SYBASE iAnywhere

SQL Anywhere® サーバー
空間データサポート

バージョン 12.0.1

2012 年 1 月

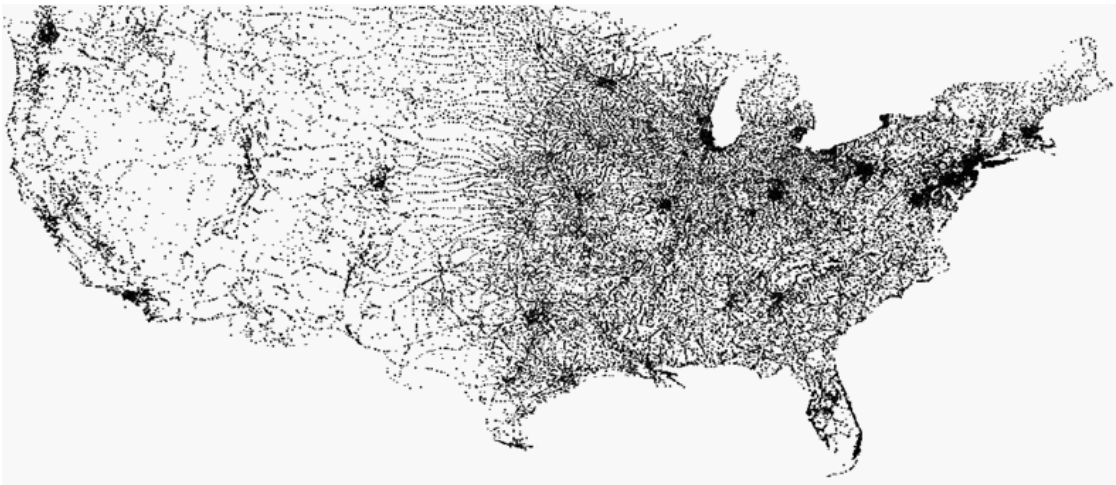
バージョン 12.0.1
2012 年 1 月

目次

はじめに

このマニュアルでは、SQL Anywhere 空間データサポートについて説明し、空間機能を使用して空間データを生成し、分析する方法について説明します。

次の図は、アメリカ合衆国における都市や町の分布を表していますが、これは空間データに対する興味深い操作の一例です。



空間データの使用

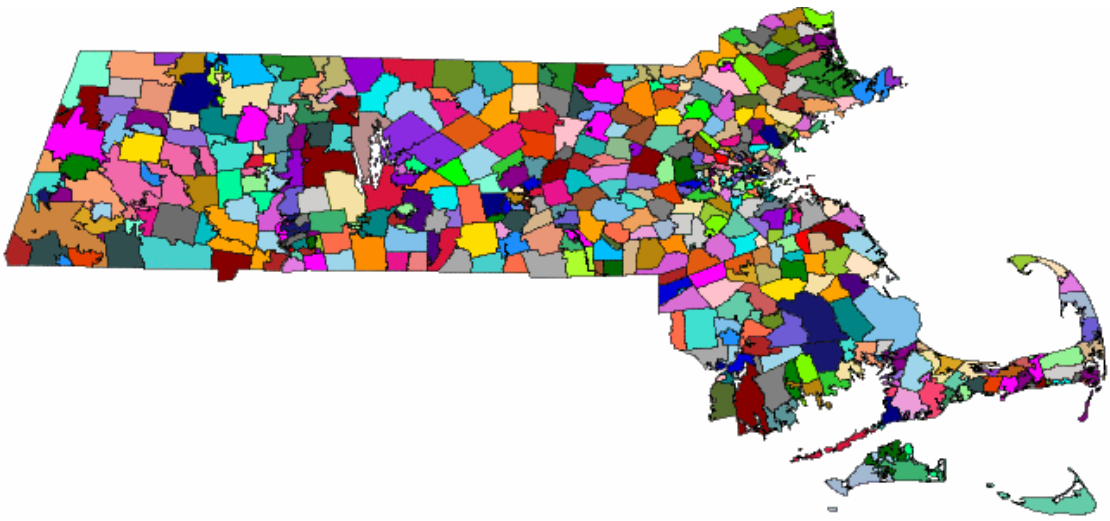
この項では、SQL Anywhere 空間サポートを紹介し、その目的、サポートされるデータ型、空間データを生成および分析する方法について説明します。

空間データのマニュアルでは、作業を行う空間参照系と空間データについての知識があることを前提としています。空間データについての知識がない場合は、「空間トピック関連の推奨ドキュメント」19 ページに追加資料へのリンクがあります。

注意
32 ビット Windows と 32 ビット Linux の空間データサポートには、SSE2 命令をサポートする CPU が必要となります。これは、Intel Pentium 4 (2001 年リリース) 以降と AMD Opteron (2003 年リリース) 以降でサポートされます。

空間データの概要

「空間データ」は、定義された空間内のオブジェクトの位置、シェイプ、方向を記述するデータです。SQL Anywhere の空間データは、ポイント、曲線 (線ストリングと円弧ストリング)、多角形の形式の 2D ジオメトリとして表現されます。たとえば、次のイメージはマサチューセッツ州の郵便番号の区域を表す多角形の論理和を表しています。



空間データに対して実行される 2 つの一般的な操作は、ジオメトリ間の距離の計算と、複数のオブジェクトの論理和または共通部分の判断です。これらの計算は、Intersects、Contains、Crosses などの述部を使用して実行されます。

空間データの使用例

SQL Anywhere の空間データサポートによって、アプリケーション開発者は既存のデータと空間情報を関連付けることができます。たとえば、会社を表すテーブルに、会社の場所をポイントと

して格納したり、会社の配送地域を多角形として格納したりできます。これは、SQL では次のように表すことができます。

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

例で使用されている空間データ型 ST_Point は単一のポイントを表し、ST_Polygon は任意の多角形を表します。このスキーマを使用すると、アプリケーションで会社の場所をすべて地図上に表示したり、次のようなクエリを使用して会社が特定の住所に配送しているかどうかを検索したりできます。

```
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 では、空間データの格納およびデータ管理の機能が提供されているため、地理的な場所、ルート情報、シェイプデータなどの情報を格納できます。

これらの情報の断片は、ポイント、さまざまな形式の多角形、線として、対応する「空間データ型」 (ST_Point、ST_Polygon など) で定義されたカラムに格納されます。空間データへのアクセスおよび操作を行うには、メソッドおよびコンストラクターを使用します。SQL Anywhere では、他の製品との互換性のために設計された一連の SQL 空間関数も提供されます。

参照

- 「サポートされる空間データ型とその階層」 7 ページ
- 「空間互換関数」 355 ページ
- 「チュートリアル：空間機能の実験」 59 ページ

空間参照系 (SRS) と空間参照系識別子 (SRID)

空間データベースのコンテキストでは、ジオメトリが記述されている定義済みの空間を「空間参照系 (SRS)」 と呼びます。空間参照系では、少なくとも次のことが定義されます。

- 基本となる座標系の測定単位 (角度、メートルなど)
- 座標の最大値と最小値 (境界とも呼ばれます)
- デフォルトの線形測定単位
- データが平面データまたは回転楕円体データのいずれであるか
- データを他の SRS に変換するための投影情報

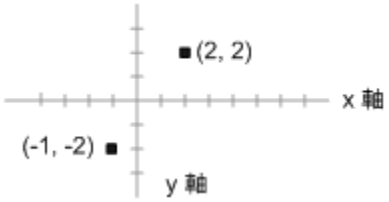
すべての空間参照系には、「空間参照系識別子 (SRID)」 と呼ばれる識別子があります。ジオメトリが別のジオメトリと接触しているかどうかを調べる操作などを SQL Anywhere が実行する

場合、空間参照系の計算を正しく実行できるように、SRID を使用してその空間参照系の定義を検索します。SQL Anywhere データベースでは、各 SRID はユニークである必要があります。

デフォルトでは、SQL Anywhere は次の空間参照系を新しいデータベースに追加します。

- **デフォルト - SRID 0**　これは、ジオメトリを構成するときに SQL に SRID が指定されておらず、ロードされる値にも含まれていない場合のデフォルトの空間参照系です。

  デフォルトは、平らな 2 次元平面でのデータを処理する直交空間参照系です。平面上の任意のポイントは、x と y が -1,000,000 から 1,000,000 の境界を持つ、x, y 座標の単一のペアを使用して定義されます。距離は垂直座標軸を使用して測定されます。この空間参照系には、SRID 「0」 が割り当てられています。

  直交空間参照系は、平面タイプの空間参照系です。

- **WGS 84 - SRID 4326**　WGS 84 標準では、地球の回転楕円体の参照面が提供されます。これは、グローバルポジショニングシステム (GPS) によって使用される空間参照系です。WGS 84 の座標原点は地球の中心であり、±1 メートルの精度であると見なされています。WGS は世界測地系 (World Geodetic System) を表しています。

  WGS 84 の座標は角度で表され、第 1 の座標は経度で境界は -180 から 180、第 2 の座標は緯度で境界は -90 から 90 です。

  WGS 84 は、曲面タイプの空間参照系であり、デフォルトの測定単位は METRE です。

- **WGS 84 (平面) - SRID 1000004326**　WGS 84 (平面) は WGS 84 と似ていますが、正距円筒投影法が使用され、長さ、面積、その他の計算がゆがめられます。たとえば、SRID 4326 と 1000004326 では、どちらも赤道での経度 1 度は約 111 km です。北緯 80 度では、SRID 4326 では経度 1 度は約 19 km ですが、SRID 1000004326 では**すべての**緯度で経度 1 度を約 111 km として扱います。SRID 1000004326 では長さにかなりのゆがみが発生します (10 分の 1 以下に減少)。ゆがみの係数は、赤道に対するジオメトリの相対的な位置によって異なります。このため、SRID 1000004326 は距離と面積の計算には使用しないでください。ST_Contains、ST_Touches、ST_Covers などの関係述部にのみ使用してください。

  WGS 84 (平面) は、平面タイプの空間参照系であり、デフォルトの測定単位は DEGREE です。

  「平面空間参照系の制限」45 ページと「サポートされる空間述部」10 ページを参照してください。

- **sa_planar_unbounded - SRID 2,147,483,646**　内部でのみ使用されます。

- **sa_octahedral_gnomonic - SRID 2,147,483,647**　内部でのみ使用されます。

空間参照系は希望の方法で、および希望の任意の SRID 番号を使用して定義できるため、空間参照系の定義 (投影、座標系など) は、データがデータベース間で移動する場合、または他の SRS に変換されるときにデータに付随している必要があります。たとえば、空間データを WKT にアンロードする場合、空間参照系の定義がファイルの先頭に含まれます。

sa_install_feature システムプロシージャーを使用した追加の空間参照系のインストール

SQL Anywhere には、使用できる定義済みの SRS も多数備えられています。ただし、これらの SRS は、新しいデータベースの作成時にデフォルトではデータベースにインストールされません。これらを追加するには、sa_install_feature システムプロシージャーを使用してください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

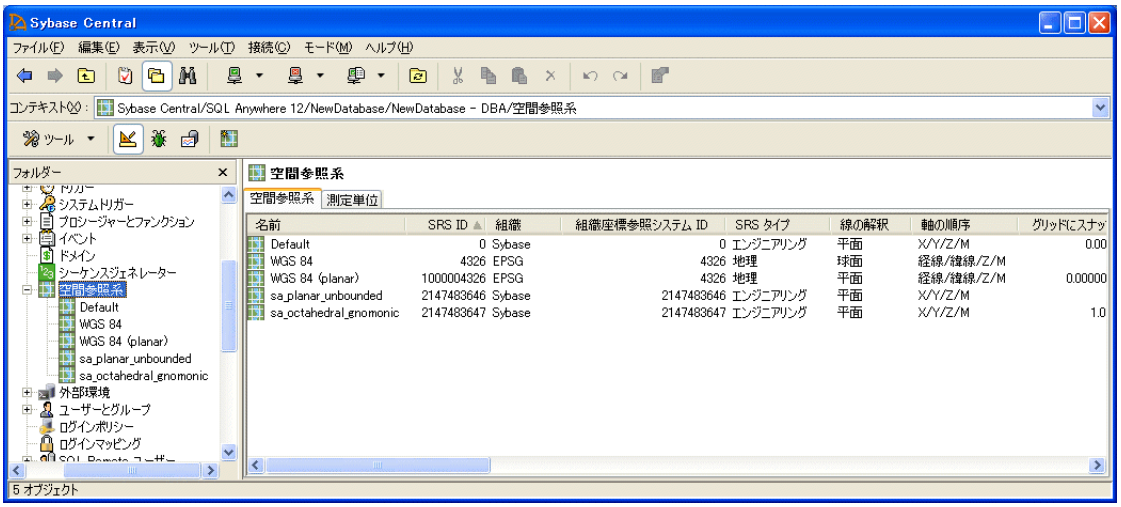
これらの追加の空間参照系の説明については、spatialreference.org と www.epsg-registry.org/を参照してください。

現在データベースにある空間参照系のリストの決定

空間参照系情報は、ISYSSPATIALREFERENCESYSTEM システムテーブルに格納されています。SRS の SRID は、このテーブルのプライマリキー値として使用されます。データベースサーバーは、SRID 値を使用して空間参照系の設定情報を検索し、これにより、設定情報がないと抽象的な空間座標を地球上の実際の位置として解釈できるようになります。

ST_SPATIAL_REFERENCE_SYSTEMS 統合ビューをクエリすることによって、空間参照系のリストを確認できます。このビューの各ローに空間参照系が定義されています。「ST_SPATIAL_REFERENCE_SYSTEMS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

データベースにインストールされている空間参照系のリストは、Sybase Central の **[空間参照系]** フォルダーを参照することによっても表示できます。



一般的なマッピングアプリケーションとの互換性

一般的な Web マッピングおよび視覚化アプリケーション (Google Earth、Bing Maps、ArcGIS Online など) には、地球の球形モデルに基づくメルカトール投影の空間参照系を使用するものがあります。この球形モデルは地球の両極での扁平を無視するため、位置で最大 800 m、縮尺で最大 0.7 パーセントの誤差が生じる可能性がありますが、アプリケーションでの投影実行がより効率的になります。

以前は、この空間参照系には SRID 900913 が市販アプリケーションで割り当てられていました。ただし、EPSG がこの投影を SRID 3857 としてリリースしました。SRID 900913 を要求するアプリケーションと互換性を保つために、以下のことを実行できます。

1. sa_install_feature システムプロシージャーを使用して、SQL Anywhere によって提供される空間参照系をすべてインストールします (SRID 3857 を含む)。
2. dbunload -n を実行して、3857 SRID 定義を取得します。
3. Sybase Central を使用し、アンロードされた SRID 定義の情報を使用して SRID 900913 で空間参照系を作成します。

参照

- 「空間参照系の作成 (Sybase Central の場合)」 40 ページ
- 「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
- 「ST_SPATIAL_REFERENCE_SYSTEMS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』
- 「CREATE SPATIAL REFERENCE SYSTEM 文」『SQL Anywhere サーバー SQL リファレンス』
- 「平面および曲面の表現方法」 44 ページ

測定単位

地理的特性は、緯度、ラジアン、またはその他の角度測定単位で測定できます。空間参照系には、地理的座標が測定される単位の名前を明示的に指定し、指定された単位からラジアンへの変換方法を含める必要があります。

投影座標系を使用している場合、個々の座標値は地球の地表面に沿った、ポイントまでの線形距離を表します。座標値は、メートル、フィート、マイル、またはヤードで測定できます。投影座標系では、座標値を表現する線形測定単位を明示的に指定する必要があります。

次の測定単位は、新しい SQL Anywhere データベースに自動的にインストールされます。

- **meter**　線形測定単位。国際メートルとしても知られています。SI 標準単位です。ISO 1000 で定義されています。
- **metre**　線形測定単位。meter のエイリアス。SI 標準単位です。ISO 1000 で定義されています。
- **radian**　角度測定単位。SI 標準単位です。ISO 1000:1992 で定義されています。
- **degree**　角度測定単位 (pi()/180.0 ラジアン)。

- **planar degree**　線形測定単位。60 海里として定義されています。PLANAR 線解釈を使用する地理的空間参照系で使用される線形測定単位です。

参照

- 「追加の事前定義済み測定単位のインストール」 6 ページ
- 「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
- 「CREATE SPATIAL REFERENCE SYSTEM 文」『SQL Anywhere サーバー SQL リファレンス』
- LINEAR UNIT OF MEASURE 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』
- ANGULAR UNIT OF MEASURE 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』
- 「ST_UNITS_OF_MEASURE 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』

追加の事前定義済み測定単位のインストール

sa_install_feature システムプロシージャーは、デフォルトではインストールされていない、事前定義済みの測定単位を新しいデータベースに追加します。

前提条件

DBA または SYS_SPATIAL_ADMIN_ROLE グループのメンバー

内容と備考

None

♦ 追加の事前定義済み測定単位のインストール

- 次の文を実行して、すべての事前定義済みの測定単位をインストールします。

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

結果

追加の測定単位はすべて、インストールされます。

次の手順

測定単位を使用する空間参照系を作成できます。

これらの追加の測定単位についての説明については、www.epsg-registry.org/を参照してください。この Web ページで、測定単位の名前を **[Name]** フィールドに入力し、**[Type]** フィールドで **Unit of Measure (UOM)** を選択して、**[Search]** をクリックします。

参照

- 「測定単位」5 ページ
- 「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
- 「CREATE SPATIAL REFERENCE SYSTEM 文」『SQL Anywhere サーバー SQL リファレンス』
- LINEAR UNIT OF MEASURE 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』
- ANGULAR UNIT OF MEASURE 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』
- 「ST_UNITS_OF_MEASURE 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』

SQL Anywhere による空間データのサポート

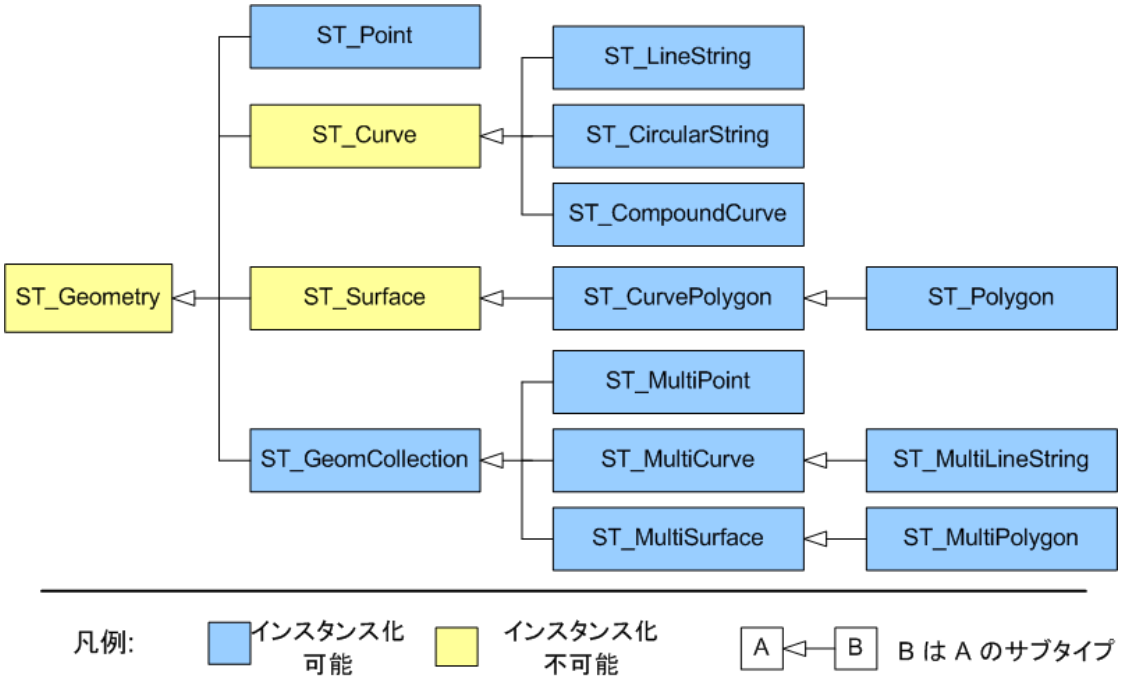
以下の項では、SQL Anywhere による空間データのサポートについて説明します。

サポートされる空間データ型とその階層

SQL Anywhere は、地理空間データの格納とアクセスにおいて SQL Multimedia (SQL/MM) 標準に準拠しています。この標準の重要な構成要素は、ST_Geometry 型の階層を使用して、地理空間データの作成方法を定義していることです。階層内では、プレフィクス ST がすべてのデータ型 (クラスまたはタイプとも呼ばれます) に対して使用されます。

カラムが特定のタイプとして識別されると、そのタイプとそのサブクラスの値をカラムに格納できます。たとえば、ST_GeomCollection として識別されたカラムには、ST_MultiPoint、ST_MultiSurface、ST_MultiCurve、ST_MultiPolygon、ST_MultiLineString の値も格納できます。

次の図は、ST_Geometry データ型とそのサブタイプの階層を示しています。



左側のタイプがスーパータイプ (または基本タイプ)、右側がそのサブタイプ (または派生タイプ) を示します。

サポートされる空間データ型の説明

SQL Anywhere では、次の空間データ型がサポートされています。

- **ポイント**　ポイントは空間内の単一の場所を定義します。ポイントジオメトリには長さまたは面積がありません。ポイントには必ず X 座標と Y 座標があります。

  空でないポイントに対しては、ST_Dimension は 0 を返します。

  GIS データでは、ポイントは、通常、住所などの場所、または山などの地理的特性を表すために使用されます。

  「ST_Point タイプ」314 ページを参照してください。

- **線ストリング**　線ストリングは長さを持つジオメトリですが、領域はありません。空でない線ストリングに対しては、ST_Dimension は 1 を返します。線ストリングは、単純か、単純ではないか、および閉じているか、閉じていないかによって特徴付けることができます。「単純」とは、それ自体が交差していない線ストリングを指します。「閉じている」とは、開始したポイントと同じポイントで終了する線ストリングを意味します。たとえば、リングは単純で閉じている線ストリングの例です。

  GIS データでは、線ストリングは、通常、河川、道路、または配送経路を表すために使用されます。

  「ST_LineString タイプ」264 ページを参照してください。

- **多角形**　多角形は空間の領域を定義します。多角形は、外部領域を定義する 1 つの外部境界リングと、領域の穴を定義する 0 個以上の内部リングによって構成されます。多角形には領域が関連付けられますが、長さはありません。

  空でない多角形に対しては、ST_Dimension は 2 を返します。

  GIS データでは、多角形は、通常、地域 (国、町、州など)、湖、公園などの大きな地理的特性を表すために使用されます。

  「多角形リングの方向操作」51 ページと「ST_Polygon タイプ」331 ページを参照してください。

- **円ストリング**　円ストリングは、一連の円弧セグメントを接続したものです。ポイント間が円弧で接続された線セグメントとよく似ています。

  「ST_CircularString タイプ」71 ページを参照してください。

- **複合曲線**　複合曲線は、円ストリングまたは線ストリングを接続したものです。

  「ST_CompoundCurve タイプ」79 ページを参照してください。

- **曲線多角形**　曲線多角形は、一般的な多角形であり、円弧の境界セグメントが含まれる場合もあります。

  「ST_CurvePolygon タイプ」93 ページを参照してください。

- **ジオメトリ**　ジオメトリという語は、ポイント、線ストリング、多角形などのオブジェクトを包含するタイプを意味します。ジオメトリタイプは、サポートされるすべての空間データ型のスーパータイプです。

  「ST_Geometry タイプ」112 ページを参照してください。

- **ジオメトリコレクション**　ジオメトリコレクションは、1 つ以上のジオメトリ (ポイント、線、多角形など) のコレクションです。

  「ST_GeomCollection タイプ」104 ページを参照してください。

- **複数ポイント**　複数ポイントは個々のポイントのコレクションです。

  GIS データでは、複数ポイントは、通常、ロケーションのセットを表すために使用されます。

  「ST_MultiPoint タイプ」288 ページを参照してください。

- **複数多角形**　複数多角形は 0 個以上の多角形の集合体です。

  GIS データでは、複数多角形は、複数の島で形成されている国の領土や、湖水系などの地理的特性 を表現するためによく使用されます。

  「ST_MultiPolygon タイプ」294 ページを参照してください。

- **複数線ストリング**　複数線ストリングは線ストリングの集合体です。

GIS データでは、複数線ストリングは、河川または高速道路網のような地理的特性を表すために使用されます。

「ST_MultiLineString タイプ」281 ページを参照してください。

- **複数面**　複数面は曲線多角形の集合体です。

「ST_MultiSurface タイプ」302 ページを参照してください。

空間データ型のオブジェクト指向型プロパティ

- サブタイプ (または派生タイプ) は、スーパータイプ (または基本タイプ) よりも限定的です。たとえば、ST_LineString は ST_Curve をより限定的にしたものです。
- サブタイプはすべてのスーパータイプのすべてのメソッドを継承します。たとえば、ST_Polygon の値は、スーパータイプ ST_Geometry、ST_Surface、ST_CurvePolygon のメソッドを呼び出せます。
- サブタイプの値は自動的に派生元のスーパータイプの値に変換されます。たとえば、point1.ST_Distance(point2) のように、ST_Geometry のパラメーターが必要な場面では、ST_Point の値を使用できます。
- カラムまたは変数には、どのサブタイプの値でも格納できます。たとえば、ST_Geometry (SRID=4326) タイプのカラムには、すべてのタイプの空間値を格納できます。
- 宣言されたタイプを使用したカラム、変数、または式は、そのままの型で処理できます。そうでない場合は、サブタイプにキャストします。たとえば、ある geom という名前の ST_Geometory のカラムの ST_Polygon の値を、TREAT 関数で宣言した ST_Surface タイプに変更することで、TREAT(geom AS ST_Surface).ST_Area() のように、ST_Area メソッドを呼び出すことができます。

参照

- 「TREAT 関数 [データ型変換]」『SQL Anywhere サーバー SQL リファレンス』
- 「CAST 関数 [データ型変換]」『SQL Anywhere サーバー SQL リファレンス』

サポートされる空間述部

述部は条件式であり、論理演算子 AND や OR と組み合わせて、WHERE 句、HAVING 句、ON 句、IF 式、CASE 式、または CHECK 制約に一連の条件を構成します。SQL では、述部は TRUE または FALSE に評価できます。多くのコンテキストで、UNKNOWN と評価される述部が FALSE として解釈されます。

空間述部は、0 または 1 を返すメンバー関数として実装されます。空間述部をテストするには、クエリで = または <> 演算子を使用して、関数の結果を 1 または 0 で比較します。次に例を示します。

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

述部を使用するのは、次のような質問で空間データを問い合わせる場合です。たとえば、2 つ以上のジオメトリの距離がどのくらい近いか、それらは交差しているのか、重なり合っているか、

あるいは、あるジオメトリが別のジオメトリ内に含まれているか、などがあります。たとえば、配送会社の場合、述部を使用して、顧客が特定の配送区域内に存在しているかどうかを判別できます。

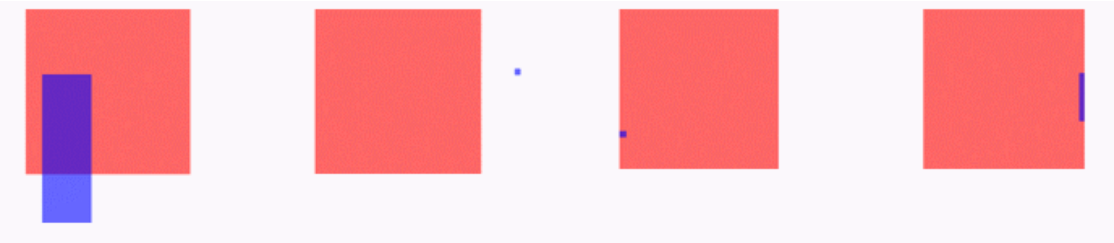
SQL Anywhere では、次の空間述部をサポートし、ジオメトリ間の空間の関係についての問い合わせに簡単に答えることができます。

- ST_Geometry タイプの ST_Contains メソッド
- ST_Geometry タイプの ST_Covers メソッド
- ST_Geometry タイプの ST_CoveredBy メソッド
- ST_Geometry タイプの ST_Crosses メソッド
- ST_Geometry タイプの ST_Disjoint メソッド
- ST_Geometry タイプの ST_IsEmpty メソッド
- ST_Geometry タイプの ST_Equals メソッド
- ST_Geometry タイプの ST_Intersects メソッド
- ST_Geometry タイプの ST_OrderingEquals メソッド
- ST_Geometry タイプの ST_Overlaps メソッド
- ST_Geometry タイプの ST_Relate メソッド
- ST_Geometry タイプの ST_Touches メソッド
- ST_Geometry タイプの ST_IsValid メソッド
- ST_Geometry タイプの ST_Within メソッド

空間述部の直感性

述部の結果は直感的ではないことがあるため、特殊なケースをテストして、望みどおりの結果を得られていることを確認する必要があります。たとえば、ジオメトリが別のジオメトリを包含するには (a.ST_Contains(b)=1)、またはジオメトリが別のジオメトリ内に含まれるためには (b.ST_Within(a)=1)、a の内部と b の内部は交差している必要があり、b のどの部分も a の外部と交差することはできません。ただし、ジオメトリが別のジオメトリ内に含まれていることを予想していたが、含まれていない場合があります。

たとえば、次の例では、a.ST_Contains(b) と b.ST_Within(a) (a が赤) に対して 0 が返されます。



ケース 1 と 2 は明白です。紫色のジオメトリは赤い正方形内に完全には含まれていません。ケース 3 と 4 は明らかではありません。これらのケースは、紫色のジオメトリが赤い正方形の境界上にのみあります。紫色のジオメトリは赤い正方形内にあるように見えますが、ST_Contains は赤い正方形内にあるとは見なしません。

ST_Covers と ST_CoveredBy は、ST_Contains と ST_Within と類似した述部です。ST_Covers と ST_CoveredBy の場合、2 つのジオメトリの内部が交差している必要がない点が異なります。ま

た、ST_Covers と ST_CoveredBy が返す結果は、ST_Contains と ST_Within より直感的でわかりやすい場合が多いです。

述部のテストで望んでいたものと異なる結果が返された場合、ST_Relate メソッドを使用して、テストする関係を厳密に指定することを検討してください。

参照

- 「空間の関係操作」55 ページ

空間標準への準拠

SQL Anywhere の空間は、次の標準に準拠しています。

- **国際標準化機構 (ISO)** SQL Anywhere のジオメトリは、空間のユーザーデータ型、ルーチン、スキーマの定義と空間データの処理において、ISO 標準に準拠しています。SQL Anywhere は、国際標準 ISO/IEC 13249-3:2006 に記述されている特定の推奨事項に準拠しています。詳細については、http://www.iso.org/iso/catalogue_detail.htm?csnumber=38651 を参照してください。

- **OGC (Open Geospatial Consortium) ジオメトリモデル** SQL Anywhere のジオメトリは、「OGC OpenGIS Implementation Specification for Geographic information - Simple feature access - Part 2: SQL option version 1.2.0 (OGC 06-104r3)」に準拠しています。http://www.opengeospatial.org/standards/sfs を参照してください。

  SQL Anywhere では、OGC 推奨の標準を使用して、空間情報をさまざまなベンダーおよびアプリケーション間で共有できるようにしています。

  SQL Anywhere の空間ジオメトリと互換性を確保するには、OGC によって指定されている標準に準拠することをおすすめします。

- **SQL Multimedia (SQL/MM)** SQL Anywhere は SQL/MM 標準に準拠し、すべてのメソッド名と関数名にプレフィクス **ST_** を使用しています。

  SQL/MM は、SQL を使用して空間データを格納、検索、処理する方法を定義した国際標準です。空間データ型の階層 (ST_Geometry など) は、空間データの検索に使用される方法の 1 つです。ST_Geometry 階層には、ST_Point、ST_Curve、ST_Polygon などの多くのサブタイプが含まれています。SQL/MM 標準では、クエリに含まれるすべての空間値は、同じ空間参照系に定義されている必要があります。

サポートと準拠についての特記事項

この項では、SQL Anywhere の空間データサポートの特記事項 (サポートされない機能、他のデータベース製品との顕著な動作の違いなど) について説明します。

- **ジオグラフィとジオメトリ** ベンダーによっては、「ジオグラフィ」 (曲面上のオブジェクトに関連する) か、「ジオメトリ」 (平面上のオブジェクト) かによって、空間オブジェクトが区

別されます。SQL Anywhere では、空間オブジェクトはすべてジオメトリと見なされ、オブジェクトの SRID によって、曲面空間参照系または平面空間参照系で処理されていることが示されます。

- **サポートされないメソッド**
  - ST_Buffer メソッド
  - ST_LocateAlong メソッド
  - ST_LocateBetween メソッド
  - ST_Segmentize メソッド
  - ST_Simplify メソッド
  - ST_Distance_Spheroid メソッド
  - ST_Length_Spheroid メソッド

サポートされる空間データのインポートフォーマットとエクスポートフォーマット

次の表に、SQL Anywhere によってサポートされる、空間データのインポートおよびエクスポートのためのデータフォーマットとファイルフォーマットをリストします。

データフォーマット	インポート	エクスポート	説明
WKT (Well Known Text)	○	○	ASCII テキストで表現された地理的データ。このフォーマットは、地理的情報の OpenGIS 実装仕様に定義されている単純特性の一部として OGC (Open Geospatial Consortium) によって保守されています。www.opengeospatial.org/standards/sfa を参照してください。 WKT でポイントを表す方法の例を次に示します。 'POINT(1 1)'

データフォーマット	インポート	エクスポート	説明
WKB (Well Known Binary)	○	○	バイナリストリームとして表現された地理的データ。このフォーマットは、地理的情報の OpenGIS 実装仕様に定義されている単純特性の一部として OGC によって保守されています。www.opengeospatial.org/standards/sfa を参照してください。 WKB でポイントを表す方法の例を次に示します。 '0101000000000000000000F03F0000000000000F03F'
EWKT (拡張 Well Known Text)	○	○	SRID 情報が埋め込まれた WKT フォーマット。このフォーマットは、PostGIS (PostgreSQL の空間データベース拡張) の一部として管理されます。postgis.refractions.net/ を参照してください。 EWKT でポイントを表す方法の例を次に示します。 'srid=101;POINT(1 1)'

データフォーマット	インポート	エクスポート	説明
EWKB (拡張 Well Known Binary)	○	○	SRID 情報が埋め込まれた WKB フォーマット。このフォーマットは、PostGIS (PostgreSQL の空間データベース拡張) の一部として管理されます。postgis.refractions.net/ を参照してください。 EWKB でポイントを表す方法の例を次に示します。 '0101000002004000000000000000F03F000000000000F03F'
GML (Geographic Markup Language)	×	○	地理的空間データを表すために使用される XML 文法。この標準は OGC (Open Geospatial Consortium) によって保守されており、インターネットを介した地理的データの交換を目的としています。www.opengeospatial.org/standards/gml を参照してください。 GML でポイントを表す方法の例を次に示します。 <gml:Point> <gml:coordinates>1,1</gml:coordinates> </gml:Point>

データフォーマット	インポート	エクスポート	説明
KML	×	○	以前の Google の Keyhole Markup Language です。この XML 文法は地理的データを表すために使用され、視覚化、ナビゲーション補助、地図とイメージに注釈を付ける機能を含んでいます。Google はこの標準を OGC に提案しました。OGC はこれをオープン標準として受け入れ、現在は KML と呼ばれています。www.opengeospatial.org/standards/kml を参照してください。 KML でポイントを表す方法の例を次に示します。 <Point> <coordinates>1,0</ coordinates> </Point>
ESRI シェイプファイル	○	×	シェイプファイル (シェイプを定義するために一緒に使用される複数のファイル) の形式で空間オブジェクトを表すための一般的な地理空間ベクトルデータフォーマット。ESRI シェイプファイルサポートの詳細については、「ESRI シェイプファイルのサポート」 18 ページを参照してください。

データフォーマット	インポート	エクスポート	説明
GeoJSON	×	○	名前／値ペア、値の順序付きリスト、一般的なプログラミング言語 (C、C++、C#、Java、JavaScript、Perl、Python など) で使用される規則に類似した規則を使用するテキストフォーマット。 GeoJSON は JSON 標準のサブセットであり、地理的情報をコード化するために使用されます。SQL Anywhere では、GeoJSON 標準をサポートしており、SQL 出力を GeoJSON フォーマットに変換するための ST_AsGeoJSON メソッドを提供しています。ST_Geometry タイプの ST_AsGeoJSON メソッド123 ページを参照してください。 GeoJSON でポイントを表す方法の例を次に示します。 {"x" : 1, "y" : 1, "spatialReference" : {"wkid" : 4326}} GeoJSON 仕様の詳細については、geojson.org/geojson-spec.html を参照してください。

データフォーマット	インポート	エクスポート	説明
SVG (Scalable Vector Graphic) ファイル	×	○	2 次元のジオメトリを表すために使用される XML ベースのフォーマット。SVG フォーマットは W3C (World Wide Web Consortium) によって保守されています。www.w3.org/Graphics/SVG/を参照してください。 SVG でポイントを表す方法の例を次に示します。 <rect width="1" height="1" fill="deepskyblue" stroke="black" stroke-width="1" x="1" y="-1"/>

ESRI シェイプファイルのサポート

SQL Anywhere では、ESRI (Environmental System Research Institute) のシェイプファイルフォーマットをサポートします。ESRI シェイプファイルは、データセット内の空間機能のジオメトリデータと属性情報を格納するために使用されます。

ESRI シェイプファイルには、少なくとも異なる種類の 3 つのファイル (*.shp*、*.shx*、*.dbf*) が含まれています。メインファイルの拡張子は *.shp*、インデックスファイルの拡張子は *.shx*、属性カラムの拡張子は *.dbf* です。すべてのファイルは同じベース名を共有し、多くの場合単一の圧縮ファイルにまとめられます。SQL Anywhere では、MultiPatch を除くすべてのシェイプタイプの ESRI シェイプファイルを読み込むことができます。これには、Z データと M データを含んだシェイプタイプが含まれます。

ESRI シェイプファイル内のデータには、通常、複数のローとカラムが含まれています。たとえば、空間チュートリアルでは、マサチューセッツ州の郵便番号区域が含まれるシェイプファイルをロードします。このシェイプファイルには、郵便番号区域ごとに 1 つのローがあり、ローにはその区域の多角形情報が含まれています。また、各郵便番号区域の追加の属性 (カラム) も含んでいます。これには、郵便番号名 (たとえば、文字列「02633」) やその他の属性が含まれています。

テーブルにシェイプファイルをロードする最も簡単な方法は、Interactive SQL の**インポートウィザード**または st_geometry_load_shapefile システムプロシージャーを使用することです。どちらのツールを使用しても、適切なカラムのテーブルが作成され、シェイプファイルのデータがロードされます。

また、LOAD TABLE 文と INPUT 文を使用することによってもシェイプファイルをロードできますが、その場合、ロード操作を実行する前に、適切なカラムのテーブルが作成されている必要があります。

LOAD TABLE 文または INPUT 文を使用してデータをロードするときに必要なカラムを知るために、sa_describe_shapefile システムプロシージャーを使用できます。

シェイプファイルの SQL Anywhere データベースへのロードの実例については、「チュートリアル：空間機能の実験」59 ページを参照してください。

シェイプファイル内でのクエリの詳細については、FROM 句の Openstring 式『SQL Anywhere サーバー SQL リファレンス』を参照してください。

ESRI シェイプファイルの詳細については、http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf を参照してください。

参照

- 「インポートウィザード (Interactive SQL) でのデータのインポート」『SQL Anywhere サーバー SQL の使用法』
- 「st_geometry_load_shapefile システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
- 「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』
- 「sa_describe_shapefile システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
- 「INPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』

空間トピック関連の推奨ドキュメント

- 地表面のマッピングおよび測定 (測地学) に使用される別の方法についての入門書、座標 (または空間) 参照系に関連する主要な概念については、www.epsg.org/guides/index.html にアクセスして、[Geodetic Awareness] を選択してください。
- 地理的情報に関する OGC OpenGIS 実装仕様 - 単純地物アクセス：www.opengeospatial.org/standards/sfs www.opengeospatial.org/standards/sfs
- 国際標準 ISO/IEC 13249-3:2006：www.iso.org/iso/catalogue_detail.htm?csnumber=38651 www.iso.org/iso/catalogue_detail.htm?csnumber=38651
- SVG (Scalable Vector Graphics) 1.1 仕様：www.w3.org/TR/SVG11/index.html www.w3.org/TR/SVG11/index.html
- GML (Geographic Markup Language) 仕様：www.opengeospatial.org/standards/gml www.opengeospatial.org/standards/gml
- KML 仕様：www.opengeospatial.org/standards/kml www.opengeospatial.org/standards/kml
- JavaScript Object Notation (JSON)：json.org json.org
- GeoJSON 仕様： geojson.org/geojson-spec.html

空間データのクイックスタート

この項では、空間データの作成、アクセス、操作を行う手順を説明します。

空間カラムの作成 (Sybase Central の場合)

空間データをサポートするカラムを追加して、空間データを任意のテーブルに追加できます。

前提条件

テーブルを変更するには、パーミッションが付与されたユーザーである必要があります。

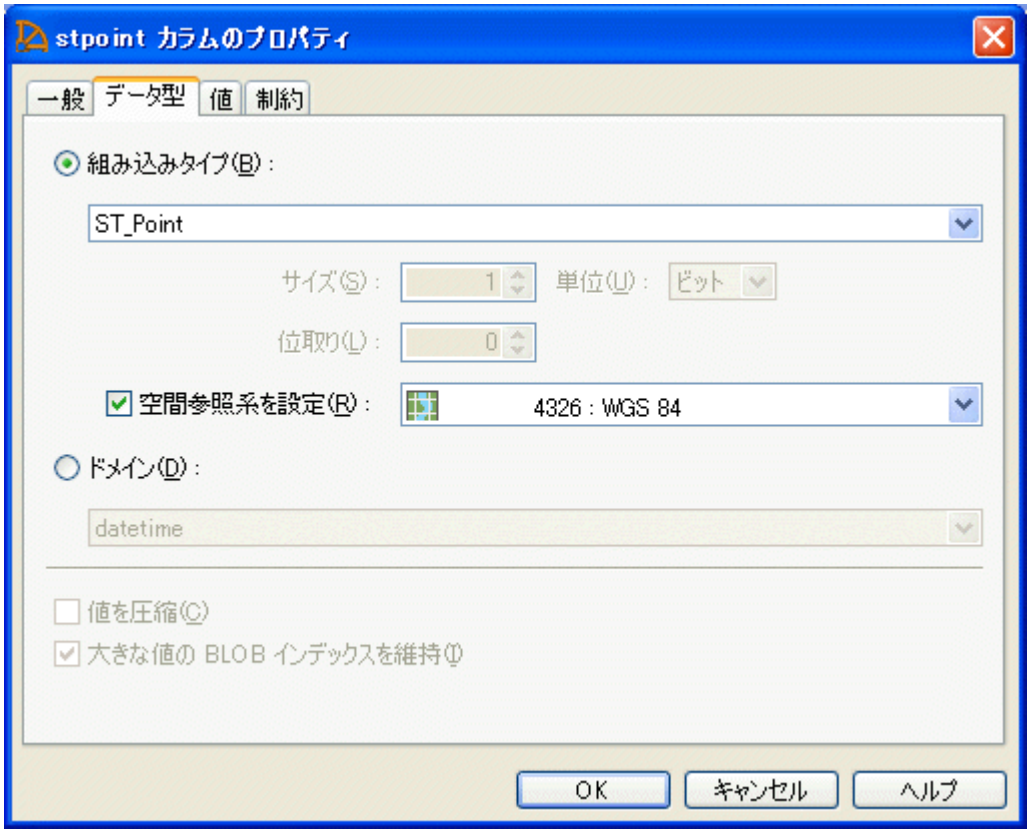
テーブルには、定義されたプライマリキーがあります。プライマリキーが定義されていない場合、空間カラムを含んでいるテーブルに対して、更新操作と削除操作がサポートされません。

内容と備考

次のとおりです。

♦ 空間カラムの作成 (Sybase Central の場合)

1. パーミッションが付与されたユーザーとしてデータベースに接続し、テーブルを変更します。
2. 新しいカラムを作成します。
   - 左ウィンドウ枠で、**[テーブル]** リストを展開します。
   - テーブルを右クリックし、**[新規]** » **[カラム]** をクリックします。
3. 空間データ型を設定します。
   a. [カラム名] を右クリックし、**[プロパティ]** をクリックします。
   b. **[データ型]** タブをクリックします。
   c. **[組み込みタイプ]** を選択し、ドロップダウン リストから空間データ型を選択します。
   d. **[空間参照系の設定]** を選択し、ドロップダウン リストから空間参照系を選択します。



e. **[OK]** をクリックします。

4. **[ファイル]** » **[保存]** をクリックします。

結果

空間カラムが、既存のテーブルに追加されます。

次の手順

ALTER TABLE 文を使用して、カラムに SRID 制約を課し、空間カラムに保存できる値を制限できます。空間データをカラムに追加することもできます。

参照

- 「サポートされる空間データ型とその階層」 7 ページ
- 「空間参照系 (SRS) と空間参照系識別子 (SRID)」 2 ページ
- 「ALTER TABLE 文」『SQL Anywhere サーバー SQL リファレンス』
- 「SRID カラム制約の追加」 22 ページ

SRID カラム制約の追加

SRID 制約を使用すると、空間カラムに格納できる値に制約を加えることができます。インデックスに空間カラムを含めるには、カラムに SRID 制約を課す必要があります。この制約は、CREATE TABLE 文と ALTER TABLE 文を使用して追加できます。

前提条件

テーブルを作成または変更するには、パーミッションが付与されたユーザーである必要があります。

テーブルに空間カラムを追加する場合、テーブルにプライマリキーが定義されていることを確認してください。プライマリキーが定義されていない場合、空間カラムを含んでいるテーブルに対して、更新操作と削除操作がサポートされません。

内容と備考

空間カラムにはプライマリーキー、ユニークインデックス、または一意性制約を入れることはできません。

♦ SRID カラム制約の追加

- 空間カラムの SRID 制約を含んだ CREATE TABLE 文または ALTER TABLE 文を実行します。

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

結果

SRID 制約が、テーブルの空間カラムに追加されます。

次の手順

空間カラムをインデックスに含めることができます。

例

たとえば、次の文を実行して、Geometry_2 カラムに SRID 制約 (SRID=4326) を持つ Test という名前のテーブルを作成します。

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

この制約は、このカラムに格納できるのは、SRID 4326 に関連付けられた値のみであることを意味します。

カラム Geometry_1 には制約はなく、任意の SRID に関連付けられた値を格納できます。

Geometry_1 カラムにはインデックスを作成できません。ただし、Geometry_2 カラムにはインデックスを作成できます。

既存の空間カラムを持つテーブルの場合は、ALTER TABLE 文を使用して、空間カラムに SRID 制約を追加できます。たとえば、次のような文を実行して、Test という名前のテーブルの Geometry_1 カラムに制約を追加します。

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

参照

- 「CREATE INDEX 文」『SQL Anywhere サーバー SQL リファレンス』
- 「CREATE TABLE 文」『SQL Anywhere サーバー SQL リファレンス』
- 「ALTER TABLE 文」『SQL Anywhere サーバー SQL リファレンス』
- 「空間カラムの作成 (Sybase Central の場合)」20 ページ
- 「空間カラムの作成 (Interactive SQL の場合)」23 ページ

空間カラムの作成 (Interactive SQL の場合)

空間データをサポートするカラムを追加して、空間データを任意のテーブルに追加できます。

前提条件

テーブルを変更するには、パーミッションが付与されたユーザーである必要があります。

内容と備考

次のとおりです。

♦ 空間カラムの作成 (Interactive SQL の場合)

1. パーミッションが付与されたユーザーとしてデータベースに接続し、テーブルを変更します。
2. ALTER TABLE 文を実行します。

結果

空間カラムが、既存のテーブルに追加されます。

次の手順

カラムに SRID 制約を課し、空間カラムに保存できる値を制限できます。

例

次の文は、Location という名前の空間カラムを Customers テーブルに追加します。新しいカラムは空間データ型 ST_Point であり、宣言された 1000004326 (平面空間参照系) という SRID を持ちます。

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

参照

- 「サポートされる空間データ型とその階層」7 ページ
- 「SRID カラム制約の追加」22 ページ
- 「空間参照系 (SRS) と空間参照系識別子 (SRID)」2 ページ
- 「ALTER TABLE 文」『SQL Anywhere サーバー SQL リファレンス』

空間カラムでのインデックスの作成

空間インデックスを作成するときは、他のデータ型のインデックスの作成と同様に、CREATE INDEX 文または **テキストインデックス作成ウィザード**を使用します。ただし、空間データに対してインデックスを作成する場合は、複数の空間カラムをインデックスに含めないことと、空間カラムをインデックス定義の最後に配置することをおすすめします。

また、空間カラムをインデックスに含めるには、カラムに SRID 制約を付ける必要があります。

空間データにインデックスを作成すると、ジオメトリ間の関係を評価する場合のコストを低減できます。たとえば、販売区域の境界を変更することを検討しており、そのことが既存の顧客に与える影響を判別するとします。提案された販売区域内に存在する顧客を判別するには、ST_Within メソッドを使用して、各顧客の住所を表すポイントと販売区域を表す多角形を比較します。インデックスがない場合、データベースサーバーは、Customer テーブルのすべての住所のポイントを販売区域の多角形に対してテストし、結果で返すかどうか判別します。この操作は、Customer テーブルが大きい場合は高コストになり、販売区域が小さい場合は非効率になります。各顧客の住所のポイントが含まれるインデックスを使用すると、より短時間で結果を返すことができます。販売区域とそれに重なり合う州を関連付ける述部をクエリに追加できる場合、州コードと住所ポイントの両方が含まれるインデックスを使用して、結果をより短時間で取得できることがあります。

空間クエリは、クラスタードインデックスによって効率がよくなることがありますが、テーブルのその他の用途を考慮してから、クラスタードインデックスの使用を決定する必要があります。実行される可能性のあるクエリのタイプを検討およびテストして、クラスタードインデックスによってパフォーマンスが向上するかどうかを判別します。

空間カラムに対してテキストインデックスを作成できますが、通常のインデックスと比較して利点はありません。代わりに、通常のインデックスの使用をおすすめします。

注意
空間カラムにはプライマリーキー、ユニークインデックス、または一意性制約を入れることはできません。

参照

- 「インデックス」『SQL Anywhere サーバー SQL の使用法』
- 「インデックスの作成」『SQL Anywhere サーバー SQL の使用法』
- 「CREATE INDEX 文」『SQL Anywhere サーバー SQL リファレンス』
- 「ALTER INDEX 文」『SQL Anywhere サーバー SQL リファレンス』
- 「クラスタードインデックスの宣言」『SQL Anywhere サーバー SQL の使用法』
- 「SRID カラム制約の追加」22 ページ

空間データ型構文の理解

SQL/MM 標準では、ANSI/SQL CREATE TYPE 文に基づいて構築される拡張ユーザー定義型 (UDT) として空間データのサポートが定義されています。SQL Anywhere ではユーザー定義型はサポートされませんが、サポートされているかのように SQL Anywhere の空間データサポートが実装されています。

UDT インスタンスのインスタンス化

次のように、コンストラクターを呼び出すことによって、ユーザー定義型の値をインスタンス化できます。

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

たとえば、クエリに次のように指定して、2 つの ST_Point 値をインスタンス化できます。

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

SQL Anywhere では、通常のオーバーロード解決ルールを使用して、*argument-list* を定義済みコンストラクターと照合します。次の状況では、エラーが返されます。

- NEW がユーザー定義型ではないタイプで使用された場合
- ユーザー定義型がインスタンス化可能ではない場合 (たとえば、ST_Geometry はインスタンス化可能なタイプではありません)
- 指定された引数型と一致するオーバーロードがない場合

次の項を参照してください。

- 「空間データへのアクセスとそのデータの操作」
- 「ST_Point タイプ」

インスタンスメソッドの使用

ユーザー定義型には、インスタンスメソッドが定義されていることがあります。インスタンスメソッドは、次のようにしてタイプの値に対して呼び出されます。

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

たとえば、次の架空の例では、Massdata.CenterPoint カラムの X 座標を SELECT しています。

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

CenterPoint というユーザー ID がある場合、データベースサーバーでは CenterPoint.ST_X() が「あいまい」であると見なされます。これは、この文が「ユーザー CenterPoint が所有するユーザー定義関数 ST_X を呼び出す」(この文の意図しない意味) とも、「Massdata.CenterPoint カラムの ST_X メソッドを呼び出す」(正しい意味) とも考えられるためです。データベースサーバーは、最初に CenterPoint という名前のユーザーに対して、大文字小文字を区別した検索を実行することによって、このようなあいまいさを解決します。ユーザーが見つかった場合、データベースサーバーは、ユーザー CenterPoint が所有する ST_X というユーザー定義関数を呼び出していると見なして処理を続行します。見つからなかった場合には、データベースサーバーは、構成体をメソッド呼び出しとして処理し、Massdata.CenterPoint カラムの ST_X メソッドを呼び出します。

次の場合、インスタンスメソッド呼び出しはエラーを返します。

- *value-expression* の宣言されたタイプがユーザー定義型ではない
- *value-expression* またはそのスーパータイプの 1 つの宣言されたタイプに、名前付きメソッドが定義されていない
- *argument-list* が、名前付きメソッドに対して定義されたオーバーロードのいずれとも一致しない

次の項を参照してください。

- 「ST_Point タイプの ST_X() メソッド」

静的メソッドの使用

ANSI/SQL 標準では、インスタンスメソッドに加えて、ユーザー定義型に静的メソッドを関連付けることができます。静的メソッドは次の構文を使用して呼び出されます。

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

たとえば、次の文はテキストを解析することによって ST_Point をインスタンス化します。

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

次の場合、静的メソッド呼び出しはエラーを返します。

- *value-expression* の宣言されたタイプがユーザー定義型ではない
- *value-expression* またはそのスーパータイプの 1 つの宣言されたタイプに、名前付きメソッドが定義されていない
- *argument-list* が、名前付きメソッドに対して定義されたオーバーロードのいずれとも一致しない

次の項を参照してください。

- 「ST_Point タイプ」
- ST_Geometry タイプの ST_GeomFromText メソッド

静的集約メソッドの使用 (SQL Anywhere 拡張)

ANSI/SQL の拡張として、SQL Anywhere では、ユーザー定義の集約を実装する静的メソッドをサポートしています。次に例を示します。

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

静的メソッドのオーバーロードは、すべてが集約であるか、またはすべてが集約ではないかのいずれかである必要があります。

次の場合、静的集約メソッド呼び出しはエラーを返します。

●静的メソッド呼び出しがエラーを返す場合

●組み込み集合関数がエラーを返す場合

●WINDOW 句が指定されている場合

次の項を参照してください。

- ST_Geometry タイプの ST_AsSVGAggr メソッド

型述部の使用

ANSI/SQL 標準では、文で値の具象タイプ (他の言語ではオブジェクトタイプとも呼ばれる) を検査できる型述部が定義されています。構文は次のとおりです。

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

value が NULL の場合、述部は UNKNOWN を返します。それ以外の場合、*value* の具象タイプが *type-name* リストの各要素と比較されます。ONLY が指定されている場合、具象タイプが指定されたタイプと同一である場合に一致と見なされます。それ以外の場合は、具象タイプが指定されたタイプまたは派生タイプ (サブタイプ) である場合に一致と見なされます。

value の具象タイプがリスト内のいずれかの要素と一致した場合は TRUE が返され、それ以外の場合は FALSE が返されます。

次の例では、Shape カラムの値に具象タイプ ST_Curve またはそのサブタイプの 1 つ (ST_LineString、ST_CircularString、または ST_CompoundCurve) が含まれているすべてのローを返します。

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

次の項を参照してください。

- 「探索条件」『SQL Anywhere サーバー SQL リファレンス』
- 「ST_Point タイプ」
- ST_Geometry タイプの ST_GeomFromText メソッド

サブタイプへの TREAT 式の使用

ANSI/SQL 標準では、式の宣言されたタイプをスーパータイプからサブタイプに効率的に変換できる、サブタイプ処理式が定義されています。式の具象タイプ (別の言語ではオブジェクトタイ

プとも呼ばれる) が指定したサブタイプ、または指定したサブタイプのサブタイプであることがわかっている場合には、この式を使用できます。CAST 関数で値のコピーが作成されるのに対して、TREAT ではコピーが作成されないため、これは CAST 関数を使用するよりも効率的な方法です。構文は次のとおりです。

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

エラー状況が発生しない場合、結果は *target-subtype* で宣言されたタイプの *value-expression* となります。

次の場合、サブタイプ処理式はエラーを返します。

●*value-expression* がユーザー定義型ではない場合

●*target-subtype* が *value-expression* の宣言されたタイプのサブタイプではない場合

●*value-expression* の動的タイプが *target-subtype* のサブタイプではない場合

次の例では、ST_Geometry の Shape カラムの宣言されたタイプを効率的な ST_Curve サブタイプに変更することによって、ST_Curve タイプの ST_Length メソッドを呼び出せるようにしています。

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

次の項を参照してください。

- 「TREAT 関数 [データ型変換]」『SQL Anywhere サーバー SQL リファレンス』
- 「ST_Point タイプ」
- ST_Geometry タイプの ST_GeomFromText メソッド

ジオメトリの作成

データベースにジオメトリを作成するには、いくつかの方法があります。

- **WKT (Well Known Text) フォーマットまたは WKB (Well Known Binary) フォーマットからのロード**　WKT フォーマットまたは WKB フォーマットのデータをロードまたは挿入できます。これらのフォーマットは OGC によって定義されており、すべての空間データベースベンダーがサポートしています。SQL Anywhere では、これらのフォーマットからジオメトリタイプへの自動的な変換が実行されます。 WKT からのロードの例については、「WKT (Well Known Text) ファイルからの空間データのロード」33 ページを参照してください。

- **ESRI シェイプファイルからのロード**　ESRI シェイプファイルのデータを新規または既存のテーブルにロードできます。これを実行する方法は数多くあります。「ESRI シェイプファイルのサポート」18 ページと「レッスン 3 : ESRI シェイプファイルデータのロード」61 ページを参照してください。

- **SELECT...FROM OPENSTRING 文の使用**　空間データを含むファイルで SELECT... FROM OPENSTRING 文を実行できます。次に例を示します。


```
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 )
```

FROM 句の Openstring 式『SQL Anywhere サーバー SQL リファレンス』を参照してください。

- **緯度値と経度値を組み合わせた座標ポイントの作成**　緯度と経度のデータを組み合わせて、空間データ型 ST_Point の座標を作成できます。たとえば、緯度と経度のカラムをすでに持つテーブルがある場合、次のような文を使用して、ポイントとして値を保持する ST_Point カラムを作成できます。

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

- **コンストラクターと静的メソッドを使用したジオメトリの作成**　コンストラクターと静的メソッドを使用してジオメトリを作成できます。「UDT インスタンスのインスタンス化」25 ページと「静的メソッドの使用」26 ページを参照してください。

空間データのイメージとしての表示 (Interactive SQL の場合)

Interactive SQL では、**[空間プレビュー]** タブを使用してジオメトリをイメージとして表示し、データベースのデータが表すものを理解できます。

前提条件

空間データを含むテーブルから選択したパーミッション。

内容と備考

Interactive SQL の各インスタンスは、データベースへのそれぞれの接続に関連付けられています。Interactive SQL 内から **[空間ビューアー]** のインスタンスを開くと、その **[空間ビューアー]** のインスタンスは Interactive SQL のインスタンスと関連付けられたままであり、データベースへの接続を共有します。

これは、**[空間ビューアー]** でクエリを実行しているときに、関連付けられている Interactive SQL のインスタンスでクエリを実行しようとすると、エラーが発生することを意味します。同様に、Interactive SQL の同じインスタンスによって作成された複数の **[空間ビューアー]** のインスタンスを開いている場合、クエリを実行できるのは、それらのインスタンスのいずれか 1 つのみです。その他のインスタンスは、そのクエリが完了するのを待機する必要があります。

注意
デフォルトでは、Interactive SQL は **[結果]** ウィンドウ枠内の値を 256 文字にトランケートします。Interactive SQL が完全なカラム値を読み込めないことを示すエラーを返す場合、トランケーション値を増やします。これを行うには、**[ツール]** » **[オプション]** をクリックし、左ウィンドウ枠で **[SQL Anywhere]** をクリックします。**[結果]** タブで、**[トランケーションの長さ]** を 5000 などの大きい値に変更します。**[OK]** をクリックして変更を保存し、文を再び実行します。

♦ 空間データのイメージとしての表示 (Interactive SQL の場合)

1. Interactive SQL のデータベースに接続します。
2. クエリを実行して、テーブルから空間データを選択します。次に例を示します。

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

3. **[結果]** ウィンドウ枠の Shapes カラムの任意の値をダブルクリックして、値を **[値]** ウィンドウに表示します。

   値は、**[値]** ウィンドウの **[テキスト]** タブにテキストとして表示されます。

4. **[空間プレビュー]** タブをクリックし、ジオメトリを SVG (Scalable Vector Graphic) として表示します。

結果

ジオメトリは、Scalable Vector Graphic (SVG) として表示されます。

次の手順

[前のロー] と **[次のロー]** ボタンを使用して結果セットの他のローを表示することで、空間データをジオメトリとして表示できます。

例

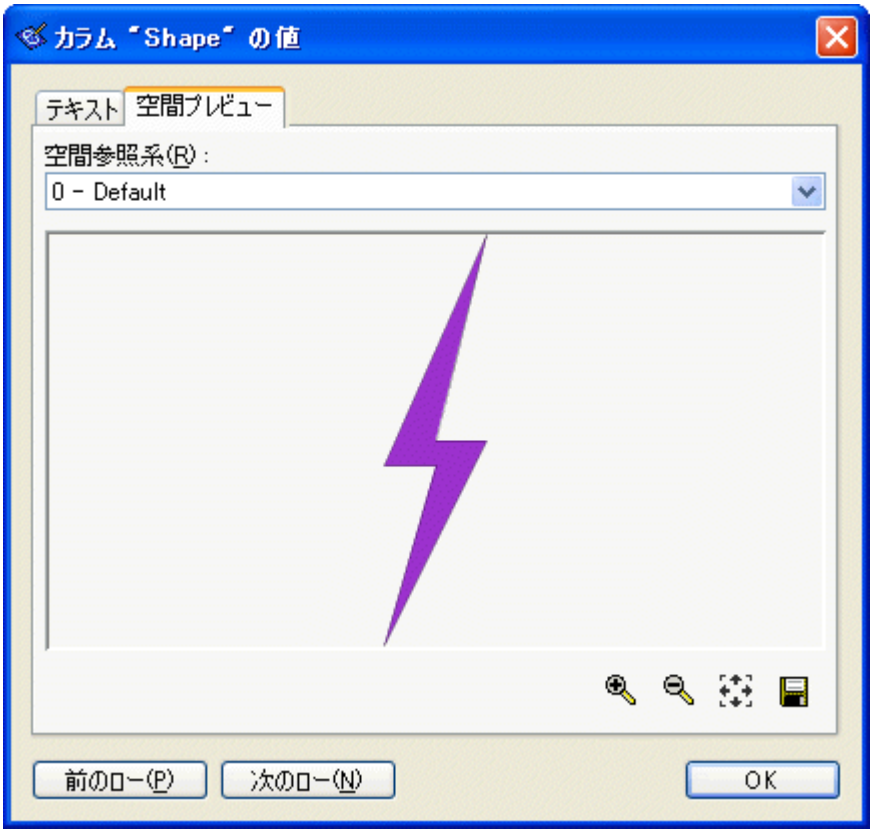
1. サンプルデータベースに接続し、次のクエリを実行します。

   ```
   SELECT * FROM SpatialShapes;
   ```

2. **[結果]** ウィンドウ枠の Shapes カラムの任意の値をダブルクリックして、値を **[値]** ウィンドウに表示します。

   値は、**[値]** ウィンドウの **[テキスト]** タブにテキストとして表示されます。

3. **[空間プレビュー]** タブをクリックし、ジオメトリを SVG (Scalable Vector Graphic) として表示します。



空間データのイメージとしての表示 (Spatial Viewer の場合)

空間ビューアーを使用して複数のジオメトリをイメージとして表示し、データベースのデータが表すものを理解できます。

前提条件

空間データを含むテーブルから選択したパーミッション。

内容と備考

画像はローが処理される順序で描画され、最新の画像が最上位に表示されるため、結果のローの順序は画像が**空間ビューアー**に表示される方法に影響を与えます。結果セットでは、後で出現するシェイプが先に出現したシェイプを覆い隠します。

♦ 空間データのイメージとしての表示 (Spatial Viewer の場合)

1. Interactive SQL のデータベースに接続し、**[ツール]** » **[空間ビューアー]** をクリックします。
2. **[空間ビューアー]** の **[SQL]** ウィンドウ枠で、次のようなクエリを実行し、**[実行]** をクリックします。

```
SELECT * FROM SpatialShapes;
```

3. **[塗りつぶしなしでポリゴンの描画]** ツールを使用すると、図形のポリゴンから色彩を削除して、すべてのシェイプのアウトラインを表示します。このツールは、保存、ズーム、パンのコントロールの近くにあるイメージの下にあります。

結果

結果セットのすべてのジオメトリは、1 つのイメージとして **[結果]** 領域に表示されます。

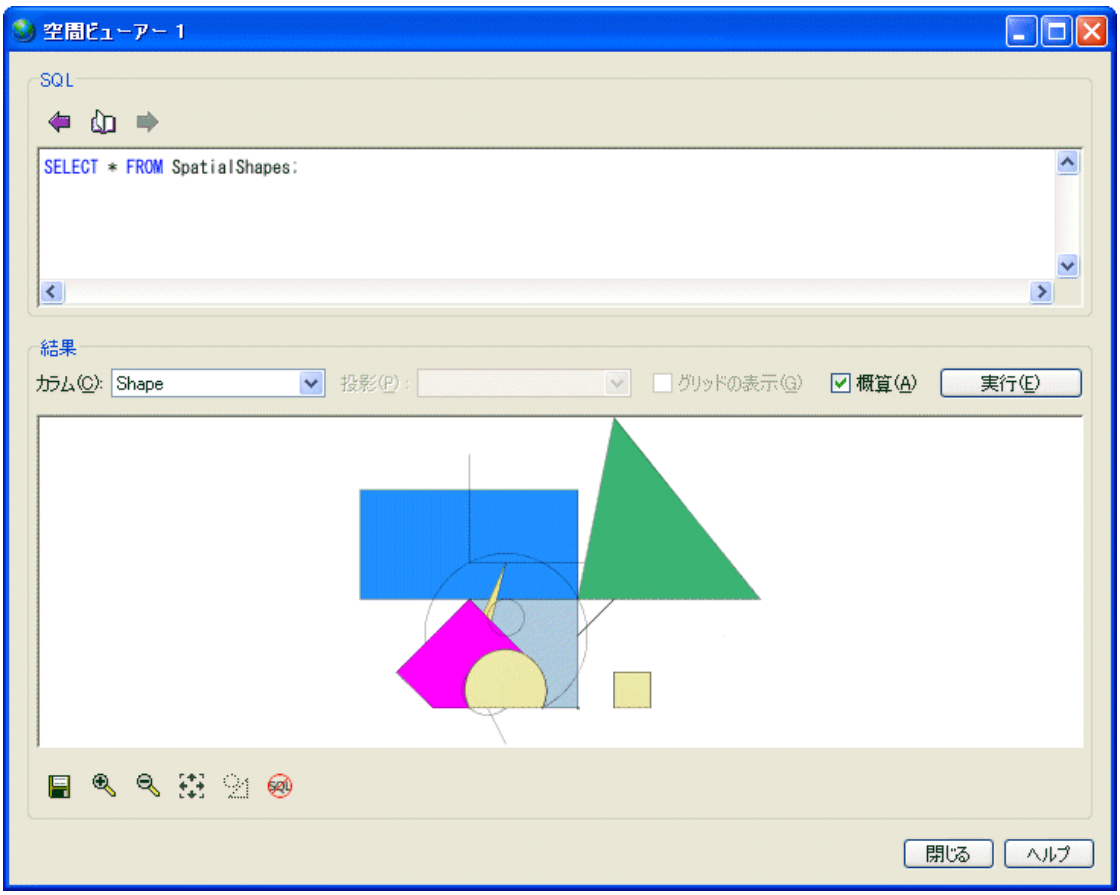
次の手順

None

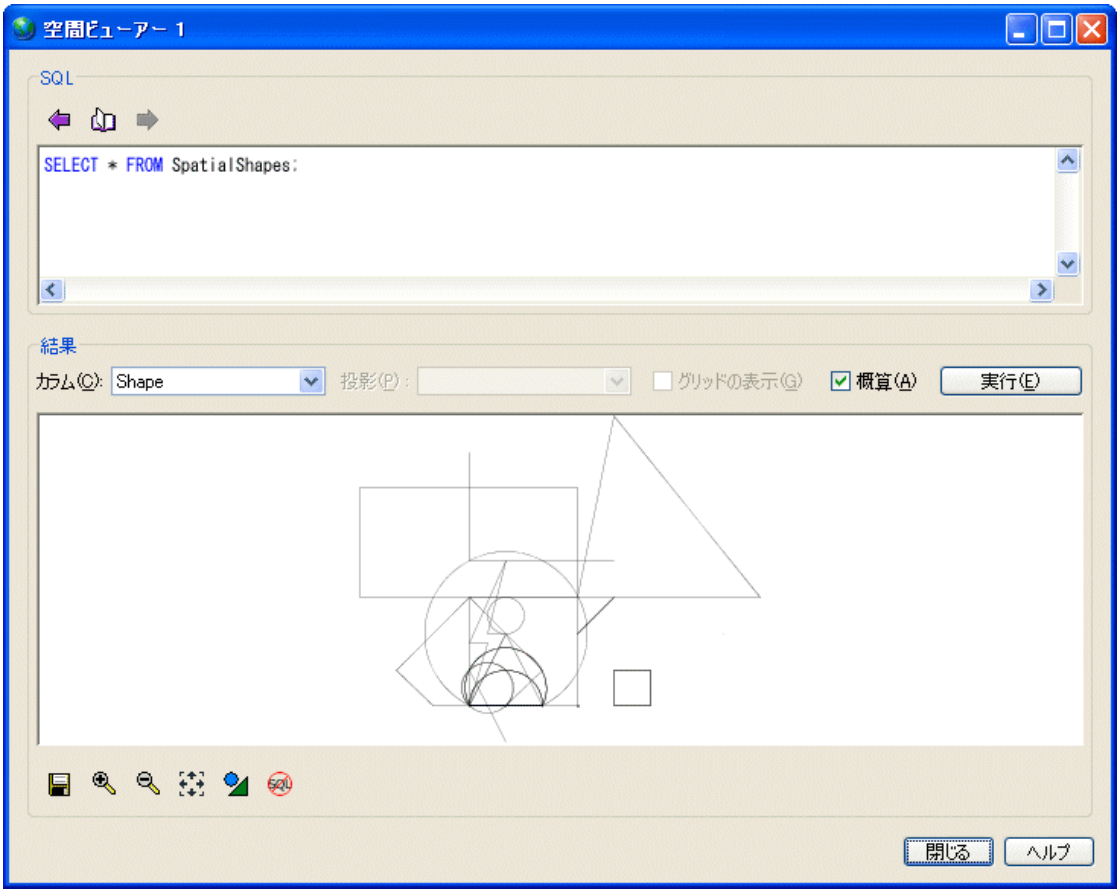
例

1. Interactive SQL からサンプルデータベースに接続します。
2. **[ツール]** » **[空間ビューアー]** をクリックします。
3. **[空間ビューアー]** の **[SQL]** ウィンドウ枠で、次のクエリを実行します。

```
SELECT * FROM SpatialShapes;
```



4. 次に、[塗りつぶしなしでポリゴンの描画] ツールを使用してイメージをアウトライン表示する例を示します。



WKT (Well Known Text) ファイルからの空間データのロード

空間データをデータベースにロードし、ジオメトリとして表示できるテキストを含んだ Well Known Text (WKT) ファイルを使用して、空間データをテーブルに追加できます。

前提条件

DBA 権限または SYS_SPATIAL_ADMIN_ROLE グループのメンバー

内容と備考

次のとおりです。

♦ WKT ファイルから空間データをロードする

1. データベースにロードできる空間データを WKT フォーマットに含めたファイルを作成します。

   ファイルは、LOAD TABLE 文によってサポートされるフォーマットです。

2. Interactive SQL で、DBA 権限を所有するユーザーまたは SYS_SPATIAL_ADMIN_ROLE グループのメンバーとして、データベースに接続します。

3. 次のような文を使用して、テーブルを作成し、ファイルからデータをロードします。

```
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 12¥¥Samples¥¥wktgeometries.csv' DELIMITED BY ',';
```

   データがテーブルにロードされます。

結果

空間データは、WKT ファイルから正常にロードされます。

次の手順

[空間ビューアー] を使用すると、Interactive SQL のデータを表示できます。

例

1. 次のテキストを、*wktgeometries.csv* という名前のテキストファイルに保存します。

   次のテキストには、WKT で定義されたジオメトリのグループが含まれます。

```
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. Interactive SQL で、DBA ユーザーとして、または SYS_SPATIAL_ADMIN_ROLE グループのメンバーとして、サンプルデータベース (*demo.db*) に接続します。

3. SA_WKT という名前のテーブルを作成し、*wktgeometries.csv* からデータをロードします。*csv* ファイルへのパスを、ファイルを保存した場所のパスに置き換えてください。

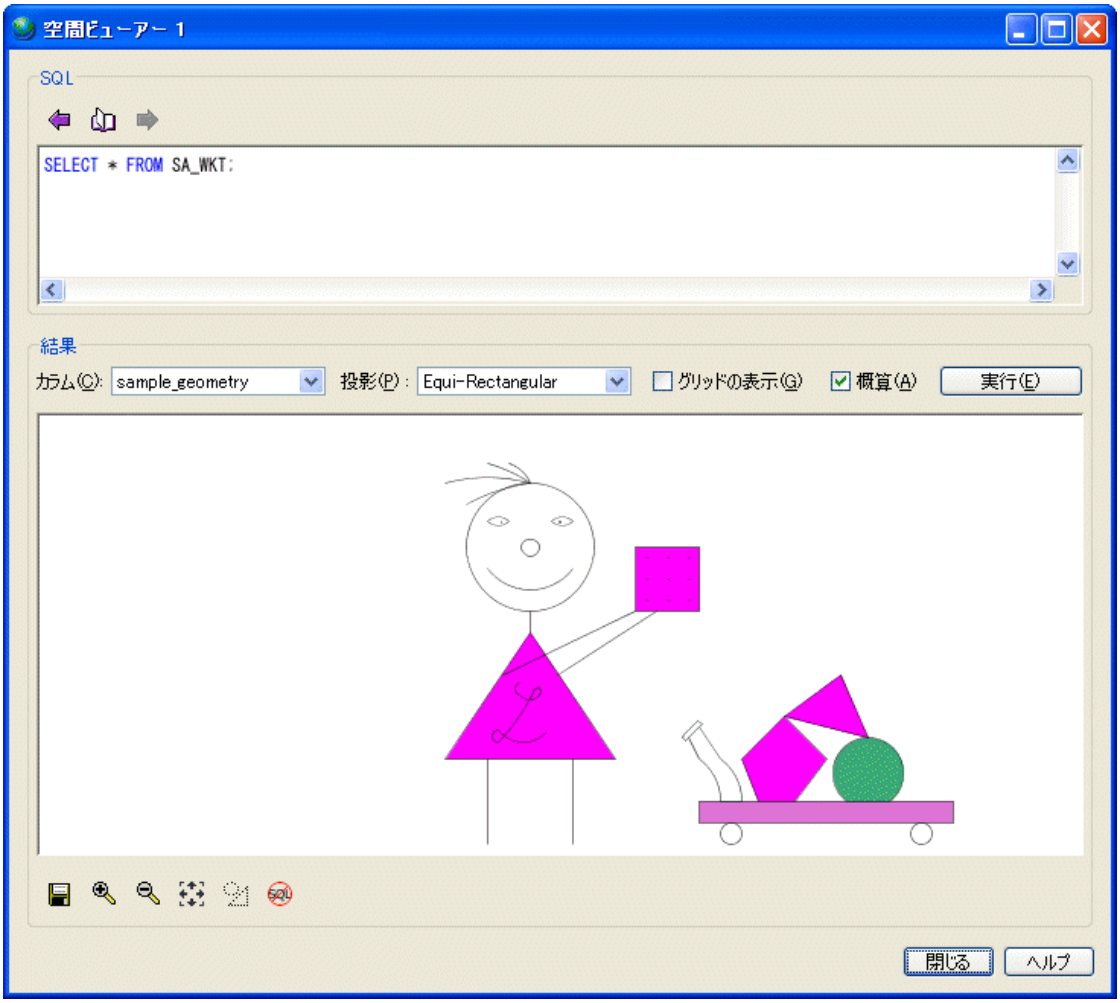
   ```
   DROP TABLE IF EXISTS SA_WKT;
   CREATE TABLE SA_WKT (
      description CHAR(24),
      sample_geometry ST_Geometry(SRID=1000004326)
   );

   LOAD TABLE SA_WKT FROM 'C:¥¥Documents and Settings¥¥All Users¥¥Documents¥¥SQL
   Anywhere 12¥¥Samples¥¥wktgeometries.csv' DELIMITED BY ',';
   ```

   データがテーブルにロードされます。

4. Interactive SQL で、**[ツール]** » **[空間ビューアー]** をクリックします。

5. **[空間ビューアー]** で、次の文を実行してジオメトリを表示します。

   ```
   SELECT * FROM SA_WKT;
   ```



6. データに複数の空間データのカラムが含まれている場合があります。サポートされる各空間データ型がそれぞれ個別のカラムに格納された WKT データのファイルを作成します。

   次のコードをテキストエディターにコピーして、ファイルを *wktgeometries2.csv* として保存します。

```
"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)))"
```

7. 次の文を実行することで、SA_WKT2 という名前のテーブルを作成し、*wktgeometries2.csv* からデータをロードします。*.csv* ファイルへのパスを、ファイルを保存した場所のパスに置き換えてください。

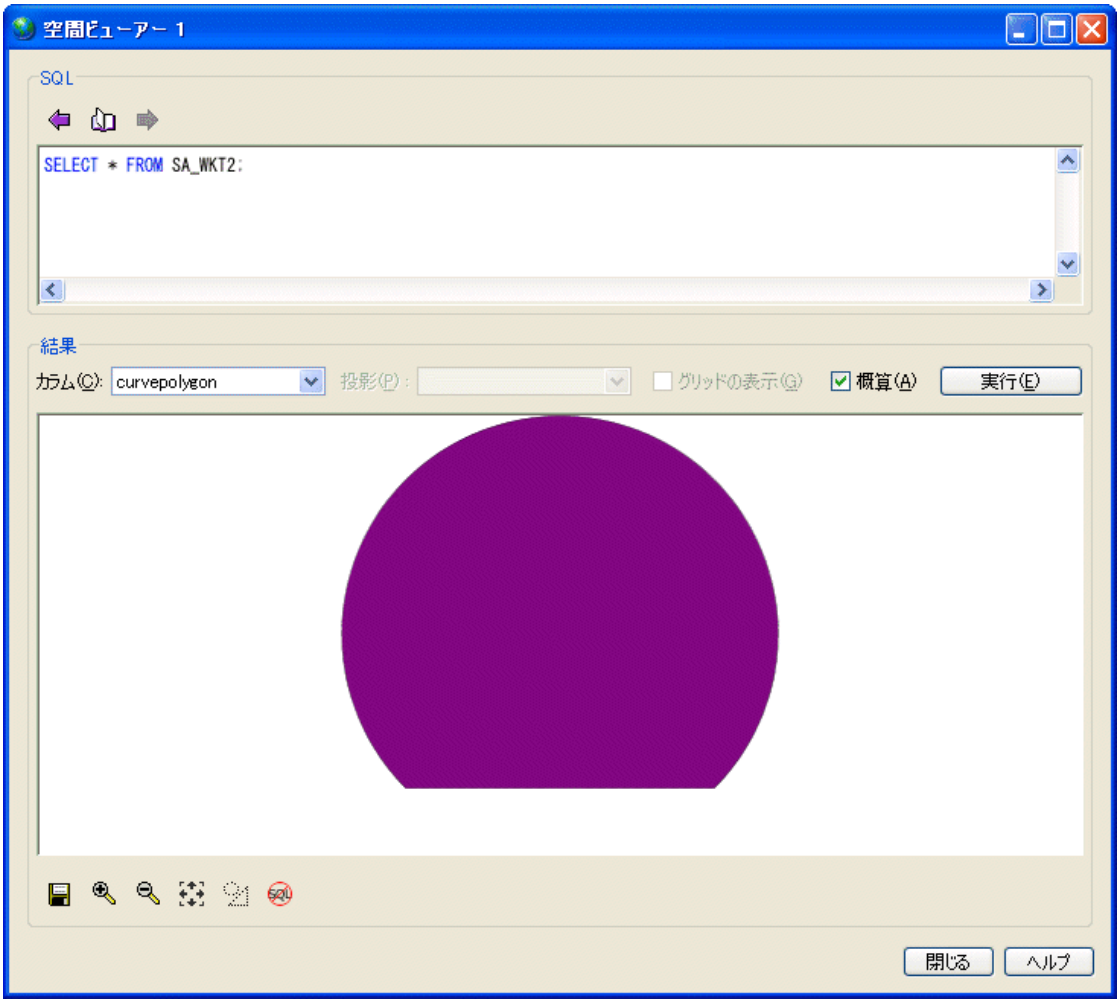
```
DROP TABLE IF EXISTS SA_WKT2;
CREATE TABLE SA_WKT2 (
   point         ST_Point,
   line          ST_LineString,
   circle        ST_CircularString,
   compoundcurve  ST_CompoundCurve,
   curve         ST_Curve,
   polygon1      ST_Polygon,
   curvepolygon   ST_CurvePolygon,
   surface       ST_Surface,
   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 12¥¥Samples¥¥wktgeometries2.csv' DELIMITED BY ',';
```

データがテーブルにロードされます。

8. **[空間ビューアー]** で、次の文を実行してジオメトリを表示します。

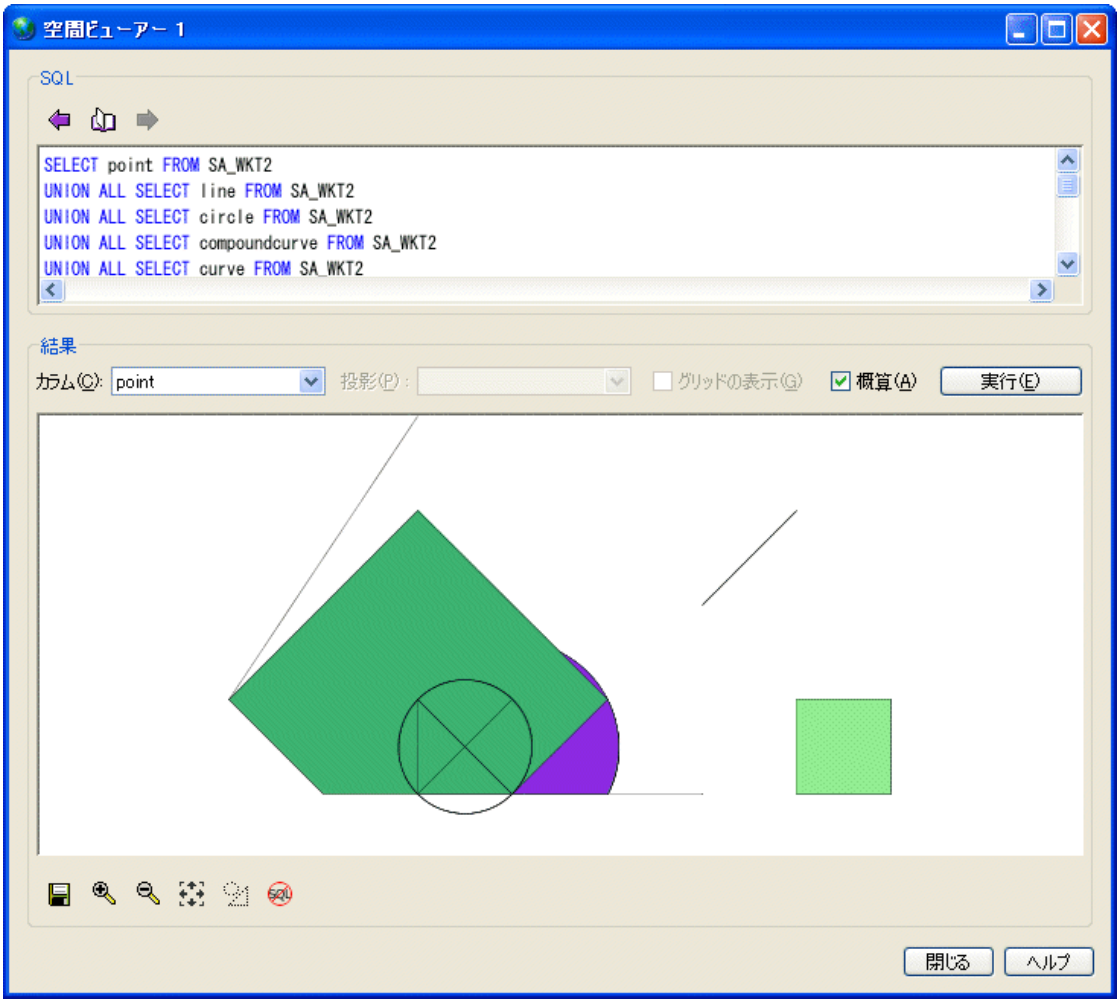
```
SELECT * FROM SA_WKT2;
```

一度に表示できるデータのカラムは 1 つのみであり、その他のカラムのジオメトリを表示するには、**[結果]** 領域の **[カラム]** ドロップダウンリストを使用する必要があります。たとえば、次の図は curvepolygon カラムのジオメトリを表示しています。



9. すべてのカラムのジオメトリを一度に表示するには、次のように各カラムへの SELECT 文を実行して、結果を UNION ALL で結合します。

```
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
```

参照

- 「空間データのイメージとしての表示 (Spatial Viewer の場合)」 31 ページ
- 「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』

測定単位の作成 (Sybase Central の場合)

複数の測定単位がインストールされています。インストールされた測定単位がデータに適切でない場合、独自の測定単位を作成できます。

前提条件

DBA 権限または SYS_SPATIAL_ADMIN_ROLE グループのメンバー

内容と備考

次のとおりです。

♦ 測定単位の作成 (Sybase Central の場合)

1. DBA 権限を所有するユーザーまたは SYS_SPATIAL_ADMIN_ROLE グループのメンバーとして、データベースに接続します。
2. 左ウィンドウ枠で、**[空間参照系]** をクリックします。
3. 右ウィンドウ枠で、**[測定単位]** タブをクリックします。
4. タブを右クリックし、**[新規]** » **[測定単位]** をクリックします。
5. **[カスタム測定単位を作成する]** をクリックし、**[次へ]** をクリックします。
6. **[新しい測定単位の名前を指定してください。]** フィールドに名前を指定し、**[次へ]** をクリックします。
7. **[作成する測定単位のタイプを指定してください。]** フィールドで、**[線形]** を選択します。
8. **測定単位の作成ウィザード**の指示に従います。
9. **[完了]** をクリックします。

結果

測定単位が作成されます。

次の手順

データベース内のユーザーにパーミッションを付与して、空間参照系と計測単位を作成、変更、または削除できます。

参照

- 「CREATE SPATIAL UNIT OF MEASURE 文」『SQL Anywhere サーバー SQL リファレンス』
- 「測定単位」 5 ページ

空間参照系の作成 (Sybase Central の場合)

[空間参照系の作成ウィザード] を使用して設定を基礎とするテンプレートとして、既存の空間参照系 (SRS) を使用する空間参照系を作成できます。

前提条件

DBA 権限または SYS_SPATIAL_ADMIN_ROLE グループのメンバー

SRS に関連付ける測定単位は、あらかじめ用意しておく必要があります。

内容と備考

空間参照系を作成する場合、既存の空間参照系をテンプレートとして使用し、設定のベースにします。このため、作成したい空間参照系と類似した空間参照系を選択してください。設定は後で編集できます。

♦ 空間参照系の作成 (Sybase Central の場合)

1. Sybase Central で、DBA 権限を持つユーザーまたは SYS_SPATIAL_ADMIN_ROLE グループのメンバーとしてデータベースに接続します。
2. 左ウィンドウ枠で、**[空間参照系]** » を右クリックし、**[新規]** » **[空間参照系]** を選択します。
3. **[事前に定義されたすべての空間参照系のリストから選択する]** を選択して、**[次へ]** をクリックします。
4. ウィザードの指示に従います。
5. 既存の空間参照系に基づいて空間参照系を作成する場合、Well Known 値に 1000000000 を加えた値を SRID 値に設定します。

   注意
   SRID を割り当てるときは、回避する番号範囲についての推奨事項を確認してください。これらの推奨は、CREATE SPATIAL REFERENCE SYSTEM 文の IDENTIFIED 句の項で説明されています。

6. 空間参照系の定義が表示されます。
7. この空間参照系の定義で問題がない場合は、**[完了]** をクリックします。

結果

新しい空間参照系がデータベースに追加されます。

次の手順

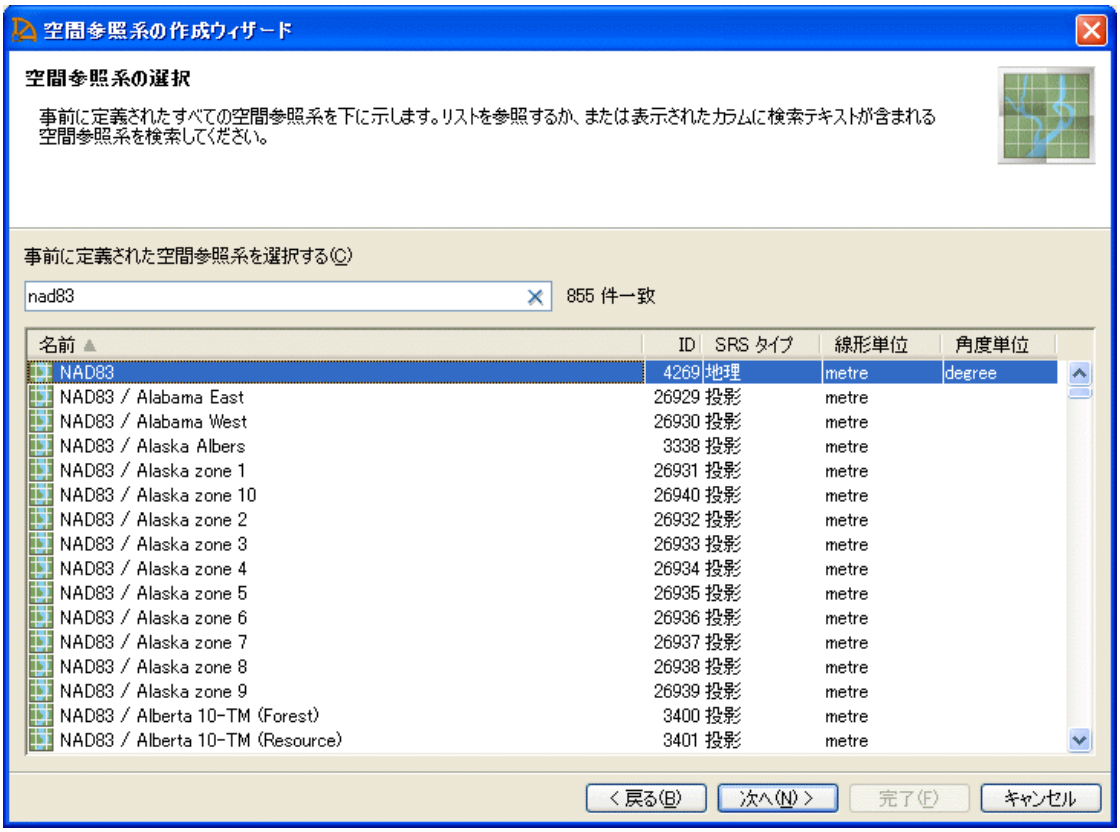
空間参照系を使用するテーブルを定義します。

例

次に、既存の空間参照系に基づいた空間参照系を作成する例を示します。

1. DBA 権限を所有するユーザーまたは SYS_SPATIAL_ADMIN_ROLE グループのメンバーとして、データベースに接続します。
2. 左ウィンドウ枠で、**[空間参照系]** » を右クリックし、**[新規]** » **[空間参照系]** を選択します。
3. **[事前に定義されたすべての空間参照系のリストから選択する]** を選択して、**[次へ]** をクリックします。

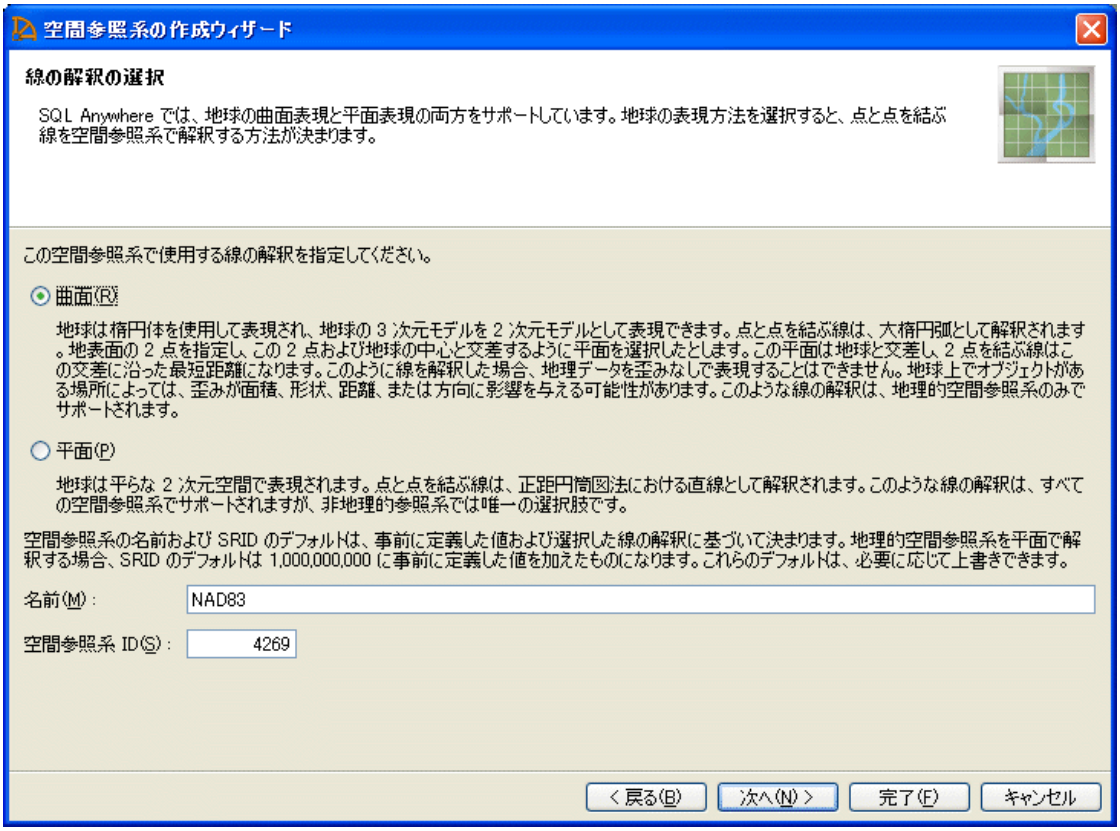
[空間参照系の選択] ウィンドウが表示されます。



4. NAD83 空間参照系に基づいた空間参照系を作成するには、**NAD83** と入力します。名前と ID を **[事前に定義された空間参照系を選択する]** フィールドに入力すると、空間参照系のリストが、テンプレートとして使用する空間参照系が表示されるように移動します。

5. **NAD83** をクリックし、**[次へ]** をクリックします。

   [線の解釈の選択] ウィンドウが表示されます。



6. 線の解釈として **[曲面]** をクリックします。
7. **[名前]** フィールドに **NAD83custom** と指定します。
8. 既存の空間参照系に基づいて空間参照系を作成する場合、Well Known 値に 1000000000 を加えた値を SRID 値に設定します。たとえば、**[空間参照系 ID]** フィールドの値を 4269 から **1000004269** に変更します。

   注意
   SRID を割り当てるときは、回避する番号範囲についての推奨事項を確認してください。これらの推奨は、CREATE SPATIAL REFERENCE SYSTEM 文の IDENTIFIED 句の項で説明されています。

9. **[次へ]** をクリックします。

   [コメントの指定] ウィンドウが表示されます。

10. 必要に応じて、空間参照系の説明を記述し、**[次へ]** をクリックします。
11. **[完了]** をクリックします。

    空間参照系の定義が表示されます。

12. **[完了]** をクリックします。

参照

- 「測定単位」 5 ページ
- 「CREATE SPATIAL UNIT OF MEASURE 文」『SQL Anywhere サーバー SQL リファレンス』
- IDENTIFIED BY 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』
- 「CREATE SPATIAL REFERENCE SYSTEM 文」『SQL Anywhere サーバー SQL リファレンス』
- 「空間参照系 (SRS) と空間参照系識別子 (SRID)」 2 ページ

空間パーミッションの付与

空間参照系および測定単位を作成、変更、または削除するには、DBA パーミッションを持つユーザーであるか、SYS_SPATIAL_ADMIN_ROLE グループに属している必要があります。

参照

- 「グループメンバーシップを既存のユーザーまたはグループに付与する」『SQL Anywhere サーバー データベース管理』

空間に関する高度なトピック

この項では、空間に関する高度なトピックについて説明します。

平面および曲面の表現方法

SQL Anywhere では、平面と曲面の両方の表現がサポートされます。「平面」参照系では、地表全体またはその一部が平坦な 2 次元平面に投射され、単純な 2D ユークリッド幾何学が使用されます。ポイント間の線は直線 (円ストリングを除く) であり、ジオメトリは平面の端をまたぐ (日付変更線を超える) ことはできません。

「曲面」空間参照系は、楕円を使用して地球を表します。ポイントは計算のために楕円にマッピングされ、すべての線は最短の経路をたどり、円弧は極に向かっています。ジオメトリは日付変更線をまたぐことができます。

平面と曲面の表現にはそれぞれ制限があります。地球のすべての特性を最適に表す完璧な地図投影法は 1 つもありません。オブジェクトの地球上での位置によって、ゆがみが面積、シェイプ、距離、または方向に影響することがあります。

曲面空間参照系の制限

曲面空間参照系 (WGS 84 など) を使用する場合、利用できない操作が多数あります。たとえば、距離の計算は、ポイントまたはポイントのコレクションに制限されます。

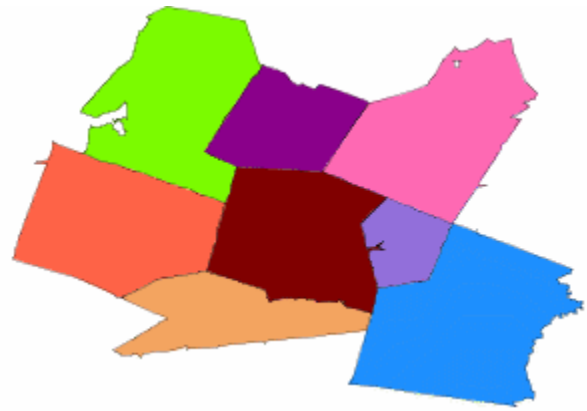
いくつかの述部と集合操作も利用できません。

円ストリングは、曲面の空間参照系では使用できません。

曲面空間参照系での計算は、対応する平面空間参照系での計算より高コストになります。

平面空間参照系の制限

平面空間参照系は、定義された投影を持つ平面の空間参照系です。「投影」を使用すると、平面空間参照系を使用して曲面データを操作する場合に発生するゆがみの問題が解決されます。投影が使用されない場合に発生するゆがみの例として、次の 2 つのイメージはマサチューセッツ州の同じ郵便番号区域のグループを示しています。最初のイメージは、データを SRID 3586 で表現した投影平面空間参照系であり、マサチューセッツ州のデータを示しています。2 番目のイメージは、投影なしに平面空間参照系 (SRID 1000004326) でデータを表現しています。ゆがみは 2 番目のイメージに現れています。距離、長さ、および面積が実際より大きく、イメージが水平方向に引き伸ばされたように見えます。



他にも平面空間参照系で可能な計算はありますが、投影が影響するため、正確に計算できるのは境界に区切られたサイズの面積のみです。

作業対象が数百キロメートル以内の距離であれば、曲面データを平面空間参照系に投影して、適度な精度で距離の計算を実行できます。平面投影空間参照系にデータを投影するには、ST_Transform メソッドを使用します。

参照

- ST_Geometry タイプの ST_Transform メソッド247 ページ

「グリッドにスナップ」と許容度が空間の計算に与える影響

「グリッドにスナップ」は、グリッド上の交差ポイントに合うように、ジオメトリのポイントを位置付けするアクションです。「グリッド」に位置付けるときは、四捨五入と同じように、X と Y の値がわずかに移動される場合があります。空間データのコンテキストでは、グリッドは、空間参照系の 2 次元表現上に定義された線のフレームワークです。SQL Anywhere では正方形のグリッドを使用します。

グリッドにスナップの最も簡単な例として、たとえば、グリッドサイズが 0.2 の場合、Point(14.2321, 28.3262) と Point(15.3721, 27.1128) を結ぶ直線は、Point(14.2, 28.4) と Point(15.4, 27.2) を結ぶ直線にスナップされます。通常、グリッドサイズはこの単純な例よりも小さいため、精度が失われる可能性はずっと低くなります。

デフォルトでは、SQL Anywhere は空間参照系の X 境界と Y 境界内のすべてのポイントに対して 12 有効桁数を格納できるようにグリッドサイズを自動的に設定します。たとえば、X 値の範囲が -180 から 180 まで、Y 値の範囲が -90 から 90 までの場合、データベースサーバーはグリッドサイズを 1e-9 (0.000000001) に設定します。つまり、水平と垂直のグリッド線間の距離が 1e-9 となります。グリッド線の交差ポイントは、空間参照系で表現できるすべてのポイントを表します。ジオメトリが作成またはロードされると、各ポイントの X 座標と Y 座標が、グリッド上の最も近いポイントにスナップされます。

「許容度」は、この範囲内であるとジオメトリの 2 つのポイントまたは部分が同一であると見なされる距離を定義します。許容度は、ペン先の太いマーカーを使用して描画したポイントと線で表現されているすべてのジオメトリにおける、ペン先の太さであると考えることができます。この太いマーカーを使用して描画した場合に接触するすべての部分が許容度内にあると見なされます。2 つのポイントが許容度からまったく同じ距離分離れている場合、これらのポイントは許容度内とは見なされません。

許容度の単純な例として、たとえば、許容度が 0.5 の場合には、Point(14.2, 28.4) と Point(14.4, 28.2) は等しいと見なされます。これは、(X と Y が同じ単位で表される) 2 点間の距離が約 0.283 であり、許容度よりも小さいためです。通常、許容度にはこの単純な例よりずっと小さな値が設定されます。

極端に小さいジオメトリは、許容度により無効になる場合があることに注意してください。長さが許容度より短い線は無効です (ポイントが等しいため)。また、すべてのポイントが許容度内にある同様の多角形は無効と見なされます。

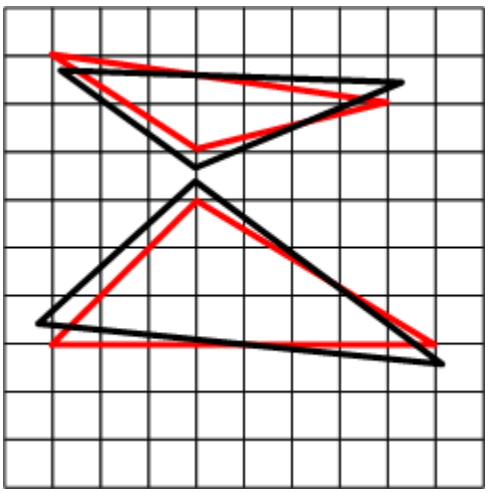
グリッドにスナップと許容度は空間参照系に設定され、X と Y (または経度の緯度) の座標には常に同じ単位が使用されます。「グリッドにスナップ」と許容度は、厳密でない算術データや不正確なデータに関する問題を解消するために一緒に使用されます。ただし、これらの動作が空間操作の結果に及ぼす影響について注意してください。

注意
平面空間参照系の場合、空間操作が不正な結果になる場合があるため、グリッドサイズを 0 に設定することはおすすめしません。曲面空間参照系の場合、グリッドサイズと許容度を 0 に設定してください。SQL Anywhere は、曲面操作を実行する場合、固定されたグリッドサイズと許容度を内部投影で使用します。

次の例は、グリッドサイズと許容度の設定が空間の計算に与える影響を示しています。

例 1 :「グリッドにスナップ」が交差の結果に与える影響

2 つの三角形 (黒で表示) が空間参照系にロードされます。ここで、許容度はグリッドサイズと同じに設定され、図のグリッドはグリッドサイズに基づいています。黒の三角形の頂点をグリッドにスナップした後の三角形を、赤い三角形で表しています。元の三角形 (黒) はそれぞれの許容度の範囲内に適切に収まっていますが、スナップされたバージョンの赤の三角形は許容度内にないことに注意してください。ST_Intersects は、これらの 2 つのジオメトリに対して 0 を返します。許容度がグリッドサイズより大きい場合、ST_Intersects はこれらの 2 つのジオメトリに対して 1 を返します。

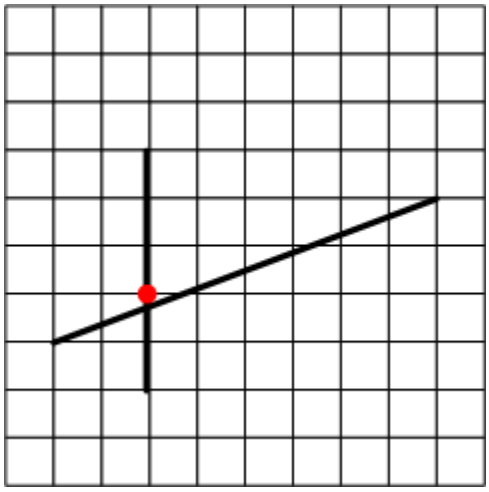


例 2 : 許容度が交差の結果に与える影響

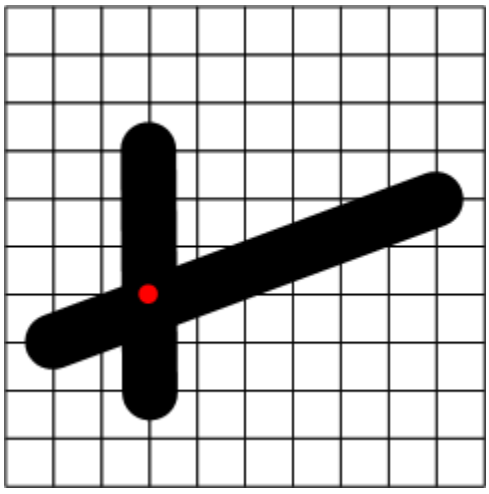
次の例では、許容度が 0 に設定された空間参照系内に 2 つの線があります。2 つの線の交差ポイントは、グリッドの最も近い頂点にスナップされています。許容度が 0 に設定されているため、2 つの線の交差ポイントが斜めの線と交差しているかどうかを検査するテストでは false が返されます。

つまり、許容度が 0 の場合、次の式は 0 を返します。

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

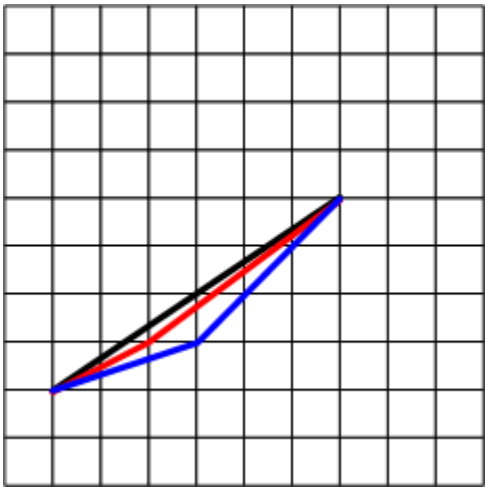


許容度をグリッドサイズと同じに設定すると (デフォルト)、交差ポイントが太い斜めの線の内側に収まります。したがって、交差ポイントが斜めの線と許容度内で交差するかどうかのテストはパスします。



例 3：許容度と推移性

許容度が使用されている場合の空間の計算では、推移性が保持される必要はありません。たとえば、許容度がグリッドサイズと等しい空間参照系に、次の 3 つの線があるとします。



ST_Equals メソッドでは、黒と赤の線および赤と青の線はそれぞれ許容度内にあると見なされますが、黒と青の線は許容度内にあるとは見なされません。ST_Equals は推移的ではありません。

ST_OrderingEquals は、これらの各線を異なると見なし、推移的であることに注意してください。

例 4：グリッドと許容度の設定が不正確なデータに与える影響

投影平面空間参照系にあるデータの精度が、10 cm 以内でほぼ正確、10 m 以内では必ず正確であるとします。この場合 3 つの選択肢があります。

1. SQL Anywhere によって選択されるデフォルトのグリッドサイズと許容度を使用します。これは、通常は、使用するデータの精度よりも高い精度になります。これにより最大精度が提供されますが、ST_Intersects、ST_Touches、ST_Equals などの述部は、ジオメトリ値の精度に応じて、一部のジオメトリに対して予期したものと異なる結果を返すことがあります。たとえば、1 つの境界を共有する隣接した 2 つの多角形において、一番右側の多角形の左端から数メートルのところに一番左側の多角形の境界データがある場合、ST_Intersect は true を返さない可能性があります。

2. グリッドサイズを、最も精度の高いデータを表すことができるくらい小さくし (この場合、10 cm)、許容度の 4 分の 1 以下に設定します。許容度は、データが常にどの程度の精度で距離を表すかを示す値に設定します (この場合、10 m)。この方式では、精度を失うことなくデータが格納され、データが 10 m 以内でしか正確でない場合でも述部が予期した結果が返されます。

3. グリッドサイズと許容度をデータの精度と同じに設定します (この場合、10 m)。この方法では、データはその精度内にスナップされますが、10 m より精度が高いデータの場合は精度が失われます。

   多くの場合、述部は予期した結果を返しますが、そうでない場合もあります。たとえば、2 つのポイントが 10 cm 以内にあるが、グリッドの交差の中間点近くにある場合、2 つのポイントはそれぞれ別の方向にスナップされて、ポイント間の距離が 10 m になってしまいます。このため、グリッドサイズと許容度をデータの精度と同じに設定することは、この場合おすすめしません。

参照

- SNAP TO GRID 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』
- TOLERANCE 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』
- ST_Geometry タイプの ST_Equals メソッド192 ページ
- ST_Geometry タイプの ST_SnapToGrid メソッド227 ページ
- ST_Geometry タイプの ST_OrderingEquals メソッド217 ページ
- 「サポートされる空間述部」 10 ページ
- 「CREATE SPATIAL REFERENCE SYSTEM 文」『SQL Anywhere サーバー SQL リファレンス』
- 「ALTER SPATIAL REFERENCE SYSTEM 文」『SQL Anywhere サーバー SQL リファレンス』

補間が空間の計算に与える影響

「補間」とは、ジオメトリ中の既知のポイントを使用して、未知のポイントを概算するプロセスです。いくつかの空間メソッドおよび述部では、円弧が関係する計算を実行するときに、補間が使用されます。補間によって、円弧は一連の直線に置き換えられます。たとえば、4 分円を表す円ストリングを補間して、コントロールポイントが 11 ある線ストリングに変換する場合などがあります。

補間の例

1. Interactive SQL で、サンプルのデータベースに接続し、次の文を実行して円ストリングを格納する arc という変数を作成します。

   ```
   CREATE VARIABLE arc ST_CircularString;
   ```

2. 次の文を実行して、円ストリングを作成し、それを arc 変数に格納します。

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

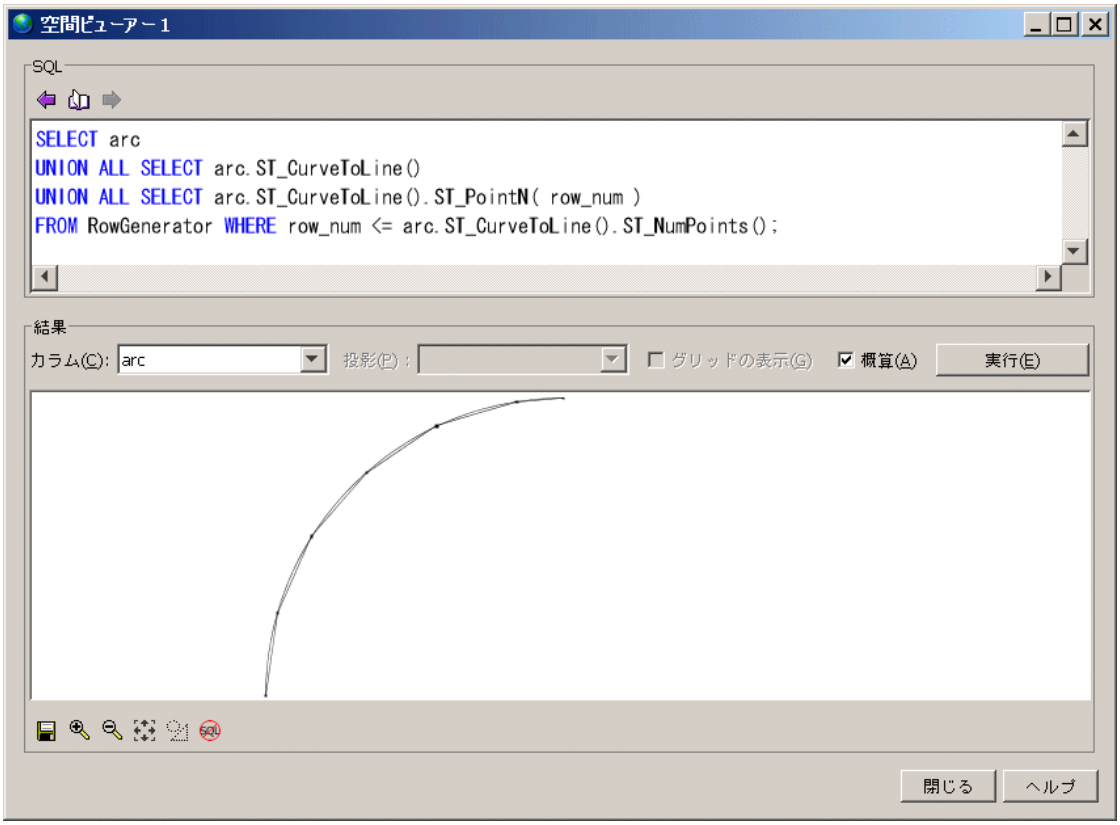
3. 次の文を実行して、st_geometry_interpolation オプションを使用し、一時的に相対許容度を 1 % に設定します。

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

   相対許容度を 1 % に設定するのはオプションですが、この例では、補間による影響をよりわかりやすくするために使用します。相対許容度とその設定方法の詳細については、「st_geometry_interpolation オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

4. (Interactive SQL で **[ツール]** [空間ビューアー] を選択して) **[空間ビューアー]** を開き、次のクエリを実行して円ストリングを表示します。 »

   ```
   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();
   ```

円弧が一連の線ストリングに分解されていることに注意してください。相対許容度が 1% に設定されているため、各直線セグメントは実際の円弧の内側にずれて表示されます。補間された線ストリングと実際の円弧との間の最大距離は、円弧の半径の 1 % になります。

参照

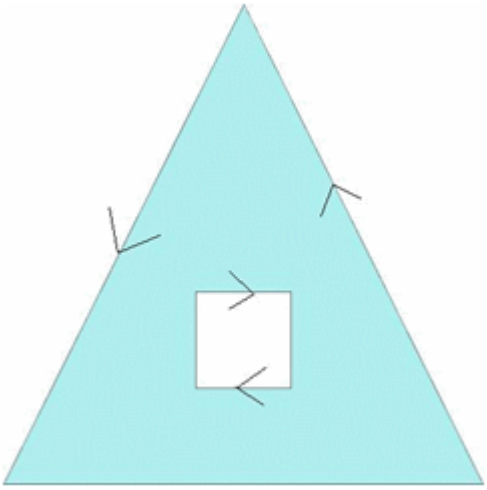
- 「st_geometry_interpolation オプション」『SQL Anywhere サーバー データベース管理』

多角形リングの方向操作

SQL Anywhere では、まず、多角形を構成するリングの方向によって多角形が解釈されます。定義された点の順序でリングを移動した場合、多角形の内側がリングの左側になります。平面と曲面の空間参照系では、同じルールが適用されます。ほとんどの場合、外部リングは反時計回りの方向で、内部リングは逆の方向 (時計回り) となります。例外は、ROUND EARTH に北極または南極が含まれるリングの場合です。

デフォルトでは、多角形が SQL Anywhere の内部的なリング方向と異なるリング方向で作成されている場合は、自動的に再方向付けされます。CREATE SPATIAL REFERENCE SYSTEM 文の POLYGON FORMAT 句を使用して、入力データの多角形リングの方向を指定します。これは、空間参照系の入力データで同じリングの方向を使用している場合にのみ実行する必要があります。POLYGON FORMAT は、一部の多角形および複数面のコンストラクターでも指定できます。

たとえば、多角形を作成し、ポイントを時計回りの順で指定すると (Polygon((0 0, 「5 10, 10 0,」 0 0), (4 2, 4 4, 6 4, 6 2, 4 2)))、データベースサーバーは、ポイントが反時計回りになるように自動的に並べ替えます ()。 Polygon((0 0, 「10 0, 5 10,」 0 0), (4 2, 4 4, 6 4, 6 2, 4 2)).



内部リングが外部リングより前に指定された場合、外部リングが最初のリングとして表示されます。

曲面空間参照系で多角形の再方向付けが機能するためには、多角形の直径は 160 度に制限されます。

参照

- POLYGON FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』.

ジオメトリの内部、外部、境界の操作

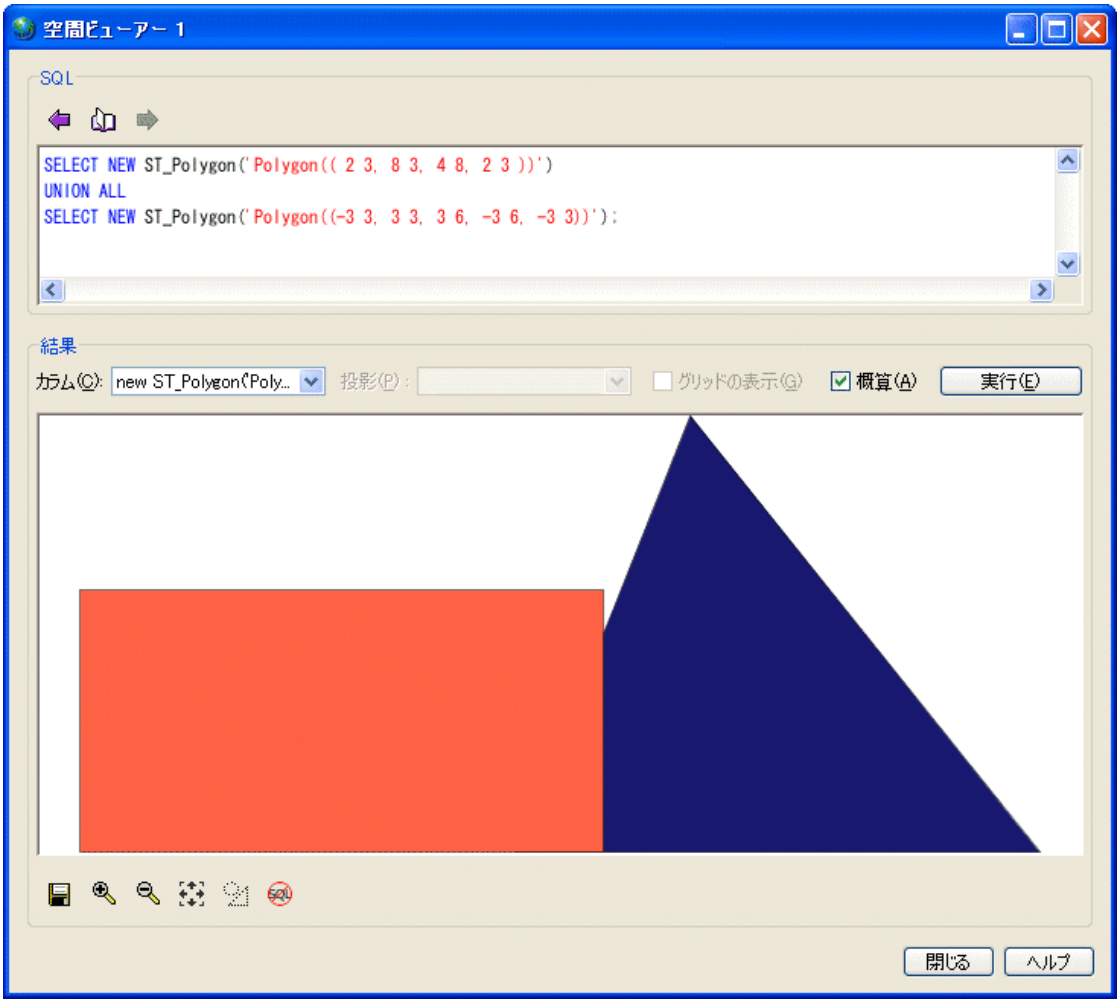
ジオメトリの「内部」とは、ジオメトリの一部であるすべてのポイントです (境界を除く)。

ジオメトリの「外部」とは、ジオメトリの部分ではないすべてのポイントです。これには、内部リングの内側の空間も含まれます。たとえば、多角形に穴の開いている場合です。同様に、線ストリングのリングの内側と外側の空間は外部と見なされます。

ジオメトリの「境界」とは、ST_Boundary メソッドによって返される内容です。

ジオメトリの境界について知っていると、別のジオメトリと比較して、2 つのジオメトリの関連を判別するときに役立ちます。ただし、すべてのジオメトリには内部と外部がありますが、すべてのジオメトリに境界があるわけではありません。また、その境界は必ずしも直感的ではありません。

境界が直感的ではないジオメトリのケースを次に示します。



- **ポイント**　ポイント (たとえば A) には境界はありません。
- **線と曲線**　線と曲線 (B、C、D、E、F) の境界は終了ポイントです。ジオメトリ B、C、E には、境界として 2 つの終了ポイントがあります。ジオメトリ D には、境界として 4 つの終了ポイントがあり、ジオメトリ F にも 4 つあります。
- **多角形**　多角形 (たとえば G) の境界は、その外部リングと内部リングです。
- **リング**　リングとは、開始ポイントが終了ポイントと同じで、交差することがない曲線 (たとえば H) であり、境界はありません。

参照

- ST_Geometry タイプの ST_Boundary メソッド171 ページ

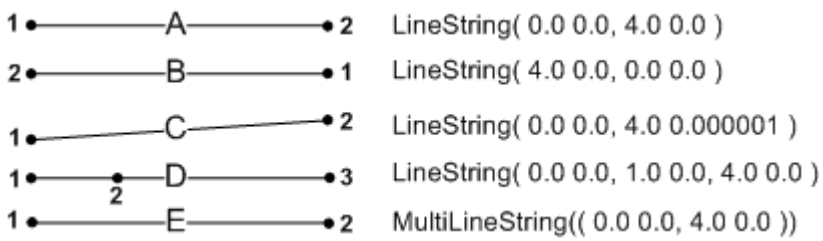
空間の比較操作

ジオメトリが別のジオメトリと等しいかどうかをテストするために使用できるメソッドには、ST_Equals と ST_OrderingEquals の 2 つがあります。これらのメソッドで実行される比較と返される結果は異なっています。

- **ST_Equals**　ポイントが指定される順序は関係ありません。ポイントの比較では許容度が考慮されます。許容度内で同じ空間を占有している場合は、ジオメトリについても等しいと見なされます。たとえば、2 つの線ストリングが同じ空間を占有しており、片方にはより多くのポイントが定義されている場合でも、2 つの線ストリングが等しいと見なされることを意味します。

- **ST_OrderingEquals**　ST_OrderingEquals では、2 つのジオメトリには同じオブジェクト階層が含まれ、その階層には ST_OrderingEquals で等しいと見なされる順序でまったく同じポイントが存在している必要があります。つまり、2 つのジオメトリがまったく同一である必要があります。

ST_Equals と ST_OrderingEquals を使用して比較を実行したときの結果の差異を確認するために、次の線を比較してみてください。ST_Equals では、これらのすべての線が等しいと見なされます (線 C が許容度内であることが前提)。ただし、ST_OrderingEquals では、これらのすべての線は等しいとは見なされません。



SQL Anywhere によるジオメトリ比較の実行方法

データベースサーバーは、ST_OrderingEquals を使用して、GROUP BY、DISTINCT などの操作を実行します。

たとえば、次のクエリを処理するとき、2 つの shape 式で ST_OrderingEquals() = 1 である場合、サーバーは 2 つのローを等しいと見なします。

```
SELECT DISTINCT Shape FROM SpatialShapes;
```

SQL 文では、等しい (=) または等しくない (<> または !=) の演算子を使用して 2 つのジオメトリを比較できます。サブクエリや ANY または ALL キーワードを含む検索条件も使用できます。ジオメトリは IN 検索条件でも使用できます。たとえば geom1 IN (geom-expr1, geom-expr2, geom-expr3) のように指定します。これらすべての検索条件では、等価性は ST_OrderingEquals セマンティックを使用して評価されます。

その他の比較演算子を使用して、ジオメトリが別のジオメトリより小さいかどうか、または大きいかどうかを判別することはできません (たとえば、geom1 < geom2 は受け入れられません)。

これは、ジオメトリ式を ORDER BY 句に含めることはできないことを意味します。ただし、集合に含まれているかどうかはテストできます。

空間の関係操作

最良のパフォーマンスを得るためには、ST_Within または ST_Touches などのメソッドを常に使用して、ジオメトリ間の単一の特定の関係をテストします。ただし、複数の関係をテストする場合、一度に複数の関係をテストできる ST_Relate メソッドの方が適しています。ST_Relate は、述部の異なる解釈をテストする場合にも役に立ちます。

ST_Relate の最も一般的な使用法は、テストする関係を厳密に指定して、述部として使用する方法です。ただし、ST_Relate を使用して、2 つのジオメトリ間で可能なすべての関係を判別することもできます。

述部としての ST_Relate の使用

ST_Relate は、内部、境界、外部の「交差テスト」を実行することによって、ジオメトリ間の関係を評価します。ジオメトリ間の関係は、DE-9IM (Dimensionally Extended 9 Intersection Model) フォーマットの 9 文字の文字列で記述されます。この文字列の各文字は、交差テストの結果の次元を表します。

ST_Relate を述部として使用する場合、テストする交差の結果を示した DE-9IM 文字列を渡します。指定した DE-9IM 文字列の条件をジオメトリが満たしている場合、ST_Relate は 「1」 を返します。条件が満たされない場合は、「0」 を返します。片方のジオメトリ、または両方が NULL の場合、「NULL」 を返します。

9 文字の DE-9IM 文字列は、内部、境界、外部間の交差テストのペアごとのマトリックスをフラットにした表現です。次の表に、実行される順序 (左から右、上から下) で 9 つの交差テストを示します。

	「g2 内部」	「g2 境界」	「g2 外部」
「g1 内部」	Interior(g1) ∩ Interior(g2)	Interior(g1) ∩ Boundary(g2)	Interior(g1) ∩ Exterior(g2)
「g1 境界」	Boundary(g1) ∩ Interior(g2)	Boundary(g1) ∩ Boundary(g2)	Boundary(g1) ∩ Exterior(g2)
「g1 外部」	Exterior(g1) ∩ Interior(g2)	Exterior(g1) ∩ Boundary(g2)	Exterior(g1) ∩ Exterior(g2)

DE-9IM 文字列を指定する場合、9 文字の各文字に *、0、1、2、T、または F を指定できます。これらの値は、交差によって作成されるジオメトリの次元数を表しています。

指定する文字	交差テストが返す結果
T	次のいずれか：0、1、2 (任意の次元の交差)

指定する文字	交差テストが返す結果
F	-1
*	-1、0、1、2 (任意の値)
0	0
1	1
2	2

ST_Relate と Within 述部用のカスタム DE-9IM 文字列を使用して、ジオメトリが別のジオメトリ内にあるかどうかをテストするとします。

```
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***' );
```

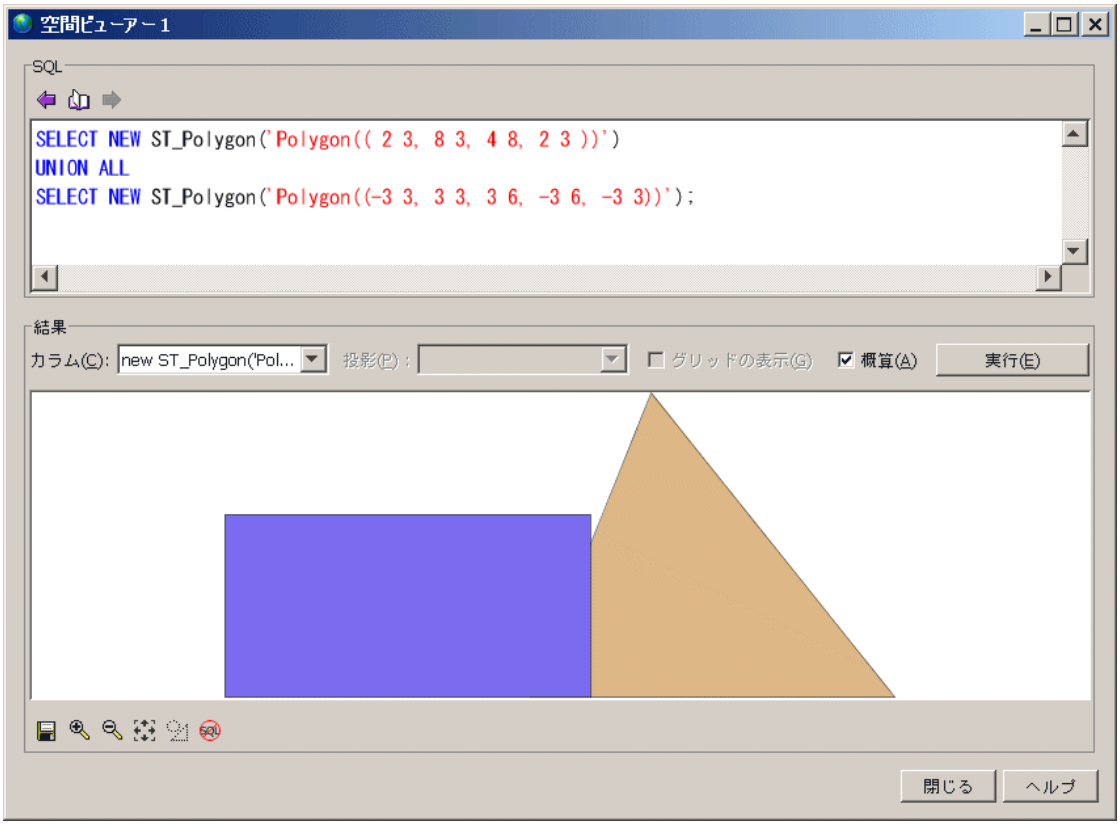
これは、交差テストを実行するときに、ST_Relate に次の条件で検索するように問い合わせるのと同じです。

	g2 内部	g2 境界	g2 外部
g1 内部	次のいずれか：0、1、2 0, 1, 2	次のいずれか：0、1、2、-1 0, 1, 2, -1	「-1」
g1 境界	次のいずれか：0、1、2、-1 0, 1, 2, -1	次のいずれか：0、1、2、-1 0, 1, 2, -1	「-1」
g1 外部	次のいずれか：0、1、2、-1 0, 1, 2, -1	次のいずれか：0、1、2、-1 0, 1, 2, -1	次のいずれか：0、1、2、-1 0, 1, 2, -1

クエリを実行すると、ST_Relate は最初のジオメトリが 2 番目のジオメトリ内にないことを示す 0 を返します。

2 つのジオメトリを表示して、その形状をテストしている内容と比較するには、次の文を Interactive SQL の空間ビューアー (**[ツール] » [空間ビューアー]**) で実行します。

```
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))');
```

参照

- 「ST_Geometry タイプの ST_Relate(ST_Geometry,CHAR(9)) メソッド」221 ページ

述部以外での ST_Relate の使用

述部以外で ST_Relate を使用すると、2 つのジオメトリ間のすべての関係が返されます。

たとえば、前の例で使用したものと同じ 2 つのジオメトリがあり、それらの関係を知りたいとします。次の文を Interactive SQL で実行すると、ジオメトリ間の関係を定義した DE-9IM 文字列が返されます。

```
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 は DE-9IM 文字列 212111212 を返します。

この値をマトリックス表示すると、多くの交差ポイントがあることがわかります。

	g2 内部	g2 境界	g2 外部
g1 内部	2	1	2

g1 境界	1	1	1
g1 外部	2	1	2

参照

- ST_Geometry タイプの ST_Intersects メソッド203 ページ
- ST_Geometry タイプの ST_Overlaps メソッド219 ページ
- ST_Geometry タイプの ST_Within メソッド251 ページ
- ST_Geometry タイプの ST_Disjoint メソッド188 ページ
- ST_Geometry タイプの ST_Touches メソッド246 ページ
- ST_Geometry タイプの ST_Crosses メソッド184 ページ
- ST_Geometry タイプの ST_Contains メソッド172 ページ
- ST_Geometry タイプの ST_Relate メソッド220 ページ
- 「ST_Geometry タイプの ST_Relate(ST_Geometry) メソッド」223 ページ

空間の次元の操作

各ジオメトリサブクラスは、独自のプロパティを持っている以外に、ST_Geometry スーパータイプからプロパティを継承しています。ジオメトリサブタイプは、次のいずれかの次元値を持っています。

- **-1** 値 -1 は、ジオメトリが空であることを示します (ポイントが 1 つもない)。
- **0** 値 0 は、ジオメトリが長さまたは面積を持たないことを示します。サブタイプ ST_Point と ST_MultiPoint は次元値 0 を持ちます。1 つのポイントは、座標の単一のペアによって表すことができるジオメトリ特性を表します。接続されていないポイントのクラスターは MultiPoint 特性を表します。
- **1** 値 1 は、ジオメトリに長さがあるが、面積がないことを示します。次元 1 を持つ一連のサブタイプは、ST_Curve のサブタイプ (ST_LineString、ST_CircularString、および ST_CompoundCurve)、またはこれらのタイプを含んでいるが面を持たないコレクションタイプです。GIS データでは、これらの次元 1 のジオメトリは、線形特性 (河川、水系、道路網など) を定義するために使用されます。
- **2** 値 2 は、ジオメトリが面積を持っていることを示します。次元 2 を持つ一連のサブタイプは、ST_Surface のサブタイプ (ST_Polygon と ST_CurvePolygon)、またはこれらのタイプを含んでいるコレクションタイプです。多角形と複数多角形は、定義された面積を囲む外周を持つジオメトリ特性 (湖、公園など) を表します。

ジオメトリの次元は、ジオメトリの各ポイントの座標次元の数とは関係がありません。

1 つの ST_GeomCollection には、次元の異なるジオメトリを含めることができ、最も高い次元のジオメトリが返されます。

参照

- ST_Geometry タイプの ST_CoordDim メソッド178 ページ
- ST_Geometry タイプの ST_Dimension メソッド187 ページ

チュートリアル：空間機能の実験

このチュートリアルでは、SQL Anywhere のいくつかの空間機能を実験します。これを行うには、まず ESRI シェイプファイルをサンプルデータベース (demo.db) にロードして、実験のための有効な空間データを用意します。

このチュートリアルは次に示す部分に分かれています。

レッスン 1：追加の測定単位と空間参照系のインストール

このレッスンでは、sa_install_feature システムプロシージャーを使用し、このチュートリアルの後半で必要となる多くの定義済みの測定単位と空間参照系をインストールする方法について説明します。

♦ 定義済みの測定単位と空間参照系のインストール

1. Interactive SQL で、サンプルデータベース (demo.db) を起動し、ユーザー DBA として、または SYS_SPATIAL_ADMIN_ROLE グループのメンバーとして接続します。

   サンプルデータベースは *%SQLANYSAMP12%* にあります。

2. Interactive SQL で次の文を実行します。

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

   文が完了すると、追加の測定単位と空間参照系がインストールされています。

   「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』と「CREATE SPATIAL REFERENCE SYSTEM 文」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

3. データベースにインストールされている測定単位を判別するには、次のクエリを実行します。

   ```
   SELECT * FROM ST_UNITS_OF_MEASURE;
   ```

「ST_UNITS_OF_MEASURE 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

4. データベースにインストールされている空間参照系を判別するには、Sybase Central で **[空間参照系]** フォルダーを表示するか、Interactive SQL で次のクエリを実行します。

   ```
   SELECT * FROM ST_SPATIAL_REFERENCE_SYSTEMS;
   ```

   「ST_SPATIAL_REFERENCE_SYSTEMS 統合ビュー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

5. 「レッスン 2：ESRI シェイプファイルデータのダウンロード」60 ページに進みます。

レッスン 2：ESRI シェイプファイルデータのダウンロード

このレッスンでは、米国国勢調査局の Web サイト (www2.census.gov) から ESRI シェイプファイルをダウンロードします。ダウンロードするシェイプファイルは、2002 年の国勢調査の集計で使用されたマサチューセッツ州の 5 桁の郵便番号情報を表しています。各郵便番号区域は、多角形または複数多角形として扱われています。

♦ サンプル空間データのダウンロード

1. *c:¥temp¥massdata* という名前のローカルディレクトリを作成します。
2. 次の URL に移動します：http://www2.census.gov/cgi-bin/shapefiles2009/national-files http://www2.census.gov/cgi-bin/shapefiles2009/national-files
3. ページの右側の **[State- and County-based Shapefiles]** ドロップダウンから **[Massachusetts]** をクリックして、**[submit]** をクリックします。
4. ページの左側で、**[5-Digit ZIP Code Tabulation Area (2002)]** をクリックして、**[Download Selected Files]** をクリックします。
5. プロンプトが表示されたら、zip ファイル *multiple_tiger_files.zip* を *c:¥temp¥massdata* に保存し、ファイルの内容を解凍します。これにより、25_MASSACHUSETTS という名前のサブフォルダーが作成され、*tl_2009_25_zcta5.zip* という名前の別の zip ファイルがそこに作成されます。
6. *tl_2009_25_zcta5.zip* の内容を *C:¥temp¥massdata* に解凍します。

   これにより、空間データをデータベースにロードするために使用する ESRI シェイプファイル (*.shp*) を含む 5 つのファイルが解凍されます。
7. 「レッスン 3：ESRI シェイプファイルデータのロード」61 ページに進みます。

レッスン 3：ESRI シェイプファイルデータのロード

このレッスンでは、ESRI シェイプファイル内のカラムを特定し、その情報を使用して、データをロードするテーブルを作成する方法を説明します。

♦ ESRI シェイプファイルからデータベースへの空間データのロード

1. 空間データは特定の空間参照系に関連付けられているため、データベースにデータをロードする場合、同じ空間参照系、または少なくとも等価な定義を持つ空間参照系にロードする必要があります。ESRI シェイプファイルの空間参照系情報を確認するには、プロジェクトファイル *c:¥temp¥massdata¥tl_2009_25_zcta5.prj* をテキストエディターで開きます。このファイルには、この手順に必要な空間参照系情報が含まれています。

   ```
   GEOGCS["「GCS_North_American_1983」", DATUM["D_North_American_1983",
   SPHEROID["GRS_1980",6378137,298.257222101]],
   PRIMEM["Greenwich",0],UNIT["Degree",0.0174532925199433295]]
   ```

   文字列「GCS_North_American_1983」 は、データに関連付けられている空間参照系の名前です。

2. ST_SPATIAL_REFERENCE_SYSTEMS ビューに問い合わせてみると (SELECT * FROM ST_SPATIAL_REFERENCE_SYSTEMS WHERE srs_name='GCS_North_American_1983';)、この名前は定義済みの SRS のリストにはありません。ただし、同じ定義の空間参照系を問い合わせて、代わりにそれを使用できます。

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

   このクエリは、 1 つの同じ定義を持つ SRID 「4269」 の空間参照系 NAD83 を返します。これがシェイプファイルからデータをロードするために割り当てる SRID です。

3. Interactive SQL で、次の文を実行して、Massdata というテーブルを作成し、シェイプファイルをテーブルにロードして、データに SRID 4269 を割り当てます。ロードには数分かかることがあります。

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

   を参照してください。

   > **注意**
   > **[インポートウィザード]**でも、シェイプファイルからのデータのロードがサポートされています。「インポートウィザード (Interactive SQL) でのデータのインポート」『SQL Anywhere サーバー SQL の使用法』を参照してください。

4. Interactive SQL で、テーブルをクエリし、シェイプファイルにあったデータを表示します。

   ```
   SELECT * FROM Massdata;
   ```

結果の各ローは、郵便番号区域のデータを表します。

geometry カラムには、多角形 (1 つの領域) または複数多角形 (複数の隣接しない領域) として、郵便番号区域のシェイプ情報が保持されています。

5. ZCTA5CE カラムには郵便番号が格納されています。このカラムをチュートリアルの後半で簡単に参照できるようにするため、Intaractive SQL で ALTER TABLE 文を実行して、カラム名を **ZIP** に変更します。:

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

6. 2 つのカラム INTPTLON と INTPTLAT は、郵便番号区域の中心ポイントの X 座標と Y 座標を表しています。Interactive SQL で次の ALTER TABLE 文を実行して、ST_Point タイプの CentePoint というカラムを作成し、それぞれの X と Y を CenterPoint の値にします。

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

これで、Massdata.CenterPoint の ST_Point 値が、Massdata.geometry に格納された郵便番号区域の中心ポイントを表すようになりました。

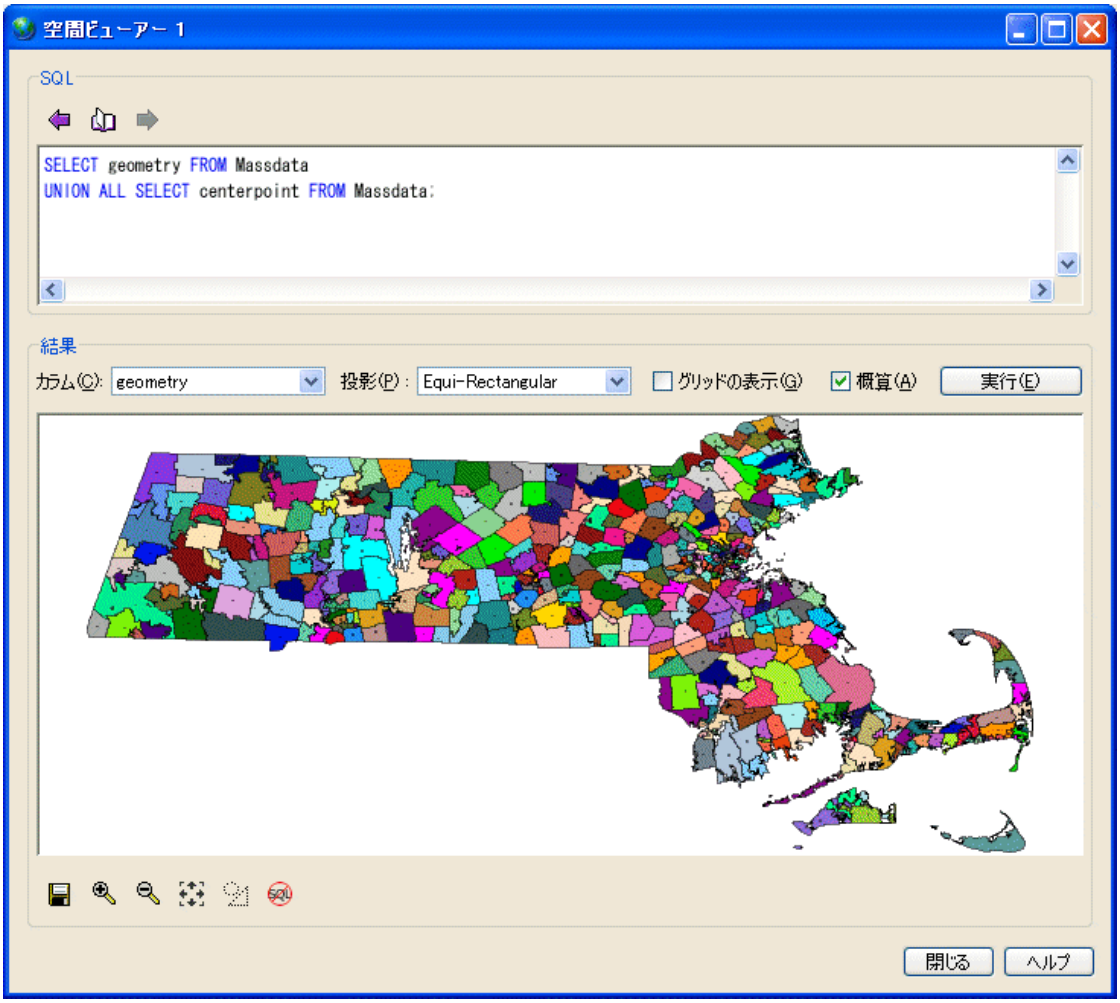
7. 個別のジオメトリ (郵便番号区域) をシェイプとして表示するには、Massdata.geometry の最初の値以外をダブルクリックし、**[カラムの値]** ウィンドウの **[空間プレビュー]** タブをクリックします。

値が大きすぎるというエラー、または結果にプライマリキーを含めることを推奨するというエラーを受け取った場合、Interactive SQL で表示するために値がトランケートされたことが原因です。これを解決するには、クエリを変更して、結果にプライマリキーカラムが含まれるようにするか、Interactive SQL の **[トランケーションの長さ]** 設定を調整します。Interactive SQL でジオメトリを表示するために、ジオメトリを問い合わせるたびにプライマリキーを含める必要をなくしたい場合は、設定の変更をおすすめします。

Interactive SQL の **[トランケーションの長さ]** 設定を変更するには、**[ツール]** » **[オプション]** » **[SQL Anywhere]** をクリックし、**[トランケーションの長さ]** を 100000 などの大きい値に設定します。

8. データセット全体を 1 つのシェイプとして表示するには、**[ツール]** » **[空間ビューアー]** をクリックして、SQL Anywhere の **[空間ビューアー]** を開き、Interactive SQL で次のクエリを実行します。

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

9. 「レッスン 4：空間データのクエリ」63 ページに進みます。

レッスン 4：空間データのクエリ

このレッスンでは、いくつかの空間メソッドを使用して、意味のあるコンテキストでデータを問い合わせる方法について説明します。

SpatialContacts テーブルと Massdata テーブルのどちらか、または両方でクエリを実行します。データベースに既に存在している SpatialContacts には、多くは Massachusetts に在住する人々の名前と連絡先情報が格納されています。

また、距離を計算する方法も学習するためデータベースに測定単位を追加する必要があります。

♦ 空間データのクエリ

1. Interactive SQL で、郵便番号区域 01775 に関連付けられるジオメトリを保持するために、@Mass_01775 という変数を作成します。

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

2. 郵便番号区域 01775 と周辺の郵便番号区域のすべての連絡先を SpatialContacts から検索するとします。このためには、交差するジオメトリ、または指定したジオメトリと同じジオメトリを返す、ST_Intersects メソッドを使用します。Interactive SQL で次の文を実行します。

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

   ST_Geometry タイプの ST_Intersects メソッド203 ページを参照してください。

3. Massdata.geometry のすべてのローは、同じ空間参照系 (SRID 4269) に関連付けられています。これは、geometry カラムを作成するときに SRID 4269 を割り当て、そのカラムにデータをロードしたためです。

   ただし、「宣言されていない」 ST_Geometry カラム (つまり、SRID が割り当てられていない) を作成することもできます。これは、異なる SRS が関連付けられている空間値を単一のカラムに格納したい場合に必要になります。これらの値に対して操作が実行されると、各値に関連付けられている空間参照系が使用されます。

   宣言されていないカラムを使用する場合のリスクの 1 つは、データベースサーバーでは、宣言されていないカラムのデータに関連付けられている空間参照系の変更が防止されないことです。

   カラムに宣言された SRID がある場合、データベースサーバーはデータに関連付けられている空間参照系を変更することを許可しません。最初に、宣言されたカラムのデータをアンロードしてトランケートしてから、空間参照系を変更し、データを再ロードする必要があります。

   ST_SRID メソッドを使用すると、宣言されているかどうかに関係なく、カラムの値に関連付けられている SRID を判別できます。たとえば、次の文は Massdata.geometry カラムの各ローに割り当てられている SRID を表示します。

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

   を参照してください。

4. ST_CoveredBy メソッドを使用すると、ジオメトリが別のジオメトリ内に完全に含まれているかどうかをチェックできます。たとえば、Massdata.CenterPoint (ST_Point タイプ) には郵便番号区域の中心の緯度／経度の座標が含まれ、Massdata.geometry には郵便番号区域を反映した多角形が含まれます。Interactive SQL で次のクエリを実行することによって、郵便番号区域外に設定されている CenterPoint 値がないことを簡単に確認できます。


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

ローが 1 つも返されないため、すべての CenterPoint 値が Massdata.geometry の関連付けられているジオメトリ内にあることを示しています。このチェックは CenterPoint 値が本当に中心にあるかどうかは検証していません。そのためには、データを平面空間参照系に投影し、ST_Centroid メソッドを使用して CenterPoint 値をチェックする必要があります。データを別の空間参照系に投影する方法については、「レッスン 6：空間データの投影」67 ページを参照してください。

ST_Geometry タイプの ST_CoveredBy メソッド179 ページを参照してください。

5. ST_Distance メソッドを使用すると、郵便番号区域の中心点間の距離を測定できます。たとえば、郵便番号区域 01775 から 100 マイル以内の郵便番号をリストしたいとします。この処理は、Interactive SQL で次のクエリを実行します。

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

ST_Geometry タイプの ST_Distance メソッド189 ページを参照してください。

6. 正確な距離を知ることが重要ではない場合、代わりに ST_WithinDistance メソッドを使用してクエリを作成できます。これは、特定のデータセット (特に、大きなジオメトリ) に対してよりよいパフォーマンスが得られます。

```
SELECT c.PostalCode, c.City, z.CenterPoint
FROM Massdata z, 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;
```

ST_Geometry タイプの ST_WithinDistance メソッド252 ページを参照してください。

7. 「レッスン 5：SVG への空間データの出力」65 ページに進みます。

レッスン 5：SVG への空間データの出力

このレッスンでは、SVG ドキュメントを作成し、WKT で表現された複数多角形を表示します。ジオメトリを SVG フォーマットにエクスポートすると、Interactive SQL または SVG に互換性のあるアプリケーションで表示できます。

♦ ジオメトリを表示するために SVG として出力する

1. Interactive SQL で次の文を実行して、ジオメトリの例が設定された変数を作成します。

```
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. Interactive SQL で、次の SELECT 文を実行して、ST_AsSVG メソッドを呼び出します。

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

結果セットには単一のローがあり、SVG イメージが含まれています。このイメージは、Interactive SQL の **[SVG プレビュー]** 機能を使用して表示できます。これを行うには、結果ローをダブルクリックして、**[SVG プレビュー]** タブを選択します。

データベースから完全な値を読み込めないことを示すエラーを受け取った場合、Interactive SQL の **[トランケーションの長さ]** 設定を変更する必要があります。これを行うには、Interactive SQL で、**[ツール]** » **[オプション]** » **[SQL Anywhere]** をクリックし、**[トランケーションの長さ]** に 100000 などの大きな値を設定します。クエリを再び実行してジオメトリを表示します。

3. 前の手順では、SVG イメージを Interactive SQL 内でプレビューする方法を説明しました。ただし、結果の SVG をファイルに書き込んで、外部のアプリケーションで読み込めるようにするとより便利です。xp_write_file システムプロシージャーまたは WRITE_CLIENT_FILE 関数 [文字列] を使用すると、データベースサーバーまたはクライアントコンピューターの相対位置にあるファイルに書き込むことができます。この例では、OUTPUT 文 [Interactive SQL] を使用します。

Interactive SQL で、次の SELECT 文を実行して ST_AsSVG メソッドを呼び出し、ジオメトリを *myPolygon.svg* という名前のファイルに出力します。

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

この文には、QUOTE '' と ESCAPES OFF 句を含める必要があります。そうしないと、ホワイトスペースを維持するために改行復帰文字と一重引用符が XML に挿入され、出力が無効な SVG ファイルになります。

4. Web ブラウザまたは SVG イメージの表示をサポートするアプリケーションで SVG を開きます。または、SVG をテキストエディターで開くと、ジオメトリの XML を表示できます。

5. ST_AsSVG メソッドは、単一のジオメトリから SVG イメージを生成します。場合によっては、グループ内のすべてのシェイプが含まれる SVG イメージを生成することがあります。ST_AsSVGAggr メソッドは、複数のジオメトリを単一の SVG イメージに結合する集合関数です。まず、Interactive SQL を使用して、SVG イメージを保持する変数を作成し、ST_AsSVGAggr メソッドを使用してそれを生成します。

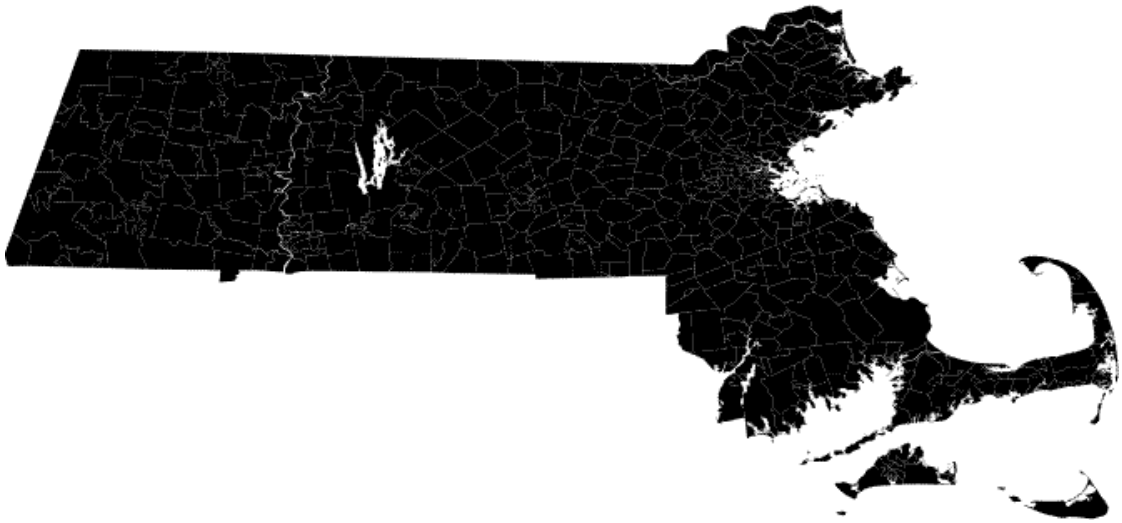
```
CREATE OR REPLACE VARIABLE @svg XML;
SELECT ST_Geometry::ST_AsSVGAggr( geometry, 'attribute=fill="black"' )
INTO @svg
FROM Massdata;
```

@svg 変数は、現在、Massdata テーブル内のすべての郵便番号区域を表す SVG イメージを保持しています。'attribute=fill="black"' は、生成されたイメージに使用される塗りつぶしの色を

指定します。指定しない場合、データベースサーバーは任意の塗りつぶしの色を選択します。目的の SVG イメージを含んだ変数が作成されたので、それをファイルに書き込んで別のアプリケーションで表示できます。Interactive SQL で次の文を実行して、SVG イメージをデータベースサーバーの相対位置にあるファイルに書き込みます。

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

WRITE_CLIENT_FILE 関数も、クライアントアプリケーションの相対位置にあるファイルに書き込むために使用できますが、適切なパーミッションを有効にするために追加の手順が必要となります。SVG データをサポートするアプリケーションで SVG イメージを開くと、次のようなイメージが表示されます。



このイメージは一様に黒ではありません。隣接する郵便番号区域の境界の間に小さなギャップがあります。ジオメトリ間には実際に白線があり、これは SVG が描画される方式の特徴です。現実にはデータにギャップはありません。太くて白い線は河川と湖です。

次の項を参照してください。

- 「OUTPUT 文 [Interactive SQL]」『SQL Anywhere サーバー SQL リファレンス』
- ST_Geometry タイプの ST_AsSVG メソッド129 ページ
- ST_Geometry タイプの ST_AsSVGAggr メソッド135 ページ
- 「xp_write_file システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』
- 「WRITE_CLIENT_FILE 関数 [文字列]」『SQL Anywhere サーバー SQL リファレンス』

6. 「レッスン 6：空間データの投影」67 ページに進みます。

レッスン 6：空間データの投影

このレッスンでは、面積の計算と距離の測定ができるように、平面モデルを使用する空間参照系にデータを投影する方法を説明します。

Massdata の空間値は、ESRI シェイプファイルからデータベースにデータをロードするときに、SRID 4269 (NAD83 空間参照系) が割り当てられています。SRID 4269 は曲面の空間参照系です。ただし、ジオメトリの面積やいくつかの空間述部などの計算は、曲面モデルではサポートされません。現在、データが曲面空間参照系に関連付けられている場合、平面空間参照系に値を投影する新しい空間カラムを作成し、そのカラムに対して計算を実行できます。

♦ データの投影

1. 郵便番号区域を表す多角形の面積を測定するには、Massdata.geometry のデータを平面空間参照系に投影する必要があります。

   Massdata.geometry のデータを投影する適切な SRID を選択するには、Interactive SQL を使用して、次のように単語 Massachusetts を含む SRID を ST_SPATIAL_REFERENCE_SYSTEMS 統合ビューに問い合わせます。

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

   これにより、Massachusetts のデータに使用するのに適したいくつかの SRID が返されます。このチュートリアルの用途には、SRID 「3586」 を使用します。

2. 次に、ST_Transform メソッドを使用してジオメトリを SRID 3586 に投影するカラム Massdata.proj_geometry を作成する必要があります。削除するには、Interactive SQL で次の文を実行します。

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

   ST_Geometry タイプの ST_Transform メソッド247 ページを参照してください。

3. Massdata.proj_geometry を使用して面積を計算できます。たとえば、Interactive SQL で次の文を実行します。

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

   注意
   ST_Area は曲面空間参照系ではサポートされません。ST_Distance はサポートされますが、ポイントジオメトリ間のみです。

4. 別の空間参照系への投影の影響を距離の計算で確認するには、次のクエリを使用し、曲面モデル (より正確) および投影された平面モデルを使用して、郵便番号区域の中心点間の距離を計算します。このデータでは両方のモデルがかなり一致します。これは、選択された投影がこのデータセットに適しているためです。

   ```
   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;
```

5. 郵便番号区域 01775 に隣接する郵便番号区域を検索するとします。これを行うには、ST_Touches メソッドを使用します。ST_Touches メソッドは、ジオメトリを比較して、ジオメトリが別のジオメトリに重なり合わずに接触しているかどうかをチェックします。ST_Touches の結果には、郵便番号 01775 のローは含まれないことに注意してください (ST_Intersects メソッドとは異なります)。

   ```
   DROP VARIABLE @Mass_01775;
   CREATE 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;
   ```

   ST_Geometry タイプの ST_Touches メソッド246 ページを参照してください。

6. ST_UnionAggr メソッドを使用すると、郵便番号区域のグループの結合を表すジオメトリが返されます。たとえば、隣接する郵便番号区域の論理和 (ただし 01775 を含まない) を反映したジオメトリを表示するとします。

   Interactive SQL で、**[ツール] » [空間ビューアー]** をクリックして、次のクエリを実行します。

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

   結果をダブルクリックして表示します。

   データベースから完全なカラムを読み込めないことを示すエラーを受け取った場合、Interactive SQL の **[トランケーションの長さ]** 設定の値を増やします。これを行うには、Interactive SQL で、**[ツール] » [オプション] » [SQL Anywhere]** をクリックして、**[トランケーションの長さ]** に大きな値を設定します。クエリを再び実行してジオメトリを表示します。

   ST_Geometry タイプの ST_UnionAggr メソッド250 ページを参照してください。

7. これで、「空間機能の実験」 チュートリアルは終了です。

8. サンプルデータベース (demo.db) を元の状態にリストアするには、「サンプルデータベースの再作成 (demo.db)」『SQL Anywhere 12 紹介』の手順に従ってください。

空間データへのアクセスとそのデータの操作

この項では、空間データへのアクセス、そのデータの操作および分析に使用できるタイプ、メソッド、コンストラクターについて説明します。空間データ型は、データ型やクラスと同じように考えることができます。各空間データ型には、データにアクセスするために使用するメソッドとコンストラクターが関連付けられています。

空間データ型の関連付けに関する概要説明については、「サポートされる空間データ型とその階層」7 ページを参照してください。

他の製品との互換性を保つために、SQL Anywhere では空間データを操作するためのいくつかの SQL 関数もサポートされています。ほとんどすべての場合において、これらの互換関数は空間メソッドのいずれかを使用して操作を実行します。そのため、基本となるメソッドへのリンクが用意されています。「空間互換関数」355 ページを参照してください。

ST_CircularString タイプ

ST_CircularString タイプは、コントロールポイント間で輪状線セグメントを使用する ST_Curve のサブタイプです。

直接のスーパータイプ

- 「ST_Curve タイプ」

コンストラクター

- 「ST_CircularString コンストラクター」

メソッド

- ST_CircularString のメソッド：

ST_NumPoints	ST_PointN		

- また、「ST_Curve タイプ」 のすべてのメソッドを ST_CircularString タイプで呼び出すことができます。

ST_CurveToLine	ST_EndPoint	ST_IsClosed	ST_IsRing
ST_Length	ST_StartPoint		

- また、「ST_Geometry タイプ」 のすべてのメソッドを ST_CircularString タイプで呼び出すことができます。

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

備考

ST_CircularString タイプは、コントロールポイント間で輪状線セグメントを使用する ST_Curve のサブタイプです。最初の 3 つのポイントは、次のようにセグメントを定義します。最初のポイントは、セグメントの開始ポイントです。2 番目のポイントは、セグメント上の開始ポイントと終了ポイント以外の任意のポイントです。3 番目のポイントは、セグメントの終了ポイントです。後続セグメントは、2 つのポイント (中間ポイントと終了ポイント) のみで定義されます。開始ポイントは、先行セグメントの終了ポイントと見なされます。

円ストリングは、開始ポイントと終了ポイントが一致する場合、3 つのポイントで構成される完全な円になることがあります。この場合、中間ポイントはセグメントの中央のポイントになります。

開始ポイント、中間ポイント、終了ポイントが同一線上にある場合、セグメントは開始ポイントと終了ポイントの間の直線セグメントになります。

正確に 3 つのポイントで構成される円ストリングは円弧です。円リングは、閉じている単純な円ストリングです。

円ストリングは、曲面の空間参照系では使用できません。たとえば、SRID 4326 の円ストリングを作成しようとすると、エラーが返されます。

標準と互換性

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

ST_CircularString コンストラクター

円ストリングを構成します。

オーバーロードリスト

名前	説明
「ST_CircularString() コンストラクター」	空のセットを表す円ストリングを構成します。
「ST_CircularString(LONG VARCHAR[, INT]) コンストラクター」	テキスト表現から円ストリングを構成します。
「ST_CircularString(LONG BINARY[, INT]) コンストラクター」	Well Known Binary (WKB) から円ストリングを構成します。
「ST_CircularString(ST_Point,ST_Point,ST_Point,...) コンストラクター」	指定した空間参照系のポイントのリストから円ストリング値を構成します。

ST_CircularString() コンストラクター

空のセットを表す円ストリングを構成します。

構文

NEW ST_CircularString()

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 標準機能

例

次の例では、値が空であることを示す 1 を返します。

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

ST_CircularString(LONG VARCHAR[, INT]) コンストラクター

テキスト表現から円ストリングを構成します。

構文

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

パラメーター

名前	型	説明
text-representation	LONG VARCHAR	円ストリングのテキスト表現を含む文字列。入力には、Well Know Text (WKT) や 拡張 Well Know Text (EWKT) など、サポートされている任意のテキスト入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

文字列表現から円ストリングを構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

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

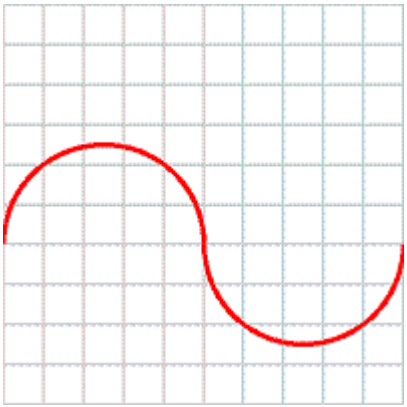
例

次の例では、CircularString (5 10, 10 12, 15 10) を返します。

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

次の例では、2 つの半円セグメントで構成される円ストリングを示します。

```
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]) コンストラクター

Well Known Binary (WKB) から円ストリングを構成します。

構文

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

パラメーター

名前	型	説明
wkb	LONG BINARY	円ストリングのバイナリ表現を含む文字列。入力には、Well Known Binary (WKB) や拡張 Well Know Binary (EWKB) など、サポートされている任意のバイナリ入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

バイナリ文字列表現から円ストリングを構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

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

例

次の例では、CircularString (5 10, 10 12, 15 10) を返します。

```
SELECT NEW
ST_CircularString(0x0108000000030000000000000000014400000000000002440000000000000244000000000000028400000000000002e400000000000002440)
```

ST_CircularString(ST_Point,ST_Point,ST_Point,...) コンストラクター

指定した空間参照系のポイントのリストから円ストリング値を構成します。

構文

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

パラメーター

名前	型	説明
pt1	ST_Point	セグメントの最初のポイント。
pt2	ST_Point	セグメント上の最初のポイントと最後のポイントとの間の任意のポイント。
pt3	ST_Point	セグメントの最後のポイント。
pt4,...,ptN	ST_Point	それ以降のセグメントを定義する追加ポイント。各ポイントは、前の終了ポイントで始まり、最初の追加ポイントを経由し、2 番目の追加ポイントで終了します。

備考

ポイントのリストから円ストリング値を構成します。少なくとも 3 つのポイントを指定してください。3 つのうち最初のポイントはセグメントの開始ポイント、3 番目のポイントはセグメントの終了ポイント、2 番目のポイントはセグメント上の最初のポイントと 3 番目のポイントとの間の任意のポイントです。追加ポイントをペアで指定して、それ以降のセグメントを円ストリングに追加できます。指定するすべてのポイントの SRID を同じにしてください。円ストリングは、この共通 SRID を使用して構成されます。指定するすべてのポイントが空ではなく、Is3D と IsMeasured に対して同じ回答を示す必要があります。すべてのポイントが 3D の場合に円ストリングも 3D になり、すべてのポイントが測定される場合に円ストリングも測定されます。

注意
ST_CircularString では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例ではエラーを返します。少なくとも 3 つのポイントの指定が必要です。

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

次の例では、結果として CircularString (0 0, 1 1, 2 0) を返します。

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

次の例ではエラーを返します。最初のセグメントでは 3 つのポイントが使用され、それ以降のセグメントでは 2 つのポイントが使用されます。

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

次の例では、結果として CircularString (0 0, 1 1, 2 0, 1 -1, 0 0) を返します。

```
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 メソッド

円ストリングを定義しているポイント数を返します。

> **注意**
> ST_NumPoints では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

circularstring-expression.**ST_NumPoints**()

戻り値

- **INT**　円ストリング値が空の場合は NULL を返し、それ以外の場合は値に含まれるポイント数を返します。

参照

- ST_CircularString タイプの ST_PointN メソッド
- ST_LineString タイプの ST_NumPoints メソッド

標準と互換性

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

例

次の例では、結果として 5 を返します。

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

ST_PointN メソッド

円ストリングの *n* 番目のポイントを返します。

> **注意**
> ST_PointN では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

circularstring-expression.**ST_PointN**(*n*)

パラメーター

名前	型	説明
n	INT	返す要素の位置 (1 ～ *circularstring-expression*.ST_NumPoints())。

戻り値

- **ST_Point**　*circular-expression* の値が空のセットの場合は、NULL を返します。指定された位置 *n* が 1 未満か、ポイント数を超えている場合は、NULL を返します。それ以外の場合は、位置 n の ST_Point 値を返します。

  結果の空間参照系識別子は、*circularstring-expression* の空間参照系と同じです。

参照

- ST_CircularString タイプの ST_NumPoints メソッド
- ST_LineString タイプの ST_PointN メソッド

標準と互換性

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

例

次の例では、結果として Point (2 0) を返します。

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

次の例では、geom の各ポイントにつき 1 つのローを返します。

```
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
```

この例では、次の結果セットを返します。

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

ST_CompoundCurve タイプ

複合曲線は一連の ST_Curve 値であり、隣接する曲線はそれぞれの終了ポイントでジョインされます。関係する曲線は、ST_LineString と ST_CircularString に制限されます。最初の曲線以降の各曲線の開始ポイントは、前の曲線の終了ポイントと一致します。

直接のスーパータイプ

- 「ST_Curve タイプ」

コンストラクター

- 「ST_CompoundCurve コンストラクター」

メソッド

- ST_CompoundCurve のメソッド：

ST_CurveN	ST_NumCurves		

- また、「ST_Curve タイプ」 のすべてのメソッドを ST_CompoundCurve タイプで呼び出すことができます。

ST_CurveToLine	ST_EndPoint	ST_IsClosed	ST_IsRing
ST_Length	ST_StartPoint		

- また、「ST_Geometry タイプ」 のすべてのメソッドを ST_CompoundCurve タイプで呼び出すことができます。

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

標準と互換性

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

ST_CompoundCurve コンストラクター

複合曲線を構成します。

オーバーロードリスト

名前	説明
「ST_CompoundCurve() コンストラクター」	空のセットを表す複合曲線を構成します。
「ST_CompoundCurve(LONG VARCHAR[, INT]) コンストラクター」	テキスト表現から複合曲線を構成します。
「ST_CompoundCurve(LONG BINARY[, INT]) コンストラクター」	Well Known Binary (WKB) から複合曲線を構成します。
「ST_CompoundCurve(ST_Curve,...) コンストラクター」	曲線のリストから複合曲線を構成します。

ST_CompoundCurve() コンストラクター

空のセットを表す複合曲線を構成します。

構文

NEW ST_CompoundCurve()

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 標準機能

例

次の例では、値が空であることを示す 1 を返します。

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

ST_CompoundCurve(LONG VARCHAR[, INT]) コンストラクター

テキスト表現から複合曲線を構成します。

構文

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

パラメーター

名前	型	説明
text-representation	LONG VARCHAR	複合曲線のテキスト表現を含む文字列。入力には、Well Know Text (WKT) や 拡張 Well Know Text (EWKT) など、サポートされている任意のテキスト入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

文字列表現から複合曲線を構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

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

例

次の例では、CompoundCurve ((0 0, 5 10), CircularString (5 10, 10 12, 15 10)) を返します。

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

ST_CompoundCurve(LONG BINARY[, INT]) コンストラクター

Well Known Binary (WKB) から複合曲線を構成します。

構文

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

パラメーター

名前	型	説明
wkb	LONG BINARY	複合曲線のバイナリ表現を含む文字列。入力には、Well Known Binary (WKB) や 拡張 Well Know Binary (EWKB) など、サポートされている任意のバイナリ入力フォーマットを使用できます。

名前	型	説明
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

バイナリ文字列表現から複合曲線を構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

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

例

次の例では、CompoundCurve ((0 0, 5 10)) を返します。

```
SELECT NEW
ST_CompoundCurve(0x01090000000100000001020000000200000000000000000000000000000000000000000000000000144000000000000002440)
```

ST_CompoundCurve(ST_Curve,...) コンストラクター

曲線のリストから複合曲線を構成します。

構文

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

パラメーター

名前	型	説明
curve1	ST_Curve	複合曲線に含める最初の曲線。
curve2,...,curveN	ST_Curve	複合曲線に含める追加の曲線。

備考

構成要素となる曲線のリストから複合曲線を構成します。最初の曲線以降の各曲線の開始ポイントは、前の曲線の終了ポイントと一致する必要があります。指定するすべての曲線の SRID を同じにしてください。複合曲線は、この共通 SRID を使用して構成されます。指定するすべての曲線が空ではなく、Is3D と IsMeasured に対して同じ回答を示す必要があります。すべてのポイントが 3D の場合に複合曲線も 3D になり、すべてのポイントが測定される場合に複合曲線も測定されます。

注意
ST_CompoundCurve では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、CompoundCurve ((0 0, 5 10), CircularString (5 10, 10 12, 15 10)) を返します。

```
SELECT NEW ST_CompoundCurve(NEW ST_LineString( 'LineString(0 0, 5 10)'),NEW
ST_CircularString('CircularString (5 10, 10 12, 15 10)'))
```

ST_CurveN メソッド

複合曲線の *n* 番目の曲線を返します。

注意
ST_CurveN では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

compoundcurve-expression.**ST_CurveN**(*n*)

パラメーター

名前	型	説明
n	INT	返す要素の位置 (1 ～ *compoundcurve-expression*.ST_NumCurves())。

戻り値

- **ST_Curve** 複合曲線の *n* 番目の曲線を返します。

  結果の空間参照系識別子は、*compoundcurve-expression* の空間参照系と同じです。

標準と互換性

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

例

次の例では、結果として CircularString (0 0, 1 1, 2 0) を返します。

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

ST_NumCurves メソッド

複合曲線を定義している曲線数を返します。

> **注意**
> ST_NumCurves では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

compoundcurve-expression.**ST_NumCurves**()

戻り値

- **INT**　対象の複合曲線に含まれている曲線数を返します。

標準と互換性

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

例

次の例では、結果として 2 を返します。

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

ST_Curve タイプ

ST_Curve タイプは、一連のポイントを使用して線を表すタイプのサブタイプです。

直接のスーパータイプ

- 「ST_Geometry タイプ」

直接のサブタイプ

- 「ST_CircularString タイプ」
- 「ST_CompoundCurve タイプ」
- 「ST_LineString タイプ」

メソッド

●ST_Curve のメソッド：

ST_CurveToLine	ST_EndPoint	ST_IsClosed	ST_IsRing
ST_Length	ST_StartPoint		

●また、「ST_Geometry タイプ」 のすべてのメソッドを ST_Curve タイプで呼び出すことができます。

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

備考

ST_Curve タイプは、一連のポイントを使用して線を表すタイプのサブタイプです。サブタイプでは、コントロールポイントを直線セグメント (ST_LineString)、輪状線セグメント (ST_CircularString)、または組み合わせ (ST_CompoundCurve) のいずれを使用してジョインするかを指定します。

ST_Curve タイプはインスタンス化可能ではありません。

ST_Curve 値がそれ自体と交差しない場合 (終了ポイントで交差する場合を除く)、その値は単純です。ST_Curve 値が終了ポイントで交差する場合、その値は閉じています。閉じていて単純な ST_Curve 値はリングと呼ばれます。

標準と互換性

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

ST_CurveToLine メソッド

ST_Curve 値の ST_LineString 補間近似値を返します。

構文

curve-expression.**ST_CurveToLine**()

戻り値

- **ST_LineString** *curve-expression* の ST_LineString 補間近似値を返します。

  結果の空間参照系識別子は、*curve-expression* の空間参照系と同じです。

備考

curve-expression が空の場合は、ST_CurveToLine メソッドは ST_LineString タイプの空のセットを返します。それ以外の場合は、ST_CurveToLine は、*curve-expression* に指定された線ストリングを含む 1 つの線ストリングを、*curve-expression* に円ストリングがある場合には、その補間近似値と結合した形で返します。

注意
ST_CurveToLine では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToLineString メソッド
- 「補間が空間の計算に与える影響」

標準と互換性

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

例

次の例では、結果として LineString (0 7, 0 4, 4 4) (元の線ストリングのコピー) を返します。

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

次の例では、結果として LineString (0 0, 5 10) (同等の線ストリングに変換された複合曲線) を返します。

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

次の例では、元の円ストリングに近似する、補間された線ストリングを返します。

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

ST_EndPoint メソッド

ST_Curve 値の終了ポイントである ST_Point 値を返します。

注意
ST_EndPoint では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

curve-expression.**ST_EndPoint**()

戻り値

- **ST_Point** 曲線が空のセットの場合は、NULL を返します。それ以外の場合は、曲線の終了ポイントを返します。

  結果の空間参照系識別子は、*curve-expression* の空間参照系と同じです。

参照

- ST_Curve タイプの ST_StartPoint メソッド

標準と互換性

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

例

次の例では、結果として Point (5 10) を返します。

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

ST_IsClosed メソッド

ST_Curve 値が閉じているかどうかをテストします。開始ポイントと終了ポイントが一致する場合、曲線は閉じています。

> **注意**
> ST_IsClosed では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

curve-expression.**ST_IsClosed**()

戻り値

- **BIT** 曲線が閉じている場合 (および空でない場合) は、1 を返します。それ以外の場合は、0 を返します。

参照

- ST_MultiCurve タイプの ST_IsClosed メソッド
- ST_Curve タイプの ST_IsRing メソッド
- ST_Geometry タイプの ST_Boundary メソッド

標準と互換性

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

例

次の例では、閉じた曲線を含む SpatialShapes のすべてのローを返します。Shape が ST_Curve のサブタイプでない場合に、TREAT 関数が実行されないようにするために、IF 式を使用する必要があります。IF 式を使用しない場合は、サーバーが WHERE 句の条件を並び替えてしまう場合があるため、エラーになります。

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

次の例では、閉じたジオメトリを含む curve_table のすべてのローを返します。この例では、geometry カラムに ST_Curve、ST_LineString、ST_CircularString、または ST_CompoundCurve タイプがあることを前提としています。

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

ST_IsRing メソッド

ST_Curve 値がリングかどうかをテストします。曲線が閉じていて単純な場合 (それ自体と交差しない場合)、その曲線はリングです。

構文

curve-expression.**ST_IsRing**()

戻り値

- **BIT**　曲線がリングの場合 (および空でない場合) は、1 を返します。それ以外の場合は、0 を返します。

参照

- ST_Curve タイプの ST_IsClosed メソッド
- ST_Geometry タイプの ST_IsSimple メソッド

標準と互換性

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

例

次の例では、リングを含む SpatialShapes のすべてのローを返します。Shape が ST_Curve のサブタイプでない場合に、TREAT 関数が実行されないようにするために、IF 式を使用する必要があります。IF 式を使用しない場合は、サーバーが WHERE 句の条件を並び替えてしまう場合があるため、エラーになります。

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

次の例では、リングのジオメトリを含む curve_table のすべてのローを返します。この例では、geometry カラムに ST_Curve、ST_LineString、ST_CircularString、または ST_CompoundCurve タイプがあることを前提としています。

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

ST_Length メソッド

ST_Curve 値の長さの測定値を返します。結果は、unit-name パラメーターで指定した単位で測定されます。

構文

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

パラメーター

名前	型	説明
unit-name	VARCHAR(128)	長さを計算する単位。デフォルトでは、空間参照系の単位が使用されます。単位名は、UNIT_TYPE が 'LINEAR' の ST_UNITS_OF_MEASURE ビュー内のローの UNIT_NAME カラムと一致させてください。

戻り値

- **DOUBLE**　曲線が空のセットの場合は、NULL を返します。それ以外の場合は、指定された単位で曲線の長さを返します。

備考

ST_Length メソッドは、*unit-name* パラメーターで指定された単位で曲線の長さを返します。曲線が空の場合は、NULL が返されます。

曲線に Z 値が含まれている場合、それらの値はジオメトリの長さの計算時には考慮されません。

注意
curve-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_Length では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_MultiCurve タイプの ST_Length メソッド
- ST_Surface タイプの ST_Area メソッド

標準と互換性

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

例

次の例では、結果として 2 を返します。

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

次の例では、半円を表す円ストリングを作成し、ST_Length を使用してジオメトリの長さを調べ、値 PI を返します。

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

次の例では、カナダのハリファックス (NS) からワーテルロー (ON) までの経路を表す線ストリングを作成し、ST_Length を使用してその経路の長さ (メートル単位) を調べ、結果として 1361967.76789 を返します。

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

次の例では、SpatialShapes テーブルの曲線の長さを返します。長さは直交座標の単位で返されます。

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

次の例では、線ストリングと測定単位の例 (example_unit_halfmetre) を作成します。ST_Length メソッドは、この測定単位でジオメトリの長さを検索し、値 4.0 を返します。

```
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 メソッド

ST_Curve 値の開始ポイントである ST_Point 値を返します。

> **注意**
> ST_StartPoint では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

curve-expression.**ST_StartPoint**()

戻り値

- **ST_Point**　曲線が空のセットの場合は、NULL を返します。それ以外の場合は、曲線の開始ポイントを返します。

  結果の空間参照系識別子は、*curve-expression* の空間参照系と同じです。

参照

- ST_Curve タイプの ST_EndPoint メソッド

標準と互換性

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

例

次の例では、結果として Point (0 0) を返します。

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

ST_CurvePolygon タイプ

ST_CurvePolygon は、1 つの外部リングと 0 個以上の内部リングで定義された平面を表します。

直接のスーパータイプ

- 「ST_Surface タイプ」

直接のサブタイプ

- 「ST_Polygon タイプ」

コンストラクター

- 「ST_CurvePolygon コンストラクター」

メソッド

- ST_CurvePolygon のメソッド：

ST_CurvePolyToPoly	ST_ExteriorRing	ST_InteriorRingN	ST_NumInteriorRing

- また、「ST_Surface タイプ」 のすべてのメソッドを ST_CurvePolygon タイプで呼び出すことができます。

ST_Area	ST_Centroid	ST_IsWorld	ST_Perimeter
ST_PointOnSurface			

- また、「ST_Geometry タイプ」 のすべてのメソッドを ST_CurvePolygon タイプで呼び出すことができます。

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

備考

ST_CurvePolygon は、1 つの外部リングと、面の穴を表す 0 個以上の内部リングで定義された平面を表します。ST_CurvePolygon の外部リングと内部リングは、任意の ST_Curve 値にすることができます。たとえば、円は、境界を表す ST_CircularString 外部リングを含む ST_CurvePolygon です。ST_CurvePolygon 内の 2 つのリングは、1 つのポイント以外で交差しないようにする必要があります。さらに、ST_CurvePolygon には、切断線、突起、または陥没を含めないようにしてください。

すべての ST_CurvePolygon の内部は、接続されたポイント集合です。

標準と互換性

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

ST_CurvePolygon コンストラクター

曲線多角形を構成します。

オーバーロードリスト

名前	説明
「ST_CurvePolygon() コンストラクター」	空のセットを表す曲線多角形を構成します。
「ST_CurvePolygon(LONG VARCHAR[, INT]) コンストラクター」	テキスト表現から曲線多角形を構成します。
「ST_CurvePolygon(LONG BINARY[, INT]) コンストラクター」	Well Known Binary (WKB) から曲線多角形を構成します。
「ST_CurvePolygon(ST_Curve,...) コンストラクター」	外部リングを表す曲線と、内部リングを表す曲線のリスト (指定した空間参照系内のすべて) から曲線多角形を作成します。
「ST_CurvePolygon(ST_MultiCurve[, VARCHAR(128)]) コンストラクター」	外部リングを含む複数曲線と、内部リングのオプションリストから曲線多角形を作成します。

ST_CurvePolygon() コンストラクター

空のセットを表す曲線多角形を構成します。

構文

NEW ST_CurvePolygon()

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 標準機能

例

次の例では、値が空であることを示す 1 を返します。

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

ST_CurvePolygon(LONG VARCHAR[, INT]) コンストラクター

テキスト表現から曲線多角形を構成します。

構文

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

パラメーター

名前	型	説明
text-representation	LONG VARCHAR	曲線多角形のテキスト表現を含む文字列。入力には、Well Know Text (WKT) や 拡張 Well Know Text (EWKT) など、サポートされている任意のテキスト入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

文字列表現から曲線多角形を構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

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

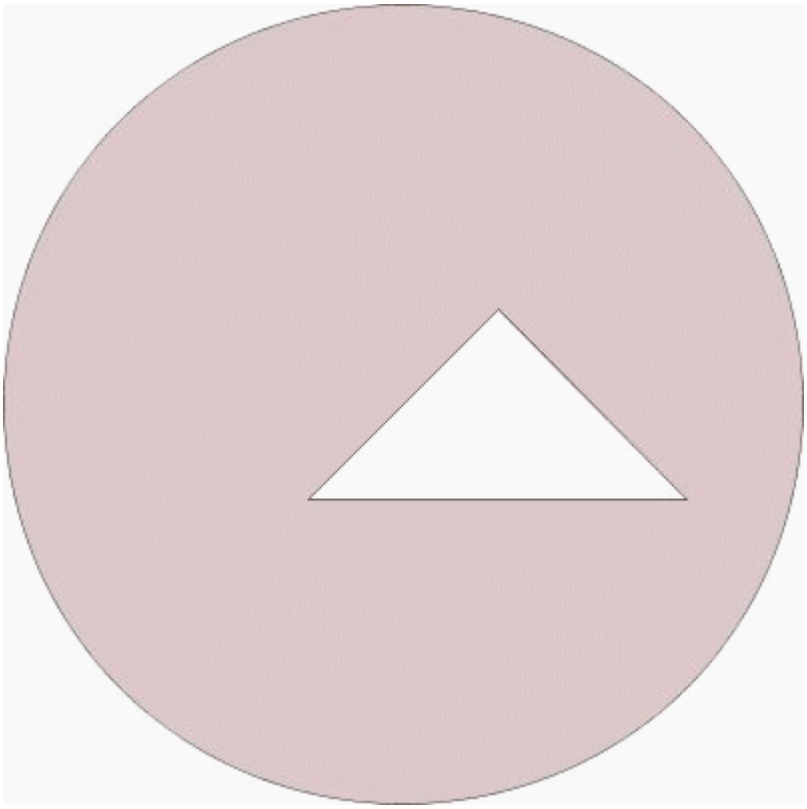
例

次の例では、CurvePolygon (CompoundCurve (CircularString (-5 -5, 0 -5, 5 -5), (5 -5, 0 5, -5 -5))) を返します。

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

次の例は、外部リングの円と三角形の内部リングを含む曲線多角形を示します。

```
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]) コンストラクター

Well Known Binary (WKB) から曲線多角形を構成します。

構文

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

パラメーター

名前	型	説明
wkb	LONG BINARY	曲線多角形のバイナリ表現を含む文字列。入力には、Well Known Binary (WKB) や 拡張 Well Know Binary (EWKB) など、サポートされている任意のバイナリ入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

バイナリ文字列表現から曲線多角形を構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

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

例

次の例では、CurvePolygon (CircularString (0 0, 10 0, 10 10, 0 10, 0 0)) を返します。

```
SELECT NEW
ST_CurvePolygon(0x010a00000001000000010800000005000000000000000000000000000000
0000000000000024400000000000000000000000000000244000000000000024400000000000000000
00000000000244000000000000000000000000000000000)
```

ST_CurvePolygon(ST_Curve,...) コンストラクター

外部リングを表す曲線と、内部リングを表す曲線のリスト (指定した空間参照系内のすべて) から曲線多角形を作成します。

構文

NEW ST_CurvePolygon(*exterior-ring*[,*interior-ring1*,...,*interior-ringN*])

パラメーター

名前	型	説明
exterior-ring	ST_Curve	曲線多角形の外部リング
interior-ring1,...,interior-ringN	ST_Curve	曲線多角形の内部リング

備考

外部リングを表す曲線と、内部リングを表す曲線のリスト (空の可能性もある) から曲線多角形を作成します。指定するすべてのリングの SRID を同じにしてください。多角形は、この共通 SRID を使用して作成されます。指定するすべてのリングが空ではなく、Is3D と IsMeasured に対して同じ回答を示す必要があります。すべてのポイントが 3D の場合に多角形も 3D になり、すべてのポイントが測定される場合に多角形も測定されます。

> **注意**
> ST_CurvePolygon では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

内部リングの可変長リストを指定する機能はベンダー拡張です。

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

例

次の例では、CurvePolygon ((-5 -1, 5 -1, 0 9, -5 -1), CircularString (-2 2, -2 4, 2 4, 2 2, -2 2)) (円孔のある三角形) を返します。

```
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)]) コンストラクター

外部リングを含む複数曲線と、内部リングのオプションリストから曲線多角形を作成します。

構文

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

パラメーター

名前	型	説明
multi-curve	ST_MultiCurve	外部リングと (オプションの) 一連の内部リングを含む複数曲線値。
polygon-format	VARCHAR(128)	指定した曲線を解釈するときに使用する多角形フォーマットの文字列。有効なフォーマットは、'CounterClockwise'、'Clockwise'、'EvenOdd' です。

備考

外部リングを含む複数曲線と、内部リングのオプションリストから曲線多角形を作成します。

polygon-format パラメーターを指定すると、リングが外部リングと内部リングのいずれであるかを判断するためにサーバーで使用されるアルゴリズムが選択されます。指定しない場合は、空間参照系の多角形フォーマットが使用されます。

polygon-format の詳細については、POLYGON FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

注意
ST_CurvePolygon では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として CurvePolygon (CircularString (-2 0, 1 -3, 4 0, 1 3, -2 0), (0 0, 1 1, 2 0, 0 0)) (三角孔のある円形の曲線多角形) を返します。

```
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 メソッド

曲線多角形の補間近似値を多角形として返します。

構文

curvepolygon-expression.**ST_CurvePolyToPoly**()

戻り値

- **ST_Polygon** *curvepolygon-expression* の補間近似値を多角形として返します。

  結果の空間参照系識別子は、*curvepolygon-expression* の空間参照系と同じです。

備考

curvepolygon-expression が空の場合は、ST_CurvePolyToPoly メソッドは ST_Polygon タイプの空のセットを返します。それ以外の場合は、ST_CurvePolyToPoly は、*curve-expression* に指定された線形リングを含む 1 つの多角形を、*curvepolygon-expression* に円ストリングまたは複合曲線リングがある場合には、その補間近似値と結合した形で返します。

> **注意**
> ST_CurvePolyToPoly では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToPolygon メソッド
- 「補間が空間の計算に与える影響」

標準と互換性

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

例

次の例では、結果として Polygon ((0 0, 2 0, 1 2, 0 0)) (元の多角形のコピー) を返します。

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


次の例では、結果として Polygon ((0 0, 5 0, 5 10, 0 0)) (同等の多角形に変換された曲線多角形) を返します。

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

次の例では、元の曲線多角形に近似する、補間された多角形を返します。

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

ST_ExteriorRing メソッド

外部リングを取り出したり、変更したりします。

オーバーロードリスト

名前	説明
「ST_CurvePolygon タイプの ST_ExteriorRing() メソッド」	曲線多角形の外部リングを返します。
「ST_CurvePolygon タイプの ST_ExteriorRing(ST_Curve) メソッド」	曲線多角形の外部リングを変更します。

ST_CurvePolygon タイプの ST_ExteriorRing() メソッド

曲線多角形の外部リングを返します。

注意
ST_ExteriorRing では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

curvepolygon-expression.**ST_ExteriorRing**()

戻り値

- **ST_Curve**　外部リングを返します。

  結果の空間参照系識別子は、*curvepolygon-expression* の空間参照系と同じです。

参照

- ST_CurvePolygon タイプの ST_InteriorRingN メソッド
- ST_Polygon タイプの ST_ExteriorRing メソッド
- ST_Geometry タイプの ST_Boundary メソッド

標準と互換性

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

例

次の例では、結果として CircularString (2 0, 5 0, 5 3, 2 3, 2 0) を返します。

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

ST_CurvePolygon タイプの ST_ExteriorRing(ST_Curve) メソッド

曲線多角形の外部リングを変更します。

注意
ST_ExteriorRing では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

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

パラメーター

名前	型	説明
exterior-ring	ST_Curve	新しい外部リング値。

戻り値

- **ST_CurvePolygon** 外部リングが指定した値に変更された曲線多角形値のコピーを返します。

  結果の空間参照系識別子は、*curvepolygon-expression* の空間参照系と同じです。

標準と互換性

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

例

次の例では、結果として CurvePolygon (CircularString (2 0, 6 1, 5 5, 1 4, 2 0), (3 1, 4 2, 5 1, 3 1)) を返します。

```
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 メソッド

曲線多角形の *n* 番目の内部リングを返します。

注意
ST_InteriorRingN では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

curvepolygon-expression.**ST_InteriorRingN**(*n*)

パラメーター

名前	型	説明
n	INT	返す要素の位置 (1 ～ *curvepolygon-expression*.ST_NumInteriorRing())。

戻り値

- **ST_Curve**　曲線多角形の *n* 番目の内部リングを返します。

  結果の空間参照系識別子は、*curvepolygon-expression* の空間参照系と同じです。

参照

- ST_CurvePolygon タイプの ST_NumInteriorRing メソッド
- ST_CurvePolygon タイプの ST_ExteriorRing メソッド
- ST_Polygon タイプの ST_InteriorRingN メソッド
- ST_Geometry タイプの ST_Boundary メソッド

標準と互換性

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

例

次の例では、結果として LineString (3 1, 4 2, 5 1, 3 1) を返します。

```
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 メソッド

曲線多角形の内部リング数を返します。

注意
ST_NumInteriorRing では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

curvepolygon-expression.**ST_NumInteriorRing**()

戻り値

- **INT**　曲線多角形の内部リング数を返します。

参照

- ST_CurvePolygon タイプの ST_InteriorRingN メソッド
- ST_Polygon タイプの ST_InteriorRingN メソッド

標準と互換性

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

例

次の例では、結果として 1 を返します。

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

ST_GeomCollection タイプ

ST_GeomCollection は、0 個以上の ST_Geometry 値のコレクションです。

直接のスーパータイプ

- 「ST_Geometry タイプ」

直接のサブタイプ

- 「ST_MultiCurve タイプ」
- 「ST_MultiPoint タイプ」
- 「ST_MultiSurface タイプ」

コンストラクター

- 「ST_GeomCollection コンストラクター」

メソッド

- ST_GeomCollection のメソッド：

ST_GeomCollectionAggr	ST_GeometryN	ST_NumGeometries	

●また、「ST_Geometry タイプ」 のすべてのメソッドを ST_GeomCollection タイプで呼び出すことができます。

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

備考

ST_GeomCollection は、0 個以上の ST_Geometry 値のコレクションです。すべての値がコレクションの値と同じ空間参照系にあります。ST_GeomCollection タイプには、異なるオブジェクトのコレクション (たとえば、ポイント、線、多角形) を含めることができます。ST_GeomCollection のサブタイプを使用して、コレクションを特定のジオメトリタイプに制限できます。

ジオメトリコレクション値の次元はその構成要素の最大次元です。

すべての構成要素が単純であり、2 つの構成要素ジオメトリがそれぞれの境界以外で交差しない場合、そのジオメトリコレクションは単純です。

標準と互換性

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

ST_GeomCollection コンストラクター

ジオメトリコレクションを構成します。

オーバーロードリスト

名前	説明
「ST_GeomCollection() コンストラクター」	空のセットを表すジオメトリコレクションを構成します。
「ST_GeomCollection(LONG VARCHAR[, INT]) コンストラクター」	テキスト表現からジオメトリコレクションを構成します。
「ST_GeomCollection(LONG BINARY[, INT]) コンストラクター」	Well Known Binary (WKB) からジオメトリコレクションを構成します。
「ST_GeomCollection(ST_Geometry,...) コンストラクター」	ジオメトリ値のリストからジオメトリコレクションを構成します。

ST_GeomCollection() コンストラクター

空のセットを表すジオメトリコレクションを構成します。

構文

NEW ST_GeomCollection()

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 標準機能

例

次の例では、値が空であることを示す 1 を返します。

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

ST_GeomCollection(LONG VARCHAR[, INT]) コンストラクター

テキスト表現からジオメトリコレクションを構成します。

構文

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

パラメーター

名前	型	説明
text-representation	LONG VARCHAR	ジオメトリコレクションのテキスト表現を含む文字列。入力には、Well Know Text (WKT) や 拡張 Well Know Text (EWKT) など、サポートされている任意のテキスト入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

文字列表現からジオメトリコレクションを構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

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

例

次の例では、GeometryCollection (CircularString (5 10, 10 12, 15 10), Polygon ((10 -5, 15 5, 5 5, 10 -5))) を返します。

```
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]) コンストラクター

Well Known Binary (WKB) からジオメトリコレクションを構成します。

構文

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

パラメーター

名前	型	説明
wkb	LONG BINARY	ジオメトリコレクションのバイナリ表現を含む文字列。入力には、Well Known Binary (WKB) や 拡張 Well Know Binary (EWKB) など、サポートされている任意のバイナリ入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

バイナリ文字列表現からジオメトリコレクションを構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

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

例

次の例では、GeometryCollection (Point (10 20)) を返します。

```
SELECT NEW
ST_GeomCollection(0x0107000000010000000101000000000000000000244000000000000034400)
```

ST_GeomCollection(ST_Geometry,...) コンストラクター

ジオメトリ値のリストからジオメトリコレクションを構成します。

構文

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

パラメーター

名前	型	説明
geo1	ST_Geometry	ジオメトリコレクションの最初のジオメトリ値。
geo2,...,geoN	ST_Geometry	ジオメトリコレクションの追加のジオメトリ値。

備考

ジオメトリ値のリストからジオメトリコレクションを構成します。指定するすべてのジオメトリ値の SRID を同じにしてください。ジオメトリコレクションは、この共通 SRID を使用して構成されます。

指定するすべてのジオメトリ値が Is3D と IsMeasured に対して同じ回答を示す必要があります。すべてのジオメトリ値が 3D の場合にジオメトリコレクションも 3D になり、すべてのジオメトリ値が測定される場合にジオメトリコレクションも測定されます。

注意
ST_GeomCollection では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、1 つのポイント 'Point (1 2)' を含むジオメトリコレクションを返します。

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

次の例では、2 つのポイント 'Point (1 2)' および 'Point (3 4)' を含むジオメトリコレクションを返します。

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

ST_GeomCollectionAggr メソッド

グループ内のすべてのジオメトリを含むジオメトリコレクションを返します。

構文

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

パラメーター

名前	型	説明
geometry-column	ST_Geometry	コレクションを生成するジオメトリ値。通常、これはカラムです。

戻り値

- **ST_GeomCollection**　グループ内のすべてのジオメトリを含むジオメトリコレクションを返します。

  結果の空間参照系識別子は、最初のパラメーターの識別子と同じです。

備考

ST_GeomCollectionAggr 集合関数を使用して、ジオメトリのグループを 1 つのコレクションに結合することができます。結合するすべてのジオメトリの SRID と座標次元の両方が同じになるようにします。

引数が NULL のローは含まれません。

空のグループ、または NULL 以外の値が含まれないグループの場合、NULL を返します。

結果の ST_GeomCollection の座標次元は、各ジオメトリの座標次元と同じになります。

ST_GeometryN から目的の順序で要素が返されるように、オプションの ORDER BY 句を使用して要素を特定の順序に並べ替えることができます。この順序が意味を持たない場合は、順序を指定しない方が効率的です。その場合、要素の順序は、クエリオプティマイザーが選択したアクセスプランにより決定されます。

ST_GeomCollectionAggr は ST_UnionAggr より効率的ですが、ジオメトリのグループの中に重複するジオメトリが存在する場合には、そのような重複を伴ったコレクションを返す可能性があります。特に、返されたコレクションに重複する面が含まれている場合に、それが他の空間メソッドの入力値として使用されると、予期しない結果をもたらす場合があります。ST_UnionAggr では、重複するジオメトリを処理できます。

注意
ST_GeomCollectionAggr では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_UnionAggr メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、SpatialShapes テーブルのすべてのジオメトリを 1 つのコレクションに結合した単一の値を返します。

```
SELECT ST_GeomCollection::ST_GeomCollectionAggr( Shape ) FROM SpatialShapes
   WHERE Shape.ST_Is3D() = 0
```

ST_GeometryN メソッド

ジオメトリコレクションの *n* 番目のジオメトリを返します。

注意
ST_GeometryN では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

geomcollection-expression.**ST_GeometryN**(*n*)

パラメーター

名前	型	説明
n	INT	返す要素の位置 (1 ～ *geomcollection-expression*.ST_NumGeometries())。

戻り値

- **ST_Geometry**　ジオメトリコレクションの *n* 番目のジオメトリを返します。

  結果の空間参照系識別子は、*geomcollection-expression* の空間参照系と同じです。

参照

- ST_GeomCollection タイプの ST_NumGeometries メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.1.5

例

次の例では、結果として Polygon ((10 -5, 15 5, 5 5, 10 -5)) を返します。

```
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 メソッド

ジオメトリコレクションに含まれているジオメトリ数を返します。

注意
ST_NumGeometries では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

geomcollection-expression.**ST_NumGeometries**()

戻り値

- **INT** 対象のコレクションに格納されているジオメトリ数を返します。

参照

- ST_GeomCollection タイプの ST_GeometryN メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.1.4

例

次の例では、結果として 3 を返します。

```
SELECT NEW ST_MultiPoint('MultiPoint ((10 10), (12 12), (14 10))')
    .ST_NumGeometries()
```

ST_Geometry タイプ

ST_Geometry タイプは、ジオメトリタイプ階層の最大のスーパータイプです。

直接のサブタイプ

- 「ST_Curve タイプ」
- 「ST_GeomCollection タイプ」
- 「ST_Point タイプ」
- 「ST_Surface タイプ」

メソッド

●ST_Geometry のメソッド：

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

備考

ST_Geometry タイプは、ジオメトリタイプ階層の最大のスーパータイプです。ST_Geometry タイプでは、任意の空間値に適用できるメソッドがサポートされています。ST_Geometry タイプはインスタンス化できません。代わりに、サブタイプをインスタンス化してください。元のフォーマット (WKT または WKB) を使用する場合は、ST_GeomFromText／ST_GeomFromWKB などのメソッドを使用して、値を元のフォーマットで表す具体的なタイプをインスタンス化できます。

ST_Geometry 値に含まれるすべての値が同じ空間参照系にあります。ST_SRID メソッドを使用して、値に関連付けられている空間参照系を取り出したり、変更したりできます。

ST_Geometry タイプまたはそのサブタイプのカラムには、プライマリーキー、ユニークなインデックス、または一意性制約を入れることはできません。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1

ST_Affine メソッド

指定した 3-D アフィン変換を適用した後の新しいジオメトリを返します。

構文

geometry-expression.**ST_Affine**(*a00*,*a01*,*a02*,*a10*,*a11*,*a12*,*a20*,*a21*,*a22*,*xoff*,*yoff*,*zoff*)

パラメーター

名前	型	説明
a00	DOUBLE	ロー 0、カラム 0 のアフィンマトリックス要素
a01	DOUBLE	ロー 0、カラム 1 のアフィンマトリックス要素
a02	DOUBLE	ロー 0、カラム 2 のアフィンマトリックス要素
a10	DOUBLE	ロー 1、カラム 0 のアフィンマトリックス要素
a11	DOUBLE	ロー 1、カラム 1 のアフィンマトリックス要素
a12	DOUBLE	ロー 1、カラム 2 のアフィンマトリックス要素

名前	型	説明
a20	DOUBLE	ロー 2、カラム 0 のアフィンマトリックス要素
a21	DOUBLE	ロー 2、カラム 1 のアフィンマトリックス要素
a22	DOUBLE	ロー 2、カラム 2 のアフィンマトリックス要素
xoff	DOUBLE	平行移動の x オフセット
yoff	DOUBLE	平行移動の y オフセット
zoff	DOUBLE	平行移動の z オフセット

戻り値

- **ST_Geometry**　指定した変換を適用した後の新しいジオメトリを返します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

アフィン変換では、回転、平行移動、スケーリングが 1 回のメソッド呼び出しでまとめて実行されます。アフィン変換は、マトリックス乗算を使用して定義されます。

ポイント (x,y,z) の場合、結果 (x',y',z') は次のように計算されます。

```
/ x' ¥ / a00 a01 a02 xoff ¥ / x ¥ | y' | = | a10 a11 a12 yoff | * | y | | z' | | a20 a21 a22 zoff | | z | ¥ w' / ¥ 0 0 0 1 / ¥ 1 /
```

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として LineString (5 6, 5 3, 9 3) を返します。X 値は 5 平行移動し、Y 値は -1 平行移動します。

```
SELECT Shape.ST_Affine( 1,0,0, 0,1,0, 0,0,1, 5,-1,0 )
FROM SpatialShapes WHERE ShapeID = 5
```

次の例では、結果として LineString (.698833 6.965029, .399334 3.980017, 4.379351 3.580683) を返します。Shape は、Z 軸周りに 0.1 ラジアン (約 5.7 度) 回転します。

```
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 メソッド

ST_Geometry 値の WKB 表現を返します。

構文

geometry-expression.**ST_AsBinary**([*format*])

パラメーター

名前	型	説明
format	VARCHAR(128)	*geometry-expression* をバイナリ表現に変換するときに使用する出力バイナリフォーマットを定義する文字列。指定しない場合は、st_geometry_asbinary_format オプションの値を使用してバイナリ表現を選択します。「st_geometry_asbinary_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

戻り値

- **LONG BINARY**　*geometry-expression* の WKB 表現を返します。

備考

ST_AsBinary メソッドは、ジオメトリを表すバイナリ文字列を返します。さまざまなバイナリフォーマットが (関連付けられているオプションとともに) サポートされており、オプションの *format* パラメーターを使用して目的のフォーマットを選択します。*format* パラメーターを指定しない場合は、st_geometry_asbinary_format オプションを使用して、使用する出力フォーマットを選択します。「st_geometry_asbinary_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

フォーマット文字列では、出力フォーマットとそのフォーマットに対するパラメーターを定義します。フォーマット文字列のフォーマットは次のいずれかです。

```
format-name
format-name(parameter1=value1;parameter2=value2;...)
parameter1=value1;parameter2=value2;...
```

最初のフォーマットでは、フォーマット名を指定し、パラメーターは指定しません。すべてのフォーマットパラメーターでデフォルト値が使用されます。2 番目のフォーマットでは、フォーマット名と名前付きパラメーター値のリストを指定します。指定しないパラメーターでは、デフォルト値が使用されます。最後のフォーマットでは、パラメーター値のみを指定し、フォーマット名にはデフォルトの 'WKB' を使用します。

次のフォーマット名を使用できます。

- **WKB** SQL/MM と OGC で定義された Well-Known-Binary フォーマット
- **EWKB** PostGIS で定義された Extended-Well-Known-Binary フォーマット。このフォーマットには、ジオメトリの SRID が含まれます。また、Z 値と M 値の表現方法が WKB と異なります。

次のフォーマットパラメーターを指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
WKB	バージョン	1.2	● **1.1** OGC SFS 1.1 で定義された WKB。このフォーマットには、Z 値と M 値は含まれません。ジオメトリに Z 値または M 値が含まれている場合、それらの値は出力では削除されます。 ● **1.2** OGC SFS 1.2 で定義された WKB。これは、バージョン 1.1 の 2D データに適合し、Z 値と M 値をサポートするようにフォーマットを拡張します。	version パラメーターでは、使用される WKB 仕様のバージョンを制御します。

注意
サーバーでは、ジオメトリ値を BINARY に変換するときに ST_AsBinary メソッドが使用されます。st_geometry_asbinary_format オプションでは、変換に使用するフォーマットを定義します。「st_geometry_asbinary_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

注意
ST_AsBinary では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.37

例

st_geometry_asbinary_format オプションでデフォルト値の 'WKB' が使用される場合、次の例では結果として 0x01b90b0000000000000000f03f0000000000000040000000000000084000000000000001040 を返します。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsBinary()
```

st_geometry_asbinary_format オプションでデフォルト値の 'WKB' が使用される場合、次の例では結果として 0x01b90b0000000000000000f03f0000000000000040000000000000084000000000000001040 を返します。サーバーでは、ジオメトリを BINARY に変換するときに ST_AsBinary メソッドが暗黙的に呼び出されます。

```
SELECT CAST( NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ) AS LONG BINARY)
```

次の例では、結果として 0x0101000000000000000000f03f0000000000000040 を返します。WKB の OGC 仕様のバージョン 1.1 では Z 値と M 値はサポートされていないため、これらの値は省略されます。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsBinary('WKB(Version=1.1;endian=little)')
```

次の例では、結果として 0x01010000e0e610000000000000000f03f0000000000000040000000000000084000000000000001040 を返します。拡張 WKB には SRID が含まれています。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsBinary('EWKB(endian=little)')
```

ST_AsGML メソッド

ST_Geometry 値の GML 表現を返します。

構文

geometry-expression.**ST_AsGML**([*format*])

パラメーター

名前	型	説明
format	VARCHAR(128)	*geometry-expression* を GML 表現に変換するときに使用するパラメーターを定義する文字列。指定しない場合、デフォルトは 'GML' です。

戻り値

- **LONG VARCHAR**　*geometry-expression* の GML 表現を返します。

備考

ST_AsGML メソッドは、ジオメトリを表す GML 文字列を返します。さまざまなフォーマットが (関連付けられているオプションとともに) サポートされており、オプションの *format* パラメーターを使用して目的のフォーマットを選択します。*format* パラメーターを指定しない場合、デフォルトは 'GML' です。

フォーマット文字列では、出力フォーマットとそのフォーマットに対するパラメーターを定義します。フォーマット文字列のフォーマットは次のいずれかです。

```
format-name
format-name(parameter1=value1;parameter2=value2;...)
parameter1=value1;parameter2=value2;...
```

最初のフォーマットでは、フォーマット名を指定し、パラメーターは指定しません。すべてのフォーマットパラメーターでデフォルト値が使用されます。2 番目のフォーマットでは、フォーマット名と名前付きパラメーター値のリストを指定します。指定しないパラメーターでは、デフォルト値が使用されます。最後のフォーマットでは、パラメーター値のみを指定し、フォーマット名にはデフォルトの 'GML' を使用します。

次のフォーマット名を使用できます。

- **GML**　ISO 19136 と OGC で定義された Geography Markup Language フォーマット。

次のフォーマットパラメーターを指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
GML	ネームスペース	none	● **local** 指定した要素 (この場合、ポイント) とそのサブ要素に対してデフォルトのネームスペース属性を設定します。 ● **global** 指定した要素とそのサブ要素に対して専用の (gml) プレフィクスを設定します。これは、集約操作の中でクエリを使用し、ある最上位レベルの要素で "gml" プレフィクスのネームスペースが定義されるようにする場合に役立ちます。 ● **none** 指定した要素 (この場合、ポイント) とそのサブ要素に対してネームスペースもプレフィクスも設定しません。	namespace パラメーターでは、ネームスペースの出力フォーマット変換を指定します。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
GML	SRSNameFormat	short	● **short** 空間参照系名に短いフォーマット (たとえば、EPSG:4326) を使用します。 ● **long** 空間参照系名に長いフォーマット (たとえば、urn:x-ogc:def:crs:EPSG:4326) を使用します。 ● **none** ジオメトリに空間参照系名の属性を含めません。	SRSNameFormat パラメーターでは、srsName 属性のフォーマットを指定します。
GML	SRSDimension	No	「Yes」 または 「No」	SRSDimension パラメーターでは、指定したジオメトリの座標値の数を指定します。これは GML(version=3) にのみ適用されます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
GML	SRSFillAll	No	「Yes」 または 「No」	SRSFillAll パラメーターでは、SRS 属性を子ジオメトリ要素に伝達するかどうかを指定します。例を挙げると、複数ジオメトリまたは複数多角形では属性を子ジオメトリに伝達します。
GML	UseDeprecated	No	「Yes」 または 「No」	UseDeprecated パラメーターは GML(version=3) にのみ適用されます。このパラメーターは、可能な場合は古い GML 表現を出力するために使用します。例を挙げると、ジオメトリに円ストリングが含まれていない場合、面を多角形として出力できます。
GML	Attribute	自動的に生成されるオプション属性	最上位レベルのジオメトリ要素に対してのみ、1 つ以上の属性を指定できます。	任意の有効な XML 属性を指定できます。
GML	SubElement	自動的に生成される GML サブ要素	最上位レベルのジオメトリ要素に対してのみ、1 つ以上のサブ要素を指定できます。	任意の有効な XML 要素を指定できます。

注意
ST_AsGML では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.39

例

次の例では、結果として <Point srsName="EPSG:4326"><pos>1 2 3 4</pos></Point> を返します。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsGML()
```

次の例では、結果として <Point srsName="EPSG:4326"><coordinates>1,2</coordinates></Point> を返します。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsGML('GML(Version=2)')
```

次の例では、結果として <gml:Point srsName="EPSG:4326"><gml:coordinates>1,2</gml:coordinates></gml:Point> を返します。Namespace=global パラメーターでは、指定した要素とそのサブ要素に対して専用の (gml) プレフィクスを設定します。これは、集約操作の中でクエリを使用し、ある最上位レベルの要素で "gml" プレフィクスのネームスペースが定義されるようにする場合に役立ちます。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsGML('GML(Version=2;Namespace=global)')
```

次の例では、結果として <Point srsName="EPSG:4326"><coordinates>1,2</coordinates></Point> を返します。出力にネームスペース情報は含まれません。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsGML('GML(Version=2;Namespace=none)')
```

次の例では、結果として <Point srsName="http://www.opengis.net/gml/srs/epsg.xml#4326"><coordinates>1,2</coordinates></Point> を返します。srsName 属性の long フォーマットが使用されています。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0,
4326 ).ST_AsGML('GML(Version=2;Namespace=none;SRSNameFormat=long)')
```

次の例では、結果として <Point srsName="urn:x-ogc:def:crs:EPSG:4326"><pos>1 2 3 4</pos></Point> を返します。srsName 属性の long フォーマットが使用され、バージョン 2 フォーマットとは異なるバージョン 3 フォーマットが使用されています。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0,
4326 ).ST_AsGML('GML(Version=3;Namespace=none;SRSNameFormat=long)')
```

ST_AsGeoJSON メソッド

ジオメトリを JSON フォーマットで表す文字列を返します。

構文

geometry-expression.**ST_AsGeoJSON**([*format*])

パラメーター

名前	型	説明
format	VARCHAR(128)	GeoJSON の結果の生成方法を制御するパラメーターを定義する文字列。指定しない場合、デフォルトは 'GeoJSON' です。

戻り値

- **LONG VARCHAR** *geometry-expression* の GeoJSON 表現を返します。

備考

GeoJSON 標準では、JavaScript Object Notation (JSON) に基づく地理空間交換フォーマットが定義されています。このフォーマットは Web ベースのアプリケーションに適しており、WKT や WKB よりも簡潔で解釈しやすくなっています。http://geojson.org/geojson-spec.html を参照してください。

ST_AsGeoJSON メソッドは、ジオメトリを表すテキスト文字列を返します。さまざまなテキストフォーマットが (関連付けられているオプションとともに) サポートされており、オプションの *format* パラメーターを使用して目的のフォーマットを選択します。*format* パラメーターを指定しない場合、デフォルトは 'GeoJSON' です。

フォーマット文字列では、出力フォーマットとそのフォーマットに対するパラメーターを定義します。フォーマット文字列のフォーマットは次のいずれかです。

```
format-name
format-name(parameter1=value1;parameter2=value2;...)
parameter1=value1;parameter2=value2;...
```

最初のフォーマットでは、フォーマット名を指定し、パラメーターは指定しません。すべてのフォーマットパラメーターでデフォルト値が使用されます。2 番目のフォーマットでは、フォーマット名と名前付きパラメーター値のリストを指定します。指定しないパラメーターでは、デフォルト値が使用されます。最後のフォーマットでは、パラメーター値のみを指定し、フォーマット名にはデフォルトの 'GeoJSON' を使用します。

次のフォーマット名を使用できます。

- **GeoJSON** GeoJSON フォーマットでは、http://geojson.org/geojson-spec.html で定義されている JavaScript Object Notation (JSON) が使用されます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
GeoJSON	Version	1.0	1.0	準拠する GeoJSON 仕様のバージョン。現時点でサポートされているのは 1.0 のみです。

注意
ST_AsGeoJSON では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として {"type":"Point", "coordinates":[1,2]} を返します。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsGeoJSON()
```

ST_AsKML メソッド

ST_Geometry 値の KML 表現を返します。

構文

geometry-expression.**ST_AsKML**([*format*])

パラメーター

名前	型	説明
format	VARCHAR(128)	*geometry-expression* を KML 表現に変換するときに使用するパラメーターを定義する文字列。指定しない場合、デフォルトは 'KML' です。

戻り値

- **LONG VARCHAR** *geometry-expression* の KML 表現を返します。

備考

ST_AsKML メソッドは、ジオメトリを表す KML 文字列を返します。さまざまなフォーマットが (関連付けられているオプションとともに) サポートされており、オプションの *format* パラメーターを使用して目的のフォーマットを選択します。*format* パラメーターを指定しない場合、デフォルトは 'KML' です。

フォーマット文字列では、出力フォーマットとそのフォーマットに対するパラメーターを定義します。フォーマット文字列のフォーマットは次のいずれかです。

```
format-name
format-name(parameter1=value1;parameter2=value2;...)
parameter1=value1;parameter2=value2;...
```

最初のフォーマットでは、フォーマット名を指定し、パラメーターは指定しません。すべてのフォーマットパラメーターでデフォルト値が使用されます。2 番目のフォーマットでは、フォーマット名と名前付きパラメーター値のリストを指定します。指定しないパラメーターでは、デフォルト値が使用されます。最後のフォーマットでは、パラメーター値のみを指定し、フォーマット名にはデフォルトの 'KML' を使用します。

次のフォーマット名を使用できます。

- **KML** OGC で定義された Keyhole Markup Language フォーマット。

次のフォーマットパラメーターを指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
KML	Version	2	2	KML バージョン 2.2 がサポートされています。
KML	Attribute	自動的に生成されるオプション属性	最上位レベルのジオメトリ要素に対してのみ、1 つ以上の属性を指定できます。	任意の有効な XML 属性を指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
KML	Namespace	none	● **local** 指定したジオメトリ要素 (この場合、ポイント) とそのサブ要素に対してデフォルトのネームスペース属性「http://www.opengis.net/kml/2.2」を設定します。 ● **global** 指定した要素とそのサブ要素に対して専用の (kml) プレフィクスを設定します。これは、集約操作の中でクエリを使用し、ある最上位レベルの要素で "kml" プレフィクスのネームスペースが定義されるようにする場合に役立ちます。 ● **none** 指定した要素 (この場合、ポイント) とそのサブ要素に対してネームスペース	namespace パラメーターでは、ネームスペースの出力フォーマット変換を指定します。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
			もプレフィクスも設定しません。	
KML	SubElement	自動的に生成される KML サブ要素	最上位レベルのジオメトリ要素に対してのみ、1 つ以上のサブ要素を指定できます。	任意の有効な XML 要素を指定できます。例を挙げると、extrude、tessellate、altitudeMode 要素を指定できます。

注意
ST_AsKML では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.39

例

次の例では、結果として <Point><coordinates>1,2,3,4</coordinates></Point> を返します。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsKML()
```

次の例では、結果として <Point><coordinates>1,2,3,4</coordinates></Point> を返します。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsKML('KML(Version=2)')
```

次の例では、結果として <kml:Point><kml:coordinates>1,2,3,4</kml:coordinates></kml:Point> を返します。Namespace=global パラメーターでは、指定した要素とそのサブ要素に対して専用の (kml) プレフィクスを設定します。これは、集約操作の中でクエリを使用し、ある最上位レベルの要素で "kml" プレフィクスのネームスペースが定義されるようにする場合に役立ちます。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsKML('KML(Version=2;Namespace=global)')
```

次の例では、結果として <Point><coordinates>1,2,3,4</coordinates></Point> を返します。出力にネームスペース情報は含まれません。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsKML('KML(Version=2;Namespace=none)')
```

次の例では、結果として <Point xmlns="http://www.opengis.net/kml/2.2"><coordinates>1,2,3,4</coordinates></Point> を返します。デフォルトの xml ネームスペースが使用されています。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsKML('KML(Version=2;Namespace=default)')
```


次の例では、結果として <Point><altitudeMode>absolute</altitudeMode><coordinates>1,2,3,4</coordinates></Point> を返します。出力に AltitudeMode サブ要素が含まれています。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsKML('SubElement=<altitudeMode>absolute</altitudeMode>')
```

ST_AsSVG メソッド

ジオメトリ値を表す SVG 図形を返します。

構文

geometry-expression.**ST_AsSVG**([*format*])

パラメーター

名前	型	説明
format	VARCHAR(128)	*geometry-expression* を SVG 表現に変換するときに使用するパラメーターを定義する文字列。指定しない場合、デフォルトは 'SVG' です。

戻り値

- **LONG VARCHAR**　*geometry-expression* をレンダリングした完全または部分的な SVG ドキュメントを返します。

備考

ST_AsSVG メソッドは、SVG ビューアーを使用してジオメトリをグラフィカルに表示するために使用できる完全または部分的な SVG ドキュメントを返します。Microsoft Internet Explorer を除く主要な Web ブラウザのほとんどに、組み込みの SVG ビューアーが備えられています。

さまざまなオプションがサポートされており、オプションの *format* パラメーターを使用して目的のフォーマットを選択します。*format* パラメーターを指定しない場合、デフォルトは 'SVG' です。

フォーマット文字列では、出力フォーマットとそのフォーマットに対するパラメーターを定義します。フォーマット文字列のフォーマットは次のいずれかです。

```
format-name
format-name(parameter1=value1;parameter2=value2;...)
parameter1=value1;parameter2=value2;...
```

最初のフォーマットでは、フォーマット名を指定し、パラメーターは指定しません。すべてのフォーマットパラメーターでデフォルト値が使用されます。2 番目のフォーマットでは、フォーマット名と名前付きパラメーター値のリストを指定します。指定しないパラメーターでは、デ

フォルト値が使用されます。最後のフォーマットでは、パラメーター値のみを指定し、フォーマット名にはデフォルトの 'SVG' を使用します。

次のフォーマット名を使用できます。

- **SVG** World Wide Web Consortium (W3C) で定義された Scalable Vector Graphics (SVG) 1.1 フォーマット。

次のフォーマットパラメーターを指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	Approximate	Yes	「Yes」 または 「No」	Approximate パラメーターでは、表示する詳細を若干減らして、出力 SVG ドキュメントのサイズを縮小するかどうかを指定します。SVG データは、最後のポイントの線の幅内にあるポイントを除外することで近似されます。複数のメガバイトジオメトリが存在する場合、これにより圧縮率が 80 % 以上になることがあります。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	Attribute	自動的に生成されるオプション属性	SVG シェイプ要素に適用できる 1 つ以上の SVG 属性	デフォルトでは、fill、stroke、stroke-width などのオプションの SVG シェイプ属性が生成されます。Attributes パラメーターを指定すると、オプションの SVG シェイプ属性は生成されず、代わりに Attribute 値が使用されます。PathDataOnly=Yes を指定している場合は無視されます。Attribute 値の最大長は 1000 バイトです。
SVG	DecimalDigits	空間参照系の「グリッドにスナップ」のグリッドサイズの小数点以下の桁数に基づく (デフォルトの最大値は 5、最小値は 0)	integer	DecimalDigits パラメーターでは、SVG 出力に生成される座標の小数点以下の桁数を制限します。負の桁数を指定すると、SVG 出力での座標の精度が完全なものになります。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	PathDataOnly	No (完全な SVG ドキュメントが生成される)	「Yes」 または 「No」	PathDataOnly パラメーターでは、SVG パス要素のデータのみを生成するかどうかを指定します。後述の PathDataOnly の例は、PathDataOnly=Yes を使用して、表示可能な完全な SVG ドキュメントを作成する方法を示します。デフォルトでは、完全な SVG ドキュメントが生成されます。PathDataOnly=Yes によって返されるパスデータを使用すると、テキストなどの他の要素を含む、より柔軟な SVG ドキュメントを作成できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	RandomFill	Yes	「Yes」 または 「No」	RandomFill パラメーターでは、ランダムに生成される色で多角形を塗りつぶすかどうかを指定します。使用される色の順序は、明確に定義された順序には従わず、通常、SVG 出力を生成するたびに変わります。「No」は、各多角形のアウトラインのみを描画することを示します。Attribute または PathDataOnly=Yes パラメーターを指定している場合、RandomFill パラメーターは無視されます。
SVG	Relative	Yes	「Yes」 または 「No」	Relative パラメーターでは、座標を相対 (オフセット) フォーマットで出力するか、絶対フォーマットで出力するかを指定します。相対座標データは、一般に絶対座標データよりもコンパクトになります。

注意
ST_AsSVG では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_AsSVGAggr メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、ランダムな色で塗りつぶされた多角形を含む完全な SVG ドキュメントを返します。

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 0 20, 60 10, 0 0 ))' )
        .ST_AsSVG()
```

次の例では、塗りつぶしなしの多角形を含む完全な SVG ドキュメントを返し、座標の小数点以下の桁数を 3 桁に制限します。

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 0 20, 60 10, 0 0 ))' )
        .ST_AsSVG( 'RandomFill=No;DecimalDigits=3' )
```

次の例では、青で塗りつぶされた多角形を含み、最大精度の座標を使用した完全な SVG ドキュメントを返します。

```
SELECT Shape.ST_AsSVG( 'Attribute=fill="blue";DecimalDigits=-1' )
FROM SpatialShapes
```

次の例では、小数点以下の桁数が 5 桁に制限された相対座標の SVG パスデータから完全な SVG ドキュメントを返します。

```
SELECT '<?xml version="1.0" standalone="yes"?>
   <!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>'
```

次の例では、小数点以下の桁数が 7 桁に制限された絶対座標を使用して SVG パスデータを返します。

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 0 20, 60 10, 0 0 ))' )
        .ST_AsSVG( 'PathDataOnly=Yes;Relative=No;DecimalDigits=7' )
```


ST_AsSVGAggr メソッド

グループ内のジオメトリをレンダリングした完全または部分的な SVG ドキュメントを返します。

構文

ST_Geometry::**ST_AsSVGAggr**(*geometry-column*[**ORDER BY** *order-by-expression* [**ASC** | **DESC**], ...] [, *format*])

パラメーター

名前	型	説明
geometry-column	ST_Geometry	SVG 図形に影響を与えるジオメトリ値。通常、これはカラムです。
format	VARCHAR(128)	各ジオメトリ値を SVG 表現に変換するときに使用するパラメーターを定義する文字列。指定しない場合、デフォルトは 'SVG' です。

戻り値

- **LONG VARCHAR**　グループ内のジオメトリをレンダリングした完全または部分的な SVG ドキュメントを返します。

備考

ST_AsSVGAggr メソッドは、SVG ビューアーを使用してジオメトリのグループの論理和をグラフィカルに表示するために使用できる完全または部分的な SVG ドキュメントを返します。Microsoft Internet Explorer を除く主要な Web ブラウザのほとんどに、組み込みの SVG ビューアーが備えられています。

さまざまなオプションがサポートされており、オプションの *format* パラメーターを使用して目的のフォーマットを選択します。*format* パラメーターを指定しない場合、デフォルトは 'SVG' です。

フォーマット文字列では、出力フォーマットとそのフォーマットに対するパラメーターを定義します。フォーマット文字列のフォーマットは次のいずれかです。

```
format-name
format-name(parameter1=value1;parameter2=value2;...)
parameter1=value1;parameter2=value2;...
```

最初のフォーマットでは、フォーマット名を指定し、パラメーターは指定しません。すべてのフォーマットパラメーターでデフォルト値が使用されます。2 番目のフォーマットでは、フォーマット名と名前付きパラメーター値のリストを指定します。指定しないパラメーターでは、デ

フォルト値が使用されます。最後のフォーマットでは、パラメーター値のみを指定し、フォーマット名にはデフォルトの 'SVG' を使用します。

次のフォーマット名を使用できます。

- **SVG** World Wide Web Consortium (W3C) で定義された Scalable Vector Graphics (SVG) 1.1 フォーマット。

次のフォーマットパラメーターを指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	Approximate	Yes	「Yes」 または 「No」	Approximate パラメーターでは、表示する詳細を若干減らして、出力 SVG ドキュメントのサイズを縮小するかどうかを指定します。SVG データは、最後のポイントの線の幅内にあるポイントを除外することで近似されます。複数のメガバイトジオメトリが存在する場合、これにより圧縮率が 80 % 以上になることがあります。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	Attribute	自動的に生成されるオプション属性	SVG シェイプ要素に適用できる 1 つ以上の SVG 属性	デフォルトでは、fill、stroke、stroke-width などのオプションの SVG シェイプ属性が生成されます。Attributes パラメーターを指定すると、オプションの SVG シェイプ属性は生成されず、代わりに Attribute 値が使用されます。PathDataOnly=Yes を指定している場合は無視されます。
SVG	DecimalDigits	空間参照系の「グリッドにスナップ」のグリッドサイズの小数点以下の桁数に基づく (デフォルトの最大値は 5、最小値は 0)	整数	DecimalDigits パラメーターでは、SVG 出力に生成される座標の小数点以下の桁数を制限します。負の桁数を指定すると、SVG 出力での座標の精度が完全なものになります。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	PathDataOnly	No (完全な SVG ドキュメントが生成される)	「Yes」 または 「No」	PathDataOnly パラメーターでは、SVG パス要素のデータのみを生成するかどうかを指定します。後述の PathDataOnly の例は、PathDataOnly=Yes を使用して、表示可能な完全な SVG ドキュメントを作成する方法を示します。デフォルトでは、完全な SVG ドキュメントが生成されます。PathDataOnly=Yes によって返されるパスデータを使用すると、テキストなどの他の要素を含む、より柔軟な SVG ドキュメントを作成できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	RandomFill	Yes	「Yes」 または 「No」	RandomFill パラメーターでは、ランダムに生成される色で多角形を塗りつぶすかどうかを指定します。使用される色の順序は、明確に定義された順序には従わず、通常、SVG 出力を生成するたびに変わります。「No」は、各多角形のアウトラインのみを描画することを示します。Attribute または PathDataOnly=Yes パラメーターを指定している場合、RandomFill パラメーターは無視されます。
SVG	Relative	Yes	「Yes」 または 「No」	Relative パラメーターでは、座標を相対 (オフセット) フォーマットで出力するか、絶対フォーマットで出力するかを指定します。相対座標データは、一般に絶対座標データよりもコンパクトになります。

ORDER BY 句を指定して、降順 (後ろから前へ) に表示されるジオメトリと重複するジオメトリの表示方法を制御できます。指定しない場合、ジオメトリの表示順序は、クエリオプティマイザーによって選択された実行プランによって決まります。この順序は、実行間で異なる場合があります。

注意
ST_AsSVGAggr では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_AsSVG メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、ランダムな色で塗りつぶされた多角形を含む完全な SVG ドキュメントを返します。

```
SELECT ST_Geometry::ST_AsSVGAggr( Shape ) FROM SpatialShapes
```

次の例では、小数点以下の桁数が 5 桁に制限された相対座標の SVG パスデータから完全な SVG ドキュメントを返します。

```
SELECT '<?xml version="1.0" standalone="yes"?>
   <!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
```

次の文では、SpatialShapes テーブルのすべてのジオメトリがレンダリングされた完全な SVG ドキュメントを返す Web サービスを作成します。-xs http オプションを指定してデータベースがを起動した場合は、SVG をサポートするブラウザを使用して SVG を表示できます。表示するには、アドレス http://localhost/demo/svg_shapes にアクセスします。この操作は、ブラウザとデータベースサーバーが同じコンピューターにインストールされており、データベースの名前が demo であることを想定しています。

```
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 メソッド

ST_Geometry 値のテキスト表現を返します。

構文

geometry-expression.**ST_AsText**([*format*])

パラメーター

名前	型	説明
format	VARCHAR(128)	*geometry-expression* をテキスト表現に変換するときに使用する出力テキストフォーマットを定義する文字列。指定しない場合は、st_geometry_astext_format オプションを使用してテキスト表現を選択します。「st_geometry_astext_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

戻り値

- **LONG VARCHAR**　*geometry-expression* のテキスト表現を返します。

備考

ST_AsText メソッドは、ジオメトリを表すテキスト文字列を返します。さまざまなテキストフォーマットが (関連付けられているオプションとともに) サポートされており、オプションの *format* パラメーターを使用して目的のフォーマットを選択します。*format* パラメーターを指定しない場合は、st_geometry_astext_format オプションを使用して、使用する出力フォーマットを選択します。「st_geometry_astext_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

フォーマット文字列では、出力フォーマットとそのフォーマットに対するパラメーターを定義します。フォーマット文字列のフォーマットは次のいずれかです。

```
format-name
format-name(parameter1=value1;parameter2=value2;...)
parameter1=value1;parameter2=value2;...
```

最初のフォーマットでは、フォーマット名を指定し、パラメーターは指定しません。すべてのフォーマットパラメーターでデフォルト値が使用されます。2 番目のフォーマットでは、フォーマット名と名前付きパラメーター値のリストを指定します。指定しないパラメーターでは、デフォルト値が使用されます。最後のフォーマットでは、パラメーター値のみを指定し、フォーマット名にはデフォルトの 'WKT' を使用します。

次のフォーマット名を使用できます。

- **WKT**　SQL/MM と OGC で定義された Well-Known-Text フォーマット。

- **EWKT** Extended-Well-Known-Text フォーマット。このフォーマットには、プレフィクスとしてジオメトリの SRID が含まれます。
- **GML** ISO 19136 と OGC で定義された Geography Markup Language フォーマット。
- **KML** OGC で定義された Keyhole Markup Language フォーマット。
- **GeoJSON** GeoJSON フォーマットでは、http://geojson.org/geojson-spec.html で定義されている JavaScript Object Notation (JSON) が使用されます。
- **SVG** World Wide Web Consortium (W3C) で定義された Scalable Vector Graphics (SVG) 1.1 フォーマット。

次のフォーマットパラメーターを指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
WKT	Version	1.2	● **1.1** OGC SFS 1.1 で定義された WKT。このフォーマットには、Z 値と M 値は含まれません。ジオメトリに Z 値または M 値が含まれている場合、それらの値は出力では削除されます。 ● **1.2** OGC SFS 1.2 で定義された WKT。これは、バージョン 1.1 の 2D データに適合し、Z 値と M 値をサポートするようにフォーマットを拡張します。 ● **PostGIS** いくつかの他のベンダーが使用している WKT フォーマット。OGC 1.2 とは異なる方法で Z 値と M 値が含まれています。	version パラメーターでは、使用される WKT 仕様のバージョンを制御します。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
GML	Version	3	● **2** GML 仕様のバージョン 2。 ● **3** GML 仕様のバージョン 3.2。	version パラメーターでは、使用される GML 仕様のバージョンを制御します。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
GML	Namespace	none	● **local** 指定した要素 (この場合、ポイント) とそのサブ要素に対してデフォルトのネームスペース属性を設定します。 ● **global** 指定した要素とそのサブ要素に対して専用の (gml) プレフィクスを設定します。これは、集約操作の中でクエリを使用し、ある最上位レベルの要素で "gml" プレフィクスのネームスペースが定義されるようにする場合に役立ちます。 ● **none** 指定した要素 (この場合、ポイント) とそのサブ要素に対してネームスペースもプレフィクスも設定しません。	namespace パラメーターでは、ネームスペースの出力フォーマット変換を指定します。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
GML	SRSNameFormat	short	● **short** 空間参照系名に短いフォーマット (たとえば、EPSG:4326) を使用します。 ● **long** 空間参照系名に長いフォーマット (たとえば、urn:x-ogc:def:crs:EPSG:4326) を使用します。 ● **none** ジオメトリに空間参照系名の属性を含めません。	SRSNameFormat パラメーターでは、srsName 属性のフォーマットを指定します。
GML	SRSDimension	No	「Yes」 または 「No」	SRSDimension パラメーターでは、指定したジオメトリの座標値の数を指定します。これは GML(version=3) にのみ適用されます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
GML	SRSFillAll	No	「Yes」 または「No」	SRSFillAll パラメーターでは、SRS 属性を子ジオメトリ要素に伝達するかどうかを指定します。例を挙げると、複数ジオメトリまたは複数多角形では属性を子ジオメトリに伝達します。
GML	UseDeprecated	No	「Yes」 または「No」	UseDeprecated パラメーターは GML(version=3) にのみ適用されます。このパラメーターは、可能な場合は古い GML 表現を出力するために使用します。例を挙げると、ジオメトリに円ストリングが含まれていない場合、面を多角形として出力できます。
GML	Attribute	自動的に生成されるオプション属性	最上位レベルのジオメトリ要素に対してのみ、1 つ以上の属性を指定できます。	任意の有効な XML 属性を指定できます。
GML	SubElement	自動的に生成される GML サブ要素	最上位レベルのジオメトリ要素に対してのみ、1 つ以上のサブ要素を指定できます。	任意の有効な XML 要素を指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
KML	Version	2	2	KML バージョン 2.2 がサポートされています。
KML	Attribute	自動的に生成されるオプション属性	最上位レベルのジオメトリ要素に対してのみ、1 つ以上の属性を指定できます。	任意の有効な XML 属性を指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
KML	Namespace	none	● **local** 指定したジオメトリ要素 (この場合、ポイント) とそのサブ要素に対してデフォルトのネームスペース属性「http://www.opengis.net/kml/2.2」を設定します。 ● **global** 指定した要素とそのサブ要素に対して専用の (kml) プレフィクスを設定します。これは、集約操作の中でクエリを使用し、ある最上位レベルの要素で "kml" プレフィクスのネームスペースが定義されるようにする場合に役立ちます。 ● **none** 指定した要素 (この場合、ポイント) とそのサブ要素に対してネームスペース	namespace パラメーターでは、ネームスペースの出力フォーマット変換を指定します。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
			もプレフィクスも設定しません。	
KML	SubElement	自動的に生成される KML サブ要素	最上位レベルのジオメトリ要素に対してのみ、1 つ以上のサブ要素を指定できます。	任意の有効な XML 要素を指定できます。例を挙げると、extrude、tessellate、altitudeMode 要素を指定できます。
GeoJSON	Version	1	1	準拠する GeoJSON 仕様のバージョン。現時点でサポートされているのは 1.0 のみです。
SVG	Approximate	Yes	「Yes」 または 「No」	Approximate パラメーターでは、表示する詳細を若干減らして、出力 SVG ドキュメントのサイズを縮小するかどうかを指定します。SVG データは、最後のポイントの線の幅内にあるポイントを除外することで近似されます。複数のメガバイトジオメトリが存在する場合、これにより圧縮率が 80 % 以上になることがあります。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	Attribute	自動的に生成されるオプション属性	SVG シェイプ要素に適用できる 1 つ以上の SVG 属性	デフォルトでは、fill、stroke、stroke-width などのオプションの SVG シェイプ属性が生成されます。Attributes パラメーターを指定すると、オプションの SVG シェイプ属性は生成されず、代わりに Attribute 値が使用されます。PathDataOnly=Yes を指定している場合は無視されます。Attribute 値の最大長は 1000 バイトです。
SVG	DecimalDigits	空間参照系の「グリッドにスナップ」のグリッドサイズの小数点以下の桁数に基づく (デフォルトの最大値は 5、最小値は 0)	整数	DecimalDigits パラメーターでは、SVG 出力に生成される座標の小数点以下の桁数を制限します。負の桁数を指定すると、SVG 出力での座標の精度が完全なものになります。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	PathDataOnly	No (完全な SVG ドキュメントが生成される)	「Yes」 または 「No」	PathDataOnly パラメーターでは、SVG パス要素のデータのみを生成するかどうかを指定します。後述の PathDataOnly の例は、PathDataOnly=Yes を使用して、表示可能な完全な SVG ドキュメントを作成する方法を示します。デフォルトでは、完全な SVG ドキュメントが生成されます。PathDataOnly=Yes によって返されるパスデータを使用すると、テキストなどの他の要素を含む、より柔軟な SVG ドキュメントを作成できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	RandomFill	Yes	「Yes」 または 「No」	RandomFill パラメーターでは、ランダムに生成される色で多角形を塗りつぶすかどうかを指定します。使用される色の順序は、明確に定義された順序には従わず、通常、SVG 出力を生成するたびに変わります。「No」は、各多角形のアウトラインのみを描画することを示します。Attribute または PathDataOnly=Yes パラメーターを指定している場合、RandomFill パラメーターは無視されます。
SVG	Relative	Yes	「Yes」 または 「No」	Relative パラメーターでは、座標を相対 (オフセット) フォーマットで出力するか、絶対フォーマットで出力するかを指定します。相対座標データは、一般に絶対座標データよりもコンパクトになります。

注意
サーバーでは、ジオメトリ値を VARCHAR または NVARCHAR に変換するときに ST_AsText メソッドが使用されます。st_geometry_astext_format オプションでは、変換に使用するフォーマットを定義します。「st_geometry_astext_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

注意
ST_AsText では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_AsGeoJSON メソッド
- ST_Geometry タイプの ST_AsGML メソッド
- ST_Geometry タイプの ST_AsKML メソッド
- ST_Geometry タイプの ST_AsSVG メソッド
- ST_Geometry タイプの ST_AsWKT メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.35

例

st_geometry_astext_format オプションの値が 'WKT' と仮定した場合、次の例では、結果として Point ZM (1 2 3 4) を返します。「st_geometry_astext_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsText()
```

st_geometry_astext_format オプションの値が 'WKT' と仮定した場合、次の例では、結果として Point ZM (1 2 3 4) を返します。ジオメトリを VARCHAR または NVARCHAR タイプに変換するときに ST_AsText メソッドが暗黙的に呼び出されます。「st_geometry_astext_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

```
SELECT CAST( NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ) as long varchar)
```

次の例では、結果として Point (1 2) を返します。Z 値と M 値は、WKT の OGC 仕様のバージョン 1.1.0 ではサポートされていないため、出力されません。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsText('WKT(Version=1.1)')
```

次の例では、結果として SRID=4326;Point ZM (1 2 3 4) を返します。結果には、プレフィクスとして SRID が含まれています。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsText('EWKT')
```

次の例では、結果として <Point srsName="EPSG:4326"><pos>1 2 3 4</pos></Point> を返します。


```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsText('GML')
```

次の例では、'{"type":"Point", "coordinates":[1,2]} ' を返します。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsText('GeoJSON')
```

次の例では、ランダムな色で塗りつぶされた多角形を含む完全な SVG ドキュメントを返します。

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 0 20, 60 10, 0 0 ))' )
        .ST_AsText( 'SVG' )
```

ST_AsWKB メソッド

ST_Geometry 値の WKB 表現を返します。

構文

geometry-expression.**ST_AsWKB**([*format*])

パラメーター

名前	型	説明
format	VARCHAR(128)	*geometry-expression* をバイナリに変換するときに使用する WKB フォーマットを定義する文字列。指定しない場合、デフォルトは 'WKB' です。

戻り値

- **LONG BINARY**　*geometry-expression* の WKB 表現を返します。

備考

ST_AsWKB メソッドは、ジオメトリを WKB フォーマットで表すバイナリ文字列を返します。さまざまなフォーマットが (関連付けられているオプションとともに) サポートされており、オプションの *format* パラメーターを使用して目的のフォーマットを選択します。*format* パラメーターを指定しない場合、デフォルトは 'WKB' です。

フォーマット文字列では、出力フォーマットとそのフォーマットに対するパラメーターを定義します。フォーマット文字列のフォーマットは次のいずれかです。

```
format-name
format-name(parameter1=value1;parameter2=value2;...)
parameter1=value1;parameter2=value2;...
```

最初のフォーマットでは、フォーマット名を指定し、パラメーターは指定しません。すべてのフォーマットパラメーターでデフォルト値が使用されます。2 番目のフォーマットでは、フォーマット名と名前付きパラメーター値のリストを指定します。指定しないパラメーターでは、デ

フォルト値が使用されます。最後のフォーマットでは、パラメーター値のみを指定し、フォーマット名にはデフォルトの 'WKB' を使用します。

次のフォーマット名を使用できます。

- **WKB** SQL/MM と OGC で定義された Well-Known-Binary フォーマット
- **EWKB** PostGIS で定義された Extended-Well-Known-Binary フォーマット。このフォーマットには、ジオメトリの SRID が含まれます。また、Z 値と M 値の表現方法が WKB と異なります。

次のフォーマットパラメーターを指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
WKB	Version	1.2	● **1.1** OGC SFS 1.1 で定義された WKB。このフォーマットには、Z 値と M 値は含まれません。ジオメトリに Z 値または M 値が含まれている場合、それらの値は出力では削除されます。 ● **1.2** OGC SFS 1.2 で定義された WKB。これは、バージョン 1.1 の 2D データに適合し、Z 値と M 値をサポートするようにフォーマットを拡張します。	version パラメーターでは、使用される WKB 仕様のバージョンを制御します。

注意
ST_AsWKB では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として 0x01b90b0000000000000000f03f0000000000000040000000000000084000000000000001040 を返します。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsWKB('endian=little')
```

次の例では、結果として 0x0101000000000000000000f03f0000000000000040 を返します。WKB の OGC 仕様のバージョン 1.1 では Z 値と M 値はサポートされていないため、これらの値は省略されます。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsWKB('WKB(Version=1.1;endian=little)')
```

次の例では、結果として 0x01010000e0e610000000000000000f03f0000000000000040000000000000084000000000000001040 を返します。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsWKB('EWKB(endian=little)')
```

ST_AsWKT メソッド

ST_Geometry 値の WKT 表現を返します。

構文

geometry-expression.**ST_AsWKT**([*format*])

パラメーター

名前	型	説明
format	VARCHAR(128)	*geometry-expression* を WKT に変換するときに使用する出力テキストフォーマットを定義する文字列。指定しない場合、フォーマット文字列のデフォルトは 'WKT' です。

戻り値

- **LONG VARCHAR**　*geometry-expression* の WKT 表現を返します。

備考

ST_AsWKT メソッドは、ジオメトリを表すテキスト文字列を返します。さまざまなテキストフォーマットが (関連付けられているオプションとともに) サポートされており、オプションの *format* パラメーターを使用して目的のフォーマットを選択します。

フォーマット文字列では、出力フォーマットとそのフォーマットに対するパラメーターを定義します。フォーマット文字列のフォーマットは次のいずれかです。

```
format-name
format-name(parameter1=value1;parameter2=value2;...)
parameter1=value1;parameter2=value2;...
```

最初のフォーマットでは、フォーマット名を指定し、パラメーターは指定しません。すべてのフォーマットパラメーターでデフォルト値が使用されます。2 番目のフォーマットでは、フォーマット名と名前付きパラメーター値のリストを指定します。指定しないパラメーターでは、デフォルト値が使用されます。最後のフォーマットでは、パラメーター値のみを指定し、フォーマット名にはデフォルトの 'WKT' を使用します。

次のフォーマット名を使用できます。

- **WKT**　SQL/MM と OGC で定義された Well-Known-Text フォーマット。
- **EWKT**　PostGIS で定義された Extended-Well-Known-Text フォーマット。このフォーマットには、ジオメトリの SRID が含まれます。また、Z 値と M 値の表現方法が WKT と異なります。

次のフォーマットパラメーターを指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
WKT	Version	1.2	● **1.1** OGC SFS 1.1 で定義された WKT。このフォーマットには、Z 値と M 値は含まれません。ジオメトリに Z 値または M 値が含まれている場合、それらの値は出力では削除されます。 ● **1.2** OGC SFS 1.2 で定義された WKT。これは、バージョン 1.1 の 2D データに適合し、Z 値と M 値をサポートするようにフォーマットを拡張します。 ● **PostGIS** いくつかの他のベンダーが使用している WKT フォーマット。OGC 1.2 とは異なる方法で Z 値と M 値が含まれています。	version パラメーターでは、使用される WKT 仕様のバージョンを制御します。

注意
ST_AsWKT では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として SRID=0;Polygon ((3 3, 8 3, 4 8, 3 3)) を返します。

```
SELECT Shape.ST_AsWKT( 'EWKT' ) FROM SpatialShapes WHERE ShapeID = 22
```

ST_AsXML メソッド

ST_Geometry 値の XML 表現を返します。

構文

geometry-expression.**ST_AsXML**([*format*])

パラメーター

名前	型	説明
format	VARCHAR(128)	*geometry-expression* を XML 表現に変換するときに使用する出力テキストフォーマットを定義する文字列。指定しない場合は、st_geometry_asxml_format オプションを使用して XML 表現を選択します。「st_geometry_asxml_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

戻り値

- **LONG VARCHAR** *geometry-expression* の XML 表現を返します。

備考

ST_AsXML メソッドは、ジオメトリを表す XML 文字列を返します。サポートされている XML フォーマットは、GML、KML、SVG です。*format* パラメーターでは、XML への変換を制御するパラメーターを指定します。*format* を指定しない場合は、st_geometry_asxml_format オプション

の値を使用して、出力フォーマットを選択します。「st_geometry_asxml_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

フォーマット文字列では、出力フォーマットとそのフォーマットに対するパラメーターを定義します。フォーマット文字列のフォーマットは次のいずれかです。

```
format-name
format-name(parameter1=value1;parameter2=value2;...)
parameter1=value1;parameter2=value2;...
```

最初のフォーマットでは、フォーマット名を指定し、パラメーターは指定しません。すべてのフォーマットパラメーターでデフォルト値が使用されます。2 番目のフォーマットでは、フォーマット名と名前付きパラメーター値のリストを指定します。指定しないパラメーターでは、デフォルト値が使用されます。最後のフォーマットでは、パラメーター値のみを指定し、フォーマット名にはデフォルトの 'GML' を使用します。

次のフォーマット名を使用できます。

- **GML** ISO 19136 と OGC で定義された Geography Markup Language フォーマット。
- **KML** OGC で定義された Keyhole Markup Language フォーマット。
- **SVG** World Wide Web Consortium (W3C) で定義された Scalable Vector Graphics (SVG) 1.1 フォーマット。

次のフォーマットパラメーターを指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
GML	Version	3	● **2** GML 仕様のバージョン 2。 ● **3** GML 仕様のバージョン 3.2。	version パラメーターでは、使用される GML 仕様のバージョンを制御します。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
GML	Namespace	none	● **local** 指定した要素 (この場合、ポイント) とそのサブ要素に対してデフォルトのネームスペース属性を設定します。 ● **global** 指定した要素とそのサブ要素に対して専用の (gml) プレフィクスを設定します。これは、集約操作の中でクエリを使用し、ある最上位レベルの要素で "gml" プレフィクスのネームスペースが定義されるようにする場合に役立ちます。 ● **none** 指定した要素 (この場合、ポイント) とそのサブ要素に対してネームスペースもプレフィクスも設定しません。	namespace パラメーターでは、ネームスペースの出力フォーマット変換を指定します。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
GML	SRSNameFormat	short	● **short** 空間参照系名に短いフォーマット (たとえば、EPSG:4326) を使用します。 ● **long** 空間参照系名に長いフォーマット (たとえば、urn:x-ogc:def:crs:EPSG:4326) を使用します。 ● **none** ジオメトリに空間参照系名の属性を含めません。	SRSNameFormat パラメーターでは、srsName 属性のフォーマットを指定します。
GML	SRSDimension	No	「Yes」 または 「No」	SRSDimension パラメーターでは、指定したジオメトリの座標値の数を指定します。これは GML(version=3) にのみ適用されます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
GML	SRSFillAll	No	「Yes」 または 「No」	SRSFillAll パラメーターでは、SRS 属性を子ジオメトリ要素に伝達するかどうかを指定します。例を挙げると、複数ジオメトリまたは複数多角形では属性を子ジオメトリに伝達します。
GML	UseDeprecated	No	「Yes」 または 「No」	UseDeprecated パラメーターは GML(version=3) にのみ適用されます。このパラメーターは、可能な場合は古い GML 表現を出力するために使用します。例を挙げると、ジオメトリに円ストリングが含まれていない場合、面を多角形として出力できます。
GML	Attribute	自動的に生成されるオプション属性	最上位レベルのジオメトリ要素に対してのみ、1 つ以上の属性を指定できます。	任意の有効な XML 属性を指定できます。
GML	SubElement	自動的に生成される GML サブ要素	最上位レベルのジオメトリ要素に対してのみ、1 つ以上のサブ要素を指定できます。	任意の有効な XML 要素を指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
KML	Version	2	2	KML バージョン 2.2 がサポートされています。
KML	Attribute	自動的に生成されるオプション属性	最上位レベルのジオメトリ要素に対してのみ、1 つ以上の属性を指定できます。	任意の有効な XML 属性を指定できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
KML	Namespace	none	● **local** 指定したジオメトリ要素 (この場合、ポイント) とそのサブ要素に対してデフォルトのネームスペース属性「http://www.opengis.net/kml/2.2」を設定します。 ● **global** 指定した要素とそのサブ要素に対して専用の (kml) プレフィクスを設定します。これは、集約操作の中でクエリを使用し、ある最上位レベルの要素で "kml" プレフィクスのネームスペースが定義されるようにする場合に役立ちます。 ● **none** 指定した要素 (この場合、ポイント) とそのサブ要素に対してネームスペース	namespace パラメーターでは、ネームスペースの出力フォーマット変換を指定します。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
			もプレフィクスも設定しません。	
KML	SubElement	自動的に生成される KML サブ要素	最上位レベルのジオメトリ要素に対してのみ、1 つ以上のサブ要素を指定できます。	任意の有効な XML 要素を指定できます。例を挙げると、extrude、tessellate、altitudeMode 要素を指定できます。
SVG	Approximate	Yes	「Yes」 または 「No」	Approximate パラメーターでは、表示する詳細を若干減らして、出力 SVG ドキュメントのサイズを縮小するかどうかを指定します。SVG データは、最後のポイントの線の幅内にあるポイントを除外することで近似されます。複数のメガバイトジオメトリが存在する場合、これにより圧縮率が 80 % 以上になることがあります。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	Attribute	自動的に生成されるオプション属性	SVG シェイプ要素に適用できる 1 つ以上の SVG 属性	デフォルトでは、fill、stroke、stroke-width などのオプションの SVG シェイプ属性が生成されます。Attributes パラメーターを指定すると、オプションの SVG シェイプ属性は生成されず、代わりに Attribute 値が使用されます。PathDataOnly=Yes を指定している場合は無視されます。Attribute 値の最大長は 1000 バイトです。
SVG	DecimalDigits	空間参照系の「グリッドにスナップ」のグリッドサイズの小数点以下の桁数に基づく (デフォルトの最大値は 5、最小値は 0)	整数	DecimalDigits パラメーターでは、SVG 出力に生成される座標の小数点以下の桁数を制限します。負の桁数を指定すると、SVG 出力での座標の精度が完全なものになります。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	PathDataOnly	No (完全な SVG ドキュメントが生成される)	「Yes」 または 「No」	PathDataOnly パラメーターでは、SVG パス要素のデータのみを生成するかどうかを指定します。後述の PathDataOnly の例は、PathDataOnly=Yes を使用して、表示可能な完全な SVG ドキュメントを作成する方法を示します。デフォルトでは、完全な SVG ドキュメントが生成されます。PathDataOnly=Yes によって返されるパスデータを使用すると、テキストなどの他の要素を含む、より柔軟な SVG ドキュメントを作成できます。

フォーマット名	パラメーター名	デフォルト値	指定可能な値	説明
SVG	RandomFill	Yes	「Yes」 または 「No」	RandomFill パラメーターでは、ランダムに生成される色で多角形を塗りつぶすかどうかを指定します。使用される色の順序は、明確に定義された順序には従わず、通常、SVG 出力を生成するたびに変わります。「No」は、各多角形のアウトラインのみを描画することを示します。Attribute または PathDataOnly=Yes パラメーターを指定している場合、RandomFill パラメーターは無視されます。
SVG	Relative	Yes	「Yes」 または 「No」	Relative パラメーターでは、座標を相対 (オフセット) フォーマットで出力するか、絶対フォーマットで出力するかを指定します。相対座標データは、一般に絶対座標データよりもコンパクトになります。

注意
サーバーでは、ジオメトリ値を XML に変換するときに ST_AsXML メソッドが使用されます。st_geometry_asxml_format オプションでは、変換に使用するフォーマットを定義します。「st_geometry_asxml_format オプション」『SQL Anywhere サーバー データベース管理』を参照してください。

注意
ST_AsXML では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_AsGML メソッド
- ST_Geometry タイプの ST_AsKML メソッド
- ST_Geometry タイプの ST_AsSVG メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

st_geometry_asxml_format オプションでデフォルト値の 'GML' が使用される場合、次の例では結果として <Point srsName="EPSG:4326"><pos>1 2 3 4</pos></Point> を返します。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsXML()
```

st_geometry_asxml_format オプションでデフォルト値の 'GML' が使用される場合、次の例では結果として <Point srsName="EPSG:4326"><pos>1 2 3 4</pos></Point> を返します。

```
SELECT CAST( NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ) AS XML)
```

次の例では、結果として <Point srsName="EPSG:4326"><coordinates>1,2</coordinates></Point> を返します。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0, 4326 ).ST_AsXML('GML(Version=2)')
```

次の例では、ランダムな色で塗りつぶされた多角形を含む完全な SVG ドキュメントを返します。

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 0 20, 60 10, 0 0 ))' )
        .ST_AsXML( 'SVG' )
```

ST_Boundary メソッド

ジオメトリ値の境界を返します。

構文

geometry-expression.**ST_Boundary**()

戻り値

- **ST_Geometry** *geometry-expression* の境界を表すジオメトリ値を返します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ST_Boundary メソッドは、*geometry-expression* の空間境界を返します。ジオメトリは、内部、境界、外部で特徴付けられます。すべてのジオメトリ値がトポロジ的に閉じられたものとして定義されます。つまり、境界はジオメトリの一部と見なされます。

ポイントジオメトリの境界は空です。曲線ジオメトリは閉じられていることもあり、その場合、境界は空になります。曲線が閉じられていない場合、曲線の開始ポイントと終了ポイントによって境界が形成されます。面ジオメトリの場合、境界は面のエッジを描く曲線のセットです。たとえば、多角形の場合、ジオメトリの境界は外部リングと内部リング (存在する場合) で構成されます。

参照： 「ジオメトリの内部、外部、境界の操作」52 ページ.

注意
geometry-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.14

例

次の例では、多角形と線ストリングを含むジオメトリコレクションを構成し、そのコレクションの境界を返します。返される境界は、多角形の外部リングと、線ストリングの 2 つの終了ポイントを含むコレクションです。これは、コレクション 'GeometryCollection (LineString (0 0, 3 0, 3 3, 0 3, 0 0), MultiPoint ((0 7), (4 4)))' と同等です。

```
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 メソッド

ジオメトリ値に別のジオメトリ値が空間的に含まれているかどうかをテストします。

構文

geometry-expression.**ST_Contains**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT** *geometry-expression* に *geo2* が含まれている場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_Contains メソッドは、*geometry-expression* に *geo2* が完全に含まれており、*geometry-expression* の内部にある *geo2* の内部ポイントが 1 つ以上存在するかどうかをテストします。

geometry-expression.ST_Contains(*geo2*) は、*geo2*.ST_Within(*geometry-expression*) と同等です。

ST_Contains メソッドと ST_Covers メソッドは似ています。相違点は、ST_Covers では内部ポイントの交差が不要であることです。

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

- ST_Geometry タイプの ST_Within メソッド
- ST_Geometry タイプの ST_Covers メソッド
- ST_Geometry タイプの ST_Intersects メソッド
- ST_Geometry タイプの ST_ContainsFilter メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.31

例

次の例では、多角形にポイントが含まれているかどうかをテストします。多角形にはポイントが完全に含まれ、ポイント (ポイント自体) の内部が多角形の内部と交差しているため、1 が返されます。

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' )
   .ST_Contains( NEW ST_Point( 1, 1 ) )
```

次の例では、多角形に線が含まれているかどうかをテストします。多角形には線が完全に含まれていますが、線の内部と多角形の内部は交差していない (線は多角形の境界でのみ多角形と交差し、その境界は内部に含まれていない) ため、0 が返されます。ST_Contains の代わりに ST_Covers を使用した場合は、1 が返されます。

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' )
   .ST_Contains( NEW ST_LineString( 'LineString( 0 0, 1 0 )' ) )
```

次の例では、指定した多角形に含まれている各 Shape ジオメトリの ShapeID をリストします。この例では、結果として 16,17,19 を返します。多角形は多角形の境界でローの Shape ポイントと交差しているため、ShapeID 1 はリストされていないことに注意してください。

```
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 メソッド

ジオメトリに別のジオメトリが含まれているかどうかの低コストのテスト。

構文

geometry-expression.**ST_ContainsFilter**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT** *geometry-expression* に *geo2* が含まれている可能性がある場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_ContainsFilter メソッドは、一方のジオメトリにもう一方のジオメトリが含まれている可能性があるかどうかを調べる効率的なテストを提供します。*geometry-expression* に *geo2* が含まれている可能性がある場合は 1 を返し、それ以外の場合は 0 を返します。

このテストは ST_Contains よりも負荷は低いですが、*geometry-expression* に実際に *geo2* が含まれていない場合でも 1 を返すことがあります。

そのため、このメソッドは、今後の処理でジオメトリの相互の影響が望ましいものであるかどうかを判断するときにプライマリフィルターとして使用すると役立ちます。

ST_ContainsFilter の実装は、格納されているジオメトリに関連付けられているメタデータに依存します。使用可能なメタデータはサーバーのバージョン間で変わる可能性があるため (データのロード方法やクエリ内で ST_ContainsFilter が使用される場所に応じて決定される)、*geometry-expression* に *geo2* が含まれていない場合、式 *geometry-expression*.ST_ContainsFilter(*geo2*) は異なる結果を返すことがあります。*geometry-expression* に *geo2* が含まれている場合、ST_ContainsFilter は常に 1 を返します。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

- ST_Geometry タイプの ST_Contains メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

ST_ConvexHull メソッド

ジオメトリ値の凸包を返します。

構文

geometry-expression.**ST_ConvexHull**()

戻り値

- **ST_Geometry** ジオメトリ値が NULL または空の値の場合は、NULL を返します。それ以外の場合は、ジオメトリ値の凸包を返します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ジオメトリの凸包は、ジオメトリ内のすべてのポイントを含む最小の凸ジオメトリです。

凸包を思い浮かべるには、まず、ジオメトリ内のすべてのポイントを含むように引き伸ばしたゴムバンドを想像します。解放すると、ゴムバンドは凸包の形になります。

ジオメトリが 1 つのポイントで構成される場合は、そのポイントが返されます。ジオメトリのすべてのポイントが 1 つの直線セグメント上にある場合は、線ストリングが返されます。それ以外の場合は、凸多角形が返されます。

凸包は、元のジオメトリの近似値となります。空間関係をテストする場合、凸包は迅速な事前フィルターとして機能します。それは、凸包との空間的共通部分が存在しない場合、元のジオメトリとの共通部分は存在しない可能性があるためです。

注意
geometry-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_ConvexHull は、円ストリングを含むジオメトリではサポートされていません。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

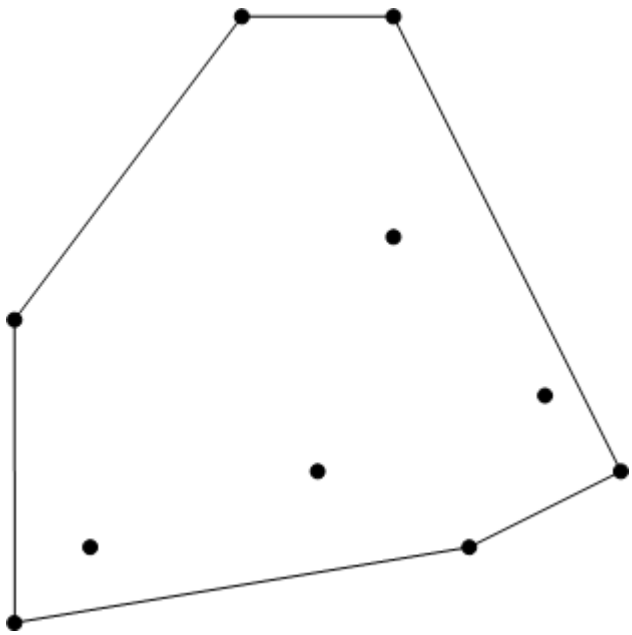
- ST_Geometry タイプの ST_ConvexHullAggr メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.16

例

次の例は、10 個のポイントから計算された凸包を示します。結果の凸包は Polygon ((1 1, 7 2, 9 3, 6 9, 4 9, 1 5, 1 1)) です。



```
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()
```

次の例では、1 つのポイント (0,0) を返します。1 つのポイントの凸包はポイントです。

```
SELECT NEW ST_Point(0,0).ST_ConvexHull()
```

次の例では、結果として LineString (0 0, 3 3) を返します。1 つの直線の凸包は、1 つのセグメントを含む線ストリングです。

```
SELECT NEW ST_LineString('LineString(0 0,1 1,2 2,3 3)').ST_ConvexHull()
```


ST_ConvexHullAggr メソッド

グループ内のすべてのジオメトリの凸包を返します。

構文

ST_Geometry::**ST_ConvexHullAggr**(*geometry-column*)

パラメーター

名前	型	説明
geometry-column	ST_Geometry	凸包を生成するジオメトリ値。通常、これはカラムです。

戻り値

- **ST_Geometry**　グループ内のすべてのジオメトリの凸包を返します。

  結果の空間参照系識別子は、最初のパラメーターの識別子と同じです。

備考

ST_ConvexHullAggr メソッドは、計算に使用されるジオメトリのグループ内のすべてのポイントを考慮し、これらすべてのポイントの凸包を返します。ジオメトリの凸包は、ジオメトリ内のすべてのポイントを含む最小の凸ジオメトリです。

凸包を思い浮かべるには、まず、ジオメトリ内のすべてのポイントを含むように引き伸ばしたゴムバンドを想像します。解放すると、ゴムバンドは凸包の形になります。

グループ内のジオメトリが 1 つのポイントで構成される場合は、そのポイントが返されます。ジオメトリのグループのすべてのポイントが 1 つの直線セグメント上にある場合は、線ストリングが返されます。それ以外の場合は、凸多角形が返されます。

凸包は、元のジオメトリの近似値となります。空間関係をテストする場合、凸包は迅速な事前フィルターとして機能します。それは、凸包との空間的共通部分が存在しない場合、元のジオメトリとの共通部分は存在しない可能性があるためです。

注意
ST_ConvexHullAggr は、円ストリングを含むジオメトリではサポートされていません。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

- ST_Geometry タイプの ST_ConvexHull メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として Polygon ((3 0, 7 2, 3 6, 0 7, -3 6, -3 3, 0 0, 3 0)) を返します。

```
SELECT ST_Geometry::ST_ConvexHullAggr( Shape )
FROM SpatialShapes WHERE ShapeID <= 16
```

ST_CoordDim メソッド

ST_Geometry 値の各ポイントで格納されている座標次元の数を返します。

構文

geometry-expression.**ST_CoordDim**()

戻り値

- **SMALLINT**　ST_Geometry 値の各ポイントで格納されている座標次元の数を示す 2 ～ 4 の値を返します。

備考

ST_CoordDim メソッドは、ジオメトリの各ポイント内に格納されている座標の数を返します。すべてのジオメトリに少なくとも 2 つの座標次元があります。地理的空間参照系の場合、これらの座標はポイントの緯度と経度です。他の空間参照系の場合、これらの座標はポイントの X 位置と Y 位置です。

ジオメトリの各ポイントには、必要に応じて Z 値と M 値を関連付けることができます。これらの追加の座標値は、空間関係や集合操作の計算時には考慮されませんが、追加情報を記録する目的で使用できます。たとえば、測定値 (M) を使用して、ジオメトリ内のさまざまなポイントでの汚染を記録できます。Z 値は一般的に標高を示すために使用しますが、その解釈はデータベースサーバーによって強制されていません。

ST_CoordDim メソッドによって返される可能性のある値は次のとおりです。

- **2**　ジオメトリには 2 つの座標だけ (緯度／経度または X/Y) が含まれます。
- **3**　ジオメトリにはポイントごとに 1 つの追加座標 (Z または M のいずれか) が含まれます。
- **4**　ジオメトリにはポイントごとに 2 つの追加座標 (Z と M の両方) が含まれます。

注意
集合操作によってジオメトリを結合する空間操作では、ジオメトリのポイントに関連付けられている Z 値も M 値も保持されません。

注意
ST_CoordDim では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_Is3D メソッド
- ST_Geometry タイプの ST_IsMeasured メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.3

例

次の例では、結果として 2 を返します。

```
SELECT NEW ST_Point(1.0, 1.0).ST_CoordDim()
```

次の例では、結果として 3 を返します。

```
SELECT NEW ST_Point(1.0, 1.0, 1.0, 0).ST_CoordDim()
```

次の例では、結果として 3 を返します。

```
SELECT NEW ST_Point('Point M (1 1 1)' ).ST_CoordDim()
```

次の例では、結果として 4 を返します。

```
SELECT NEW ST_Point('Point ZM (1 1 1 1)' ).ST_CoordDim()
```

ST_CoveredBy メソッド

ジオメトリ値が別のジオメトリ値に空間的に含まれているかどうかをテストします。

構文

geometry-expression.**ST_CoveredBy**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT** *geometry-expression* に *geo2* が含まれている場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_CoveredBy メソッドは、*geometry-expression* が *geo2* に完全に含まれているかどうかをテストします。

geometry-expression.ST_CoveredBy(*geo2*) は、*geo2*.ST_Covers(*geometry-expression*) と同等です。

この述部は、1 つのわずかな違いを除いて、ST_Within と同じです。ST_Within 述部では、*geometry-expression* の 1 つ以上の内部ポイントが *geo2* の内部にあることが必要となります。ST_CoveredBy() メソッドの場合、2 つのジオメトリの内部ポイントが交差しているかどうかに関係なく、*geo2* の外部に *geometry-expression* のポイントがなければ、1 が返されます。ST_CoveredBy は、曲面の空間参照系のジオメトリで使用できます。

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

参照

- ST_Geometry タイプの ST_Covers メソッド
- ST_Geometry タイプの ST_Within メソッド
- ST_Geometry タイプの ST_Intersects メソッド
- ST_Geometry タイプの ST_CoveredByFilter メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、ポイントが多角形に含まれているかどうかをテストします。ポイントは多角形に完全に含まれているため、1 が返されます。

```
SELECT NEW ST_Point( 1, 1 )
   .ST_CoveredBy( NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' ) )
```

次の例では、線が多角形に含まれているかどうかをテストします。線は多角形に完全に含まれているため、1 が返されます。ST_CoveredBy の代わりに ST_Within を使用した場合は、0 が返されます。

```
SELECT NEW ST_LineString( 'LineString( 0 0, 1 0 )' )
   .ST_CoveredBy( NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' ) )
```

次の例では、指定したポイントが Shape ジオメトリ内にある ShapeID をリストします。この例では、結果として 3,5,6 を返します。ポイントは多角形の境界でのみローの Shape 多角形と交差していますが、ShapeID 6 はリストされていることに注意してください。

```
SELECT LIST( ShapeID ORDER BY ShapeID )
FROM SpatialShapes
WHERE NEW ST_Point( 1, 4 ).ST_CoveredBy( Shape ) = 1
```

ST_CoveredByFilter メソッド

ジオメトリが別のジオメトリに含まれているかどうかの低コストのテスト。

構文

geometry-expression.**ST_CoveredByFilter**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT**　*geometry-expression* が *geo2* に含まれている可能性がある場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_CoveredByFilter メソッドは、一方のジオメトリがもう一方のジオメトリに含まれている可能性があるかどうかを調べる効率的なテストを提供します。

このテストは ST_CoveredBy よりも負荷は低いですが、*geometry-expression* が実際に *geo2* に含まれていない場合でも 1 を返すことがあります。

そのため、このメソッドは、今後の処理でジオメトリの相互の影響が望ましいものであるかどうかを判断するときにプライマリフィルターとして使用すると役立ちます。

ST_CoveredByFilter の実装は、格納されているジオメトリに関連付けられているメタデータに依存します。使用可能なメタデータはサーバーのバージョン間で変わる可能性があるため (データのロード方法やクエリ内で ST_CoveredByFilter が使用される場所に応じて決定される)、*geometry-expression* が *geo2* に含まれていない場合、式 *geometry-expression*.ST_CoveredByFilter(*geo2*) は異なる結果を返すことがあります。*geometry-expression* が *geo2* に含まれている場合、ST_CoveredByFilter は常に 1 を返します。

参照

- ST_Geometry タイプの ST_CoveredBy メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

ST_Covers メソッド

ジオメトリ値に別のジオメトリ値が空間的に含まれているかどうかをテストします。

構文

geometry-expression.**ST_Covers**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT** *geometry-expression* に *geo2* が含まれている場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_Covers メソッドは、*geometry-expression* に *geo2* が完全に含まれているかどうかをテストします。*geometry-expression*.ST_Covers(*geo2*) は、*geo2*.ST_CoveredBy(*geometry-expression*) と同等です。

この述部は、1 つのわずかな違いを除いて、ST_Contains と同じです。ST_Contains 述部では、*geo2* の 1 つ以上の内部ポイントが *geometry-expression* の内部にあることが必要となります。ST_Covers() メソッドの場合、*geometry-expression* の外部に *geo2* のポイントがなければ、1 が返されます。また、ST_Covers は曲面の空間参照系のジオメトリで使用できるのに対し、ST_Contains は使用できません。

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

参照

- ST_Geometry タイプの ST_CoveredBy メソッド
- ST_Geometry タイプの ST_Contains メソッド
- ST_Geometry タイプの ST_Intersects メソッド
- ST_Geometry タイプの ST_CoversFilter メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、多角形にポイントが含まれているかどうかをテストします。多角形にポイントが完全に含まれているため、1 が返されます。

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' )
   .ST_Covers( NEW ST_Point( 1, 1 ) )
```

次の例では、多角形に線が含まれているかどうかをテストします。多角形に線が完全に含まれているため、1 が返されます。ST_Covers の代わりに ST_Contains を使用した場合は、0 が返されます。

```
SELECT NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' )
   .ST_Covers( NEW ST_LineString( 'LineString( 0 0, 1 0 )' ) )
```


次の例では、指定した多角形に含まれている各 Shape ジオメトリの ShapeID をリストします。この例では、結果として 1,16,17,19,26 を返します。多角形は多角形の境界でのみローの Shape ポイントと交差していますが、ShapeID 1 はリストされていることに注意してください。

```
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 メソッド

ジオメトリに別のジオメトリが含まれているかどうかの低コストのテスト。

構文

geometry-expression.**ST_CoversFilter**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT**　*geometry-expression* に *geo2* が含まれている可能性がある場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_CoversFilter メソッドは、一方のジオメトリにもう一方のジオメトリが含まれている可能性があるかどうかを調べる効率的なテストを提供します。*geometry-expression* に *geo2* が含まれている可能性がある場合は 1 を返し、それ以外の場合は 0 を返します。

このテストは ST_Covers よりも負荷は低いですが、*geometry-expression* に実際に *geo2* が含まれていない場合でも 1 を返すことがあります。

そのため、このメソッドは、今後の処理でジオメトリの相互の影響が望ましいものであるかどうかを判断するときにプライマリフィルターとして使用すると役立ちます。

ST_CoversFilter の実装は、格納されているジオメトリに関連付けられているメタデータに依存します。使用可能なメタデータはサーバーのバージョン間で変わる可能性があるため (データのロード方法やクエリ内で ST_CoversFilter が使用される場所に応じて決定される)、*geometry-expression* に *geo2* が含まれていない場合、式 *geometry-expression*.ST_CoversFilter(*geo2*) は異なる結果を返すことがあります。*geometry-expression* に *geo2* が含まれている場合、ST_CoversFilter は常に 1 を返します。

参照

- ST_Geometry タイプの ST_Covers メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

ST_Crosses メソッド

ジオメトリ値が別のジオメトリ値と交差しているかどうかをテストします。

構文

geometry-expression.**ST_Crosses**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT**　*geometry-expression* が *geo2* と交差する場合は 1 を返し、それ以外の場合は 0 を返します。*geometry-expression* が面または複数面の場合、あるいは *geo2* がポイントまたは複数ポイントの場合は NULL を返します。

備考

ジオメトリ値が別のジオメトリ値と交差しているかどうかをテストします。

geometry-expression と *geo2* の両方が曲線または複数曲線の場合、それぞれの内部が 1 つ以上のポイントで交差していれば、これらのジオメトリは相互に交差していることになります。交差によって曲線または複数曲線が生じる場合、ジオメトリは交差していません。すべての交点が境界のポイントである場合、ジオメトリは交差していません。

geometry-expression の次元が *geo2* より低い場合、*geometry-expression* の一部が *geo2* の内部にあり、*geometry-expression* の一部が *geo2* の外部にもあれば、*geometry-expression* は *geo2* と交差しています。

より厳密には、*geometry-expression*.ST_Crosses(*geo2*) は、次の条件を満たす場合に 1 を返します。

```
( 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 )
```

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

- ST_Geometry タイプの ST_Intersects メソッド
- ST_Geometry タイプの ST_Dimension メソッド
- ST_Geometry タイプの ST_Relate メソッド
- ST_Geometry タイプの ST_Overlaps メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.29

例

次の例では、結果として 1 を返します。

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

次の例では、結果として 0 を返します。これは、2 つの線の内部が交差していない (最初の線ストリングの境界でのみ交差している) ためです。

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

次の例では、NULL を返します。これは、最初のジオメトリが面であるためです。

```
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 メソッド

2 つのジオメトリの差集合を表すジオメトリ値を返します。

構文

geometry-expression.**ST_Difference**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* から減算するもう一方のジオメトリ値。

戻り値

- **ST_Geometry** 2 つのジオメトリの差集合を表すジオメトリ値を返します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ST_Difference メソッドは、2 つのジオメトリの空間的差異を調べます。ポイントが結果に含まれるのは、それが *geometry-expression* に存在し、*geo2* に存在しない場合です。

他の空間集合操作 (ST_Union、ST_Intersection、ST_SymDifference) と異なり、ST_Difference() メソッドは非対称です。このメソッドは、 A.ST_Difference(B) と B.ST_Difference(A) に対して異なる回答を示すことがあります。

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

参照

- ST_Geometry タイプの ST_Intersection メソッド
- ST_Geometry タイプの ST_SymDifference メソッド
- ST_Geometry タイプの ST_Union メソッド

標準と互換性

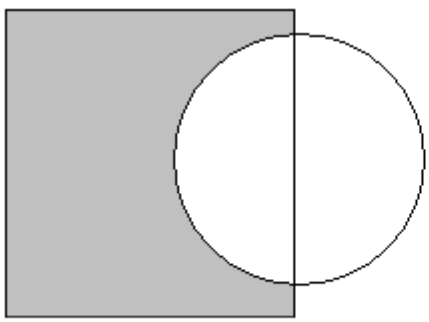
- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.20

例

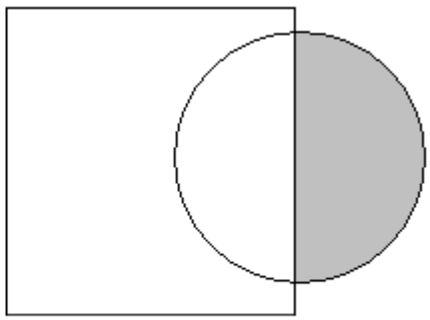
次の例は、正方形 (A) から円 (B) を削除した場合の差 (C) と円 (B) から正方形 (A) を削除した場合の差 (D) を示します。

```
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
```

次の図は、差 C=A-B と D=B-A (図の網掛け部分) を示します。それぞれの差は、左項のジオメトリにあり、右項のジオメトリにはないすべてのポイントを含む 1 つの面です。



C



D

ST_Dimension メソッド

ST_Geometry 値の次元を返します。ポイントの次元は 0、線の次元は 1、面の次元は 2 です。空のジオメトリの次元は -1 です。

構文

geometry-expression.**ST_Dimension**()

戻り値

- **SMALLINT**　*geometry-expression* の次元を -1 ～ 2 の SMALLINT として返します。

備考

ST_Dimension メソッドは、ジオメトリが占めている空間次元を返します。次の値が返されます。

- **-1**　ジオメトリは空のセットに対応します。
- **0**　ジオメトリは個々のポイントでのみ構成されます (たとえば、ST_Point または ST_MultiPoint)。
- **1**　ジオメトリには少なくとも 1 つの曲線が含まれ、面は含まれません (たとえば、ST_LineString または ST_MultiCurve)。
- **2**　ジオメトリは少なくとも 1 つの面で構成されます (たとえば、ST_Polygon または ST_MultiPolygon)。

コレクションの次元の計算時には、要素の最大次元が返されます。たとえば、ジオメトリコレクションに曲線とポイントが含まれている場合、ST_Dimension はそのコレクションについて 1 が返されます。

「空間の次元の操作」58 ページを参照してください。

注意
ST_Dimension では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_CoordDim メソッド
- ST_Geometry タイプの ST_Relate メソッド
- ST_Geometry タイプの ST_Relate メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.2

例

次の例では、結果として 0 を返します。

```
SELECT NEW ST_Point(1.0,1.0).ST_Dimension()
```

次の例では、結果として 1 を返します。

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

ST_Disjoint メソッド

ジオメトリ値が別のジオメトリ値から空間的に分断されているかどうかをテストします。

構文

geometry-expression.**ST_Disjoint**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT** *geometry-expression* が *geo2* から空間的に分断されている場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ジオメトリ値が別のジオメトリ値から空間的に分断されているかどうかをテストします。2 つのジオメトリの共通部分が空の場合、これらのジオメトリは分断されています。つまり、*geometry-expression* 内のどこにも *geo2* との共通ポイントが存在しない場合、これらのジオメトリは分断されています。

geometry-expression.ST_Disjoint(*geo2*) = 1 は、*geometry-expression*.ST_Intersects(*geo2*) = 0 と同等です。

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

参照

- ST_Geometry タイプの ST_Intersects メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.26

例

次の例では、指定した三角形との共通ポイントを持たない各シェイプについて、1 ローずつの結果を返します。


```
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
```

この例では、次の結果セットを返します。

ShapeID	Description
1	Point
22	Triangle

ST_Distance メソッド

geometry-expression と指定したジオメトリ値の間の最短距離を返します。

構文

geometry-expression.**ST_Distance**(*geo2*[, *unit-name*])

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* から距離が測定されるもう一方のジオメトリ値。
unit-name	VARCHAR(128)	距離を計算する単位。デフォルトでは、空間参照系の単位が使用されます。単位名は、UNIT_TYPE が 'LINEAR' の ST_UNITS_OF_MEASURE ビュー内のローの UNIT_NAME カラムと一致させてください。

戻り値

- **DOUBLE**　指定した線形測定単位で *geometry-expression* と *geo2* の間の最短距離を返します。*geometry-expression* または *geo2* のいずれかが空の場合は、NULL が返されます。

備考

ST_Distance メソッドは、2 つのジオメトリ間の最短距離を計算します。平面の空間参照系の場合、距離は、平面内の直交座標系における距離として、関連付けられている空間参照系の線形測定単位で計算されます。曲面の空間参照系の場合、距離は、空間参照系定義で楕円パラメーターを使用して、地表面の曲率を考慮して計算されます。

注意
曲面の空間参照系では、ST_Distance メソッドは *geometry-expression* と *geo2* にポイントだけが含まれている場合にのみサポートされます。

注意
ST_Distance では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Surface タイプの ST_Area メソッド
- ST_Curve タイプの ST_Length メソッド
- ST_Surface タイプの ST_Perimeter メソッド
- ST_Geometry タイプの ST_WithinDistance メソッド
- ST_Geometry タイプの ST_WithinDistanceFilter メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.23

例

次の例では、各シェイプとポイント (2,3) からのそれぞれの距離について、1 ローずつの結果セット (順序付けされたもの) を返します。

```
SELECT ShapeID, ROUND( Shape.ST_Distance( NEW ST_Point( 2, 3 ) ), 2 ) AS dist
FROM SpatialShapes
WHERE ShapeID < 17
ORDER BY dist
```

この例では、次の結果セットを返します。

ShapeID	dist
2	0.0
3	0.0
5	1.0
6	1.21
16	1.41
1	5.1

次の例では、カナダのハリファックス (NS) とワーテルロー (ON) を表すポイントを作成し、2 つのポイント間の距離をマイル単位で調べ、結果として 846 を返します。この例は、sa_install_feature システムプロシージャーによって 'st_geometry_predefined_uom' 機能がインス

トールされていることを前提としています。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

```
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 メソッド

ジオメトリ値の外接矩形を返します。

構文

geometry-expression.**ST_Envelope**()

戻り値

- **ST_Polygon** *geometry-expression* の外接矩形である多角形を返します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ST_Envelope メソッドは、*geometry-expression* の外接矩形 (軸と平行) である多角形を構成します。包絡線はジオメトリ全体を含み、ジオメトリの単純近似として使用できます。

注意
geometry-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

- ST_Geometry タイプの ST_EnvelopeAggr メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.15

例

次の例では、結果として Polygon ((0 0, 1 0, 1 4, 0 4, 0 0)) を返します。

```
SELECT Shape.ST_Envelope()
FROM SpatialShapes WHERE ShapeID = 6
```

ST_EnvelopeAggr メソッド

グループ内のすべてのジオメトリの外接矩形を返します。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

構文

ST_Geometry::**ST_EnvelopeAggr**(*geometry-column*)

パラメーター

名前	型	説明
geometry-column	ST_Geometry	外接矩形を生成するジオメトリ値。通常、これはカラムです。

戻り値

- **ST_Polygon**　グループ内のすべてのジオメトリの外接矩形である多角形を返します。

  結果の空間参照系識別子は、最初のパラメーターの識別子と同じです。

参照

- ST_Geometry タイプの ST_Envelope メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として Polygon ((-3 -1, 8 -1, 8 8, -3 8, -3 -1)) を返します。

```
SELECT ST_Geometry::ST_EnvelopeAggr( Shape ) FROM SpatialShapes
```

ST_Equals メソッド

ST_Geometry 値が別の ST_Geometry 値と空間的に等しいかどうかをテストします。

構文

geometry-expression.**ST_Equals**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT** 2 つのジオメトリ値が空間的に等しい場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_Geometry 値が別の ST_Geometry 値と等しいかどうかをテストします。

空間的に等しいかどうかのテストでは、まず 2 つのジオメトリの外接矩形が比較されます。それらの等価性が許容範囲外の場合、2 つのジオメトリは等しくないと見なされ、0 が返されます。それ以外の場合、geometry-expression.ST_SymDifference(geo2) が空のセットであれば 1 が返され、そうでなければ 0 が返されます。

SQL/MM 標準では、ST_SymDifference の観点でのみ ST_Equals が定義され、その他のバウンディングボックス比較は考慮されないことに注意してください。ジオメトリの中には、バウンディングボックスが等しくないのに、ST_SymDifference について空の結果を生成するものもあります。これらのジオメトリは SQL/MM 標準では等しいと見なされますが、SQL Anywhere では等しくないと見なされます。このような違いは、一方または両方のジオメトリに突起や陥没が含まれている場合に生じることがあります。

2 つのジオメトリ値は、それぞれの表現が異なっていても、等しいと見なされることがあります。たとえば、2 つの線ストリングの方向が反対でも、これらの線ストリングには空間内の同じポイントセットが含まれている場合があります。これら 2 つの線ストリングは ST_Equals では等しいと見なされますが、ST_OrderingEquals では等しくないと見なされます。「空間の比較操作」54 ページを参照してください。

ST_Equals は、空間参照系の精度またはデータの精度によって制限される場合があります。

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

参照

- ST_Geometry タイプの ST_OrderingEquals メソッド
- ST_Geometry タイプの ST_EqualsFilter メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.24

例

次の例では、結果として 16 を返します。結果の ShapeID 16 に対応する Shape には、指定した多角形と同じポイントが含まれていますが、それらの順序は異なります。

```
SELECT ShapeID FROM SpatialShapes
WHERE Shape.ST_Equals( NEW ST_Polygon( 'Polygon ((2 0, 1 2, 0 0, 2 0))' ) ) = 1
```

次の例では、結果として 1 を返します。この結果は、ポイント数も順序の指定も異なり、中間ポイントも線上に正確にあるわけではないのに、2 つの線ストリングが等しいことを示しています。中間ポイントは、2 つのポイントだけで構成される線から約 3.33e-7 離れていますが、その距離は「デフォルト」空間参照系 (SRID 0) の許容範囲 (1e-6) に収まっています。

```
SELECT NEW ST_LineString( 'LineString( 0 0, 0.333333 1, 1 3 )' )
    .ST_Equals( NEW ST_LineString( 'LineString( 1 3, 0 0 )' ) )
```

ST_EqualsFilter メソッド

ジオメトリが別のジオメトリと等しいかどうかの低コストのテスト。

構文

geometry-expression.**ST_EqualsFilter**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT** *geometry-expression* のバウンディングボックスと *geo2* のバウンディングボックスの等価性が許容範囲に収まっている場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_EqualsFilter メソッドは、一方のジオメトリがもう一方のジオメトリと等しい可能性があるかどうかを調べる効率的なテストを提供します。ST_EqualsFilter は、*geometry-expression* が *geo2* と等しい可能性がある場合は 1 を返し、それ以外の場合は 0 を返します。

このテストは ST_Equals よりも負荷は低いですが、*geometry-expression* が実際に *geo2* と等しくない場合でも 1 を返すことがあります。

そのため、このメソッドは、今後の処理でジオメトリの相互の影響が望ましいものであるかどうかを判断するときにプライマリフィルターとして使用すると役立ちます。

ST_EqualsFilter の実装は、格納されているジオメトリに関連付けられているメタデータに依存します。使用可能なメタデータはサーバーのバージョン間で変わる可能性があるため (データのロード方法やクエリ内で ST_EqualsFilter が使用される場所に応じて決定される)、*geometry-expression* が *geo2* と等しくない場合、式 *geometry-expression*.ST_EqualsFilter(*geo2*) は異なる結果を返すことがあります。*geometry-expression* が *geo2* と等しい場合、ST_EqualsFilter は常に 1 を返します。

参照

- ST_Geometry タイプの ST_Equals メソッド
- ST_Geometry タイプの ST_OrderingEquals メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

ST_GeomFromBinary メソッド

バイナリ文字列表現からジオメトリを構成します。

構文

ST_Geometry::**ST_GeomFromBinary**(*binary-string*[, *srid*])

パラメーター

名前	型	説明
binary-string	LONG BINARY	ジオメトリのバイナリ表現を含む文字列。入力には、WKB や EWKB を含め、サポートされている任意のバイナリフォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、入力文字列にも SRID が含まれていなければ、デフォルトの 0 が使用されます。

戻り値

- **ST_Geometry**　ソース文字列に基づいて、適切なタイプのジオメトリ値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

サポートされているフォーマットの 1 つを含む文字列を解析し、適切なタイプのジオメトリ値を作成します。このメソッドは、バイナリ文字列からジオメトリタイプへのキャストを評価するときにサーバーで使用されます。

一部の入力フォーマットには、SRID 定義が含まれます。*srid* パラメーターを指定する場合は、入力文字列から得られる値と一致させてください。

参照

- ST_Geometry タイプの ST_GeomFromWKB メソッド
- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として Point (10 20) を返します。

```
SELECT
ST_Geometry::ST_GeomFromBinary( 0x0101000000000000000000244000000000000003440 )
```

ST_GeomFromShape メソッド

ESRI シェイプレコードを含む文字列を解析し、適切なタイプのジオメトリ値を作成します。

構文

ST_Geometry::**ST_GeomFromShape**(*shape*[, *srid*])

パラメーター

名前	型	説明
shape	LONG BINARY	ESRI シェイプフォーマットのジオメトリを含む文字列。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_Geometry**　ソース文字列に基づいて、適切なタイプのジオメトリ値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

1 つの ESRI シェイプを含む文字列を解析し、適切なタイプのジオメトリ値を作成します。レコードは、ESRI シェイプファイルの *.shp* ファイル内の 1 つのレコードです。または、ジオデータベース内の 1 つの文字列値の場合もあります。

Shape 表現は、空間データを表すために広く使用されています。シェイプ定義の詳細については、ESRI の Web サイト http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf.を参照してください。

ESRI シェイプファイルをロードするときには、ほとんどの場合、ST_GeomFromShape メソッドを使用する代わりに、LOAD TABLE 文の FORMAT 句で SHAPEFILE フォーマットを使用するか、FROM 句で OPENSTRING 式を使用する方が簡単です。「LOAD TABLE 文」『SQL Anywhere サーバー SQL リファレンス』と「FROM 句」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

ST_GeomFromText メソッド

文字列表現からジオメトリを構成します。

構文

ST_Geometry::**ST_GeomFromText**(*character-string*[, *srid*])

パラメーター

名前	型	説明
character-string	LONG VARCHAR	ジオメトリのテキスト表現を含む文字列。入力には、Well Know Text (WKT) や 拡張 Well Know Text (EWKT) など、サポートされている任意のテキスト入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、入力文字列にも SRID が含まれていなければ、デフォルトの 0 が使用されます。

戻り値

- **ST_Geometry**　ソース文字列に基づいて、適切なタイプのジオメトリ値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ジオメトリを表すテキスト文字列を解析し、適切なタイプのジオメトリ値を作成します。このメソッドは、文字列からジオメトリタイプへのキャストを評価するときにサーバーで使用されます。

サーバーで入力文字列のフォーマットが検出されます。一部の入力フォーマットには、SRID 定義が含まれます。*srid* パラメーターを指定する場合は、入力文字列から得られる値と一致させてください。

参照

- ST_Geometry タイプの ST_GeomFromBinary メソッド
- ST_Geometry タイプの ST_GeomFromWKT メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.40

例

次の例では、結果として LineString (1 2, 5 7) を返します。

```
SELECT ST_Geometry::ST_GeomFromText( 'LineString( 1 2, 5 7 )', 4326 )
```

ST_GeomFromWKB メソッド

ジオメトリの WKB または EWKB 表現を含む文字列を解析し、適切なタイプのジオメトリ値を作成します。

構文

ST_Geometry::**ST_GeomFromWKB**(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	ジオメトリ値の WKB または EWKB 表現を含む文字列。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_Geometry**　ソース文字列に基づいて、適切なタイプのジオメトリ値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ジオメトリ値の WKB または EWKB 表現を含む文字列を解析し、適切なタイプのジオメトリ値を作成します。

参照

- ST_Geometry タイプの ST_GeomFromBinary メソッド
- ST_Geometry タイプの ST_GeomFromWKT メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.41

ST_GeomFromWKT メソッド

ジオメトリの WKT または EWKT 表現を含む文字列を解析し、適切なタイプのジオメトリ値を作成します。

構文

ST_Geometry::**ST_GeomFromWKT**(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	ジオメトリ値の WKT または EWKT 表現を含む文字列。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_Geometry**　ソース文字列に基づいて、適切なタイプのジオメトリ値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ジオメトリ値の WKT または EWKT 表現を含む文字列を解析し、適切なタイプのジオメトリ値を作成します。

参照

- ST_Geometry タイプの ST_GeomFromText メソッド
- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

ST_GeometryType メソッド

ST_Geometry 値のタイプの名前を返します。

構文

geometry-expression.**ST_GeometryType**()

戻り値

- **VARCHAR(128)**　ジオメトリ値のデータ型をテキスト文字列として返します。このメソッドを使用して、値の動的タイプを確認できます。

備考

ST_GeometryType メソッドは、*geometry-expression* の特定のタイプの名前を含む文字列を返します。

value IS OF(type) 構文を使用して、特定のタイプの値を判断することもできます。

注意
ST_GeometryType では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.4

例

次の例では、結果として 2,3,6,16,22,24,25 を返します。このリストは、指定したタイプのいずれかに該当する Shape の ShapeID です。

```
SELECT LIST( ShapeID ORDER BY ShapeID )
FROM SpatialShapes
WHERE Shape.ST_GeometryType() IN( 'ST_Polygon', 'ST_CurvePolygon' )
```

ST_GeometryTypeFromBaseType メソッド

タイプ文字列を定義している文字列を解析します。

構文

ST_Geometry::**ST_GeometryTypeFromBaseType**(*base-type-str*)

パラメーター

名前	型	説明
base-type-str	VARCHAR(128)	ベースタイプ文字列を含む文字列

戻り値

- **VARCHAR(128)** ベースタイプ文字列からジオメトリタイプを返します (SRID 定義が含まれている場合もあります)。タイプ文字列が有効なジオメトリタイプ文字列でない場合は、エラーが返されます。

備考

ST_Geometry::ST_GeometryTypeFromBaseType メソッドを使用して、タイプ文字列の定義からジオメトリタイプの名前を解析できます。

参照

- ST_Geometry タイプの ST_SRIDFromBaseType メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として ST_Geometry を返します。

```
SELECT ST_Geometry::ST_GeometryTypeFromBaseType('ST_Geometry')
```

次の例では、結果として ST_Point を返します。

```
SELECT ST_Geometry::ST_GeometryTypeFromBaseType('ST_Point(SRID=4326)')
```

次の例では、ストアドプロシージャーパラメーターによって受け入れられるジオメトリタイプ (ST_Point) を調べます。

```
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 メソッド

2 つのジオメトリの積集合を表すジオメトリ値を返します。

構文

geometry-expression.**ST_Intersection**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と交差するもう一方のジオメトリ値。

戻り値

- **ST_Geometry**　2 つのジオメトリの積集合を表すジオメトリ値を返します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ST_Intersection メソッドは、2 つのジオメトリの空間的共通部分を調べます。ポイントが入力ジオメトリの両方に存在する場合、そのポイントは共通部分に含まれています。2 つのジオメトリで共通ポイントを共有していない場合、結果は空のジオメトリです。

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

参照

- ST_Geometry タイプの ST_Difference メソッド
- ST_Geometry タイプの ST_IntersectionAggr メソッド
- ST_Geometry タイプの ST_SymDifference メソッド
- ST_Geometry タイプの ST_Union メソッド

標準と互換性

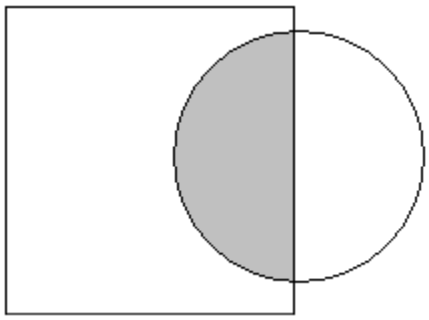
- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.18

例

次の例は、正方形 (A) と円 (B) の共通部分 (C) を示します。

```
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
```

次の図では、共通部分は網掛けになっています。これは、正方形にも円にも存在するすべてのポイントを含む 1 つの面です。



ST_IntersectionAggr メソッド

グループ内のすべてのジオメトリの空間的共通部分を返します。

構文

ST_Geometry::**ST_IntersectionAggr**(*geometry-column*)

パラメーター

名前	型	説明
geometry-column	ST_Geometry	空間的共通部分を生成するジオメトリ値。通常、これはカラムです。

戻り値

- **ST_Geometry**　グループ内のすべてのジオメトリの空間的共通部分であるジオメトリを返します。

  結果の空間参照系識別子は、最初のパラメーターの識別子と同じです。

備考

引数が NULL のローは含まれません。

空のグループ、または NULL 以外の値が含まれないグループの場合、NULL を返します。

グループに NULL 以外のジオメトリが 1 つだけ含まれている場合は、そのジオメトリが返されます。それ以外の場合、共通部分は、ST_Intersection メソッドを繰り返し適用して、一度に 2 つのジオメトリを結合することで論理的に計算されます。ST_Geometry タイプの ST_Intersection メソッド 201 ページを参照してください。

参照

- ST_Geometry タイプの ST_Intersection メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として Polygon ((0 0, 1 2, .5 2, .75 3, .555555 3, 0 1.75, .5 1.75, 0 0)) を返します。

```
SELECT ST_Geometry::ST_IntersectionAggr( Shape )
FROM SpatialShapes WHERE ShapeID IN ( 2, 6 )
```

ST_Intersects メソッド

ジオメトリ値が別の値と空間的に交差しているかどうかをテストします。

構文

geometry-expression.**ST_Intersects**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT**　*geometry-expression* が *geo2* と空間的に交差している場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ジオメトリ値が別の値と空間的に交差しているかどうかをテストします。2 つのジオメトリが 1 つ以上の共通ポイントを共有している場合、これらのジオメトリは交差しています。

geometry-expression.ST_Intersects(*geo2*) = 1 は、*geometry-expression*.ST_Disjoint(*geo2*) = 0 と同等です。

> **注意**
> *geometry-expression* に円ストリングが含まれている場合、それらは線ストリングに補間されます。

参照

- ST_Geometry タイプの ST_IntersectsRect メソッド
- ST_Geometry タイプの ST_Disjoint メソッド
- ST_Geometry タイプの ST_IntersectsFilter メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.27

例

次の例では、指定した線と交差している各シェイプについて、1 ローずつの結果を返します。

```
SELECT ShapeID, "Description"
FROM SpatialShapes
WHERE NEW ST_LineString( 'LineString( 2 2, 4 4 )' ).ST_Intersects( Shape ) = 1
ORDER BY ShapeID
```

この例では、次の結果セットを返します。

ShapeID	Description
2	Square
3	Rectangle
5	L shape line
18	CircularString
22	Triangle

SpatialShapes テーブル内のジオメトリが上記の例の線とどのように交差しているかを視覚化するには、Interactive SQL 空間ビューアーで次のクエリを実行します。

```
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 メソッド

2 つのジオメトリが交差しているかどうかの低コストのテスト。

構文

geometry-expression.**ST_IntersectsFilter**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT**　*geometry-expression* が *geo2* と交差している可能性がある場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_IntersectsFilter メソッドは、2 つのジオメトリが交差しているかどうかを調べる効率的なテストを提供します。*geometry-expression* が *geo2* と交差している可能性がある場合は 1 を返し、それ以外の場合は 0 を返します。

このテストは ST_Intersects よりも負荷は低いですが、ジオメトリが実際に交差していない場合でも 1 を返すことがあります。そのため、このメソッドは、今後の処理でジオメトリが本当に交差しているかどうかを判断するときにプライマリフィルターとして使用すると役立ちます。

ST_IntersectsFilter の実装は、格納されているジオメトリに関連付けられているメタデータに依存します。使用可能なメタデータはサーバーのバージョン間で変わる可能性があるため (データのロード方法やクエリ内で ST_IntersectsFilter が使用される場所に応じて決定される)、*geometry-expression* が *geo2* と交差していない場合、式 *geometry-expression*.ST_IntersectsFilter(*geo2*) は異なる結果を返すことがあります。*geometry-expression* が *geo2* と交差している場合、ST_IntersectsFilter は常に 1 を返します。

参照

- ST_Geometry タイプの ST_Intersects メソッド
- ST_Geometry タイプの ST_IntersectsRect メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

ST_IntersectsRect メソッド

ジオメトリが長方形と交差しているかどうかをテストします。

構文

geometry-expression.**ST_IntersectsRect**(*pmin*,*pmax*)

パラメーター

名前	型	説明
pmin	ST_Point	*geometry-expression* と比較する最小ポイント値。
pmax	ST_Point	*geometry-expression* と比較する最大ポイント値。

戻り値

- **BIT**　*geometry-expression* が指定した長方形と交差している場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_IntersectsRect メソッドは、ジオメトリが指定した外接矩形 (軸と平行) と交差しているかどうかをテストします。

このメソッドは、*geometry-expression*.ST_Intersects(NEW ST_Polygon(pmin, pmax)) と同等です。

そのため、このメソッドは、指定した長方形 (軸と平行) と交差しているすべてのジオメトリを検索する範囲検索を記述する場合に役立ちます。

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

参照

- ST_Geometry タイプの ST_Intersects メソッド
- ST_Geometry タイプの ST_IntersectsFilter メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、2 つのポイントの包絡線によって指定された長方形が交差している Shape ジオメトリの ShapeID をリストします。この例では、結果として 3,5,6,18 を返します。

```
SELECT LIST( ShapeID ORDER BY ShapeID )
FROM SpatialShapes
WHERE Shape.ST_IntersectsRect( NEW ST_Point( 0, 4 ), NEW ST_Point( 2, 5 ) ) = 1
```

次の例では、線ストリングが長方形と交差しているかどうかをテストします。指定した線ストリングは、2 つのポイントで識別された長方形と交差していません (ただし、線ストリングの包絡線は 2 つのポイントの包絡線と交差しています)。


```
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 メソッド

ジオメトリ値に Z 座標値が含まれているかどうかを調べます。

注意
ST_Is3D では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

geometry-expression.**ST_Is3D**()

戻り値

- **BIT**　ジオメトリ値に Z 座標値が含まれている場合は 1 を返し、それ以外の場合は 0 を返します。

参照

- ST_Geometry タイプの ST_CoordDim メソッド
- ST_Geometry タイプの ST_IsMeasured メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.10

例

次の例では、結果として 1 を返します。

```
SELECT ShapeID FROM SpatialShapes WHERE Shape.ST_Is3D() = 1
```

ST_IsEmpty メソッド

ジオメトリ値が空のセットを表すかどうかを調べます。

構文

geometry-expression.**ST_IsEmpty**()

戻り値

- **BIT**　ジオメトリ値が空の場合は 1 を返し、それ以外の場合は 0 を返します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.7

例

次の例では、結果として 1 を返します。

```
SELECT NEW ST_LineString().ST_IsEmpty()
```

ST_IsMeasured メソッド

ジオメトリ値に測定値が関連付けられているかどうかを調べます。

> **注意**
> ST_IsMeasured では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

geometry-expression.**ST_IsMeasured**()

戻り値

- **BIT** ジオメトリ値に測定値が含まれている場合は 1 を返し、それ以外の場合は 0 を返します。

参照

- ST_Geometry タイプの ST_CoordDim メソッド
- ST_Geometry タイプの ST_Is3D メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.11

例

次の例では、結果として 1 を返します。

```
SELECT ST_Geometry::ST_GeomFromText( 'LineString M( 1 2 4, 5 7 3 )' ).ST_IsMeasured()
```

次の例では、結果として 0 を返します。

```
SELECT count(*) FROM SpatialShapes WHERE Shape.ST_IsMeasured() = 1
```

ST_IsSimple メソッド

ジオメトリ値が単純かどうかを調べます (それ自体と交差しないことや他の不規則性など)。

構文

geometry-expression.**ST_IsSimple**()

戻り値

- **BIT**　ジオメトリ値が単純な場合は 1 を返し、それ以外の場合は 0 を返します。

参照

- ST_Geometry タイプの ST_IsValid メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.8

例

次の例では、結果として 29 を返します。これは、対応する複数線ストリングに含まれる 2 つの線が交差しているためです。

```
SELECT ShapeID FROM SpatialShapes WHERE Shape.ST_IsSimple() = 0
```

ST_IsValid メソッド

ジオメトリが有効な空間オブジェクトであるかどうかを調べます。

構文

geometry-expression.**ST_IsValid**()

戻り値

- **BIT**　ジオメトリ値が有効な場合は 1 を返し、それ以外の場合は 0 を返します。

備考

デフォルトでは、空間データが他のフォーマットから作成またはインポートされた場合、サーバーでデータの検証は行われません。ST_IsValid メソッドを使用して、インポートされたデータが有効なジオメトリを表すかどうかを検証できます。無効なジオメトリに対して操作を行うと、未定義の結果が返されます。

参照

- ST_Geometry タイプの ST_IsSimple メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.9

例

次の例では、結果として 0 を返します。これは、多角形にリボン形が含まれている (リングがそれ自体と交差している) ためです。

```
SELECT ST_Geometry::ST_GeomFromText( 'Polygon(( 0 0, 4 0, 4 5, 0 -1, 0 0 ))' )
   .ST_IsValid()
```

次の例では、結果として 0 を返します。これは、ジオメトリ内の多角形が面でそれ自体と交差しているためです。有限な数のポイントでのジオメトリコレクションのそれ自体との交差は有効であると見なされることに注意してください。

```
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 メソッド

ジオメトリの最北の緯度を取り出します。

構文

geometry-expression.**ST_LatNorth**()

戻り値

- **DOUBLE**　*geometry-expression* の最北の緯度を返します。

備考

geometry-expression の最北の緯度値を返します。曲面モデルでは、最北の緯度はジオメトリを定義しているどのポイントの緯度にも対応しない場合があることに注意してください。

注意
geometry-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_LatNorth では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_LatSouth メソッド
- ST_Geometry タイプの ST_LongEast メソッド
- ST_Geometry タイプの ST_LongWest メソッド
- ST_Geometry タイプの ST_YMax メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として 49.74 を返します。


```
SELECT ROUND( NEW ST_LineString( 'LineString( -122 49, -96 49 )', 4326 )
        .ST_LatNorth(), 2 )
```

ST_LatSouth メソッド

ジオメトリの最南の緯度を取り出します。

構文

geometry-expression.**ST_LatSouth**()

戻り値

- **DOUBLE**　*geometry-expression* の最南の緯度を返します。

備考

geometry-expression の最南の緯度値を返します。曲面モデルでは、最南の緯度はジオメトリを定義しているどのポイントの緯度にも対応しない場合があることに注意してください。

注意
geometry-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_LatSouth では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_LatNorth メソッド
- ST_Geometry タイプの ST_LongEast メソッド
- ST_Geometry タイプの ST_LongWest メソッド
- ST_Geometry タイプの ST_YMin メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として 49 を返します。

```
SELECT ROUND( NEW ST_LineString( 'LineString( -122 49, -96 49 )', 4326 )
        .ST_LatSouth(), 2 )
```

ST_LinearHash メソッド

ジオメトリの線形ハッシュであるバイナリ文字列を返します。

構文

geometry-expression.**ST_LinearHash**()

戻り値

- **BINARY(32)** ジオメトリの線形ハッシュであるバイナリ文字列を返します。

備考

空間インデックスサポートでは、ジオメトリの線形ハッシュを使用して、テーブル内のジオメトリを B ツリーインデックスの線形順序にマップします。ST_LinearHash メソッドは、B ツリーインデックスのローの順序を指定するバイナリ文字列を返すことによって、このマッピングを公開します。ハッシュ文字列には、「ジオメトリ *A* にジオメトリ *B* が含まれている場合は A.ST_LinearHash() >= B.ST_LinearHash()」というプロパティがあります。

線形ハッシュは ORDER BY 句で使用できます。たとえば、SELECT 文からデータをアップロードする場合、ST_LinearHash を使用して、空間インデックスのクラスタリングと一致するデータファイルを生成できます。

参照

- ST_Geometry タイプの ST_LinearUnHash メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

ST_LinearUnHash メソッド

インデックスハッシュを表すジオメトリを返します。

構文

ST_Geometry::**ST_LinearUnHash**(*index-hash*[, *srid*])

パラメーター

名前	型	説明
index-hash	BINARY(32)	インデックスハッシュ文字列。
srid	INT	インデックスハッシュの SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_Geometry**　指定した線形ハッシュの典型的なジオメトリを返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_LinearUnHash メソッドは、ST_LinearHash() によって生成された線形ハッシュ文字列の典型的なジオメトリを生成します。サーバーはジオメトリを空間インデックスの線形順序にマップし、ST_LinearHash メソッドはこの線形順序を定義するバイナリ文字列を提供します。
ST_LinearUnHash メソッドは、この操作を逆に実行して、特定のハッシュ文字列を表すジオメトリを提供します。複数の異なるジオメトリが同じバイナリ文字列のハッシュ値を持つ場合があるという点で、ハッシュ操作は損失を伴います。ST_LinearUnHash メソッドは、指定した線形ハッシュにマップするジオメトリを含むジオメトリを返します。

空間インデックスを使用するクエリのグラフィカルなプランは、空間インデックスの調査に使用される線形ハッシュ値を示します。ST_LinearUnHash メソッドを使用して、これらのハッシュを表すジオメトリを生成できます。

参照

- ST_Geometry タイプの ST_LinearHash メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

ST_LoadConfigurationData メソッド

バイナリ設定データを返します。内部でのみ使用。

構文

ST_Geometry::**ST_LoadConfigurationData**(*configuration-name*)

パラメーター

名前	型	説明
configuration-name	VARCHAR(128)	ロードする設定データ項目の名前。

戻り値

- **LONG BINARY**　バイナリ設定データを返します。内部でのみ使用されます。

備考

このメソッドは、インストールされているファイルから設定データをロードするときにサーバーで使用されます。サーバーに設定ファイルがインストールされていない場合は、NULL が返されます。内部でのみ使用。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

ST_LongEast メソッド

ジオメトリの東の境界の経度を取り出します。

構文

geometry-expression.**ST_LongEast**()

戻り値

- **DOUBLE**　*geometry-expression* の東の境界の経度を取り出します。

備考

geometry-expression の東の境界の経度を返します。曲面モデルでは、ジオメトリが日付変更線と交差すると、ST_LongWest は ST_LongEast 値より大きくなります。

注意
geometry-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_LongEast では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_LongWest メソッド
- ST_Geometry タイプの ST_LatNorth メソッド
- ST_Geometry タイプの ST_LatSouth メソッド
- ST_Geometry タイプの ST_XMax メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として -157.8 を返します。

```
SELECT NEW ST_LineString( 'LineString( -157.8 21.3, 144.5 13 )', 4326 )
        .ST_LongEast()
```

ST_LongWest メソッド

ジオメトリの西の境界の経度を取り出します。

構文

geometry-expression.**ST_LongWest**()

戻り値

- **DOUBLE**　*geometry-expression* の西の境界の経度を取り出します。

備考

geometry-expression の西の境界の経度を返します。曲面モデルでは、ジオメトリが日付変更線と交差すると、ST_LongWest は ST_LongEast 値より大きくなります。

> **注意**
> *geometry-expression* が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

> **注意**
> ST_LongWest では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_LongEast メソッド
- ST_Geometry タイプの ST_LatNorth メソッド
- ST_Geometry タイプの ST_LatSouth メソッド
- ST_Geometry タイプの ST_XMin メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として 144.5 を返します。

```
SELECT NEW ST_LineString( 'LineString( -157.8 21.3, 144.5 13 )', 4326 )
        .ST_LongWest()
```

ST_MMax メソッド

ジオメトリの最大 M 座標値を取り出します。

構文

geometry-expression.**ST_MMax**()

戻り値

- **DOUBLE**　*geometry-expression* の最大 M 座標値を返します。

備考

geometry-expression の最大 M 座標値を返します。この値は、ジオメトリ内のすべてのポイントの M 属性を比較して計算されます。

> **注意**
> *geometry-expression* が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

> **注意**
> ST_MMax では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_XMin メソッド
- ST_Geometry タイプの ST_XMax メソッド
- ST_Geometry タイプの ST_YMin メソッド
- ST_Geometry タイプの ST_YMax メソッド
- ST_Geometry タイプの ST_ZMin メソッド
- ST_Geometry タイプの ST_ZMax メソッド
- ST_Geometry タイプの ST_MMin メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として 8 を返します。

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_MMax()
```

ST_MMin メソッド

ジオメトリの最小 M 座標値を取り出します。

構文

geometry-expression.**ST_MMin**()

戻り値

- **DOUBLE**　*geometry-expression* の最小 M 座標値を返します。

備考

geometry-expression の最小 M 座標値を返します。この値は、ジオメトリ内のすべてのポイントの M 属性を比較して計算されます。

> **注意**
> *geometry-expression* が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

> **注意**
> ST_MMin では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_XMin メソッド
- ST_Geometry タイプの ST_XMax メソッド
- ST_Geometry タイプの ST_YMin メソッド
- ST_Geometry タイプの ST_YMax メソッド
- ST_Geometry タイプの ST_ZMin メソッド
- ST_Geometry タイプの ST_ZMax メソッド
- ST_Geometry タイプの ST_MMax メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として 4 を返します。

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_MMin()
```

ST_OrderingEquals メソッド

ジオメトリが別のジオメトリと同一であるかどうかをテストします。

構文

geometry-expression.**ST_OrderingEquals**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

● **BIT**　2 つのジオメトリ値が完全に等しい場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_Geometry 値が別の ST_Geometry 値と同一かどうかをテストします。2 つのジオメトリには同じオブジェクト階層が含まれ、その階層には ST_OrderingEquals で等しいと見なされる順序でまったく同じポイントが存在している必要があります。

ST_OrderingEquals メソッドは、曲線の方向が考慮されるという点で ST_Equals とは異なります。2 つの曲線には、まったく同じポイントが逆の順序で含まれていることがあります。これら 2 つの曲線は ST_Equals では等しいと見なされますが、ST_OrderingEquals では等しくないと見なされます。さらに、ST_OrderingEquals では、両方のジオメトリ内の各ポイントが完全に等しいことが要求されます (空間参照系で指定された許容範囲距離内で等しいだけでは不十分です)。

ST_OrderingEquals メソッドは、比較述部 (= と <>)、IN リスト述部、DISTINCT、GROUP BY に使用されるセマンティックを定義します。2 つの空間値が空間的に等しい (セット内に同じポイントセットが含まれている) かどうかを比較する場合、ST_Equals メソッドを使用できます。

詳細については、「空間の比較操作」 54 ページを参照してください。

注意
SQL/MM 標準では、ST_OrderingEquals は相対的な順序を返すように定義されています。2 つのジオメトリが空間的に等しい場合は (ST_Equals に基づく) 0 が返され、等しくない場合は 1 が返されます。SQL Anywhere の実装は業界の慣行に従っており、ブール値 (ジオメトリが等しいことを示す 1 と等しくないことを示す 0) を返すという点で SQL/MM とは異なります。さらに、ST_OrderingEquals の実装は、値が空間的に等しい (空間内に同じポイントセットが存在する) かどうかではなく、値が同一 (同じ順序の同じオブジェクト階層が存在する) かどうかをテストするという点で SQL/MM とは異なります。

参照

● ST_Geometry タイプの ST_Equals メソッド

標準と互換性

● **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.43

例

次の例では、結果として 16 を返します。結果の ShapeID 16 に対応する Shape には、指定した多角形とまったく同じポイントがまったく同じ順序で含まれています。

```
SELECT ShapeID FROM SpatialShapes
WHERE Shape.ST_OrderingEquals( NEW ST_Polygon( 'Polygon ((0 0, 2 0, 1 2, 0 0))' ) ) = 1
```


ST_Overlaps メソッド

ジオメトリ値が別のジオメトリ値と重なり合うかどうかをテストします。

構文

geometry-expression.**ST_Overlaps**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT** *geometry-expression* が *geo2* と重なり合っている場合は 1 を返し、それ以外の場合は 0 を返します。*geometry-expression* と *geo2* の次元が異なる場合は NULL を返します。

備考

次の条件をすべて満たす場合、2 つのジオメトリは重なり合っています。

●両方のジオメトリの次元が同じである。

●*geometry-expression* ジオメトリと *geo2* ジオメトリの共通部分の次元が *geometry-expression* と同じである。

●元のジオメトリのどちらも他方のサブセットではない。

より厳密には、*geometry-expression*.ST_Overlaps(*geo2*) は、次の条件を満たす場合に 1 を返します。

```
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
```

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

- ST_Geometry タイプの ST_Dimension メソッド
- ST_Geometry タイプの ST_Intersects メソッド
- ST_Geometry タイプの ST_Covers メソッド
- ST_Geometry タイプの ST_Crosses メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.32

例

次の例では、結果として 1 を返します。これは、2 つの線ストリングの共通部分も線ストリングになっており、どちらのジオメトリも他方のサブセットではないためです。

```
SELECT NEW ST_LineString( 'LineString( 0 0, 5 0 )' )
       .ST_Overlaps( NEW ST_LineString( 'LineString( 2 0, 3 0, 3 3 )' ) )
```

次の例では、結果として NULL を返します。これは、線ストリングとポイントの次元が異なるためです。

```
SELECT NEW ST_LineString( 'LineString( 0 0, 5 0 )' )
       .ST_Overlaps( NEW ST_Point( 1, 0 ) )
```

次の例では、結果として 0 を返します。これは、ポイントが複数ポイントのサブセットであるためです。

```
SELECT NEW ST_MultiPoint( 'MultiPoint(( 2 3 ), ( 1 0 ))' )
       .ST_Overlaps( NEW ST_Point( 1, 0 ) )
```

次の例では、結果として 24,25,28,31 を返します。このリストは、指定した多角形と重なり合う ShapeID です。

```
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 メソッド

ジオメトリ値が、交点マトリックスで指定されているとおりに別のジオメトリ値と空間的に関係しているかどうかをテストします。ST_Relate メソッドは、Dimensionally Extended 9 Intersection Model (DE-9IM) からの 9 文字の文字列を使用して、2 つの空間データ項目間のペアワイズの関係を示します。たとえば、ST_Relate メソッドは、ジオメトリ間に共通部分が存在するかどうかを確認し、存在する場合は結果の共通部分のジオメトリを調べます。参照： 「空間の関係操作」 55 ページ.

オーバーロードリスト

名前	説明
「ST_Geometry タイプの ST_Relate(ST_Geometry,CHAR(9)) メソッド」	ジオメトリ値が、交点マトリックスで指定されているとおりに別のジオメトリ値と空間的に関係しているかどうかをテストします。ST_Relate メソッドは、Dimensionally Extended 9 Intersection Model (DE-9IM) からの 9 文字の文字列を使用して、2 つの空間データ項目間のペアワイズの関係を示します。たとえば、ST_Relate メソッドは、ジオメトリ間に共通部分が存在するかどうかを確認し、存在する場合は結果の共通部分のジオメトリを調べます。
「ST_Geometry タイプの ST_Relate(ST_Geometry) メソッド」	交点マトリックスを返すことによって、ジオメトリ値がどのように別のジオメトリ値と空間的に関係しているかを確認します。ST_Relate メソッドは、Dimensionally Extended 9 Intersection Model (DE-9IM) からの 9 文字の文字列を返すことで、2 つの空間データ項目間のペアワイズの関係を示します。たとえば、ST_Relate メソッドは、ジオメトリ間に共通部分が存在するかどうかを確認し、存在する場合は結果の共通部分のジオメトリを調べます。

ST_Geometry タイプの ST_Relate(ST_Geometry,CHAR(9)) メソッド

ジオメトリ値が、交点マトリックスで指定されているとおりに別のジオメトリ値と空間的に関係しているかどうかをテストします。ST_Relate メソッドは、Dimensionally Extended 9 Intersection Model (DE-9IM) からの 9 文字の文字列を使用して、2 つの空間データ項目間のペアワイズの関係を示します。たとえば、ST_Relate メソッドは、ジオメトリ間に共通部分が存在するかどうかを確認し、存在する場合は結果の共通部分のジオメトリを調べます。

構文

geometry-expression.**ST_Relate**(*geo2*,*relate-matrix*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較する 2 番目のジオメトリ値。

名前	型	説明
relate-matrix	CHAR(9)	Dimensionally Extended 9 Intersection Model でマトリックスを表す 9 文字の文字列。9 文字の文字列に定義された各文字は、2 つのジオメトリの内部、境界、外部の間の考えられる 9 つの共通部分のいずれかで許可される共通部分タイプを表します。

戻り値

- **BIT**　2 つのジオメトリに指定の関係がある場合は 1 を返し、それ以外の場合は 0 を返します。

備考

2 つのジオメトリの内部、境界、外部の間の共通部分をテストすることによって、ジオメトリ値が、交点マトリックスで指定されているとおりに別のジオメトリ値と空間的に関係しているかどうかをテストします。参照：「空間の関係操作」55 ページ.

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.25

例

次の例では、指定した線と '0F***T***' の関係を持つ各シェイプについて、1 ローずつの結果を返します。'0' は、両方のジオメトリの内部が交差し、ポイントまたは複数ポイントを形成する必要があることを意味します。'F' は、線の内部と Shape の境界が交差してはならないことを意味します。'T' は、線の外部と Shape の内部が交差する必要があることを意味します。

```
SELECT ShapeID, "Description" From SpatialShapes
WHERE NEW ST_LineString( 'LineString( 0 0, 10 0 )' )
   .ST_Relate( Shape, '0F****T**' ) = 1
ORDER BY ShapeID
```

この例では、次の結果セットを返します。

ShapeID	Description
18	CircularString

ShapeID	Description
30	Multicurve

ST_Geometry タイプの ST_Relate(ST_Geometry) メソッド

交点マトリックスを返すことによって、ジオメトリ値がどのように別のジオメトリ値と空間的に関係しているかを確認します。ST_Relate メソッドは、Dimensionally Extended 9 Intersection Model (DE-9IM) からの 9 文字の文字列を返すことで、2 つの空間データ項目間のペアワイズの関係を示します。たとえば、ST_Relate メソッドは、ジオメトリ間に共通部分が存在するかどうかを確認し、存在する場合は結果の共通部分のジオメトリを調べます。

構文

geometry-expression.**ST_Relate**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較する 2 番目のジオメトリ値。

戻り値

- **CHAR(9)** Dimensionally Extended 9 Intersection Model でマトリックスを表す 9 文字の文字列を返します。9 文字の文字列内の各文字は、2 つのジオメトリの内部、境界、外部の間の考えられる 9 つの共通部分のいずれかの共通部分タイプを表します。

備考

2 つのジオメトリの内部、境界、外部の間の共通部分をテストすることによって、ジオメトリ値が、交点マトリックスで指定されているとおりに別のジオメトリ値と空間的に関係しているかどうかをテストします。

交点マトリックスは文字列として返されます。このメソッドの変形版を使用して、返された文字列を調べることによって空間関係をテストすることはできますが、パターン文字列を 2 番目のパラメーターとして渡すか、ST_Contains や ST_Intersects などの特定の空間述部を使用することによって、関係をテストする方が効率的です。参照： 「空間の関係操作」55 ページ.

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として 1F2001102 を返します。

```
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 メソッド

要素の順序を逆にしたジオメトリを返します。

構文

geometry-expression.**ST_Reverse**()

戻り値

- **ST_Geometry**　要素の順序を逆にしたジオメトリを返します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

要素の順序を逆にしたジオメトリを返します。曲線の場合、頂点の順序を逆にした曲線が返されます。コレクションの場合、子ジオメトリの順序を逆にしたコレクションが返されます。

> **注意**
> ST_Reverse では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として LineString (3 4, 1 2) を返します。これは、線ストリング内のポイントの順序が ST_Reverse によってどのように逆にされるかを示しています。

```
SELECT NEW ST_LineString( NEW ST_Point(1,2), NEW ST_Point(3,4) ).ST_Reverse()
```

ST_SRID メソッド

ジオメトリ値に関連付けられている空間参照系を取り出したり、変更したりします。

オーバーロードリスト

名前	説明
「ST_Geometry タイプの ST_SRID() メソッド」	ジオメトリの SRID を返します。
「ST_Geometry タイプの ST_SRID(INT) メソッド」	値を変更することなく、ジオメトリに関連付けられている空間参照系を変更します。

ST_Geometry タイプの ST_SRID() メソッド

ジオメトリの SRID を返します。

構文

geometry-expression.**ST_SRID**()

戻り値

- **INT**　ジオメトリの SRID を返します。

備考

ジオメトリの SRID を返します。すべてのジオメトリが空間参照系に関連付けられています。

注意
ST_SRID では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.5

例

次の例では、結果として 0 を返します。これは、テーブル内のすべての Shape の SRID が 0 (「デフォルト」空間参照系) であることを示しています。

```
SELECT DISTINCT Shape.ST_SRID() FROM SpatialShapes
```

ST_Geometry タイプの ST_SRID(INT) メソッド

値を変更することなく、ジオメトリに関連付けられている空間参照系を変更します。

構文

geometry-expression.**ST_SRID**(*srid*)

パラメーター

名前	型	説明
srid	INT	結果に使用する SRID。

戻り値

- **ST_Geometry**　指定した空間参照系を使用するジオメトリ値のコピーを返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_SRID メソッドは、srid パラメーターで指定された SRID を使用する *geometry-expression* のコピーを作成します。ST_SRID は、ソースとターゲットの両方の空間参照系で同じ座標系が使用されている場合に使用できます。

異なる座標系を使用する 2 つの空間参照系間でジオメトリを変換する場合は、ST_Transform メソッドを使用してください。

注意
ST_SRID では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_Transform メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.5

例

次の例では、結果として SRID=1000004326;Point (-118 34) を返します。

```
SELECT NEW ST_Point( -118, 34, 4326 ).ST_SRID( 1000004326 ).ST_AsText( 'EWKT' )
```

ST_SRIDFromBaseType メソッド

タイプ文字列を定義している文字列を解析します。

構文

ST_Geometry::**ST_SRIDFromBaseType**(*base-type-str*)

パラメーター

名前	型	説明
base-type-str	VARCHAR(128)	ベースタイプ文字列を含む文字列

戻り値

- **INT**　タイプ文字列から SRID を返します。文字列で SRID が指定されていない場合は NULL を返します。タイプ文字列が有効なジオメトリタイプ文字列でない場合は、エラーが返されます。

備考

ST_Geometry::ST_SRIDFromBaseType メソッドを使用して、タイプ文字列の定義から SRID を解析できます。

参照

- ST_Geometry タイプの ST_GeometryTypeFromBaseType メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として NULL を返します。

```
SELECT ST_Geometry::ST_SRIDFromBaseType('ST_Geometry')
```

次の例では、結果として 4326 を返します。

```
SELECT ST_Geometry::ST_SRIDFromBaseType('ST_Geometry(SRID=4326)')
```

ST_SnapToGrid メソッド

指定したグリッドにすべてのポイントがスナップされたジオメトリのコピーを返します。

オーバーロードリスト

名前	説明
「ST_Geometry タイプの ST_SnapToGrid(DOUBLE) メソッド」	指定したグリッドにすべてのポイントがスナップされたジオメトリのコピーを返します。
「ST_Geometry タイプの ST_SnapToGrid(ST_Point,DOUBLE,DOUBLE,DOUBLE,DOUBLE) メソッド」	指定したグリッドにすべてのポイントがスナップされたジオメトリのコピーを返します。

ST_Geometry タイプの ST_SnapToGrid(DOUBLE) メソッド

指定したグリッドにすべてのポイントがスナップされたジオメトリのコピーを返します。

構文

geometry-expression.**ST_SnapToGrid**(*cell-size*)

パラメーター

名前	型	説明
cell-size	DOUBLE	グリッドのセルサイズ。

戻り値

● **ST_Geometry**　グリッドにすべてのポイントがスナップされたジオメトリを返します。

結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ST_SnapToGrid メソッドを使用して、原点とセルサイズで定義したグリッドにジオメトリ内のすべてのポイントをスナップすることによって、データの精度を下げることができます。

X 座標と Y 座標がグリッドにスナップされます。Z 値と M 値は変更されません。

注意
精度を下げると、結果のジオメトリに別のプロパティが含まれる可能性があります。たとえば、単純な線ストリングがそれ自体と交差したり、無効なジオメトリが生成されたりする可能性があります。

注意
各空間参照系で、すべてのジオメトリが自動的にスナップされるグリッドが定義されます。この事前に定義されたグリッドよりも高い精度を格納することはできません。

注意
ST_SnapToGrid では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

● 「「グリッドにスナップ」と許容度が空間の計算に与える影響」

標準と互換性

● **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として LineString (1.536133 6.439453, 2.173828 6.100586) を返します。

```
SELECT NEW ST_LineString( 'LineString( 1.5358 6.4391, 2.17401 6.10018 )' )
          .ST_SnapToGrid( 0.001 )
```

それぞれの X 座標と Y 座標は、約 0.001 のグリッドサイズを使用して、最も近いグリッドポイントに移動されます。実際に使用されるグリッドサイズは、指定したグリッドサイズとは完全には一致しません。

ST_Geometry タイプの ST_SnapToGrid(ST_Point,DOUBLE,DOUBLE,DOUBLE,DOUBLE) メソッド

指定したグリッドにすべてのポイントがスナップされたジオメトリのコピーを返します。

構文

geometry-expression.**ST_SnapToGrid**(*origin*,*cell-size-x*,*cell-size-y*,*cell-size-z*,*cell-size-m*)

パラメーター

名前	型	説明
origin	ST_Point	グリッドの原点。
cell-size-x	DOUBLE	X 次元のグリッドのセルサイズ。
cell-size-y	DOUBLE	Y 次元のグリッドのセルサイズ。
cell-size-z	DOUBLE	Z 次元のグリッドのセルサイズ。
cell-size-m	DOUBLE	M 次元のグリッドのセルサイズ。

戻り値

- **ST_Geometry**　グリッドにすべてのポイントがスナップされたジオメトリを返します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ST_SnapToGrid メソッドを使用して、原点とセルサイズで定義したグリッドにジオメトリ内のすべてのポイントをスナップすることによって、データの精度を下げることができます。

次元ごとに異なるセルサイズを指定できます。1 次元のポイントをスナップしない場合は、セルサイズ 0 を使用できます。

注意
精度を下げると、結果のジオメトリに別のプロパティが含まれる可能性があります。たとえば、単純な線ストリングがそれ自体と交差したり、無効なジオメトリが生成されたりする可能性があります。

注意
各空間参照系で、すべてのジオメトリが自動的にスナップされるグリッドが定義されます。この事前に定義されたグリッドよりも高い精度を格納することはできません。

注意
ST_SnapToGrid では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「「グリッドにスナップ」と許容度が空間の計算に与える影響」

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として LineString (1.010101 20.20202, 1.015625 20.203125, 1.01 20.2) を返します。

```
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 ) )
```

線ストリングの最初のポイントは、SRID 0 に定義されたグリッドにスナップされたポイント ST_Point(1.010101, 20.202020) です。

線ストリングの 2 番目のポイントは、ポイント (0.0 0.0) の原点で定義されたグリッドにスナップされた同一ポイントです。ここで、セルサイズ x は POWER(2, -6)、セルサイズ y は POWER(2, -7) です。

線ストリングの 3 番目のポイントは、ポイント (1.01 20.2) の原点で定義されたグリッドにスナップされた同一ポイントです。ここで、セルサイズ x は POWER(2, -6)、セルサイズ y は POWER(2, -7) です。

ST_SymDifference メソッド

2 つのジオメトリの対称差を表すジオメトリ値を返します。

構文

geometry-expression.**ST_SymDifference**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	対称差を調べるために *geometry-expression* から減算するもう一方のジオメトリ値。

戻り値

- **ST_Geometry**　2 つのジオメトリの対称差を表すジオメトリ値を返します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ST_SymDifference メソッドは、2 つのジオメトリの対称差を調べます。対称差は、2 つのジオメトリのいずれかにのみ存在するすべてのポイントで構成されます。2 つのジオメトリ値が同じポイントで構成される場合、ST_SymDifference メソッドは空のジオメトリを返します。

> **注意**
> *geometry-expression* に円ストリングが含まれている場合、それらは線ストリングに補間されます。

参照

- ST_Geometry タイプの ST_Difference メソッド
- ST_Geometry タイプの ST_Intersection メソッド
- ST_Geometry タイプの ST_Union メソッド

標準と互換性

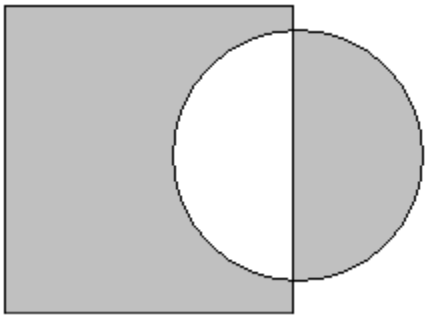
- **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.21

例

次の例は、正方形 (A) と円 (B) の対称差 (C) を示します。

```
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
```

次の図は、対称差の結果 (図の網掛け部分) を示します。対称差は、2 つの面を含む複数面です。一方の面には、正方形には存在し、円には存在しないすべてのポイントが含まれます。もう一方の面には、円には存在し、正方形には存在しないすべてのポイントが含まれます。



ST_ToCircular メソッド

ジオメトリを円ストリングに変換します。

構文

geometry-expression.**ST_ToCircular**()

戻り値

- **ST_CircularString** *geometry-expression* が ST_CircularString タイプの場合は、*geometry-expression* を返します。*geometry-expression* が、ST_CircularString タイプの 1 つの要素を含む ST_CompoundCurve タイプの場合は、その要素を返します。*geometry-expression* が、ST_CircularString タイプの 1 つの要素を含むジオメトリコレクションの場合は、その要素を返します。*geometry-expression* が空のセットの場合は、ST_CircularString タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

このジオメトリを円ストリングに変換します。このロジックは、CAST(*geometry-expression* AS ST_CircularString) に使用されるロジックと同等です。

geometry-expression が ST_CircularString 値であるとすでにわかっている場合は、ST_ToCircular メソッドよりも TREAT(*geometry-expression* AS ST_CircularString) を使用する方が効率的です。

注意
ST_ToCircular では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToCompound メソッド
- ST_Geometry タイプの ST_ToCurve メソッド
- ST_Geometry タイプの ST_ToLineString メソッド
- ST_Geometry タイプの ST_ToMultiCurve メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

例

次の例では、結果として CircularString (0 0, 1 1, 2 0) を返します。

```
SELECT NEW ST_CompoundCurve( 'CompoundCurve(CircularString( 0 0, 1 1, 2 0 ))' ).ST_ToCircular()
```

ST_ToCompound メソッド

ジオメトリを複合曲線に変換します。

構文

geometry-expression.**ST_ToCompound**()

戻り値

- **ST_CompoundCurve** *geometry-expression* が ST_CompoundCurve タイプの場合は、*geometry-expression* を返します。*geometry-expression* が ST_LineString または ST_CircularString タイプの場合は、1 つの要素 (*geometry-expression*) を含む複合曲線を返します。*geometry-expression* が、ST_Curve タイプの 1 つの要素を含むジオメトリコレクションの場合は、その要素キャストを ST_CompoundCurve として返します。*geometry-expression* が空のセットの場合は、ST_CompoundCurve タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ジオメトリを円ストリングに変換します。このロジックは、CAST(*geometry-expression* AS ST_CompoundCurve) に使用されるロジックと同等です。

geometry-expression が ST_CompoundCurve 値であるとすでにわかっている場合は、ST_ToCompound メソッドよりも TREAT(*geometry-expression* AS ST_CompoundCurve) を使用する方が効率的です。

注意
ST_ToCompound では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToCircular メソッド
- ST_Geometry タイプの ST_ToCurve メソッド
- ST_Geometry タイプの ST_ToLineString メソッド
- ST_Geometry タイプの ST_ToMultiCurve メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

例

次の例では、結果として CompoundCurve ((0 0, 2 1)) を返します。

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

ST_ToCurve メソッド

ジオメトリを曲線に変換します。

構文

geometry-expression.**ST_ToCurve**()

戻り値

- **ST_Curve** *geometry-expression* が ST_Curve タイプの場合は、*geometry-expression* を返します。*geometry-expression* が、ST_Curve タイプの 1 つの要素を含むジオメトリコレクションの場合は、その要素を返します。*geometry-expression* が空のセットの場合は、ST_LineString タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ジオメトリを曲線に変換します。このロジックは、**CAST(** *geometry-expression* **AS ST_Curve)** に使用されるロジックと同等です。

geometry-expression が ST_Curve 値であるとすでにわかっている場合は、ST_ToCurve メソッドよりも TREAT(*geometry-expression* AS ST_Curve) を使用する方が効率的です。

> **注意**
> ST_ToCurve では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToCircular メソッド
- ST_Geometry タイプの ST_ToCompound メソッド
- ST_Geometry タイプの ST_ToLineString メソッド
- ST_Geometry タイプの ST_ToMultiCurve メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として LineString (0 0, 1 1, 2 0) を返します。

```
SELECT NEW ST_GeomCollection( 'GeometryCollection(LineString(0 0, 1 1, 2 0))' ).ST_ToCurve()
```

ST_ToCurvePoly メソッド

ジオメトリを曲線多角形に変換します。

構文

geometry-expression.**ST_ToCurvePoly**()

戻り値

- **ST_CurvePolygon**　*geometry-expression* が ST_CurvePolygon タイプの場合は、*geometry-expression* を返します。*geometry-expression* が、ST_CurvePolygon タイプの 1 つの要素を含むジオメトリコレクションの場合は、その要素を返します。*geometry-expression* が空のセットの場合は、ST_CurvePolygon タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

geometry-expression が ST_CurvePolygon 値であるとすでにわかっている場合は、ST_ToCurvePoly メソッドよりも TREAT(*geometry-expression* AS ST_CurvePolygon) を使用する方が効率的です。

> **注意**
> ST_ToCurvePoly では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToPolygon メソッド
- ST_Geometry タイプの ST_ToSurface メソッド
- ST_Geometry タイプの ST_ToMultiSurface メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

例

次の例では、結果として Polygon ((0 0, 2 0, 1 2, 0 0)) を返します。

```
SELECT NEW ST_MultiPolygon('MultiPolygon(((0 0, 2 0, 1 2, 0 0)))' ).ST_ToCurvePoly()
```

ST_ToGeomColl メソッド

ジオメトリをジオメトリコレクションに変換します。

構文

geometry-expression.**ST_ToGeomColl**()

戻り値

- **ST_GeomCollection** *geometry-expression* が ST_GeomCollection タイプの場合は、*geometry-expression* を返します。*geometry-expression* が ST_Point、ST_Curve、または ST_Surface タイプの場合は、1 つの要素 (*geometry-expression*) を含むジオメトリコレクションを返します。*geometry-expression* が空のセットの場合は、ST_GeomCollection タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

geometry-expression が ST_GeomCollection 値であるとすでにわかっている場合は、ST_ToGeomColl メソッドよりも TREAT(*geometry-expression* AS ST_GeomCollection) を使用する方が効率的です。

> **注意**
> ST_ToGeomColl では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToMultiCurve メソッド
- ST_Geometry タイプの ST_ToMultiPoint メソッド
- ST_Geometry タイプの ST_ToMultiSurface メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

例

次の例では、結果として GeometryCollection (Point (0 1)) を返します。


```
SELECT NEW ST_Point( 0, 1 ).ST_ToGeomColl()
```

ST_ToLineString メソッド

ジオメトリを線ストリングに変換します。

構文

geometry-expression.**ST_ToLineString**()

戻り値

- **ST_LineString** *geometry-expression* が ST_LineString タイプの場合は、*geometry-expression* を返します。*geometry-expression* が ST_CircularString または ST_CompoundCurve の場合は、*geometry-expression*.ST_CurveToLine() を返します。*geometry-expression* が、ST_Curve タイプの 1 つの要素を含むジオメトリコレクションの場合は、その要素キャストを ST_LineString として返します。*geometry-expression* が空のセットの場合は、ST_LineString タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ジオメトリを線ストリングに変換します。このロジックは、CAST(*geometry-expression* AS ST_LineString) に使用されるロジックと同等です。*geometry-expression* が円ストリングまたは複合曲線の場合は、ST_CurveToLine() を使用して補間されます。

geometry-expression が ST_LineString 値であるとすでにわかっている場合は、ST_ToLineString メソッドよりも TREAT(*geometry-expression* AS ST_LineString) を使用する方が効率的です。

注意
ST_ToLineString では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToMultiLine メソッド
- ST_Geometry タイプの ST_ToCircular メソッド
- ST_Geometry タイプの ST_ToCompound メソッド
- ST_Geometry タイプの ST_ToCurve メソッド
- ST_Curve タイプの ST_CurveToLine メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

例

次の例では、Shape カラムが ST_Geometry タイプであり、ST_Geometry では ST_Length メソッドがサポートされていないため、エラーが返されます。

```
SELECT Shape.ST_Length()
FROM SpatialShapes WHERE ShapeID = 5
```

次の例では、ST_ToLineString を使用して、Shape カラム式のタイプを ST_LineString に変更します。ST_Length は、結果として 7 を返します。

```
SELECT Shape.ST_ToLineString().ST_Length()
FROM SpatialShapes WHERE ShapeID = 5
```

この場合、Shape カラムの値は ST_LineString タイプであるとわかっているため、TREAT を使用して式のタイプを効率的に変更できます。ST_Length は、結果として 7 を返します。

```
SELECT TREAT( Shape AS ST_LineString ).ST_Length()
FROM SpatialShapes WHERE ShapeID = 5
```

ST_ToMultiCurve メソッド

ジオメトリを複数曲線値に変換します。

構文

geometry-expression.**ST_ToMultiCurve**()

戻り値

- **ST_MultiCurve** *geometry-expression* が ST_MultiCurve タイプの場合は、*geometry-expression* を返します。*geometry-expression* が曲線のみを含むジオメトリコレクションの場合は、*geometry-expression* の要素を含む複数曲線オブジェクトを返します。*geometry-expression* が ST_Curve タイプの場合は、1 つの要素 (*geometry-expression*) を含む複数曲線値を返します。*geometry-expression* が空のセットの場合は、ST_MultiCurve タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

geometry-expression が ST_MultiCurve 値であるとすでにわかっている場合は、ST_ToMultiCurve メソッドよりも TREAT(*geometry-expression* AS ST_MultiCurve) を使用する方が効率的です。

注意
ST_ToMultiCurve では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToMultiLine メソッド
- ST_Geometry タイプの ST_ToGeomColl メソッド
- ST_Geometry タイプの ST_ToCurve メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

例

次の例では、結果として MultiCurve ((0 7, 0 4, 4 4)) を返します。

```
SELECT Shape.ST_ToMultiCurve()
FROM SpatialShapes WHERE ShapeID = 5
```

ST_ToMultiLine メソッド

ジオメトリを複数線ストリング値に変換します。

構文

geometry-expression.**ST_ToMultiLine**()

戻り値

- **ST_MultiLineString** *geometry-expression* が ST_MultiLineString タイプの場合は、*geometry-expression* を返します。*geometry-expression* が線のみを含むジオメトリコレクションの場合は、*geometry-expression* の要素を含む複数線ストリングオブジェクトを返します。*geometry-expression* が ST_LineString タイプの場合は、1 つの要素 (*geometry-expression*) を含む複数線ストリング値を返します。*geometry-expression* が空のセットの場合は、ST_MultiCurve タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

geometry-expression が ST_MultiLineString 値であるとすでにわかっている場合は、ST_ToMultiLine メソッドよりも TREAT(*geometry-expression* AS ST_MultiLineString) を使用する方が効率的です。

> **注意**
> ST_ToMultiLine では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToMultiCurve メソッド
- ST_Geometry タイプの ST_ToGeomColl メソッド
- ST_Geometry タイプの ST_ToLineString メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

例

次の例では、Shape カラムが ST_Geometry タイプであり、ST_Geometry では ST_Length メソッドがサポートされていないため、エラーが返されます。

```
SELECT Shape.ST_Length()
FROM SpatialShapes WHERE ShapeID = 29
```

次の例では、ST_ToMultiLine を使用して、Shape カラム式のタイプを ST_MultiLineString に変更します。この例は、Shape 値が ST_LineString タイプである ShapeID 5 にも使用できます。ST_Length は、結果として 4.236068 を返します。

```
SELECT Shape.ST_ToMultiLine().ST_Length()
FROM SpatialShapes WHERE ShapeID = 29
```

この場合、Shape カラムの値は ST_MultiLineString タイプであるとわかっているため、TREAT を使用して式のタイプを効率的に変更できます。この例は、Shape 値が ST_LineString タイプである ShapeID 5 には使用できません。「」ST_Length は、結果として 4.236068 を返します。

```
SELECT TREAT( Shape AS ST_MultiLineString ).ST_Length()
FROM SpatialShapes WHERE ShapeID = 29
```

ST_ToMultiPoint メソッド

ジオメトリを複数ポイント値に変換します。

構文

geometry-expression.**ST_ToMultiPoint**()

戻り値

- **ST_MultiPoint** *geometry-expression* が ST_MultiPoint タイプの場合は、*geometry-expression* を返します。*geometry-expression* がポイントのみを含むジオメトリコレクションの場合は、*geometry-expression* の要素を含む複数ポイントオブジェクトを返します。*geometry-expression* が ST_Point タイプの場合は、1 つの要素 (*geometry-expression*) を含む複数ポイント値を返します。*geometry-expression* が空のセットの場合は、ST_MultiPoint タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

geometry-expression が ST_MultiPoint 値であるとすでにわかっている場合は、ST_ToMultiPoint メソッドよりも TREAT(*geometry-expression* AS ST_MultiPoint) を使用する方が効率的です。

> **注意**
> ST_ToMultiPoint では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToGeomColl メソッド
- ST_Geometry タイプの ST_ToPoint メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

例

次の例では、結果として MultiPoint EMPTY を返します。

```
SELECT NEW ST_GeomCollection().ST_ToMultiPoint()
```

ST_ToMultiPolygon メソッド

ジオメトリを複数多角形値に変換します。

構文

geometry-expression.**ST_ToMultiPolygon**()

戻り値

- **ST_MultiPolygon** *geometry-expression* が ST_MultiPolygon タイプの場合は、*geometry-expression* を返します。*geometry-expression* が多角形のみを含むジオメトリコレクションの場合は、*geometry-expression* の要素を含む複数多角形オブジェクトを返します。*geometry-expression* が ST_Polygon タイプの場合は、1 つの要素 (*geometry-expression*) を含む複数多角形値を返します。*geometry-expression* が空のセットの場合は、ST_MultiSurface タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

geometry-expression が ST_MultiPolygon 値であるとすでにわかっている場合は、ST_ToMultiPolygon メソッドよりも TREAT(*geometry-expression* AS ST_MultiPolygon) を使用する方が効率的です。

注意
ST_ToMultiPolygon では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToMultiSurface メソッド
- ST_Geometry タイプの ST_ToGeomColl メソッド
- ST_Geometry タイプの ST_ToPolygon メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

例

次の例では、結果として MultiPolygon EMPTY を返します。

```
SELECT NEW ST_GeomCollection().ST_ToMultiPolygon()
```

次の例では、Shape カラムが ST_Geometry タイプであり、ST_Geometry では ST_Area メソッドがサポートされていないため、エラーを返します。

```
SELECT Shape.ST_Area()
FROM SpatialShapes WHERE ShapeID = 27
```

次の例では、ST_ToMultiPolygon を使用して、Shape カラム式のタイプを ST_MultiPolygon に変更します。この例は、Shape 値が ST_LineString タイプである ShapeID 22 にも使用できます。ST_Area は、結果として 8 を返します。

```
SELECT Shape.ST_ToMultiPolygon().ST_Area()
FROM SpatialShapes WHERE ShapeID = 27
```

この場合、Shape カラムの値は ST_MultiPolygon タイプであるとわかっているため、TREAT を使用して式のタイプを効率的に変更できます。この例は、Shape 値が ST_Polygon タイプである ShapeID 22 には使用できません。「」ST_Area は、結果として 8 を返します。

```
SELECT TREAT( Shape AS ST_MultiPolygon ).ST_Area()
FROM SpatialShapes WHERE ShapeID = 27
```

ST_ToMultiSurface メソッド

ジオメトリを複数面値に変換します。

構文

geometry-expression.**ST_ToMultiSurface**()

戻り値

- **ST_MultiSurface** *geometry-expression* が ST_MultiSurface タイプの場合は、*geometry-expression* を返します。*geometry-expression* が面のみを含むジオメトリコレクションの場合

は、*geometry-expression* の要素を含む複数面オブジェクトを返します。*geometry-expression* が ST_Surface タイプの場合は、1 つの要素 (*geometry-expression*) を含む複数面値を返します。*geometry-expression* が空のセットの場合は、ST_MultiSurface タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

geometry-expression が ST_MultiSurface 値であるとすでにわかっている場合は、ST_ToMultiSurface メソッドよりも TREAT(*geometry-expression* AS ST_MultiSurface) を使用する方が効率的です。

注意
ST_ToMultiSurface では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToMultiPolygon メソッド
- ST_Geometry タイプの ST_ToGeomColl メソッド
- ST_Geometry タイプの ST_ToSurface メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

例

次の例では、結果として MultiSurface EMPTY を返します。

```
SELECT NEW ST_GeomCollection().ST_ToMultiSurface()
```

次の例では、結果として MultiSurface (((3 3, 8 3, 4 8, 3 3))) を返します。

```
SELECT Shape.ST_ToMultiSurface()
FROM SpatialShapes WHERE ShapeID = 22
```

ST_ToPoint メソッド

ジオメトリをポイントに変換します。

構文

geometry-expression.**ST_ToPoint**()

戻り値

- **ST_Point** *geometry-expression* が ST_Point タイプの場合は、*geometry-expression* を返します。*geometry-expression* が、ST_Point タイプの 1 つの要素を含むジオメトリコレクションの場合

は、その要素を返します。*geometry-expression* が空のセットの場合は、ST_Point タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ジオメトリをポイントに変換します。このロジックは、**CAST(** *geometry-expression* **AS ST_Point)** に使用されるロジックと同等です。

geometry-expression が ST_Point 値であるとすでにわかっている場合は、ST_ToPoint メソッドよりも TREAT(*geometry-expression* AS ST_Point) を使用する方が効率的です。

注意
ST_ToPoint では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToMultiPoint メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

例

次の例では、結果として Point (1 2) を返します。

```
SELECT NEW ST_GeomCollection( NEW ST_Point(1,2) ).ST_ToPoint()
```

ST_ToPolygon メソッド

ジオメトリを多角形に変換します。

構文

geometry-expression.**ST_ToPolygon**()

戻り値

- **ST_Polygon** *geometry-expression* が ST_Polygon タイプの場合は、*geometry-expression* を返します。*geometry-expression* が ST_CurvePolygon タイプの場合は、*geometry-expression*.ST_CurvePolyToPoly() を返します。*geometry-expression* が、ST_CurvePolygon タイプの 1 つの要素を含むジオメトリコレクションの場合は、その要素を返します。*geometry-expression* が空のセットの場合は、ST_Polygon タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ジオメトリを多角形に変換します。このロジックは、**CAST(** *geometry-expression* **AS ST_Polygon)** に使用されるロジックと同等です。*geometry-expression* が曲線多角形の場合は、ST_CurvePolyToPoly() を使用して補間されます。

geometry-expression が ST_Polygon 値であるとすでにわかっている場合は、ST_ToPolygon メソッドよりも TREAT(*geometry-expression* AS ST_Polygon) を使用する方が効率的です。

> **注意**
> ST_ToPolygon では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToCurvePoly メソッド
- ST_Geometry タイプの ST_ToSurface メソッド
- ST_Geometry タイプの ST_ToMultiPolygon メソッド
- ST_CurvePolygon タイプの ST_CurvePolyToPoly メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.33

例

次の例では、結果として Polygon EMPTY を返します。

```
SELECT NEW ST_GeomCollection().ST_ToPolygon()
```

次の例では、Shape カラムが ST_Geometry タイプであり、ST_Geometry では ST_Area メソッドがサポートされていないため、エラーを返します。

```
SELECT Shape.ST_Area()
FROM SpatialShapes WHERE ShapeID = 22
```

次の例では、ST_ToPolygon を使用して、Shape カラム式のタイプを ST_Polygon に変更します。ST_Area は、結果として 12.5 を返します。

```
SELECT Shape.ST_ToPolygon().ST_Area()
FROM SpatialShapes WHERE ShapeID = 22
```

この場合、Shape カラムの値は ST_Polygon タイプであるとわかっているため、TREAT を使用して式のタイプを効率的に変更できます。ST_Area は、結果として 12.5 を返します。

```
SELECT TREAT( Shape AS ST_Polygon ).ST_Area()
FROM SpatialShapes WHERE ShapeID = 22
```

ST_ToSurface メソッド

ジオメトリを面に変換します。

構文

geometry-expression.**ST_ToSurface**()

戻り値

- **ST_Surface**　*geometry-expression* が ST_Surface タイプの場合は、*geometry-expression* を返します。*geometry-expression* が、ST_Surface タイプの 1 つの要素を含むジオメトリコレクションの場合は、その要素を返します。*geometry-expression* が空のセットの場合は、ST_Polygon タイプの空のセットを返します。それ以外の場合は、例外条件が発生します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ジオメトリを面に変換します。このロジックは、**CAST(** *geometry-expression* **AS ST_Surface)** に使用されるロジックと同等です。

geometry-expression が ST_Surface 値であるとすでにわかっている場合は、ST_ToSurface メソッドよりも TREAT(*geometry-expression* AS ST_Surface) を使用する方が効率的です。

注意
ST_ToSurface では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_ToCurvePoly メソッド
- ST_Geometry タイプの ST_ToPolygon メソッド
- ST_Geometry タイプの ST_ToMultiSurface メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として Polygon EMPTY を返します。

```
SELECT NEW ST_GeomCollection().ST_ToSurface()
```

ST_Touches メソッド

ジオメトリ値が別のジオメトリ値と空間的に接触しているかどうかをテストします。

構文

geometry-expression.**ST_Touches**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT** *geometry-expression* が *geo2* と接触している場合は 1 を返し、それ以外の場合は 0 を返します。*geometry-expression* と *geo2* の次元がどちらも 0 の場合は NULL を返します。

備考

ジオメトリ値が別のジオメトリ値と空間的に接触しているかどうかをテストします。2 つのジオメトリの内部は交差しておらず、一方の値の 1 つ以上の境界ポイントがもう一方の値の内部または境界と交差している場合、これらのジオメトリは空間的に接触しています。

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

- ST_Geometry タイプの ST_Intersects メソッド
- ST_Geometry タイプの ST_Boundary メソッド
- ST_Geometry タイプの ST_Dimension メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.28

例

次の例では、両方の入力がポイントであり、境界を持たないため、NULL を返します。

```
SELECT NEW ST_Point(1,1).ST_Touches( NEW ST_Point( 1,1 ) )
```

次の例では、ShapeID 6 の "Lighting Bolt" シェイプと接触しているジオメトリの ShapeID をリストします。この例では、結果として 5,16,26 を返します。接触している 3 つの各ジオメトリは境界でのみ Lighting Bolt と交差しています。

```
SELECT List( ShapeID ORDER BY ShapeID )
FROM SpatialShapes
WHERE Shape.ST_Touches( ( SELECT Shape FROM SpatialShapes WHERE ShapeID = 6 ) ) = 1
```

ST_Transform メソッド

指定した空間参照系に変換されたジオメトリ値のコピーを作成します。

構文

geometry-expression.**ST_Transform**(*srid*)

パラメーター

名前	型	説明
srid	INT	結果の SRID。

戻り値

- **ST_Geometry**　指定した空間参照系に変換されたジオメトリ値のコピーを返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_Transform メソッドは、現在の空間参照系の *geometry-expression* を指定の空間参照系に変換します。変換時には、両方の空間参照系の変換定義が使用されます。変換は、PROJ.4 ライブラリを使用して実行されます。

ST_Transform は、異なる座標系間での変換に必要です。たとえば、ST_Transform を使用して、緯度と経度を使用するジオメトリを SRID 3310 "NAD83 / California Albers" のジオメトリに変換できます。"NAD83 / California Albers" 空間参照系は、Albers 投影アルゴリズムとその線形測定単位のメートルを使用するカリフォルニア州のデータの平面投影です。

緯度経度系から直交座標系への変換では、極のポイントに問題が生じることがあります。データベースサーバーで北極または南極に近いポイントを変換できない場合、変換が成功するように、ポイントの緯度値が同じ経度に沿って極から若干 (1e-10 ラジアン強) 離れます。

同じ座標系を使用する 2 つの空間参照系間でジオメトリを変換する場合は、ST_Transform の代わりに ST_SRID メソッドを使用できます。

空間チュートリアルには、空間参照系間でデータを変換する方法を示す手順が記載されています。「チュートリアル：空間機能の実験」 59 ページを参照してください。

参照

- ST_Geometry タイプの ST_SRID メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.6

例

次の例では、結果として Point (184755.86861 -444218.175691) を返します。緯度と経度で指定されたロサンゼルス内のポイントを投影平面 SRID 3310 ("NAD83 / California Albers") に変換します。この例は、sa_install_feature システムプロシージャーによって 'st_geometry_predefined_srs' 機能がインストールされていることを前提としています。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。.

```
SELECT NEW ST_Point( -118, 34, 4326 ).ST_Transform( 3310 )
```


ST_Union メソッド

2 つのジオメトリの和集合を表すジオメトリ値を返します。

構文

geometry-expression.**ST_Union**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* との論理和を求めるもう一方のジオメトリ値。

戻り値

- **ST_Geometry** 2 つのジオメトリの和集合を表すジオメトリ値を返します。

  結果の空間参照系識別子は、*geometry-expression* の空間参照系と同じです。

備考

ST_Union メソッドは、2 つのジオメトリの空間的論理和を調べます。ポイントが 2 つの入力ジオメトリのいずれかに存在する場合、そのポイントは論理和に含まれています。

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

参照

- ST_Geometry タイプの ST_Difference メソッド
- ST_Geometry タイプの ST_Intersection メソッド
- ST_Geometry タイプの ST_SymDifference メソッド
- ST_Geometry タイプの ST_UnionAggr メソッド

標準と互換性

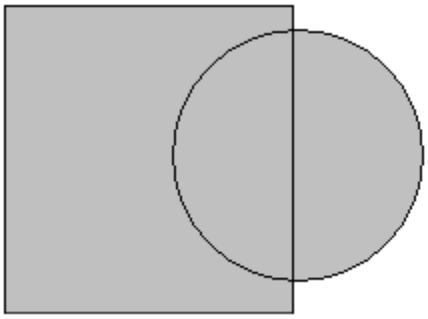
- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.19

例

次の例は、正方形 (A) と円 (B) の論理和 (C) を示します。

```
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
```

次の図では、論理和は網掛けになっています。論理和は、A または B に存在するすべてのポイントを含む 1 つの面です。



ST_UnionAggr メソッド

グループ内のすべてのジオメトリの空間的論理和を返します。

構文

ST_Geometry::**ST_UnionAggr**(*geometry-column*)

パラメーター

名前	型	説明
geometry-column	ST_Geometry	空間的論理和を生成するジオメトリ値。通常、これはカラムです。

戻り値

- **ST_Geometry**　グループ内のすべてのジオメトリの空間的論理和であるジオメトリを返します。

  結果の空間参照系識別子は、最初のパラメーターの識別子と同じです。

備考

引数が NULL のローは含まれません。

空のグループ、または NULL 以外の値が含まれないグループの場合、NULL を返します。

グループに NULL 以外のジオメトリが 1 つだけ含まれている場合は、そのジオメトリが返されます。それ以外の場合、論理和は、ST_Union メソッドを繰り返し適用して、一度に 2 つのジオメトリを結合することで論理的に計算されます。ST_Geometry タイプの ST_Union メソッド 249 ページを参照してください。

参照

- ST_Geometry タイプの ST_Union メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として Polygon ((.555555 3, 0 3, 0 1.75, 0 0, 3 0, 3 3, .75 3, 1 4, .555555 3)) を返します。

```
SELECT ST_Geometry::ST_UnionAggr( Shape )
FROM SpatialShapes WHERE ShapeID IN ( 2, 6 )
```

ST_Within メソッド

ジオメトリ値が別のジオメトリ値内に空間的に含まれているかどうかをテストします。

構文

geometry-expression.**ST_Within**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT** *geometry-expression* が *geo2* 内にある場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_Within メソッドは、*geometry-expression* が完全に *geo2* 内にあり、*geometry-expression* の内部にある *geo2* の内部ポイントが 1 つ以上存在するかどうかをテストします。

geometry-expression.ST_Within(*geo2*) は、*geo2*.ST_Contains(*geometry-expression*) と同等です。

ST_Within メソッドと ST_CoveredBy メソッドは似ています。相違点は、ST_CoveredBy では内部ポイントの交差が不要であることです。

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

- ST_Geometry タイプの ST_Contains メソッド
- ST_Geometry タイプの ST_CoveredBy メソッド
- ST_Geometry タイプの ST_Intersects メソッド
- ST_Geometry タイプの ST_WithinFilter メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.30

例

次の例では、ポイントが多角形内にあるかどうかをテストします。ポイントが完全に多角形内にあり、ポイント (ポイント自体) の内部が多角形の内部と交差しているため、1 が返されます。

```
SELECT NEW ST_Point( 1, 1 )
   .ST_Within( NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' ) )
```

次の例では、線が多角形内にあるかどうかをテストします。線は完全に多角形内にありますが、線の内部と多角形の内部は交差していない (線は多角形の境界でのみ多角形と交差し、その境界は内部に含まれていない) ため、0 が返されます。ST_Within の代わりに ST_CoveredBy を使用した場合は、1 が返されます。

```
SELECT NEW ST_LineString( 'LineString( 0 0, 1 0 )' )
   .ST_Within( NEW ST_Polygon( 'Polygon(( 0 0, 2 0, 1 2, 0 0 ))' ) )
```

次の例では、指定したポイントが Shape ジオメトリ内にある ShapeID をリストします。この例では、結果として 3,5 を返します。ポイントは多角形の境界でローの Shape 多角形と交差しているため、ShapeID 6 はリストされていないことに注意してください。

```
SELECT LIST( ShapeID ORDER BY ShapeID )
FROM SpatialShapes
WHERE NEW ST_Point( 1, 4 ).ST_Within( Shape ) = 1
```

ST_WithinDistance メソッド

2 つのジオメトリが指定の相互距離内にあるかどうかをテストします。

構文

geometry-expression.**ST_WithinDistance**(*geo2*,*distance*[, *unit-name*])

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* から距離が測定されるもう一方のジオメトリ値。

名前	型	説明
distance	DOUBLE	2 つのジオメトリ間の許容範囲距離。
unit-name	VARCHAR(128)	distance パラメーターを解釈するときに使用する単位。デフォルトでは、空間参照系の単位が使用されます。単位名は、UNIT_TYPE が 'LINEAR' の ST_UNITS_OF_MEASURE ビュー内のローの UNIT_NAME カラムと一致させてください。

戻り値

- **BIT**　*geometry-expression* と *geo2* が指定の相互距離内にある場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_WithinDistance メソッドは、2 つのジオメトリ間の最短距離が、許容範囲を考慮に入れたうえで指定距離を超えていないかどうかをテストします。

より厳密には、*d* が *geometry-expression* と *geo2* の間の最短距離を示すようにします。式 geometry-expression.ST_WithinDistance(geo2, distance[, unit_name]) は、*d* <= *distance* の場合、または関連付けられている空間参照系の許容範囲内の長さの分だけ *d* が *distance* を超えている場合は、1 と評価されます。

平面の空間参照系の場合、距離は、平面内の直交座標系における距離として、関連付けられている空間参照系の線形測定単位で計算されます。曲面の空間参照系の場合、距離は、空間参照系定義で楕円パラメーターを使用して、地表面の曲率を考慮して計算されます。

注意
geometry-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

注意
曲面の空間参照系では、ST_WithinDistance メソッドは *geometry-expression* と *geo2* にポイントだけが含まれている場合にのみサポートされます。

参照

- ST_Geometry タイプの ST_Distance メソッド
- ST_Geometry タイプの ST_WithinDistanceFilter メソッド
- ST_Geometry タイプの ST_Intersects メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、ポイント (2,3) の距離 1.4 までの範囲内にある各シェイプについて、1 ローずつの結果セット (順序付けされたもの) を返します。

```
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
```

この例では、次の結果セットを返します。

ShapeID	dist
2	0.0
3	0.0
5	1.0
6	1.21

次の例では、カナダのハリファックス (NS) とワーテルロー (ON) を表すポイントを作成し、2 つのポイント間の距離が 840 マイルではなく 850 マイルまでの範囲内にあることを示します。この例は、sa_install_feature システムプロシージャーによって 'st_geometry_predefined_uom' 機能がインストールされていることを前提としています。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

```
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
```

この例では、次の結果セットを返します。

within850	within840
1	0

ST_WithinDistanceFilter メソッド

2 つのジオメトリが指定距離内にあるかどうかを判定するための負荷の低い方法。

構文

geometry-expression.**ST_WithinDistanceFilter**(*geo2*,*distance*[, *unit-name*])

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* から距離が測定されるもう一方のジオメトリ値。
distance	DOUBLE	2 つのジオメトリ間の許容範囲距離。
unit-name	VARCHAR(128)	distance パラメーターを解釈するときに使用する単位。デフォルトでは、空間参照系の単位が使用されます。単位名は、UNIT_TYPE が 'LINEAR' の ST_UNITS_OF_MEASURE ビュー内のローの UNIT_NAME カラムと一致させてください。

戻り値

- **BIT**　*geometry-expression* と *geo2* が指定の相互距離内にある可能性がある場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_WithinDistanceFilter メソッドは、(ST_WithinDistance メソッドのように) 2 つのジオメトリが指定の相互距離内にある可能性があるかどうかを調べる効率的なテストを提供します。*geometry-expression* が *geo2* との間の指定距離内にある可能性がある場合は 1 を返し、それ以外の場合は 0 を返します。

このテストは ST_WithinDistance よりも低コストですが、2 つのジオメトリ間の最短距離が実際に指定距離より長い場合でも 1 を返すことがあります。そのため、このメソッドは、今後の処理でジオメトリ間の本当の距離を判断するときにプライマリフィルターとして使用すると役立ちます。

ST_WithinDistanceFilter の実装は、格納されているジオメトリに関連付けられているメタデータに依存します。使用可能なメタデータはサーバーのバージョン間で変わる可能性があるため (データのロード方法やクエリ内で ST_WithinDistanceFilter が使用される場所に応じて決定される)、*geometry-expression* が *geo2* との間の指定距離内にない場合、式 *geometry-expression*.ST_WithinDistanceFilter(*geo2*, *distance* [, *unit_name*]) は異なる結果を返すことがあります。*geometry-expression* が *geo2* との間の指定距離内にある場合、ST_WithinDistanceFilter は常に 1 を返します。

注意
ST_WithinDistanceFilter では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_Distance メソッド
- ST_Geometry タイプの ST_WithinDistance メソッド
- ST_Geometry タイプの ST_IntersectsFilter メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、ポイント (2,3) の距離 1.4 までの範囲内にある各シェイプについて、1 ローずつの結果セット (順序付けされたもの) を返します。結果には実際に指定距離内にないシェイプが含まれています。

```
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
```

この例では、次の結果セットを返します。

ShapeID	dist
2	0.0
3	0.0
5	1.0
6	1.21
16	1.41

次の例では、カナダのハリファックス (NS) とワーテルロー (ON) を表すポイントを作成し、ST_WithinDistanceFilter を使用して、2 つのポイント間の距離は明らかに 750 マイル以内ではありませんが、850 マイルまでの範囲内にあることを示します。この例は、sa_install_feature システムプロシージャーによって st_geometry_predefined_uom 機能がインストールされていることを前提としています。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

```
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
```

この例では、次の結果セットを返します。

within850	within750
1	0

ST_WithinFilter メソッド

ジオメトリが別のジオメトリ内にあるかどうかの低コストのテスト。

構文

geometry-expression.**ST_WithinFilter**(*geo2*)

パラメーター

名前	型	説明
geo2	ST_Geometry	*geometry-expression* と比較するもう一方のジオメトリ値。

戻り値

- **BIT**　*geometry-expression* が *geo2* 内にある可能性がある場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_WithinFilter メソッドは、一方のジオメトリがもう一方のジオメトリ内にある可能性があるかどうかを調べる効率的なテストを提供します。*geometry-expression* が *geo2* 内にある可能性がある場合は 1 を返し、それ以外の場合は 0 を返します。

このテストは ST_Within よりも負荷が低いですが、*geometry-expression* が実際に空間的に *geo2* 内にない場合でも 1 を返すことがあります。

そのため、このメソッドは、今後の処理でジオメトリの相互の影響が望ましいものであるかどうかを判断するときにプライマリフィルターとして使用すると役立ちます。

ST_WithinFilter の実装は、格納されているジオメトリに関連付けられているメタデータに依存します。使用可能なメタデータはサーバーのバージョン間で変わる可能性があるため (データのロード方法やクエリ内で ST_WithinFilter が使用される場所に応じて決定される)、*geometry-expression* が *geo2* 内にない場合、式 *geometry-expression*.ST_WithinFilter(*geo2*) は異なる結果を返すことがあります。*geometry-expression* が *geo2* 内にある場合、ST_WithinFilter は常に 1 を返します。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

- ST_Geometry タイプの ST_Within メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

ST_XMax メソッド

ジオメトリの最大 X 座標値を取り出します。

構文

geometry-expression.**ST_XMax**()

戻り値

- **DOUBLE** *geometry-expression* の最大 X 座標値を返します。

備考

geometry-expression の最大 X 座標値を返します。この値は、ジオメトリ内のすべてのポイントの X 属性を比較して計算されます。

地理的空間参照系では、返される値は軸順序の最初の座標に対応しています。軸順序が lat/lon/a/m の場合、最小値は ST_LongWest によって返される *geometry-expression* の西の境界に対応し、最大値は ST_LongEast によって返される *geometry-expression* の東の境界に対応します。そのため、曲面モデルでは、*geometry-expression* が日付変更線と交差すると、最小値は最大値より大きくなります。軸順序が lon/lat/z/m の場合、最小値は ST_LatSouth によって返される *geometry-expression* の最南ポイントに対応し、最大値は ST_LatNorth によって返される *geometry-expression* の最北ポイントに対応します。

注意
geometry-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_XMax では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_XMin メソッド
- ST_Geometry タイプの ST_YMin メソッド
- ST_Geometry タイプの ST_YMax メソッド
- ST_Geometry タイプの ST_ZMin メソッド
- ST_Geometry タイプの ST_ZMax メソッド
- ST_Geometry タイプの ST_MMin メソッド
- ST_Geometry タイプの ST_MMax メソッド
- ST_Geometry タイプの ST_LongEast メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として 5 を返します。

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_XMax()
```

ST_XMin メソッド

ジオメトリの最小 X 座標値を取り出します。

構文

geometry-expression.**ST_XMin**()

戻り値

- **DOUBLE**　*geometry-expression* の最小 X 座標値を返します。

備考

geometry-expression の最小 X 座標値を返します。この値は、ジオメトリ内のすべてのポイントの X 属性を比較して計算されます。

地理的空間参照系では、返される値は軸順序の最初の座標に対応しています。軸順序が lat/lon/a/m の場合、最小値は ST_LongWest によって返される *geometry-expression* の西の境界に対応し、最大値は ST_LongEast によって返される *geometry-expression* の東の境界に対応します。そのため、曲面モデルでは、*geometry-expression* が日付変更線と交差すると、最小値は最大値より大きくなります。軸順序が lon/lat/z/m の場合、最小値は ST_LatSouth によって返される *geometry-expression* の最南ポイントに対応し、最大値は ST_LatNorth によって返される *geometry-expression* の最北ポイントに対応します。

> **注意**
> *geometry-expression* が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_XMin では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_XMax メソッド
- ST_Geometry タイプの ST_YMin メソッド
- ST_Geometry タイプの ST_YMax メソッド
- ST_Geometry タイプの ST_ZMin メソッド
- ST_Geometry タイプの ST_ZMax メソッド
- ST_Geometry タイプの ST_MMin メソッド
- ST_Geometry タイプの ST_MMax メソッド
- ST_Geometry タイプの ST_LongWest メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として 1 を返します。

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_XMin()
```

ST_YMax メソッド

ジオメトリの最大 Y 座標値を取り出します。

構文

geometry-expression.**ST_YMax**()

戻り値

- **DOUBLE**　*geometry-expression* の最大 Y 座標値を返します。

備考

geometry-expression の最大 Y 座標値を返します。この値は、ジオメトリ内のすべてのポイントの Y 属性を比較して計算されます。

地理的空間参照系では、返される値は軸順序の最初の座標に対応しています。軸順序が lon/lat/z/m の場合、最小値は ST_LatSouth によって返される *geometry-expression* の最南ポイントに対応し、最大値は ST_LatNorth によって返される *geometry-expression* の最北ポイントに対応します。軸順序が lat/lon/a/m の場合、最小値は ST_LongWest によって返される *geometry-expression* の西の境界に対応し、最大値は ST_LongEast によって返される *geometry-expression* の東の境界に対応します。そのため、曲面モデルでは、*geometry-expression* が日付変更線と交差すると、最小値は最大値より大きくなります。

注意
geometry-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_YMax では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_XMin メソッド
- ST_Geometry タイプの ST_XMax メソッド
- ST_Geometry タイプの ST_YMin メソッド
- ST_Geometry タイプの ST_ZMin メソッド
- ST_Geometry タイプの ST_ZMax メソッド
- ST_Geometry タイプの ST_MMin メソッド
- ST_Geometry タイプの ST_MMax メソッド
- ST_Geometry タイプの ST_LatNorth メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として 6 を返します。

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_YMax()
```

ST_YMin メソッド

ジオメトリの最小 Y 座標値を取り出します。

構文

geometry-expression.**ST_YMin**()

戻り値

- **DOUBLE** *geometry-expression* の最小 Y 座標値を返します。

備考

geometry-expression の最小 Y 座標値を返します。この値は、ジオメトリ内のすべてのポイントの Y 属性を比較して計算されます。

地理的空間参照系では、返される値は軸順序の最初の座標に対応しています。軸順序が lon/lat/z/m の場合、最小値は ST_LatSouth によって返される *geometry-expression* の最南ポイントに対応

し、最大値は ST_LatNorth によって返される *geometry-expression* の最北ポイントに対応します。軸順序が lat/lon/a/m の場合、最小値は ST_LongWest によって返される *geometry-expression* の西の境界に対応し、最大値は ST_LongEast によって返される *geometry-expression* の東の境界に対応します。そのため、曲面モデルでは、*geometry-expression* が日付変更線と交差すると、最小値は最大値より大きくなります。

注意
geometry-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_YMin では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_XMin メソッド
- ST_Geometry タイプの ST_XMax メソッド
- ST_Geometry タイプの ST_YMax メソッド
- ST_Geometry タイプの ST_ZMin メソッド
- ST_Geometry タイプの ST_ZMax メソッド
- ST_Geometry タイプの ST_MMin メソッド
- ST_Geometry タイプの ST_MMax メソッド
- ST_Geometry タイプの ST_LatSouth メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として 2 を返します。

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_YMin()
```

ST_ZMax メソッド

ジオメトリの最大 Z 座標値を取り出します。

構文

geometry-expression.**ST_ZMax**()

戻り値

- **DOUBLE** *geometry-expression* の最大 Z 座標値を返します。

備考

geometry-expression の最大 Z 座標値を返します。この値は、ジオメトリ内のすべてのポイントの Z 属性を比較して計算されます。

> **注意**
> *geometry-expression* が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

> **注意**
> ST_ZMax では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_XMin メソッド
- ST_Geometry タイプの ST_XMax メソッド
- ST_Geometry タイプの ST_YMin メソッド
- ST_Geometry タイプの ST_YMax メソッド
- ST_Geometry タイプの ST_ZMin メソッド
- ST_Geometry タイプの ST_MMin メソッド
- ST_Geometry タイプの ST_MMax メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として **7** を返します。

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_ZMax()
```

ST_ZMin メソッド

ジオメトリの最小 Z 座標値を取り出します。

構文

geometry-expression.**ST_ZMin**()

戻り値

- **DOUBLE** *geometry-expression* の最小 Z 座標値を返します。

備考

geometry-expression の最小 Z 座標値を返します。この値は、ジオメトリ内のすべてのポイントの Z 属性を比較して計算されます。

注意
geometry-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_ZMin では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_XMin メソッド
- ST_Geometry タイプの ST_XMax メソッド
- ST_Geometry タイプの ST_YMin メソッド
- ST_Geometry タイプの ST_YMax メソッド
- ST_Geometry タイプの ST_ZMax メソッド
- ST_Geometry タイプの ST_MMin メソッド
- ST_Geometry タイプの ST_MMax メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として 3 を返します。

```
SELECT NEW ST_LineString( 'LineString ZM( 1 2 3 4, 5 6 7 8 )' ).ST_ZMin()
```

ST_LineString タイプ

ST_LineString タイプは、コントロールポイント間で直線セグメントを使用する ST_Curve のサブタイプです。

直接のスーパータイプ

- 「ST_Curve タイプ」

コンストラクター

- 「ST_LineString コンストラクター」

メソッド

- ST_LineString のメソッド：

ST_LineStringAggr	ST_NumPoints	ST_PointN	

●また、「ST_Curve タイプ」 のすべてのメソッドを ST_LineString タイプで呼び出すことができます。

ST_CurveToLine	ST_EndPoint	ST_IsClosed	ST_IsRing
ST_Length	ST_StartPoint		

●また、「ST_Geometry タイプ」 のすべてのメソッドを ST_LineString タイプで呼び出すことができます。

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

備考

ST_LineString タイプは、コントロールポイント間で直線セグメントを使用する ST_Curve のサブタイプです。連続するポイントの各ペアは直線セグメントでジョインされます。

線は、正確に 2 つのポイントで構成される ST_LineString 値です。線形リングは、閉じている単純な ST_LineString 値です。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.2

ST_LineString コンストラクター

線ストリングを構成します。

オーバーロードリスト

名前	説明
「ST_LineString() コンストラクター」	空のセットを表す線ストリングを構成します。
「ST_LineString(LONG VARCHAR[, INT]) コンストラクター」	テキスト表現から線ストリングを構成します。
「ST_LineString(LONG BINARY[, INT]) コンストラクター」	Well Known Binary (WKB) から線ストリングを構成します。
「ST_LineString(ST_Point,ST_Point,...) コンストラクター」	指定した空間参照系のポイントのリストから線ストリング値を構成します。

ST_LineString() コンストラクター

空のセットを表す線ストリングを構成します。

構文

NEW ST_LineString()

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 標準機能

例

次の例では、値が空であることを示す 1 を返します。

```
SELECT NEW ST_LineString().ST_IsEmpty()
```

ST_LineString(LONG VARCHAR[, INT]) コンストラクター

テキスト表現から線ストリングを構成します。

構文

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

パラメーター

名前	型	説明
text-representation	LONG VARCHAR	線ストリングのテキスト表現を含む文字列。入力には、Well Know Text (WKT) や拡張 Well Know Text (EWKT) など、サポートされている任意のテキスト入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

文字列表現から線ストリングを構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.2.2

例

次の例では、LineString (0 0, 5 10) を返します。

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

ST_LineString(LONG BINARY[, INT]) コンストラクター

Well Known Binary (WKB) から線ストリングを構成します。

構文

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

パラメーター

名前	型	説明
wkb	LONG BINARY	線ストリングのバイナリ表現を含む文字列。入力には、Well Known Binary (WKB) や拡張 Well Know Binary (EWKB) など、サポートされている任意のバイナリ入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

バイナリ文字列表現から線ストリングを構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 7.2.2

例

次の例では、LineString (0 0, 5 10) を返します。

```
SELECT NEW
ST_LineString(0x0102000000020000000000000000000000000000000000000000000000000000000000000000001440000000000000002440)
```

ST_LineString(ST_Point,ST_Point,...) コンストラクター

指定した空間参照系のポイントのリストから線ストリング値を構成します。

構文

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

パラメーター

名前	型	説明
pt1	ST_Point	線ストリングの最初のポイント。
pt2	ST_Point	線ストリングの 2 番目のポイント。
pt3,...,ptN	ST_Point	線ストリングの追加ポイント。

備考

ポイントのリストから線ストリング値を構成します。すべてのポイントの SRID を同じにしてください。結果の線ストリングは、この共通 SRID を使用して構成されます。指定するすべてのポイントが空ではなく、Is3D と IsMeasured に対して同じ回答を示す必要があります。すべてのポイントが 3D の場合に線ストリングも 3D になり、すべてのポイントが測定される場合に線ストリングも測定されます。

注意
ST_LineString では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として LineString (0 0, 1 1) を返します。

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

次の例では、結果として LineString (0 0, 1 1, 2 0) を返します。

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

ST_LineStringAggr メソッド

グループ内の順序付けされたポイントから構成された線ストリングを返します。

構文

ST_LineString::**ST_LineStringAggr**(*point*[**ORDER BY** *order-by-expression* [**ASC** | **DESC**], ...])

パラメーター

名前	型	説明
point	ST_Point	線ストリングを生成するポイント。通常、これはカラムです。

戻り値

- **ST_LineString**　グループ内のポイントから構成された線ストリングを返します。

  結果の空間参照系識別子は、最初のパラメーターの識別子と同じです。

備考

ST_LineStringAggr 集合メソッドを使用して、順序付けされたポイントのグループから線ストリングを構成できます。結合するすべてのジオメトリカラムの SRID を同じにしてください。また、結合するすべてのポイントが空ではなく、各ポイントの座標次元が同じになるようにしてください。

ポイントが NULL のローは含まれません。

空のグループ、または NULL 以外の値が含まれないグループの場合、NULL を返します。

結果の線ストリングの座標次元は、各ポイントの座標次元と同じになります。

注意
線ストリング内のポイントの順序を制御するには、ORDER BY 句を指定してください。この句が存在しない場合、線ストリング内のポイントの順序は、クエリオプティマイザーによって選択されたアクセスプランに応じて変わります。

注意
ST_LineStringAggr では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として LineString (0 0, 2 0, 1 1) を返します。

```
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 メソッド

線ストリングを定義しているポイント数を返します。

注意
ST_NumPoints では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

linestring-expression.**ST_NumPoints**()

戻り値

- **INT**　線ストリング値が空の場合は NULL を返し、それ以外の場合は値に含まれるポイント数を返します。

参照

- ST_LineString タイプの ST_PointN メソッド
- ST_CircularString タイプの ST_NumPoints メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　7.2.4

例

次の例では、結果として 3 を返します。

```
SELECT TREAT( Shape AS ST_LineString ).ST_NumPoints()
FROM SpatialShapes WHERE ShapeID = 5
```

ST_PointN メソッド

線ストリングの *n* 番目のポイントを返します。

注意
ST_PointN では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

linestring-expression.**ST_PointN**(*n*)

パラメーター

名前	型	説明
n	INT	返す要素の位置 (1 ～ *linestring-expression*.ST_NumPoints())。

戻り値

- **ST_Point**　*linestring-expression* の値が空のセットの場合は、NULL を返します。指定された位置 *n* が 1 未満か、ポイント数を超えている場合は、NULL を返します。それ以外の場合は、位置 n の ST_Point 値を返します。

  結果の空間参照系識別子は、*linestring-expression* の空間参照系と同じです。

参照

- ST_LineString タイプの ST_NumPoints メソッド
- ST_CircularString タイプの ST_PointN メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　7.2.5

例

次の例では、結果として Point (0 4) を返します。

```
SELECT TREAT( Shape AS ST_LineString ).ST_PointN( 2 )
FROM SpatialShapes WHERE ShapeID = 5
```

次の例では、geom の各ポイントにつき 1 つのローを返します。

```
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
```

この例では、次の結果セットを返します。

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

ST_MultiCurve タイプ

ST_MultiCurve は 0 個以上の ST_Curve 値のコレクションであり、すべての曲線が空間参照系内にあります。

直接のスーパータイプ

- 「ST_GeomCollection タイプ」

直接のサブタイプ

- 「ST_MultiLineString タイプ」

コンストラクター

- 「ST_MultiCurve コンストラクター」

メソッド

- ST_MultiCurve のメソッド：

ST_IsClosed	ST_Length	ST_MultiCurveAggr	

- また、「ST_GeomCollection タイプ」 のすべてのメソッドを ST_MultiCurve タイプで呼び出すことができます。

ST_GeomCollectionAggr	ST_GeometryN	ST_NumGeometries	

- また、「ST_Geometry タイプ」 のすべてのメソッドを ST_MultiCurve タイプで呼び出すことができます。

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

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.3

ST_MultiCurve コンストラクター

複数曲線を構成します。

オーバーロードリスト

名前	説明
「ST_MultiCurve() コンストラクター」	空のセットを表す複数曲線を構成します。
「ST_MultiCurve(LONG VARCHAR[, INT]) コンストラクター」	テキスト表現から複数曲線を構成します。

名前	説明
「ST_MultiCurve(LONG BINARY[, INT]) コンストラクター」	Well Known Binary (WKB) から複数曲線を構成します。
「ST_MultiCurve(ST_Curve,...) コンストラクター」	曲線値のリストから複数曲線を構成します。

ST_MultiCurve() コンストラクター

空のセットを表す複数曲線を構成します。

構文

NEW ST_MultiCurve()

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 標準機能

例

次の例では、値が空であることを示す 1 を返します。

```
SELECT NEW ST_MultiCurve().ST_IsEmpty()
```

ST_MultiCurve(LONG VARCHAR[, INT]) コンストラクター

テキスト表現から複数曲線を構成します。

構文

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

パラメーター

名前	型	説明
text-representation	LONG VARCHAR	複数曲線のテキスト表現を含む文字列。入力には、Well Know Text (WKT) や 拡張 Well Know Text (EWKT) など、サポートされている任意のテキスト入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

文字列表現から複数曲線を構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.3.2

例

次の例では、MultiCurve ((10 10, 12 12), CircularString (5 10, 10 12, 15 10)) を返します。

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

ST_MultiCurve(LONG BINARY[, INT]) コンストラクター

Well Known Binary (WKB) から複数曲線を構成します。

構文

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

パラメーター

名前	型	説明
wkb	LONG BINARY	複数曲線のバイナリ表現を含む文字列。入力には、Well Known Binary (WKB) や 拡張 Well Know Binary (EWKB) など、サポートされている任意のバイナリ入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

バイナリ文字列表現から複数曲線を構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.3.2

例

次の例では、MultiCurve (CircularString (5 10, 10 12, 15 10)) を返します。

```
SELECT NEW
ST_MultiCurve(0x010b0000000100000001080000000300000000000000000014400000000000002440
0000000000002440000000000000284000000000000002e40000000000002440)
```


ST_MultiCurve(ST_Curve,...) コンストラクター

曲線値のリストから複数曲線を構成します。

構文

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

パラメーター

名前	型	説明
curve1	ST_Curve	複数曲線の最初の曲線値。
curve2,...,curveN	ST_Curve	複数曲線の追加の曲線値。

備考

曲線値のリストから複数曲線を構成します。指定するすべての曲線値の SRID を同じにしてください。複数曲線は、この共通 SRID を使用して構成されます。

指定するすべての曲線値が Is3D と IsMeasured に対して同じ回答を示す必要があります。すべての曲線値が 3D の場合に複数曲線も 3D になり、すべての曲線値が測定される場合に複数曲線も測定されます。

注意
ST_MultiCurve では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として MultiCurve ((0 0, 1 1)) を返します。

```
SELECT NEW ST_MultiCurve( NEW ST_LineString('LineString (0 0, 1 1)' ) )
```

次の例では、結果として MultiCurve ((0 0, 1 1), CircularString (0 0, 1 1, 2 0)) を返します。

```
SELECT NEW ST_MultiCurve(
   NEW ST_LineString('LineString (0 0, 1 1)' ),
   NEW ST_CircularString( 'CircularString( 0 0, 1 1, 2 0)' ) )
```

ST_IsClosed メソッド

ST_MultiCurve 値が閉じているかどうかをテストします。開始ポイントと終了ポイントが一致する場合、曲線は閉じています。空ではなく、空の境界を持つ複数曲線は閉じています。

注意
ST_IsClosed では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

multicurve-expression.**ST_IsClosed**()

戻り値

- **BIT**　複数曲線が閉じている場合は 1 を返し、それ以外の場合は 0 を返します。

参照

- ST_Curve タイプの ST_IsClosed メソッド
- ST_Geometry タイプの ST_Boundary メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.3.3

例

次の例では、複数曲線の境界に 2 つのポイントがあるため、結果として 0 を返します。

```
SELECT NEW ST_MultiCurve( 'MultiCurve((0 0, 1 1))' ).ST_IsClosed()
```

次の例では、閉じたジオメトリを含む multicurve_table のすべてのローを返します。この例では、geometry カラムのタイプが ST_MultiCurve または ST_MultiLineString であるとします。

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

ST_Length メソッド

ST_MultiCurve 値の長さの測定値を返します。結果は、パラメーターで指定した単位で測定されます。

構文

multicurve-expression.**ST_Length**([*unit-name*])

パラメーター

名前	型	説明
unit-name	VARCHAR(128)	長さを計算する単位。デフォルトでは、空間参照系の単位が使用されます。単位名は、UNIT_TYPE が 'LINEAR' の ST_UNITS_OF_MEASURE ビュー内のローの UNIT_NAME カラムと一致させてください。

戻り値

- **DOUBLE**　ST_MultiCurve 値の長さの測定値を返します。

備考

ST_Length メソッドは、*unit-name* パラメーターで指定された単位で複数曲線の長さを返します。複数曲線の長さは、含まれている曲線の長さの合計です。曲線が空の場合は、NULL が返されます。

曲線に Z 値が含まれている場合、それらの値はジオメトリの長さの計算時には考慮されません。

注意
multicurve-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_Length では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Curve タイプの ST_Length メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.3.4

例

次の例では、複数曲線を作成し、ST_Length を使用してジオメトリの長さを調べ、値 PI+1 を返します。

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

次の例では、100 マイルを超える道路の名前と長さを返します。この例では、road テーブルが存在しており、geometry カラムのタイプが ST_MultiCurve または ST_MultiLineString で、st_geometry_predefined_uom のロードに sa_install_feature システムプロシージャーが使用されているものとします。

```
SELECT name, geometry.ST_Length( 'Statute Mile' ) len
FROM roads WHERE len > 100
```

ST_MultiCurveAggr メソッド

グループ内のすべての曲線を含む複数曲線を返します。

構文

ST_MultiCurve::**ST_MultiCurveAggr**(*geometry-column*[**ORDER BY** *order-by-expression* [**ASC** | **DESC**], ...])

パラメーター

名前	型	説明
geometry-column	ST_Curve	コレクションを生成するジオメトリ値。通常、これはカラムです。

戻り値

- **ST_MultiCurve**　グループ内のすべてのジオメトリを含む複数曲線を返します。

  結果の空間参照系識別子は、最初のパラメーターの識別子と同じです。

備考

ST_MultiCurveAggr 集合関数を使用して、曲線のグループを 1 つのコレクションに結合することができます。結合するすべてのジオメトリの SRID と座標次元の両方が同じになるようにします。

引数が NULL のローは含まれません。

空のグループ、または NULL 以外の値が含まれないグループの場合、NULL を返します。

結果の ST_MultiCurve の座標次元は、各曲線の座標次元と同じになります。

ST_GeometryN から目的の順序で要素が返されるように、オプションの ORDER BY 句を使用して要素を特定の順序に並べ替えることができます。この順序が意味を持たない場合は、順序を指定しない方が効率的です。その場合、要素の順序は、クエリオプティマイザーが選択したアクセスプランにより決定されます。

ST_MultiCurveAggr は ST_UnionAggr より効率的ですが、曲線のグループの中に重複する曲線が存在する場合には、そのような重複を伴ったコレクションを返す可能性があります。ST_UnionAggr では、重複するジオメトリを処理できます。

注意
ST_MultiCurveAggr では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_UnionAggr メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、SpatialShapes テーブルの ST_Curve タイプのすべてのジオメトリを、単一コレクションである ST_MultiCurve タイプに結合した単一の値を返します。Shape カラムが ST_Curve タイプの場合は、TREAT 関数と WHERE 句は必要ではありません。

```
SELECT ST_MultiCurve::ST_MultiCurveAggr( TREAT( Shape AS ST_Curve ) )
  FROM SpatialShapes WHERE Shape IS OF( ST_Curve )
```

ST_MultiLineString タイプ

ST_MultiLineString は 0 個以上の ST_LineString 値のコレクションであり、すべての線ストリングが空間参照系内にあります。

直接のスーパータイプ

- 「ST_MultiCurve タイプ」

コンストラクター

- 「ST_MultiLineString コンストラクター」

メソッド

- ST_MultiLineString のメソッド：

ST_MultiLineStringAggr			

- また、「ST_MultiCurve タイプ」 のすべてのメソッドを ST_MultiLineString タイプで呼び出すことができます。

ST_IsClosed	ST_Length	ST_MultiCurveAggr	

- また、「ST_GeomCollection タイプ」 のすべてのメソッドを ST_MultiLineString タイプで呼び出すことができます。

ST_GeomCollectionAggr	ST_GeometryN	ST_NumGeometries	

●また、「ST_Geometry タイプ」 のすべてのメソッドを ST_MultiLineString タイプで呼び出すことができます。

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

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.4

ST_MultiLineString コンストラクター

複数線ストリングを構成します。

オーバーロードリスト

名前	説明
「ST_MultiLineString() コンストラクター」	空のセットを表す複数線ストリングを構成します。
「ST_MultiLineString(LONG VARCHAR[, INT]) コンストラクター」	テキスト表現から複数線ストリングを構成します。
「ST_MultiLineString(LONG BINARY[, INT]) コンストラクター」	Well Known Binary (WKB) から複数線ストリングを構成します。
「ST_MultiLineString(ST_LineString,...) コンストラクター」	線ストリング値のリストから複数線ストリングを構成します。

ST_MultiLineString() コンストラクター

空のセットを表す複数線ストリングを構成します。

構文

NEW ST_MultiLineString()

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 標準機能

例

次の例では、値が空であることを示す 1 を返します。

```
SELECT NEW ST_MultiLineString().ST_IsEmpty()
```

ST_MultiLineString(LONG VARCHAR[, INT]) コンストラクター

テキスト表現から複数線ストリングを構成します。

構文

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

パラメーター

名前	型	説明
text-representation	LONG VARCHAR	複数線ストリングのテキスト表現を含む文字列。入力には、Well Know Text (WKT) や拡張 Well Know Text (EWKT) など、サポートされている任意のテキスト入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

文字列表現から複数線ストリングを構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.4.2

例

次の例では、MultiLineString ((10 10, 12 12), (14 10, 16 12)) を返します。

```
SELECT NEW ST_MultiLineString('MultiLineString ((10 10, 12 12), (14 10, 16 12))')
```

ST_MultiLineString(LONG BINARY[, INT]) コンストラクター

Well Known Binary (WKB) から複数線ストリングを構成します。

構文

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

パラメーター

名前	型	説明
wkb	LONG BINARY	複数線ストリングのバイナリ表現を含む文字列。入力には、Well Known Binary (WKB) や 拡張 Well Know Binary (EWKB) など、サポートされている任意のバイナリ入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

バイナリ文字列表現から複数線ストリングを構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.4.2

例

次の例では、MultiLineString ((10 10, 12 12)) を返します。

```
SELECT NEW
ST_MultiLineString(0x0105000000010000000102000000020000000000000000002440000000000000244000000000000028400000000000002840)
```

ST_MultiLineString(ST_LineString,...) コンストラクター

線ストリング値のリストから複数線ストリングを構成します。

構文

NEW ST_MultiLineString(*linestring1*[,*linestring2*,...,*linestringN*])

パラメーター

名前	型	説明
linestring1	ST_LineString	複数線ストリングの最初の線ストリング値。
linestring2,...,linestringN	ST_LineString	複数線ストリングの追加の線ストリング値。

備考

線ストリング値のリストから複数線ストリングを構成します。指定するすべての線ストリング値の SRID を同じにしてください。複数線ストリングは、この共通 SRID を使用して構成されます。

指定するすべての線ストリング値が Is3D と IsMeasured に対して同じ回答を示す必要があります。すべての線ストリング値が 3D の場合に複数線ストリングも 3D になり、すべての線ストリング値が測定される場合に複数線ストリングも測定されます。

注意
ST_MultiLineString では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、1 つの線ストリングを含む複数線ストリングを返します。これは、WKT 'MultiLineString ((0 0, 1 1))' と同等です。

```
SELECT NEW ST_MultiLineString( NEW ST_LineString('LineString (0 0, 1 1)' ) )
```

次の例では、2 つの線ストリングを含む複数線ストリングを返します。これは、WKT 'MultiLineString ((0 0, 1 1), (0 0, 1 1, 2 0))' と同等です。

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

ST_MultiLineStringAggr メソッド

グループ内のすべての線ストリングを含む複数線ストリングを返します。

構文

ST_MultiLineString::**ST_MultiLineStringAggr**(*geometry-column*[**ORDER BY** *order-by-expression* [**ASC** | **DESC**], ...])

パラメーター

名前	型	説明
geometry-column	ST_LineString	コレクションを生成するジオメトリ値。通常、これはカラムです。

戻り値

- **ST_MultiLineString**　グループ内のすべてのジオメトリを含む複数線ストリングを返します。

  結果の空間参照系識別子は、最初のパラメーターの識別子と同じです。

備考

ST_MultiLineStringAggr 集合関数を使用して、線ストリングのグループを 1 つのコレクションに結合することができます。結合するすべてのジオメトリの SRID と座標次元の両方が同じになるようにします。

引数が NULL のローは含まれません。

空のグループ、または NULL 以外の値が含まれないグループの場合、NULL を返します。

結果の ST_MultiLineString の座標次元は、各線ストリングの座標次元と同じになります。

ST_GeometryN から目的の順序で要素が返されるように、オプションの ORDER BY 句を使用して要素を特定の順序に並べ替えることができます。この順序が意味を持たない場合は、順序を指定しない方が効率的です。その場合、要素の順序は、クエリオプティマイザーが選択したアクセスプランにより決定されます。

ST_MultiLineStringAggr は ST_UnionAggr より効率的ですが、線ストリングのグループの中に重複する線ストリングが存在する場合には、そのような重複を伴ったコレクションを返す可能性があります。ST_UnionAggr では、重複するジオメトリを処理できます。

注意
ST_MultiLineStringAggr では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_UnionAggr メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、SpatialShapes テーブルの ST_LineString タイプのすべてのジオメトリを、単一コレクションである ST_MultiLineString タイプに結合した単一の値を返します。Shape カラムが ST_LineString タイプの場合は、TREAT 関数と WHERE 句は必要ではありません。

```
SELECT ST_MultiLineString::ST_MultiLineStringAggr( TREAT( Shape AS ST_LineString ) )
  FROM SpatialShapes WHERE Shape IS OF( ST_LineString )
```

ST_MultiPoint タイプ

ST_MultiPoint は 0 個以上の ST_Point 値のコレクションであり、すべてのポイントが空間参照系内にあります。

直接のスーパータイプ

- 「ST_GeomCollection タイプ」

コンストラクター

- 「ST_MultiPoint コンストラクター」

メソッド

- ST_MultiPoint のメソッド：

ST_MultiPointAggr			

- また、「ST_GeomCollection タイプ」 のすべてのメソッドを ST_MultiPoint タイプで呼び出すことができます。

ST_GeomCollectionAggr	ST_GeometryN	ST_NumGeometries	

- また、「ST_Geometry タイプ」 のすべてのメソッドを ST_MultiPoint タイプで呼び出すことができます。

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

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.2

ST_MultiPoint コンストラクター

複数ポイントを構成します。

オーバーロードリスト

名前	説明
「ST_MultiPoint() コンストラクター」	空のセットを表す複数ポイントを構成します。
「ST_MultiPoint(LONG VARCHAR[, INT]) コンストラクター」	テキスト表現から複数ポイントを構成します。

名前	説明
「ST_MultiPoint(LONG BINARY[, INT]) コンストラクター」	Well Known Binary (WKB) から複数ポイントを構成します。
「ST_MultiPoint(ST_Point,...) コンストラクター」	ポイント値のリストから複数ポイントを構成します。

ST_MultiPoint() コンストラクター

空のセットを表す複数ポイントを構成します。

構文

NEW ST_MultiPoint()

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 標準機能

例

次の例では、値が空であることを示す 1 を返します。

```
SELECT NEW ST_MultiPoint().ST_IsEmpty()
```

ST_MultiPoint(LONG VARCHAR[, INT]) コンストラクター

テキスト表現から複数ポイントを構成します。

構文

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

パラメーター

名前	型	説明
text-representation	LONG VARCHAR	複数ポイントのテキスト表現を含む文字列。入力には、Well Know Text (WKT) や拡張 Well Know Text (EWKT) など、サポートされている任意のテキスト入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

文字列表現から複数ポイントを構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.2.2

例

次の例では、MultiPoint ((10 10), (12 12), (14 10)) を返します。

```
SELECT NEW ST_MultiPoint('MultiPoint ((10 10), (12 12), (14 10))')
```

ST_MultiPoint(LONG BINARY[, INT]) コンストラクター

Well Known Binary (WKB) から複数ポイントを構成します。

構文

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

パラメーター

名前	型	説明
wkb	LONG BINARY	複数ポイントのバイナリ表現を含む文字列。入力には、Well Known Binary (WKB) や拡張 Well Know Binary (EWKB) など、サポートされている任意のバイナリ入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

バイナリ文字列表現から複数ポイントを構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.2.2

例

次の例では、MultiPoint ((10 10), (12 12), (14 10)) を返します。

```
SELECT NEW
ST_MultiPoint(0x0104000000030000000101000000000000000024400000000000002440010100000
000000000000284000000000000028400101000000000000000002c40000000000002440)
```

ST_MultiPoint(ST_Point,...) コンストラクター

ポイント値のリストから複数ポイントを構成します。

構文

NEW ST_MultiPoint(*point1*[,*point2*,...,*pointN*])

パラメーター

名前	型	説明
point1	ST_Point	複数ポイントの最初のポイント値。
point2,...,pointN	ST_Point	複数ポイントの追加のポイント値。

備考

ポイント値のリストから複数ポイントを構成します。指定するすべてのポイント値の SRID を同じにしてください。複数ポイントは、この共通 SRID を使用して構成されます。

指定するすべてのポイント値が Is3D と IsMeasured に対して同じ回答を示す必要があります。すべてのポイント値が 3D の場合に複数ポイントも 3D になり、すべてのポイント値が測定される場合に複数ポイントも測定されます。

注意
ST_MultiPoint では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、1 つのポイント 'Point (1 2)' を含む複数ポイントを返します。

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

次の例では、2 つのポイント 'Point (1 2)' および 'Point (3 4)' を含む複数ポイントを返します。

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


ST_MultiPointAggr メソッド

グループ内のすべてのポイントを含む MultiPoint を返します。

構文

ST_MultiPoint::**ST_MultiPointAggr**(*geometry-column*[**ORDER BY** *order-by-expression* [**ASC** | **DESC**], ...])

パラメーター

名前	型	説明
geometry-column	ST_Point	コレクションを生成するジオメトリ値。通常、これはカラムです。

戻り値

- **ST_MultiPoint**　グループ内のすべてのジオメトリを含む MultiPoint を返します。

  結果の空間参照系識別子は、最初のパラメーターの識別子と同じです。

備考

ST_MultiPointAggr 集合関数を使用して、ポイントのグループを 1 つのコレクションに結合することができます。結合するすべてのジオメトリの SRID と座標次元の両方が同じになるようにします。

引数が NULL のローは含まれません。

空のグループ、または NULL 以外の値が含まれないグループの場合、NULL を返します。

結果の ST_MultiPoint の座標次元は、各ポイントの座標次元と同じになります。

ST_GeometryN から目的の順序で要素が返されるように、オプションの ORDER BY 句を使用して要素を特定の順序に並べ替えることができます。この順序が意味を持たない場合は、順序を指定しない方が効率的です。その場合、要素の順序は、クエリオプティマイザーが選択したアクセスプランにより決定されます。

ST_MultiPointAggr は ST_UnionAggr より効率的ですが、ポイントのグループの中に重複するポイントが存在する場合には、そのような重複を伴ったコレクションを返す可能性があります。ST_UnionAggr では、重複するジオメトリを処理できます。

注意
ST_MultiPointAggr では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_UnionAggr メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、SpatialShapes テーブルの ST_Point タイプのすべてのジオメトリを、単一コレクションである ST_MultiPoint タイプに結合した単一の値を返します。Shape カラムが ST_Point タイプの場合は、TREAT 関数と WHERE 句は必要ではありません。

```
SELECT ST_MultiPoint::ST_MultiPointAggr( TREAT( Shape AS ST_Point ) )
  FROM SpatialShapes WHERE Shape IS OF( ST_Point )
```

ST_MultiPolygon タイプ

ST_MultiPolygon は 0、またはそれ以上の ST_Polygon 値のコレクションであり、すべての多角形が空間参照系内にあります。

直接のスーパータイプ

- 「ST_MultiSurface タイプ」

コンストラクター

- 「ST_MultiPolygon コンストラクター」

メソッド

- ST_MultiPolygon のメソッド：

ST_MultiPolygonAggr			

- また、「ST_MultiSurface タイプ」 のすべてのメソッドを ST_MultiPolygon タイプで呼び出すことができます。

ST_Area	ST_Centroid	ST_MultiSurfaceAggr	ST_Perimeter
ST_PointOnSurface			

- また、「ST_GeomCollection タイプ」 のすべてのメソッドを ST_MultiPolygon タイプで呼び出すことができます。

ST_GeomCollectionAggr	ST_GeometryN	ST_NumGeometries	

●また、「ST_Geometry タイプ」 のすべてのメソッドを ST_MultiPolygon タイプで呼び出すことができます。

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

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.6

ST_MultiPolygon コンストラクター

複数多角形を構成します。

オーバーロードリスト

名前	説明
「ST_MultiPolygon() コンストラクター」	空のセットを表す複数多角形を構成します。
「ST_MultiPolygon(LONG VARCHAR[, INT]) コンストラクター」	テキスト表現から複数多角形を構成します。
「ST_MultiPolygon(LONG BINARY[, INT]) コンストラクター」	Well Known Binary (WKB) から複数多角形を構成します。
「ST_MultiPolygon(ST_Polygon,...) コンストラクター」	多角形値のリストから複数多角形を構成します。
「ST_MultiPolygon(ST_MultiLineString[, VARCHAR(128)]) コンストラクター」	外部リングを含む複数線ストリングと、内部リングのオプションリストから複数多角形を作成します。

ST_MultiPolygon() コンストラクター

空のセットを表す複数多角形を構成します。

構文

NEW ST_MultiPolygon()

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 標準機能

例

次の例では、値が空であることを示す 1 を返します。

```
SELECT NEW ST_MultiPolygon().ST_IsEmpty()
```


ST_MultiPolygon(LONG VARCHAR[, INT]) コンストラクター

テキスト表現から複数多角形を構成します。

構文

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

パラメーター

名前	型	説明
text-representation	LONG VARCHAR	複数多角形のテキスト表現を含む文字列。入力には、Well Know Text (WKT) や 拡張 Well Know Text (EWKT) など、サポートされている任意のテキスト入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

文字列表現から複数多角形を構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.6.2

例

次の例では、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))) を返します。

```
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]) コンストラクター

Well Known Binary (WKB) から複数多角形を構成します。

構文

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

パラメーター

名前	型	説明
wkb	LONG BINARY	複数多角形のバイナリ表現を含む文字列。入力には、Well Known Binary (WKB) や 拡張 Well Know Binary (EWKB) など、サポートされている任意のバイナリ入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

バイナリ文字列表現から複数多角形を構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.6.2

例

次の例では、MultiPolygon (((10 -5, 15 5, 5 5, 10 -5))) を返します。

```
SELECT NEW
ST_MultiPolygon(0x010600000001000000010300000001000000040000000000000000000024400000000000000014c00000000000002e400000000000001440000000000000014400000000000001440000000000000244000000000000014c0)
```

ST_MultiPolygon(ST_Polygon,...) コンストラクター

多角形値のリストから複数多角形を構成します。

構文

NEW ST_MultiPolygon(*polygon1*[,*polygon2*,...,*polygonN*])

パラメーター

名前	型	説明
polygon1	ST_Polygon	複数多角形の最初の多角形値。
polygon2,...,polygonN	ST_Polygon	複数多角形の追加の多角形値。

備考

多角形値のリストから複数多角形を構成します。指定するすべての多角形値の SRID を同じにしてください。複数多角形は、この共通 SRID を使用して構成されます。

指定するすべての多角形値が Is3D と IsMeasured に対して同じ回答を示す必要があります。すべての多角形値が 3D の場合に複数多角形も 3D になり、すべての多角形値が測定される場合に複数多角形も測定されます。

注意
ST_MultiPolygon では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として MultiPolygon (((0 0, 1 0, 1 1, 0 1, 0 0))) を返します。

```
SELECT NEW ST_MultiPolygon( NEW ST_Polygon('Polygon ((0 0, 0 1, 1 1, 1 0, 0 0))' ) )
```

次の例では、結果として MultiPolygon (((0 0, 1 0, 1 1, 0 1, 0 0)), ((5 5, 10 5, 10 10, 5 10, 5 5))) を返します。

```
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)]) コンストラクター

外部リングを含む複数線ストリングと、内部リングのオプションリストから複数多角形を作成します。

構文

NEW ST_MultiPolygon(*multi-linestring*[, *polygon-format*])

パラメーター

名前	型	説明
multi-linestring	ST_MultiLineString	外部リングと (オプションの) 一連の内部リングを含む複数線ストリング値。

名前	型	説明
polygon-format	VARCHAR(128)	指定した線ストリングを解釈するときに使用する多角形フォーマットの文字列。有効なフォーマットは、'CounterClockwise'、'Clockwise'、'EvenOdd' です。

備考

外部リングを含む複数線ストリングと、内部リングのオプションリストから複数多角形を作成します。複数線ストリングには、線形リングのみを含めてください。

polygon-format パラメーターを指定すると、リングが外部リングと内部リングのいずれであるかを判断するためにサーバーで使用されるアルゴリズムが選択されます。指定しない場合は、空間参照系の多角形フォーマットが使用されます。

polygon-format の詳細については、POLYGON FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

注意
ST_MultiPolygon では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、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))) (それぞれ三角孔のある 2 つの正方形) を返します。

```
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 メソッド

グループ内のすべての多角形を含む複数多角形を返します。

構文

ST_MultiPolygon::**ST_MultiPolygonAggr**(*geometry-column*[**ORDER BY** *order-by-expression* [**ASC** | **DESC**], ...])

パラメーター

名前	型	説明
geometry-column	ST_Polygon	コレクションを生成するジオメトリ値。通常、これはカラムです。

戻り値

- **ST_MultiPolygon**　グループ内のすべてのジオメトリを含む複数多角形を返します。

  結果の空間参照系識別子は、最初のパラメーターの識別子と同じです。

備考

ST_MultiPolygonAggr 集合関数を使用して、多角形のグループを 1 つのコレクションに結合することができます。結合するすべてのジオメトリの SRID と座標次元の両方が同じになるようにします。

引数が NULL のローは含まれません。

空のグループ、または NULL 以外の値が含まれないグループの場合、NULL を返します。

結果の ST_MultiPolygon の座標次元は、各多角形の座標次元と同じになります。

ST_GeometryN から目的の順序で要素が返されるように、オプションの ORDER BY 句を使用して要素を特定の順序に並べ替えることができます。この順序が意味を持たない場合は、順序を指定しない方が効率的です。その場合、要素の順序は、クエリオプティマイザーが選択したアクセスプランにより決定されます。

ST_MultiPolygonAggr は ST_UnionAggr より効率的ですが、多角形のグループの中に重複する多角形が存在する場合には、そのような重複を伴ったコレクションを返す可能性があります。特に、返されたコレクションに重複する面が含まれている場合に、それが他の空間メソッドの入力値として使用されると、予期しない結果をもたらす場合があります。ST_UnionAggr では、重複するジオメトリを処理できます。

注意
ST_MultiPolygonAggr では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_UnionAggr メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、SpatialShapes テーブルの ST_Polygon タイプのすべてのジオメトリを、単一コレクションである ST_MultiPolygon タイプに結合した単一の値を返します。Shape カラムが ST_Polygon タイプの場合は、TREAT 関数と WHERE 句は必要ではありません。

```
SELECT ST_MultiPolygon::ST_MultiPolygonAggr( TREAT( Shape AS ST_Polygon ) )
  FROM SpatialShapes WHERE Shape IS OF( ST_Polygon )
```

ST_MultiSurface タイプ

ST_MultiSurface は 0 個以上の ST_Surface 値のコレクションであり、すべての面が空間参照系内にあります。

直接のスーパータイプ

- 「ST_GeomCollection タイプ」

直接のサブタイプ

- 「ST_MultiPolygon タイプ」

コンストラクター

- 「ST_MultiSurface コンストラクター」

メソッド

- ST_MultiSurface のメソッド：

ST_Area	ST_Centroid	ST_MultiSurfaceAggr	ST_Perimeter
ST_PointOnSurface			

- また、「ST_GeomCollection タイプ」 のすべてのメソッドを ST_MultiSurface タイプで呼び出すことができます。

ST_GeomCollectionAggr	ST_GeometryN	ST_NumGeometries	

- また、「ST_Geometry タイプ」 のすべてのメソッドを ST_MultiSurface タイプで呼び出すことができます。

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

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.5

ST_MultiSurface コンストラクター

複数面を構成します。

オーバーロードリスト

名前	説明
「ST_MultiSurface() コンストラクター」	空のセットを表す複数面を構成します。
「ST_MultiSurface(LONG VARCHAR[, INT]) コンストラクター」	テキスト表現から複数面を構成します。
「ST_MultiSurface(LONG BINARY[, INT]) コンストラクター」	Well Known Binary (WKB) から複数面を構成します。
「ST_MultiSurface(ST_Surface,...) コンストラクター」	面の値のリストから複数面を構成します。
「ST_MultiSurface(ST_MultiCurve[, VARCHAR(128)]) コンストラクター」	外部リングを含む複数曲線と、内部リングのオプションリストから複数面を作成します。

ST_MultiSurface() コンストラクター

空のセットを表す複数面を構成します。

構文

NEW ST_MultiSurface()

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 標準機能

例

次の例では、値が空であることを示す 1 を返します。

```
SELECT NEW ST_MultiSurface().ST_IsEmpty()
```

ST_MultiSurface(LONG VARCHAR[, INT]) コンストラクター

テキスト表現から複数面を構成します。

構文

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

パラメーター

名前	型	説明
text-representation	LONG VARCHAR	複数面のテキスト表現を含む文字列。入力には、Well Know Text (WKT) や 拡張 Well Know Text (EWKT) など、サポートされている任意のテキスト入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

文字列表現から複数面を構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.5.2

例

次の例では、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))) を返します。

```
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]) コンストラクター

Well Known Binary (WKB) から複数面を構成します。

構文

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

パラメーター

名前	型	説明
wkb	LONG BINARY	複数面のバイナリ表現を含む文字列。入力には、Well Known Binary (WKB) や 拡張 Well Know Binary (EWKB) など、サポートされている任意のバイナリ入力フォーマットを使用できます。

名前	型	説明
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

バイナリ文字列表現から複数面を構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.5.2

例

次の例では、MultiSurface (CurvePolygon (CircularString (0 0, 10 0, 10 10, 0 10, 0 0))) を返します。

```
SELECT NEW
ST_MultiSurface(0x010c00000001000000010a00000001000000010800000005000000000000000000000000
0000000000000000000000000000002440000000000000000000000000002440000000000000002440
000000000000000000000000000024400000000000000000000000000000000000)
```

ST_MultiSurface(ST_Surface,...) コンストラクター

面の値のリストから複数面を構成します。

構文

NEW ST_MultiSurface(*surface1*[,*surface2*,...,*surfaceN*])

パラメーター

名前	型	説明
surface1	ST_Surface	複数面の最初の面の値。
surface2,...,surfaceN	ST_Surface	複数面の追加の面の値。

備考

面の値のリストから複数面を構成します。指定するすべての面の値の SRID を同じにしてください。複数面は、この共通 SRID を使用して構成されます。

指定するすべての面の値が Is3D と IsMeasured に対して同じ回答を示す必要があります。すべての面の値が 3D の場合に複数面も 3D になり、すべての面の値が測定される場合に複数面も測定されます。

注意
ST_MultiSurface では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として MultiSurface (((0 0, 1 0, 1 1, 0 1, 0 0))) を返します。

```
SELECT NEW ST_MultiSurface( NEW ST_Polygon('Polygon ((0 0, 0 1, 1 1, 1 0, 0 0))' ) )
```

次の例では、結果として MultiSurface (((0 0, 1 0, 1 1, 0 1, 0 0)), ((5 5, 10 5, 10 10, 5 10, 5 5))) を返します。

```
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)]) コンストラクター

外部リングを含む複数曲線と、内部リングのオプションリストから複数面を作成します。

構文

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

パラメーター

名前	型	説明
multi-curve	ST_MultiCurve	外部リングと (オプションの) 一連の内部リングを含む複数曲線値。
polygon-format	VARCHAR(128)	指定した曲線を解釈するときに使用する多角形フォーマットの文字列。有効なフォーマットは、'CounterClockwise'、'Clockwise'、'EvenOdd' です。

備考

外部リングを含む複数曲線と、内部リングのオプションリストから複数面を作成します。複数曲線には、任意の曲線タイプを含めることができます。

polygon-format パラメーターを指定すると、リングが外部リングと内部リングのいずれであるかを判断するためにサーバーで使用されるアルゴリズムが選択されます。指定しない場合は、空間参照系の多角形フォーマットが使用されます。

polygon-format の詳細については、POLYGON FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

注意
ST_MultiSurface では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、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))) を返します。

```
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 メソッド

指定した単位で複数面の面積を計算します。

構文

multisurface-expression.**ST_Area**([*unit-name*])

パラメーター

名前	型	説明
unit-name	VARCHAR(128)	面積を計算する単位。デフォルトでは、空間参照系の単位が使用されます。単位名は、UNIT_TYPE が 'LINEAR' の ST_UNITS_OF_MEASURE ビュー内のローの UNIT_NAME カラムと一致させてください。

戻り値

- **DOUBLE** 複数面の面積を返します。

備考

指定した単位で複数面の面積を計算します。複数面の面積は、含まれている面の面積の合計です。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

- ST_MultiSurface タイプの ST_Perimeter メソッド
- ST_Surface タイプの ST_Area メソッド
- ST_MultiCurve タイプの ST_Length メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.5.3

例

次の例では、結果として 8 を返します。

```
SELECT TREAT( Shape AS ST_MultiSurface ).ST_Area()
FROM SpatialShapes WHERE ShapeID = 27
```

次の例では、架空の region テーブルから、multipoly_geometry カラムの領域を平方マイルで返します。

```
SELECT name, multipoly_geometry.ST_Area( 'Statute Mile' )
FROM region
```

ST_Centroid メソッド

複数面の数学的重心である ST_Point を計算します。

構文

multisurface-expression.**ST_Centroid**()

戻り値

- **ST_Point** 複数面が空のセットの場合は、NULL を返します。それ以外の場合は、面の数学的重心を返します。

  結果の空間参照系識別子は、*multisurface-expression* の空間参照系と同じです。

備考

複数面の数学的重心である ST_Point を計算します。このポイントは必ずしも面上のポイントにはならないことに注意してください。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

- ST_Surface タイプの ST_Centroid メソッド
- ST_MultiSurface タイプの ST_PointOnSurface メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 9.5.5

例

次の例では、結果として Point (1.865682 .664892) を返します。

```
SELECT TREAT( Shape AS ST_MultiSurface ).ST_Centroid()
FROM SpatialShapes WHERE ShapeID = 28
```

ST_MultiSurfaceAggr メソッド

グループ内のすべての面を含む複数面を返します。

構文

ST_MultiSurface::**ST_MultiSurfaceAggr**(*geometry-column*[**ORDER BY** *order-by-expression* [**ASC** | **DESC**], ...])

パラメーター

名前	型	説明
geometry-column	ST_Surface	コレクションを生成するジオメトリ値。通常、これはカラムです。

戻り値

- **ST_MultiSurface** グループ内のすべてのジオメトリを含む複数面を返します。

  結果の空間参照系識別子は、最初のパラメーターの識別子と同じです。

備考

ST_MultiSurfaceAggr 集合関数を使用して、面のグループを 1 つのコレクションに結合することができます。結合するすべてのジオメトリの SRID と座標次元の両方が同じになるようにします。

引数が NULL のローは含まれません。

空のグループ、または NULL 以外の値が含まれないグループの場合、NULL を返します。

結果の ST_MultiSurface の座標次元は、各面の座標次元と同じになります。

ST_GeometryN から目的の順序で要素が返されるように、オプションの ORDER BY 句を使用して要素を特定の順序に並べ替えることができます。この順序が意味を持たない場合は、順序を指

定しない方が効率的です。その場合、要素の順序は、クエリオプティマイザーが選択したアクセスプランにより決定されます。

ST_MultiSurfaceAggrr は ST_UnionAggr より効率的ですが、曲のグループの中に重複する曲が存在する場合には、そのような重複を伴ったコレクションを返す可能性があります。特に、返されたコレクションに重複する面が含まれている場合に、それが他の空間メソッドの入力値として使用されると、予期しない結果をもたらす場合があります。ST_UnionAggr では、重複するジオメトリを処理できます。

注意
ST_MultiSurfaceAggr では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_UnionAggr メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、SpatialShapes テーブルの ST_Surface タイプのすべてのジオメトリを、単一コレクションである ST_MultiSurface タイプに結合した単一の値を返します。Shape カラムが ST_Surface タイプの場合は、TREAT 関数と WHERE 句は必要ではありません。

```
SELECT ST_MultiSurface::ST_MultiSurfaceAggr( TREAT( Shape AS ST_Surface ) )
  FROM SpatialShapes WHERE Shape IS OF( ST_Surface )
```

ST_Perimeter メソッド

指定した単位で複数面の周囲の長さを計算します。

構文

multisurface-expression.**ST_Perimeter**([*unit-name*])

パラメーター

名前	型	説明
unit-name	VARCHAR(128)	周囲の長さを計算する単位。デフォルトでは、空間参照系の単位が使用されます。単位名は、UNIT_TYPE が 'LINEAR' の ST_UNITS_OF_MEASURE ビュー内のローの UNIT_NAME カラムと一致させてください。

戻り値

- **DOUBLE**　複数面の周囲の長さを返します。

備考

ST_Perimeter メソッドは、*unit-name* パラメーターで指定された単位で複数面の周囲の長さを返します。複数面が空の場合は、NULL が返されます。

複数面に Z 値が含まれている場合、それらの値はジオメトリの周囲の長さの計算時には考慮されません。

多角形の周囲の長さには、すべてのリング (外部と内部) の長さが含まれます。

注意
multisurface-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_Perimeter では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Surface タイプの ST_Perimeter メソッド
- ST_Geometry タイプの ST_Boundary メソッド
- ST_MultiCurve タイプの ST_Length メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.5.4

例

次の例では、2 つの多角形を含む複数面を作成し、ST_Perimeter を使用して周囲の長さを調べ、結果として 44 を返します。

```
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()
```

次の例では、2 つの多角形を含む複数面と測定単位の例 (example_unit_halfmetre) を作成します。ST_Perimeter は、周囲の長さを調べ、値 88.0 を返します。

```
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 メソッド

複数面内の面上にあることが保証されるポイントを返します。

構文

multisurface-expression.**ST_PointOnSurface**()

戻り値

- **ST_Point**　複数面が空のセットの場合は、NULL を返します。それ以外の場合は、ST_MultiSurface 値と空間的に交差することが保証される ST_Point 値を返します。

  結果の空間参照系識別子は、*multisurface-expression* の空間参照系と同じです。

備考

複数面のいずれかの面の内部にあるポイントを返します。

> **注意**
> *multisurface-expression* に円ストリングが含まれている場合、それらは線ストリングに補間されます。

参照

- ST_Surface タイプの ST_PointOnSurface メソッド
- ST_MultiSurface タイプの ST_Centroid メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.5.6

例

次の例では、複数面と交差するポイントを返します。

```
SELECT TREAT( Shape AS ST_MultiSurface ).ST_PointOnSurface()
FROM SpatialShapes WHERE ShapeID = 27
```

ST_Point タイプ

ST_Point タイプは、0 次元のジオメトリであり、1 つのロケーションを表します。

直接のスーパータイプ

- 「ST_Geometry タイプ」

コンストラクター

- 「ST_Point コンストラクター」

メソッド

- ST_Point のメソッド：

ST_Lat	ST_Long	ST_M	ST_X
ST_Y	ST_Z		

- また、「ST_Geometry タイプ」 のすべてのメソッドを ST_Point タイプで呼び出すことができます。

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

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1

ST_Point コンストラクター

ポイントを構成します。

注意
座標から ST_Point 値を作成する場合、取得されるオーバーロードは必ずしも予測できるとはかぎりません。たとえば、式 "NEW ST_Point(1,2,3)" は、X=1、Y=2、SRID=3 の 2D ポイントを作成します。式 "NEW ST_Point(1,2,3.0)" は、Z=3.0 の 3D ポイントを作成します。

オーバーロードリスト

名前	説明
「ST_Point() コンストラクター」	空のセットを表すポイントを構成します。

名前	説明
「ST_Point(LONG VARCHAR[, INT]) コンストラクター」	テキスト表現からポイントを構成します。
「ST_Point(LONG BINARY[, INT]) コンストラクター」	Well Known Binary (WKB) からポイントを構成します。
「ST_Point(DOUBLE,DOUBLE[, INT]) コンストラクター」	(X, Y) 座標から 2D ポイントを構成します。
「ST_Point(DOUBLE,DOUBLE,DOUBLE[, INT]) コンストラクター」	(X, Y, Z) 座標から 3D ポイントを構成します。
「ST_Point(DOUBLE,DOUBLE,DOUBLE,DOUBLE[, INT]) コンストラクター」	(X, Y, Z) 座標と測定値から 3D の測定ポイントを構成します。

ST_Point() コンストラクター

空のセットを表すポイントを構成します。

構文

NEW ST_Point()

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 標準機能

例

次の例では、値が空であることを示す 1 を返します。

```
SELECT NEW ST_Point().ST_IsEmpty()
```

ST_Point(LONG VARCHAR[, INT]) コンストラクター

テキスト表現からポイントを構成します。

構文

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

パラメーター

名前	型	説明
text-representation	LONG VARCHAR	ポイントのテキスト表現を含む文字列。入力には、Well Know Text (WKT) や 拡張 Well Know Text (EWKT) など、サポートされている任意のテキスト入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

文字列表現からポイントを構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.2

例

次の例では、Point (10 20) を返します。

```
SELECT NEW ST_Point('Point (10 20)')
```

ST_Point(LONG BINARY[, INT]) コンストラクター

Well Known Binary (WKB) からポイントを構成します。

構文

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

パラメーター

名前	型	説明
wkb	LONG BINARY	ポイントのバイナリ表現を含む文字列。入力には、Well Known Binary (WKB) や 拡張 Well Know Binary (EWKB) など、サポートされている任意のバイナリ入力フォーマットを使用できます。

名前	型	説明
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

バイナリ文字列表現からポイントを構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.2

例

次の例では、Point (10 20) を返します。

```
SELECT NEW ST_Point(0x01010000000000000000002440000000000000003440)
```

ST_Point(DOUBLE,DOUBLE[, INT]) コンストラクター

(X, Y) 座標から 2D ポイントを構成します。

構文

NEW ST_Point(*x*,*y*[, *srid*])

パラメーター

名前	型	説明
x	DOUBLE	X 座標値。
y	DOUBLE	Y 座標値。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.2

例

次の例では、Point (10 20) を返します。

```
SELECT NEW ST_Point(10.0,20.0,0)
```

ST_Point(DOUBLE,DOUBLE,DOUBLE[, INT]) コンストラクター

(X, Y, Z) 座標から 3D ポイントを構成します。

構文

NEW ST_Point(*x*,*y*,*z*[, *srid*])

パラメーター

名前	型	説明
x	DOUBLE	X 座標値。
y	DOUBLE	Y 座標値。
z	DOUBLE	Z 座標値。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.2

例

次の例では、Point Z (10 20 100) を返します。

```
SELECT NEW ST_Point(10.0,20.0,100.0,0)
```

ST_Point(DOUBLE,DOUBLE,DOUBLE,DOUBLE[, INT]) コンストラクター

(X, Y, Z) 座標と測定値から 3D の測定ポイントを構成します。

構文

NEW ST_Point(*x*,*y*,*z*,*m*[, *srid*])

パラメーター

名前	型	説明
x	DOUBLE	X 座標値。
y	DOUBLE	Y 座標値。
z	DOUBLE	Z 座標値。
m	DOUBLE	測定値。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.2

例

次の例では、Point ZM (10 20 100 1224) を返します。

```
SELECT NEW ST_Point(10.0,20.0,100.0,1224.0,0)
```

ST_Lat メソッド

ST_Point 値の緯度座標を返します。

オーバーロードリスト

名前	説明
「ST_Point タイプの ST_Lat() メソッド」	ST_Point 値の緯度座標を返します。
「ST_Point タイプの ST_Lat(DOUBLE) メソッド」	緯度座標が指定した緯度値に設定されたポイントのコピーを返します。

ST_Point タイプの ST_Lat() メソッド

ST_Point 値の緯度座標を返します。

注意
point-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_Lat では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

point-expression.**ST_Lat**()

戻り値

- **DOUBLE** ST_Point 値の緯度座標を返します。

参照

- ST_Point タイプの ST_Long メソッド
- ST_Point タイプの ST_Y メソッド
- ST_Geometry タイプの ST_LatNorth メソッド
- ST_Geometry タイプの ST_LatSouth メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、0 で識別された空間座標系は地理的空間座標系ではないため、エラーが発生します。

```
SELECT NEW ST_Point( 10.0, 20.0, 0 ).ST_Lat()
```

次の例では、結果として 20.0 を返します。

```
SELECT NEW ST_Point( 10.0, 20.0, 4326 ).ST_Lat()
```

ST_Point タイプの ST_Lat(DOUBLE) メソッド

緯度座標が指定した緯度値に設定されたポイントのコピーを返します。

注意
point-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_Lat では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

point-expression.**ST_Lat**(*latitude-val*)

パラメーター

名前	型	説明
latitude-val	DOUBLE	新しい緯度値。

戻り値

- **ST_Point**　緯度が指定した値に設定されたポイントのコピーを返します。

  結果の空間参照系識別子は、*point-expression* の空間参照系と同じです。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

ST_Long メソッド

ST_Point 値の経度座標を返します。

オーバーロードリスト

名前	説明
「ST_Point タイプの ST_Long() メソッド」	ST_Point 値の経度座標を返します。
「ST_Point タイプの ST_Long(DOUBLE) メソッド」	経度座標が指定した経度値に設定されたポイントのコピーを返します。

ST_Point タイプの ST_Long() メソッド

ST_Point 値の経度座標を返します。

注意
point-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_Long では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

point-expression.**ST_Long**()

戻り値

- **DOUBLE** ST_Point 値の経度座標を返します。

参照

- ST_Point タイプの ST_Lat メソッド
- ST_Point タイプの ST_X メソッド
- ST_Geometry タイプの ST_LongEast メソッド
- ST_Geometry タイプの ST_LongWest メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、0 で識別された空間座標系は地理的空間座標系ではないため、エラーが発生します。

```
SELECT NEW ST_Point( 10.0, 20.0, 0 ).ST_Long()
```

次の例では、結果として 10.0 を返します。

```
SELECT NEW ST_Point( 10.0, 20.0, 4326 ).ST_Long()
```

ST_Point タイプの ST_Long(DOUBLE) メソッド

経度座標が指定した経度値に設定されたポイントのコピーを返します。

注意
point-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_Long では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

point-expression.**ST_Long**(*longitude-val*)

パラメーター

名前	型	説明
longitude-val	DOUBLE	新しい経度値。

戻り値

- **ST_Point**　経度が指定した値に設定されたポイントのコピーを返します。

  結果の空間参照系識別子は、*point-expression* の空間参照系と同じです。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

ST_M メソッド

ポイントの測定値を取り出したり、変更したりします。

オーバーロードリスト

名前	説明
「ST_Point タイプの ST_M() メソッド」	ST_Point 値の測定値を返します。
「ST_Point タイプの ST_M(DOUBLE) メソッド」	測定値が指定した mcoord 値に設定されたポイントのコピーを返します。

ST_Point タイプの ST_M() メソッド

ST_Point 値の測定値を返します。

注意
point-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_M では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

point-expression.**ST_M**()

戻り値

- **DOUBLE** ST_Point 値の測定値を返します。

参照

- ST_Point タイプの ST_X メソッド
- ST_Point タイプの ST_Y メソッド
- ST_Geometry タイプの ST_MMin メソッド
- ST_Geometry タイプの ST_MMax メソッド
- ST_Geometry タイプの ST_IsMeasured メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.6

例

次の例では、結果として 40.0 を返します。

```
SELECT NEW ST_Point( 10.0, 20.0, 30.0, 40.0, 0 ).ST_M()
```


ST_Point タイプの ST_M(DOUBLE) メソッド

測定値が指定した mcoord 値に設定されたポイントのコピーを返します。

注意
point-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_M では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

point-expression.**ST_M**(*mcoord*)

パラメーター

名前	型	説明
mcoord	DOUBLE	新しい測定値。

戻り値

- **ST_Point**　測定値が指定した mcoord 値に設定されたポイントのコピーを返します。

  結果の空間参照系識別子は、*point-expression* の空間参照系と同じです。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　6.1.6

例

次の例では、結果として Point ZM (1 2 3 5) を返します。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0, 4.0 ).ST_M( 5.0 )
```

ST_X メソッド

ポイントの X 座標値を取り出したり、変更したりします。

オーバーロードリスト

名前	説明
「ST_Point タイプの ST_X() メソッド」	ST_Point 値の X 座標を返します。

名前	説明
「ST_Point タイプの ST_X(DOUBLE) メソッド」	X 座標が指定した xcoord 値に設定されたポイントのコピーを返します。

ST_Point タイプの ST_X() メソッド

ST_Point 値の X 座標を返します。

注意
point-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_X では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

point-expression.**ST_X**()

戻り値

- **DOUBLE** ST_Point 値の X 座標を返します。

参照

- ST_Point タイプの ST_Y メソッド
- ST_Point タイプの ST_Lat メソッド
- ST_Geometry タイプの ST_XMin メソッド
- ST_Geometry タイプの ST_XMax メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.3

例

次の例では、結果として 10.0 を返します。

```
SELECT NEW ST_Point( 10.0, 20.0, 30.0, 40.0, 0 ).ST_X()
```

ST_Point タイプの ST_X(DOUBLE) メソッド

X 座標が指定した xcoord 値に設定されたポイントのコピーを返します。

注意
point-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_X では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

point-expression.**ST_X**(*xcoord*)

パラメーター

名前	型	説明
xcoord	DOUBLE	新しい X 座標値。

戻り値

- **ST_Point** X 座標が指定した xcoord 値に設定されたポイントのコピーを返します。

  結果の空間参照系識別子は、*point-expression* の空間参照系と同じです。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.3

ST_Y メソッド

ポイントの Y 座標値を取り出したり、変更したりします。

オーバーロードリスト

名前	説明
「ST_Point タイプの ST_Y() メソッド」	ST_Point 値の Y 座標を返します。
「ST_Point タイプの ST_Y(DOUBLE) メソッド」	Y 座標が指定した ycoord 値に設定されたポイントのコピーを返します。

ST_Point タイプの ST_Y() メソッド

ST_Point 値の Y 座標を返します。

注意
point-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_Y では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

point-expression.**ST_Y**()

戻り値

- **DOUBLE** ST_Point 値の Y 座標を返します。

参照

- ST_Point タイプの ST_X メソッド
- ST_Point タイプの ST_Long メソッド
- ST_Geometry タイプの ST_YMin メソッド
- ST_Geometry タイプの ST_YMax メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.4

例

次の例では、結果として 20.0 を返します。

```
SELECT NEW ST_Point( 10.0, 20.0, 30.0, 40.0, 0 ).ST_Y()
```

ST_Point タイプの ST_Y(DOUBLE) メソッド

Y 座標が指定した ycoord 値に設定されたポイントのコピーを返します。

注意
point-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_Y では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

point-expression.**ST_Y**(*ycoord*)

パラメーター

名前	型	説明
ycoord	DOUBLE	新しい Y 座標値。

戻り値

- **ST_Point**　Y 座標が指定した ycoord 値に設定されたポイントのコピーを返します。

  結果の空間参照系識別子は、*point-expression* の空間参照系と同じです。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　6.1.4

例

次の例では、結果として Point (1 3) を返します。

```
SELECT NEW ST_Point( 1, 2 ).ST_Y( 3 )
```

ST_Z メソッド

ポイントの Z 座標値を取り出したり、変更したりします。

オーバーロードリスト

名前	説明
「ST_Point タイプの ST_Z() メソッド」	ST_Point 値の Z 座標を返します。
「ST_Point タイプの ST_Z(DOUBLE) メソッド」	Z 座標が指定した zcoord 値に設定されたポイントのコピーを返します。

ST_Point タイプの ST_Z() メソッド

ST_Point 値の Z 座標を返します。

注意
point-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_Z では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

point-expression.**ST_Z**()

戻り値

- **DOUBLE**　ST_Point 値の Z 座標を返します。

参照

- ST_Point タイプの ST_X メソッド
- ST_Point タイプの ST_Y メソッド
- ST_Geometry タイプの ST_ZMin メソッド
- ST_Geometry タイプの ST_ZMax メソッド
- ST_Geometry タイプの ST_Is3D メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　6.1.4

例

次の例では、結果として 30.0 を返します。

```
SELECT NEW ST_Point( 10.0, 20.0, 30.0, 40.0, 0 ).ST_Z()
```

ST_Point タイプの ST_Z(DOUBLE) メソッド

Z 座標が指定した zcoord 値に設定されたポイントのコピーを返します。

注意
point-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_Z では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

point-expression.**ST_Z**(*zcoord*)

パラメーター

名前	型	説明
zcoord	DOUBLE	新しい Z 座標値。

戻り値

- **ST_Point**　Z 座標が指定した zcoord 値に設定されたポイントのコピーを返します。

結果の空間参照系識別子は、*point-expression* の空間参照系と同じです。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 6.1.5

例

次の例では、結果として Point Z (1 2 5) を返します。

```
SELECT NEW ST_Point( 1.0, 2.0, 3.0 ).ST_Z( 5.0 )
```

ST_Polygon タイプ

ST_Polygon は、線形リングである内部リングと外部リングで形成される ST_CurvePolygon です。

直接のスーパータイプ

- 「ST_CurvePolygon タイプ」

コンストラクター

- 「ST_Polygon コンストラクター」

メソッド

●ST_Polygon のメソッド：

ST_ExteriorRing	ST_InteriorRingN		

●また、「ST_CurvePolygon タイプ」 のすべてのメソッドを ST_Polygon タイプで呼び出すことができます。

ST_CurvePolyToPoly	ST_ExteriorRing	ST_InteriorRingN	ST_NumInteriorRing

●また、「ST_Surface タイプ」 のすべてのメソッドを ST_Polygon タイプで呼び出すことができます。

ST_Area	ST_Centroid	ST_IsWorld	ST_Perimeter
ST_PointOnSurface			

●また、「ST_Geometry タイプ」 のすべてのメソッドを ST_Polygon タイプで呼び出すことができます。

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

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.3

ST_Polygon コンストラクター

多角形を構成します。

オーバーロードリスト

名前	説明
「ST_Polygon() コンストラクター」	空のセットを表す多角形を構成します。
「ST_Polygon(LONG VARCHAR[, INT]) コンストラクター」	テキスト表現から多角形を構成します。
「ST_Polygon(LONG BINARY[, INT]) コンストラクター」	Well Known Binary (WKB) から多角形を構成します。
「ST_Polygon(ST_Point,ST_Point) コンストラクター」	左下角と右上角を表す 2 つのポイントから、軸と平行の長方形を作成します。
「ST_Polygon(ST_MultiLineString[, VARCHAR(128)]) コンストラクター」	外部リングを含む複数線ストリングと、内部リングのオプションリストから多角形を作成します。
「ST_Polygon(ST_LineString,...) コンストラクター」	外部リングを表す線ストリングと、内部リングを表す線ストリングのオプションリストから多角形を作成します。

ST_Polygon() コンストラクター

空のセットを表す多角形を構成します。

構文

NEW ST_Polygon()

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 標準機能

例

次の例では、値が空であることを示す 1 を返します。

```
SELECT NEW ST_Polygon().ST_IsEmpty()
```

ST_Polygon(LONG VARCHAR[, INT]) コンストラクター

テキスト表現から多角形を構成します。

構文

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

パラメーター

名前	型	説明
text-representation	LONG VARCHAR	多角形のテキスト表現を含む文字列。入力には、Well Know Text (WKT) や 拡張 Well Know Text (EWKT) など、サポートされている任意のテキスト入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

文字列表現から多角形を構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.3.2

例

次の例では、Polygon ((-5 -5, 5 -5, 0 5, -5 -5), (-2 -2, -2 0, 2 0, 2 -2, -2 -2)) を返します。

```
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]) コンストラクター

Well Known Binary (WKB) から多角形を構成します。

構文

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

パラメーター

名前	型	説明
wkb	LONG BINARY	多角形のバイナリ表現を含む文字列。入力には、Well Known Binary (WKB) や 拡張 Well Know Binary (EWKB) など、サポートされている任意のバイナリ入力フォーマットを使用できます。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

備考

バイナリ文字列表現から多角形を構成します。データベースサーバーでは、指定された文字列を検査して入力フォーマットを判断します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.3.2

例

次の例では、Polygon ((10 -5, 15 5, 5 5, 10 -5)) を返します。

```
SELECT NEW
ST_Polygon(0x0103000000010000000400000000000000000024400000000000000014c0000000000000
2e4000000000000001440000000000000014400000000000000144000000000000002440000000000000014c
0)
```

ST_Polygon(ST_Point,ST_Point) コンストラクター

左下角と右上角を表す 2 つのポイントから、軸と平行の長方形を作成します。

構文

NEW ST_Polygon(*pmin*,*pmax*)

パラメーター

名前	型	説明
pmin	ST_Point	長方形の左下角のポイント。
pmax	ST_Point	長方形の右上角のポイント。

備考

2 つのポイントの包絡線として定義された長方形を返します。

コンストラクターは、 NEW ST_MultiPoint(pmin, pmax).ST_Envelope() と同等です。

対応する長方形の角が入力ポイントに指定されていない場合は、サーバーは指定されたポイントを含む正しい長方形を決定します。

注意
入力ポイントが曲面の空間参照系の場合は、生成される多角形は軸と平行の長方形にはなりません。西の境界と東の境界に関しては、それぞれ対になった東のポイントと西のポイントを結ぶ経線と一致していますが、北のエッジと南のエッジは平行線にはならず、エッジとポイントを含む大きな楕円形の弧を描きます。また、サーバーは可能な場合には常に有効な多角形を生成します。そのような有効な多角形が日付変更線と交差した場合は、西のポイントの経度は東のポイントの経度より高くなり、日付変更線と交差しない多角形の場合と逆の関係が成立します。

注意
ST_Polygon では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、Polygon ((0 0, 4 0, 4 10, 0 10, 0 0)) を返します。

```
SELECT NEW ST_Polygon(NEW ST_Point(0.0, 0.0), NEW ST_Point(4.0, 10.0))
```

ST_Polygon(ST_MultiLineString[, VARCHAR(128)]) コンストラクター

外部リングを含む複数線ストリングと、内部リングのオプションリストから多角形を作成します。

構文

NEW ST_Polygon(*multi-linestring*[, *polygon-format*])

パラメーター

名前	型	説明
multi-linestring	ST_MultiLineString	外部リングと (オプションの) 一連の内部リングを含む複数線ストリング値。

名前	型	説明
polygon-format	VARCHAR(128)	指定した線ストリングを解釈するときに使用する多角形フォーマットの文字列。有効なフォーマットは、'CounterClockwise'、'Clockwise'、'EvenOdd' です。

備考

外部リングを含む複数線ストリングと、内部リングのオプションリストから多角形を作成します。複数線ストリングには、線形リングのみを含めてください。

polygon-format パラメーターを指定すると、リングが外部リングと内部リングのいずれであるかを判断するためにサーバーで使用されるアルゴリズムが選択されます。指定しない場合は、空間参照系の多角形フォーマットが使用されます。

polygon-format の詳細については、POLYGON FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

注意
ST_Polygon では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.3.2

例

次の例では、Polygon ((-5 -1, 5 -1, 0 9, -5 -1), (-2 0, 0 4, 2 0, -2 0)) (三角孔のある三角形) を返します。

```
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,...) コンストラクター

外部リングを表す線ストリングと、内部リングを表す線ストリングのオプションリストから多角形を作成します。

構文

NEW ST_Polygon(*exterior-ring*[,*interior-ring1*,...,*interior-ringN*])

パラメーター

名前	型	説明
exterior-ring	ST_LineString	多角形の外部リング
interior-ring1,...,interior-ringN	ST_LineString	多角形の内部リング

備考

外部リングを表す線ストリングと、内部リングを表す線ストリングのリスト (空の可能性もある) から多角形を作成します。指定するすべての線ストリング値の SRID を同じにしてください。結果の多角形は、この共通 SRID を使用して構成されます。

指定するすべての線ストリングが空ではなく、Is3D と IsMeasured に対して同じ回答を示す必要があります。すべての線ストリングが 3D の場合に多角形も 3D になり、すべての線ストリングが測定される場合に多角形も測定されます。

注意
ST_Polygon では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

標準と互換性

内部リングの可変長リストを指定する機能はベンダー拡張です。

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.3.2

例

次の例では、Polygon ((-5 -1, 5 -1, 0 9, -5 -1), (-2 0, 0 4, 2 0, -2 0)) (三角孔のある三角形) を返します。

```
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 メソッド

外部リングを取り出したり、変更したりします。

オーバーロードリスト

名前	説明
「ST_Polygon タイプの ST_ExteriorRing() メソッド」	多角形の外部リングを返します。

名前	説明
「ST_Polygon タイプの ST_ExteriorRing(ST_Curve) メソッド」	多角形の外部リングを変更します。

ST_Polygon タイプの ST_ExteriorRing() メソッド

多角形の外部リングを返します。

注意
ST_ExteriorRing では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

polygon-expression.**ST_ExteriorRing**()

戻り値

- **ST_LineString**　多角形の外部リングを返します。

  結果の空間参照系識別子は、*polygon-expression* の空間参照系と同じです。

参照

- ST_Polygon タイプの ST_InteriorRingN メソッド
- ST_CurvePolygon タイプの ST_ExteriorRing メソッド
- ST_Geometry タイプの ST_Boundary メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　8.3.3

例

次の例では、結果として LineString (0 0, 10 0, 5 10, 0 0) を返します。

```
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_Polygon タイプの ST_ExteriorRing(ST_Curve) メソッド

多角形の外部リングを変更します。

注意
ST_ExteriorRing では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

polygon-expression.**ST_ExteriorRing**(*curve*)

パラメーター

名前	型	説明
curve	ST_Curve	多角形の新しい外部リング。これは線形リング値にしてください。

戻り値

- **ST_Polygon**　指定した外部リングを含む多角形のコピーを返します。

  結果の空間参照系識別子は、*polygon-expression* の空間参照系と同じです。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　8.3.3

例

次の例では、結果として Polygon ((0 1, 10 1, 5 10, 0 1), (3 3, 3 5, 7 5, 7 3, 3 3)) を返します。

```
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 メソッド

多角形の *n* 番目の内部リングを返します。

注意
ST_InteriorRingN では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

構文

polygon-expression.**ST_InteriorRingN**(*n*)

パラメーター

名前	型	説明
n	INT	返す要素の位置 (1 ～ *polygon-expression*.ST_NumInteriorRing())。

戻り値

- **ST_LineString**　多角形の *n* 番目の内部リングを返します。

  結果の空間参照系識別子は、*polygon-expression* の空間参照系と同じです。

参照

- ST_CurvePolygon タイプの ST_NumInteriorRing メソッド
- ST_Polygon タイプの ST_ExteriorRing メソッド
- ST_CurvePolygon タイプの ST_InteriorRingN メソッド
- ST_Geometry タイプの ST_Boundary メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　8.3.5

例

次の例では、結果として LineString (3 3, 3 5, 7 5, 7 3, 3 3) を返します。

```
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 タイプ

ST_SpatialRefSys タイプは、空間参照系を操作するためのルーチンを定義します。

メソッド

●ST_SpatialRefSys のメソッド：

ST_CompareWKT	ST_FormatTransformDefinition	ST_FormatWKT	ST_GetUnProjectedTransformDefinition
ST_ParseWKT	ST_TransformGeom	ST_World	

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　13.1

ST_CompareWKT メソッド

2 つの空間参照系定義を比較します。

構文

ST_SpatialRefSys::**ST_CompareWKT**(*transform-definition-1*,*transform-definition-2*)

パラメーター

名前	型	説明
transform-definition-1	LONG VARCHAR	最初の空間参照系の定義テキスト
transform-definition-2	LONG VARCHAR	2 番目の空間参照系の定義テキスト

戻り値

- **BIT**　2 つの空間参照系が論理的に同等の場合は 1 を返し、それ以外の場合は 0 を返します。

備考

2 つの空間参照系 (WKT で定義) が論理的に同等かどうかを調べます。2 つの空間参照系が同じ権限によって同じ識別子を使用して定義されている場合、または文字列が完全に等しい場合、これらの空間参照系は論理的に等しいと見なされます。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例は、名前の異なる 2 つの空間参照系でも等しいと見なされることを示します。

```
SELECT ST_SpatialRefSys::ST_CompareWKT(
   'GEOGCS["WGS 84",DATUM["WGS_1984",SPHEROID["WGS 84",
6378137,298.257223563,AUTHORITY["EPSG","7030"]],AUTHORITY["EPSG","6326"]],PRIMEM["Green
wich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",
0.0174532925199433,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["Green
wich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",
0.0174532925199433,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG","4326"]]'
) Considered_Equal
```

次の例は、異なる権限によって定義されているために等しくないと見なされる 2 つの空間参照系を示します。

```
SELECT ST_SpatialRefSys::ST_CompareWKT(
   'GEOGCS["WGS 84",DATUM["WGS_1984",SPHEROID["WGS 84",
6378137,298.257223563,AUTHORITY["EPSG","7030"]],AUTHORITY["EPSG","6326"]],PRIMEM["Green
wich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",
0.0174532925199433,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["Green
wich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",
0.01745329251994328,AUTHORITY["EPSG","9122"]],AUTHORITY["AnotherAuthority","4326"]]'
) Considered_NotEqual
```

ST_FormatTransformDefinition メソッド

変換定義のフォーマット済みコピーを返します。

構文

ST_SpatialRefSys::**ST_FormatTransformDefinition**(*transform-definition*)

パラメーター

名前	型	説明
transform-definition	LONG VARCHAR	空間参照系の変換定義テキスト

戻り値

- **LONG VARCHAR**　変換定義を定義しているテキスト文字列を返します。

備考

変換定義のフォーマット済みコピーを返します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として +proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs +towgs84=0,0,0 +no_defs を返します。

```
SELECT ST_SpatialRefSys::ST_FormatTransformDefinition('+proj=longlat +ellps=WGS84
+datum=WGS84 +no_defs')
```

ST_FormatWKT メソッド

Well Known Text (WKT) 定義のフォーマット済みコピーを返します。

構文

ST_SpatialRefSys::**ST_FormatWKT**(*definition*)

パラメーター

名前	型	説明
definition	LONG VARCHAR	空間参照系の定義テキスト

戻り値

- **LONG VARCHAR**　空間参照系を WKT で定義しているテキスト文字列を返します。

備考

WKT 空間参照系定義のフォーマット済みコピーを返します。

空間参照系を記述する Well Known Text (WKT) は、ジオメトリを記述する WKT とはフォーマットが異なります。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として 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", 0.0174532925199433,AUTHORITY["EPSG","9122"]], AUTHORITY["EPSG","4326"]] を返します。

```
SELECT  ST_SpatialRefSys::ST_FormatWKT('GEOGCS["WGS
84",DATUM["WGS_1984",SPHEROID["WGS 84",
6378137,298.257223563,AUTHORITY["EPSG","7030"]],AUTHORITY["EPSG","6326"]],PRIMEM["Green
wich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",
0.0174532925199433,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG","4326"]]')
```

ST_GetUnProjectedTransformDefinition メソッド

投影のソースとなっている空間参照系の変換定義を返します。

構文

ST_SpatialRefSys::**ST_GetUnProjectedTransformDefinition**(*transform-definition*)

パラメーター

名前	型	説明
transform-definition	LONG VARCHAR	空間参照系の変換定義テキスト

戻り値

- **LONG VARCHAR**　未投影の空間参照系の変換定義を定義しているテキスト文字列を返します。

備考

transform-definition パラメーターで投影後の空間参照系が定義されている場合は、ソースの空間参照系の定義を返します。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として +proj=latlong +a=6371000 +b=6371000 +no_defs を返します。

```
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 メソッド

名前付き要素を空間参照系の Well Known Text (WKT) 定義から取り出します。

構文

ST_SpatialRefSys::**ST_ParseWKT**(*element*,*srs-text*)

パラメーター

名前	型	説明
element	VARCHAR(128)	WKT から取り出す要素。次の名前付き要素を取り出すことができます。 ● **srs_name**　空間参照系の名前。 ● **srs_type**　座標系のタイプ。 ● **organization**　空間参照系を定義した組織の名前。 ● **organization_id**　空間参照系を定義した組織によって割り当てられた整数の識別子。 ● **linear_unit_of_measure**　線形測定単位の名前。 ● **linear_unit_of_measure_factor**　線形測定単位の変換係数。 ● **angular_unit_of_measure**　角度測定単位の名前。 ● **angular_unit_of_measure_factor**　角度測定単位の変換係数。
srs-text	LONG VARCHAR	空間参照系の定義テキスト

戻り値

● **LONG VARCHAR**　名前付き要素を空間参照系の WKT 定義から取り出します。

備考

名前付き要素を空間参照系の WKT 定義から取り出します。WKT で名前付き要素が定義されていない場合は、NULL が返されます。

空間参照系を記述する Well Known Text (WKT) は、ジオメトリを記述する WKT とはフォーマットが異なります。

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、各名前付き要素について、1 ローずつの結果を返します。

```
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"]],PRIMEM["Green
wich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",
0.0174532925199433,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_me
asure_factor,angular_unit_of_measure,angular_unit_of_measure_factor') D
)
SELECT element, ST_SpatialRefSys::ST_ParseWKT( element, srs_text ) parsed
FROM V
```

この例では、次の結果セットを返します。

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 メソッド

指定した変換定義を使用して変換されたジオメトリを返します。

構文

ST_SpatialRefSys::**ST_TransformGeom**(*geom*,*target-transform-definition*[, *source-transform-definition*])

パラメーター

名前	型	説明
geom	ST_Geometry	変換するジオメトリ。
target-transform-definition	LONG VARCHAR	ターゲットの空間参照系の変換定義テキスト。
source-transform-definition	LONG VARCHAR	ソースの空間参照系の変換定義テキスト。指定しない場合は、*geom* パラメーターの空間参照系の変換定義が使用されます。

戻り値

- **ST_Geometry**　指定した変換定義を使用して変換された入力ジオメトリを返します。

  結果の空間参照系識別子は sa_planar_unbounded です (SRID 2147483646)。

備考

ST_TransformGeom メソッドは、ターゲットの変換定義を指定した 1 つのジオメトリを変換します。変換は、PROJ.4 ライブラリを使用して実行されます。このメソッドは、適切な空間参照系がまだデータベースに作成されていない場合の選択状況で使用できます。適切な空間参照系が使用可能であれば、多くの場合、ST_Transform メソッドはより妥当なものになります。

緯度経度系から直交座標系への変換では、極のポイントに問題が生じることがあります。データベースサーバーで北極または南極に近いポイントを変換できない場合、変換が成功するように、ポイントの緯度値が同じ経度に沿って極から若干 (1e-10 ラジアン強) 離れます。

参照

- ST_Geometry タイプの ST_Transform メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　ベンダー拡張

例

次の例では、結果として Point (-5387692.968586 4763459.253243) を返します。

```
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 メソッド

空間参照系内のすべてのポイントを表すジオメトリを返します。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

構文

ST_SpatialRefSys::**ST_World**(*srid*)

パラメーター

名前	型	説明
srid	INT	結果に使用する SRID。

戻り値

- **ST_Surface** *srid* パラメーターで識別された空間参照系内のすべてのポイントを表すジオメトリを返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

参照

- ST_Surface タイプの ST_IsWorld メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** ベンダー拡張

例

次の例では、結果として Polygon ((-1000000 -1000000, 1000000 -1000000, 1000000 1000000, -1000000 1000000, -1000000 -1000000)) を返します。

```
SELECT ST_SpatialRefSys::ST_World(0)
```

ST_Surface タイプ

ST_Surface タイプは、2 次元のジオメトリタイプのスーパータイプです。ST_Surface タイプはインスタンス化できません。

直接のスーパータイプ

- 「ST_Geometry タイプ」

直接のサブタイプ

- 「ST_CurvePolygon タイプ」

メソッド

- ST_Surface のメソッド：

ST_Area	ST_Centroid	ST_IsWorld	ST_Perimeter
ST_PointOnSurface			

- また、「ST_Geometry タイプ」 のすべてのメソッドを ST_Surface タイプで呼び出すことができます。

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

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.1

ST_Area メソッド

指定した単位で面の面積を計算します。

構文

surface-expression.**ST_Area**([*unit-name*])

パラメーター

名前	型	説明
unit-name	VARCHAR(128)	長さを計算する単位。デフォルトでは、空間参照系の単位が使用されます。単位名は、UNIT_TYPE が 'LINEAR' の ST_UNITS_OF_MEASURE ビュー内のローの UNIT_NAME カラムと一致させてください。

戻り値

- **DOUBLE** 面の面積を返します。

備考

ST_Area メソッドは、面の面積を計算します。面積を表すために使用される単位は、指定した線形測定単位に基づきます。たとえば、指定した線形測定単位がフィートの場合、面積に使用される単位は平方フィートです。

注意
このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

- ST_Surface タイプの ST_Perimeter メソッド
- ST_MultiSurface タイプの ST_Area メソッド
- ST_Curve タイプの ST_Length メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.1.2

例

次の例では、結果として 12.5 を返します。

```
SELECT TREAT( Shape AS ST_Polygon ).ST_Area()
FROM SpatialShapes WHERE ShapeID = 22
```

次の例では、架空の region テーブルから、poly_geometry カラムの領域を平方マイルで返します。

```
SELECT name, poly_geometry.ST_Area( 'Statute Mile' )
FROM region
```

ST_Centroid メソッド

面の値の数学的重心である ST_Point 値を返します。

構文

surface-expression.**ST_Centroid**()

戻り値

- **ST_Point** 面が空のセットの場合は、NULL を返します。それ以外の場合は、面の数学的重心を返します。

  結果の空間参照系識別子は、*surface-expression* の空間参照系と同じです。

備考

面の値の数学的重心である ST_Point 値を返します。このポイントは必ずしも面上のポイントにはならないことに注意してください。

> **注意**
> このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

参照

- ST_MultiSurface タイプの ST_Centroid メソッド
- ST_Surface タイプの ST_PointOnSurface メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 8.1.4

例

次の例では、結果として Point (5 4.666667) を返します。

```
SELECT TREAT( Shape as ST_Surface ).ST_Centroid()
FROM SpatialShapes WHERE ShapeID = 22
```

ST_IsWorld メソッド

ST_Surface に空間全体が含まれているかどうかをテストします。

> **注意**
> このメソッドは、曲面の空間参照系のジオメトリでは使用できません。

構文

surface-expression.**ST_IsWorld**()

戻り値

- **BIT**　面に空間全体が含まれている場合は 1 を返し、それ以外の場合は 0 を返します。

参照

- ST_SpatialRefSys タイプの ST_World メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　8.1.6

例

次の例では、結果として 1 を返します。

```
SELECT NEW ST_Polygon( NEW ST_Point( -180, -90, 1000004326 ),
                       NEW ST_Point( 180, 90, 1000004326 ) ).ST_IsWorld()
```

ST_Perimeter メソッド

指定した単位で面の周囲の長さを計算します。

構文

surface-expression.**ST_Perimeter**([*unit-name*])

パラメーター

名前	型	説明
unit-name	VARCHAR(128)	長さを計算する単位。デフォルトでは、空間参照系の単位が使用されます。単位名は、UNIT_TYPE が 'LINEAR' の ST_UNITS_OF_MEASURE ビュー内のローの UNIT_NAME カラムと一致させてください。

戻り値

- **DOUBLE**　指定した測定単位で面の周囲の長さを返します。

備考

ST_Perimeter メソッドは、*unit-name* パラメーターで指定された単位で面の周囲の長さを返します。面が空の場合は、NULL が返されます。

面に Z 値が含まれている場合、それらの値はジオメトリの周囲の長さの計算時には考慮されません。

多角形の周囲の長さには、すべてのリング (外部と内部) の長さが含まれます。

注意
surface-expression が空のジオメトリ (ST_IsEmpty()=1) の場合、このメソッドは NULL を返します。

注意
ST_Perimeter では、デフォルトで、使用可能な場合はジオメトリの元のフォーマットが使用されます。それ以外の場合は、内部フォーマットが使用されます。内部フォーマットと元のフォーマットの詳細については、STORAGE FORMAT 句、CREATE SPATIAL REFERENCE SYSTEM 文『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_MultiSurface タイプの ST_Perimeter メソッド
- ST_Geometry タイプの ST_Boundary メソッド
- ST_Curve タイプの ST_Length メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　8.1.3

例

次の例では、結果として 18 を返します。

```
SELECT TREAT( Shape as ST_Surface ).ST_Perimeter()
FROM SpatialShapes WHERE ShapeID = 3
```

次の例では、架空の region テーブルから、poly_geometry カラムの周囲の長さをマイルで返します。

```
SELECT name, poly_geometry.ST_Perimeter( 'Statute Mile' )
FROM region
```

ST_PointOnSurface メソッド

ST_Surface 値と空間的に交差することが保証される ST_Point 値を返します。

注意
surface-expression に円ストリングが含まれている場合、それらは線ストリングに補間されます。

構文

surface-expression.**ST_PointOnSurface**()

戻り値

- **ST_Point**　面が空のセットの場合は、NULL を返します。それ以外の場合は、ST_Surface 値と空間的に交差することが保証される ST_Point 値を返します。

  結果の空間参照系識別子は、*surface-expression* の空間参照系と同じです。

参照

- ST_MultiSurface タイプの ST_PointOnSurface メソッド
- ST_Surface タイプの ST_Centroid メソッド
- ST_Geometry タイプの ST_Intersects メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　8.1.5

例

次の例では、多角形と交差するポイントを返します。

```
SELECT NEW ST_Polygon( 'Polygon(( 1 0, 0 10, 1 1, 2 10, 1 0 ))' )
    .ST_PointOnSurface()
```

空間互換関数

SQL/MM 標準では、空間操作を実行するために使用できる多数の関数が定義されています。 ほとんどの場合、これらの関数は空間データ型のメソッドまたはコンストラクターの機能を複製したものです。

関数

名前	説明
「ST_BdMPolyFromText 関数 [空間]」	複数線ストリングの Well Known Text (WKT) 表現で構成された ST_MultiPolygon 値を返します。
「ST_BdMPolyFromWKB 関数 [空間]」	複数線ストリングの Well Known Binary (WKB) 表現で構成された ST_MultiPolygon 値を返します。
「ST_BdPolyFromText 関数 [空間]」	複数線ストリングの Well Known Text (WKT) 表現で構成された ST_Polygon 値を返します。
「ST_BdPolyFromWKB 関数 [空間]」	複数線ストリングの Well Known Binary (WKB) 表現で構成された ST_Polygon 値を返します。
「ST_CPolyFromText 関数 [空間]」	ST_CurvePolygon の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_CurvePolygon 値を返します。
「ST_CPolyFromWKB 関数 [空間]」	ST_CurvePolygon の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_CurvePolygon 値を返します。
「ST_CircularFromTxt 関数 [空間]」	ST_CircularString の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_CircularString 値を返します。
「ST_CircularFromWKB 関数 [空間]」	ST_CircularString の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_CircularString 値を返します。
「ST_CompoundFromTxt 関数 [空間]」	ST_CompoundCurve の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_CompoundCurve 値を返します。
「ST_CompoundFromWKB 関数 [空間]」	ST_CompoundCurve の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_CompoundCurve 値を返します。
「ST_GeomCollFromTxt 関数 [空間]」	ST_GeomCollection の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_GeomCollection 値を返します。

名前	説明
「ST_GeomCollFromWKB 関数 [空間]」	ST_GeomCollection の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_GeomCollection 値を返します。
「ST_GeomFromText 関数 [空間]」	ST_Geometry の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_Geometry 値を返します。
「ST_GeomFromWKB 関数 [空間]」	ST_Geometryn の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_Geometry 値を返します。
「ST_LineFromText 関数 [空間]」	ST_LineString の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_LineString 値を返します。
「ST_LineFromWKB 関数 [空間]」	ST_LineString の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_LineString 値を返します。
「ST_MCurveFromText 関数 [空間]」	ST_MultiCurve の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_MultiCurve 値を返します。
「ST_MCurveFromWKB 関数 [空間]」	ST_MultiCurve の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_MultiCurve 値を返します。
「ST_MLineFromText 関数 [空間]」	ST_MultiLineString の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_MultiLineString 値を返します。
「ST_MLineFromWKB 関数 [空間]」	ST_MultiLineString の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_MultiLineString 値を返します。
「ST_MPointFromText 関数 [空間]」	ST_MultiPoint の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_MultiPoint 値を返します。
「ST_MPointFromWKB 関数 [空間]」	ST_MultiPoint の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_MultiPoint 値を返します。

名前	説明
「ST_MPolyFromText 関数 [空間]」	ST_MultiPolygon の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_MultiPolygon 値を返します。
「ST_MPolyFromWKB 関数 [空間]」	ST_MultiPolygon の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_MultiPolygon 値を返します。
「ST_MSurfaceFromTxt 関数 [空間]」	ST_MultiSurface の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_MultiSurface 値を返します。
「ST_MSurfaceFromWKB 関数 [空間]」	ST_MultiSurface の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_MultiSurface 値を返します。
「ST_OrderingEquals 関数 [空間]」	ジオメトリが別のジオメトリと同一であるかどうかをテストします。
「ST_PointFromText 関数 [空間]」	ST_Point の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_Point 値を返します。
「ST_PointFromWKB 関数 [空間]」	ST_Point の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_Point 値を返します。
「ST_PolyFromText 関数 [空間]」	ST_Polygon の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_Polygon 値を返します。
「ST_PolyFromWKB 関数 [空間]」	ST_Polygon の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_Polygon 値を返します。

ST_BdMPolyFromText 関数 [空間]

複数線ストリングの Well Known Text (WKT) 表現で構成された ST_MultiPolygon 値を返します。

構文

[DBO.]ST_BdMPolyFromText(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	複数線ストリング値の WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_MultiPolygon**　複数線ストリングの WKT 表現で構成された ST_MultiPolygon 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_BdMPolyFromText 関数は次のものと同等です。

```
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
```

注意
ST_BdMPolyFromText 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_Polygon コンストラクター」

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.6.7

ST_BdMPolyFromWKB 関数 [空間]

複数線ストリングの Well Known Binary (WKB) 表現で構成された ST_MultiPolygon 値を返します。

構文

[DBO.]ST_BdMPolyFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	複数線ストリング値の WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_MultiPolygon**　複数線ストリングの WKB 表現で構成された ST_MultiPolygon 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_BdMPolyFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_BdMPolyFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_Polygon コンストラクター」

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.6.8

ST_BdPolyFromText 関数 [空間]

複数線ストリングの Well Known Text (WKT) 表現で構成された ST_Polygon 値を返します。

構文

[DBO.]ST_BdPolyFromText(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	複数線ストリング値の WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

● **ST_Polygon**　複数線ストリングの WKT 表現で構成された ST_Polygon 値を返します。

結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_BdPolyFromText 関数は次のものと同等です。

```
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
```

注意
ST_BdPolyFromText 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

● 「ST_Polygon コンストラクター」

標準と互換性

● **SQL/MM (ISO/IEC 13249-3: 2006)**　8.3.9

ST_BdPolyFromWKB 関数 [空間]

複数線ストリングの Well Known Binary (WKB) 表現で構成された ST_Polygon 値を返します。

構文

[DBO.]ST_BdPolyFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	複数線ストリング値の WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_Polygon**　複数線ストリングの WKB 表現で構成された ST_Polygon 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_BdPolyFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_BdPolyFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_Polygon コンストラクター」

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　8.3.10

ST_CPolyFromText 関数 [空間]

ST_CurvePolygon の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_CurvePolygon 値を返します。

構文

[DBO.]ST_CPolyFromText(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_CurvePolygon**　入力文字列から作成された ST_CurvePolygon 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_CPolyFromText 関数は次のものと同等です。

```
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
```

注意
ST_CPolyFromText 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_CurvePolygon コンストラクター」
- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　8.2.8

ST_CPolyFromWKB 関数 [空間]

ST_CurvePolygon の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_CurvePolygon 値を返します。

構文

[DBO.]ST_CPolyFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_CurvePolygon**　入力文字列から作成された ST_CurvePolygon 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_CPolyFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_CPolyFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_CurvePolygon コンストラクター」
- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　8.2.9

ST_CircularFromTxt 関数 [空間]

ST_CircularString の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_CircularString 値を返します。

構文

[DBO.]ST_CircularFromTxt(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_CircularString**　入力文字列から作成された ST_CircularString 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_CircularFromTxt 関数は次のものと同等です。

```
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
```

注意
ST_CircularFromTxt 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_CircularString コンストラクター」
- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　7.3.9

ST_CircularFromWKB 関数 [空間]

ST_CircularString の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_CircularString 値を返します。

構文

[DBO.]ST_CircularFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_CircularString**　入力文字列から作成された ST_CircularString 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_CircularFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_CircularFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_CircularString コンストラクター」
- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　7.3.10

ST_CompoundFromTxt 関数 [空間]

ST_CompoundCurve の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_CompoundCurve 値を返します。

構文

[DBO.]ST_CompoundFromTxt(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_CompoundCurve**　入力文字列から作成された ST_CompoundCurve 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_CompoundFromTxt 関数は次のものと同等です。

```
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
```

注意
ST_CompoundFromTxt 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_CompoundCurve コンストラクター」
- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　7.4.8

ST_CompoundFromWKB 関数 [空間]

ST_CompoundCurve の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_CompoundCurve 値を返します。

構文

[DBO.]ST_CompoundFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_CompoundCurve**　入力文字列から作成された ST_CompoundCurve 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_CompoundFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_CompoundFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_CompoundCurve コンストラクター」
- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　7.4.9

ST_GeomCollFromTxt 関数 [空間]

ST_GeomCollection の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_GeomCollection 値を返します。

構文

[DBO.]ST_GeomCollFromTxt(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_GeomCollection**　入力文字列から作成された ST_GeomCollection 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_GeomCollFromTxt 関数は次のものと同等です。

```
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
```

注意
ST_GeomCollFromTxt 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_GeomCollection コンストラクター」
- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.1.6

ST_GeomCollFromWKB 関数 [空間]

ST_GeomCollection の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_GeomCollection 値を返します。

構文

[DBO.]ST_GeomCollFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_GeomCollection**　入力文字列から作成された ST_GeomCollection 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_GeomCollFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_GeomCollFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_GeomCollection コンストラクター」
- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.1.7

ST_GeomFromText 関数 [空間]

ST_Geometry の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_Geometry 値を返します。

構文

[DBO.]ST_GeomFromText(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_Geometry**　入力文字列から作成された ST_Geometry 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_GeomFromText 関数は次のものと同等です。

```
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
```

注意
ST_GeomFromText 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.40

ST_GeomFromWKB 関数 [空間]

ST_Geometry の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_Geometry 値を返します。

構文

[DBO.]ST_GeomFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_Geometry**　入力文字列から作成された ST_Geometry 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_GeomFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_GeomFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　5.1.41

ST_LineFromText 関数 [空間]

ST_LineString の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_LineString 値を返します。

構文

[DBO.]ST_LineFromText(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_LineString**　入力文字列から作成された ST_LineString 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_LineFromText 関数は次のものと同等です。

```
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
```

注意
ST_LineFromText 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_LineString コンストラクター」
- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　7.2.8

ST_LineFromWKB 関数 [空間]

ST_LineString の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_LineString 値を返します。

構文

[DBO.]ST_LineFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_LineString**　入力文字列から作成された ST_LineString 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_LineFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_LineFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_LineString コンストラクター」
- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　7.2.9

ST_MCurveFromText 関数 [空間]

ST_MultiCurve の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_MultiCurve 値を返します。

構文

[DBO.]ST_MCurveFromText(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_MultiCurve**　入力文字列から作成された ST_MultiCurve 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_MCurveFromText 関数は次のものと同等です。

```
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
```

注意
ST_MCurveFromText 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_MultiCurve コンストラクター」
- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.3.6

ST_MCurveFromWKB 関数 [空間]

ST_MultiCurve の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_MultiCurve 値を返します。

構文

[DBO.]ST_MCurveFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_MultiCurve**　入力文字列から作成された ST_MultiCurve 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_MCurveFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_MCurveFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_MultiCurve コンストラクター」
- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.3.7

ST_MLineFromText 関数 [空間]

ST_MultiLineString の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_MultiLineString 値を返します。

構文

[DBO.]ST_MLineFromText(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_MultiLineString**　入力文字列から作成された ST_MultiLineString 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_MLineFromText 関数は次のものと同等です。

```
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
```

注意
ST_MLineFromText 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_MultiLineString コンストラクター」
- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.4.4

ST_MLineFromWKB 関数 [空間]

ST_MultiLineString の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_MultiLineString 値を返します。

構文

[DBO.]ST_MLineFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_MultiLineString**　入力文字列から作成された ST_MultiLineString 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_MLineFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_MLineFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_MultiLineString コンストラクター」
- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.4.5

ST_MPointFromText 関数 [空間]

ST_MultiPoint の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_MultiPoint 値を返します。

構文

[DBO.]ST_MPointFromText(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_MultiPoint**　入力文字列から作成された ST_MultiPoint 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_MPointFromText 関数は次のものと同等です。

```
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
```

注意
ST_MPointFromText 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_MultiPoint コンストラクター」
- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.2.4

ST_MPointFromWKB 関数 [空間]

ST_MultiPoint の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_MultiPoint 値を返します。

構文

[DBO.]ST_MPointFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_MultiPoint**　入力文字列から作成された ST_MultiPoint 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_MPointFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_MPointFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_MultiPoint コンストラクター」
- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.2.5

ST_MPolyFromText 関数 [空間]

ST_MultiPolygon の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_MultiPolygon 値を返します。

構文

[DBO.]ST_MPolyFromText(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_MultiPolygon**　入力文字列から作成された ST_MultiPolygon 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_MPolyFromText 関数は次のものと同等です。

```
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
```

注意
ST_MPolyFromText 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_MultiPolygon コンストラクター」
- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.6.4

ST_MPolyFromWKB 関数 [空間]

ST_MultiPolygon の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_MultiPolygon 値を返します。

構文

[DBO.]ST_MPolyFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_MultiPolygon**　入力文字列から作成された ST_MultiPolygon 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_MPolyFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_MPolyFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_MultiPolygon コンストラクター」
- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.6.5

ST_MSurfaceFromTxt 関数 [空間]

ST_MultiSurface の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_MultiSurface 値を返します。

構文

[DBO.]ST_MSurfaceFromTxt(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_MultiSurface**　入力文字列から作成された ST_MultiSurface 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_MSurfaceFromTxt 関数は次のものと同等です。

```
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
```

注意
ST_MSurfaceFromTxt 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_MultiSurface コンストラクター」
- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.5.8

ST_MSurfaceFromWKB 関数 [空間]

ST_MultiSurface の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_MultiSurface 値を返します。

構文

[DBO.]ST_MSurfaceFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_MultiSurface**　入力文字列から作成された ST_MultiSurface 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_MSurfaceFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_MSurfaceFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。 sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_MultiSurface コンストラクター」
- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　9.5.9

ST_OrderingEquals 関数 [空間]

ジオメトリが別のジオメトリと同一であるかどうかをテストします。

構文

[DBO.]ST_OrderingEquals(*geo1*,*geo2*)

パラメーター

名前	型	説明
geo1	ST_Geometry	順序付ける最初のジオメトリ値。
geo2	ST_Geometry	順序付ける 2 番目のジオメトリ値。

戻り値

- **INT** *geo1* が *geo2* と完全に等しい場合は 1 を返し、それ以外の場合は 0 を返します。

備考

ST_OrderingEquals 関数は次のものと同等です。

```
CREATE FUNCTION DBO.ST_OrderingEquals( geo1 ST_Geometry, geo2 ST_Geometry )
RETURNS INT
BEGIN
   RETURN geo1.ST_OrderingEquals( geo2 );
END
```

注意
ST_OrderingEquals 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- ST_Geometry タイプの ST_OrderingEquals メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)** 5.1.43

ST_PointFromText 関数 [空間]

ST_Point の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_Point 値を返します。

構文

[DBO.]ST_PointFromText(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_Point**　入力文字列から作成された ST_Point 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_PointFromText 関数は次のものと同等です。

```
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
```

注意
ST_PointFromText 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_Point コンストラクター」
- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　6.1.8

ST_PointFromWKB 関数 [空間]

ST_Point の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_Point 値を返します。

構文

[DBO.]ST_PointFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_Point**　入力文字列から作成された ST_Point 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_PointFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_PointFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_Point コンストラクター」
- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　6.1.9

ST_PolyFromText 関数 [空間]

ST_Polygon の Well Known Text (WKT) 表現を含む LONG VARCHAR 値から変換された ST_Polygon 値を返します。

構文

[DBO.]ST_PolyFromText(*wkt*[, *srid*])

パラメーター

名前	型	説明
wkt	LONG VARCHAR	WKT 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_Polygon**　入力文字列から作成された ST_Polygon 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_PolyFromText 関数は次のものと同等です。

```
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
```

注意
ST_PolyFromText 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_Polygon コンストラクター」
- ST_Geometry タイプの ST_GeomFromText メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　8.3.6

ST_PolyFromWKB 関数 [空間]

ST_Polygon の Well Known Binary (WKB) 表現を含む LONG BINARY 値から変換された ST_Polygon 値を返します。

構文

[DBO.]ST_PolyFromWKB(*wkb*[, *srid*])

パラメーター

名前	型	説明
wkb	LONG BINARY	WKB 表現。
srid	INT	結果の SRID。指定しない場合、デフォルトは 0 です。

戻り値

- **ST_Polygon**　入力文字列から作成された ST_Polygon 値を返します。

  結果の空間参照系識別子は、*srid* パラメーターで指定したものです。

備考

ST_PolyFromWKB 関数は次のものと同等です。

```
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
```

注意
ST_PolyFromWKB 関数は、新しく作成されたデータベースにはデフォルトでは存在しません。sa_install_feature システムプロシージャーを使用して、空間 SQL 互換関数をインストールしてください。「sa_install_feature システムプロシージャー」『SQL Anywhere サーバー SQL リファレンス』を参照してください。

参照

- 「ST_Polygon コンストラクター」
- ST_Geometry タイプの ST_GeomFromWKB メソッド

標準と互換性

- **SQL/MM (ISO/IEC 13249-3: 2006)**　8.3.7

サポートされているすべてのメソッドのリスト

次に、サポートされているすべての空間メソッドのリストを示します。曲面カラムの X は、曲面空間参照系でもメソッドがサポートされていることを示しています。SQL/MM カラムには、SQL/MM 標準 (ISO/IEC 13249-3: 2006) との準拠が反映されています。

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_Affine メソッド	「ST_Geometry タイプ」	回転、平行移動、スケーリングを 1 回の呼び出しで実行するアフィン変換を実行します。		ベンダー拡張
ST_Area メソッド	「ST_MultiSurface タイプ」	指定した単位で複数面の面積を計算します。		9.5.3
ST_Area メソッド	「ST_Surface タイプ」	指定した単位で面の面積を計算します。		8.1.2
ST_AsBinary メソッド	「ST_Geometry タイプ」	ST_Geometry 値の WKB 表現を返します。	X	5.1.37
ST_AsGML メソッド	「ST_Geometry タイプ」	ST_Geometry 値の GML 表現を返します。	X	5.1.39
ST_AsGeoJSON メソッド	「ST_Geometry タイプ」	ジオメトリを JSON フォーマットで表す文字列を返します。	X	ベンダー拡張
ST_AsKML メソッド	「ST_Geometry タイプ」	ST_Geometry 値の KML 表現を返します。	X	5.1.39
ST_AsSVG メソッド	「ST_Geometry タイプ」	ジオメトリ値を表す SVG 図形を返します。	X	ベンダー拡張
ST_AsText メソッド	「ST_Geometry タイプ」	ST_Geometry 値のテキスト表現を返します。	X	5.1.35
ST_AsWKB メソッド	「ST_Geometry タイプ」	ST_Geometry 値の WKB 表現を返します。	X	ベンダー拡張

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_AsWKT メソッド	「ST_Geometry タイプ」	ST_Geometry 値の WKT 表現を返します。	X	ベンダー拡張
ST_AsXML メソッド	「ST_Geometry タイプ」	ST_Geometry 値の XML 表現を返します。	X	ベンダー拡張
ST_Boundary メソッド	「ST_Geometry タイプ」	ジオメトリ値の境界を返します。		5.1.14
ST_Centroid メソッド	「ST_MultiSurface タイプ」	複数面の数学的重心である ST_Point を計算します。		9.5.5
ST_Centroid メソッド	「ST_Surface タイプ」	面の値の数学的重心である ST_Point 値を返します。		8.1.4
ST_Contains メソッド	「ST_Geometry タイプ」	ジオメトリ値に別のジオメトリ値が空間的に含まれているかどうかをテストします。		5.1.31
ST_ContainsFilter メソッド	「ST_Geometry タイプ」	ジオメトリに別のジオメトリが含まれているかどうかの低コストのテスト。		ベンダー拡張
ST_ConvexHull メソッド	「ST_Geometry タイプ」	ジオメトリ値の凸包を返します。		5.1.16
ST_CoordDim メソッド	「ST_Geometry タイプ」	ST_Geometry 値の各ポイントで格納されている座標次元の数を返します。	X	5.1.3

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_CoveredBy メソッド	「ST_Geometry タイプ」	ジオメトリ値が別のジオメトリ値に空間的に含まれているかどうかをテストします。	X	ベンダー拡張
ST_CoveredByFilter メソッド	「ST_Geometry タイプ」	ジオメトリが別のジオメトリに含まれているかどうかの低コストのテスト。	X	ベンダー拡張
ST_Covers メソッド	「ST_Geometry タイプ」	ジオメトリ値に別のジオメトリ値が空間的に含まれているかどうかをテストします。	X	ベンダー拡張
ST_CoversFilter メソッド	「ST_Geometry タイプ」	ジオメトリに別のジオメトリが含まれているかどうかの低コストのテスト。	X	ベンダー拡張
ST_Crosses メソッド	「ST_Geometry タイプ」	ジオメトリ値が別のジオメトリ値と交差しているかどうかをテストします。		5.1.29
ST_CurveN メソッド	「ST_CompoundCurve タイプ」	複合曲線の *n* 番目の曲線を返します。	X	7.4.5
ST_CurvePolyToPoly メソッド	「ST_CurvePolygon タイプ」	曲線多角形の補間近似値を多角形として返します。	X	8.2.7
ST_CurveToLine メソッド	「ST_Curve タイプ」	ST_Curve 値の ST_LineString 補間近似値を返します。	X	7.1.7

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_Difference メソッド	「ST_Geometry タイプ」	2 つのジオメトリの差集合を表すジオメトリ値を返します。	X	5.1.20
ST_Dimension メソッド	「ST_Geometry タイプ」	ST_Geometry 値の次元を返します。ポイントの次元は 0、線の次元は 1、面の次元は 2 です。空のジオメトリの次元は -1 です。	X	5.1.2
ST_Disjoint メソッド	「ST_Geometry タイプ」	ジオメトリ値が別のジオメトリ値から空間的に分断されているかどうかをテストします。	X	5.1.26
ST_Distance メソッド	「ST_Geometry タイプ」	${selfexpr} と指定したジオメトリ値間の最短距離を返します。	X	5.1.23
ST_EndPoint メソッド	「ST_Curve タイプ」	ST_Curve 値の終了ポイントである ST_Point 値を返します。	X	7.1.4
ST_Envelope メソッド	「ST_Geometry タイプ」	ジオメトリ値の外接矩形を返します。		5.1.15
ST_Equals メソッド	「ST_Geometry タイプ」	ST_Geometry 値が別の ST_Geometry 値と空間的に等しいかどうかをテストします。	X	5.1.24

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_EqualsFilter メソッド	「ST_Geometry タイプ」	ジオメトリが別のジオメトリと等しいかどうかの低コストのテスト。	X	ベンダー拡張
ST_ExteriorRing メソッド	「ST_CurvePolygon タイプ」	外部リングを取り出したり、変更したりします。	X	8.2.3
ST_ExteriorRing メソッド	「ST_Polygon タイプ」	外部リングを取り出したり、変更したりします。	X	8.3.3
ST_GeometryN メソッド	「ST_GeomCollection タイプ」	ジオメトリコレクションの *n* 番目のジオメトリを返します。	X	9.1.5
ST_GeometryType メソッド	「ST_Geometry タイプ」	ST_Geometry 値のタイプの名前を返します。	X	5.1.4
ST_InteriorRingN メソッド	「ST_CurvePolygon タイプ」	曲線多角形の *n* 番目の内部リングを返します。	X	8.2.6
ST_InteriorRingN メソッド	「ST_Polygon タイプ」	多角形の *n* 番目の内部リングを返します。	X	8.3.5
ST_Intersection メソッド	「ST_Geometry タイプ」	2 つのジオメトリの積集合を表すジオメトリ値を返します。	X	5.1.18
ST_Intersects メソッド	「ST_Geometry タイプ」	ジオメトリ値が別の値と空間的に交差しているかどうかをテストします。	X	5.1.27

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_IntersectsFilter メソッド	「ST_Geometry タイプ」	2 つのジオメトリが交差しているかどうかの低コストのテスト。	X	ベンダー拡張
ST_IntersectsRect メソッド	「ST_Geometry タイプ」	ジオメトリが長方形と交差しているかどうかをテストします。	X	ベンダー拡張
ST_Is3D メソッド	「ST_Geometry タイプ」	ジオメトリ値に Z 座標値が含まれているかどうかを調べます。	X	5.1.10
ST_IsClosed メソッド	「ST_Curve タイプ」	ST_Curve 値が閉じているかどうかをテストします。開始ポイントと終了ポイントが一致する場合、曲線は閉じています。	X	7.1.5
ST_IsClosed メソッド	「ST_MultiCurve タイプ」	ST_MultiCurve 値が閉じているかどうかをテストします。開始ポイントと終了ポイントが一致する場合、曲線は閉じています。空ではなく、空の境界を持つ複数曲線は閉じています。	X	9.3.3
ST_IsEmpty メソッド	「ST_Geometry タイプ」	ジオメトリ値が空のセットを表すかどうかを調べます。	X	5.1.7

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_IsMeasured メソッド	「ST_Geometry タイプ」	ジオメトリ値に測定値が関連付けられているかどうかを調べます。	X	5.1.11
ST_IsRing メソッド	「ST_Curve タイプ」	ST_Curve 値がリングかどうかをテストします。曲線が閉じていて単純な場合 (それ自体と交差しない場合)、その曲線はリングです。	X	7.1.6
ST_IsSimple メソッド	「ST_Geometry タイプ」	ジオメトリ値が単純かどうかを調べます (それ自体と交差しないことや他の不規則性など)。	X	5.1.8
ST_IsValid メソッド	「ST_Geometry タイプ」	ジオメトリが有効な空間オブジェクトであるかどうかを調べます。	X	5.1.9
ST_IsWorld メソッド	「ST_Surface タイプ」	ST_Surface に空間全体が含まれているかどうかをテストします。		8.1.6
ST_Lat メソッド	「ST_Point タイプ」	ST_Point 値の緯度座標を返します。	X	ベンダー拡張
ST_LatNorth メソッド	「ST_Geometry タイプ」	ジオメトリの最北の緯度を取り出します。	X	ベンダー拡張
ST_LatSouth メソッド	「ST_Geometry タイプ」	ジオメトリの最南の緯度を取り出します。	X	ベンダー拡張

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_Length メソッド	「ST_Curve タイプ」	曲線値の長さの測定値を取り出します。	X	7.1.2
ST_Length メソッド	「ST_MultiCurve タイプ」	ST_MultiCurve 値の長さの測定値を返します。結果は、パラメーターで指定した単位で測定されます。	X	9.3.4
ST_LinearHash メソッド	「ST_Geometry タイプ」	ジオメトリの線形ハッシュであるバイナリ文字列を返します。	X	ベンダー拡張
ST_Long メソッド	「ST_Point タイプ」	ST_Point 値の経度座標を返します。	X	ベンダー拡張
ST_LongEast メソッド	「ST_Geometry タイプ」	ジオメトリの東の境界の経度を取り出します。	X	ベンダー拡張
ST_LongWest メソッド	「ST_Geometry タイプ」	ジオメトリの西の境界の経度を取り出します。	X	ベンダー拡張
ST_M メソッド	「ST_Point タイプ」	ポイントの測定値を取り出したり、変更したりします。	X	6.1.6
ST_MMax メソッド	「ST_Geometry タイプ」	ジオメトリの最大 M 座標値を取り出します。	X	ベンダー拡張
ST_MMin メソッド	「ST_Geometry タイプ」	ジオメトリの最小 M 座標値を取り出します。	X	ベンダー拡張
ST_NumCurves メソッド	「ST_CompoundCurve タイプ」	複合曲線を定義している曲線数を返します。	X	7.4.4

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_NumGeometries メソッド	「ST_GeomCollection タイプ」	ジオメトリコレクションに含まれているジオメトリ数を返します。	X	9.1.4
ST_NumInteriorRing メソッド	「ST_CurvePolygon タイプ」	曲線多角形の内部リング数を返します。	X	8.2.5
ST_NumPoints メソッド	「ST_CircularString タイプ」	円ストリングを定義しているポイント数を返します。	X	7.3.4
ST_NumPoints メソッド	「ST_LineString タイプ」	線ストリングを定義しているポイント数を返します。	X	7.2.4
ST_OrderingEquals メソッド	「ST_Geometry タイプ」	ジオメトリが別のジオメトリと同一であるかどうかをテストします。	X	5.1.43
ST_Overlaps メソッド	「ST_Geometry タイプ」	ジオメトリ値が別のジオメトリ値と重なり合うかどうかをテストします。		5.1.32
ST_Perimeter メソッド	「ST_MultiSurface タイプ」	指定した単位で複数面の周囲の長さを計算します。	X	9.5.4
ST_Perimeter メソッド	「ST_Surface タイプ」	指定した単位で面の周囲の長さを計算します。	X	8.1.3
ST_PointN メソッド	「ST_CircularString タイプ」	円ストリングの *n* 番目のポイントを返します。	X	7.3.5

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_PointN メソッド	「ST_LineString タイプ」	線ストリングの *n* 番目のポイントを返します。	X	7.2.5
ST_PointOnSurface メソッド	「ST_MultiSurface タイプ」	複数面内の面上にあることが保証されるポイントを返します。	X	9.5.6
ST_PointOnSurface メソッド	「ST_Surface タイプ」	ST_Surface 値と空間的に交差することが保証される ST_Point 値を返します。	X	8.1.5

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_Relate メソッド	「ST_Geometry タイプ」	ジオメトリ値が、交点マトリックスで指定されているとおりに別のジオメトリ値と空間的に関係しているかどうかをテストします。ST_Relate メソッドは、Dimensionally Extended 9 Intersection Model (DE-9IM) からの 9 文字の文字列を使用して、2 つの空間データ項目間のペアワイズの関係を示します。たとえば、ST_Relate メソッドは、ジオメトリ間に共通部分が存在するかどうかを確認し、存在する場合は結果の共通部分のジオメトリを調べます。参照： 「空間の関係操作」 55 ページ.		5.1.25, ベンダー拡張
ST_Reverse メソッド	「ST_Geometry タイプ」	要素の順序を逆にしたジオメトリを返します。	X	ベンダー拡張
ST_SRID メソッド	「ST_Geometry タイプ」	ジオメトリ値に関連付けられている空間参照系を取り出したり、変更したりします。	X	5.1.5

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_SnapToGrid メソッド	「ST_Geometry タイプ」	指定したグリッドにすべてのポイントがスナップされたジオメトリのコピーを返します。	X	ベンダー拡張
ST_StartPoint メソッド	「ST_Curve タイプ」	ST_Curve 値の開始ポイントである ST_Point 値を返します。	X	7.1.3
ST_SymDifference メソッド	「ST_Geometry タイプ」	2 つのジオメトリの対称差を表すジオメトリ値を返します。	X	5.1.21
ST_ToCircular メソッド	「ST_Geometry タイプ」	ジオメトリを円ストリングに変換します。	X	5.1.33
ST_ToCompound メソッド	「ST_Geometry タイプ」	ジオメトリを複合曲線に変換します。	X	5.1.33
ST_ToCurve メソッド	「ST_Geometry タイプ」	ジオメトリを曲線に変換します。	X	ベンダー拡張
ST_ToCurvePoly メソッド	「ST_Geometry タイプ」	ジオメトリを曲線多角形に変換します。	X	5.1.33
ST_ToGeomColl メソッド	「ST_Geometry タイプ」	ジオメトリをジオメトリコレクションに変換します。	X	5.1.33
ST_ToLineString メソッド	「ST_Geometry タイプ」	ジオメトリを線ストリングに変換します。	X	5.1.33
ST_ToMultiCurve メソッド	「ST_Geometry タイプ」	ジオメトリを複数曲線値に変換します。	X	5.1.33

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_ToMultiLine メソッド	「ST_Geometry タイプ」	ジオメトリを複数線ストリング値に変換します。	X	5.1.33
ST_ToMultiPoint メソッド	「ST_Geometry タイプ」	ジオメトリを複数ポイント値に変換します。	X	5.1.33
ST_ToMultiPolygon メソッド	「ST_Geometry タイプ」	ジオメトリを複数多角形値に変換します。	X	5.1.33
ST_ToMultiSurface メソッド	「ST_Geometry タイプ」	ジオメトリを複数面値に変換します。	X	5.1.33
ST_ToPoint メソッド	「ST_Geometry タイプ」	ジオメトリをポイントに変換します。	X	5.1.33
ST_ToPolygon メソッド	「ST_Geometry タイプ」	ジオメトリを多角形に変換します。	X	5.1.33
ST_ToSurface メソッド	「ST_Geometry タイプ」	ジオメトリを面に変換します。	X	ベンダー拡張
ST_Touches メソッド	「ST_Geometry タイプ」	ジオメトリ値が別のジオメトリ値と空間的に接触しているかどうかをテストします。		5.1.28
ST_Transform メソッド	「ST_Geometry タイプ」	指定した空間参照系に変換されたジオメトリ値のコピーを作成します。	X	5.1.6
ST_Union メソッド	「ST_Geometry タイプ」	2 つのジオメトリの和集合を表すジオメトリ値を返します。	X	5.1.19

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_Within メソッド	「ST_Geometry タイプ」	ジオメトリ値が別のジオメトリ値内に空間的に含まれているかどうかをテストします。		5.1.30
ST_WithinDistance メソッド	「ST_Geometry タイプ」	2 つのジオメトリが指定の相互距離内にあるかどうかをテストします。	X	ベンダー拡張
ST_WithinDistanceFilter メソッド	「ST_Geometry タイプ」	2 つのジオメトリが指定距離内にあるかどうかを判定するための負荷の低い方法。	X	ベンダー拡張
ST_WithinFilter メソッド	「ST_Geometry タイプ」	ジオメトリが別のジオメトリ内にあるかどうかの低コストのテスト。		ベンダー拡張
ST_X メソッド	「ST_Point タイプ」	ポイントの X 座標値を取り出したり、変更したりします。	X	6.1.3
ST_XMax メソッド	「ST_Geometry タイプ」	ジオメトリの最大 X 座標値を取り出します。	X	ベンダー拡張
ST_XMin メソッド	「ST_Geometry タイプ」	ジオメトリの最小 X 座標値を取り出します。	X	ベンダー拡張
ST_Y メソッド	「ST_Point タイプ」	ポイントの Y 座標値を取り出したり、変更したりします。	X	6.1.4

メソッド	タイプ	説明	Round-Earth	SQL/MM
ST_YMax メソッド	「ST_Geometry タイプ」	ジオメトリの最大 Y 座標値を取り出します。	X	ベンダー拡張
ST_YMin メソッド	「ST_Geometry タイプ」	ジオメトリの最小 Y 座標値を取り出します。	X	ベンダー拡張
ST_Z メソッド	「ST_Point タイプ」	ポイントの Z 座標値を取り出したり、変更したりします。	X	6.1.4, 6.1.5
ST_ZMax メソッド	「ST_Geometry タイプ」	ジオメトリの最大 Z 座標値を取り出します。	X	ベンダー拡張
ST_ZMin メソッド	「ST_Geometry タイプ」	ジオメトリの最小 Z 座標値を取り出します。	X	ベンダー拡張

サポートされているすべてのコンストラクターのリスト

次に、サポートされているすべての空間コンストラクターのリストを示します。曲面カラムの X は、 曲面空間参照系でもメソッドがサポートされていることを示しています。SQL/MM カラムには、 SQL/MM 標準 (ISO/IEC 13249-3: 2006) との準拠が反映されています。

コンストラクター	説明	曲面	SQL/MM	
「ST_CircularString コンストラクター」	円ストリングを構成します。	X	7.3.2, ベンダー拡張	
「ST_CompoundCurve コンストラクター」	複合曲線を構成します。	X	7.4.2, ベンダー拡張	
「ST_CurvePolygon コンストラクター」	曲線多角形を構成します。	X	8.2.2, ベンダー拡張	

コンストラクター	説明	曲面	SQL/MM	
「ST_GeomCollection コンストラクター」	ジオメトリコレクションを構成します。	X	9.1.2, ベンダー拡張	
「ST_LineString コンストラクター」	線ストリングを構成します。	X	7.2.2, ベンダー拡張	
「ST_MultiCurve コンストラクター」	複数曲線を構成します。	X	9.3.2, ベンダー拡張	
「ST_MultiLineString コンストラクター」	複数線ストリングを構成します。	X	9.4.2, ベンダー拡張	
「ST_MultiPoint コンストラクター」	複数ポイントを構成します。	X	9.2.2, ベンダー拡張	
「ST_MultiPolygon コンストラクター」	複数多角形を構成します。	X	9.6.2, ベンダー拡張	
「ST_MultiSurface コンストラクター」	複数面を構成します。	X	9.5.2, ベンダー拡張	
「ST_Point コンストラクター」	ポイントを構成します。	X	6.1.2	
「ST_Polygon コンストラクター」	多角形を構成します。	X	8.3.2, ベンダー拡張	

静的メソッドのリスト

次に、空間データで使用できるすべての静的メソッドのリストを示します。曲面カラムの X は、 曲面空間参照系でもメソッドがサポートされていることを示しています。SQL/MM カラムには、 SQL/MM 標準 (ISO/IEC 13249-3: 2006) との準拠が反映されています。

メソッド	タイプ	説明	曲面	SQL/MM
ST_AsSVGAggr メソッド	「ST_Geometry タイプ」	グループ内のジオメトリをレンダリングした完全または部分的な SVG ドキュメントを返します。	X	ベンダー拡張
ST_CompareWKT メソッド	「ST_SpatialRefSys タイプ」	2 つの空間参照系定義を比較します。	X	ベンダー拡張
ST_ConvexHullAggr メソッド	「ST_Geometry タイプ」	グループ内のすべてのジオメトリの凸包を返します。		ベンダー拡張
ST_EnvelopeAggr メソッド	「ST_Geometry タイプ」	グループ内のすべてのジオメトリの外接矩形を返します。		ベンダー拡張
ST_FormatTransformDefinition メソッド	「ST_SpatialRefSys タイプ」	変換定義のフォーマット済みコピーを返します。	X	ベンダー拡張
ST_FormatWKT メソッド	「ST_SpatialRefSys タイプ」	Well Known Text (WKT) 定義のフォーマット済みコピーを返します。	X	ベンダー拡張
ST_GeomCollectionAggr メソッド	「ST_GeomCollection タイプ」	グループ内のすべてのジオメトリを含むジオメトリコレクションを返します。	X	ベンダー拡張
ST_GeomFromBinary メソッド	「ST_Geometry タイプ」	バイナリ文字列表現からジオメトリを構成します。	X	ベンダー拡張

メソッド	タイプ	説明	曲面	SQL/MM
ST_GeomFromShape メソッド	「ST_Geometry タイプ」	ESRI シェイプレコードを含む文字列を解析し、適切なタイプのジオメトリ値を作成します。	X	ベンダー拡張
ST_GeomFromText メソッド	「ST_Geometry タイプ」	文字列表現からジオメトリを構成します。	X	5.1.40
ST_GeomFromWKB メソッド	「ST_Geometry タイプ」	ジオメトリの WKB または EWKB 表現を含む文字列を解析し、適切なタイプのジオメトリ値を作成します。	X	5.1.41
ST_GeomFromWKT メソッド	「ST_Geometry タイプ」	ジオメトリの WKT または EWKT 表現を含む文字列を解析し、適切なタイプのジオメトリ値を作成します。	X	ベンダー拡張
ST_GeometryTypeFromBaseType メソッド	「ST_Geometry タイプ」	タイプ文字列を定義している文字列を解析します。	X	ベンダー拡張
ST_GetUnProjectedTransformDefinition メソッド	「ST_SpatialRefSys タイプ」	投影のソースとなっている空間参照系の変換定義を返します。	X	ベンダー拡張
ST_IntersectionAggr メソッド	「ST_Geometry タイプ」	グループ内のすべてのジオメトリの空間的共通部分を返します。	X	ベンダー拡張

メソッド	タイプ	説明	曲面	SQL/MM
ST_LineStringAggr メソッド	「ST_LineString タイプ」	グループ内の順序付けされたポイントから構成された線ストリングを返します。	X	ベンダー拡張
ST_LinearUnHash メソッド	「ST_Geometry タイプ」	インデックスハッシュを表すジオメトリを返します。	X	ベンダー拡張
ST_LoadConfigurationData メソッド	「ST_Geometry タイプ」	バイナリ設定データを返します。内部でのみ使用。	X	ベンダー拡張
ST_MultiCurveAggr メソッド	「ST_MultiCurve タイプ」	グループ内のすべての曲線を含む複数曲線を返します。	X	ベンダー拡張
ST_MultiLineStringAggr メソッド	「ST_MultiLineString タイプ」	グループ内のすべての線ストリングを含む複数線ストリングを返します。	X	ベンダー拡張
ST_MultiPointAggr メソッド	「ST_MultiPoint タイプ」	グループ内のすべてのポイントを含む複数ポイントを返します。	X	ベンダー拡張
ST_MultiPolygonAggr メソッド	「ST_MultiPolygon タイプ」	グループ内のすべての多角形を含む複数多角形を返します。	X	ベンダー拡張
ST_MultiSurfaceAggr メソッド	「ST_MultiSurface タイプ」	グループ内のすべての面を含む複数面を返します。	X	ベンダー拡張

メソッド	タイプ	説明	曲面	SQL/MM
ST_ParseWKT メソッド	「ST_SpatialRefSys タイプ」	名前付き要素を空間参照系の Well Known Text (WKT) 定義から取り出します。	X	ベンダー拡張
ST_SRIDFromBaseType メソッド	「ST_Geometry タイプ」	タイプ文字列を定義している文字列を解析します。	X	ベンダー拡張
ST_TransformGeom メソッド	「ST_SpatialRefSys タイプ」	指定した変換定義を使用して変換されたジオメトリを返します。	X	ベンダー拡張
ST_UnionAggr メソッド	「ST_Geometry タイプ」	グループ内のすべてのジオメトリの空間的論理和を返します。	X	ベンダー拡張
ST_World メソッド	「ST_SpatialRefSys タイプ」	空間参照系内のすべてのポイントを表すジオメトリを返します。		ベンダー拡張

集約メソッドのリスト

次に、空間データで使用できる集約メソッドのリストを示します。曲面カラムの X は、 曲面空間参照系でもメソッドがサポートされていることを示しています。SQL/MM カラムには、 SQL/MM 標準 (ISO/IEC 13249-3: 2006) との準拠が反映されています。

メソッド	タイプ	説明	曲面	SQL/MM
ST_AsSVGAggr メソッド	「ST_Geometry タイプ」	グループ内のジオメトリをレンダリングした完全または部分的な SVG ドキュメントを返します。	X	ベンダー拡張

メソッド	タイプ	説明	曲面	SQL/MM
ST_ConvexHullAggr メソッド	「ST_Geometry タイプ」	グループ内のすべてのジオメトリの凸包を返します。		ベンダー拡張
ST_EnvelopeAggr メソッド	「ST_Geometry タイプ」	グループ内のすべてのジオメトリの外接矩形を返します。		ベンダー拡張
ST_GeomCollectionAggr メソッド	「ST_GeomCollection タイプ」	グループ内のすべてのジオメトリを含むジオメトリコレクションを返します。	X	ベンダー拡張
ST_IntersectionAggr メソッド	「ST_Geometry タイプ」	グループ内のすべてのジオメトリの空間的共通部分を返します。	X	ベンダー拡張
ST_LineStringAggr メソッド	「ST_LineString タイプ」	グループ内の順序付けされたポイントから構成された線ストリングを返します。	X	ベンダー拡張
ST_MultiCurveAggr メソッド	「ST_MultiCurve タイプ」	グループ内のすべての曲線を含む複数曲線を返します。	X	ベンダー拡張
ST_MultiLineStringAggr メソッド	「ST_MultiLineString タイプ」	グループ内のすべての線ストリングを含む複数線ストリングを返します。	X	ベンダー拡張
ST_MultiPointAggr メソッド	「ST_MultiPoint タイプ」	グループ内のすべてのポイントを含む複数ポイントを返します。	X	ベンダー拡張

メソッド	タイプ	説明	曲面	SQL/MM
ST_MultiPolygonAggr メソッド	「ST_MultiPolygon タイプ」	グループ内のすべての多角形を含む複数多角形を返します。	X	ベンダー拡張
ST_MultiSurfaceAggr メソッド	「ST_MultiSurface タイプ」	グループ内のすべての面を含む複数面を返します。	X	ベンダー拡張
ST_UnionAggr メソッド	「ST_Geometry タイプ」	グループ内のすべてのジオメトリの空間的論理和を返します。	X	ベンダー拡張

集合操作メソッドのリスト

次に、空間データで使用できる集合操作メソッドのリストを示します。曲面カラムの X は、曲面空間参照系でもメソッドがサポートされていることを示しています。SQL/MM カラムには、SQL/MM 標準 (ISO/IEC 13249-3: 2006) との準拠が反映されています。

メソッド	タイプ	説明	曲面	SQL/MM
ST_Difference メソッド	「ST_Geometry タイプ」	2 つのジオメトリの差集合を表すジオメトリ値を返します。	X	5.1.20
ST_Intersection メソッド	「ST_Geometry タイプ」	2 つのジオメトリの積集合を表すジオメトリ値を返します。	X	5.1.18
ST_IntersectionAggr メソッド	「ST_Geometry タイプ」	グループ内のすべてのジオメトリの空間的共通部分を返します。	X	ベンダー拡張
ST_SymDifference メソッド	「ST_Geometry タイプ」	2 つのジオメトリの対称差を表すジオメトリ値を返します。	X	5.1.21

メソッド	タイプ	説明	曲面	SQL/MM
ST_Union メソッド	「ST_Geometry タイプ」	2 つのジオメトリの和集合を表すジオメトリ値を返します。	X	5.1.19
ST_UnionAggr メソッド	「ST_Geometry タイプ」	グループ内のすべてのジオメトリの空間的論理和を返します。	X	ベンダー拡張

空間述部のリスト

次に、空間データで使用できる述部メソッドのリストを示します。曲面カラムの X は、 曲面空間参照系でもメソッドがサポートされていることを示しています。SQL/MM カラムには、 SQL/MM 標準 (ISO/IEC 13249-3: 2006) との準拠が反映されています。

メソッド	タイプ	説明	曲面	SQL/MM
ST_Contains メソッド	「ST_Geometry タイプ」	ジオメトリ値に別のジオメトリ値が空間的に含まれているかどうかをテストします。		5.1.31
ST_ContainsFilter メソッド	「ST_Geometry タイプ」	ジオメトリに別のジオメトリが含まれているかどうかの低コストのテスト。		ベンダー拡張
ST_CoveredBy メソッド	「ST_Geometry タイプ」	ジオメトリ値が別のジオメトリ値に空間的に含まれているかどうかをテストします。	X	ベンダー拡張
ST_CoveredByFilter メソッド	「ST_Geometry タイプ」	ジオメトリが別のジオメトリに含まれているかどうかの低コストのテスト。	X	ベンダー拡張

メソッド	タイプ	説明	曲面	SQL/MM
ST_Covers メソッド	「ST_Geometry タイプ」	ジオメトリ値に別のジオメトリ値が空間的に含まれているかどうかをテストします。	X	ベンダー拡張
ST_CoversFilter メソッド	「ST_Geometry タイプ」	ジオメトリに別のジオメトリが含まれているかどうかの低コストのテスト。	X	ベンダー拡張
ST_Crosses メソッド	「ST_Geometry タイプ」	ジオメトリ値が別のジオメトリ値と交差しているかどうかをテストします。		5.1.29
ST_Disjoint メソッド	「ST_Geometry タイプ」	ジオメトリ値が別のジオメトリ値から空間的に分断されているかどうかをテストします。	X	5.1.26
ST_Equals メソッド	「ST_Geometry タイプ」	ST_Geometry 値が別の ST_Geometry 値と空間的に等しいかどうかをテストします。	X	5.1.24
ST_EqualsFilter メソッド	「ST_Geometry タイプ」	ジオメトリが別のジオメトリと等しいかどうかの低コストのテスト。	X	ベンダー拡張
ST_Intersects メソッド	「ST_Geometry タイプ」	ジオメトリ値が別の値と空間的に交差しているかどうかをテストします。	X	5.1.27

メソッド	タイプ	説明	曲面	SQL/MM
ST_IntersectsFilter メソッド	「ST_Geometry タイプ」	2 つのジオメトリが交差しているかどうかの低コストのテスト。	X	ベンダー拡張
ST_IntersectsRect メソッド	「ST_Geometry タイプ」	ジオメトリが長方形と交差しているかどうかをテストします。	X	ベンダー拡張
ST_OrderingEquals メソッド	「ST_Geometry タイプ」	ジオメトリが別のジオメトリと同一であるかどうかをテストします。	X	5.1.43
ST_Overlaps メソッド	「ST_Geometry タイプ」	ジオメトリ値が別のジオメトリ値と重なり合うかどうかをテストします。		5.1.32

メソッド	タイプ	説明	曲面	SQL/MM
ST_Relate メソッド	「ST_Geometry タイプ」	ジオメトリ値が、交点マトリックスで指定されているとおりに別のジオメトリ値と空間的に関係しているかどうかをテストします。ST_Relate メソッドは、Dimensionally Extended 9 Intersection Model (DE-9IM) からの 9 文字の文字列を使用して、2 つの空間データ項目間のペアワイズの関係を示します。たとえば、ST_Relate メソッドは、ジオメトリ間に共通部分が存在するかどうかを確認し、存在する場合は結果の共通部分のジオメトリを調べます。参照： 「空間の関係操作」 55 ページ.		5.1.25、ベンダー拡張
ST_Touches メソッド	「ST_Geometry タイプ」	ジオメトリ値が別のジオメトリ値と空間的に接触しているかどうかをテストします。		5.1.28

メソッド	タイプ	説明	曲面	SQL/MM
ST_Within メソッド	「ST_Geometry タイプ」	ジオメトリ値が別のジオメトリ値内に空間的に含まれているかどうかをテストします。		5.1.30
ST_WithinDistance メソッド	「ST_Geometry タイプ」	2 つのジオメトリが指定の相互距離内にあるかどうかをテストします。	X	ベンダー拡張
ST_WithinDistanceFilter メソッド	「ST_Geometry タイプ」	2 つのジオメトリが指定距離内にあるかどうかを判定するための負荷の低い方法。	X	ベンダー拡張
ST_WithinFilter メソッド	「ST_Geometry タイプ」	ジオメトリが別のジオメトリ内にあるかどうかの低コストのテスト。		ベンダー拡張

索引

記号

A

B

D

E

G

I

J

K

N

O

S

U

W

あ

い

え

か

き

く

こ

さ

し

せ

そ

た

ち

て

と

な

ひ

ふ

へ

ほ

め

ら

り