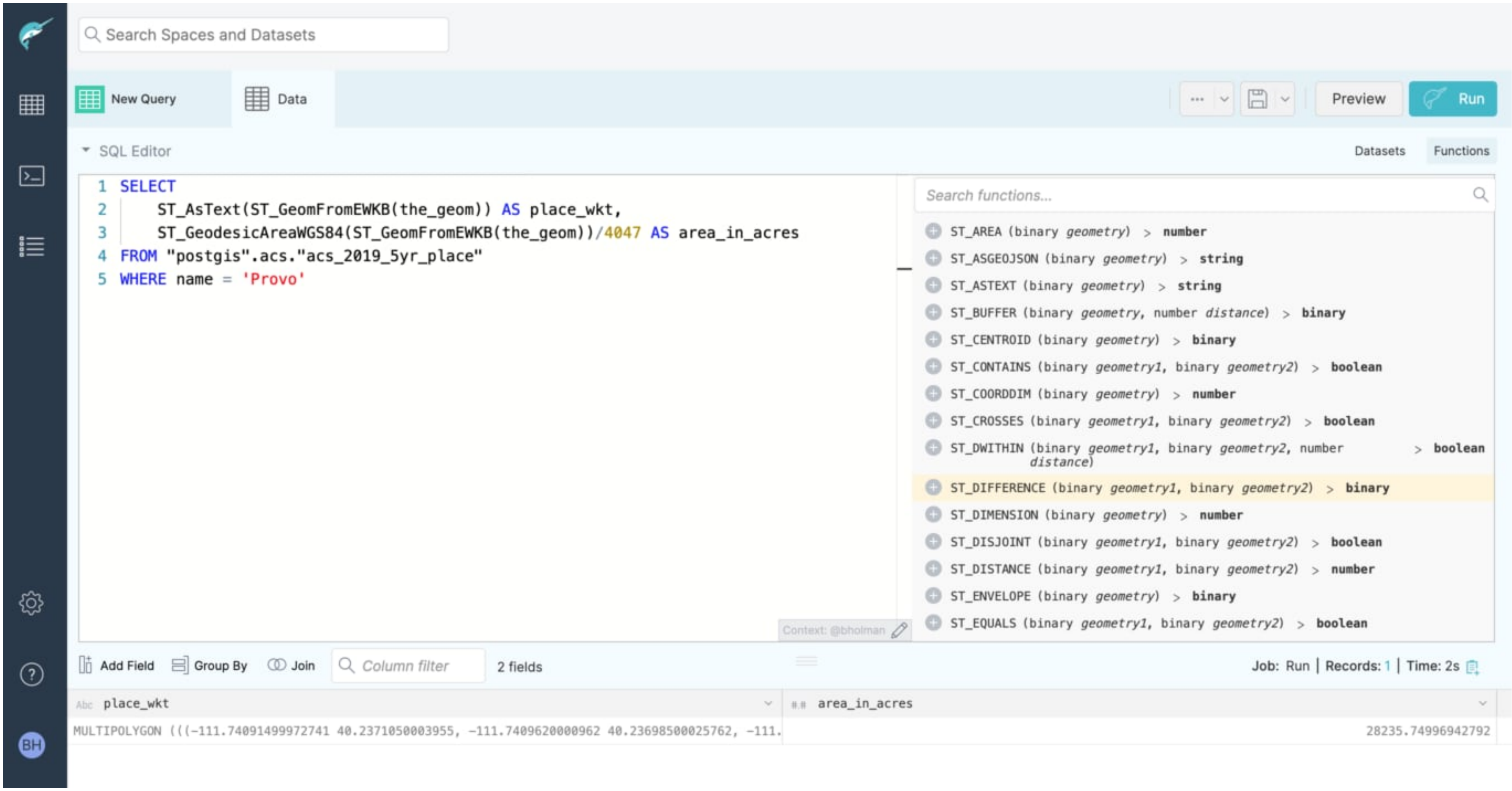
# GIS Extensions for Dremio - SQL Function Reference

## Authored by Brian Holman bholman@dezota.com

*This independent project is not affiliated with, sponsored, or endorsed by Dremio Corporation. Dremio is a registered trademark of Dremio Corporation and they retain all trademark and other intellectual property rights. "Dremio" is used here by reference to integrating with their published User-Defined Functions Specification for advanced users to develop their own custom functions for use in SQL queries.*



```
SELECT
    ST_AsText(ST_GeomFromEWKB(the_geom)) AS place_wkt,
    ST_GeodesicAreaWGS84(ST_GeomFromEWKB(the_geom))/4047 AS area_in_acres
FROM "postgis".acs."acs_2019_5yr_place"
WHERE name = 'Provo'
```

- ST_AREA (binary *geometry*) > **number**
- ST_ASGEOJSON (binary *geometry*) > **string**
- ST_ASTEXT (binary *geometry*) > **string**
- ST_BUFFER (binary *geometry*, number *distance*) > **binary**
- ST_CENTROID (binary *geometry*) > **binary**
- ST_CONTAINS (binary *geometry1*, binary *geometry2*) > **boolean**
- ST_COORDDIM (binary *geometry*) > **number**
- ST_CROSSES (binary *geometry1*, binary *geometry2*) > **boolean**
- ST_DWITHIN (binary *geometry1*, binary *geometry2*, number *distance*) > **boolean**
- ST_DIFFERENCE (binary *geometry1*, binary *geometry2*) > **binary**
- ST_DIMENSION (binary *geometry*) > **number**
- ST_DISJOINT (binary *geometry1*, binary *geometry2*) > **boolean**
- ST_DISTANCE (binary *geometry1*, binary *geometry2*) > **number**
- ST_ENVELOPE (binary *geometry*) > **binary**
- ST_EQUALS (binary *geometry1*, binary *geometry2*) > **boolean**

| place_wkt | area_in_acres |
|---|---|
| MULTIPOLYGON (((-111.74091499972741 40.2371050003955, -111.7409620000962 40.23698500025762, -111. | 28235.74996942792 |

Figure 1: DAC with GIS extensions

The **GIS Extensions** allow Dremio to perform standard GIS functions within Dremio SQL with 66 industry-standard GIS functions. These extensions use the *Esri Java Geometry Library* for the underlying implementation of the core geometry functions. The author made heavy use of Esri's *Spatial Framework for Hadoop* as a reference for a similar implementation that also relies on the same library.

There were two significant gaps in the Geometry Library supplied by Esri that limited transforming geometries from `EPSG: 4326` to other coordinate systems and performing geodesic rather than 2D area and length calculations. Geodesic area function helpers backing the `ST_GeodesicAreaWGS84` function are copied almost exactly from the *Trino Geospatial Library* as found in our `FunctionHelpers.stSphericalArea()` and `FunctionHelpers.computeSphericalExcess()`. Conversion to other coordinate systems in the `ST_Transform` function leverages the Proj4J Library. All of the referenced works are also published under the *Apache 2.0 License*.

## (1) ST_Area

### Definition

Returns the area of polygon or multipolygon

### Syntax

ST_Area(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Area(ST_GeomFromText('POLYGON ((0 0, 8 0, 0 8, 0 0), (1 1, 1 5, 5 1, 1 1))'))` | `24.0` |

## (2) ST_AsGeoJSON

### Definition

Returns the GeoJSON representation of *geometry*.

### Syntax

ST_AsGeoJSON(binary geometry)

### Return Type

`string`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsGeoJSON(ST_Point(1, 2))` | `'{"type":"Point","coordinates":[1,2],"crs":{"type":"name","properties":{"na` |

## (3) ST_AsText

### Definition

Returns the Well-Known Text (WKT) representation of *geometry*.

### Syntax

ST_AsText(binary geometry)

### Return Type

`string`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_Point(1, 2))` | `'POINT (1 2)'` |

## (4) ST_Buffer

### Definition

Returns geometry object that is the buffer surrounding source *geometry* at specified *distance*.

### Syntax

ST_Buffer(binary geometry, number distance)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Buffer(ST_Point(0, 0), 1)` | `polygon approximating a unit circle` |

## (5) ST_Centroid

### Definition

Takes a polygon, multipolygon, or multilinestring and returns the point that is in the center of the geometry's envelope. That means that the centroid point is halfway between the geometry's minimum and maximum x and y extents.

### Syntax

ST_Centroid(binary geometry)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_Centroid(ST_GeomFromText('point (2 3)')))` | `'POINT(2 3)'` |
| `SELECT ST_AsText(ST_Centroid(ST_GeomFromText('MULTIPOINT ((0 0), (1 1), (1 -1), (6 0))')))` | `'POINT(2 0)'` |
| `SELECT ST_AsText(ST_Centroid(ST_GeomFromText('linestring (0 0, 6 0)')))` | `'POINT(3 0)'` |
| `SELECT ST_AsText(ST_Centroid(ST_GeomFromText('POLYGON ((0 0, 0 8, 8 8, 8 0, 0 0))')))` | `'POINT(4 4)'` |
| `SELECT ST_AsText(ST_Centroid(ST_GeomFromText('POLYGON ((1 1, 5 1, 3 4))')))` | `'POINT(3 2)'` |

## (6) ST_Contains

### Definition

Returns true if *geometry1* contains *geometry2*.

### Syntax

ST_Contains(binary geometry1, binary geometry2)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Contains(ST_GeomFromText('POLYGON ((1 1, 1 4, 4 4, 4 1))'), ST_Point(2, 3))` | `true` |
| `SELECT ST_Contains(ST_GeomFromText('POLYGON ((1 1, 1 4, 4 4, 4 1))'), ST_Point(8, 8))` | `false` |

## (7) ST_CoordDim

### Definition

Returns count of coordinate components.

### Syntax

ST_CoordDim(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_CoordDim(ST_Point(1.5, 2.5))` | 2 |
| `SELECT ST_CoordDim(ST_GeomFromText('POINTZ (1.5 2.5 3)'))` | 3 |

## (8) ST_Crosses

### Definition

Returns true if *geometry1* crosses *geometry2*, otherwise false.

### Syntax

ST_Crosses(binary geometry1, binary geometry2)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Crosses(ST_GeomFromText('LINESTRING (0 0, 1 1)'), ST_GeomFromText('LINESTRING (1 0, 0 1)'))` | `true` |
| `SELECT ST_Crosses(ST_GeomFromText('LINESTRING (2 0, 2 3)'), ST_GeomFromText('POLYGON ((1 1, 1 4, 4 4, 4 1))'))` | `true` |
| `SELECT ST_Crosses(ST_GeomFromText('LINESTRING (0 2, 0 1)'), ST_GeomFromText('LINESTRING (2 0, 1 0)'))` | `false` |

## (9) ST_Difference

### Definition

Returns a geometry object that is the difference of the source objects.

### Syntax

ST_Difference(binary geometry1, binary geometry2)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_Difference(ST_GeomFromText('MULTIPOINT (1 1, 1.5 1.5, 2 2)'), ST_Point(1.5, 1.5)))` | `'MULTIPOINT ((1 1), (2 2))'` |
| `SELECT ST_AsText(ST_Difference(ST_GeomFromText('POLYGON ((0 0, 0 10, 10 10, 10 0))'), ST_GeomFromText('POLYGON ((0 0, 0 5, 5 5, 5 0))')))` | `'POLYGON ((5 0, 10 0, 10 10, 0 10, 0 5, 5 5, 5 0))'` |

## (10) ST_Dimension

### Definition

Returns spatial dimension of geometry.

### Syntax

ST_Dimension(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Dimension(ST_Point(1.5, 2.5))` | `0` |
| `SELECT ST_Dimension(ST_GeomFromText('LINESTRING (1.5 2.5, 3.0 2.2)'))` | `1` |
| `SELECT ST_Dimension(ST_GeomFromText('POLYGON ((2 0, 2 3, 3 0))'))` | `2` |

## (11) ST_Disjoint

### Definition

Returns true if the intersection of the two geometries produces an empty set; otherwise, it returns false.

### Syntax

ST_Disjoint(binary geometry1, binary geometry2)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Disjoint(ST_GeomFromText('LINESTRING (0 0, 0 1)'), ST_GeomFromText('LINESTRING (1 1, 1 0)'))` | `true` |
| `SELECT ST_Disjoint(ST_GeomFromText('LINESTRING (0 0, 1 1)'), ST_GeomFromText('LINESTRING (1 0, 0 1)'))` | `false` |

## (12) ST_Distance

### Definition

Returns the distance between two geometry objects.

### Syntax

ST_Distance(binary geometry1, binary geometry2)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Distance(ST_Point(0.0,0.0), ST_Point(3.0,4.0))` | `5.0` |

## (13) ST_DWithin

### Definition

Returns true if the two geometries are within the specified distance of one another; otherwise, it returns false.

### Syntax

ST_DWithin(binary geometry1, binary geometry2, number distance)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_DWithin(ST_GeomFromText('POLYGON ((10.02 20.01, 11.92 35.64, 25.02 34.15, 19.15 33.94, 10.02 20.01))'), ST_Point (1,2),100)` | `true` |
| `SELECT ST_DWithin(ST_GeomFromText('POLYGON ((101.02 200.01, 111.92 350.64, 250.02 340.15, 190.15 330.94, 101.02 200.01))'), ST_Point (10.02,20.01), 100)` | `false` |

## (14) ST_Envelope

### Definition

Returns the minimum bounding box of the geometry object as a polygon

### Syntax

ST_Envelope(binary geometry)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_Envelope(ST_GeomFromText('LINESTRING (0 0, 2 2))')))` | `'POLYGON ((0 0, 2 0, 2 2, 0 2, 0 0))'` |
| `SELECT ST_AsText(ST_Envelope(ST_GeomFromText('POLYGON ((2 0, 2 3, 3 0))')))` | `'POLYGON ((2 0, 3 0, 3 3, 2 3, 2 0))'` |

## (15) ST_Equals

### Definition

Returns true if the two geometries occupy the same space even if they have a different number of vertices, otherwise it returns false.

### Syntax

ST_Equals(binary geometry1, binary geometry2)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Equals(ST_GeomFromText('LINESTRING (0 0, 1 1)'),ST_GeomFromText('LINESTRING (1 1, 0 0)'))` | `true` |
| `SELECT ST_Equals(ST_GeomFromText('LINESTRING (0 0, 1 1)'),ST_GeomFromText('LINESTRING (1 0, 0 1)'))` | `false` |
| `SELECT ST_Equals(ST_GeomFromText('LINESTRING (0 0, 3 3)'),ST_GeomFromText('LINESTRING (3 3, 2 2, 1 1, 0 0)'))` | `true` |

## (16) ST_ExteriorRing

### Definition

Returns the exterior ring of a polygon as a linestring.

### Syntax

ST_ExteriorRing(binary geometry)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_ExteriorRing(ST_GeomFromText('POLYGON ((1 1, 1 4, 4 1))')))` | `'LINESTRING (1 1, 4 1, 1 4, 1 1)'` |
| `SELECT ST_AsText(ST_ExteriorRing(ST_GeomFromText('POLYGON ((0 0, 8 0, 0 8, 0 0), (1 1, 1 5, 5 1, 1 1))')))` | `'LINESTRING (0 0, 8 0, 0 8, 0 0)'` |

## (17) ST_Generalize

### Definition

Simplifies geometries using the Douglas-Peucker algorithm. *maxDeviation* is the maximum allowed deviation from the generalized geometry to the original geometry. When *removeDegenerateParts* is true, the degenerate parts of the geometry will be removed from the output.

### Syntax

ST_Generalize(binary geometry, number maxDeviation, boolean removeDegenerateParts)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_Generalize(ST_GeomFromText('POLYGON ((0 0, 1 1, 2 0, 3 2, 4 1, 5 0, 5 10, 0 10))'), 2, true))` | `'POLYGON ((0 0, 5 0, 5 10, 0 10, 0 0))'` |

## (18) ST_GeodesicAreaWGS84

### Definition

Returns the area in square meters of a geometry on the Earth's surface using spherical model. Requires the geometry to be in the WGS84 spatial reference.

### Syntax

ST_GeodesicAreaWGS84(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_GeodesicAreaWGS84(ST_GeomFromText('POLYGON ((-114.04702599994988 39.90609700007656, -114.0500520000997 37.0001909997149, -109.04517199998776 36.99897700038832, -109.05002599989996 41.000691000389395, -111.04681499981234 40.997875000031286, -111.04671399965133 42.00170200004732, -114.04147700036322 41.99387299963928, -114.04702599994988 39.90609700007656))'))/4047 AS utah_acreage` | `5.416484897473004E7` |

## (19) ST_GeodesicLengthWGS84

### Definition

Returns distance along line on WGS84 spheroid, in meters, for geographic coordinates. Requires the geometry to be in the WGS84 spatial reference.

### Syntax

ST_GeodesicLengthWGS84(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT SELECT ST_GeodesicLengthWGS84(ST_GeomFromText('MultiLineString((0.0 80.0, 0.3 80.4))', 4326))` | `45026.96274781222` |

## (20) ST_GeometryN

### Definition

Takes a geometry collection and an integer index (1-based index) and returns the nth geometry object in the collection.

### Syntax

ST_GeometryN(binary geometry, number index)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_GeometryN(ST_GeomFromText('MULTIPOINT (10 40, 40 30, 20 20, 30 10)'), 3))` | `'POINT (20 20)'` |
| `SELECT ST_AsText(ST_GeometryN(ST_GeomFromText('MULTILINESTRING ((2 4, 10 10), (20 20, 7 8))'), 2))` | `'LINESTRING (20 20, 7 8)'` |

## (21) ST_GeometryType

### Definition

Takes a geometry object and returns its geometry type (for example, Point, Line, Polygon, MultiPoint) as a string.

### Syntax

ST_GeometryType(binary geometry)

### Return Type

`string`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_GeometryType(ST_Point(1.5, 2.5))` | `'ST_POINT'` |
| `SELECT ST_GeometryType(ST_GeomFromText('LINESTRING (1.5 2.5, 3.0 2.2)'))` | `'ST_LINESTRING'` |
| `SELECT ST_GeometryType(ST_GeomFromText('POLYGON ((2 0, 2 3, 3 0))'))` | `'ST_POLYGON'` |

## (22) ST_GeomFromEWKB

### Definition

Converts a Hex encoded binary string from Postgres/PostGIS geometry to native geometry including embedded SRID.

### Syntax

ST_GeomFromEWKB(string hexEncodedGeometry)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT SELECT ST_AsText(ST_GeomFromEWKB(the_geom)) FROM table("postgis".external_query('SELECT ST_GeomFromText(''POINT(-71.064544 42.28787)'',4326) AS the_geom'))` | `'POINT (-71.064544 42.28787)'` |

## (23) ST_GeomFromGeoJSON

### Definition

Constructs a geometry from GeoJSON.

### Syntax

ST_GeomFromGeoJSON(string geoJsonString)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_GeomFromGeoJSON('{"type":"Point", "coordinates":[1.2, 2.4]}'))` | `'POINT (1.2 2.4)'` |
| `SELECT ST_AsText(ST_GeomFromGeoJSON('{"type":"LineString", "coordinates":[[1,2], [3,4]]}'))` | `'LINESTRING (1 2, 3 4)'` |

## (24) ST_GeomFromText

### Definition

Takes a well-known text representation and returns a geometry object.

### Syntax

ST_GeomFromText(string wktString)

### Return Type

`binary`

## (25) ST_GeomFromText

### Definition

Takes a well-known text representation and a spatial reference ID and returns a geometry object.

### Syntax

ST_GeomFromText(string wktString, number SRID)

### Return Type

`binary`

---

## (26) ST_GeomFromWKB

### Definition

Takes a well-known binary (WKB) representation and returns a geometry object.

### Syntax

ST_GeomFromWKB(binary wkbValue)

### Return Type

`binary`

---

## (27) ST_GeomFromWKB

### Definition

Takes a well-known binary (WKB) representation and a spatial reference ID and returns a geometry object.

### Syntax

ST_GeomFromWKB(binary wkbValue, number SRID)

### Return Type

`binary`

---

## (28) ST_GeoSize

### Definition

Takes a geometry object and returns its size in bytes.

### Syntax

ST_GeoSize(binary geometry)

### Return Type

`number`

---

## (29) ST_InteriorRingN

### Definition

Returns a LineString which is the nth interior ring of the input Polygon (1-based index)

### Syntax

ST_InteriorRingN(binary geometry, number index)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_InteriorRingN(ST_GeomFromText('polygon ((0 0, 8 0, 0 8, 0 0), (1 1, 1 5, 5 1, 1 1))'), 1))` | `'LINESTRING (1 1, 1 5, 5 1, 1 1)'` |

## (30) ST_Intersection

### Definition

Returns a geometry object that is the geometric intersection of the source objects.

### Syntax

ST_Intersection(binary geometry1, binary geometry2)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_Intersection(ST_Point(1,1), ST_Point(1,1)))` | `'POINT (1 1)'` |
| `SELECT ST_AsText(ST_Intersection(ST_GeomFromText('LINESTRING(0 2, 0 0, 2 0)'), ST_GeomFromText('LINESTRING(0 3, 0 1, 1 0, 3 0)')))` | `'MULTILINESTRING ((1 0, 2 0), (0 2, 0 1))'` |
| `SELECT ST_AsText(ST_Intersection(ST_GeomFromText('POLYGON ((2 0, 2 3, 3 0))'), ST_GeomFromText('POLYGON ((1 1, 4 1, 4 4, 1 4))')))` | `'POLYGON ((2 1, 2.666666666666667 1, 2 3, 2 1))'` |

## (31) ST_Intersects

### Definition

Returns true if *geometry1* intersects with *geometry2*, otherwise returns false.

### Syntax

ST_Intersects(binary geometry1, binary geometry2)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Intersects(ST_GeomFromText('LINESTRING (2 0, 2 3)'), ST_GeomFromText('POLYGON ((1 1, 4 1, 4 4, 1 4))'))` | `true` |
| `SELECT ST_Intersects(ST_GeomFromText('LINESTRING (8 7, 7 8)'), ST_GeomFromText('POLYGON ((1 1, 4 1, 4 4, 1 4))'))` | `false` |

## (32) ST_Is3D

### Definition

Returns true if the geometry object is three-dimensional including height ‘Z’, otherwise returns false.

### Syntax

ST_Is3D(binary geometry)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Is3D(ST_GeomFromText('POLYGON ((1 1, 1 4, 4 4, 4 1))'))` | `false` |
| `SELECT ST_Is3D(ST_GeomFromText('LINESTRING (0 0, 3 4, 0 4, 0 0)'))` | `false` |
| `SELECT ST_Is3D(ST_Point(3, 4))` | `false` |
| `SELECT ST_Is3D(ST_PointZ(3, 4, 2))` | `true` |

## (33) ST_IsClosed

### Definition

Return true if the linestring or multi-line has start and end points that are coincident.

### Syntax

ST_IsClosed(binary geometry)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_IsClosed(ST_GeomFromText('LINESTRING(0 0, 3 4, 0 4, 0 0)'))` | `true` |
| `SELECT ST_IsClosed(ST_GeomFromText('LINESTRING(0 0, 3 4)'))` | `false` |

## (34) ST_IsEmpty

### Definition

Return true if the geometry object is empty of geometric information.

### Syntax

ST_IsEmpty(binary geometry)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_IsEmpty(ST_Point(1.5, 2.5))` | `false` |
| `SELECT ST_IsEmpty(ST_GeomFromText('POINT EMPTY'))` | `true` |

## (35) ST_IsMeasured

### Definition

Returns true if the geometry object is measured including an additional dimension ‘M’, otherwise returns false.

### Syntax

ST_IsMeasured(binary geometry)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_IsMeasured(ST_PointZ(3, 4, 2))` | `false` |
| `SELECT ST_IsMeasured(ST_GeomFromText('POINT M (1 1 80)'))` | `true` |
| `SELECT ST_IsMeasured(ST_GeomFromText('POINT ZM (1 1 5 60)'))` | `true` |

## (36) ST_IsRing

### Definition

Returns true if the geometry is a linestring and the linestring is closed and simple.

### Syntax

ST_IsRing(binary geometry)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_IsRing(ST_GeomFromText('LINESTRING (0 0, 3 4, 0 4, 0 0)'))` | `true` |
| `SELECT ST_IsRing(ST_GeomFromText('LINESTRING (0 0, 1 1, 1 2, 2 1, 1 1, 0 0)'))` | `false` |
| `SELECT ST_IsRing(ST_GeomFromText('LINESTRING (0 0, 3 4)'))` | `false` |

## (37) ST_IsSimple

### Definition

Returns true if the geometry object is simple as defined by the Open Geospatial Consortium (OGC), otherwise, it returns false

### Syntax

ST_IsSimple(binary geometry)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_IsSimple(ST_Point(1.5, 2.5))` | `true` |
| `SELECT ST_IsSimple(ST_GeomFromText('LINESTRING (0 0, 1 1, 0 1, 1 0)'))` | `false` |

## (38) ST_Length

### Definition

Returns the length of a line string or multiline string.

### Syntax

ST_Length(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Length(ST_GeomFromText('LINESTRING (0 0, 3 4)'))` | 5.0 |
| `SELECT ST_Length(ST_GeomFromText('MULTILINESTRING ((1 0, 2 0), (0 2, 0 1))'))` | 2.0 |

## (39) ST_M

### Definition

Takes a Point as an input parameter and returns its measure m-coordinate.

### Syntax

ST_M(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_M(ST_GeomFromText('POINT M (1 1 80)'))` | 80.0 |
| `SELECT ST_M(ST_GeomFromText('POINT ZM (1 1 5 60)'))` | 60.0 |

## (40) ST_MaxM

### Definition

Takes a geometry as an input parameter and returns its maximum measure m-coordinate.

### Syntax

ST_MaxM(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_MaxM(ST_GeomFromText('LINESTRING M (1.5 2.5 2, 3.0 2.2 1)'))` | `2.0` |
| `SELECT ST_MaxM(ST_GeomFromText('POINT M (1.5 2.5 3)'))` | `3.0` |

## (41) ST_MaxX

### Definition

Takes a geometry as an input parameter and returns its maximum x-coordinate.

### Syntax

ST_MaxX(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_MaxX(ST_GeomFromText('LINESTRING M (1.5 2.5 2, 3.0 2.2 1)'))` | `3.0` |
| `SELECT ST_MaxX(ST_GeomFromText('POINT M (1.5 2.5 3)'))` | `1.5` |

## (42) ST_MaxY

### Definition

Takes a geometry as an input parameter and returns its maximum y-coordinate.

### Syntax

ST_MaxY(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| SELECT ST_MaxY(ST_GeomFromText('LINESTRING M (1.5 2.5 2, 3.0 2.2 1)')) | 2.5 |
| SELECT ST_MaxY(ST_GeomFromText('POINT M (1.5 2.5 3)')) | 2.5 |

## (43) ST_MaxZ

### Definition

Takes a geometry as an input parameter and returns its maximum z-coordinate.

### Syntax

ST_MaxZ(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| SELECT ST_MaxZ(ST_GeomFromText('LINESTRING ZM (1.5 2.5 2 60, 3.0 2.2 1 80)')) | 2.0 |
| SELECT ST_MaxZ(ST_GeomFromText('LINESTRING Z (1.5 2.5 3, 3.0 2.2 4)')) | 4.0 |

## (44) ST_MinM

### Definition

Takes a geometry as an input parameter and returns its minimum m-coordinate.

### Syntax

ST_MinM(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_MinM(ST_GeomFromText('LINESTRING M (1.5 2.5 2, 3.0 2.2 1)'))` | `1.0` |
| `SELECT ST_MinM(ST_GeomFromText('POINT M (1.5 2.5 3)'))` | `3.0` |

## (45) ST_MinX

### Definition

Takes a geometry as an input parameter and returns its minimum x-coordinate.

### Syntax

ST_MinX(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_MinX(ST_GeomFromText('LINESTRING M (1.25 2.5 2, 3.0 2.2 1)'))` | `1.25` |
| `SELECT ST_MinX(ST_GeomFromText('POINT M (1.75 2.5 3)'))` | `1.75` |

## (46) ST_MinY

### Definition

Takes a geometry as an input parameter and returns its minimum y-coordinate.

### Syntax

ST_MinY(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_MinY(ST_GeomFromText('LINESTRING M (1.5 2.5 2, 3.0 2.2 1)'))` | `2.2` |
| `SELECT ST_MinY(ST_GeomFromText('POINT M (1.5 2.25 3)'))` | `2.25` |

## (47) ST_MinZ

### Definition

Takes a geometry as an input parameter and returns its minimum z-coordinate.

### Syntax

ST_MinZ(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_MinZ(ST_GeomFromText('LINESTRING ZM (1.5 2.5 2 60, 3.0 2.2 1 80)'))` | `1.0` |
| `SELECT ST_MinZ(ST_GeomFromText('LINESTRING Z (1.5 2.5 3, 3.0 2.2 4)'))` | `3.0` |

## (48) ST_NumGeometries

### Definition

Returns the number of geometries in the geometry collection.

### Syntax

ST_NumGeometries(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_NumGeometries(ST_GeomFromText('MULTIPOINT ((10 40), (40 30), (20 20), (30 10))'))` | 4 |
| `SELECT ST_NumGeometries(ST_GeomFromText('MULTILINESTRING ((2 4, 10 10), (20 20, 7 8))'))` | 2 |

## (49) ST_NumInteriorRing

### Definition

Returns the number of interior rings in the polygon geometry.

### Syntax

ST_NumInteriorRing(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_NumInteriorRing(ST_GeomFromText('POLYGON ((0 0, 8 0, 0 8, 0 0), (1 1, 1 5, 5 1, 1 1))'))` | 1 |

## (50) ST_NumPoints

### Definition

Returns the number of points (vertices) in the geometry. For polygons, both the starting and ending vertices are counted, even though they occupy the same location.

### Syntax

ST_NumPoints(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_NumPoints(ST_Point(1.5, 2.5))` | `1` |
| `SELECT ST_NumPoints(ST_GeomFromText('LINESTRING (1.5 2.5, 3.0 2.2)'))` | `2` |
| `SELECT ST_NumPoints((ST_GeomFromText('POLYGON ((0 0, 10 0, 0 10, 0 0))')))` | `4` |

## (51) ST_Overlaps

### Definition

Returns true if *geometry1* overlaps *geometry2*.

### Syntax

ST_Overlaps(binary geometry1, binary geometry2)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Overlaps(ST_GeomFromText('POLYGON ((2 0, 2 3, 3 0))'), ST_GeomFromText('POLYGON ((1 1, 1 4, 4 4, 4 1))'))` | `true` |
| `SELECT ST_Overlaps(ST_GeomFromText('POLYGON ((2 0, 2 1, 3 1))'), ST_GeomFromText('POLYGON ((1 1, 1 4, 4 4, 4 1))'))` | `false` |

## (52) ST_Point

### Definition

Returns a 2D point geometry from the provided lon (x) and lat (y) values.

### Syntax

ST_Pointnumber lon, number lat

### Return Type

`binary`

## (53) ST_PointN

### Definition

Returns the point that is the nth vertex in an LineString or MultiPoint (1-based index)

### Syntax

ST_PointN(binary geometry, number index)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_PointN(ST_GeomFromText('LINESTRING (1.5 2.5, 3.0 2.2)'), 2))` | `'POINT (3 2.2)'` |

## (54) ST_PointZ

### Definition

Returns a 3D point geometry from the provided lon (x), lat (y), and elev (z) values.

### Syntax

ST_PointZnumber lon, number lat, number elev

### Return Type

`binary`

---

## (55) ST_Relate

### Definition

Compares the two geometries and returns true if the geometries meet the conditions specified by the DE-9IM pattern matrix string, otherwise, false is returned.

### Syntax

ST_Relate(binary geometry1, binary geometry2, string relation)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Relate(ST_GeomFromText('POLYGON ((2 0, 2 1, 3 1))'), ST_GeomFromText('POLYGON ((1 1, 1 4, 4 4, 4 1))'), '****T****')` | `true` |
| `SELECT ST_Relate(ST_GeomFromText('POLYGON ((2 0, 2 1, 3 1))'), ST_GeomFromText('POLYGON ((1 1, 1 4, 4 4, 4 1))'), 'T********')` | `false` |
| `SELECT ST_Relate(ST_GeomFromText('LINESTRING (0 0, 3 3)'), ST_GeomFromText('LINESTRING (1 1, 4 4)'), 'T********')` | `true` |
| `SELECT ST_Relate(ST_GeomFromText('LINESTRING (0 0, 3 3)'), ST_GeomFromText('LINESTRING (1 1, 4 4)'), '****T****')` | `false` |

## (56) ST_SetSRID

### Definition

Sets the Spatial Reference ID of *SRID* of the geometry.

### Syntax

ST_SetSRID(binary geometry, number SRID)

### Return Type

`binary`

## (57) ST_Simplify

### Definition

Simplifies the geometry or determines if the geometry is simple. The goal is to produce a geometry that is valid to store without additional processing.

### Syntax

ST_Simplify(binary geometry)

### Return Type

`binary`

## (58) ST_StartPoint

### Definition

Returns the first point of a Linestring.

### Syntax

ST_StartPoint(binary geometry)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_StartPoint(ST_GeomFromText('LINESTRING (1.5 2.5, 3.0 2.2)')))` | `'POINT(1.5 2.5)'` |

## (59) ST_SymmetricDiff

### Definition

Returns a geometry object that is the symmetric difference of the source objects.

### Syntax

ST_SymmetricDiff(binary geometry1, binary geometry2)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_SymmetricDiff(ST_GeomFromText('LINESTRING (0 2, 2 2)'), ST_GeomFromText('LINESTRING (1 2, 3 2)')))` | `'MULTILINESTRING ((0 2, 1 2), (2 2, 3 2))'` |
| `SELECT ST_AsText(ST_SymmetricDiff(ST_GeomFromText('POLYGON ((0 0, 2 0, 2 2, 0 2, 0 0))'), ST_GeomFromText('POLYGON ((1 1, 3 1, 3 3, 1 3, 1 1))'))) --> 'MULTIPOLYGON (((0 0, 2 0, 2 1, 1 1, 1 2, 0 2, 0 0)), ((2 1, 3 1, 3 3, 1 3, 1 2, 2 2, 2 1)))'` | `undefined` |

## (60) ST_Touches

### Definition

Returns true if none of the points common to both geometries intersect the interiors of both geometries, otherwise, it returns false. At least one geometry must be a LineString, Polygon, MultiLineString, or MultiPolygon.

### Syntax

ST_Touches(binary geometry1, binary geometry2)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Touches(ST_Point(1, 2), ST_GeomFromText('POLYGON ((1 1, 1 4, 4 4, 4 1))'))` | `true` |
| `SELECT ST_Touches(ST_Point(8, 8), ST_GeomFromText('POLYGON ((1 1, 1 4, 4 4, 4 1))'))` | `false` |

## (61) ST_Transform

### Definition

Takes the two-dimensional geometry as input and returns values converted from the spatial source reference specified by *sourceSRID* to the one specified by *targetSRID*.

### Syntax

ST_Transform(binary geometry, number sourceSRID, number targetSRID)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_Transform(ST_GeomFromText('POLYGON ((-114.04702599994988 39.90609700007656, -114.0500520000997 37.0001909997149, -109.04517199998776 36.99897700038832, -109.05002599989996 41.000691000389395, -111.04681499981234 40.997875000031286, -111.04671399965133 42.00170200004732, -114.04147700036322 41.99387299963928, -114.04702599994988 39.90609700007656))'), 4326, 3857))` | `'POLYGON ((-12695656.860801652 4852305.919673687, -12695993.71359747 4439133.410181124, -12138853.020503571 4438964.195256694, -12139393.365302108 5012443.58678148, -12361674.899993964 5012028.231889712, -12361663.65670747 5161234.398812287, -12695039.148993252 5160061.69329091, -12695656.860801652 4852305.919673687))'` |

## (62) ST_Union

### Definition

Returns a geometry as the union of the two supplied geometries.

### Syntax

ST_Union(binary geometry1, binary geometry2)

### Return Type

`binary`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_AsText(ST_Union(ST_GeomFromText('POLYGON ((1 1, 1 4, 4 4, 4 1))'), ST_GeomFromText('POLYGON ((4 1, 4 4, 4 8, 8 1))')))` | `'POLYGON ((1 1, 4 1, 8 1, 4 8, 4 4, 1 4, 1 1))'` |

## (63) ST_Within

### Definition

Returns true if *geometry1* is completely inside *geometry2*.

### Syntax

ST_Within(binary geometry1, binary geometry2)

### Return Type

`boolean`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Within(ST_Point(2, 3), ST_GeomFromText('POLYGON ((1 1, 1 4, 4 4, 4 1))'))` | `true` |
| `SELECT ST_Within(ST_Point(8, 8), ST_GeomFromText('POLYGON ((1 1, 1 4, 4 4, 4 1))'))` | `false` |

## (64) ST_X

### Definition

Takes a Point as an input parameter and returns its longitude (x) coordinate.

### Syntax

ST_X(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_X(ST_Point(5, 7))` | `5.0` |

## (65) ST_Y

### Definition

Takes a Point as an input parameter and returns its latitude (y) coordinate.

### Syntax

ST_Y(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Y(ST_GeomFromText('POINT (5 7)'))` | `7.0` |

## (66) ST_Z

### Definition

Takes a Point as an input parameter and returns its elevation (z) coordinate.

### Syntax

ST_Z(binary geometry)

### Return Type

`number`

### Examples

| Query | Result |
|---|---|
| `SELECT ST_Z(ST_GeomFromText('POINT Z (5 7 9)'))` | `9.0` |