beispiel gibt den Namen und die Länge aller Straßen zurück, die länger als 100 Meilen sind. In diesem Beispiel wird davon ausgegangen, dass die Tabelle road vorhanden ist, dass die Geometriespalte vom Typ ST_MultiCurve oder ST_MultiLineString ist und die sa_install_feature-Systemprozedur verwendet wurde, um st_geometry_predefined_uom zu laden.

```
SELECT name, geometry.ST_Length( 'Statute Mile' ) len  
FROM roads WHERE len > 100
```

ST_MultiCurveAggr-Methode

Gibt eine Mehrfachkurve mit allen Kurven in einer Gruppe zurück.

Syntax

```
ST_MultiCurve::ST_MultiCurveAggr(geometry-column[ ORDER BY order-by-expression [ ASC |  
DESC ], ... ])
```

Parameter

Name	Typ	Beschreibung
geometry-column	ST_Curve	Die Geometriewerte zum Generieren der Sammlung. Das ist üblicherweise eine Spalte.

Rückgabe

- **ST_MultiCurve** Gibt eine Mehrfachkurve zurück, die alle Geometrien in einer Gruppe enthält.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist derselbe wie der für den ersten Parameter.

Bemerkungen

Die ST_MultiCurveAggr-Aggregatfunktion kann verwendet werden, um eine Gruppe von Kurven in einer einzigen Sammlung zu kombinieren. Alle Geometrien, die kombiniert werden sollen, müssen die gleiche SRID und dieselbe Koordinatendimension haben.

Zeilen, in denen das Argument NULL ist, werden nicht eingeschlossen.

Gibt für eine leere Gruppe oder eine Gruppe ohne Nicht-NULL-Werte NULL zurück.

Das sich daraus ergebende ST_MultiCurve-Element hat dieselbe Koordinatendimension wie jede der Kurven.

Die optionale ORDER BY-Klausel kann verwendet werden, um die Elemente in einer bestimmten Reihenfolge anzuordnen, damit ST_GeometryN sie in der gewünschten Reihenfolge zurückgibt. Wenn diese Reihenfolge nicht von Bedeutung ist, sollte aus Effizienzgründen keine Sortierfolge angegeben werden. In diesem Fall hängt die Reihenfolge der Elemente vom Zugriffsplan ab, der vom Abfrageoptimierer ausgewählt wird.

ST_MultiCurveAggr ist effizienter als ST_UnionAggr, aber ST_MultiCurveAggr kann eine Gruppe mit mehrfach vorhandenen oder sich überschneidenden Kurven zurückgeben, wenn sie in der Kurvengruppe vorhanden sind. ST_UnionAggr verarbeitet mehrfach vorhandene und überlappende Geometrien.

Hinweis

Standardmäßig verwendet ST_MultiCurveAggr das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_UnionAggr-Methode für Typ ST_Geometry](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt einen einzelnen Wert zurück, der alle Geometrien vom Typ ST_Curve aus der SpatialShapes-Tabelle in einer einzigen Sammlung vom Typ ST_MultiCurve kombiniert. Wenn die Shape-Spalte vom Typ ST_Curve ist, sind die TREAT-Funktion und die WHERE-Klausel nicht erforderlich.

```
SELECT ST_MultiCurve::ST_MultiCurveAggr( TREAT( Shape AS ST_Curve ) )  
FROM SpatialShapes WHERE Shape IS OF( ST_Curve )
```

ST_MultiLineString-Datentyp

ST_MultiLineString ist eine Sammlung von null oder mehr ST_LineString-Werten, wobei sich alle Linienfolgen sich im räumlichen Bezugssystem befinden.

Direkt übergeordneter Typ

- „ST_MultiCurve-Datentyp“

Konstruktor

- „ST_MultiLineString-Konstruktor“

Methoden

- Methoden von ST_MultiLineString:

ST_MultiLineStringAggr

- Alle Methoden von „ST_MultiCurve-Datentyp“ können auch für einen ST_MultiLineString-Datentyp aufgerufen werden:

ST_IsClosed	ST_Length	ST_MultiCurveAggr
-------------	-----------	-------------------

- Alle Methoden von „ST_GeomCollection-Datentyp“ können auch für einen ST_MultiLineString-Datentyp aufgerufen werden:

ST_GeomCollectionAggr	ST_GeometryN	ST_NumGeometries
-----------------------	--------------	------------------

- Alle Methoden von „ST_Geometry-Datentyp“ können auch für einen ST_MultiLineString-Datentyp aufgerufen werden:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAggr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr
ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType

ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth
ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBaseType	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine
ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.4

ST_MultiLineString-Konstruktor

Konstruiert eine Mehrfachlinienfolge.

Überladungsliste

Name	Beschreibung
„ST_MultiLineString()-Konstruktor“	Konstruiert eine Mehrfachlinienfolge, die die leere Menge darstellt.

Name	Beschreibung
„ST_MultiLineString(LONG VARCHAR[, INT])-Konstruktor“	Konstruiert eine Mehrfachlinienfolge aus einer Textdarstellung.
„ST_MultiLineString(LONG BINARY[, INT])-Konstruktor“	Konstruiert eine Mehrfachlinienfolge aus Well-Known-Binary-Werten (WKB).
„ST_MultiLineString(ST_LineString,...)-Konstruktor“	Konstruiert eine Mehrfachlinienfolge aus einer Liste von Linienfolgenwerten.

ST_MultiLineString()-Konstruktor

Konstruiert eine Mehrfachlinienfolge, die die leere Menge darstellt.

Syntax

NEW ST_MultiLineString()

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Standardfunktion

Beispiel

Der folgende Code gibt 1 zurück und zeigt damit an, dass der Wert leer ist.

```
SELECT NEW ST_MultiLineString().ST_IsEmpty()
```

ST_MultiLineString(LONG VARCHAR[, INT])-Konstruktor

Konstruiert eine Mehrfachlinienfolge aus einer Textdarstellung.

Syntax

NEW ST_MultiLineString(*text-representation*[, *srid*])

Parameter

Name	Typ	Beschreibung
text-representation	LONG VARCHAR	Eine Zeichenfolge, die die Textdarstellung einer Mehrfachlinienfolge enthält. Die Eingabe kann in jedem unterstützten Texteingabeformat erfolgen, einschließlich Well-Known-Text (WKT) oder Extended-Well-Known-Text (EWKT).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

ST_MultiLineString(ST_LineString,...)-Konstruktor

Konstruiert eine Mehrfachlinienfolge aus einer Liste von Linienfolgenwerten.

Syntax

NEW ST_MultiLineString(*linestring1*[,*linestring2*,...,*linestringN*])

Parameter

Name	Typ	Beschreibung
linestring1	ST_LineString	Die erste Linienfolge der Mehrfachlinienfolge.
linestring2,...,linestringN	ST_LineString	Zusätzliche Linienfolgenwerte der Mehrfachlinienfolge.

Bemerkungen

Konstruiert eine Mehrfachlinienfolge aus einer Liste von Linienfolgenwerten. Alle angegebenen Linienfolgenwerte müssen dieselbe SRID haben und die Mehrfachlinienfolge wird mit dieser gemeinsamen SRID konstruiert.

Alle angegebenen Linienfolgenwerte müssen die gleichen Antworten für Is3D und IsMeasured aufweisen. Die Mehrfachlinienfolge ist 3D, wenn alle Linienfolgenwerte 3D sind, und die Mehrfachlinienfolge wird gemessen, wenn alle Linienfolgenwerte gemessen werden.

Hinweis

Standardmäßig verwendet ST_MultiLineString das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Der folgende Code gibt eine Mehrfachlinienfolge zurück, die eine einzelne Linienfolge enthält und dem folgenden WKT gleichwertig ist: 'MultiLineString ((0 0, 1 1))'

```
SELECT NEW ST_MultiLineString( NEW ST_LineString('LineString (0 0, 1 1)' ) )
```

Der folgende Code gibt eine Mehrfachlinienfolge zurück, die zwei Linienfolgen enthält und dem folgenden WKT gleichwertig ist: 'MultiLineString ((0 0, 1 1), (0 0, 1 1, 2 0))'.

```
SELECT NEW ST_MultiLineString(
    NEW ST_LineString( 'LineString (0 0, 1 1)' ),
    NEW ST_LineString( 'LineString (0 0, 1 1, 2 0)' ) )
```

ST_MultiLineStringAggr-Methode

Gibt eine Mehrfachlinienfolge zurück, die alle Linienfolgen in einer Gruppe enthält.

Syntax

```
ST_MultiLineString::ST_MultiLineStringAggr(geometry-column[ ORDER BY order-by-expression
[ ASC | DESC ], ... ] )
```

Parameter

Name	Typ	Beschreibung
<i>geometry-column</i>	ST_LineString	Die Geometriewerte zum Generieren der Sammlung. Das ist üblicherweise eine Spalte.

Rückgabe

- **ST_MultiLineString** Gibt eine Mehrfachlinienfolge zurück, die alle Geometrien in einer Gruppe enthält.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist derselbe wie der für den ersten Parameter.

Bemerkungen

Die ST_MultiLineStringAggr-Aggregatfunktion kann verwendet werden, um eine Gruppe von Linienfolgen in einer einzigen Sammlung zu kombinieren. Alle Geometrien, die kombiniert werden sollen, müssen die gleiche SRID und dieselbe Koordinatendimension haben.

Zeilen, in denen das Argument NULL ist, werden nicht eingeschlossen.

Gibt für eine leere Gruppe oder eine Gruppe ohne Nicht-NULL-Werte NULL zurück.

Das sich daraus ergebende ST_MultiLineString-Element hat dieselbe Koordinatendimension wie jede der Linienfolgen.

Die optionale ORDER BY-Klausel kann verwendet werden, um die Elemente in einer bestimmten Reihenfolge anzuordnen, damit ST_GeometryN sie in der gewünschten Reihenfolge zurückgibt. Wenn diese Reihenfolge nicht von Bedeutung ist, sollte aus Effizienzgründen keine Sortierfolge angegeben werden. In diesem Fall hängt die Reihenfolge der Elemente vom Zugriffsplan ab, der vom Abfrageoptimierer ausgewählt wird.

ST_MultiLineStringAggr ist effizienter als ST_UnionAggr, aber ST_MultiLineStringAggr kann eine Sammlung mit mehrfach vorhandenen oder sich überschneidenden Linienfolgen zurückgeben, wenn diese in der Gruppe von Linienfolgen existieren. ST_UnionAggr verarbeitet mehrfach vorhandene und überlappende Geometrien.

Hinweis

Standardmäßig verwendet ST_MultiLineStringAggr das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_UnionAggr-Methode für Typ ST_Geometry](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt einen einzelnen Wert zurück, der alle Geometrien vom Typ ST_LineString aus der SpatialShapes-Tabelle in einer einzigen Sammlung vom Typ ST_MultiLineString kombiniert. Wenn die Shape-Spalte vom Typ ST_LineString ist, sind die TREAT-Funktion und die WHERE-Klausel nicht erforderlich.

```
SELECT ST_MultiLineString::ST_MultiLineStringAggr( TREAT( Shape AS
ST_LineString ) )
FROM SpatialShapes WHERE Shape IS OF( ST_LineString )
```

ST_MultiPoint-Datentyp

ST_MultiPoint ist eine Sammlung von null oder mehr ST_Point-Werten, wobei sich alle Punkte im räumlichen Bezugssystem befinden.

Direkt übergeordneter Typ

- „ST_GeomCollection-Datentyp“

Konstruktor

- „ST_MultiPoint-Konstruktor“

Methoden

- Methoden von ST_MultiPoint:

ST_MultiPointAggr

- Alle Methoden von „ST_GeomCollection-Datentyp“ können auch für einen ST_MultiPoint-Datentyp aufgerufen werden:

ST_GeomCollectionAggr	ST_GeometryN	ST_NumGeometries
---------------------------------------	------------------------------	----------------------------------

- Alle Methoden von „ST_Geometry-Datentyp“ können auch für einen ST_MultiPoint-Datentyp aufgerufen werden:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAgr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr
ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType
ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth
ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBaseType	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine
ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.2

ST_MultiPoint-Konstruktor

Konstruiert einen Mehrfachpunkt.

Überladungsliste

Name	Beschreibung
„ST_MultiPoint()-Konstruktor“	Konstruiert einen Mehrfachpunkt, der die leere Menge darstellt.
„ST_MultiPoint(LONG VARCHAR[, INT])-Konstruktor“	Konstruiert einen Mehrfachpunkt aus einer Textdarstellung.
„ST_MultiPoint(LONG BINARY[, INT])-Konstruktor“	Konstruiert einen Mehrfachpunkt aus Well-Known-Binary-Werten (WKB).
„ST_MultiPoint(ST_Point,...)-Konstruktor“	Konstruiert einen Mehrfachpunkt aus einer Liste von Punktwerten.

ST_MultiPoint()-Konstruktor

Konstruiert einen Mehrfachpunkt, der die leere Menge darstellt.

Syntax

NEW ST_MultiPoint()

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Standardfunktion

Beispiel

Der folgende Code gibt 1 zurück und zeigt damit an, dass der Wert leer ist.

```
SELECT NEW ST_MultiPoint().ST_IsEmpty()
```

ST_MultiPoint(LONG VARCHAR[, INT])-Konstruktor

Konstruiert einen Mehrfachpunkt aus einer Textdarstellung.

Syntax

NEW ST_MultiPoint(text-representation[, srid])

Parameter

Name	Typ	Beschreibung
text-representation	LONG VAR- CHAR	Eine Zeichenfolge, die die Textdarstellung eines Mehrfachpunkts enthält. Die Eingabe kann in jedem unterstützten Texteingabeformat erfolgen, einschließlich Well-Known-Text (WKT) oder Extended-Well-Known-Text (EWKT).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert einen Mehrfachpunkt aus einer Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.2.2

Beispiel

Die folgende Anweisung gibt MultiPoint ((10 10), (12 12), (14 10)) zurück.

```
SELECT NEW ST_MultiPoint('MultiPoint ((10 10), (12 12), (14 10))')
```

ST_MultiPoint(LONG BINARY[, INT])-Konstruktor

Konstruiert einen Mehrfachpunkt aus Well-Known-Binary-Werten (WKB).

Syntax

```
NEW ST_MultiPoint(wkb[, srid])
```

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Eine Zeichenfolge mit der binären Darstellung eines Mehrfachpunkts. Die Eingabe kann in jedem unterstützten binären Eingabeformat erfolgen, einschließlich Well-Known-Binary-Datentypen (WKB) oder Extended-Well-Known-Binary-Datentypen (EWKB).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert einen Mehrfachpunkt aus einer binären Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Der folgende Code gibt einen Mehrfachpunkt zurück, der zwei Punkte 'Point (1 2)' und 'Point (3 4)' enthält.

```
SELECT NEW ST_MultiPoint( NEW ST_Point( 1.0, 2.0 ), NEW ST_Point( 3.0, 4.0 ) )
```

ST_MultiPointAggr-Methode

Gibt einen Mehrfachpunkt zurück, der alle Punkte in einer Gruppe enthält.

Syntax

```
ST_MultiPoint::ST_MultiPointAggr(geometry-column[ ORDER BY order-by-expression [ ASC | DESC ], ... ] )
```

Parameter

Name	Typ	Beschreibung
<i>geometry-column</i>	ST_Point	Die Geometriewerte zum Generieren der Sammlung. Das ist üblicherweise eine Spalte.

Rückgabe

- **ST_MultiPoint** Gibt einen Mehrfachpunkt zurück, der alle Geometrien in einer Gruppe enthält.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist derselbe wie der für den ersten Parameter.

Bemerkungen

Die ST_MultiPointAggr-Aggregatfunktion kann verwendet werden, um eine Gruppe von Punkten in einer einzigen Sammlung zu kombinieren. Alle Geometrien, die kombiniert werden sollen, müssen die gleiche SRID und dieselbe Koordinatendimension haben.

Zeilen, in denen das Argument NULL ist, werden nicht eingeschlossen.

Gibt für eine leere Gruppe oder eine Gruppe ohne Nicht-NULL-Werte NULL zurück.

Das sich daraus ergebende ST_MultiPoint-Element hat dieselbe Koordinatendimension wie jeder der Punkte.

Die optionale ORDER BY-Klausel kann verwendet werden, um die Elemente in einer bestimmten Reihenfolge anzuordnen, damit ST_GeometryN sie in der gewünschten Reihenfolge zurückgibt. Wenn diese Reihenfolge nicht von Bedeutung ist, sollte aus Effizienzgründen keine Sortierfolge angegeben werden. In diesem Fall hängt die Reihenfolge der Elemente vom Zugriffsplan ab, der vom Abfrageoptimierer ausgewählt wird.

ST_MultiPointAggr ist effizienter als ST_UnionAggr, aber ST_MultiPointAggr kann eine Sammlung mit mehrfach vorhandenen oder sich überschneidenden Punkten zurückgeben, wenn sie in der Gruppe der Punkte vorhanden sind. ST_UnionAggr verarbeitet mehrfach vorhandene und überlappende Geometrien.

Hinweis

Standardmäßig verwendet ST_MultiPointAggr das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_UnionAggr-Methode](#) für Typ ST_Geometry

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt einen einzelnen Wert zurück, der alle Geometrien vom Typ ST_Point in der SpatialShapes-Tabelle in einer einzigen Sammlung vom Typ ST_MultiPoint kombiniert. Wenn die Shape-Spalte vom Typ ST_Point ist, sind die TREAT-Funktion und die WHERE-Klausel nicht erforderlich.

```
SELECT ST_MultiPoint::ST_MultiPointAggr( TREAT( Shape AS ST_Point ) )
FROM SpatialShapes WHERE Shape IS OF( ST_Point )
```

ST_MultiPolygon-Datentyp

ST_MultiPolygon ist eine Sammlung von null oder mehr ST_Polygon-Werten, wobei sich alle Polygone im räumlichen Bezugssystem befinden.

Direkt übergeordneter Typ

- „ST_MultiSurface-Datentyp“

Konstruktor

- „ST_MultiPolygon-Konstruktor“

Methoden

- Methoden von ST_MultiPolygon:

[ST_MultiPolygonAggr](#)

- Alle Methoden von „ST_MultiSurface-Datentyp“ können auch für einen ST_MultiPolygon-Datentyp aufgerufen werden:

ST_Area	ST_Centroid	ST_MultiSurfaceAggr	ST_Perimeter
ST_PointOnSurface			

- Alle Methoden von „[ST_GeomCollection-Datentyp](#)“ können auch für einen ST_MultiPolygon-Datentyp aufgerufen werden:

ST_GeomCollectionAggr	ST_GeometryN	ST_NumGeometries
---------------------------------------	------------------------------	----------------------------------

- Alle Methoden von „[ST_Geometry-Datentyp](#)“ können auch für einen ST_MultiPolygon-Datentyp aufgerufen werden:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAgr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr
ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType
ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth
ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBaseType	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine
ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint

ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 9.6

ST_MultiPolygon-Konstruktor

Konstruiert ein Multipolygon.

Überladungsliste

Name	Beschreibung
„ST_MultiPolygon()-Konstruktor“	Konstruiert ein Multipolygon, das die leere Menge darstellt.
„ST_MultiPolygon(LONG VARCHAR[, INT])-Konstruktor“	Konstruiert ein Multipolygon aus einer Textdarstellung.
„ST_MultiPolygon(LONG BINARY[, INT])-Konstruktor“	Konstruiert ein Multipolygon aus Well-Known-Binary-Werten (WKB).
„ST_MultiPolygon(ST_Polygon,...)-Konstruktor“	Konstruiert ein Multipolygon aus einer Liste von Polygonwerten.
„ST_MultiPolygon(ST_MultiLineString[, VARCHAR(128)))-Konstruktor“	Erstellt ein Multipolygon aus einer Mehrfachlinienfolge, die äußere Ringe und eine optionale Liste von inneren Ringen enthält.

ST_MultiPolygon()-Konstruktor

Konstruiert ein Multipolygon, das die leere Menge darstellt.

Syntax

NEW ST_MultiPolygon()

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) Standardfunktion

Beispiel

Der folgende Code gibt 1 zurück und zeigt damit an, dass der Wert leer ist.

```
SELECT NEW ST_MultiPolygon().ST_IsEmpty()
```

ST_MultiPolygon(LONG VARCHAR[, INT])-Konstruktor

Konstruiert ein Multipolygon aus einer Textdarstellung.

Syntax

```
NEW ST_MultiPolygon(text-representation[, srid])
```

Parameter

Name	Typ	Beschreibung
text-representation	LONG VARCHAR	Eine Zeichenfolge, die die Textdarstellung eines Multipolygons enthält. Die Eingabe kann in jedem unterstützten Texteingabeformat erfolgen, einschließlich Well-Known-Text (WKT) oder Extended-Well-Known-Text (EWKT).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert ein Multipolygon aus einer Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 9.6.2

Beispiel

Der folgende Code gibt MultiPolygon (((-5 -5, 5 -5, 0 5, -5 -5), (-2 -2, -2 0, 2 0, 2 -2, -2 -2)), ((10 -5, 15 5, 5 5, 10 -5))) zurück.

```
SELECT NEW ST_MultiPolygon('MultiPolygon (((-5 -5, 5 -5, 0 5, -5 -5), (-2 -2, -2 0, 2 0, 2 -2, -2 -2)), ((10 -5, 15 5, 5 5, 10 -5)))')
```

ST_MultiPolygon(LONG BINARY[, INT])-Konstruktor

Konstruiert ein Multipolygon aus Well-Known-Binary-Werten (WKB).

Syntax

```
NEW ST_MultiPolygon(wkb[, srid])
```

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Eine Zeichenfolge mit der binären Darstellung eines Multipolygons. Die Eingabe kann in jedem unterstützten binären Eingabeformat erfolgen, einschließlich Well-Known-Binary-Datentypen (WKB) oder Extended-Well-Known-Binary-Datentypen (EWKB).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert ein Multipolygon aus einer binären Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.6.2

Beispiel

Die folgende Anweisung gibt MultiPolygon (((10 -5, 15 5, 5 5, 10 -5))) zurück.

```
SELECT NEW
ST_MultiPolygon(0x0106000000001000000010300000001000000040000000000000000244
000000000000014c00000000000002e4000000000000014400000000000014400000000000
144000000000000024400000000000014c0)
```

ST_MultiPolygon(ST_Polygon,...)-Konstruktor

Konstruiert ein Multipolygon aus einer Liste von Polygonwerten.

Syntax

NEW ST_MultiPolygon(*polygon1*[,*polygon2*,...,*polygonN*])

Parameter

Name	Typ	Beschreibung
<i>polygon1</i>	ST_Polygon	Der erste Polygonwert des Multipolygons.
<i>polygon2</i> ,..., <i>polygonN</i>	ST_Polygon	Zusätzliche Polygonwerte des Multipolygons.

Bemerkungen

Konstruiert ein Multipolygon aus einer Liste von Polygonwerten. Alle angegebenen Polygonwerte müssen dieselbe SRID haben und das Multipolygon wird mit dieser gemeinsamen SRID konstruiert.

Alle angegebenen Polygonwerte müssen die gleichen Antworten für Is3D und IsMeasured aufweisen. Das Multipolygon ist 3D, wenn alle Polygonwerte 3D sind, und das Multipolygon wird gemessen, wenn alle Polygonwerte gemessen werden.

Hinweis
Standardmäßig verwendet ST_MultiPolygon das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert MultiPolygon (((0 0, 1 0, 1 1, 0 1, 0 0))) zurück.

```
SELECT NEW ST_MultiPolygon( NEW ST_Polygon('Polygon ((0 0, 0 1, 1 1, 1 0, 0 0))' ) )
```

Das folgende Beispiel gibt den Wert MultiPolygon (((0 0, 1 0, 1 1, 0 1, 0 0)), ((5 5, 10 5, 10 10, 5 10, 5 5))) zurück.

```
SELECT NEW ST_MultiPolygon(
    NEW ST_Polygon('Polygon ((0 0, 0 1, 1 1, 1 0, 0 0))' ),
    NEW ST_Polygon('Polygon ((5 5, 5 10, 10 10, 10 5, 5 5))' ) )
```

ST_MultiPolygon(ST_MultiLineString[, VARCHAR(128)])-Konstruktor

Erstellt ein Multipolygon aus einer Mehrfachlinienfolge, die äußere Ringe und eine optionale Liste von inneren Ringen enthält.

Syntax

```
NEW ST_MultiPolygon(multi-linestring[, polygon-format])
```

Parameter

Name	Typ	Beschreibung
multi-linestring	ST_MultiLineString	Ein Mehrfachlinienfolgenwert, der die äußeren Ringe und (optional) eine Menge innerer Ringe enthält.
polygon-format	VARCHAR(128)	Eine Zeichenfolge mit dem Polygonformat, das beim Interpretieren der angegebenen Zeichenfolgen verwendet werden soll. Gültige Formate sind 'CounterClockwise', 'Clockwise' und 'EvenOdd'.

Bemerkungen

Erstellt ein Multipolygon aus einer Mehrfachlinienfolge, die äußere Ringe und eine optionale Liste von inneren Ringen enthält. Die Mehrfachlinienfolge darf nur lineare Ringe enthalten.

Wenn dieser Parameter benutzt wird, wählt der *polygon-format*-Parameter den Algorithmus aus, den der Server verwendet, um zu ermitteln, ob ein Ring ein äußerer oder innerer Ring ist. Wenn der Parameter nicht festgelegt ist, wird das Polygonformat des räumlichen Bezugssystems verwendet.

Weitere Hinweise zum *polygon-format* finden Sie unter [POLYGON FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Hinweis

Standardmäßig verwendet ST_MultiPolygon das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Der folgende Code gibt MultiPolygon (((-4 -4, 4 -4, 4 4, -4 4, -4 -4), (-2 1, -3 3, -1 3, -2 1)), ((6 -4, 14 -4, 14 4, 6 4, 6 -4), (8 1, 7 3, 9 3, 8 1))) (zwei Quadratpolygone mit jeweils einem dreieckigen Loch) zurück.

```
SELECT NEW ST_MultiPolygon(
    NEW ST_MultiLineString ('MultiLineString ((-4 -4, 4 -4, 4 4, -4 4, -4
-4), (-2 1, -3 3, -1 3, -2 1), (6 -4, 14 -4, 14 4, 6 4, 6 -4), (8 1, 7 3, 9
3, 8 1)))')
```

ST_MultiPolygonAggr-Methode

Gibt ein Multipolygon zurück, das alle Polygone in einer Gruppe enthält.

Syntax

```
ST_MultiPolygon::ST_MultiPolygonAggr(geometry-column [ ORDER BY order-by-expression [ ASC | DESC ], ... ])
```

Parameter

Name	Typ	Beschreibung
<i>geometry-column</i>	ST_Polygon	Die Geometriewerte zum Generieren der Sammlung. Das ist üblicherweise eine Spalte.

Rückgabe

- **ST_MultiPolygon** Gibt ein Multipolygon zurück, das alle Geometrien in einer Gruppe enthält.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist derselbe wie der für den ersten Parameter.

Bemerkungen

Die ST_MultiPolygonAggr-Aggregatfunktion kann verwendet werden, um eine Gruppe von Polygonen in einer einzigen Sammlung zu kombinieren. Alle Geometrien, die kombiniert werden sollen, müssen die gleiche SRID und dieselbe Koordinatendimension haben.

Zeilen, in denen das Argument NULL ist, werden nicht eingeschlossen.

Gibt für eine leere Gruppe oder eine Gruppe ohne Nicht-NULL-Werte NULL zurück.

Das sich daraus ergebende ST_MultiPolygon-Element hat dieselbe Koordinatendimension wie jedes der Polygone.

Die optionale ORDER BY-Klausel kann verwendet werden, um die Elemente in einer bestimmten Reihenfolge anzuordnen, damit ST_GeometryN sie in der gewünschten Reihenfolge zurückgibt. Wenn diese Reihenfolge nicht von Bedeutung ist, sollte aus Effizienzgründen keine Sortierfolge angegeben werden. In diesem Fall hängt die Reihenfolge der Elemente vom Zugriffsplan ab, der vom Abfrageoptimierer ausgewählt wird.

ST_MultiPolygonAggr ist effizienter als ST_UnionAggr, aber ST_MultiPolygonAggr kann eine Sammlung mit mehrfach vorhandenen oder sich überschneidenden Polygonen zurückgeben, wenn sie in der Gruppe von Polygonen vorhanden sind. Insbesondere zurückgegebene Sammlungen, die einander überschneidende Oberflächen enthalten, können unerwartete Ergebnisse hervorrufen, wenn sie als Eingabe für andere räumlichen Methoden verwendet werden. ST_UnionAggr verarbeitet mehrfach vorhandene und überlappende Geometrien.

Hinweis

Standardmäßig verwendet ST_MultiPolygonAggr das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Siehe auch

- [ST_UnionAggr-Methode für Typ ST_Geometry](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt nur einen einzigen Wert zurück, der alle Geometrien vom Typ ST_Polygon aus der SpatialShapes-Tabelle in einer einzigen Sammlung vom Typ ST_MultiPolygon kombiniert. Wenn die Shape-Spalte vom Typ ST_Polygon ist, sind die TREAT-Funktion und die WHERE-Klausel nicht erforderlich.

```
SELECT ST_MultiPolygon::ST_MultiPolygonAggr( TREAT( Shape AS ST_Polygon ) )
FROM SpatialShapes WHERE Shape IS OF( ST_Polygon )
```

ST_MultiSurface-Datentyp

ST_MultiSurface ist eine Sammlung von null oder mehr ST_Surface-Werten, wobei sich alle Flächen im räumlichen Bezugssystem befinden.

Direkt übergeordneter Typ

- „ST_GeomCollection-Datentyp“

Direkt untergeordnete Typen

- „ST_MultiPolygon-Datentyp“

Konstruktor

- „ST_MultiSurface-Konstruktor“

Methoden

- Methoden von ST_MultiSurface:

ST_Area	ST_Centroid	ST_MultiSurfaceAggr	ST_Perimeter
ST_PointOnSurface			

- Alle Methoden von „ST_GeomCollection-Datentyp“ können auch für einen ST_MultiSurface-Datentyp aufgerufen werden:

ST_GeomCollectionAggr	ST_GeometryN	ST_NumGeometries
-----------------------	--------------	------------------

- Alle Methoden von „ST_Geometry-Datentyp“ können auch für einen ST_MultiSurface-Datentyp aufgerufen werden:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAggr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr
ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType

ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth
ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBaseType	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine
ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9,5

ST_MultiSurface-Konstruktor

Konstruiert eine Mehrfachoberfläche.

Überladungsliste

Name	Beschreibung
„ST_MultiSurface()-Konstruktor“	Konstruiert eine Mehrfachoberfläche, die die leere Menge darstellt.

Name	Beschreibung
„ST_MultiSurface(LONG VARCHAR[, INT])-Konstruktor“	Konstruiert eine Mehrfachoberfläche aus einer Textdarstellung.
„ST_MultiSurface(LONG BINARY[, INT])-Konstruktor“	Konstruiert eine Mehrfachoberfläche aus Well-Known-Binary-Werten (WKB).
„ST_MultiSurface(ST_Surface,...)-Konstruktor“	Konstruiert eine Mehrfachoberfläche aus einer Liste von Flächenwerten.
„ST_MultiSurface(ST_MultiCurve[, VARCHAR(128)])-Konstruktor“	Erstellt eine Mehrfachoberfläche aus einer Mehrfachkurve, die äußere Ringe und eine optionale Liste von inneren Ringen enthält.

ST_MultiSurface()-Konstruktor

Konstruiert eine Mehrfachoberfläche, die die leere Menge darstellt.

Syntax

NEW ST_MultiSurface()

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Standardfunktion

Beispiel

Der folgende Code gibt 1 zurück und zeigt damit an, dass der Wert leer ist.

```
SELECT NEW ST_MultiSurface().ST_IsEmpty()
```

ST_MultiSurface(LONG VARCHAR[, INT])-Konstruktor

Konstruiert eine Mehrfachoberfläche aus einer Textdarstellung.

Syntax

NEW ST_MultiSurface(*text-representation*[, *srid*])

Parameter

Name	Typ	Beschreibung
text-representation	LONG VARCHAR	Eine Zeichenfolge, die die Textdarstellung einer Mehrfachoberfläche enthält. Die Eingabe kann in jedem unterstützten Texteingabeformat erfolgen, einschließlich Well-Known-Text (WKT) oder Extended-Well-Known-Text (EWKT).

Name	Typ	Beschreibung
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert eine Mehrfachoberfläche aus einer Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.5.2

Beispiel

Der folgende Code gibt MultiSurface (((-5 -5, 5 -5, 0 5, -5 -5), (-2 -2, -2 0, 2 0, 2 -2, -2 -2)), ((10 -5, 15 5, 5 5, 10 -5))) zurück.

```
SELECT NEW ST_MultiSurface('MultiSurface (((-5 -5, 5 -5, 0 5, -5 -5), (-2 -2, -2 0, 2 0, 2 -2, -2 -2)), ((10 -5, 15 5, 5 5, 10 -5)))')
```

ST_MultiSurface(LONG BINARY[, INT])-Konstruktor

Konstruiert eine Mehrfachoberfläche aus Well-Known-Binary-Werten (WKB).

Syntax

```
NEW ST_MultiSurface(wkb[, srid])
```

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Eine Zeichenfolge mit der binären Darstellung einer Mehrfachoberfläche. Die Eingabe kann in jedem unterstützten binären Eingabeformat erfolgen, einschließlich Well-Known-Binary-Datentypen (WKB) oder Extended-Well-Known-Binary-Datentypen (EWKB).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert eine Mehrfachoberfläche aus einer binären Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.5.2


```
SELECT NEW ST_MultiSurface( NEW ST_Polygon('Polygon ((0 0, 0 1, 1 1, 1 0, 0 0))' ) )
```

Das folgende Beispiel gibt den Wert MultiSurface (((0 0, 1 0, 1 1, 0 1, 0 0)), ((5 5, 10 5, 10 10, 5 10, 5 5))) zurück.

```
SELECT NEW ST_MultiSurface(
    NEW ST_Polygon('Polygon ((0 0, 0 1, 1 1, 1 0, 0 0))' ),
    NEW ST_Polygon('Polygon ((5 5, 5 10, 10 10, 10 5, 5 5))' ) )
```

ST_MultiSurface(ST_MultiCurve[, VARCHAR(128)])-Konstruktor

Erstellt eine Mehrfachoberfläche aus einer Mehrfachkurve, die äußere Ringe und eine optionale Liste von inneren Ringen enthält.

Syntax

```
NEW ST_MultiSurface(multi-curve[, polygon-format])
```

Parameter

Name	Typ	Beschreibung
multi-curve	ST_MultiCurve	Ein Mehrfachkurvenwert, der die äußeren Ringe und (optional) eine Menge innerer Ringe enthält.
polygon-format	VAR-CHAR(128)	Eine Zeichenfolge mit dem Polygonformat, das beim Interpretieren der angegebenen Kurven verwendet werden soll. Gültige Formate sind 'CounterClockwise', 'Clockwise' und 'EvenOdd'.

Bemerkungen

Erstellt eine Mehrfachoberfläche aus einer Mehrfachkurve, die äußere Ringe und eine optionale Liste von inneren Ringen enthält. Die Mehrfachkurve kann jeden Kurventyp enthalten.

Wenn dieser Parameter benutzt wird, wählt der *polygon-format*-Parameter den Algorithmus aus, den der Server verwendet, um zu ermitteln, ob ein Ring ein äußerer oder innerer Ring ist. Wenn der Parameter nicht festgelegt ist, wird das Polygonformat des räumlichen Bezugssystems verwendet.

Weitere Hinweise zum *polygon-format* finden Sie unter [POLYGON FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Hinweis

Standardmäßig verwendet ST_MultiSurface das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung \[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Die folgende Anweisung gibt MultiSurface (CurvePolygon ((-4 -4, 4 -4, 4 4, -4 4, -4 -4), (-2 1, -3 3, -1 3, -2 1)), CurvePolygon ((6 -4, 14 -4, 14 4, 6 4, 6 -4), CircularString (9 -1, 9 1, 11 1, 11 -1, 9 -1))) zurück.

```
SELECT NEW ST_MultiSurface(NEW ST_MultiCurve ('MultiCurve ((-4 -4, 4 -4, 4
4, -4 4, -4 -4), (-2 1, -3 3, -1 3, -2 1), (6 -4, 14 -4, 14 4, 6 4, 6 -4),
CircularString (9 -1, 9 1, 11 1, 11 -1, 9 -1))'))
```

ST_Area-Methode

Berechnet den Bereich der Mehrfachoberfläche in den angegebenen Einheiten.

Syntax

multisurface-expression.**ST_Area**([*unit-name*])

Parameter

Name	Typ	Beschreibung
unit-name	VAR-CHAR(128)	Die Einheiten, in denen der Bereich berechnet werden soll. Ist standardmäßig die Einheit des räumlichen Bezugssystems. Der unit-name-Parameter muss mit der UNIT_NAME-Spalte einer Zeile in der ST_UNITS_OF_MEASURE-Ansicht übereinstimmen, wo UNIT_TYPE auf 'LINEAR' eingestellt ist.

Rückgabe

- **DOUBLE** Gibt den Bereich der Mehrfachoberfläche zurück.

Bemerkungen

Berechnet den Bereich der Mehrfachoberfläche in den angegebenen Einheiten. Der Bereich der Mehrfachoberfläche ist die Summe der Bereiche der enthaltenen Flächen.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

- [ST_Perimeter-Methode](#) für den ST_MultiSurface-Datentyp
- [ST_Area-Methode](#) für den ST_Surface-Datentyp
- [ST_Length-Methode](#) für den ST_MultiCurve-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.5.3

Beispiel

Das folgende Beispiel gibt den Wert 8 zurück.

```
SELECT TREAT( Shape AS ST_MultiSurface ).ST_Area()  
FROM SpatialShapes WHERE ShapeID = 27
```

Der folgende Code gibt den Bereich der multipoly_geometry-Spalte in Quadratmeilen aus der fiktiven region-Tabelle zurück.

```
SELECT name, multipoly_geometry.ST_Area( 'Statute Mile' )  
FROM region
```

ST_Centroid-Methode

Berechnet den ST_Point, der der mathematische Schwerpunkt der Mehrfachoberfläche ist.

Syntax

multisurface-expression.ST_Centroid()

Rückgabe

- **ST_Point** Wenn die Mehrfachoberfläche eine leere Menge ist, wird NULL zurückgegeben. Andernfalls wird der mathematische Schwerpunkt der Fläche zurückgegeben.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *multisurface-expression*.

Bemerkungen

Berechnet den ST_Point, der der mathematische Schwerpunkt der Mehrfachoberfläche ist. Dieser Punkt muss nicht unbedingt ein Punkt auf der Oberfläche sein.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

- [ST_Centroid-Methode für den ST_Surface-Datentyp](#)
- [ST_PointOnSurface-Methode für den ST_MultiSurface-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.5.5

Beispiel

Das folgende Beispiel gibt den Wert Point (1.865682 .664892) zurück.

```
SELECT TREAT( Shape AS ST_MultiSurface ).ST_Centroid()  
FROM SpatialShapes WHERE ShapeID = 28
```

ST_MultiSurfaceAggr-Methode

Gibt eine Mehrfachoberfläche mit allen Flächen in einer Gruppe zurück.

Syntax

ST_MultiSurface::ST_MultiSurfaceAggr(*geometry-column*[**ORDER BY** *order-by-expression* [**ASC** | **DESC**], ...])

Parameter

Name	Typ	Beschreibung
<i>geometry-column</i>	ST_Surface	Die Geometriewerte zum Generieren der Sammlung. Das ist üblicherweise eine Spalte.

Rückgabe

- **ST_MultiSurface** Gibt eine Mehrfachoberfläche zurück, die alle Geometrien in einer Gruppe enthält.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist derselbe wie der für den ersten Parameter.

Bemerkungen

Die ST_MultiSurfaceAggr-Aggregatfunktion kann verwendet werden, um eine Gruppe von Flächen in einer einzigen Sammlung zu kombinieren. Alle Geometrien, die kombiniert werden sollen, müssen die gleiche SRID und dieselbe Koordinatendimension haben.

Zeilen, in denen das Argument NULL ist, werden nicht eingeschlossen.

Gibt für eine leere Gruppe oder eine Gruppe ohne Nicht-NULL-Werte NULL zurück.

Das sich daraus ergebende ST_MultiSurface-Element hat dieselbe Koordinatendimension wie jede der Flächen.

Die optionale ORDER BY-Klausel kann verwendet werden, um die Elemente in einer bestimmten Reihenfolge anzuordnen, damit ST_GeometryN sie in der gewünschten Reihenfolge zurückgibt. Wenn diese Reihenfolge nicht von Bedeutung ist, sollte aus Effizienzgründen keine Sortierfolge angegeben werden. In diesem Fall hängt die Reihenfolge der Elemente vom Zugriffsplan ab, der vom Abfrageoptimierer ausgewählt wird.

ST_MultiSurfaceAggr ist effizienter als ST_UnionAggr, aber ST_MultiSurfaceAggr kann eine Sammlung mit mehrfach vorhandenen oder sich überschneidenden Flächen zurückgeben, wenn sie in der Gruppe von Flächen vorhanden sind. Insbesondere zurückgegebene Sammlungen, die einander überschneidende Oberflächen enthalten, können unerwartete Ergebnisse hervorrufen, wenn sie als Eingabe für andere räumlichen Methoden verwendet werden. ST_UnionAggr verarbeitet mehrfach vorhandene und überlappende Geometrien.

Hinweis

Standardmäßig verwendet ST_MultiSurfaceAggr das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_UnionAggr-Methode für Typ ST_Geometry](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt nur einen einzigen Wert zurück, der alle Geometrien vom Typ ST_Surface aus der SpatialShapes-Tabelle in einer einzigen Sammlung vom Typ ST_MultiSurface kombiniert. Wenn die Shape-Spalte vom Typ ST_Surface ist, sind die TREAT-Funktion und die WHERE-Klausel nicht erforderlich.

```
SELECT ST_MultiSurface::ST_MultiSurfaceAggr( TREAT( Shape AS ST_Surface ) )
FROM SpatialShapes WHERE Shape IS OF( ST_Surface )
```

ST_Perimeter-Methode

Berechnet den Umfang der Mehrfachoberfläche in den angegebenen Einheiten.

Syntax

```
multisurface-expression.ST_Perimeter([ unit-name])
```

Parameter

Name	Typ	Beschreibung
unit-name	VAR-CHAR(128)	Die Einheiten, in denen der Umfang berechnet werden soll. Ist standardmäßig die Einheit des räumlichen Bezugssystems. Der unit-name-Parameter muss mit der UNIT_NAME-Spalte einer Zeile in der ST_UNITS_OF_MEASURE-Ansicht übereinstimmen, wo UNIT_TYPE auf 'LINEAR' eingestellt ist.

Rückgabe

- **DOUBLE** Gibt den Umfang der Mehrfachoberfläche zurück.

Bemerkungen

Die ST_Perimeter-Methode gibt die Länge des Umfangs einer Mehrfachoberfläche in den Einheiten zurück, die durch den *unit_name*-Parameter festgelegt sind. Wenn die Mehrfachoberfläche leer ist, wird NULL zurückgegeben.

Wenn die Mehrfachoberfläche Z-Werte enthält, werden diese beim Berechnen des Umfangs der Geometrie nicht in Betracht gezogen.

Der Umfang eines Polygons enthält die Länge aller Ringe (äußere und innere).

Hinweis

Wenn *multisurface-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_Perimeter das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_Perimeter-Methode](#) für den ST_Surface-Datentyp
- [ST_Boundary-Methode](#) für den ST_Geometry-Datentyp
- [ST_Length-Methode](#) für den ST_MultiCurve-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.5.4

Beispiel

Im folgenden Beispiel wird eine Mehrfachoberfläche mit zwei Polygonen erstellt und ST_Perimeter für die Suche nach der Länge der Umfangs verwendet. Die Rückgabe lautet 44.

```
SELECT NEW ST_MultiSurface( NEW ST_Polygon('Polygon((0 0, 1 0, 1 1,0 1, 0
0))')
                        , NEW ST_Polygon('Polygon((10 10, 20 10, 20 20,10 20, 10
10))') )
      .ST_Perimeter()
```

Im folgenden Beispiel wird eine Mehrfachoberfläche mit zwei Polygonen und einer Beispiel-Maßeinheit (example_unit_halfmetre) erstellt. ST_Perimeter sucht die Länge des Umfangs und gibt den Wert 88,0 zurück.

```
CREATE SPATIAL UNIT OF MEASURE IF NOT EXISTS "example_unit_halfmetre" TYPE
LINEAR CONVERT USING .5;
SELECT NEW ST_MultiSurface( NEW ST_Polygon('Polygon((0 0, 1 0, 1 1,0 1, 0
0))')
                        , NEW ST_Polygon('Polygon((10 10, 20 10, 20 20,10 20, 10
10))') )
      .ST_Perimeter('example_unit_halfmetre');
```

ST_PointOnSurface-Methode

Gibt einen Punkt zurück, der garantiert auf einer Fläche im Mehrfachoberflächenobjekt liegt

Syntax

multisurface-expression.**ST_PointOnSurface()**

Rückgabe

- **ST_Point** Wenn die Mehrfachoberfläche eine leere Menge ist, wird NULL zurückgegeben. Andernfalls wird ein ST_Point-Wert zurückgegeben, bei dem garantiert wird, dass er räumlich eine Schnittmenge mit dem ST_MultiSurface-Wert hat.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *multisurface-expression*.

Bemerkungen

Gibt einen Punkt zurück, der sich im Innenbereich einer der Flächen einer Mehrfachoberfläche befindet.

Hinweis

Wenn *multisurface-expression* Kreisbogenfolgen enthält, werden diese in Linienfolgen interpoliert.

Siehe auch

- [ST_PointOnSurface-Methode für den ST_Surface-Datentyp](#)
- [ST_Centroid-Methode für den ST_MultiSurface-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.5.6

Beispiel

Der folgende Code gibt einen Punkt zurück, der die Mehrfachoberfläche schneidet.

```
SELECT TREAT( Shape AS ST_MultiSurface ).ST_PointOnSurface()  
FROM SpatialShapes WHERE ShapeID = 27
```

ST_Point-Datentyp

Der ST_Point-Datentyp ist eine 0-dimensionale Geometrie und stellt einen einzigen Standort dar.

Direkt übergeordneter Typ

- „ST_Geometry-Datentyp“

Konstruktor

- „ST_Point-Konstruktor“

Methoden

- Methoden von ST_Point:

ST_Lat	ST_Long	ST_M	ST_X
ST_Y	ST_Z		

- Alle Methoden von „ST_Geometry-Datentyp“ können auch für einen ST_Point-Datentyp aufgerufen werden:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAggr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr
ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType
ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth
ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBaseType	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine
ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin

ST_YMax	ST_YMin	ST_ZMax	ST_ZMin
---------	---------	---------	---------

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 6.1

ST_Point-Konstruktor

Konstruiert einen Punkt.

Hinweis

Bei der Erstellung eines ST_Point-Werts aus Koordinaten ist die ausgewählte Überladung nicht immer vorhersehbar. Beispiel: Der Ausdruck "NEW ST_Point(1,2,3)" erstellt einen 2D-Punkt mit X=1, Y=2 und SRID=3. Der Ausdruck "NEW ST_Point(1,2,3.0)" erstellt einen 3D-Punkt mit Z=3.0.

Überladungsliste

Name	Beschreibung
„ST_Point()-Konstruktor“	Konstruiert einen Punkt, der die leere Menge darstellt.
„ST_Point(LONG VARCHAR[, INT])-Konstruktor“	Konstruiert einen Punkt aus einer Textdarstellung.
„ST_Point(LONG BINARY[, INT])-Konstruktor“	Konstruiert einen Punkt aus Well-Known-Binary-Werten (WKB).
„ST_Point(DOUBLE,DOUBLE[, INT])-Konstruktor“	Konstruiert einen 2D-Punkt aus den X,Y-Koordinaten.
„ST_Point(DOUBLE,DOUBLE,DOUBLE[, INT])-Konstruktor“	Konstruiert einen 3D-Punkt aus den X,Y,Z-Koordinaten.
„ST_Point(DOUBLE,DOUBLE,DOUBLE,DOUBLE[, INT])-Konstruktor“	Konstruiert einen gemessenen 3D-Punkt aus X,Y,Z-Koordinaten und einem Messwert.

ST_Point()-Konstruktor

Konstruiert einen Punkt, der die leere Menge darstellt.

Syntax

NEW ST_Point()

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Standardfunktion

Beispiel

Der folgende Code gibt 1 zurück und zeigt damit an, dass der Wert leer ist.

```
SELECT NEW ST_Point().ST_IsEmpty()
```

ST_Point(LONG VARCHAR[, INT])-Konstruktor

Konstruiert einen Punkt aus einer Textdarstellung.

Syntax

```
NEW ST_Point(text-representation[, srid])
```

Parameter

Name	Typ	Beschreibung
text-representation	LONG VARCHAR	Eine Zeichenfolge, die die Textdarstellung eines Punkts enthält. Die Eingabe kann in jedem unterstützten Texteingabeformat erfolgen, einschließlich Well-Known-Text (WKT) oder Extended-Well-Known-Text (EWKT).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert einen Punkt aus einer Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.2

Beispiel

Die folgende Anweisung gibt Point (10 20) zurück.

```
SELECT NEW ST_Point('Point (10 20)')
```

ST_Point(LONG BINARY[, INT])-Konstruktor

Konstruiert einen Punkt aus Well-Known-Binary-Werten (WKB).

Syntax

```
NEW ST_Point(wkb[, srid])
```

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Eine Zeichenfolge mit der binären Darstellung eines Punkts. Die Eingabe kann in jedem unterstützten binären Eingabeformat erfolgen, einschließlich Well-Known-Binary-Datentypen (WKB) oder Extended-Well-Known-Binary-Datentypen (EWKB).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert einen Punkt aus einer binären Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.2

Beispiel

Die folgende Anweisung gibt Point (10 20) zurück.

```
SELECT NEW ST_Point(0x0101000000000000000000000024400000000000003440)
```

ST_Point(DOUBLE,DOUBLE[, INT])-Konstruktor

Konstruiert einen 2D-Punkt aus den X,Y-Koordinaten.

Syntax

NEW ST_Point(*x*,*y*[, *srid*])

Parameter

Name	Typ	Beschreibung
x	DOUBLE	Der X-Koordinatenwert.
y	DOUBLE	Der Y-Koordinatenwert.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.2

Beispiel

Die folgende Anweisung gibt Point (10 20) zurück.

```
SELECT NEW ST_Point(10.0,20.0,0)
```

ST_Point(DOUBLE,DOUBLE,DOUBLE[, INT])-Konstruktor

Konstruiert einen 3D-Punkt aus den X,Y,Z-Koordinaten.

Syntax

```
NEW ST_Point(x,y,z[, srid])
```

Parameter

Name	Typ	Beschreibung
x	DOUBLE	Der X-Koordinatenwert.
y	DOUBLE	Der Y-Koordinatenwert.
z	DOUBLE	Der Z-Koordinatenwert.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.2

Beispiel

Die folgende Anweisung gibt Point Z (10 20 100) zurück.

```
SELECT NEW ST_Point(10.0,20.0,100.0,0)
```

ST_Point(DOUBLE,DOUBLE,DOUBLE,DOUBLE[, INT])-Konstruktor

Konstruiert einen gemessenen 3D-Punkt aus X,Y,Z-Koordinaten und einem Messwert.

Syntax

```
NEW ST_Point(x,y,z,m[, srid])
```

Parameter

Name	Typ	Beschreibung
x	DOUBLE	Der X-Koordinatenwert.
y	DOUBLE	Der Y-Koordinatenwert.
z	DOUBLE	Der Z-Koordinatenwert.
m	DOUBLE	Der Messwert.

Name	Typ	Beschreibung
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.2

Beispiel

Die folgende Anweisung gibt ZM (10 20 100 1224) zurück.

```
SELECT NEW ST_Point(10.0,20.0,100.0,1224.0,0)
```

ST_Lat-Methode

Gibt die Breitengrad-Koordinate des ST_Point-Werts zurück.

Überladungsliste

Name	Beschreibung
„ST_Lat()-Methode für den ST_Point-Datentyp“	Gibt die Breitengrad-Koordinate des ST_Point-Werts zurück.
„ST_Lat(DOUBLE)-Methode für den ST_Point-Datentyp“	Gibt eine Kopie des Punkts mit der auf den angegebenen Breitengradwert eingestellten Breitengradkoordinate zurück.

ST_Lat()-Methode für den ST_Point-Datentyp

Gibt die Breitengrad-Koordinate des ST_Point-Werts zurück.

Hinweis

Wenn *point-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_Lat das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

point-expression.ST_Lat()

Rückgabe

- **DOUBLE** Gibt die Breitengrad-Koordinate des ST_Point-Werts zurück.

Siehe auch

- [ST_Long-Methode](#) für den ST_Point-Datentyp
- [ST_Y-Methode](#) für den ST_Point-Datentyp
- [ST_LatNorth-Methode](#) für den ST_Geometry-Datentyp
- [ST_LatSouth-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Beim folgenden Beispiel wird ein Fehler ausgegeben, weil das räumliches Bezugssystem, das mit 0 identifiziert wurde, kein geografisches räumliches Bezugssystem ist.

```
SELECT NEW ST_Point( 10.0, 20.0, 0 ).ST_Lat()
```

Das folgende Beispiel gibt den Wert 20.0 zurück.

```
SELECT NEW ST_Point( 10.0, 20.0, 4326 ).ST_Lat()
```

ST_Lat(DOUBLE)-Methode für den ST_Point-Datentyp

Gibt eine Kopie des Punkts mit der auf den angegebenen Breitengradwert eingestellten Breitengradkoordinate zurück.

Hinweis

Wenn *point-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_Lat das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Syntax

point-expression.**ST_Lat**(*latitude-val*)

Parameter

Name	Typ	Beschreibung
latitude-val	DOUBLE	Der neue Breitengradwert

Rückgabe

- **ST_Point** Gibt eine Kopie des Punkts mit den auf den angegebenen Wert eingestellten Breitengradkoordinaten zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *point-expression*.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

ST_Long-Methode

Gibt die Längengradkoordinate des ST_Point-Werts zurück.

Überladungsliste

Name	Beschreibung
„ST_Long()-Methode für den ST_Point-Datentyp“	Gibt die Längengradkoordinate des ST_Point-Werts zurück.
„ST_Long(DOUBLE)-Methode für den ST_Point-Datentyp“	Gibt eine Kopie des Punkts mit den auf den angegebenen Längengradwert eingestellten Längengradkoordinaten zurück.

ST_Long()-Methode für den ST_Point-Datentyp

Gibt die Längengradkoordinate des ST_Point-Werts zurück.

Hinweis

Wenn *point-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_Long das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

point-expression.ST_Long()

Rückgabe

- **DOUBLE** Gibt die Längengradkoordinate des ST_Point-Werts zurück.

Siehe auch

- [ST_Lat-Methode](#) für den ST_Point-Datentyp
- [ST_X-Methode](#) für den ST_Point-Datentyp
- [ST_LongEast-Methode](#) für den ST_Geometry-Datentyp
- [ST_LongWest-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Beim folgenden Beispiel wird ein Fehler ausgegeben, weil das räumliches Bezugssystem, das mit 0 identifiziert wurde, kein geografisches räumliches Bezugssystem ist.

```
SELECT NEW ST_Point( 10.0, 20.0, 0 ).ST_Long()
```

Das folgende Beispiel gibt den Wert 10.0 zurück.

```
SELECT NEW ST_Point( 10.0, 20.0, 4326 ).ST_Long()
```

ST_Long(DOUBLE)-Methode für den ST_Point-Datentyp

Gibt eine Kopie des Punkts mit den auf den angegebenen Längengradwert eingestellten Längengradkoordinaten zurück.

Hinweis

Wenn *point-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_Long das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

point-expression.**ST_Long**(*longitude-val*)

Parameter

Name	Typ	Beschreibung
longitude-val	DOUBLE	Der neue Längengradwert

Rückgabe

- **ST_Point** Gibt eine Kopie des Punkts mit den auf den angegebenen Wert eingestellten Längengradkoordinaten zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *point-expression*.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

ST_M-Methode

Ruft den Messwert eines Punkts ab oder ändert ihn.

Überladungsliste

Name	Beschreibung
„ST_M()-Methode für den ST_Point-Datentyp“	Gibt den Messwert des ST_Point-Werts zurück.
„ST_M(DOUBLE)-Methode für den ST_Point-Datentyp“	Gibt eine Kopie des Punkts mit dem auf den angegebenen M-Koordinatenwert eingestellten Messwert zurück.

ST_M()-Methode für den ST_Point-Datentyp

Gibt den Messwert des ST_Point-Werts zurück.

Hinweis

Wenn *point-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_M das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

point-expression.**ST_M()**

Rückgabe

- **DOUBLE** Gibt den Messwert des ST_Point-Werts zurück.

Siehe auch

- [ST_X-Methode für den ST_Point-Datentyp](#)
- [ST_Y-Methode für den ST_Point-Datentyp](#)
- [ST_MMin-Methode für den ST_Geometry-Datentyp](#)
- [ST_MMax-Methode für den ST_Geometry-Datentyp](#)
- [ST_IsMeasured-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.6

Beispiel

Das folgende Beispiel gibt den Wert 40.0 zurück.

```
SELECT NEW ST_Point( 10.0, 20.0, 30.0, 40.0, 0 ).ST_M()
```

ST_M(DOUBLE)-Methode für den ST_Point-Datentyp

Gibt eine Kopie des Punkts mit dem auf den angegebenen M-Koordinatenwert eingestellten Messwert zurück.

Hinweis

Wenn *point-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_M das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

point-expression.**ST_M**(*mcoord*)

Parameter

Name	Typ	Beschreibung
mcoord	DOUBLE	Der neue Messwert.

Rückgabe

- **ST_Point** Gibt eine Kopie des Punkts mit dem auf den angegebenen M-Koordinatenwert eingestellten Messwert zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *point-expression*.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.6

Beispiel

Das folgende Beispiel gibt den Wert `Point ZM (1 2 3 5)` zurück.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0 ).ST_M( 5.0 )
```

ST_X-Methode

Ruft die X-Koordinate eines Punkts ab oder ändert sie.

Überladungsliste

Name	Beschreibung
„ST_X()-Methode für den ST_Point-Datentyp“	Gibt die X-Koordinate des ST_Point-Werts zurück.
„ST_X(DOUBLE)-Methode für den ST_Point-Datentyp“	Gibt eine Kopie des Punkts mit der auf den angegebenen X-Koordinatenwert eingestellten X-Koordinate zurück.

ST_X()-Methode für den ST_Point-Datentyp

Gibt die X-Koordinate des ST_Point-Werts zurück.

Hinweis

Wenn *point-expression* eine leere Geometrie (`ST_IsEmpty()`=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_X das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

point-expression.**ST_X**()

Rückgabe

- **DOUBLE** Gibt die X-Koordinate des ST_Point-Werts zurück.

Siehe auch

- [ST_Y-Methode für den ST_Point-Datentyp](#)
- [ST_Lat-Methode für den ST_Point-Datentyp](#)
- [ST_XMin-Methode für den ST_Geometry-Datentyp](#)
- [ST_XMax-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.3

Beispiel

Das folgende Beispiel gibt den Wert 10.0 zurück.

```
SELECT NEW ST_Point( 10.0, 20.0, 30.0, 40.0, 0 ).ST_X()
```

ST_X(DOUBLE)-Methode für den ST_Point-Datentyp

Gibt eine Kopie des Punkts mit der auf den angegebenen X-Koordinatenwert eingestellten X-Koordinate zurück.

Hinweis

Wenn *point-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_X das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel, CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

point-expression.**ST_X**(*xcoord*)

Parameter

Name	Typ	Beschreibung
xcoord	DOUBLE	Der neue X-Koordinatenwert.

Rückgabe

- **ST_Point** Gibt eine Kopie des Punkts mit der auf den angegebenen X-Koordinatenwert eingestellten X-Koordinate zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *point-expression*.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.3

ST_Y-Methode

Ruft die Y-Koordinate eines Punkts ab oder ändert sie.

Überladungsliste

Name	Beschreibung
„ST_Y()-Methode für den ST_Point-Datentyp“	Gibt die Y-Koordinate des ST_Point-Werts zurück.
„ST_Y(DOUBLE)-Methode für den ST_Point-Datentyp“	Gibt eine Kopie des Punkts mit der auf den angegebenen Y-Koordinatenwert eingestellten Y-Koordinate zurück.

ST_Y()-Methode für den ST_Point-Datentyp

Gibt die Y-Koordinate des ST_Point-Werts zurück.

Hinweis

Wenn *point-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_Y das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

point-expression.ST_Y()

Rückgabe

- **DOUBLE** Gibt die Y-Koordinate des ST_Point-Werts zurück.

Siehe auch

- [ST_X-Methode für den ST_Point-Datentyp](#)
- [ST_Long-Methode für den ST_Point-Datentyp](#)
- [ST_YMin-Methode für den ST_Geometry-Datentyp](#)
- [ST_YMax-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.4

Beispiel

Das folgende Beispiel gibt den Wert 20.0 zurück.

```
SELECT NEW ST_Point( 10.0, 20.0, 30.0, 40.0, 0 ).ST_Y()
```

ST_Y(DOUBLE)-Methode für den ST_Point-Datentyp

Gibt eine Kopie des Punkts mit der auf den angegebenen Y-Koordinatenwert eingestellten Y-Koordinate zurück.

Hinweis

Wenn *point-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_Y das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

point-expression.ST_Y(*ycoord*)

Parameter

Name	Typ	Beschreibung
ycoord	DOUBLE	Der neue Y-Koordinatenwert.

Rückgabe

- **ST_Point** Gibt eine Kopie des Punkts mit der auf den angegebenen Y-Koordinatenwert eingestellten Y-Koordinate zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *point-expression*.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.4

Beispiel

Das folgende Beispiel gibt den Wert Point (1 3) zurück.

```
SELECT NEW ST_Point( 1, 2 ).ST_Y( 3 )
```

ST_Z-Methode

Ruft die Z-Koordinate eines Punkts ab oder ändert sie.

Überladungsliste

Name	Beschreibung
„ST_Z()-Methode für den ST_Point-Datentyp“	Gibt die Z-Koordinate des ST_Point-Werts zurück.
„ST_Z(DOUBLE)-Methode für den ST_Point-Datentyp“	Gibt eine Kopie des Punkts mit der auf den angegebenen Z-Koordinatenwert eingestellten Z-Koordinate zurück.

ST_Z()-Methode für den ST_Point-Datentyp

Gibt die Z-Koordinate des ST_Point-Werts zurück.

Hinweis

Wenn *point-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_Z das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

point-expression.ST_Z()

Rückgabe

- **DOUBLE** Gibt die Z-Koordinate des ST_Point-Werts zurück.

Siehe auch

- [ST_X-Methode für den ST_Point-Datentyp](#)
- [ST_Y-Methode für den ST_Point-Datentyp](#)
- [ST_ZMin-Methode für den ST_Geometry-Datentyp](#)
- [ST_ZMax-Methode für den ST_Geometry-Datentyp](#)
- [ST_Is3D-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.4

Beispiel

Das folgende Beispiel gibt den Wert 30.0 zurück.

```
SELECT NEW ST_Point( 10.0, 20.0, 30.0, 40.0, 0 ).ST_Z()
```

ST_Z(DOUBLE)-Methode für den ST_Point-Datentyp

Gibt eine Kopie des Punkts mit der auf den angegebenen Z-Koordinatenwert eingestellten Z-Koordinate zurück.

Hinweis

Wenn *point-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_Z das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

point-expression.ST_Z(*zcoord*)

Parameter

Name	Typ	Beschreibung
zcoord	DOUBLE	Der neue Z-Koordinatenwert.

Rückgabe

- **ST_Point** Gibt eine Kopie des Punkts mit der auf den angegebenen Z-Koordinatenwert eingestellten Z-Koordinate zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *point-expression*.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.5

Beispiel

Das folgende Beispiel gibt den Wert `Point Z (1 2 5)` zurück.

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0 ).ST_Z( 5.0 )
```

ST_Polygon-Datentyp

ST_Polygon ist ein ST_CurvePolygon, das mit inneren und äußeren Ringen geformt wird, die lineare Ringe sind.

Direkt übergeordneter Typ

- „ST_CurvePolygon-Datentyp“

Konstruktor

- „ST_Polygon-Konstruktor“

Methoden

- Methoden von ST_Polygon:

ST_ExteriorRing	ST_InteriorRingN
-----------------	------------------

- Alle Methoden von „ST_CurvePolygon-Datentyp“ können auch für einen ST_Polygon-Datentyp aufgerufen werden:

ST_CurvePolyToPoly	ST_ExteriorRing	ST_InteriorRingN	ST_NumInteriorRing
--------------------	-----------------	------------------	--------------------

- Alle Methoden von „ST_Surface-Datentyp“ können auch für einen ST_Polygon-Datentyp aufgerufen werden:

ST_Area	ST_Centroid	ST_IsWorld	ST_Perimeter
ST_PointOnSurface			

- Alle Methoden von „ST_Geometry-Datentyp“ können auch für einen ST_Polygon-Datentyp aufgerufen werden:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAgr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr
ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType
ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth

ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurati- onData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBase- Type	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine
ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFil- ter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 8.3

ST_Polygon-Konstruktor

Konstruiert ein Polygon.

Überladungsliste

Name	Beschreibung
„ST_Polygon()-Konstruktor“	Konstruiert ein Polygon, das die leere Menge darstellt.
„ST_Polygon(LONG VARCHAR[, INT])-Konstruktor“	Konstruiert ein Polygon aus einer Textdarstellung.
„ST_Polygon(LONG BINARY[, INT])-Konstruktor“	Konstruiert ein Polygon aus Well-Known-Binary-Werten (WKB).
„ST_Polygon(ST_Point,ST_Point)-Konstruktor“	Erstellt ein achsenausgerichtetes Rechteck aus zwei Punkten, die die linke untere und die rechte obere Ecke darstellen.

Name	Beschreibung
„ST_Polygon(ST_MultiLineString[, VARCHAR(128)])-Konstruktor“	Erstellt ein Polygon aus einer Mehrfachlinienfolge mit einem äußeren Ring und einer optionalen Liste von inneren Ringen.
„ST_Polygon(ST_LineString,...)-Konstruktor“	Erstellt ein Polygon aus einer Mehrfachlinienfolge, die den äußeren Ring darstellt, und einer optionalen Liste von Liniensequenzen, die innere Ringe darstellen.

ST_Polygon()-Konstruktor

Konstruiert ein Polygon, das die leere Menge darstellt.

Syntax

NEW ST_Polygon()

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Standardfunktion

Beispiel

Der folgende Code gibt 1 zurück und zeigt damit an, dass der Wert leer ist.

```
SELECT NEW ST_Polygon().ST_IsEmpty()
```

ST_Polygon(LONG VARCHAR[, INT])-Konstruktor

Konstruiert ein Polygon aus einer Textdarstellung.

Syntax

NEW ST_Polygon(*text-representation*[, *srid*])

Parameter

Name	Typ	Beschreibung
text-representation	LONG VARCHAR	Eine Zeichenfolge, die die Textdarstellung eines Polygons enthält. Die Eingabe kann in jedem unterstützten Texteingabeformat erfolgen, einschließlich Well-Known-Text (WKT) oder Extended-Well-Known-Text (EWKT).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert ein Polygon aus einer Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.3.2

Beispiel

Der folgende Code gibt Polygon ((-5 -5, 5 -5, 0 5, -5 -5), (-2 -2, -2 0, 2 0, 2 -2, -2 -2)) zurück.

```
SELECT NEW ST_Polygon('Polygon ((-5 -5, 5 -5, 0 5, -5 -5), (-2 -2, -2 0, 2 0, 2 -2, -2 -2))')
```

ST_Polygon(LONG BINARY[, INT])-Konstruktor

Konstruiert ein Polygon aus Well-Known-Binary-Werten (WKB).

Syntax

NEW ST_Polygon(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Eine Zeichenfolge mit der binären Darstellung eines Polygons. Die Eingabe kann in jedem unterstützten binären Eingabeformat erfolgen, einschließlich Well-Known-Binary-Datentypen (WKB) oder Extended-Well-Known-Binary-Datentypen (EWKB).
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Bemerkungen

Konstruiert ein Polygon aus einer binären Zeichenfolgendarstellung. Der Datenbankserver ermittelt das Eingabeformat durch Überprüfen der angegebenen Zeichenfolge.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.3.2

Beispiel

Die folgende Anweisung gibt Polygon ((10 -5, 15 5, 5 5, 10 -5)) zurück.

```
SELECT NEW
ST_Polygon(0x010300000001000000040000000000000024400000000000014c0000000
0000002e4000000000000001440000000000001440000000000001440000000000002440000
00000000014c0)
```

ST_Polygon(ST_Point,ST_Point)-Konstruktor

Erstellt ein achsenausgerichtetes Rechteck aus zwei Punkten, die die linke untere und die rechte obere Ecke darstellen.

Syntax

```
NEW ST_Polygon(pmin,pmax)
```

Parameter

Name	Typ	Beschreibung
pmin	ST_Point	Ein Punkt, der die linke untere Ecke des Rechtecks ist.
pmax	ST_Point	Ein Punkt, der die rechte obere Ecke des Rechtecks ist.

Bemerkungen

Gibt ein Rechteck zurück, das als Rahmen von zwei Punkten definiert ist.

Der Konstruktor entspricht dem Folgenden: `NEW ST_MultiPoint(pmin, pmax).ST_Envelope()`

Wenn die Eingabepunkte nicht die entsprechenden Ecken eines Rechtecks angeben, ermittelt der Server das korrekte Rechteck für die angegebenen Punkte.

Hinweis

Wenn Eingabepunkte sich in einem räumlichen Bezugssystem mit gewölbter Erddarstellung befinden, ist das erzeugte Polygon kein achsenausgerichtetes Rechteck. Die östlichen und westlichen Begrenzungen stimmen mit den Meridianen zwischen den östlichen und westlichen Punktepaaren überein, während die nördlichen und südlichen Begrenzungen nicht mit den Parallelen übereinstimmen, sondern hohen elliptischen Bögen folgen, die die Kantenendpunkte enthalten. Außerdem generiert der Server immer ein gültiges Polygon, wenn dies möglich ist. Wenn ein gültiges Polygon die Datumsgrenze überschreitet, ist der Längengrad der westlichen Punkte höher als der der östlichen Punkte. Dies steht im Gegensatz zu der Beziehung zu Polygonen, die die Datumsgrenze nicht überschreiten.

Hinweis

Standardmäßig verwendet ST_Polygon das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Die folgende Anweisung gibt Polygon ((0 0, 4 0, 4 10, 0 10, 0 0)) zurück.

```
SELECT NEW ST_Polygon(NEW ST_Point(0.0, 0.0), NEW ST_Point(4.0, 10.0))
```

ST_Polygon(ST_MultiLineString[, VARCHAR(128)])-Konstruktor

Erstellt ein Polygon aus einer Mehrfachlinienfolge mit einem äußeren Ring und einer optionalen Liste von inneren Ringen.

Syntax

```
NEW ST_Polygon(multi-linestring[, polygon-format])
```

Parameter

Name	Typ	Beschreibung
multi-linestring	ST_MultiLineString	Ein Mehrfachlinienfolgenwert, der einen äußeren Ring und (optional) einer Menge von inneren Ringen enthält.
polygon-format	VARCHAR(128)	Eine Zeichenfolge mit dem Polygonformat, das beim Interpretieren der angegebenen Zeichenfolgen verwendet werden soll. Gültige Formate sind 'CounterClockwise', 'Clockwise' und 'EvenOdd'.

Bemerkungen

Erstellt ein Polygon aus einer Mehrfachlinienfolge mit einem äußeren Ring und einer optionalen Liste von inneren Ringen. Die Mehrfachlinienfolge darf nur lineare Ringe enthalten.

Wenn dieser Parameter benutzt wird, wählt der *polygon-format*-Parameter den Algorithmus aus, den der Server verwendet, um zu ermitteln, ob ein Ring ein äußerer oder innerer Ring ist. Wenn der Parameter nicht festgelegt ist, wird das Polygonformat des räumlichen Bezugssystems verwendet.

Weitere Hinweise zu *polygon-format* finden Sie unter [POLYGON FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Hinweis

Standardmäßig verwendet ST_Polygon das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 8.3.2

Beispiel

Der folgende Code gibt Polygon ((-5 -1, 5 -1, 0 9, -5 -1), (-2 0, 0 4, 2 0, -2 0)) (Rechteck mit einem rechteckigen Loch) zurück.

```
SELECT NEW ST_Polygon(
    NEW ST_MultiLineString ('MultiLineString ((-5 -1, 5 -1, 0 9, -5 -1), (-2
0, 0 4, 2 0, -2 0))'))
```

ST_Polygon(ST_LineString,...)-Konstruktor

Erstellt ein Polygon aus einer Mehrfachlinienfolge, die den äußeren Ring darstellt, und einer optionalen Liste von Linienfolgen, die innere Ringe darstellen.

Syntax

NEW ST_Polygon(*exterior-ring* [, *interior-ring1*, ..., *interior-ringN*])

Parameter

Name	Typ	Beschreibung
exterior-ring	ST_LineString	Der äußere Ring des Polygons
interior-ring1,...,interior-ringN	ST_LineString	Innere Ringe des Polygons

Bemerkungen

Erstellt ein Polygon aus einer Linienfolge, die den äußeren Ring darstellt, und einer (möglicherweise leeren) Liste von Linienfolgen, die innere Ringe darstellen. Alle angegebenen Linienfolgenwerte müssen die gleiche SRID haben. Das sich daraus ergebende Polygon wird mit dieser gemeinsamen SRID erstellt.

Die angegebenen Linienfolgen dürfen nicht leer sein und müssen dieselben Antworten für Is3D und IsMeasured aufweisen. Das Polygon ist 3D, wenn alle Linienfolgen 3D sind, und das Polygon wird gemessen, wenn alle Linienfolgen gemessen werden.

Hinweis

Standardmäßig verwendet ST_Polygon das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Standards und Kompatibilität

Die Möglichkeit für die Angabe einer Liste variabler Längen von inneren Ringen ist eine Erweiterung des Herstellers.

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.3.2

Beispiel

Der folgende Code gibt Polygon ((-5 -1, 5 -1, 0 9, -5 -1), (-2 0, 0 4, 2 0, -2 0)) (Rechteck mit einem rechteckigen Loch) zurück.

```
SELECT NEW ST_Polygon(  
    NEW ST_LineString ('LineString (-5 -1, 5 -1, 0 9, -5 -1)'),  
    NEW ST_LineString ('LineString (-2 0, 0 4, 2 0, -2 0)'))
```

ST_ExteriorRing-Methode

Ruft den äußeren Ring ab oder ändert ihn.

Überladungsliste

Name	Beschreibung
„ST_ExteriorRing()-Methode für den ST_Polygon-Datentyp“	Gibt den äußeren Ring des Polygons zurück.
„ST_ExteriorRing(ST_Curve)-Methode für den ST_Polygon-Datentyp“	Ändert den äußeren Ring des Polygons.

ST_ExteriorRing()-Methode für den ST_Polygon-Datentyp

Gibt den äußeren Ring des Polygons zurück.

Hinweis

Standardmäßig verwendet ST_ExteriorRing das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

polygon-expression.ST_ExteriorRing()

Rückgabe

- **ST_LineString** Gibt den äußeren Ring des Polygons zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *polygon-expression*.

Siehe auch

- [ST_InteriorRingN-Methode für den ST_Polygon-Datentyp](#)
- [ST_ExteriorRing-Methode für den ST_CurvePolygon-Datentyp](#)
- [ST_Boundary-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.3.3

Beispiel

Das folgende Beispiel gibt den Wert LineString (0 0, 10 0, 5 10, 0 0) zurück.

```
SELECT NEW ST_Polygon('Polygon ((0 0, 10 0, 5 10, 0 0), (3 3, 3 5, 7 5, 7 3, 3 3))')
        .ST_ExteriorRing()
```

ST_ExteriorRing(ST_Curve)-Methode für den ST_Polygon-Datentyp

Ändert den äußeren Ring des Polygons.

Hinweis

Standardmäßig verwendet ST_ExteriorRing das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

polygon-expression.ST_ExteriorRing(*curve*)

Parameter

Name	Typ	Beschreibung
curve	ST_Curve	Der neue äußere Ring des Polygons. Es muss ein linearer Ringwert sein.

Rückgabe

- ST_Polygon** Gibt eine Kopie des Polygons mit dem angegebenen äußeren Ring zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *polygon-expression*.

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006)** 8.3.3

Beispiel

Das folgende Beispiel gibt den Wert Polygon ((0 1, 10 1, 5 10, 0 1), (3 3, 3 5, 7 5, 7 3, 3 3)) zurück.

```
SELECT NEW ST_Polygon('Polygon ((0 0, 10 0, 5 10, 0 0), (3 3, 3 5, 7 5, 7 3, 3 3))')
.ST_ExteriorRing( NEW ST_LineString( 'LineString(0 1, 10 1, 5 10, 0 1)' ) )
```

ST_InteriorRingN-Methode

Gibt den *n*-ten inneren Ring im Polygon zurück.

Hinweis

Standardmäßig verwendet ST_InteriorRingN das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Syntax

polygon-expression.ST_InteriorRingN(*n*)

Parameter

Name	Typ	Beschreibung
n	INT	Die Position des zurückzugebenden Elements, von 1 bis <i>polygon-expression</i> .ST_NumInteriorRing().

Rückgabe

- **ST_LineString** Gibt den *n*-ten inneren Ring im Polygon zurück.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *polygon-expression*.

Siehe auch

- [ST_NumInteriorRing-Methode](#) für den ST_CurvePolygon-Datentyp
- [ST_ExteriorRing-Methode](#) für den ST_Polygon-Datentyp
- [ST_InteriorRingN-Methode](#) für den ST_CurvePolygon-Datentyp
- [ST_Boundary-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.3.5

Beispiel

Das folgende Beispiel gibt den Wert LineString (3 3, 3 5, 7 5, 7 3, 3 3) zurück.

```
SELECT NEW ST_Polygon('Polygon ((0 0, 10 0, 5 10, 0 0), (3 3, 3 5, 7 5, 7 3, 3 3))')
.ST_InteriorRingN( 1 )
```

ST_SpatialRefSys-Datentyp

Der ST_SpatialRefSys-Datentyp definiert Routinen für die Arbeit mit räumlichen Bezugssystemen.

Methoden

- Methoden von ST_SpatialRefSys:

ST_CompareWKT	ST_FormatTransformDefinition	ST_FormatWKT	ST_GetUnProjectedTransformDefinition
ST_ParseWKT	ST_TransformGeom	ST_World	

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 13.1

ST_CompareWKT-Methode

Vergleicht zwei Definitionen von räumlichen Bezugssystemen.

Syntax

ST_SpatialRefSys::ST_CompareWKT(*transform-definition-1*,*transform-definition-2*)

Parameter

Name	Typ	Beschreibung
transform-definition-1	LONG VAR-CHAR	Der Definitionstext des ersten räumlichen Bezugssystems
transform-definition-2	LONG VAR-CHAR	Der Definitionstext des zweiten räumlichen Bezugssystems

Rückgabe

- **BIT** Gibt 1 zurück, wenn die beiden räumlichen Bezugssysteme logisch gleichwertig sind, sonst 0.

Bemerkungen

Ermittelt, ob zwei räumliche Bezugssysteme (definiert durch WKT) logisch gleichwertig sind. Die Systeme werden als logisch gleich angesehen, wenn sie von derselben Autorität mit demselben Bezeichner definiert wurden oder die Zeichenfolgen exakt gleich sind.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Im folgenden Beispiel wird gezeigt, dass zwei räumliche Bezugssysteme als gleichwertig angesehen werden, auch wenn sie verschiedene Namen haben:

```
SELECT ST_SpatialRefSys::ST_CompareWKT(
    'GEOGCS["WGS 84",DATUM["WGS_1984",SPHEROID["WGS 84",
6378137,298.257223563,AUTHORITY["EPSG","7030"]],AUTHORITY["EPSG","6326"]],PRIMEM["Greenwich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",
0.01745329251994328,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG","4326"]]',
    'GEOGCS["WGS 84 alternate name",DATUM["WGS_1984",SPHEROID["WGS 84",
6378137,298.257223563,AUTHORITY["EPSG","7030"]],AUTHORITY["EPSG","6326"]],PRIMEM["Greenwich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",
0.01745329251994328,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG","4326"]]'
) Considered_Equal
```

Im folgenden Beispiel werden zwei räumliche Bezugssysteme gezeigt, die als nicht gleich angesehen werden, weil sie von verschiedenen Autoritäten definiert wurden:

```
SELECT ST_SpatialRefSys::ST_CompareWKT(
  'GEOGCS["WGS 84",DATUM["WGS_1984",SPHEROID["WGS 84",
6378137,298.257223563,AUTHORITY["EPSG","7030"]],AUTHORITY["EPSG","6326"]],PRIMEM["Greenwich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",
0.01745329251994328,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG","4326"]]'
,
  'GEOGCS["WGS 84",DATUM["WGS_1984",SPHEROID["WGS 84",
6378137,298.257223563,AUTHORITY["EPSG","7030"]],AUTHORITY["EPSG","6326"]],PRIMEM["Greenwich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",
0.01745329251994328,AUTHORITY["EPSG","9122"]],AUTHORITY["AnotherAuthority","4326"]]'
) Considered_NotEqual
```

ST_FormatTransformDefinition-Methode

Gibt eine formatierte Kopie der Transformationsdefinition zurück.

Syntax

ST_SpatialRefSys::ST_FormatTransformDefinition(*transform-definition*)

Parameter

Name	Typ	Beschreibung
transform-definition	LONG VARCHAR	Der Transformationsdefinitionstext des räumlichen Bezugssystems.

Rückgabe

- **LONG VARCHAR** Gibt eine Textzeichenfolge zurück, die die Transformationsdefinition festlegt

Bemerkungen

Gibt eine formatierte Kopie der Transformationsdefinition zurück.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert `+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs +towgs84=0,0,0 +no_defs` zurück.

```
SELECT ST_SpatialRefSys::ST_FormatTransformDefinition('+proj=longlat
+ellps=WGS84 +datum=WGS84 +no_defs')
```

ST_FormatWKT-Methode

Gibt eine formatierte Kopie der WKT-Definition zurück.

Syntax

ST_SpatialRefSys::ST_FormatWKT(*definition*)

Parameter

Name	Typ	Beschreibung
definition	LONG VARCHAR	Der Definitionstext des räumlichen Bezugssystems

Rückgabe

- **LONG VARCHAR** Gibt eine Textzeichenfolge zurück, die das räumliche Bezugssystem in WKT definiert.

Bemerkungen

Gibt eine formatierte Kopie der WKT-Definition des räumlichen Bezugssystems zurück.

Der Well-Known-Text-Wert, der räumliche Bezugssysteme beschreibt, hat ein anderes Format als der Well-Known-Text-Wert, der Geometrien beschreibt.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert GEOGCS["WGS 84" , DATUM["WGS_1984" , SPHEROID["WGS 84" , 6378137 , 298.257223563 , AUTHORITY["EPSG" , "7030"]] , AUTHORITY["EPSG" , "6326"]] , PRIMEM["Greenwich" , 0 , AUTHORITY["EPSG" , "8901"]] , UNIT["degree" , 0.01745329251994328 , AUTHORITY["EPSG" , "9122"]] , AUTHORITY["EPSG" , "4326"]] zurück.

```
SELECT ST_SpatialRefSys::ST_FormatWKT('GEOGCS[ "WGS
84" , DATUM[ "WGS_1984" , SPHEROID[ "WGS 84" ,
6378137 , 298.257223563 , AUTHORITY[ "EPSG" , "7030" ] ] , AUTHORITY[ "EPSG" , "6326" ] ] , PRI
MEM[ "Greenwich" , 0 , AUTHORITY[ "EPSG" , "8901" ] ] , UNIT[ "degree" ,
0.01745329251994328 , AUTHORITY[ "EPSG" , "9122" ] ] , AUTHORITY[ "EPSG" , "4326" ] ] ' )
```

ST_GetUnProjectedTransformDefinition-Methode

Gibt die Transformationsdefinition des räumlichen Bezugssystems zurück, das die Quelle der Projektion ist.

Syntax

```
ST_SpatialRefSys::ST_GetUnProjectedTransformDefinition(transform-definition)
```

Parameter

Name	Typ	Beschreibung
transform-definition	LONG VAR-CHAR	Der Transformationsdefinitionstext des räumlichen Bezugssystems.

Rückgabe

- **LONG VARCHAR** Gibt eine Textzeichenfolge zurück, die die Transformationsdefinition des nicht projizierten räumlichen Bezugssystems festlegt.

Bemerkungen

Wenn der *transform-definition*-Parameter ein projiziertes räumliches Bezugssystem definiert, wird damit die Definition des räumlichen Quellbezugssystems zurückgegeben.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert `+proj=latlong +a=6371000 +b=6371000 +no_defs` zurück.

```
SELECT ST_SpatialRefSys::ST_GetUnProjectedTransformDefinition( '+proj=robin  
+lon_0=0 +x_0=0 +y_0=0 +a=6371000 +b=6371000 +units=m no_defs' )
```

ST_ParseWKT-Methode

Ruft ein benanntes Element aus der Well-Known-Text-Definition (WKT) eines räumlichen Bezugssystems ab.

Syntax

ST_SpatialRefSys::ST_ParseWKT(*element*,*srs-text*)

Parameter

Name	Typ	Beschreibung
element	VAR-CHAR(128)	<p>Das Element, das aus WKT abgerufen werden soll. Die folgenden benannten Elemente können abgerufen werden:</p> <ul style="list-style-type: none">• srs_name Der Name des räumlichen Bezugssystems• srs_type Der Typ des Koordinatensystems.• organization Der Name der Organisation, die das räumliche Bezugssystem definiert.• organization_id Die Ganzzahl-ID, die von der Organisation zugewiesen wurde, die das räumliche Bezugssystem definiert.• linear_unit_of_measure Der Name der linearen Maßeinheit.• linear_unit_of_measure Der Konvertierungsfaktor für die lineare Maßeinheit.• angular_unit_of_measure Der Name der Winkelmaßeinheit.• angular_unit_of_measure Der Konvertierungsfaktor für die Winkelmaßeinheit.
srs-text	LONG VAR-CHAR	Der Definitionstext des räumlichen Bezugssystems

Rückgabe

- **LONG VARCHAR** Ruft ein benanntes Element aus der WKT-Definition eines räumlichen Bezugssystems ab.

Bemerkungen

Ruft ein benanntes Element aus der WKT-Definition eines räumlichen Bezugssystems ab. Wenn das benannte Element von WKT nicht definiert ist, wird NULL zurückgegeben.

Der Well-Known-Text-Wert, der räumliche Bezugssysteme beschreibt, hat ein anderes Format als der Well-Known-Text-Wert, der Geometrien beschreibt.

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt eine Ergebnismenge mit einer Zeile für jedes der benannten Elemente zurück.

```
with V(element,srs_text) as (  
  SELECT row_value as element, 'GEOGCS["WGS  
84",DATUM["WGS_1984",SPHEROID["WGS 84",
```

```

6378137,298.257223563,AUTHORITY["EPSG","7030"]],AUTHORITY["EPSG","6326"]],PR
MEM["Greenwich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",
0.01745329251994328,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG","4326"]]' as
srs_text
FROM
sa_split_list('srs_name,srs_type,organization,organization_id,linear_unit_of_
measure,linear_unit_of_measure_factor,angular_unit_of_measure,angular_unit_of_
_measure_factor') D
)
SELECT element, ST_SpatialRefSys::ST_ParseWKT( element, srs_text ) parsed
FROM V

```

Die Abfrage erzeugt die folgende Ergebnismenge:

element	parsed
srs_name	WGS 84
srs_type	GEOGRAPHIC
organization	EPSG
organization_id	4326
linear_unit_of_measure	NULL
linear_unit_of_measure_factor	NULL
angular_unit_of_measure	degree
angular_unit_of_measure_factor	.017453292519943282

ST_TransformGeom-Methode

Gibt die mit der angegebenen Transformationsdefinition umgewandelte Geometrie zurück.

Syntax

```
ST_SpatialRefSys::ST_TransformGeom(geom,target-transform-definition[, source-transform-definition])
```

Parameter

Name	Typ	Beschreibung
geom	ST_Geometry	Die umzuwandelnde Geometrie
target-trans- form-definition	LONG VAR- CHAR	Der Transformationsdefinitionstext für das räumliche Zielbezugs- system

Name	Typ	Beschreibung
source-trans-form-definition	LONG VAR-CHAR	Der Transformationsdefinitionstext für das räumliche Quellbezugssystem. Wenn dieser Parameter nicht eingegeben wird, wird die Transformationsdefinition aus dem räumlichen Bezugssystem des <i>geom</i> -Parameters verwendet.

Rückgabe

- **ST_Geometry** Gibt die mit der angegebenen Transformationsdefinition umgewandelte Eingangsgeometrie zurück.

Der Bezeichner des räumlichen Bezugssystems des Ergebnisses ist *sa_planar_unbounded* (mit SRID 2147483646).

Bemerkungen

Die *ST_TransformGeom*-Methode wandelt eine einzelne Geometrie mit der gegebenen Transformationsdefinition des Ziels um. Die Umwandlung erfolgt mit der Bibliothek PROJ.4 Diese Methode kann in ausgewählten Situationen verwendet werden, wenn die entsprechenden räumlichen Bezugssysteme in der Datenbank noch nicht erstellt wurden. Wenn die entsprechenden räumlichen Bezugssysteme verfügbar sind, ist die *ST_Transform*-Methode oft besser geeignet.

Umwandlungen aus einem Längengrad-/Breitengradsystem in ein kartesisches System können bei Polarpunkten problematisch sein. Wenn der Datenbankserver einen Punkt nahe dem Nord- oder Südpol nicht umwandeln kann, verschiebt sich der Breitengrad des Punktes geringfügig (kaum mehr als 1e-10 rad (Bogenmaß) weg vom Pol und entlang desselben Längengrads, damit die Umwandlung erfolgreich verläuft.

Siehe auch

- [ST_Transform-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert *Point (-5387692.968586 4763459.253243)* zurück.

```
SELECT ST_SpatialRefSys::ST_TransformGeom( NEW ST_Point(-63.57,44.65,4326),
'+proj=robin +lon_0=0 +x_0=0 +y_0=0 +a=6371000 +b=6371000 +units=m
no_defs' ).ST_AsText('DecimalDigits=6')
```

ST_World-Methode

Gibt eine Geometrie zurück, die alle Punkte des räumlichen Bezugssystems darstellt.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Syntax

ST_SpatialRefSys::ST_World(*srid*)

Parameter

Name	Typ	Beschreibung
srid	INT	Die für das Ergebnis zu verwendende SRID.

Rückgabe

- **ST_Surface** Gibt eine Geometrie zurück, die alle Punkte im räumlichen Bezugssystem darstellt, das durch den *srid*-Parameter identifiziert wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Siehe auch

- [ST_IsWorld-Methode für den ST_Surface-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** Erweiterung des Herstellers

Beispiel

Das folgende Beispiel gibt den Wert Polygon ((-1000000 -1000000, 1000000 -1000000, 1000000 1000000, -1000000 1000000, -1000000 -1000000)) zurück.

```
SELECT ST_SpatialRefSys::ST_World(0)
```

ST_Surface-Datentyp

Der ST_Surface-Datentyp ist ein übergeordneter Datentyp für 2-dimensionale Geometriedatentypen. Der ST_Surface-Datentyp ist nicht instanzierbar.

Direkt übergeordneter Typ

- „ST_Geometry-Datentyp“

Direkt untergeordnete Typen

- „ST_CurvePolygon-Datentyp“

Methoden

- Methoden von ST_Surface:

ST_Area	ST_Centroid	ST_IsWorld	ST_Perimeter
ST_PointOnSurface			

- Alle Methoden von „ST_Geometry-Datentyp“ können auch für einen ST_Surface-Datentyp aufgerufen werden:

ST_Affine	ST_AsBinary	ST_AsGML	ST_AsGeoJSON
ST_AsKML	ST_AsSVG	ST_AsSVGAgr	ST_AsText
ST_AsWKB	ST_AsWKT	ST_AsXML	ST_Boundary
ST_Contains	ST_ContainsFilter	ST_ConvexHull	ST_ConvexHullAggr
ST_CoordDim	ST_CoveredBy	ST_CoveredByFilter	ST_Covers
ST_CoversFilter	ST_Crosses	ST_Difference	ST_Dimension
ST_Disjoint	ST_Distance	ST_Envelope	ST_EnvelopeAggr
ST_Equals	ST_EqualsFilter	ST_GeomFromBinary	ST_GeomFromShape
ST_GeomFromText	ST_GeomFromWKB	ST_GeomFromWKT	ST_GeometryType
ST_GeometryTypeFromBaseType	ST_Intersection	ST_IntersectionAggr	ST_Intersects
ST_IntersectsFilter	ST_IntersectsRect	ST_Is3D	ST_IsEmpty
ST_IsMeasured	ST_IsSimple	ST_IsValid	ST_LatNorth
ST_LatSouth	ST_LinearHash	ST_LinearUnHash	ST_LoadConfigurationData
ST_LongEast	ST_LongWest	ST_MMax	ST_MMin
ST_OrderingEquals	ST_Overlaps	ST_Relate	ST_Reverse
ST_SRID	ST_SRIDFromBaseType	ST_SnapToGrid	ST_SymDifference
ST_ToCircular	ST_ToCompound	ST_ToCurve	ST_ToCurvePoly
ST_ToGeomColl	ST_ToLineString	ST_ToMultiCurve	ST_ToMultiLine

ST_ToMultiPoint	ST_ToMultiPolygon	ST_ToMultiSurface	ST_ToPoint
ST_ToPolygon	ST_ToSurface	ST_Touches	ST_Transform
ST_Union	ST_UnionAggr	ST_Within	ST_WithinDistance
ST_WithinDistanceFilter	ST_WithinFilter	ST_XMax	ST_XMin
ST_YMax	ST_YMin	ST_ZMax	ST_ZMin

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.1

ST_Area-Methode

Berechnet den Bereich einer Fläche in den angegebenen Einheiten.

Syntax

surface-expression.**ST_Area**([*unit-name*])

Parameter

Name	Typ	Beschreibung
unit-name	VAR-CHAR(128)	Die Einheiten, in denen die Länge berechnet werden soll. Ist standardmäßig die Einheit des räumlichen Bezugssystems. Der unit-name-Parameter muss mit der UNIT_NAME-Spalte einer Zeile in der ST_UNITS_OF_MEASURE-Ansicht übereinstimmen, wo UNIT_TYPE auf 'LINEAR' eingestellt ist.

Rückgabe

- **DOUBLE** Gibt den Bereich der Fläche zurück.

Bemerkungen

Die ST_Area-Methode berechnet den Bereich einer Fläche. Die für die Darstellung des Bereichs verwendeten Einheiten basieren auf der angegebenen linearen Maßeinheit. Beispiel: Wenn die angegebene lineare Maßeinheit Fuß ist, lautet die Einheit für den Bereich Quadratfuß.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

- [ST_Perimeter-Methode](#) für den [ST_Surface-Datentyp](#)
- [ST_Area-Methode](#) für den [ST_MultiSurface-Datentyp](#)
- [ST_Length-Methode](#) für den [ST_Curve-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.1.2

Beispiel

Das folgende Beispiel gibt den Wert 12.5 zurück.

```
SELECT TREAT( Shape AS ST_Polygon ).ST_Area()  
FROM SpatialShapes WHERE ShapeID = 22
```

Der folgende Code gibt den Bereich der `poly_geometry`-Spalte in Quadratmeilen aus der fiktiven `region`-Tabelle zurück.

```
SELECT name, poly_geometry.ST_Area( 'Statute Mile' )  
FROM region
```

ST_Centroid-Methode

Gibt den `ST_Point`-Wert zurück, der den mathematischen Schwerpunkt des Flächenwerts darstellt.

Syntax

surface-expression.**ST_Centroid**()

Rückgabe

- **ST_Point** Wenn die Fläche die leere Menge ist, wird `NULL` zurückgegeben. Andernfalls wird der mathematische Schwerpunkt der Fläche zurückgegeben.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *surface-expression*.

Bemerkungen

Gibt den `ST_Point`-Wert zurück, der den mathematischen Schwerpunkt des Flächenwerts darstellt. Dieser Punkt muss nicht unbedingt ein Punkt auf der Oberfläche sein.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Siehe auch

- [ST_Centroid-Methode](#) für den [ST_MultiSurface-Datentyp](#)
- [ST_PointOnSurface-Methode](#) für den [ST_Surface-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.1.4

Beispiel

Das folgende Beispiel gibt den Wert `Point (5 4.666667)` zurück.

```
SELECT TREAT( Shape as ST_Surface ).ST_Centroid()
FROM SpatialShapes WHERE ShapeID = 22
```

ST_IsWorld-Methode

Testet, ob ST_Surface den gesamten Raum abdeckt.

Hinweis

Diese Methode kann mit Geometrien in räumlichen Bezugssystemen mit gewölbter Erddarstellung nicht verwendet werden.

Syntax

surface-expression.**ST_IsWorld()**

Rückgabe

- **BIT** Gibt 1 zurück, wenn die Fläche den gesamten Bereich abdeckt, sonst 0.

Siehe auch

- [ST_World-Methode für den ST_SpatialRefSys-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.1.6

Beispiel

Das folgende Beispiel gibt den Wert 1 zurück.

```
SELECT NEW ST_Polygon( NEW ST_Point( -180, -90, 1000004326 ),
                      NEW ST_Point( 180, 90, 1000004326 ) ).ST_IsWorld()
```

ST_Perimeter-Methode

Berechnet den Umfang einer Fläche in den angegebenen Einheiten.

Syntax

surface-expression.**ST_Perimeter**([*unit-name*])

Parameter

Name	Typ	Beschreibung
unit-name	VAR-CHAR(128)	Die Einheiten, in denen die Länge berechnet werden soll. Ist standardmäßig die Einheit des räumlichen Bezugssystems. Der unit-name-Parameter muss mit der UNIT_NAME-Spalte einer Zeile in der ST_UNITS_OF_MEASURE-Ansicht übereinstimmen, wo UNIT_TYPE auf 'LINEAR' eingestellt ist.

Rückgabe

- **DOUBLE** Gibt den Umfang der Fläche in der angegebenen Maßeinheit zurück.

Bemerkungen

Die ST_Perimeter-Methode gibt die Länge des Umfangs einer Fläche in den Einheiten zurück, die durch den *unit_name*-Parameter festgelegt sind. Wenn die Fläche leer ist, wird NULL zurückgegeben.

Wenn die Fläche Z-Werte enthält, werden diese beim Berechnen des Umfangs der Geometrie nicht in Betracht gezogen.

Der Umfang eines Polygons enthält die Länge aller Ringe (äußere und innere).

Hinweis

Wenn *surface-expression* eine leere Geometrie (ST_IsEmpty()=1) ist, gibt diese Methode NULL zurück.

Hinweis

Standardmäßig verwendet ST_Perimeter das Originalformat für eine Geometrie, wenn es verfügbar ist. Andernfalls wird das interne Format verwendet. Weitere Hinweise zu den internen Formaten und Originalformaten finden Sie unter [STORAGE FORMAT-Klausel](#), [CREATE SPATIAL REFERENCE SYSTEM-Anweisung](#) [*SQL Anywhere Server - SQL-Referenzhandbuch*].

Siehe auch

- [ST_Perimeter-Methode für den ST_MultiSurface-Datentyp](#)
- [ST_Boundary-Methode für den ST_Geometry-Datentyp](#)
- [ST_Length-Methode für den ST_Curve-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.1.3

Beispiel

Das folgende Beispiel gibt den Wert 18 zurück.

```
SELECT TREAT( Shape as ST_Surface ).ST_Perimeter()  
FROM SpatialShapes WHERE ShapeID = 3
```

Der folgende Code gibt den Umfang der `poly_geometry`-Spalte in Meilen aus der fiktiven `region`-Tabelle zurück.

```
SELECT name, poly_geometry.ST_Perimeter( 'Statute Mile' )
FROM region
```

ST_PointOnSurface-Methode

Gibt einen `ST_Point`-Wert zurück, der garantiert eine räumliche Schnittmenge mit dem `ST_Surface`-Wert aufweist.

Hinweis

Wenn *surface-expression* Kreisbogenfolgen enthält, werden diese in Linienfolgen interpoliert.

Syntax

surface-expression.**ST_PointOnSurface**()

Rückgabe

- **ST_Point** Wenn die Fläche die leere Menge ist, wird `NULL` zurückgegeben. Andernfalls wird ein `ST_Point`-Wert zurückgegeben, bei dem garantiert wird, dass er räumlich eine Schnittmenge mit dem `ST_Surface`-Wert hat.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis ist der gleiche wie beim räumlichen Bezugssystem von *surface-expression*.

Siehe auch

- [ST_PointOnSurface-Methode für den ST_MultiSurface-Datentyp](#)
- [ST_Centroid-Methode für den ST_Surface-Datentyp](#)
- [ST_Intersects-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.1.5

Beispiel

Der folgende Code gibt einen Punkt zurück, der das Polygon schneidet.

```
SELECT NEW ST_Polygon( 'Polygon(( 1 0, 0 10, 1 1, 2 10, 1 0 ))' )
.ST_PointOnSurface()
```

Funktionen der räumlichen Kompatibilität

Der SQL/MM-Standard definiert eine Anzahl von Funktionen, die für räumliche Operationen verwendet werden können. In den meisten Fällen werden durch diese Funktionen die Funktionen von Methoden oder Konstruktoren von räumlichen Datentypen nachgebildet.

Funktionen

Name	Beschreibung
„ST_BdMPolyFromText-Funktion [Räumlich]“	Gibt einen ST_MultiPolygon-Wert zurück, der aus der Well-Known-Text-Darstellung (WKT) einer Mehrfachlinienfolge erstellt wird.
„ST_BdMPolyFromWKB-Funktion [Räumlich]“	Gibt einen ST_MultiPolygon-Wert zurück, der aus der Well-Known-Binary-Darstellung (WKB) einer Mehrfachlinienfolge erstellt wird.
„ST_BdPolyFromText-Funktion [Räumlich]“	Gibt einen ST_Polygon-Wert zurück, der aus der Well-Known-Text-Darstellung (WKT) einer Mehrfachlinienfolge erstellt wird.
„ST_BdPolyFromWKB-Funktion [Räumlich]“	Gibt einen ST_Polygon-Wert zurück, der aus der Well-Known-Binary-Darstellung (WKB) einer Mehrfachlinienfolge erstellt wird.
„ST_CPolyFromText-Funktion [Räumlich]“	Gibt einen ST_CurvePolygon-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_CurvePolygon-Elements enthält.
„ST_CPolyFromWKB-Funktion [Räumlich]“	Gibt einen ST_CurvePolygon-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_CurvePolygon-Elements enthält.
„ST_CircularFromTxt-Funktion [Räumlich]“	Gibt einen ST_CircularString-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_CircularString-Elements enthält.
„ST_CircularFromWKB-Funktion [Räumlich]“	Gibt einen ST_CircularString-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_CircularString-Elements enthält.
„ST_CompoundFromTxt-Funktion [Räumlich]“	Gibt einen ST_CompoundCurve-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_CompoundCurve-Elements enthält.
„ST_CompoundFromWKB-Funktion [Räumlich]“	Gibt einen ST_CompoundCurve-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_CompoundCurve-Elements enthält.
„ST_GeomCollFromTxt-Funktion [Räumlich]“	Gibt einen ST_GeomCollection-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_GeomCollection-Elements enthält.
„ST_GeomCollFromWKB-Funktion [Räumlich]“	Gibt einen ST_GeomCollection-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_GeomCollection-Elements enthält.

Name	Beschreibung
„ST_GeomFromText-Funktion [Räumlich]“	Gibt einen ST_Geometry-Wert zurück, der aus einem LONG VAR-CHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_Geometry-Elements enthält.
„ST_GeomFromWKB-Funktion [Räumlich]“	Gibt einen ST_Geometry-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_Geometry-Elements enthält.
„ST_LineFromText-Funktion [Räumlich]“	Gibt einen ST_LineString-Wert zurück, der aus einem LONG VAR-CHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_LineString-Elements enthält.
„ST_LineFromWKB-Funktion [Räumlich]“	Gibt einen ST_LineString-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_LineString-Elements enthält.
„ST_MCurveFromText-Funktion [Räumlich]“	Gibt einen ST_MultiCurve-Wert zurück, der aus einem LONG VAR-CHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_MultiCurve-Elements enthält.
„ST_MCurveFromWKB-Funktion [Räumlich]“	Gibt einen ST_MultiCurve-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_MultiCurve-Elements enthält.
„ST_MLineFromText-Funktion [Räumlich]“	Gibt einen ST_MultiLineString-Wert zurück, der aus einem LONG VAR-CHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_MultiLineString-Elements enthält.
„ST_MLineFromWKB-Funktion [Räumlich]“	Gibt einen ST_MultiLineString-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_MultiLineString-Elements enthält.
„ST_MPointFromText-Funktion [Räumlich]“	Gibt einen ST_MultiPoint-Wert zurück, der aus einem LONG VAR-CHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_MultiPoint-Elements enthält.
„ST_MPointFromWKB-Funktion [Räumlich]“	Gibt einen ST_MultiPoint-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_MultiPoint-Elements enthält.
„ST_MPolyFromText-Funktion [Räumlich]“	Gibt einen ST_MultiPolygon-Wert zurück, der aus einem LONG VAR-CHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_MultiPolygon-Elements enthält.

Name	Beschreibung
„ST_MPolyFromWKB-Funktion [Räumlich]“	Gibt einen ST_MultiPolygon-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_MultiPolygon-Elements enthält.
„ST_MSurfaceFromTxt-Funktion [Räumlich]“	Gibt einen ST_MultiSurface-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_MultiSurface-Elements enthält.
„ST_MSurfaceFromWKB-Funktion [Räumlich]“	Gibt einen ST_MultiSurface-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_MultiSurface-Elements enthält.
„ST_OrderingEquals-Funktion [Räumlich]“	Testet, ob eine Geometrie identisch mit einer anderen Geometrie ist.
„ST_PointFromText-Funktion [Räumlich]“	Gibt einen ST_Point-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_Point-Elements enthält.
„ST_PointFromWKB-Funktion [Räumlich]“	Gibt einen ST_Point-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_Point-Elements enthält.
„ST_PolyFromText-Funktion [Räumlich]“	Gibt einen ST_Polygon-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_Polygon-Elements enthält.
„ST_PolyFromWKB-Funktion [Räumlich]“	Gibt einen ST_Polygon-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_Polygon-Elements enthält.

ST_BdMPolyFromText-Funktion [Räumlich]

Gibt einen ST_MultiPolygon-Wert zurück, der aus der Well-Known-Text-Darstellung (WKT) einer Mehrfachlinienfolge erstellt wird.

Syntax

```
[DBO.]ST_BdMPolyFromText(wkt[, srid])
```

Parameter

Name	Typ	Beschreibung
wkt	LONG VARCHAR	Die WKT-Darstellung eines Mehrfachlinienfolgenwerts.

Name	Typ	Beschreibung
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_MultiPolygon** Gibt einen ST_MultiPolygon-Wert zurück, der aus der WKT-Darstellung einer Mehrfachlinienfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_BdMPolyFromText-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_BdMPolyFromText( awkt LONG VARCHAR, srid INT DEFAULT
0 )
RETURNS ST_MultiPolygon
BEGIN
    DECLARE mls ST_MultiLineString;
    SET mls = NEW ST_MultiLineString( awkt, srid );
    RETURN NEW ST_MultiPolygon( mls );
END
```

Hinweis

Die ST_BdMPolyFromText-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_Polygon-Konstruktor“

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 9.6.7

ST_BdMPolyFromWKB-Funktion [Räumlich]

Gibt einen ST_MultiPolygon-Wert zurück, der aus der Well-Known-Binary-Darstellung (WKB) einer Mehrfachlinienfolge erstellt wird.

Syntax

```
[DBO.]ST_BdMPolyFromWKB(wkb[, srid])
```

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung eines Mehrfachlinienfolgenwerts.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_MultiPolygon** Gibt einen ST_MultiPolygon-Wert zurück, der aus der WKB-Darstellung einer Mehrfachlinienfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_BdMPolyFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_BdMPolyFromWKB( awkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_MultiPolygon
BEGIN
    DECLARE mls ST_MultiLineString;
    SET mls = NEW ST_MultiLineString( awkb, srid );
    RETURN NEW ST_MultiPolygon( mls );
END
```

Hinweis

Die ST_BdMPolyFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_Polygon-Konstruktor“

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.6.8

ST_BdPolyFromText-Funktion [Räumlich]

Gibt einen ST_Polygon-Wert zurück, der aus der Well-Known-Text-Darstellung (WKT) einer Mehrfachlinienfolge erstellt wird.

Syntax

[DBO.]ST_BdPolyFromText(*wkt*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkt	LONG VARCHAR	Die WKT-Darstellung eines Mehrfachlinienfolgenwerts.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_Polygon** Gibt einen ST_Polygon-Wert zurück, der aus der WKT-Darstellung einer Mehrfachlinienfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_BdPolyFromText-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_BdPolyFromText( awkt LONG VARCHAR, srid INT DEFAULT
0 )
RETURNS ST_Polygon
BEGIN
    DECLARE mls ST_MultiLineString;
    SET mls = NEW ST_MultiLineString( awkt, srid );
    RETURN NEW ST_Polygon( mls );
END
```

Hinweis

Die ST_BdPolyFromText-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_Polygon-Konstruktor“

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.3.9

ST_BdPolyFromWKB-Funktion [Räumlich]

Gibt einen ST_Polygon-Wert zurück, der aus der Well-Known-Binary-Darstellung (WKB) einer Mehrfachlinienfolge erstellt wird.

Syntax

[DBO.]ST_BdPolyFromWKB(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung eines Mehrfachlinienfolgenwerts.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_Polygon** Gibt einen ST_Polygon-Wert zurück, der aus der WKB-Darstellung einer Mehrfachlinienfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_BdPolyFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_BdPolyFromWKB( awkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_Polygon
BEGIN
    DECLARE mls ST_MultiLineString;
    SET mls = NEW ST_MultiLineString( awkb, srid );
    RETURN NEW ST_Polygon( mls );
END
```

Hinweis

Die ST_BdPolyFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_Polygon-Konstruktor“

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 8.3.10

ST_CPolyFromText-Funktion [Räumlich]

Gibt einen ST_CurvePolygon-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_CurvePolygon-Elements enthält.

Syntax

[DBO.]ST_CPolyFromText(*wkt*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkt	LONG VARCHAR	Die WKT-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_CurvePolygon** Gibt einen ST_CurvePolygon-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_CPolyFromText-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_CPolyFromText( wkt LONG VARCHAR, srid INT DEFAULT 0 )
RETURNS ST_CurvePolygon
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromText( wkt, srid );
    RETURN CAST( geo AS ST_CurvePolygon);
END
```

Hinweis

Die ST_CPolyFromText-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_CurvePolygon-Konstruktor“
- ST_GeomFromText-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 8.2.8

ST_CPolyFromWKB-Funktion [Räumlich]

Gibt einen ST_CurvePolygon-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_CurvePolygon-Elements enthält.

Syntax

[DBO.]ST_CPolyFromWKB(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_CurvePolygon** Gibt einen ST_CurvePolygon-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_CPolyFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_CPolyFromWKB( wkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_CurvePolygon
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromWKB( wkb, srid );
    RETURN CAST( geo AS ST_CurvePolygon);
END
```

Hinweis

Die ST_CPolyFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_CurvePolygon-Konstruktor“
- ST_GeomFromWKB-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 8.2.9

ST_CircularFromTxt-Funktion [Räumlich]

Gibt einen ST_CircularString-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_CircularString-Elements enthält.

Syntax

`[DBO.]ST_CircularFromTxt(wkt[, srid])`

Parameter

Name	Typ	Beschreibung
wkt	LONG VAR-CHAR	Die WKT-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_CircularString** Gibt einen ST_CircularString-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_CircularFromTxt-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_CircularFromTxt( wkt LONG VARCHAR, srid INT DEFAULT
0 )
RETURNS ST_CircularString
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromText( wkt, srid );
    RETURN CAST( geo AS ST_CircularString);
END
```

Hinweis

Die ST_CircularFromTxt-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die `sa_install_feature`-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „[sa_install_feature-Systemprozedur](#)“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_CircularString-Konstruktor“
- [ST_GeomFromText-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.3.9

ST_CircularFromWKB-Funktion [Räumlich]

Gibt einen ST_CircularString-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_CircularString-Elements enthält.

Syntax

[DBO.]ST_CircularFromWKB(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_CircularString** Gibt einen ST_CircularString-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_CircularFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_CircularFromWKB( wkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_CircularString
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromWKB( wkb, srid );
    RETURN CAST( geo AS ST_CircularString);
END
```

Hinweis
Die ST_CircularFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [\[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Siehe auch

- „ST_CircularString-Konstruktor“
- ST_GeomFromWKB-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.3.10

ST_CompoundFromTxt-Funktion [Räumlich]

Gibt einen ST_CompoundCurve-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_CompoundCurve-Elements enthält.

Syntax

[DBO.]ST_CompoundFromTxt(*wkt*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkt	LONG VARCHAR	Die WKT-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_CompoundCurve** Gibt einen ST_CompoundCurve-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_CompoundFromTxt-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_CompoundFromTxt( wkt LONG VARCHAR, srid INT DEFAULT
0 )
RETURNS ST_CompoundCurve
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromText( wkt, srid );
    RETURN CAST( geo AS ST_CompoundCurve);
END
```

Hinweis

Die ST_CompoundFromTxt-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_CompoundCurve-Konstruktor“
- [ST_GeomFromText-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.4.8

ST_CompoundFromWKB-Funktion [Räumlich]

Gibt einen ST_CompoundCurve-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_CompoundCurve-Elements enthält.

Syntax

[DBO.]ST_CompoundFromWKB(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_CompoundCurve** Gibt einen ST_CompoundCurve-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_CompoundFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_CompoundFromWKB( wkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_CompoundCurve
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromWKB( wkb, srid );
    RETURN CAST( geo AS ST_CompoundCurve );
END
```

Hinweis

Die ST_CompoundFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die `sa_install_feature`-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „[sa_install_feature-Systemprozedur](#)“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_CompoundCurve-Konstruktor“
- ST_GeomFromWKB-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.4.9

ST_GeomCollFromTxt-Funktion [Räumlich]

Gibt einen ST_GeomCollection-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_GeomCollection-Elements enthält.

Syntax

[DBO.]ST_GeomCollFromTxt(*wkt*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkt	LONG VARCHAR	Die WKT-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_GeomCollection** Gibt einen ST_GeomCollection-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_GeomCollFromTxt-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_GeomCollFromTxt( wkt LONG VARCHAR, srid INT DEFAULT
0 )
RETURNS ST_GeomCollection
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromText( wkt, srid );
    RETURN CAST( geo AS ST_GeomCollection);
END
```

Hinweis

Die ST_GeomCollFromTxt-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „[sa_install_feature-Systemprozedur](#)“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_GeomCollection-Konstruktor“
- ST_GeomFromText-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 9.1.6

ST_GeomCollFromWKB-Funktion [Räumlich]

Gibt einen ST_GeomCollection-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_GeomCollection-Elements enthält.

Syntax

[DBO.]ST_GeomCollFromWKB(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_GeomCollection** Gibt einen ST_GeomCollection-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_GeomCollFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_GeomCollFromWKB( wkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_GeomCollection
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromWKB( wkb, srid );
    RETURN CAST( geo AS ST_GeomCollection);
END
```

Hinweis

Die ST_GeomCollFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_GeomCollection-Konstruktor“
- ST_GeomFromWKB-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 9.1.7

ST_GeomFromText-Funktion [Räumlich]

Gibt einen ST_Geometry-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_Geometry-Elements enthält.

Syntax

[DBO.]ST_GeomFromText(*wkt* [, *srid*])

Parameter

Name	Typ	Beschreibung
wkt	LONG VARCHAR	Die WKT-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_Geometry** Gibt einen ST_Geometry-Wert zurück, der aus der Eingabezeichenfolge erstellt wurde.

Die räumliche Referenz-ID des Ergebnisses ist die vom Parameter *srid* angegebene ID.

Bemerkungen

Die ST_GeomFromText-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_GeomFromText( wkt LONG VARCHAR, srid INT DEFAULT 0 )
RETURNS ST_Geometry
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromText( wkt, srid );
    RETURN CAST( geo AS ST_Geometry);
END
```

Hinweis

Die ST_GeomFromText-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- [ST_GeomFromText-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.40

ST_GeomFromWKB-Funktion [Räumlich]

Gibt einen ST_Geometry-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_Geometry-Elements enthält.

Syntax

[DBO.]ST_GeomFromWKB(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_Geometry** Gibt einen ST_Geometry-Wert zurück, der aus der Eingabezeichenfolge erstellt wurde.

Die räumliche Referenz-ID des Ergebnisses ist die vom Parameter *srid* angegebene ID.

Bemerkungen

Die ST_GeomFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_GeomFromWKB( wkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_Geometry
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromWKB( wkb, srid );
    RETURN CAST( geo AS ST_Geometry );
END
```

Hinweis

Die ST_GeomFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die `sa_install_feature`-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „[sa_install_feature-Systemprozedur](#)“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- [ST_GeomFromWKB-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.41

ST_LineFromText-Funktion [Räumlich]

Gibt einen ST_LineString-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_LineString-Elements enthält.

Syntax

[DBO.]**ST_LineFromText**(*wkt*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkt	LONG VARCHAR	Die WKT-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_LineString** Gibt einen ST_LineString-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_LineFromText-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_LineFromText( wkt LONG VARCHAR, srid INT DEFAULT 0 )
RETURNS ST_LineString
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromText( wkt, srid );
    RETURN CAST( geo AS ST_LineString);
END
```

Hinweis

Die ST_LineFromText-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_LineString-Konstruktor“
- ST_GeomFromText-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 7.2.8

ST_LineFromWKB-Funktion [Räumlich]

Gibt einen ST_LineString-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_LineString-Elements enthält.

Syntax

[DBO.]ST_LineFromWKB(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_LineString** Gibt einen ST_LineString-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_LineFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_LineFromWKB( wkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_LineString
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromWKB( wkb, srid );
    RETURN CAST( geo AS ST_LineString);
END
```

Hinweis

Die ST_LineFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_LineString-Konstruktor“
- [ST_GeomFromWKB-Methode](#) für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.2.9

ST_MCurveFromText-Funktion [Räumlich]

Gibt einen ST_MultiCurve-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_MultiCurve-Elements enthält.

Syntax

[DBO.]ST_MCurveFromText(*wkt*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkt	LONG VARCHAR	Die WKT-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_MultiCurve** Gibt einen ST_MultiCurve-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_MCurveFromText-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_MCurveFromText( wkt LONG VARCHAR, srid INT DEFAULT 0 )
RETURNS ST_MultiCurve
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromText( wkt, srid );
    RETURN CAST( geo AS ST_MultiCurve);
END
```

Hinweis

Die ST_MCurveFromText-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_MultiCurve-Konstruktor“
- ST_GeomFromText-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 9.3.6

ST_MCurveFromWKB-Funktion [Räumlich]

Gibt einen ST_MultiCurve-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_MultiCurve-Elements enthält.

Syntax

[DBO.]ST_MCurveFromWKB(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_MultiCurve** Gibt einen ST_MultiCurve-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_MCurveFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_MCurveFromWKB( wkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_MultiCurve
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromWKB( wkb, srid );
    RETURN CAST( geo AS ST_MultiCurve);
END
```

Hinweis

Die ST_MCurveFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_MultiCurve-Konstruktor“
- ST_GeomFromWKB-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 9.3.7

ST_MLineFromText-Funktion [Räumlich]

Gibt einen ST_MultiLineString-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_MultiLineString-Elements enthält.

Syntax

[DBO.]ST_MLineFromText(*wkt*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkt	LONG VARCHAR	Die WKT-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_MultiLineString** Gibt einen ST_MultiLineString-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_MLineFromText-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_MLineFromText( wkt LONG VARCHAR, srid INT DEFAULT 0 )
RETURNS ST_MultiLineString
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromText( wkt, srid );
```

```
RETURN CAST( geo AS ST_MultiLineString);  
END
```

Hinweis

Die ST_MLineFromText-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_MultiLineString-Konstruktor“
- ST_GeomFromText-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 9.4.4

ST_MLineFromWKB-Funktion [Räumlich]

Gibt einen ST_MultiLineString-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_MultiLineString-Elements enthält.

Syntax

```
[DBO.]ST_MLineFromWKB(wkb, srid)
```

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_MultiLineString** Gibt einen ST_MultiLineString-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_MLineFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_MLineFromWKB( wkb LONG BINARY, srid INT DEFAULT 0 )  
RETURNS ST_MultiLineString  
BEGIN  
    DECLARE geo ST_Geometry;
```

```

    set geo = ST_Geometry::ST_GeomFromWKB( wkb, srid );
    RETURN CAST( geo AS ST_MultiLineString);
END

```

Hinweis

Die ST_MLineFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_MultiLineString-Konstruktor“
- ST_GeomFromWKB-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 9.4.5

ST_MPointFromText-Funktion [Räumlich]

Gibt einen ST_MultiPoint-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_MultiPoint-Elements enthält.

Syntax

```
[DBO.]ST_MPointFromText(wkt[, srid])
```

Parameter

Name	Typ	Beschreibung
wkt	LONG VARCHAR	Die WKT-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_MultiPoint** Gibt einen ST_MultiPoint-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_MPointFromText-Funktion ist äquivalent zu:

```

CREATE FUNCTION DBO.ST_MPointFromText( wkt LONG VARCHAR, srid INT DEFAULT 0 )
RETURNS ST_MultiPoint
BEGIN

```

```
DECLARE geo ST_Geometry;

set geo = ST_Geometry::ST_GeomFromText( wkt, srid );
RETURN CAST( geo AS ST_MultiPoint);
END
```

Hinweis

Die ST_MPointFromText-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [\[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Siehe auch

- [„ST_MultiPoint-Konstruktor“](#)
- [ST_GeomFromText-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.2.4

ST_MPointFromWKB-Funktion [Räumlich]

Gibt einen ST_MultiPoint-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_MultiPoint-Elements enthält.

Syntax

[DBO.]ST_MPointFromWKB(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_MultiPoint** Gibt einen ST_MultiPoint-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_MPointFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_MPointFromWKB( wkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_MultiPoint
```

```

BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromWKB( wkb, srid );
    RETURN CAST( geo AS ST_MultiPoint);
END

```

Hinweis

Die ST_MPointFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_MultiPoint-Konstruktor“
- ST_GeomFromWKB-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.2.5

ST_MPolyFromText-Funktion [Räumlich]

Gibt einen ST_MultiPolygon-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_MultiPolygon-Elements enthält.

Syntax

[DBO.]ST_MPolyFromText(*wkt*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkt	LONG VARCHAR	Die WKT-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_MultiPolygon** Gibt einen ST_MultiPolygon-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_MPolyFromText-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_MPolyFromText( wkt LONG VARCHAR, srid INT DEFAULT 0 )
RETURNS ST_MultiPolygon
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromText( wkt, srid );
    RETURN CAST( geo AS ST_MultiPolygon);
END
```

Hinweis

Die ST_MPolyFromText-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_MultiPolygon-Konstruktor“
- ST_GeomFromText-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 9.6.4

ST_MPolyFromWKB-Funktion [Räumlich]

Gibt einen ST_MultiPolygon-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_MultiPolygon-Elements enthält.

Syntax

[DBO.]ST_MPolyFromWKB(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_MultiPolygon** Gibt einen ST_MultiPolygon-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_MPolyFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_MPolyFromWKB( wkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_MultiPolygon
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromWKB( wkb, srid );
    RETURN CAST( geo AS ST_MultiPolygon);
END
```

Hinweis

Die ST_MPolyFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_MultiPolygon-Konstruktor“
- ST_GeomFromWKB-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.6.5

ST_MSurfaceFromTxt-Funktion [Räumlich]

Gibt einen ST_MultiSurface-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_MultiSurface-Elements enthält.

Syntax

[DBO.]ST_MSurfaceFromTxt(*wkt*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkt	LONG VARCHAR	Die WKT-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_MultiSurface** Gibt einen ST_MultiSurface-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_MSurfaceFromTxt-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_MSurfaceFromTxt( wkt LONG VARCHAR, srid INT DEFAULT
0 )
RETURNS ST_MultiSurface
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromText( wkt, srid );
    RETURN CAST( geo AS ST_MultiSurface);
END
```

Hinweis

Die ST_MSurfaceFromTxt-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [\[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Siehe auch

- [„ST_MultiSurface-Konstruktor“](#)
- [ST_GeomFromText-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.5.8

ST_MSurfaceFromWKB-Funktion [Räumlich]

Gibt einen ST_MultiSurface-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_MultiSurface-Elements enthält.

Syntax

```
[DBO.]ST_MSurfaceFromWKB(wkb[, srid])
```

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_MultiSurface** Gibt einen ST_MultiSurface-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_MSurfaceFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_MSurfaceFromWKB( wkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_MultiSurface
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromWKB( wkb, srid );
    RETURN CAST( geo AS ST_MultiSurface);
END
```

Hinweis

Die ST_MSurfaceFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_MultiSurface-Konstruktor“
- ST_GeomFromWKB-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 9.5.9

ST_OrderingEquals-Funktion [Räumlich]

Testet, ob eine Geometrie identisch mit einer anderen Geometrie ist.

Syntax

[DBO.]ST_OrderingEquals(*geo1*,*geo2*)

Parameter

Name	Typ	Beschreibung
geo1	ST_Geometry	Der erste Geometriewert, der sortiert werden soll.
geo2	ST_Geometry	Der zweite Geometriewert, der sortiert werden soll.

Rückgabe

- **INT** Gibt 1 zurück, wenn *geo1* genau gleich *geo2* ist, sonst 0.

Bemerkungen

Die ST_OrderingEquals-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_OrderingEquals( geo1 ST_Geometry, geo2 ST_Geometry )
RETURNS INT
BEGIN
    RETURN geo1.ST_OrderingEquals( geo2 );
END
```

Hinweis

Die ST_OrderingEquals-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [\[SQL Anywhere Server - SQL-Referenzhandbuch\]](#).

Siehe auch

- [ST_OrderingEquals-Methode für den ST_Geometry-Datentyp](#)

Standards und Kompatibilität

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.43

ST_PointFromText-Funktion [Räumlich]

Gibt einen ST_Point-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_Point-Elements enthält.

Syntax

```
[DBO.]ST_PointFromText(wkt[, srid])
```

Parameter

Name	Typ	Beschreibung
wkt	LONG VAR-CHAR	Die WKT-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_Point** Gibt einen ST_Point-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_PointFromText-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_PointFromText( wkt LONG VARCHAR, srid INT DEFAULT 0 )
RETURNS ST_Point
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromText( wkt, srid );
    RETURN CAST( geo AS ST_Point);
END
```

Hinweis

Die ST_PointFromText-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_Point-Konstruktor“
- ST_GeomFromText-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 6.1.8

ST_PointFromWKB-Funktion [Räumlich]

Gibt einen ST_Point-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_Point-Elements enthält.

Syntax

[DBO.]ST_PointFromWKB(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_Point** Gibt einen ST_Point-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_PointFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_PointFromWKB( wkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_Point
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromWKB( wkb, srid );
    RETURN CAST( geo AS ST_Point);
END
```

Hinweis

Die ST_PointFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_Point-Konstruktor“
- ST_GeomFromWKB-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 6.1.9

ST_PolyFromText-Funktion [Räumlich]

Gibt einen ST_Polygon-Wert zurück, der aus einem LONG VARCHAR-Wert umgewandelt wurde, der eine Well-Known-Text-Darstellung (WKT) eines ST_Polygon-Elements enthält.

Syntax

[DBO.]ST_PolyFromText(*wkt*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkt	LONG VAR-CHAR	Die WKT-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_Polygon** Gibt einen ST_Polygon-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_PolyFromText-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_PolyFromText( wkt LONG VARCHAR, srid INT DEFAULT 0 )
RETURNS ST_Polygon
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromText( wkt, srid );
    RETURN CAST( geo AS ST_Polygon);
END
```

Hinweis

Die ST_PolyFromText-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_Polygon-Konstruktor“
- ST_GeomFromText-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 8.3.6

ST_PolyFromWKB-Funktion [Räumlich]

Gibt einen ST_Polygon-Wert zurück, der aus einem LONG BINARY-Wert umgewandelt wurde, der eine Well-Known-Binary-Darstellung (WKB) eines ST_Polygon-Elements enthält.

Syntax

[DBO.]ST_PolyFromWKB(*wkb*[, *srid*])

Parameter

Name	Typ	Beschreibung
wkb	LONG BINARY	Die WKB-Darstellung.
srid	INT	Die SRID des Ergebnisses. Wenn sie nicht festgelegt wird, ist der Standardwert 0.

Rückgabe

- **ST_Polygon** Gibt einen ST_Polygon-Wert zurück, der aus der Eingabezeichenfolge erstellt wird.

Der Bezeichner des räumlichen Bezugssystems für das Ergebnis wird durch den Parameter *srid* angegeben.

Bemerkungen

Die ST_PolyFromWKB-Funktion ist äquivalent zu:

```
CREATE FUNCTION DBO.ST_PolyFromWKB( wkb LONG BINARY, srid INT DEFAULT 0 )
RETURNS ST_Polygon
BEGIN
    DECLARE geo ST_Geometry;

    set geo = ST_Geometry::ST_GeomFromWKB( wkb, srid );
    RETURN CAST( geo AS ST_Polygon);
END
```

Hinweis
Die ST_PolyFromWKB-Funktion ist in neu erstellten Datenbanken standardmäßig nicht vorhanden. Verwenden Sie die sa_install_feature-Systemprozedur, um die SQL-Kompatibilitätsfunktionen für räumliche Daten zu installieren. Siehe „sa_install_feature-Systemprozedur“ [[SQL Anywhere Server - SQL-Referenzhandbuch](#)].

Siehe auch

- „ST_Polygon-Konstruktor“
- ST_GeomFromWKB-Methode für den ST_Geometry-Datentyp

Standards und Kompatibilität

- SQL/MM (ISO/IEC 13249-3: 2006) 8.3.7

Liste aller unterstützten Methoden

Im Folgenden finden Sie eine Liste aller unterstützten räumlichen Methoden. Ein X in der Spalte "Gewölbte Erddarstellung" zeigt an, dass die Methode auch in räumlichen Bezugssystemen mit gewölbter Erddarstellung unterstützt wird. Die SQL/MM-Spalte zeigt die Kompatibilität mit den SQL/MM-Standards (ISO/IEC 13249-3: 2006) an.

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_Affine-Methode	„ST_Geometry-Datentyp“	Führen Sie eine affine Transformation durch, die Rotation, Konvertierung und Skalierung in einem einzigen Aufruf ausführt.		Erweiterung des Herstellers
ST_Area-Methode	„ST_MultiSurface-Datentyp“	Berechnet den Bereich der Mehrfachoberfläche in den angegebenen Einheiten.		9.5.3
ST_Area-Methode	„ST_Surface-Datentyp“	Berechnet den Bereich einer Fläche in den angegebenen Einheiten.		8.1.2

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_AsBinary-Methode	„ST_Geometry-Datentyp“	Gibt die WKB-Darstellung eines ST_Geometry Werts zurück.	X	5.1.37
ST_AsGML-Methode	„ST_Geometry-Datentyp“	Gibt die GML-Darstellung eines ST_Geometry-Werts zurück.	X	5.1.39
ST_AsGeoJSON-Methode	„ST_Geometry-Datentyp“	Gibt eine Zeichenfolge zurück, die eine Geometrie im JSON-Format darstellt.	X	Erweiterung des Herstellers
ST_AsKML-Methode	„ST_Geometry-Datentyp“	Gibt die KML-Darstellung eines ST_Geometry-Werts zurück.	X	5.1.39
ST_AsSVG-Methode	„ST_Geometry-Datentyp“	Gibt eine SVG-Abbildung für einen Geometriewert zurück.	X	Erweiterung des Herstellers
ST_AsText-Methode	„ST_Geometry-Datentyp“	Gibt die Textdarstellung eines ST_Geometry-Werts zurück.	X	5.1.35
ST_AsWKB-Methode	„ST_Geometry-Datentyp“	Gibt die WKB-Darstellung eines ST_Geometry Werts zurück.	X	Erweiterung des Herstellers
ST_AsWKT-Methode	„ST_Geometry-Datentyp“	Gibt die WKT-Darstellung eines ST_Geometry-Werts zurück.	X	Erweiterung des Herstellers
ST_AsXML-Methode	„ST_Geometry-Datentyp“	Gibt die XML-Darstellung eines ST_Geometry-Werts zurück.	X	Erweiterung des Herstellers
ST_Boundary-Methode	„ST_Geometry-Datentyp“	Gibt die Begrenzung des Domänenwerts zurück		5.1.14

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_Centroid-Methode	„ST_MultiSurface-Datentyp“	Berechnet den ST_Point, der der mathematische Schwerpunkt der Mehrfachoberfläche ist.		9.5.5
ST_Centroid-Methode	„ST_Surface-Datentyp“	Gibt den ST_Point-Wert zurück, der den mathematischen Schwerpunkt des Flächenwerts darstellt.		8.1.4
ST_Contains-Methode	„ST_Geometry-Datentyp“	Testet, ob ein räumlicher Geometriewert einen anderen räumlichen Geometriewert enthält.		5.1.31
ST_ContainsFilter-Methode	„ST_Geometry-Datentyp“	Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie eine andere enthält.		Erweiterung des Herstellers
ST_ConvexHull-Methode	„ST_Geometry-Datentyp“	Gibt die konvexe Hülle des Geometriewerts zurück.		5.1.16
ST_CoordDim-Methode	„ST_Geometry-Datentyp“	Gibt die Anzahl der Koordinatendimensionen zurück, die für jeden Punkt des ST_Geometry-Werts gespeichert sind.	X	5.1.3
ST_CoveredBy-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert räumlich von einem anderen Geometriewert abgedeckt wird.	X	Erweiterung des Herstellers
ST_CoveredByFilter-Methode	„ST_Geometry-Datentyp“	Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie von einer anderen abgedeckt wird.	X	Erweiterung des Herstellers
ST_Covers-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert räumlich einen anderen Geometriewert abdeckt.	X	Erweiterung des Herstellers

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_CoversFilter-Methode	„ST_Geometry-Datentyp“	Ein-kostengünstiger Test, um zu prüfen, ob eine Geometrie eine andere abdeckt.	X	Erweiterung des Herstellers
ST_Crosses-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert einen anderen Geometriewert kreuzt.		5.1.29
ST_CurveN-Methode	„ST_Compound-Curve-Datentyp“	Gibt die <i>n</i> -te Kurve in der Verbundkurve zurück.	X	7.4.5
ST_CurvePolyToPoly-Methode	„ST_CurvePolygon-Datentyp“	Gibt die Interpolation des Kurvenpolygons als Polygon zurück.	X	8.2.7
ST_CurveToLine-Methode	„ST_Curve-Datentyp“	Gibt die ST_LineString-Interpolation für einen ST_Curve-Wert zurück.	X	7.1.7
ST_Difference-Methode	„ST_Geometry-Datentyp“	Gibt den Geometriewert zurück, der die Punktmengendifferenz von zwei Geometrien darstellt.	X	5.1.20
ST_Dimension-Methode	„ST_Geometry-Datentyp“	Gibt die Dimension des ST_Geometry-Werts zurück. Die Punkte haben die Dimension 0, Linien haben die Dimension 1 und Flächen haben die Dimension 2. Eine leere Geometrie hat die Dimension -1.	X	5.1.2
ST_Disjoint-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert von einem anderen Wert räumlich unabhängig ist.	X	5.1.26
ST_Distance-Methode	„ST_Geometry-Datentyp“	Gibt die kleinste Entfernung zwischen <code>{selfexpr}</code> und dem angegebenen Geometriewert zurück.	X	5.1.23
ST_EndPoint-Methode	„ST_Curve-Datentyp“	Gibt einen ST_Point-Wert zurück, der der Endpunkt der ST_Curve-Kurve ist.	X	7.1.4

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_Envelope-Methode	„ST_Geometry-Datentyp“	Gibt das begrenzende Rechteck für den Geometriewert zurück.		5.1.15
ST_Equals-Methode	„ST_Geometry-Datentyp“	Testet, ob ein ST_Geometry-Wert räumlich gleich einem anderen ST_Geometry-Wert ist.	X	5.1.24
ST_EqualsFilter-Methode	„ST_Geometry-Datentyp“	Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie gleich einer anderen ist.	X	Erweiterung des Herstellers
ST_ExteriorRing-Methode	„ST_CurvePolygon-Datentyp“	Ruft den äußeren Ring ab oder ändert ihn.	X	8.2.3
ST_ExteriorRing-Methode	„ST_Polygon-Datentyp“	Ruft den äußeren Ring ab oder ändert ihn.	X	8.3.3
ST_GeometryN-Methode	„ST_GeomCollection-Datentyp“	Gibt die <i>n</i> -te Geometrie in der Geometriesammlung zurück.	X	9.1.5
ST_GeometryType-Methode	„ST_Geometry-Datentyp“	Gibt den Namen des Typs des ST_Geometry-Werts zurück.	X	5.1.4
ST_InteriorRingN-Methode	„ST_CurvePolygon-Datentyp“	Gibt den <i>n</i> -ten inneren Ring im Kurvenpolygon zurück.	X	8.2.6
ST_InteriorRingN-Methode	„ST_Polygon-Datentyp“	Gibt den <i>n</i> -ten inneren Ring im Polygon zurück.	X	8.3.5
ST_Intersection-Methode	„ST_Geometry-Datentyp“	Gibt den Geometriewert zurück, der die Punktmengenschnittmenge von zwei Geometrien darstellt.	X	5.1.18
ST_Intersects-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert räumlich eine Schnittmenge mit einem anderen Wert hat.	X	5.1.27
ST_IntersectsFilter-Methode	„ST_Geometry-Datentyp“	Ein kostengünstiger Test, um zu prüfen, ob sich die beiden Geometrien schneiden.	X	Erweiterung des Herstellers

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_Intersects-Rect-Methode	„ST_Geometry-Datentyp“	Testet, ob eine Geometrie eine Schnittmenge mit einem Rechteck hat.	X	Erweiterung des Herstellers
ST_Is3D-Methode	„ST_Geometry-Datentyp“	Ermittelt, ob der Geometrie-Wert Z-Koordinatenwerte hat.	X	5.1.10
ST_IsClosed-Methode	„ST_Curve-Datentyp“	Testet, ob der ST_Curve-Wert geschlossen ist. Eine Kurve ist geschlossen, wenn der Start- und der Endpunkt zusammenfallen.	X	7.1.5
ST_IsClosed-Methode	„ST_MultiCurve-Datentyp“	Testet, ob der ST_MultiCurve-Wert geschlossen ist. Eine Kurve ist geschlossen, wenn der Start- und der Endpunkt zusammenfallen. Eine Mehrfachkurve ist geschlossen, wenn sie nicht leer ist und eine leere Begrenzung hat.	X	9.3.3
ST_IsEmpty-Methode	„ST_Geometry-Datentyp“	Ermittelt, ob der Geometrie-Wert für eine leere Menge steht.	X	5.1.7
ST_IsMeasured-Methode	„ST_Geometry-Datentyp“	Ermittelt, ob der Geometrie-Wert zugeordnete Messwerte hat.	X	5.1.11
ST_IsRing-Methode	„ST_Curve-Datentyp“	Testet, ob der ST_Curve-Wert ein Ring ist. Eine Kurve ist ein Ring, wenn sie geschlossen und einfach ist (keine Schnittmengen mit sich selbst).	X	7.1.6
ST_IsSimple-Methode	„ST_Geometry-Datentyp“	Legt fest, ob der Geometrie-Wert einfach (keine Eigenschnittmengen oder andere Unregelmäßigkeiten) ist.	X	5.1.8
ST_IsValid-Methode	„ST_Geometry-Datentyp“	Ermittelt, ob die Geometrie ein gültiges räumliches Objekt ist.	X	5.1.9

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_IsWorld-Methode	„ST_Surface-Datentyp“	Testet, ob ST_Surface den gesamten Raum abdeckt.		8.1.6
ST_Lat-Methode	„ST_Point-Datentyp“	Gibt die Breitengrad-Koordinate des ST_Point-Werts zurück.	X	Erweiterung des Herstellers
ST_LatNorth-Methode	„ST_Geometry-Datentyp“	Ruft den nördlichsten Breitengrad einer Geometrie ab.	X	Erweiterung des Herstellers
ST_LatSouth-Methode	„ST_Geometry-Datentyp“	Ruft den südlichsten Breitengrad einer Geometrie ab.	X	Erweiterung des Herstellers
ST_Length-Methode	„ST_Curve-Datentyp“	Ruft die Längenmessung des Kurvenwerts ab.	X	7.1.2
ST_Length-Methode	„ST_MultiCurve-Datentyp“	Gibt die Längenmessung des ST_MultiCurve-Werts zurück. Das Ergebnis wird mit den Einheiten gemessen, die durch den Parameter festgelegt sind.	X	9.3.4
ST_LinearHash-Methode	„ST_Geometry-Datentyp“	Gibt eine Binärzeichenfolge zurück, die einen linearen Hash der Geometrie darstellt.	X	Erweiterung des Herstellers
ST_Long-Methode	„ST_Point-Datentyp“	Gibt die Längengradkoordinate des ST_Point-Werts zurück.	X	Erweiterung des Herstellers
ST_LongEast-Methode	„ST_Geometry-Datentyp“	Ruft den Längengrad der östlichen Begrenzung einer Geometrie ab.	X	Erweiterung des Herstellers

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_LongWest-Methode	„ST_Geometry-Datentyp“	Ruft den Längengrad der westlichen Begrenzung einer Geometrie ab.	X	Erweiterung des Herstellers
ST_M-Methode	„ST_Point-Datentyp“	Ruft den Messwert eines Punkts ab oder ändert ihn.	X	6.1.6
ST_MMax-Methode	„ST_Geometry-Datentyp“	Ruft den maximalen M-Koordinatenwert einer Geometrie ab.	X	Erweiterung des Herstellers
ST_MMin-Methode	„ST_Geometry-Datentyp“	Ruft den minimalen M-Koordinatenwert einer Geometrie ab.	X	Erweiterung des Herstellers
ST_NumCurves-Methode	„ST_CompoundCurve-Datentyp“	Gibt die Anzahl der Kurven zurück, die die Definition der Verbundkurve definieren.	X	7.4.4
ST_NumGeometries-Methode	„ST_GeomCollection-Datentyp“	Gibt die Anzahl der Geometrien in der Geometriesammlung zurück	X	9.1.4
ST_NumInteriorRing-Methode	„ST_CurvePolygon-Datentyp“	Gibt die Anzahl der inneren Ringe im Kurvenpolygon zurück.	X	8.2.5
ST_NumPoints-Methode	„ST_CircularString-Datentyp“	Gibt die Anzahl der Punkte zurück, die die Kreisbogenfolge definieren.	X	7.3.4
ST_NumPoints-Methode	„ST_LineString-Datentyp“	Gibt die Anzahl der Punkte zurück, die die Linienfolge definieren.	X	7.2.4
ST_OrderingEquals-Methode	„ST_Geometry-Datentyp“	Testet, ob eine Geometrie identisch mit einer anderen Geometrie ist.	X	5.1.43
ST_Overlaps-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert einen anderen Geometriewert überlappt.		5.1.32

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_Perimeter-Methode	„ST_MultiSurface-Datentyp“	Berechnet den Umfang der Mehrfachoberfläche in den angegebenen Einheiten.	X	9.5.4
ST_Perimeter-Methode	„ST_Surface-Datentyp“	Berechnet den Umfang einer Fläche in den angegebenen Einheiten.	X	8.1.3
ST_PointN-Methode	„ST_CircularString-Datentyp“	Gibt den <i>n</i> -ten Punkt in der Kreisbogenfolge zurück.	X	7.3.5
ST_PointN-Methode	„ST_LineString-Datentyp“	Gibt den <i>n</i> -ten Punkt in der Linienfolge zurück.	X	7.2.5
ST_PointOnSurface-Methode	„ST_MultiSurface-Datentyp“	Gibt einen Punkt zurück, der garantiert auf einer Fläche im Mehrfachoberflächenobjekt liegt	X	9.5.6
ST_PointOnSurface-Methode	„ST_Surface-Datentyp“	Gibt einen ST_Point-Wert zurück, der garantiert eine räumliche Schnittmenge mit dem ST_Surface-Wert aufweist.	X	8.1.5

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_Relate-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert räumlich mit einem anderen, in der Schnittmatrix angegebenen Geometriewert in Beziehung steht. Die ST_Relate-Methode verwendet eine aus 9 Stellen bestehende Zeichenfolge aus dem Dimensionally Extended 9 Intersection Model (DE-9IM), um die paarweise Beziehung zwischen zwei räumlichen Datenelementen zu beschreiben. Die ST_Relate-Methode ermittelt beispielsweise, ob eine Schnittmenge zwischen den Geometrien vorliegt und gegebenenfalls die Geometrie der resultierenden Schnittmenge. Siehe auch: „Funktionsweise räumlicher Beziehungen“ auf Seite 52.		5.1.25, Erweiterung des Herstellers
ST_Reverse-Methode	„ST_Geometry-Datentyp“	Gibt die Geometrie mit einer umgekehrten Elementreihenfolge zurück.	X	Erweiterung des Herstellers
ST_SRID-Methode	„ST_Geometry-Datentyp“	Ruft das mit dem Geometriewert verbundene räumliche Bezugssystem ab oder verändert es.	X	5.1.5
ST_SnapToGrid-Methode	„ST_Geometry-Datentyp“	Gibt eine Kopie der Geometrie zurück, in der alle Punkte am angegebenen Raster ausgerichtet sind.	X	Erweiterung des Herstellers
ST_StartPoint-Methode	„ST_Curve-Datentyp“	Gibt einen ST_Point-Wert zurück, der der Startpunkt des ST_Curve-Werts ist.	X	7.1.3
ST_SymDifference-Methode	„ST_Geometry-Datentyp“	Gibt den Geometriewert zurück, der die symmetrische Differenz der Punktmenge zweier Geometrien darstellt.	X	5.1.21

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_ToCircular-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in eine Kreisbogenfolge.	X	5.1.33
ST_ToCompound-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in eine Verbundkurve.	X	5.1.33
ST_ToCurve-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in eine Kurve.	X	Erweiterung des Herstellers
ST_ToCurvePoly-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in ein Kurvenpolygon.	X	5.1.33
ST_ToGeomColl-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in eine Geometriegruppe.	X	5.1.33
ST_ToLineString-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in eine Linienfolge.	X	5.1.33
ST_ToMultiCurve-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in einen Mehrfachkurvenwert.	X	5.1.33
ST_ToMultiLine-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in einen Mehrlinienfolgenwert.	X	5.1.33
ST_ToMultiPoint-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in einen Mehrpunktwert.	X	5.1.33
ST_ToMultiPolygon-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in einen Multipolygonwert.	X	5.1.33
ST_ToMultiSurface-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in einen Mehrfachoberflächenwert.	X	5.1.33
ST_ToPoint-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in einen Punkt.	X	5.1.33
ST_ToPolygon-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in ein Polygon.	X	5.1.33
ST_ToSurface-Methode	„ST_Geometry-Datentyp“	Konvertiert die Geometrie in eine Fläche.	X	Erweiterung des Herstellers

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_Touches-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert räumlich einen anderen Geometriewert berührt.		5.1.28
ST_Transform-Methode	„ST_Geometry-Datentyp“	Erstellt eine Kopie des in das angegebene räumliche Bezugssystem konvertierten Geometriewerts.	X	5.1.6
ST_Union-Methode	„ST_Geometry-Datentyp“	Gibt den Geometriewert zurück, der die Punktmengenvereinigung von zwei Geometrien darstellt.	X	5.1.19
ST_Within-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert räumlich in einem anderen Geometriewert enthalten ist.		5.1.30
ST_WithinDistance-Methode	„ST_Geometry-Datentyp“	Testet, ob sich zwei Geometrien innerhalb eines angegebenen Abstands voneinander befinden.	X	Erweiterung des Herstellers
ST_WithinDistanceFilter-Methode	„ST_Geometry-Datentyp“	Ein kostengünstiger Test, ob zwei Geometrien in einem angegebenen Abstand voneinander liegen.	X	Erweiterung des Herstellers
ST_WithinFilter-Methode	„ST_Geometry-Datentyp“	Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie innerhalb einer anderen liegt.		Erweiterung des Herstellers
ST_X-Methode	„ST_Point-Datentyp“	Ruft die X-Koordinate eines Punkts ab oder ändert sie.	X	6.1.3
ST_XMax-Methode	„ST_Geometry-Datentyp“	Ruft den maximalen X-Koordinatenwert einer Geometrie ab.	X	Erweiterung des Herstellers
ST_XMin-Methode	„ST_Geometry-Datentyp“	Ruft den minimalen X-Koordinatenwert einer Geometrie ab.	X	Erweiterung des Herstellers

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_Y-Methode	„ST_Point-Datentyp“	Ruft die Y-Koordinate eines Punkts ab oder ändert sie.	X	6.1.4
ST_YMax-Methode	„ST_Geometry-Datentyp“	Ruft den maximalen Y-Koordinatenwert einer Geometrie ab.	X	Erweiterung des Herstellers
ST_YMin-Methode	„ST_Geometry-Datentyp“	Ruft den minimalen Y-Koordinatenwert einer Geometrie ab.	X	Erweiterung des Herstellers
ST_Z-Methode	„ST_Point-Datentyp“	Ruft die Z-Koordinate eines Punkts ab oder ändert sie.	X	6.1.4, 6.1.5
ST_ZMax-Methode	„ST_Geometry-Datentyp“	Ruft den maximalen Z-Koordinatenwert einer Geometrie ab.	X	Erweiterung des Herstellers
ST_ZMin-Methode	„ST_Geometry-Datentyp“	Ruft den minimalen Z-Koordinatenwert einer Geometrie ab.	X	Erweiterung des Herstellers

Liste aller unterstützten Konstruktoren

Im Folgenden finden Sie eine Liste aller unterstützten räumlichen Konstruktoren. Ein X in der Spalte "Gewölbte Erddarstellung" zeigt an, dass die Methode auch in räumlichen Bezugssystemen mit gewölbter Erddarstellung unterstützt wird. Die SQL/MM-Spalte zeigt die Kompatibilität mit den SQL/MM-Standards (ISO/IEC 13249-3: 2006) an.

Konstruktor	Beschreibung	Gewölbte Erddarstellung	SQL/MM
„ST_CircularString-Konstruktor“	Konstruiert eine Kreisbogenfolge.	X	7.3.2, Erweiterung des Herstellers
„ST_CompoundCurve-Konstruktor“	Konstruiert eine Verbundkurve.	X	7.4.2, Erweiterung des Herstellers

Konstruktor	Beschreibung	Gewölbte Erddarstellung	SQL/MM
„ST_CurvePolygon-Konstruktor“	Konstruiert ein Kurvenpolygon.	X	8.2.2, Erweiterung des Herstellers
„ST_GeomCollection-Konstruktor“	Konstruiert eine Geometriesammlung.	X	9.1.2, Erweiterung des Herstellers
„ST_LineString-Konstruktor“	Konstruiert eine Linienfolge.	X	7.2.2, Erweiterung des Herstellers
„ST_MultiCurve-Konstruktor“	Konstruiert eine Mehrfachkurve.	X	9.3.2, Erweiterung des Herstellers
„ST_MultiLineString-Konstruktor“	Konstruiert eine Mehrfachlinienfolge.	X	9.4.2, Erweiterung des Herstellers
„ST_MultiPoint-Konstruktor“	Konstruiert einen Mehrfachpunkt.	X	9.2.2, Erweiterung des Herstellers
„ST_MultiPolygon-Konstruktor“	Konstruiert ein Multipolygon.	X	9.6.2, Erweiterung des Herstellers
„ST_MultiSurface-Konstruktor“	Konstruiert eine Mehrfachoberfläche.	X	9.5.2, Erweiterung des Herstellers
„ST_Point-Konstruktor“	Konstruiert einen Punkt.	X	6.1.2
„ST_Polygon-Konstruktor“	Konstruiert ein Polygon.	X	8.3.2, Erweiterung des Herstellers

Liste der statischen Methoden

Im Folgenden finden Sie eine Liste aller statischen Methoden, die mit räumlichen Daten verwendet werden können. Ein X in der Spalte "Gewölbte Erddarstellung" zeigt an, dass die Methode auch in räumlichen Bezugssystemen mit gewölbter Erddarstellung unterstützt wird. Die SQL/MM-Spalte zeigt die Kompatibilität mit den SQL/MM-Standards (ISO/IEC 13249-3: 2006) an.

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_AsSVGAggr-Methode	„ST_Geometry-Datentyp“	Gibt ein vollständiges oder teilweises SVG-Dokument zurück, das die Geometrien in einer Gruppe darstellt.	X	Erweiterung des Herstellers

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_CompareWKT-Methode	„ST_SpatialRefSys-Datentyp“	Vergleicht zwei Definitionen von räumlichen Bezugssystemen.	X	Erweiterung des Herstellers
ST_ConvexHullAggr-Methode	„ST_Geometry-Datentyp“	Gibt die konvexe Hülle für alle Geometrien in einer Gruppe zurück		Erweiterung des Herstellers
ST_EnvelopeAggr-Methode	„ST_Geometry-Datentyp“	Gibt das begrenzende Rechteck für alle Geometrien in einer Gruppe zurück		Erweiterung des Herstellers
ST_FormatTransformDefinition-Methode	„ST_SpatialRefSys-Datentyp“	Gibt eine formatierte Kopie der Transformationsdefinition zurück.	X	Erweiterung des Herstellers
ST_FormatWKT-Methode	„ST_SpatialRefSys-Datentyp“	Gibt eine formatierte Kopie der WKT-Definition zurück.	X	Erweiterung des Herstellers
ST_GeomCollectionAggr-Methode	„ST_GeomCollection-Datentyp“	Gibt eine Geometriesammlung zurück, die alle Geometrien in einer Gruppe enthält.	X	Erweiterung des Herstellers
ST_GeomFromBinary-Methode	„ST_Geometry-Datentyp“	Konstruiert eine Geometrie aus einer binären Zeichenfolgendarstellung.	X	Erweiterung des Herstellers
ST_GeomFromShape-Methode	„ST_Geometry-Datentyp“	Führt die syntaktische Analyse einer Zeichenfolge mit einem ESRI-Formdatensatz durch und erstellt einen Geometriewert des entsprechenden Datentyps.	X	Erweiterung des Herstellers
ST_GeomFromText-Methode	„ST_Geometry-Datentyp“	Konstruiert eine Geometrie aus einer Zeichenfolgendarstellung.	X	5.1.40

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_GeomFromWKB-Methode	„ST_Geometry-Datentyp“	Führt die syntaktische Analyse einer Zeichenfolge durch, die eine WKB- oder EWKB-Darstellung einer Geometrie enthält und erstellt einen Geometriewert des entsprechenden Typs.	X	5.1.41
ST_GeomFromWKT-Methode	„ST_Geometry-Datentyp“	Führt die syntaktische Analyse einer Zeichenfolge durch, die die WKT- oder EWKT-Darstellung eines Geometriewerts enthält und erstellt einen Geometriewert des entsprechenden Typs.	X	Erweiterung des Herstellers
ST_GeometryTypeFromBase-Type-Methode	„ST_Geometry-Datentyp“	Führt die syntaktische Analyse einer Zeichenfolge durch, die die Typzeichenfolge definiert.	X	Erweiterung des Herstellers
ST_GetUnProjectedTransformDefinition-Methode	„ST_SpatialRefSys-Datentyp“	Gibt die Transformationsdefinition des räumlichen Bezugssystems zurück, das die Quelle der Projektion ist.	X	Erweiterung des Herstellers
ST_IntersectionAggr-Methode	„ST_Geometry-Datentyp“	Gibt die räumliche Schnittmenge aller Geometrien in einer Gruppe zurück	X	Erweiterung des Herstellers
ST_LineStringAggr-Methode	„ST_LineString-Datentyp“	Gibt eine Linienfolge zurück, die aus den sortierten Punkten in einer Gruppe gebildet wird.	X	Erweiterung des Herstellers
ST_LinearUnHash-Methode	„ST_Geometry-Datentyp“	Gibt eine Geometrie zurück, die den Index-Hash darstellt.	X	Erweiterung des Herstellers
ST_LoadConfigurationData-Methode	„ST_Geometry-Datentyp“	Gibt binäre Konfigurationsdaten zurück. Wird nur intern verwendet.	X	Erweiterung des Herstellers

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_MultiCurveAggr-Methode	„ST_MultiCurve-Datentyp“	Gibt eine Mehrfachkurve mit allen Kurven in einer Gruppe zurück.	X	Erweiterung des Herstellers
ST_MultiLineStringAggr-Methode	„ST_MultiLineString-Datentyp“	Gibt eine Mehrfachlinienfolge zurück, die alle Linienfolgen in einer Gruppe enthält.	X	Erweiterung des Herstellers
ST_MultiPointAggr-Methode	„ST_MultiPoint-Datentyp“	Gibt einen Mehrfachpunkt zurück, der alle Punkte in einer Gruppe enthält.	X	Erweiterung des Herstellers
ST_MultiPolygonAggr-Methode	„ST_MultiPolygon-Datentyp“	Gibt ein Multipolygon zurück, das alle Polygone in einer Gruppe enthält.	X	Erweiterung des Herstellers
ST_MultiSurfaceAggr-Methode	„ST_MultiSurface-Datentyp“	Gibt eine Mehrfachoberfläche mit allen Flächen in einer Gruppe zurück.	X	Erweiterung des Herstellers
ST_ParseWKT-Methode	„ST_SpatialRefSys-Datentyp“	Ruft ein benanntes Element aus der Well-Known-Text-Definition (WKT) eines räumlichen Bezugssystems ab.	X	Erweiterung des Herstellers
ST_SRIDFromBaseType-Methode	„ST_Geometry-Datentyp“	Führt die syntaktische Analyse einer Zeichenfolge durch, die die Typzeichenfolge definiert.	X	Erweiterung des Herstellers
ST_TransformGeom-Methode	„ST_SpatialRefSys-Datentyp“	Gibt die mit der angegebenen Transformationsdefinition umgewandelte Geometrie zurück.	X	Erweiterung des Herstellers
ST_UnionAggr-Methode	„ST_Geometry-Datentyp“	Gibt die räumliche Vereinigung aller Geometrien in einer Gruppe zurück.	X	Erweiterung des Herstellers
ST_World-Methode	„ST_SpatialRefSys-Datentyp“	Gibt eine Geometrie zurück, die alle Punkte des räumlichen Bezugssystems darstellt.		Erweiterung des Herstellers

Liste der Aggregatmethoden

Im Folgenden finden Sie eine Liste der Aggregatmethoden, die mit räumlichen Daten verwendet werden können. Ein X in der Spalte "Gewölbte Erddarstellung" zeigt an, dass die Methode auch in räumlichen Bezugssystemen mit gewölbter Erddarstellung unterstützt wird. Die SQL/MM-Spalte zeigt die Kompatibilität mit den SQL/MM-Standards (ISO/IEC 13249-3: 2006) an.

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_AsSV-GAggr-Methode	„ST_Geometry-Datentyp“	Gibt ein vollständiges oder teilweises SVG-Dokument zurück, das die Geometrien in einer Gruppe darstellt.	X	Erweiterung des Herstellers
ST_Convex-HullAggr-Methode	„ST_Geometry-Datentyp“	Gibt die konvexe Hülle für alle Geometrien in einer Gruppe zurück		Erweiterung des Herstellers
ST_EnvelopeAggr-Methode	„ST_Geometry-Datentyp“	Gibt das begrenzende Rechteck für alle Geometrien in einer Gruppe zurück		Erweiterung des Herstellers
ST_GeomCollectionAggr-Methode	„ST_GeomCollection-Datentyp“	Gibt eine Geometriesammlung zurück, die alle Geometrien in einer Gruppe enthält.	X	Erweiterung des Herstellers
ST_IntersectionAggr-Methode	„ST_Geometry-Datentyp“	Gibt die räumliche Schnittmenge aller Geometrien in einer Gruppe zurück	X	Erweiterung des Herstellers
ST_LineStringAggr-Methode	„ST_LineString-Datentyp“	Gibt eine Linienfolge zurück, die aus den sortierten Punkten in einer Gruppe gebildet wird.	X	Erweiterung des Herstellers
ST_MultiCurveAggr-Methode	„ST_MultiCurve-Datentyp“	Gibt eine Mehrfachkurve mit allen Kurven in einer Gruppe zurück.	X	Erweiterung des Herstellers
ST_MultiLineStringAggr-Methode	„ST_MultiLineString-Datentyp“	Gibt eine Mehrfachlinienfolge zurück, die alle Linienfolgen in einer Gruppe enthält.	X	Erweiterung des Herstellers
ST_MultiPointAggr-Methode	„ST_MultiPoint-Datentyp“	Gibt einen Mehrfachpunkt zurück, der alle Punkte in einer Gruppe enthält.	X	Erweiterung des Herstellers

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_MultiPolygonAggr-Methode	„ST_MultiPolygon-Datentyp“	Gibt ein Multipolygon zurück, das alle Polygone in einer Gruppe enthält.	X	Erweiterung des Herstellers
ST_MultiSurfaceAggr-Methode	„ST_MultiSurface-Datentyp“	Gibt eine Mehrfachoberfläche mit allen Flächen in einer Gruppe zurück.	X	Erweiterung des Herstellers
ST_UnionAggr-Methode	„ST_Geometry-Datentyp“	Gibt die räumliche Vereinigung aller Geometrien in einer Gruppe zurück	X	Erweiterung des Herstellers

Liste der Set-Operationsmethoden

Im Folgenden finden Sie eine Liste der Set-Operationsmethoden, die mit räumlichen Daten verwendet werden können. Ein X in der Spalte "Gewölbte Erddarstellung" zeigt an, dass die Methode auch in räumlichen Bezugssystemen mit gewölbter Erddarstellung unterstützt wird. Die SQL/MM-Spalte zeigt die Kompatibilität mit den SQL/MM-Standards (ISO/IEC 13249-3: 2006) an.

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_Difference-Methode	„ST_Geometry-Datentyp“	Gibt den Geometriewert zurück, der die Punktmengendifferenz von zwei Geometrien darstellt.	X	5.1.20
ST_Intersection-Methode	„ST_Geometry-Datentyp“	Gibt den Geometriewert zurück, der die Punktmengenschnittmenge von zwei Geometrien darstellt.	X	5.1.18
ST_IntersectionAggr-Methode	„ST_Geometry-Datentyp“	Gibt die räumliche Schnittmenge aller Geometrien in einer Gruppe zurück	X	Erweiterung des Herstellers
ST_SymDifference-Methode	„ST_Geometry-Datentyp“	Gibt den Geometriewert zurück, der die symmetrische Differenz der Punktmenge zweier Geometrien darstellt.	X	5.1.21
ST_Union-Methode	„ST_Geometry-Datentyp“	Gibt den Geometriewert zurück, der die Punktmengenvereinigung von zwei Geometrien darstellt.	X	5.1.19

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_Unio-nAggr-Methode	„ST_Geometry-Datentyp“	Gibt die räumliche Vereinigung aller Geometrien in einer Gruppe zurück	X	Erweiterung des Herstellers

Liste räumlicher Prädikate

Im Folgenden finden Sie eine Liste der Prädikatmethoden, die mit räumlichen Daten verwendet werden können. Ein X in der Spalte "Gewölbte Erddarstellung" zeigt an, dass die Methode auch in räumlichen Bezugssystemen mit gewölbter Erddarstellung unterstützt wird. Die SQL/MM-Spalte zeigt die Kompatibilität mit den SQL/MM-Standards (ISO/IEC 13249-3: 2006) an.

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_Contains-Methode	„ST_Geometry-Datentyp“	Testet, ob ein räumlicher Geometriewert einen anderen räumlichen Geometriewert enthält.		5.1.31
ST_Contains-Filter-Methode	„ST_Geometry-Datentyp“	Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie eine andere enthält.		Erweiterung des Herstellers
ST_CoveredBy-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert räumlich von einem anderen Geometriewert abgedeckt wird.	X	Erweiterung des Herstellers
ST_CoveredByFilter-Methode	„ST_Geometry-Datentyp“	Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie von einer anderen abgedeckt wird.	X	Erweiterung des Herstellers
ST_Covers-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert räumlich einen anderen Geometriewert abdeckt.	X	Erweiterung des Herstellers
ST_CoversFilter-Methode	„ST_Geometry-Datentyp“	Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie eine andere abdeckt.	X	Erweiterung des Herstellers

Methode	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_Crosses-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert einen anderen Geometriewert kreuzt.		5.1.29
ST_Disjoint-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert von einem anderen Wert räumlich unabhängig ist.	X	5.1.26
ST_Equals-Methode	„ST_Geometry-Datentyp“	Testet, ob ein ST_Geometry-Wert räumlich gleich einem anderen ST_Geometry-Wert ist.	X	5.1.24
ST_EqualsFilter-Methode	„ST_Geometry-Datentyp“	Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie gleich einer anderen ist.	X	Erweiterung des Herstellers
ST_Intersects-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert räumlich eine Schnittmenge mit einem anderen Wert hat.	X	5.1.27
ST_IntersectsFilter-Methode	„ST_Geometry-Datentyp“	Ein kostengünstiger Test, um zu prüfen, ob sich die beiden Geometrien schneiden.	X	Erweiterung des Herstellers
ST_IntersectsRect-Methode	„ST_Geometry-Datentyp“	Testet, ob eine Geometrie eine Schnittmenge mit einem Rechteck hat.	X	Erweiterung des Herstellers
ST_OrderingEquals-Methode	„ST_Geometry-Datentyp“	Testet, ob eine Geometrie identisch mit einer anderen Geometrie ist.	X	5.1.43
ST_Overlaps-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert einen anderen Geometriewert überlappt.		5.1.32

Methoden	Typ	Beschreibung	Gewölbte Erddarstellung	SQL/MM
ST_Relate-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert räumlich mit einem anderen, in der Schnittmatrix angegebenen Geometriewert in Beziehung steht. Die ST_Relate-Methode verwendet eine aus 9 Stellen bestehende Zeichenfolge aus dem Dimensionally Extended 9 Intersection Model (DE-9IM), um die paarweise Beziehung zwischen zwei räumlichen Datenelementen zu beschreiben. Die ST_Relate-Methode ermittelt beispielsweise, ob eine Schnittmenge zwischen den Geometrien vorliegt und gegebenenfalls die Geometrie der resultierenden Schnittmenge. Siehe auch: „Funktionsweise räumlicher Beziehungen“ auf Seite 52.		5.1.25, Erweiterung des Herstellers
ST_Touches-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert räumlich einen anderen Geometriewert berührt.		5.1.28
ST_Within-Methode	„ST_Geometry-Datentyp“	Testet, ob ein Geometriewert räumlich in einem anderen Geometriewert enthalten ist.		5.1.30
ST_WithinDistance-Methode	„ST_Geometry-Datentyp“	Testet, ob sich zwei Geometrien innerhalb eines angegebenen Abstands voneinander befinden.	X	Erweiterung des Herstellers
ST_WithinDistanceFilter-Methode	„ST_Geometry-Datentyp“	Ein kostengünstiger Test, ob zwei Geometrien in einem angegebenen Abstand voneinander liegen.	X	Erweiterung des Herstellers
ST_WithinFilter-Methode	„ST_Geometry-Datentyp“	Ein kostengünstiger Test, um zu prüfen, ob eine Geometrie innerhalb einer anderen liegt.		Erweiterung des Herstellers

Index

Symbole

1000004326, SRID
WGS 84 (planar),3
3857, SRID
Kompatibilität mit verbreiteten
Zuordnungsanwendungen,5
4326, SRID
WGS 84,3
900913, SRID
Kompatibilität mit verbreiteten
Zuordnungsanwendungen,5

A

Am Raster ausrichten
Wie sich Ausrichten am Raster und Toleranz auf
räumliche Berechnungen auswirken,44
Ambiguität
Instanzierungsmethoden für räumliche Datentypen
aufrufen,22
ArcGIS Online, Daten
Kompatibilität mit verbreiteten
Zuordnungsanwendungen,5
Außenbereiche
räumliche Daten,51

B

Besondere Hinweise
räumliche Daten,12
Beziehungen
räumliche Daten,52
Bing Maps, Daten
Kompatibilität mit verbreiteten
Zuordnungsanwendungen,5
Bogenmaß
räumliche Daten,6

C

Clustered-Indizes
Indizierung räumlicher Daten,21
compatibility-Funktionen
spatial data,343

D

Datentypen
Räumliche Datentypen,71
DE-9IM
Info,53
DISTINCT-Klausel
Geometrievergleiche bei räumlichen Daten,52

E

EWKB, Format
räumliche Daten, unterstützte Formate,13
EWKT, Format
räumliche Daten, unterstützte Formate,13
Exportieren
räumliche Daten, unterstützte Formate,13
Extended Well Known Binary (EWKB), Format
räumliche Daten, unterstützte Formate,13
Extended Well Known Text (EWKT), Format
räumliche Daten, unterstützte Formate,13

F

Formdateien
Beispiel für das Einlesen einer ESRI-Formdatei,59
praktische Einführung zum Laden,57
räumliche Daten, unterstützte Formate,15
Unterstützung von ESRI-Formdateien,15

G

Geografien und Geometrien
Unterschiede der räumlichen Terminologie in SQL
Anywhere,12
Geographic Markup Language (GML), Format
räumliche Daten, unterstützte Formate,13
GeoJSON, Format
räumliche Daten, unterstützte Formate,13
Geometriegruppen
Info,9
Geometrien
anzeigen, Spatial Viewer,26
Ausgabe in SVG,65
Info,9
Geometrien und Geografien
Unterschiede der räumlichen Terminologie in SQL
Anywhere,12
Geometrietyp
Definition,7
GML, Format

- räumliche Daten, unterstützte Formate,13
- Google Earth, Daten
 - Kompatibilität mit verbreiteten
 - Zuordnungsanwendungen,5
- GROUP BY-Klausel
 - Geometrievergleiche bei räumlichen Daten,52

I

- Importieren
 - räumliche Daten, unterstützte Formate,13
- Indizes
 - Indexbeschränkungen für räumliche Spalten,21
 - räumliche Spalten,21
- Innenbereiche
 - räumliche Daten,51
- Installieren
 - Maßeinheiten,6
- Instanzmethoden
 - räumliche Datentypen,22
- Interactive SQL
 - Anzeigen räumlicher Daten,26
 - räumliche Daten anzeigen,26
- Interpolation
 - Auswirkungen der Interpolation auf räumliche
 - Berechnungen,48
 - Info,48
- IS NOT OF, Ausdrücke
 - räumliche Prädikate verwenden,24
- IS OF, Ausdrücke
 - räumliche Prädikate verwenden,24

J

- JavaScript Object Notation (JSON), Format
 - räumliche Daten, unterstützte Formate,13
- JSON, Format
 - räumliche Daten, unterstützte Formate,13

K

- Klassen
 - Räumliche Datentypen,71
- KML, Format
 - räumliche Daten, unterstützte Formate,13
- Kombinierte Kurven
 - Info,9
- Konstrukturen
 - räumliche Daten,22
- Kreisbogenfolgen

- Info,9
- Kurvenpolygone
 - Info,9

L

- Linienfolge, Geometrietyp
 - Definition,7
- Linienfolgen
 - Info,8

M

- Maßeinheit
 - erstellen,37
 - installieren, Beispiel,57
- Maßeinheiten
 - Info,6
- Mehrfachoberflächen
 - Info,9
- Mehrlinienfolge, Geometrietyp
 - Definition,7
- Mehrlinienfolgen
 - Info,9
- Mehrpunktangabe
 - Info,9
- Mehrpunktangabe, Geometrietyp
 - Definition,7
- Multipolygon, Geometrietyp
 - Definition,7
- Multipolygone
 - Info,9

N

- NEW, Schlüsselwort
 - Erstellung räumlicher Objekte,22

O

- ORDER BY-Klausel
 - Geometrievergleiche bei räumlichen Daten,52
- OUTPUT-Anweisung
 - Ausgabe einer Geometrie in SVG zur Anzeige,65

P

- Plane Erde
 - WGS 84 (planar), Info,3
- Plane Erde, Modell
 - räumliche Daten,42
- Polygon, Geometrietyp

- Definition,7
- Polygone
 - Info,8
 - Ringausrichtung,50
- Prädikate
 - räumlich,10
- Praktische Einführungen
 - mit räumlichen Funktionen experimentieren,57
- Projektion in räumliche Bezugssystemen mit planer Erddarstellung
 - Info,43
- Punkt, Geometrietyp
 - Definition,7
- Punkte
 - Info,8

R

- Räumliche Bezugssysteme
 - Erstellen,38
 - Info,2
 - Liste unterstützter Systeme abfragen,4
 - Liste unterstützter Typen,2
- Räumliche Daten
 - benutzerdefinierte Typen,22
 - besondere Hinweise,12
 - Beziehungen,52
 - Clustered-Indizes verwenden,21
 - Einführung,1
 - empfohlene Literatur,16
 - Geometrien anzeigen,65
 - Geometrien in der Datenbank erstellen,25
 - Geometrien in Interactive SQL anzeigen,26
 - Geometrievergleiche bei räumlichen Daten,52
 - Indexbeschränkungen,21
 - Indexempfehlungen,21
 - Info,1
 - Instanzmethoden,22
 - Kompatibilität mit verbreiteten Zuordnungsanwendungen,5
 - Konformität mit Standards,12
 - Liste der unterstützten Aggregatmethoden,395
 - Liste der unterstützten Konstruktoren,390
 - Liste der unterstützten Methoden,378
 - Liste der unterstützten Prädikate,397
 - Liste der unterstützten Set-Operationen,396
 - Liste der unterstützten statischen Methoden,391
 - Liste unterstützter Import- und Exportformate,13
- Maßeinheiten,6
 - mit planer und gewölbter Erddarstellung,42
 - nicht unterstützte Methoden,12
 - nicht unterstützte Methoden in SQL Anywhere,13
 - praktische Einführung: mit räumlichen Funktionen experimentieren,57
 - räumliche Datentypenhierarchie,7
 - räumliche Prädikate,10
 - räumliches Bezugssystem, Bezeichner,2
 - räumliches Standardbezugssystem,3
 - sa_octahedral_gnomonic, räumliches Bezugssystem,4
 - sa_planar_unbounded, räumliches Bezugssystem,4
 - Spalte für räumliche Daten erstellen,17
 - Spaltenintegritätsregeln,18
 - Spaltenintegritätsregeln hinzufügen,18
 - statische Aggregatmethoden,24
 - statische Methoden,23
 - Syntax für räumliche Datentypen,22
 - testen, Beziehungen,52
 - Textindizes,21
 - Typen, Methoden und Konstruktoren,71
 - unterstützte Geometrietypen,7
 - unterstützte Importformate,13
 - unterstützte räumliche Bezugssysteme,2
 - Well Known Text, Ladebeispiel,31
 - WGS 84 (planar), räumliches Bezugssystem,3
 - WGS 84, räumliches Bezugssystem,3
 - Zugriff,71
- Räumliche Datentypen
 - Instanzmethoden,22
 - statische Aggregatmethoden,24
- Räumliche Formate
 - Unterstützung von ESRI-Formdateien,15
- Räumliche SQL-API
 - ST_BdMPolyFromText-Funktion [Räumlich],346
 - ST_BdMPolyFromWKB-Funktion [Räumlich],347
 - ST_BdPolyFromText-Funktion [Räumlich],348
 - ST_BdPolyFromWKB-Funktion [Räumlich],349
 - ST_CircularFromTxt-Funktion [Räumlich],352
 - ST_CircularFromWKB-Funktion [Räumlich],354
 - ST_CompoundFromTxt-Funktion [Räumlich],355
 - ST_CompoundFromWKB-Funktion [Räumlich],356
 - ST_CPolyFromText-Funktion [Räumlich],350
 - ST_CPolyFromWKB-Funktion [Räumlich],351
 - ST_GeomCollFromTxt-Funktion [Räumlich],357

- ST_GeomCollFromWKB-Funktion [Räumlich],358
- ST_GeomFromText-Funktion [Räumlich],359
- ST_GeomFromWKB-Funktion [Räumlich],360
- ST_LineFromText-Funktion [Räumlich],361
- ST_LineFromWKB-Funktion [Räumlich],362
- ST_MCurveFromText-Funktion [Räumlich],363
- ST_MCurveFromWKB-Funktion [Räumlich],364
- ST_MLineFromText-Funktion [Räumlich],365
- ST_MLineFromWKB-Funktion [Räumlich],366
- ST_MPointFromText-Funktion [Räumlich],367
- ST_MPointFromWKB-Funktion [Räumlich],368
- ST_MPolyFromText-Funktion [Räumlich],369
- ST_MPolyFromWKB-Funktion [Räumlich],370
- ST_MSurfaceFromText-Funktion [Räumlich],371
- ST_MSurfaceFromWKB-Funktion [Räumlich],372
- ST_OrderingEquals-Funktion [Räumlich],373
- ST_PointFromText-Funktion [Räumlich],374
- ST_PointFromWKB-Funktion [Räumlich],375
- ST_PolyFromText-Funktion [Räumlich],376
- ST_PolyFromWKB-Funktion [Räumlich],377
- Räumliche Vorschau
 - Geometrien in Interactive SQL anzeigen,26
- Räumliches Standardbezugssystem
 - räumliche Daten,3
- Ringausrichtung
 - Polygone,50
- Runde Erde
 - WGS 84, Info,3
- Runde Erde, Modell
 - räumliche Daten,42
- S**
- sa_install_feature-Systemprozedur
 - Maßeinheiten,6
- sa_octahedral_gnomonic
 - räumliche Bezugssysteme,4
- sa_planar_unbounded
 - räumliche Bezugssysteme,4
- Scalable Vector Graphic (SVG), Format
 - räumliche Daten, unterstützte Formate,13
- Schnittpunkttests
 - räumliche Daten,53
- Spalten
 - räumlich,17
- spatial data
 - compatibility-Funktionen,343
- Spatial Viewer
 - Geometrien in Interactive SQL anzeigen,26
- SQL-API für räumliche Daten
 - ST_CircularString-Datentyp,71
 - ST_CompoundCurve-Datentyp,79
 - ST_Curve-Datentyp,85
 - ST_CurvePolygon-Datentyp,92
 - ST_GeomCollection-Datentyp,104
 - ST_Geometry-Datentyp,112
 - ST_LineString-Datentyp,254
 - ST_MultiCurve-Datentyp,262
 - ST_MultiLineString-Datentyp,271
 - ST_MultiPoint-Datentyp,277
 - ST_MultiPolygon-Datentyp,283
 - ST_MultiSurface-Datentyp,291
 - ST_Point-Datentyp,302
 - ST_Polygon-Datentyp,319
 - ST_SpatialRefSys-Datentyp,329
 - ST_Surface-Datentyp,337
- SQL-Standards
 - räumliche Daten,12
- SQL/MM-Standard
 - benutzerdefinierte Typen,22
 - Info,12
- SRIDs
 - als Integritätsregeln,18
 - als Integritätsregeln hinzufügen,18
 - Info,2
 - räumliche Referenz-IDs, Info,2
- SRS
 - Räumliche Bezugssysteme, Info,2
- ST_Affine
 - Methode, ST_Geometry-Datentyp,114
- ST_Area
 - Methode, ST_MultiSurface-Datentyp,297
 - Methode, ST_Surface-Datentyp,339
- ST_AsBinary
 - Methode, ST_Geometry-Datentyp,115
- ST_AsGeoJSON
 - Methode, ST_Geometry-Datentyp,122
- ST_AsGML
 - Methode, ST_Geometry-Datentyp,118
- ST_AsKML
 - Methode, ST_Geometry-Datentyp,124
- ST_AsSVG
 - Methode, ST_Geometry-Datentyp,127
- ST_AsSVGAggr
 - Methode, ST_Geometry-Datentyp,131

ST_AsText	ST_ConvexHullAggr
Methode, ST_Geometry-Datentyp,136	Methode, ST_Geometry-Datentyp,166
ST_AsWKB	ST_CoordDim
Methode, ST_Geometry-Datentyp,147	Methode, ST_Geometry-Datentyp,167
ST_AsWKT	ST_CoveredBy
Methode, ST_Geometry-Datentyp,149	Methode, ST_Geometry-Datentyp,168
ST_AsXML	ST_CoveredByFilter
Methode, ST_Geometry-Datentyp,150	Methode, ST_Geometry-Datentyp,170
ST_BdMPolyFromText	ST_Covers
Funktion, SQL-API für räumliche Daten,346	Methode, ST_Geometry-Datentyp,171
ST_BdMPolyFromWKB	ST_CoversFilter
Funktion, SQL-API für räumliche Daten,347	Methode, ST_Geometry-Datentyp,172
ST_BdPolyFromText	ST_CPolyFromText
Funktion, SQL-API für räumliche Daten,348	Funktion, SQL-API für räumliche Daten,350
ST_BdPolyFromWKB	ST_CPolyFromWKB
Funktion, SQL-API für räumliche Daten,349	Funktion, SQL-API für räumliche Daten,351
ST_Boundary	ST_Crosses
Methode, ST_Geometry-Datentyp,160	Methode, ST_Geometry-Datentyp,173
Methode, zusätzliche Informationen,51	ST_Curve
ST_Centroid	Typ, Beschreibung,85
Methode, ST_MultiSurface-Datentyp,298	Typ, ST_CurveToLine-Methode,87
Methode, ST_Surface-Datentyp,340	Typ, ST_EndPoint-Methode,88
ST_CircularFromTxt	Typ, ST_IsClosed-Methode,88
Funktion, SQL-API für räumliche Daten,352	Typ, ST_IsRing-Methode,89
ST_CircularFromWKB	Typ, ST_Length-Methode,90
Funktion, SQL-API für räumliche Daten,354	Typ, ST_StartPoint-Methode,92
ST_CircularString	ST_CurveN
Konstruktor, [SQL-API für räumliche Daten],73	Methode, ST_CompoundCurve-Datentyp,83
Typ, Beschreibung,71	ST_CurvePolygon
Typ, ST_NumPoints-Methode,77	Konstruktor, [SQL-API für räumliche Daten],94
Typ, ST_PointN-Methode,77	Typ, Beschreibung,92
ST_CompareWKT	Typ, ST_CurvePolyToPoly-Methode,100
Methode, ST_SpatialRefSys-Datentyp,330	Typ, ST_ExteriorRing-Methode,101
ST_CompoundCurve	Typ, ST_InteriorRingN-Methode,103
Konstruktor, [SQL-API für räumliche Daten],80	Typ, ST_NumInteriorRing-Methode,104
Typ, Beschreibung,79	ST_CurvePolyToPoly
Typ, ST_CurveN-Methode,83	Methode, ST_CurvePolygon-Datentyp,100
Typ, ST_NumCurves-Methode,84	ST_CurveToLine
ST_CompoundFromTxt	Methode, ST_Curve-Datentyp,87
Funktion, SQL-API für räumliche Daten,355	ST_Difference
ST_CompoundFromWKB	Methode, ST_Geometry-Datentyp,174
Funktion, SQL-API für räumliche Daten,356	ST_Dimension
ST_Contains	Methode, ST_Geometry-Datentyp,176
Methode, ST_Geometry-Datentyp,161	Methode, zusätzliche Informationen,56
ST_ContainsFilter	ST_Disjoint
Methode, ST_Geometry-Datentyp,163	Methode, ST_Geometry-Datentyp,177
ST_ConvexHull	ST_Distance
Methode, ST_Geometry-Datentyp,164	Methode, ST_Geometry-Datentyp,178

- ST_EndPoint
 - Methode, ST_Curve-Datentyp, 88
- ST_Envelope
 - Methode, ST_Geometry-Datentyp, 180
- ST_EnvelopeAggr
 - Methode, ST_Geometry-Datentyp, 181
- ST_Equals
 - Methode, ST_Geometry-Datentyp, 182
 - Methode, Verwendung in Geometrievergleichen, 51
- ST_EqualsFilter
 - Methode, ST_Geometry-Datentyp, 183
- ST_ExteriorRing
 - Methode, ST_CurvePolygon-Datentyp, 101
 - Methode, ST_Polygon-Datentyp, 326
- ST_FormatTransformDefinition
 - Methode, ST_SpatialRefSys-Datentyp, 331
- ST_FormatWKT
 - Methode, ST_SpatialRefSys-Datentyp, 331
- ST_GeomCollection
 - Konstruktor, [SQL-API für räumliche Daten], 106
 - Typ, Beschreibung, 104
 - Typ, ST_GeomCollectionAggr-Methode, 109
 - Typ, ST_GeometryN-Methode, 111
 - Typ, ST_NumGeometries-Methode, 111
- ST_GeomCollectionAggr
 - Methode, ST_GeomCollection-Datentyp, 109
- ST_GeomCollFromTxt
 - Funktion, SQL-API für räumliche Daten, 357
- ST_GeomCollFromWKB
 - Funktion, SQL-API für räumliche Daten, 358
- ST_Geometry
 - Typ, Beschreibung, 112
 - Typ, ST_Affine-Methode, 114
 - Typ, ST_AsBinary-Methode, 115
 - Typ, ST_AsGeoJSON-Methode, 122
 - Typ, ST_AsGML-Methode, 118
 - Typ, ST_AsKML-Methode, 124
 - Typ, ST_AsSVG-Methode, 127
 - Typ, ST_AsSVGAgr-Methode, 131
 - Typ, ST_AsText-Methode, 136
 - Typ, ST_AsWKB-Methode, 147
 - Typ, ST_AsWKT-Methode, 149
 - Typ, ST_AsXML-Methode, 150
 - Typ, ST_Boundary-Methode, 160
 - Typ, ST_Contains-Methode, 161
 - Typ, ST_ContainsFilter-Methode, 163
 - Typ, ST_ConvexHull-Methode, 164
 - Typ, ST_ConvexHullAggr-Methode, 166
 - Typ, ST_CoordDim-Methode, 167
 - Typ, ST_CoveredBy-Methode, 168
 - Typ, ST_CoveredByFilter-Methode, 170
 - Typ, ST_Covers-Methode, 171
 - Typ, ST_CoversFilter-Methode, 172
 - Typ, ST_Crosses-Methode, 173
 - Typ, ST_Difference-Methode, 174
 - Typ, ST_Dimension-Methode, 176
 - Typ, ST_Disjoint-Methode, 177
 - Typ, ST_Distance-Methode, 178
 - Typ, ST_Envelope-Methode, 180
 - Typ, ST_EnvelopeAggr-Methode, 181
 - Typ, ST_Equals-Methode, 182
 - Typ, ST_EqualsFilter-Methode, 183
 - Typ, ST_GeometryType-Methode, 189
 - Typ, ST_GeometryTypeFromBaseType-Methode, 189
 - Typ, ST_GeomFromBinary-Methode, 184
 - Typ, ST_GeomFromShape-Methode, 185
 - Typ, ST_GeomFromText-Methode, 186
 - Typ, ST_GeomFromWKB-Methode, 187
 - Typ, ST_GeomFromWKT-Methode, 188
 - Typ, ST_Intersection-Methode, 190
 - Typ, ST_IntersectionAggr-Methode, 192
 - Typ, ST_Intersects-Methode, 193
 - Typ, ST_IntersectsFilter-Methode, 194
 - Typ, ST_IntersectsRect-Methode, 195
 - Typ, ST_Is3D-Methode, 196
 - Typ, ST_IsEmpty-Methode, 197
 - Typ, ST_IsMeasured-Methode, 197
 - Typ, ST_IsSimple-Methode, 198
 - Typ, ST_IsValid-Methode, 198
 - Typ, ST_LatNorth-Methode, 199
 - Typ, ST_LatSouth-Methode, 200
 - Typ, ST_LinearHash-Methode, 201
 - Typ, ST_LinearUnHash-Methode, 202
 - Typ, ST_LoadConfigurationData-Methode, 203
 - Typ, ST_LongEast-Methode, 203
 - Typ, ST_LongWest-Methode, 204
 - Typ, ST_MMax-Methode, 205
 - Typ, ST_MMin-Methode, 206
 - Typ, ST_OrderingEquals-Methode, 207
 - Typ, ST_Overlaps-Methode, 208
 - Typ, ST_Relate-Methode, 210
 - Typ, ST_Reverse-Methode, 213
 - Typ, ST_SnapToGrid-Methode, 216
 - Typ, ST_SRID-Methode, 214
 - Typ, ST_SRIDFromBaseType-Methode, 215

Typ, ST_SymDifference-Methode,220
 Typ, ST_ToCircular-Methode,221
 Typ, ST_ToCompound-Methode,222
 Typ, ST_ToCurve-Methode,223
 Typ, ST_ToCurvePoly-Methode,224
 Typ, ST_ToGeomColl-Methode,225
 Typ, ST_ToLineString-Methode,226
 Typ, ST_ToMultiCurve-Methode,227
 Typ, ST_ToMultiLine-Methode,228
 Typ, ST_ToMultiPoint-Methode,230
 Typ, ST_ToMultiPolygon-Methode,231
 Typ, ST_ToMultiSurface-Methode,232
 Typ, ST_ToPoint-Methode,233
 Typ, ST_ToPolygon-Methode,234
 Typ, ST_ToSurface-Methode,235
 Typ, ST_Touches-Methode,236
 Typ, ST_Transform-Methode,237
 Typ, ST_Union-Methode,239
 Typ, ST_UnionAggr-Methode,240
 Typ, ST_Within-Methode,241
 Typ, ST_WithinDistance-Methode,242
 Typ, ST_WithinDistanceFilter-Methode,244
 Typ, ST_WithinFilter-Methode,246
 Typ, ST_XMax-Methode,247
 Typ, ST_XMin-Methode,249
 Typ, ST_YMax-Methode,250
 Typ, ST_YMin-Methode,251
 Typ, ST_ZMax-Methode,252
 Typ, ST_ZMin-Methode,253
 ST_GeometryN
 Methode, ST_GeomCollection-Datentyp,111
 ST_GeometryType
 Methode, ST_Geometry-Datentyp,189
 ST_GeometryTypeFromBaseType
 Methode, ST_Geometry-Datentyp,189
 ST_GeomFromBinary
 Methode, ST_Geometry-Datentyp,184
 ST_GeomFromShape
 Methode, ST_Geometry-Datentyp,185
 ST_GeomFromText
 Funktion, SQL-API für räumliche Daten,359
 Methode, ST_Geometry-Datentyp,186
 ST_GeomFromWKB
 Funktion, SQL-API für räumliche Daten,360
 Methode, ST_Geometry-Datentyp,187
 ST_GeomFromWKT
 Methode, ST_Geometry-Datentyp,188
 ST_GetUnProjectedTransformDefinition
 Methode, ST_SpatialRefSys-Datentyp,332
 ST_InteriorRingN
 Methode, ST_CurvePolygon-Datentyp,103
 Methode, ST_Polygon-Datentyp,328
 ST_Intersection
 Methode, ST_Geometry-Datentyp,190
 ST_IntersectionAggr
 Methode, ST_Geometry-Datentyp,192
 ST_Intersects
 Methode, Beispiel,62
 Methode, ST_Geometry-Datentyp,193
 ST_IntersectsFilter
 Methode, ST_Geometry-Datentyp,194
 ST_IntersectsRect
 Methode, ST_Geometry-Datentyp,195
 ST_Is3D
 Methode, ST_Geometry-Datentyp,196
 ST_IsClosed
 Methode, ST_Curve-Datentyp,88
 Methode, ST_MultiCurve-Datentyp,267
 ST_IsEmpty
 Methode, ST_Geometry-Datentyp,197
 ST_IsMeasured
 Methode, ST_Geometry-Datentyp,197
 ST_IsRing
 Methode, ST_Curve-Datentyp,89
 ST_IsSimple
 Methode, ST_Geometry-Datentyp,198
 ST_IsValid
 Methode, ST_Geometry-Datentyp,198
 ST_IsWorld
 Methode, ST_Surface-Datentyp,341
 ST_Lat
 Methode, ST_Point-Datentyp,308
 ST_LatNorth
 Methode, ST_Geometry-Datentyp,199
 ST_LatSouth
 Methode, ST_Geometry-Datentyp,200
 ST_Length
 Methode, ST_Curve-Datentyp,90
 Methode, ST_MultiCurve-Datentyp,268
 ST_LinearHash
 Methode, ST_Geometry-Datentyp,201
 ST_LinearUnHash
 Methode, ST_Geometry-Datentyp,202
 ST_LineFromText
 Funktion, SQL-API für räumliche Daten,361
 ST_LineFromWKB

- Funktion, SQL-API für räumliche Daten,362
- ST_LineString
 - Konstruktor, [SQL-API für räumliche Daten],256
 - Typ, Beschreibung,254
 - Typ, ST_LineStringAggr-Methode,259
 - Typ, ST_NumPoints-Methode,260
 - Typ, ST_PointN-Methode,261
- ST_LineStringAggr
 - Methode, ST_LineString-Datentyp,259
- ST_LoadConfigurationData
 - Methode, ST_Geometry-Datentyp,203
- ST_Long
 - Methode, ST_Point-Datentyp,310
- ST_LongEast
 - Methode, ST_Geometry-Datentyp,203
- ST_LongWest
 - Methode, ST_Geometry-Datentyp,204
- ST_M
 - Methode, ST_Point-Datentyp,312
- ST_MCurveFromText
 - Funktion, SQL-API für räumliche Daten,363
- ST_MCurveFromWKB
 - Funktion, SQL-API für räumliche Daten,364
- ST_MLineFromText
 - Funktion, SQL-API für räumliche Daten,365
- ST_MLineFromWKB
 - Funktion, SQL-API für räumliche Daten,366
- ST_MMax
 - Methode, ST_Geometry-Datentyp,205
- ST_MMin
 - Methode, ST_Geometry-Datentyp,206
- ST_MPointFromText
 - Funktion, SQL-API für räumliche Daten,367
- ST_MPointFromWKB
 - Funktion, SQL-API für räumliche Daten,368
- ST_MPolyFromText
 - Funktion, SQL-API für räumliche Daten,369
- ST_MPolyFromWKB
 - Funktion, SQL-API für räumliche Daten,370
- ST_MSurfaceFromTxt
 - Funktion, SQL-API für räumliche Daten,371
- ST_MSurfaceFromWKB
 - Funktion, SQL-API für räumliche Daten,372
- ST_MultiCurve
 - Konstruktor, [SQL-API für räumliche Daten],264
 - Typ, Beschreibung,262
 - Typ, ST_IsClosed-Methode,267
 - Typ, ST_Length-Methode,268
 - Typ, ST_MultiCurveAggr-Methode,269
- ST_MultiCurveAggr
 - Methode, ST_MultiCurve-Datentyp,269
- ST_MultiLineString
 - Konstruktor, [SQL-API für räumliche Daten],272
 - Typ, Beschreibung,271
 - Typ, ST_MultiLineStringAggr-Methode,276
- ST_MultiLineStringAggr
 - Methode, ST_MultiLineString-Datentyp,276
- ST_MultiPoint
 - Konstruktor, [SQL-API für räumliche Daten],279
 - Typ, Beschreibung,277
 - Typ, ST_MultiPointAggr-Methode,282
- ST_MultiPointAggr
 - Methode, ST_MultiPoint-Datentyp,282
- ST_MultiPolygon
 - Konstruktor, [SQL-API für räumliche Daten],285
 - Typ, Beschreibung,283
 - Typ, ST_MultiPolygonAggr-Methode,289
- ST_MultiPolygonAggr
 - Methode, ST_MultiPolygon-Datentyp,289
- ST_MultiSurface
 - Konstruktor, [SQL-API für räumliche Daten],292
 - Typ, Beschreibung,291
 - Typ, ST_Area-Methode,297
 - Typ, ST_Centroid-Methode,298
 - Typ, ST_MultiSurfaceAggr-Methode,298
 - Typ, ST_Perimeter-Methode,300
 - Typ, ST_PointOnSurface-Methode,301
- ST_MultiSurfaceAggr
 - Methode, ST_MultiSurface-Datentyp,298
- ST_NumCurves
 - Methode, ST_CompoundCurve-Datentyp,84
- ST_NumGeometries
 - Methode, ST_GeomCollection-Datentyp,111
- ST_NumInteriorRing
 - Methode, ST_CurvePolygon-Datentyp,104
- ST_NumPoints
 - Methode, ST_CircularString-Datentyp,77
 - Methode, ST_LineString-Datentyp,260
- ST_OrderingEquals
 - Funktion, SQL-API für räumliche Daten,373
 - Methode, ST_Geometry-Datentyp,207
 - Methode, Verwendung in Geometrievergleichen,51
- ST_Overlaps
 - Methode, ST_Geometry-Datentyp,208
- ST_ParseWKT
 - Methode, ST_SpatialRefSys-Datentyp,333

ST_Perimeter
 Methode, ST_MultiSurface-Datentyp,300
 Methode, ST_Surface-Datentyp,341

ST_Point
 Konstruktor, [SQL-API für räumliche Daten],304
 Typ, Beschreibung,302
 Typ, ST_Lat-Methode,308
 Typ, ST_Long-Methode,310
 Typ, ST_M-Methode,312
 Typ, ST_X-Methode,314
 Typ, ST_Y-Methode,316
 Typ, ST_Z-Methode,317

ST_PointFromText
 Funktion, SQL-API für räumliche Daten,374

ST_PointFromWKB
 Funktion, SQL-API für räumliche Daten,375

ST_PointN
 Methode, ST_CircularString-Datentyp,77
 Methode, ST_LineString-Datentyp,261

ST_PointOnSurface
 Methode, ST_MultiSurface-Datentyp,301
 Methode, ST_Surface-Datentyp,343

ST_PolyFromText
 Funktion, SQL-API für räumliche Daten,376

ST_PolyFromWKB
 Funktion, SQL-API für räumliche Daten,377

ST_Polygon
 Konstruktor, [SQL-API für räumliche Daten],321
 Typ, Beschreibung,319
 Typ, ST_ExteriorRing-Methode,326
 Typ, ST_InteriorRingN-Methode,328

ST_Relate
 Methode, als Prädikat verwendet,53
 Methode, ST_Geometry-Datentyp,210
 Methode, Verwendung nicht als Prädikat,55
 Methode, zusätzliche Informationen,52

ST_Reverse
 Methode, ST_Geometry-Datentyp,213

ST_SnapToGrid
 Methode, ST_Geometry-Datentyp,216

ST_SPATIAL_REFERENCE_SYSTEMS
 verwenden,4

ST_SpatialRefSys
 Typ, Beschreibung,329
 Typ, ST_CompareWKT-Methode,330
 Typ, ST_FormatTransformDefinition-Methode,331
 Typ, ST_FormatWKT-Methode,331

 Typ, ST_GetUnProjectedTransformDefinition-Methode,332
 Typ, ST_ParseWKT-Methode,333
 Typ, ST_TransformGeom-Methode,335
 Typ, ST_World-Methode,336

ST_SRID
 Methode, ST_Geometry-Datentyp,214

ST_SRIDFromBaseType
 Methode, ST_Geometry-Datentyp,215

ST_StartPoint
 Methode, ST_Curve-Datentyp,92

ST_Surface
 Typ, Beschreibung,337
 Typ, ST_Area-Methode,339
 Typ, ST_Centroid-Methode,340
 Typ, ST_IsWorld-Methode,341
 Typ, ST_Perimeter-Methode,341
 Typ, ST_PointOnSurface-Methode,343

ST_SymDifference
 Methode, ST_Geometry-Datentyp,220

ST_ToCircular
 Methode, ST_Geometry-Datentyp,221

ST_ToCompound
 Methode, ST_Geometry-Datentyp,222

ST_ToCurve
 Methode, ST_Geometry-Datentyp,223

ST_ToCurvePoly
 Methode, ST_Geometry-Datentyp,224

ST_ToGeomColl
 Methode, ST_Geometry-Datentyp,225

ST_ToLineString
 Methode, ST_Geometry-Datentyp,226

ST_ToMultiCurve
 Methode, ST_Geometry-Datentyp,227

ST_ToMultiLine
 Methode, ST_Geometry-Datentyp,228

ST_ToMultiPoint
 Methode, ST_Geometry-Datentyp,230

ST_ToMultiPolygon
 Methode, ST_Geometry-Datentyp,231

ST_ToMultiSurface
 Methode, ST_Geometry-Datentyp,232

ST_ToPoint
 Methode, ST_Geometry-Datentyp,233

ST_ToPolygon
 Methode, ST_Geometry-Datentyp,234

ST_ToSurface
 Methode, ST_Geometry-Datentyp,235

- ST_Touches
 - Methode, Beispiel,67
 - Methode, ST_Geometry-Datentyp,236
- ST_Transform
 - Methode, ST_Geometry-Datentyp,237
- ST_TransformGeom
 - Methode, ST_SpatialRefSys-Datentyp,335
- ST_Union
 - Methode, ST_Geometry-Datentyp,239
- ST_UnionAggr
 - Methode, Beispiel,62
 - Methode, ST_Geometry-Datentyp,240
- ST_Within
 - Methode, Beispiel,62
 - Methode, ST_Geometry-Datentyp,241
- ST_WithinDistance
 - Methode, ST_Geometry-Datentyp,242
- ST_WithinDistanceFilter
 - Methode, ST_Geometry-Datentyp,244
- ST_WithinFilter
 - Methode, ST_Geometry-Datentyp,246
- ST_World
 - Methode, ST_SpatialRefSys-Datentyp,336
- ST_X
 - Methode, ST_Point-Datentyp,314
- ST_XMax
 - Methode, ST_Geometry-Datentyp,247
- ST_XMin
 - Methode, ST_Geometry-Datentyp,249
- ST_Y
 - Methode, ST_Point-Datentyp,316
- ST_YMax
 - Methode, ST_Geometry-Datentyp,250
- ST_YMin
 - Methode, ST_Geometry-Datentyp,251
- ST_Z
 - Methode, ST_Point-Datentyp,317
- ST_ZMax
 - Methode, ST_Geometry-Datentyp,252
- ST_ZMin
 - Methode, ST_Geometry-Datentyp,253
- Statische Aggregatmethoden
 - räumliche Datentypen,24
- Statische Methoden
 - räumliche Datentypen,23
- SVG, Format
 - Ausgabe einer Geometrie in SVG zur Anzeige,65
 - räumliche Daten, unterstützte Formate,13

- SVGs
 - in Interactive SQL anzeigen,26
 - Info,13

T

- Textindizes
 - für räumliche Spalten,21
- Toleranz
 - wie Ausrichtung am Raster und Toleranz
 - räumliche Berechnungen beeinflussen,44

U

- UDT-Typen
 - Syntax räumlicher Datentypen,22
- Unterstützte Geometrietypen
 - räumliche Daten,7

V

- Vergleichsoperatoren
 - Geometrievergleiche bei räumlichen Daten,52

W

- Well Known Binary (WKB), Format
 - räumliche Daten, unterstützte Formate,13
- Well Known Text
 - Ladebeispiel,31
- Well Known Text (WKT)
 - räumliche Daten, unterstützte Formate,13
- WGS 84
 - räumliches Bezugssystem,3
- WGS 84 (planar)
 - räumliches Bezugssystem,3
- WKB, Format
 - räumliche Daten, unterstützte Formate,13
- WKT
 - Ladebeispiel,31
- WKT, Format
 - räumliche Daten, unterstützte Formate,13

Z

- Zugriff auf räumliche Daten und Analyse
 - Info,71