

# Thomas Hütter

Data Saturday  
Croatia 2021

## A refresher on geospatial data in SQL Server

# A refresher on geospatial data in SQL Server

Thomas Hütter, Diplom-Betriebswirt

- Application developer, consultant, accidental DBA, author
- Worked at consultancies, ISVs, end user companies
- SQL Server > 6.5, Dynamics Nav > 3.0, R > 3.1.2
- Speaker at SQL events around Europe



 @DerFredo <https://twitter.com/DerFredo>

 [de.linkedin.com/in/derfredo](https://de.linkedin.com/in/derfredo)

 [www.xing.com/profile/Thomas\\_Huetter](https://www.xing.com/profile/Thomas_Huetter)



Pure expertise at the  
SQL Server Konferenz 2018  
FEB 26th - 28th, 2018 | DARMSTADT

Sign up now at [sqlkonferenz.de](https://sqlkonferenz.de)



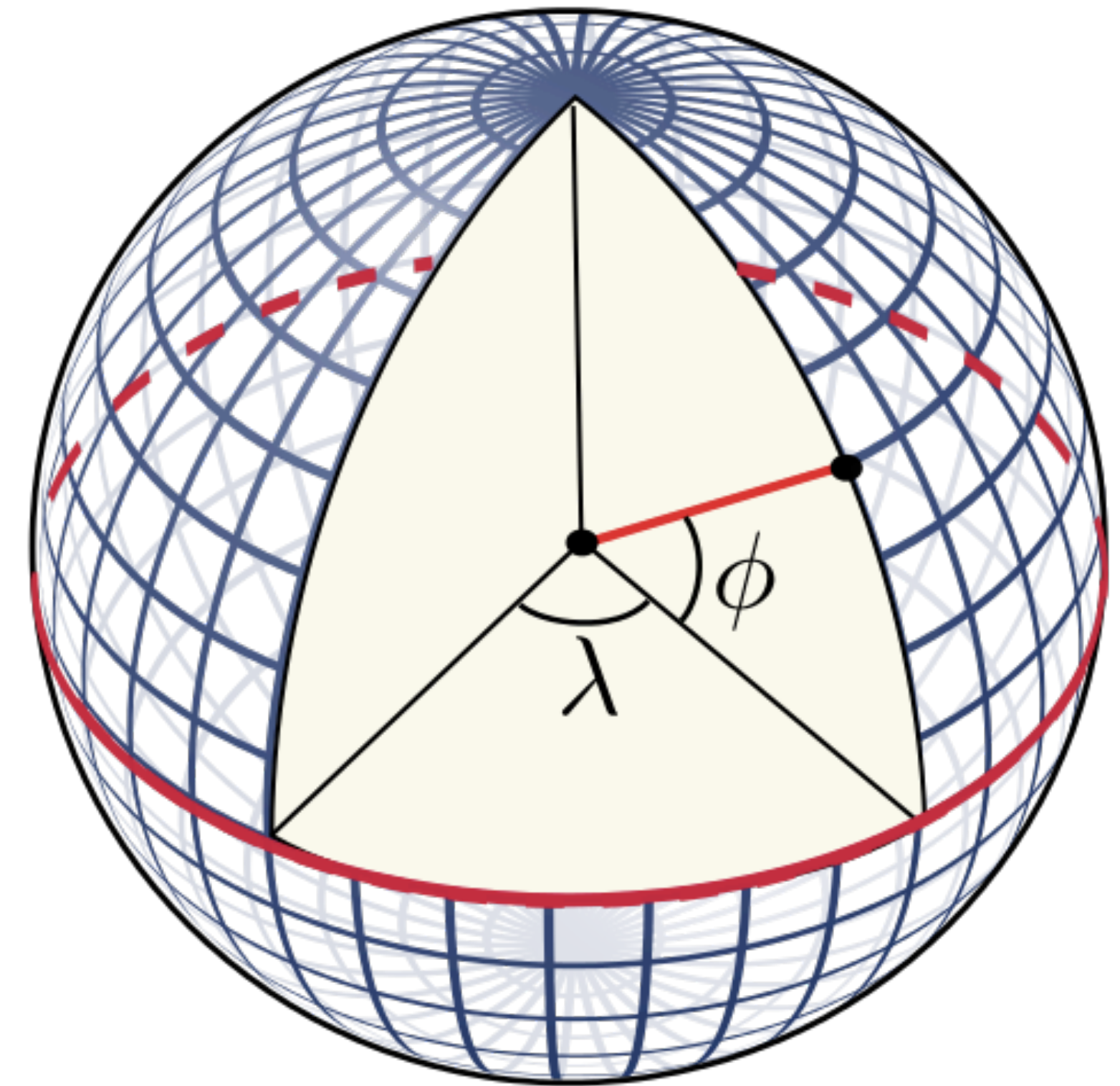
# Agenda

- The concept of geospatial data
- History of geospatials in SQL Server
- From 0 to 2 dimensions: spatial types overview
- Getting spatial data into and out of SQL tables
- Functions, functions, functions...
- Practical applications
- Round-up; resources & credits; Q&A

# The concept of geospatial data

Everything has a position (on the earth),  
purposes include visualization, analysis, design

- Geographic data
  - position on the spheric surface of the earth
  - coordinates in degrees latitude + longitude
  - addresses, roads, cities, districts, countries...

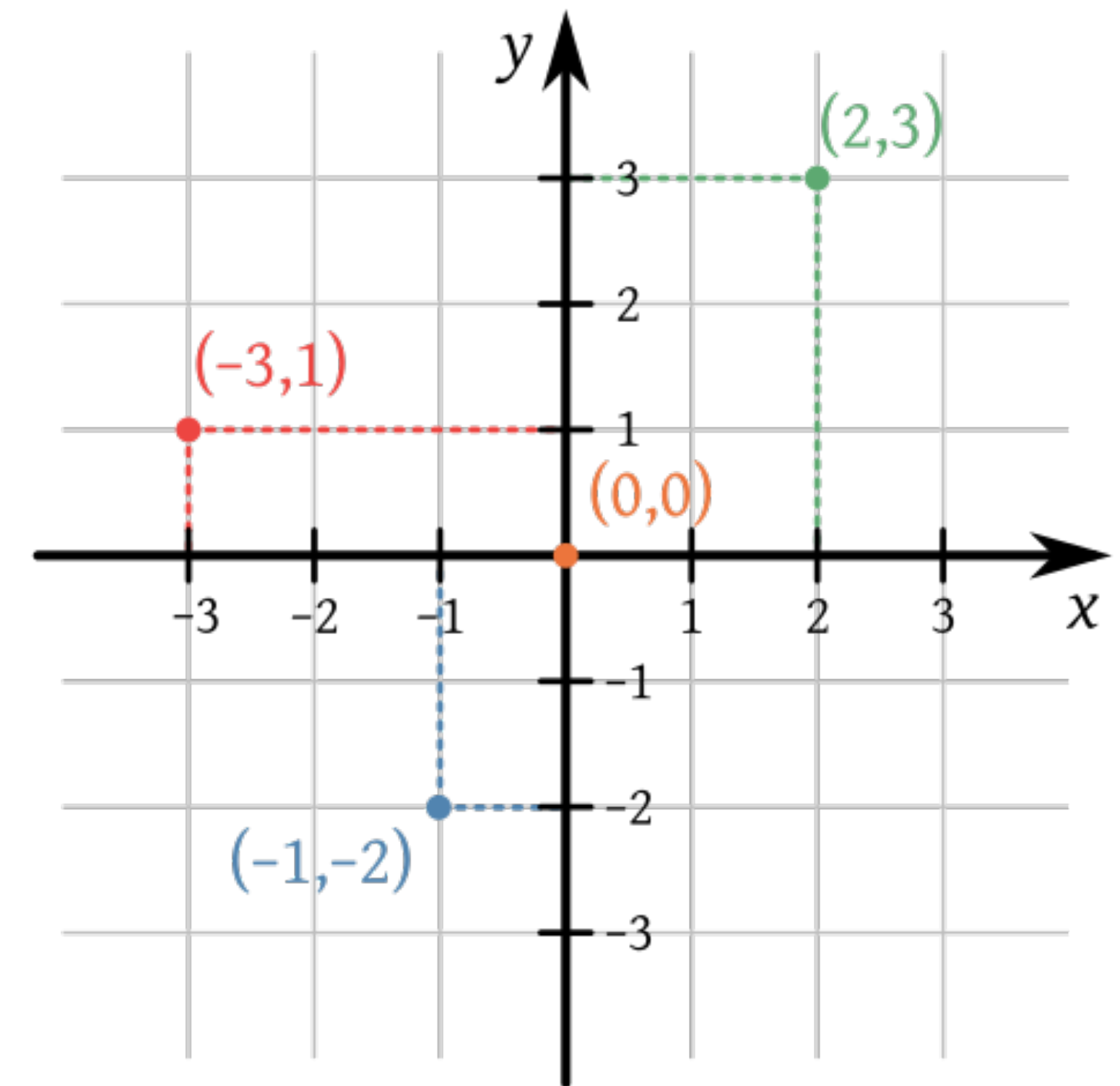




# The concept of geospatial data

Euclidian geometry dealing with points, lines, shapes, (bodies) in a Cartesian system

- Geometric data
  - position on a planar surface
  - coordinates in distance units X, Y
  - shop floor layout, warehouse, furniture...



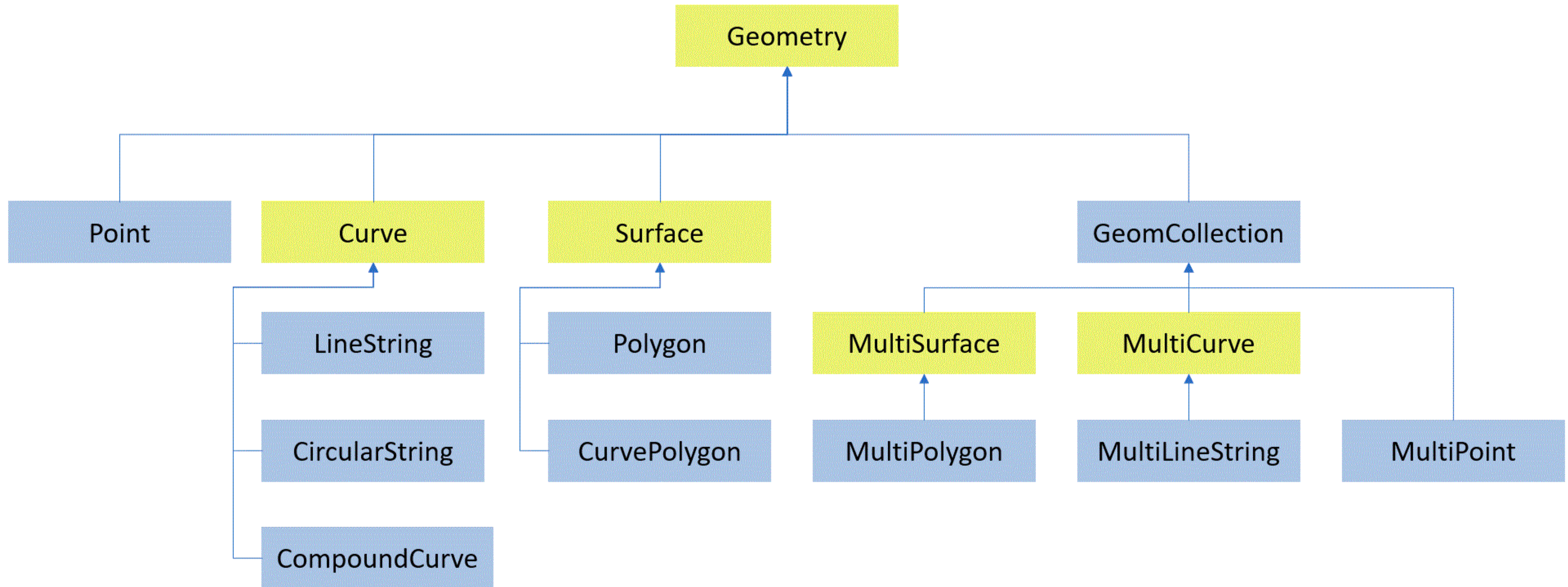
# History of geospatials in SQL Server

## SQL Server Versions with geospatial news

- 2008: New native geometry and geography data types and functions
- 2012: Enhancements: everything curved and „full globe“, aggregate functions, improvements in performance and precision
- 2014: . . .
- 2016: . . .
- 2017: . . .
- 2019: . . .



# From 0 to 2 dimensions: spatial types overview



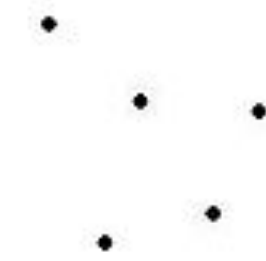


# From 0 to 2 dimensions: spatial types overview

- 0 dimensions

Point: defined by a single pair of coordinate values

MultiPoint: collection of Points



- 1 dimension

LineString: straight path segments connecting 2 or more points

CircularString: arc shaped line connecting 3 or more points

CompoundCurve: continuous curve between a set of points (Line or CircularString)

MultiLineString: collections of LineStrings

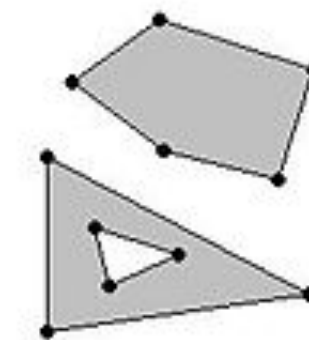


- 2 dimensions

Polygons: area defined by (at least) an outer closed LineString

CurvePolygons: area of LineString, CircularString or CompoundCurve

MultiPolygon: collection of Polygons



- Special cases

FullGlobe: represents the whole surface of the earth

Empty geometries: geoms not containing any objects



# From 0 to 2 dimensions: spatial types overview

CLR implementation, follows Open geospatial consortium (OGC) standards.

To make things comparable / relatable, we need a unified reference system

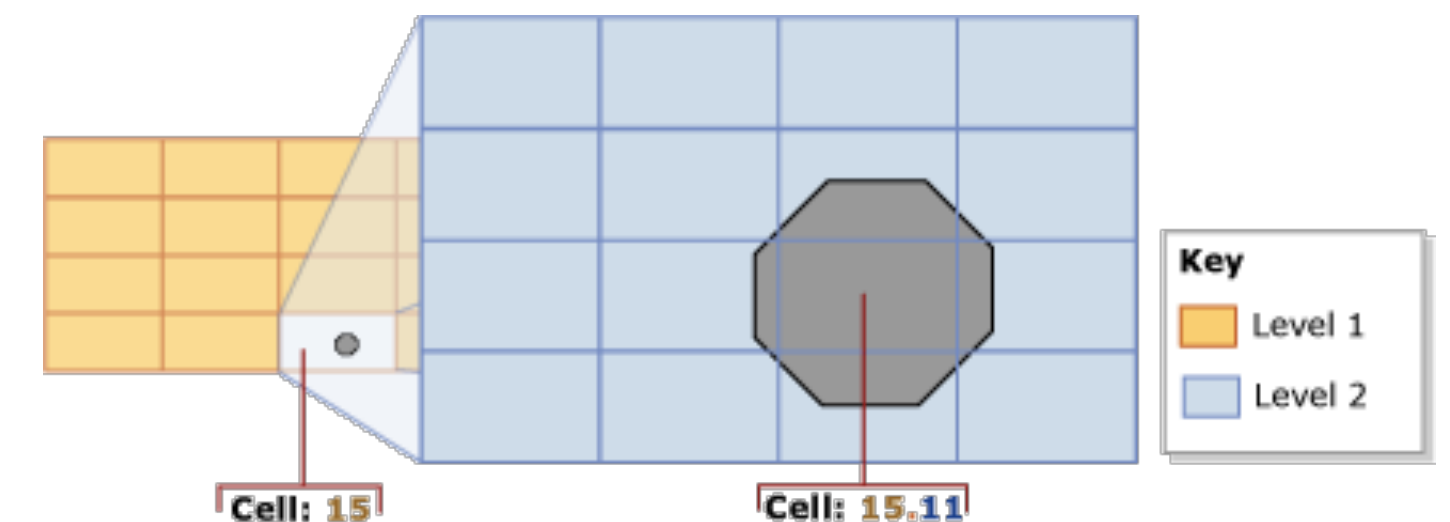
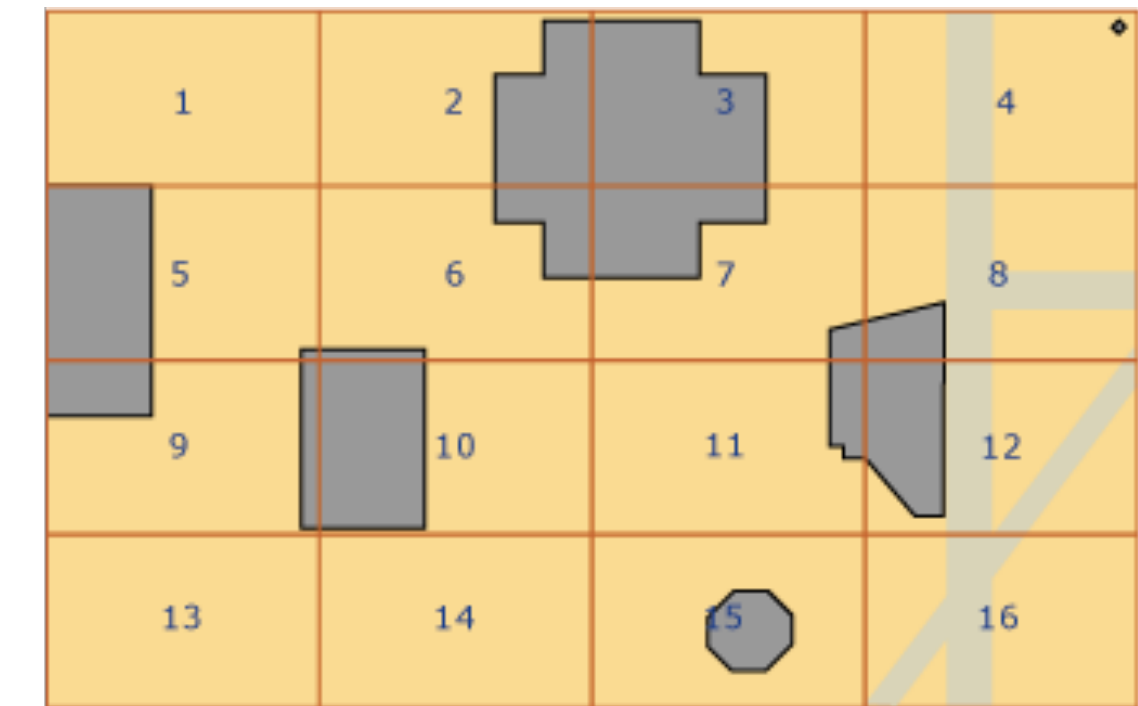
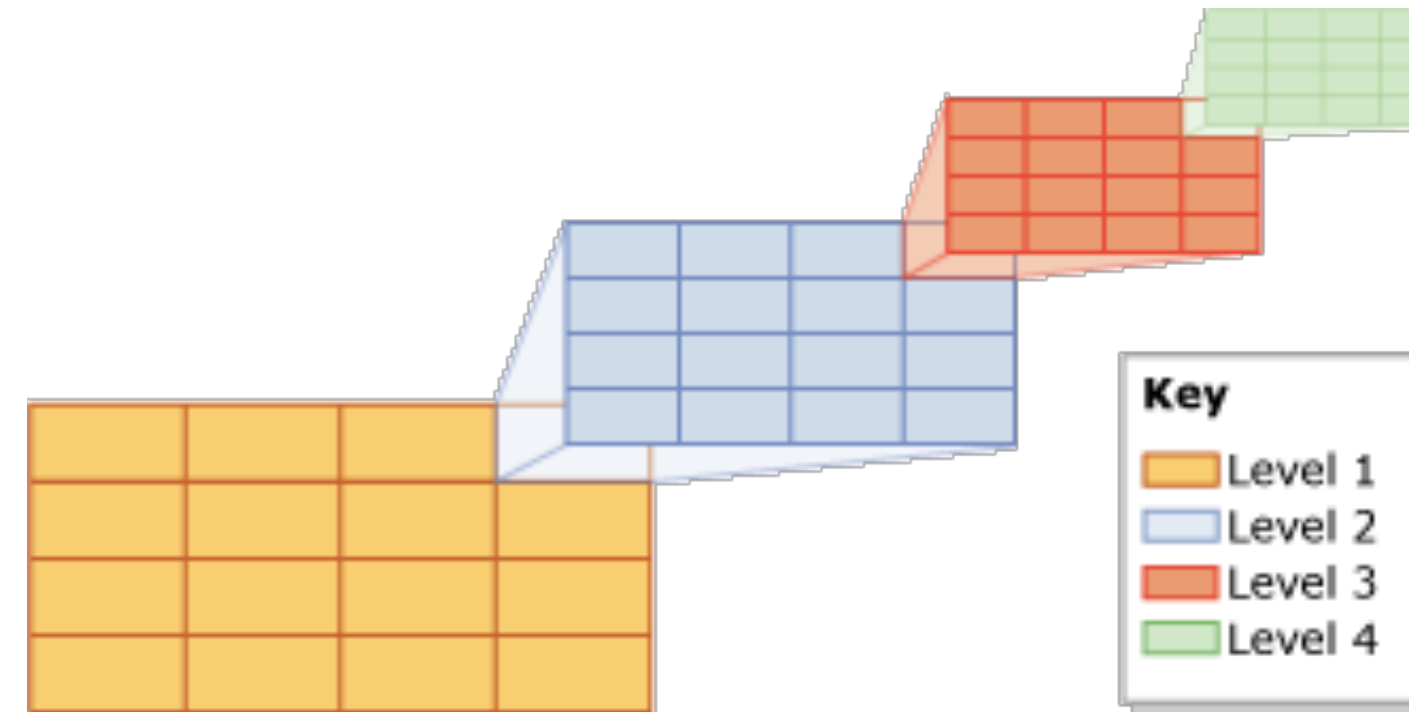
SQL Server 2012 comes with > 390 different SRIDs

- Our default Spatial reference ID: EPSG 4326
  - Coordinate system: geographic ref WGS1984
  - Datum: ellipsoid according to World geodetic system 1984
  - Prime meridian: Greenwich
  - Projection: None
  - Unit of measurement: Degree

# From 0 to 2 dimensions: spatial types overview

## Spatial Indexes

- 4 level grid hierarchy
- Variable grid density per level
- Tessellation rules: covering, cells per object, deepest cell
- Optimized tessellation schemes for geometry/geography
- Support queries that include a spatial operator in the WHERE clause
- Implemented using B-Trees



Overview:

<https://docs.microsoft.com/en-us/sql/relational-databases/spatial/spatial-indexes-overview>

# Getting spatial data into and out of SQL tables

- Input from and output to (choices onboard):

WKT = well known text POINT(30 10)

WKB = well known binary 0x01010000000000000000000003E400000...

GML = geometry markup language (yet another XML dialect)

graphical output also to the SSMS spatial results tab

- Tools:

free Windows app: Shape2SQL (2008)

free command line tool: ogr2ogr

commercial packages: Safe FME, ArcGIS, QGIS...

or: write your own app ;-)



# Getting spatial data into and out of SQL tables

- Spatial data input from WKT = well known text

generic functions:

STGeomFromText(WKT, SRID) and Parse(WKT) for SRID = 0

specific functions, include type check:

STxxxFromText, xxx ∈ Point, Line, Poly, MPoint, MLine, MPoly, GeomColl

examples:

geometry::STPointFromText('POINT (30 10)', 0)

geometry::STPolyFromText('POLYGON ((30 10, 40 40, 20 40, 10 20, 30 10), 0)

- Spatial data output to WKT

SELECT geom.STAsText() results in POINT (30 10)

SELECT geom.AsTextZM() and geom.ToString() include any Z (elevation) and M (measure) values: POINT (30 10 5 17)

# Getting spatial data into and out of SQL tables

- Spatial data input from WKB = well known binary

generic function STGeomFromWKB(WKB, SRID) and specific functions, including type check:  
STxxxFromWKB, xxx ∈ Point, Line, Poly, MPoint, MLine, MPoly, GeomColl

- Spatial data output to WKB  
SELECT geom.STAsBinary()

- Spatial data input from GML = geometry markup language

generic function GeomFromGML(GML, SRID)

- Spatial data output to GML  
SELECT geom.AsGML()

# Functions, functions, functions

## Properties of a geometry

- `STDimension()` returns the max number of dimensions  
point = 0, line string = 1, polygon = 2, empty = -1
- `STGeometryType()` returns a text description of the type of the geom,  
i.e. Point, LineString, MultiPolygon ...
- `InstanceOf(geom_type)` tests if a geom is of a specified type,  
e.g. `InstanceOf('CircularString')`, returns boolean 0 or 1
- `STIsSimple()` is true if the geom does not intersect itself
- `STIsClosed()` is true if the start and end point are the same
- `STIsRing()` is equal to the geom being simple and closed



# Functions, functions, functions

## Properties of a geometry

- STNumPoints() returns the number of points in the geometry
- STIsEmpty() is geom an empty geometry (= 0 points)?
- STStartPoint(), STPointN(n), STEndPoint()  
return the start point, nth point, end point of the geometry
- STNumGeometries() returns the number of geometries
- STGeometryN(n) returns the nth geometry in a collection
- STPointOnSurface() returns an arbitrary point within the geom
- STX, STY, Long, Lat, Z, M, HasZ, HasM  
return the respective coordinates (or their existence)

# Functions, functions, functions

## Properties of a geometry

- STCentroid() / EnvelopeCenter() for geography  
return a point defining the centroid („center of gravity“)
- STBoundary() returns the boundaries of the geometry
- STEnvelope() / STEnvelopeAngle() returns the geom's bounding box
- STConvexHull() returns the convex hull for the geometry
- STBuffer(dist) returns a buffer zone with radius dist around the geom  
see also BufferWithTolerance(...), BufferWithCurves(...)
- STLength(), STArea() return the length and area of a geometry
- STSrid returns or sets the Spatial Reference ID of the geom

# Practical applications

- `GeomA.STUnion(GeomB)` creates a union of two spatial items
- `GeomA.STDifference(GeomB)` forms a geometry from all the points in `GeomA` that are not also in `GeomB` - this is not symmetric, while `A.STSymDifference(B)` is symmetric: points in either A or B, not both
- Aggregate functions on single geo columns: `Union~`, `Envelope~` `ConvexHull~` and `CollectionAggregate(geocolumn)`



# Practical applications

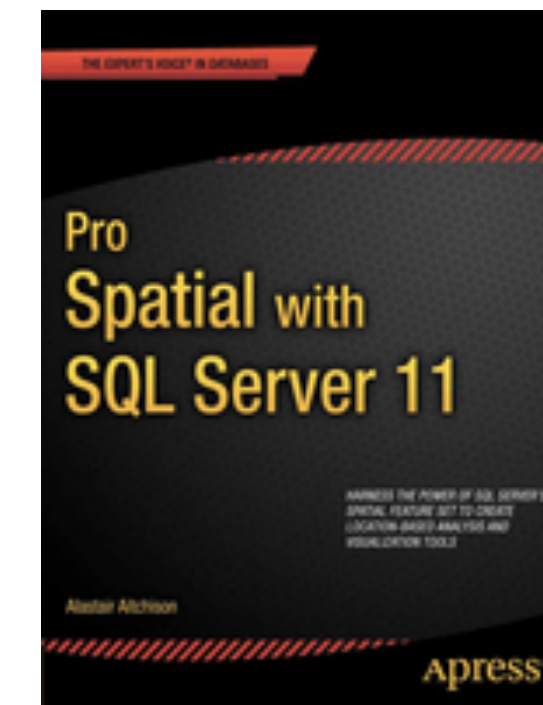
- `GeomA.STDistance(GeomB)` calculates the shortest distance
- `GeomA.ShortestLineTo(GeomB)` forms a geometry representing the shortest line connecting two geometries
- `GeomA.STIntersects(GeomB)` if `GeomA` intersects with `GeomB`, with complementary function `STDisjoint()`, special cases, for geometry objects only: `STCrosses()`, `STTouches()`, `STOverlaps()`, `STContains()`
- `GeomA.STIntersection(GeomB)` returns that part of `GeomA` which intersects with `GeomB`

# Round-up

- Geospatial data type in SQL Server since 2008, added features 2012
- Geography for spheric data, geometry for planar data
- Data types for all kind of geo objects, calculations only up to 2D
- Can be constructed via text, binary or GML
- Dozens of built-in functions to query, compare, analyze geom objects
- Write spatial queries to answer practical business questions
- Foundation to build up on

# Resources on- and offline, credits

- Microsoft docs: (© MS for most illustrations used here)  
<https://docs.microsoft.com/en-us/sql/relational-databases/spatial/spatial-data-sql-server>
- WGS84: [https://en.wikipedia.org/wiki/World\\_Geodetic\\_System#WGS84](https://en.wikipedia.org/wiki/World_Geodetic_System#WGS84)
- Well-known text / binary: [https://en.wikipedia.org/wiki/Well-known\\_text](https://en.wikipedia.org/wiki/Well-known_text)
- Open geospatial consortium: <http://www.ogc.org/>
- GML Standard at OGC: <http://www.ogc.org/standards/gml>
- EPSG Geodetic Parameter Dataset: <https://epsg.org/home.html>
- Pro Spatial with SQL Server 2012, Alastair Aitchison,  
Apress, ISBN 978-1430234913





# Resources on- and offline, credits

- <https://gdz.bkg.bund.de/> Shapefiles for administrative areas of Germany (© GeoBasis-DE / BKG 2018)
- <https://gadm.org/data.html> Shapefiles by country
- ([www.mygeoposition.com](http://www.mygeoposition.com) Geocoding) **currently out of service**  
<http://www.gpsvisualizer.com/geocoding.html> (bring your own Google/Bing API key)
- SQL Server 2008 (!) Spatial Tools (Shape2SQL, SQLSpatial Query Tool) :  
<https://www.sharpgis.net/page/SQL-Server-2008-Spatial-Tools>

# A refresher on geospatial data in SQL Server

Puno hvala for your time and interest & keep in touch:

 @DerFredo <https://twitter.com/DerFredo>

 [de.linkedin.com/in/derfredo](https://de.linkedin.com/in/derfredo)

 [www.xing.com/profile/Thomas\\_Huetter](https://www.xing.com/profile/Thomas_Huetter)



This file and all demo scripts can be found at:

<https://bit.ly/DerFredoCroatia21>