THE BEST-RUN BUSINESSES RUN SAP™

HANA Spatial

SELF-GUIDED DEMONSTRATION

Version 1.2

September 7, 2017

Demo purpose

This script walks you through how to demonstrate the Spatial feature in SAP HANA. It is intended to give you a quick overview of how spatial data works and outlines some of the available SQL functions that can be used with this feature.

Using the script

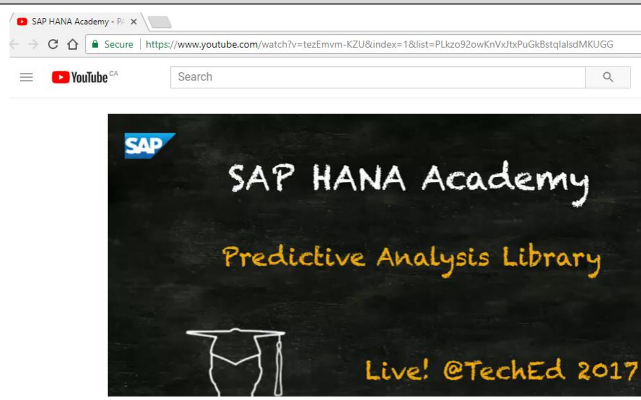
The demo is presented in a four-column format. This format is designed to accommodate your time restrictions and interests. The script is meant to be followed from beginning to end.

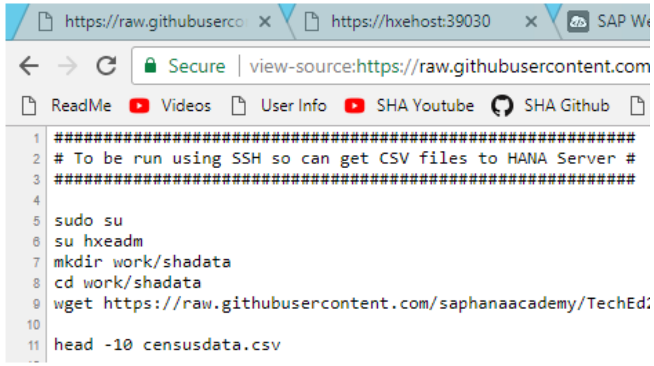
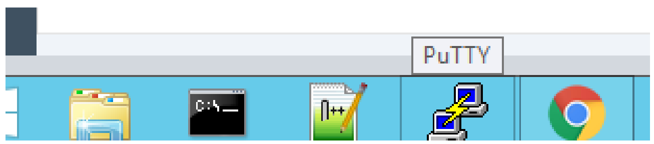
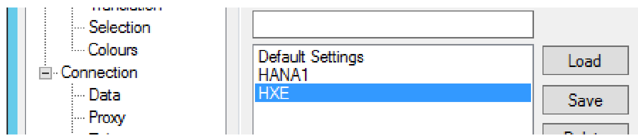
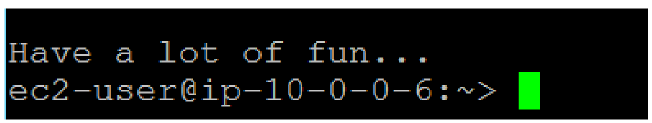
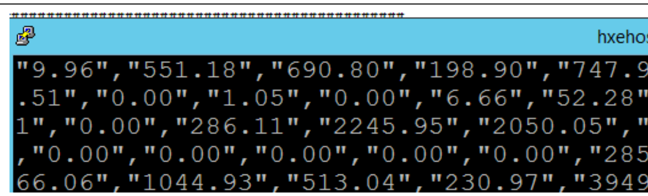
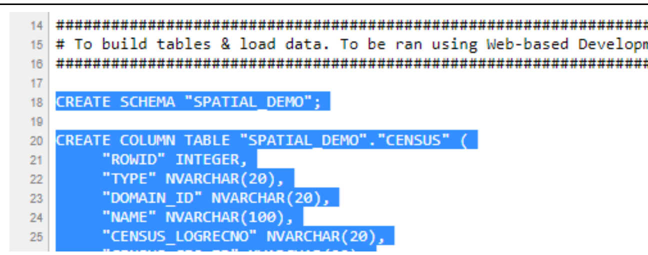
Step	What to do	What you should see	Notes
Step number	Step by step instructions	Screen-shots of what you'll see	Details about what you are doing/seeing

Preparation

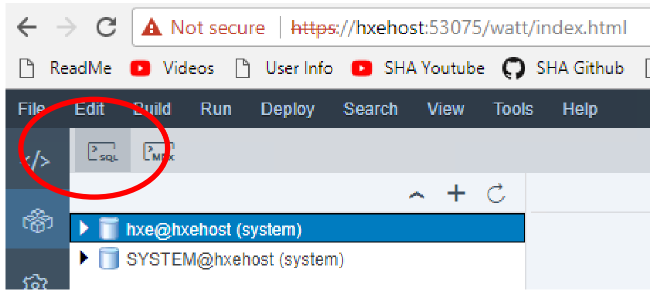
This script was written for use on SAP HANA Express (HXE).

As you will be using HANA Express there is some configuration in Step P1 below that you will need to do in order to set up this system properly. This is very important if you plan to do any development on this system.

Step	What to do	What you should see	Notes
P1	Please open and watch the [video here to get started](). The title says Predictive Analysis Library but the configuration steps for HXE are in this video.		1) Follow the steps from the start to 5:20 in the video. 2) You don't need to do the steps in 5:20 to 7:20 <u>if you don't plan on using predictive analysis on your HXE system</u>. 3) From 7:20 to 8:00 you need to do as you'll want to create a connection to your HXE database as the system user on that tenant. Keep the Chrome tab open. 4) You don't need to watch from 8:00 on <u>if you don't plan on using predictive analysis</u>.

P2	Open the script on importing the spatial demo data [by copying this link]() and pasting into a new tab in Chrome. Copy lines 5 to 11.		We're going to use a script based method to import a .csv file from GitHub. Running import scripts on the HANA system can be more efficient than running imports in HANA Studio (Eclipse with HANA Tools) or via the Web ID or Web Development Workbench.
P3	Open Putty from the Windows Taskbar. Double click on the HXE connection to connect to our HXE's Linux system.		There's already a secure connection created to your HXE system taking into account the edit (mapping) that you would have done to your Hosts file as per the video you watched and followed earlier.
P4	Paste in the syntax that you copied a couple of steps earlier.		The script that you ran has created a work directory on your Linux system, copied a .csv file from github to that directory, and previewed the first 10 lines of that file.
P5	Go back to the Chrome tab where you have the data import script open. Copy lines 18 to 131.		In the next steps we'll use the HXE WebIDE to create our spatial demo schema, a table, and then import our .csv data into that table.

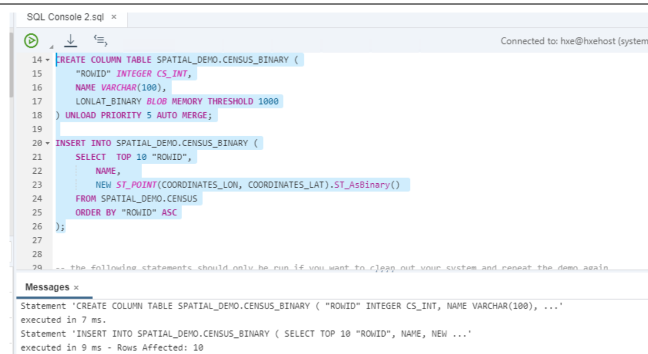
<table>
<tr>
<td>P6</td>
<td>You should have a tab in Chrome open from before where you created a couple of connections to your HANA system.

Click on the hxe@hxehost connection and press the Open SQL Console button.</td>
<td>
</td>
<td>If you closed your WebIDE tab click on the HXE bookmark in Chrome and then click on the WebIDE link in the lower right of your HXE landing page. Then click on the Database Explorer button on the left toolbar of the WebIDE.</td>
</tr>
<tr>
<td>P7</td>
<td>Paste in the code that you copied a couple of steps earlier.

Click on the Run button.</td>
<td>
</td>
<td>You have created a new schema in your HXE db and then a new table and then imported the census data from the .csv file into that table.

You should also see a top 10 record set returned from that new table. Close this SQL Console (but don't close the WebIDE) as the preparation steps are not yet complete.</td>
</tr>
<tr>
<td>P8</td>
<td>Open the data schema prep file here and copy the syntax.

Open another SQL Console and paste in the syntax.

Highlight and then run lines 14 to 26.</td>
<td>
</td>
<td>We're creating another table and adding some coordinate data into that table.

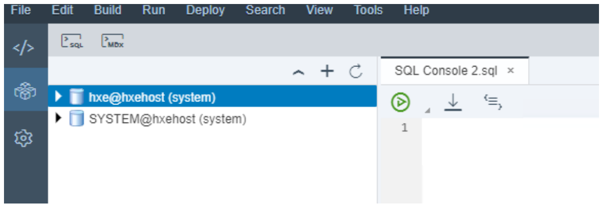
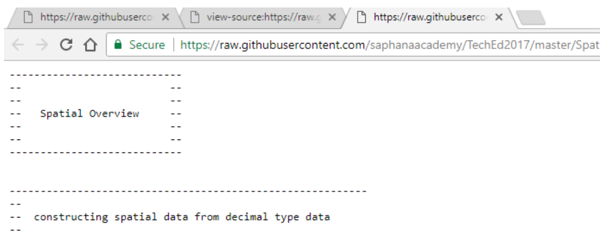
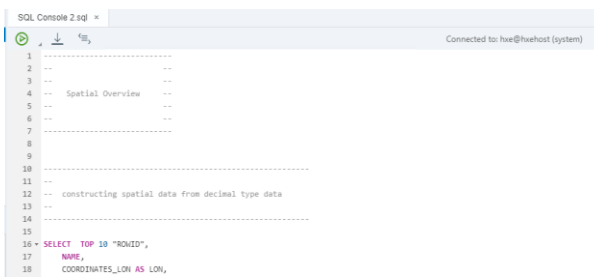
In this case though we're transforming the data into binary data using the spatial function .ST_AsBinary. The reason we're doing this is to mimic a common storage method for coordinate data. Note that this is not the recommended way to store data in HANA if you're using spatial features on that data.

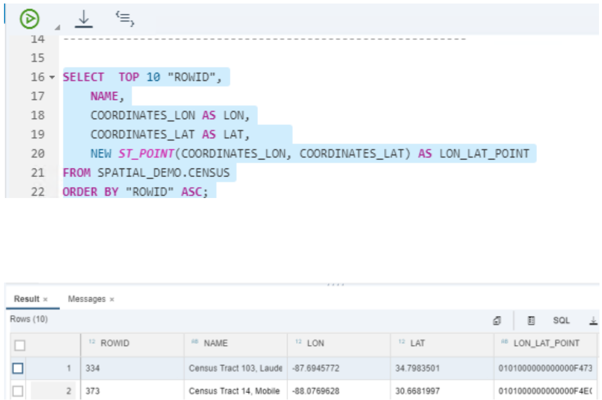
We'll be showing later how to convert binary coordinate data into proper spatial type data.

You are now done the preparation steps.</td>
</tr>
</table>

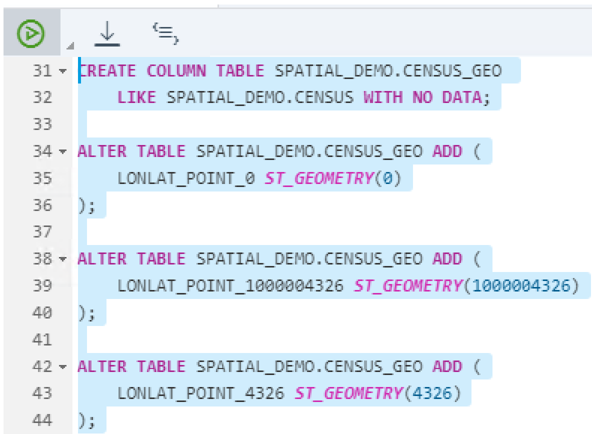
Step-by-step live demo script

It is imperative that you have already completed all of the steps in the previous Preparation section.

	What to do	What you should see	Notes
1	In the WebIDE open up a new SQL Console using the hxe@hxehost connection. Open this link to the spatial demo syntax and copy all. Paste the syntax into the SQL Console you just opened in the WebIDE. Do not run all of this syntax yet as you will be running different parts of the file throughout this demo.	  	What is SAP HANA Spatial? Column-oriented data structures and in-memory computing have developed into powerful components of today's enterprise applications. While the focus of these developments has primarily been on analyzing sales data, the potential for using these technologies to analyze geographic information is significant. Support for the processing of spatial data represents a key evolution in SAP HANA. To deliver vastly improved performance and results in everything from modeling and storage to analysis and presentation of your spatial data, SAP HANA includes a multilayered spatial engine and supports spatial columns, spatial access methods, and spatial reference systems. With these enhanced GIS features, SAP HANA now provides a common database for both your business and spatial data. Spatial data is data that describes the position, shape, and orientation of objects in a defined space. Spatial data is represented as 2D geometries in the form of points, line strings, and polygons. Two common operations performed on spatial data are calculating the distance between geometries, and determining the union or intersection of multiple objects.

<table>
<tr><th></th><th>What to do</th><th>What you should see</th><th>Notes</th></tr>
<tr><td>2</td><td>Select and run lines 16 to 22.</td><td>
</td><td>This syntax shows how to construct a spatial point type (ST_POINT) in SAP HANA SQL.

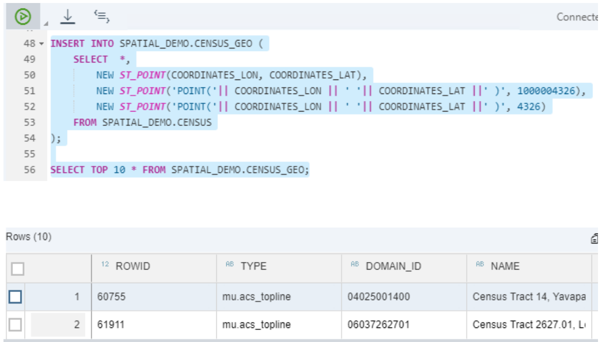
The LON and LAT coordinate columns in this particular case are stored as Decimal types in this CENSUS table. To get started using Spatial in HANA you can simply:
a) Create a table that contains a Spatial Type column (ST_POINT or ST_GEOMETRY)
or
b) You can use extended SQL functions against existing data, like in this example.

The raw output of the point we've constructed (LON_LAT_POINT) is in binary format. Later on we'll look at methods to make the output more readable. Option (a) above is recommended for performance reasons.</td></tr>
<tr><td>3</td><td>Select and run lines 31 to 44.</td><td>
</td><td>We're creating a copy of the CENSUS table, without data, and then adding 3 different Spatial Type (ST_GEOMETRY) columns to the new table.

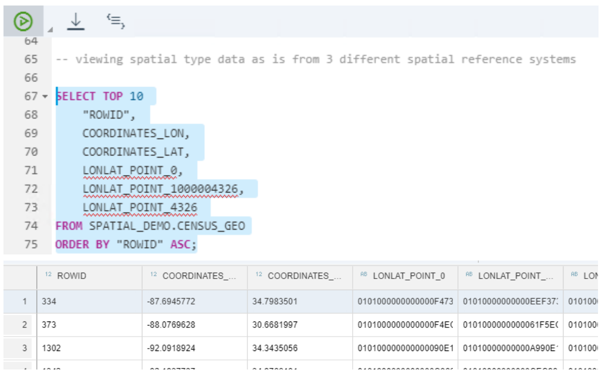
The purpose of this and the next couple of steps is to show how to create a new table or alter an existing table to include a Spatial Type column. While this is not 100% necessary, as you can construct Spatial data on the fly like in Step 2 above, this will make the creation of SQL queries, views, and web services a lot easier and lead to better performance. Having your data stored as spatial type data is the recommended option.

Three main spatial systems for storing data in HANA are:
1) The default which is a Cartesian system (SRID 0),
2) WGS 84 Planar (SRID 1000004326) which is a flat projected system (e.g. a map), and
3) WGS 84 (SRID 4326) which is the spheroidal system used for GPS.

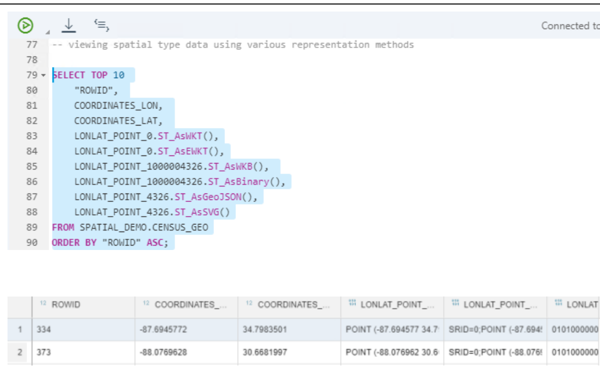
Each system has different advantages (and use cases) which we'll also look at later on. Please see the Spatial Reference Guide for more information on these different systems.</td></tr>
</table>

<table>
<tr><th></th><th>What to do</th><th>What you should see</th><th>Notes</th></tr>
<tr><td>4</td><td>Select and run lines 50 to 58.</td><td>
</td><td>Using existing longitude and latitude columns (stored as Decimal or Float type data) to create a new ST_GEOMETRY or ST_POINT column is relatively easy as it simply requires constructing a NEW ST_Point() like we used earlier in Step 2.

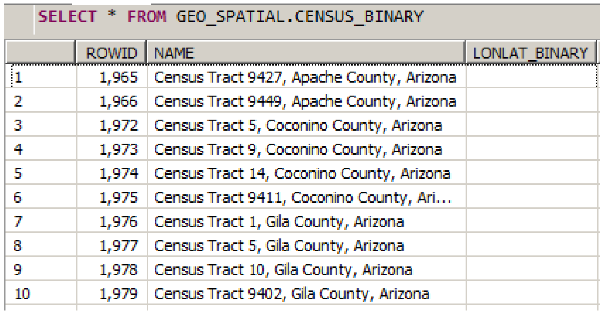
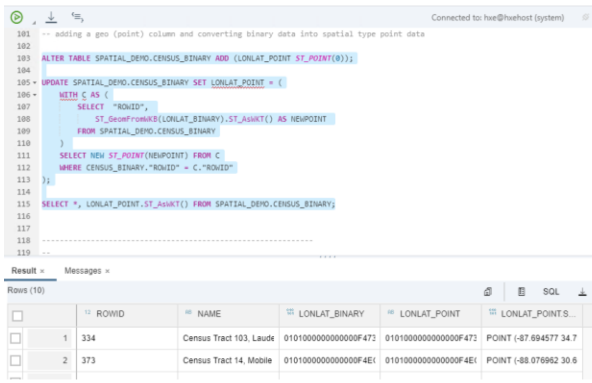
This means that any table brought into HANA that contains longitude and latitude columns can be easily altered to contain a spatial column and existing data can be transformed and inserted into those new spatial columns, like in our example here.

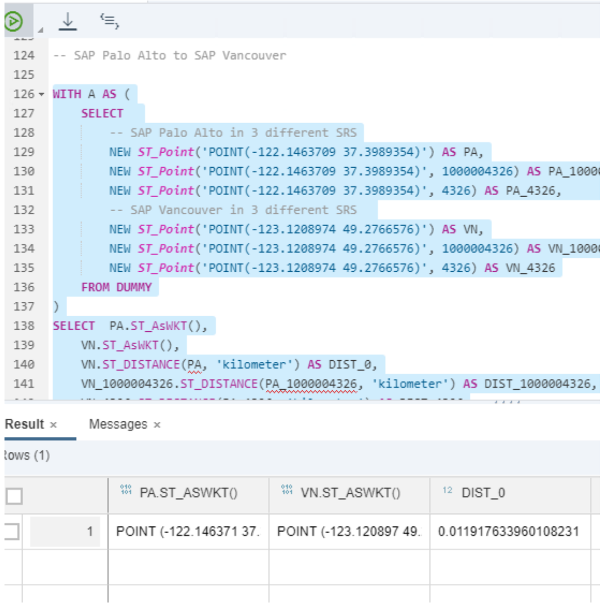
Note that the order of the two dimensions in the constructor is longitude and then latitude which corresponds to an X and then a Y dimension. Some other databases and some mapping tools use the order of latitude and then longitude.</td></tr>
<tr><td>5</td><td>Select and run lines 67 to 75.</td><td>
</td><td>The 3 new ST_Geometry columns in our new table are returned by default in a binary format. The binary output of the 3 columns is identical as, even though these points are stored in different Spatial Reference Systems, the underlying point data remains unchanged.

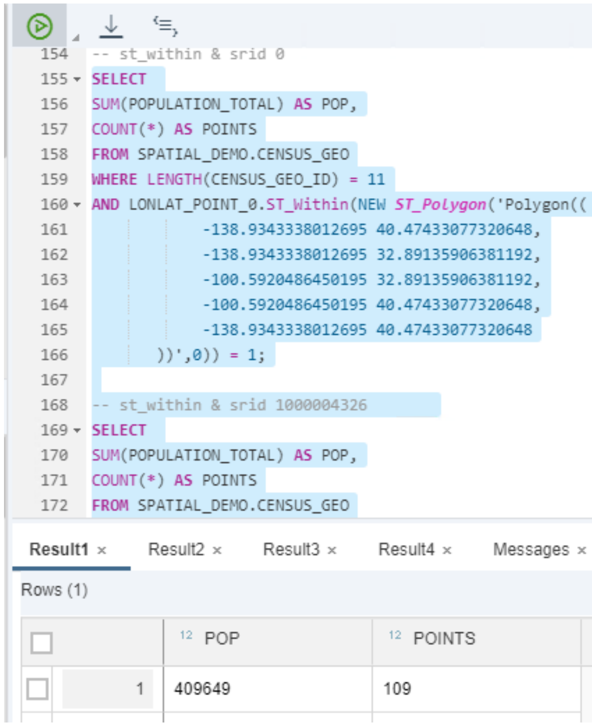
Earlier we created these 3 new columns as ST_Geometry Type columns. We could have also created ST_Point Type columns as we know in this case that these columns will only contain point (longitude and latitude) data.

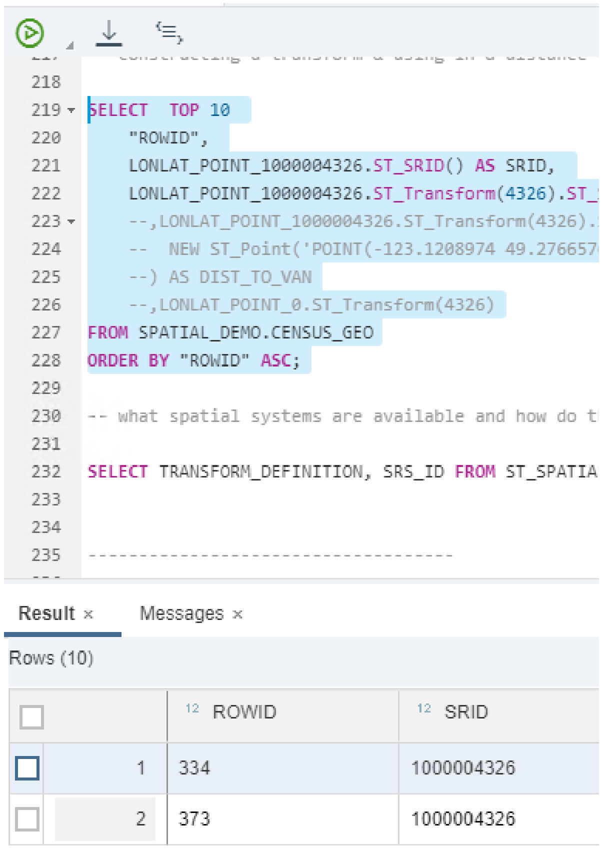
ST_Geometry is the super-type for all geometries and includes points, polygons, line-strings etc. We'll look at these sub-types later on.</td></tr>
<tr><td>6</td><td>Select and run lines 79 to 90.</td><td>
</td><td>This query contains several methods of changing the output of our geometry columns. Two of the more readable methods are AsWKT() or 'As Well Known Text' and AsEWKT() or as 'Extended Well Known Text'.

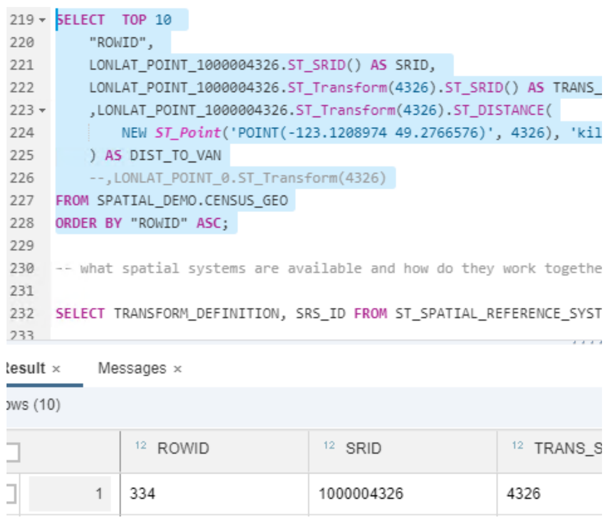
There is also a geo-JSON output that can be consumed in web applications as well as an SVG output.</td></tr>
</table>

	What to do	What you should see	Notes
7	Select and run line 99.		Spatial data may exist as point data stored in binary format in a table. Earlier in the preparation steps we created a table that data stored this way. You will want to convert these types of columns into a Spatial Type column so that it is easier to use the converted data in SQL queries etc. This is also recommended for performance reasons. We'll alter this table to have new HANA Spatial data type columns in the next step.
8	Select and run lines 103 to 115.		In this step we've altered the table with a binary to include a true ST_Point type column. Note that spatial type columns can be either ST_Geometry or ST_Point. We then converted the existing binary column using this syntax: ST_GeomFromWKB(LONLAT_BINARY) This conversion method output is then inserted into the new ST_Point column. We've now converted a table that had a binary column to a table with a HANA Spatial column.

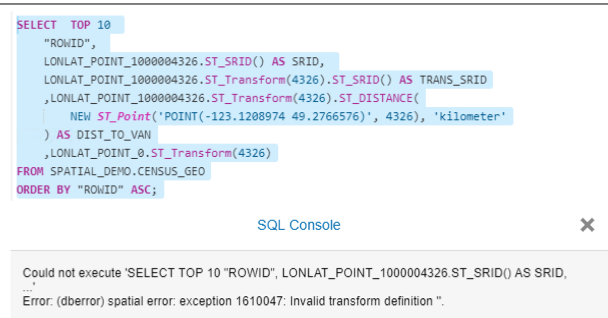
	What to do	What you should see	Notes
9	Select and run lines 126 to 143.		In the top SELECT statement we're deriving some records for the locations of the SAP Vancouver and Palo Alto offices. There is one point for each of the 3 systems: Cartesian, WGS 84 Planar, and WGS 84 Spheroid. The bottom SELECT statement performs 3 distance calculations, using the 3 different spatial systems, between the two offices. Which of these is the most accurate given that the output could be in kilometers? The real distance is about 1321+ kilometers so the most accurate is SRS 4326...remember that this is the system used for GPS. WGS Planar 1000004326 is close in this case but distances are distorted in this system due to the projection of a spherical system onto a flat system. These calculations are more distorted as one uses points that are close to either of Earth's two poles. The Cartesian calculation is far off as it is using a measurement derived from the Pythagoras Theorem. It is not therefore using a Spatial calculation but instead is using a spatial distance calculation. The conclusion to this particular exercise is to utilize SRS 4326 (WGS 84 Spheroid) when accurate distance calculations are required. **NOTE: The differences in the calculated distances are not unique to SAP HANA as they are typical of the spatial systems used.** It is therefore very important that decide which spatial system(s) you will use.

	What to do	What you should see	Notes
10	Select and run lines 155 to 208.		In this set of queries we'll look at the performance of each of the 3 systems we've been using by calculating the number of geographical points within a pre-determined polygon. We're using 3 different methods for the calculation: ST_Within, ST_Contains, & ST_CoveredBy. ST_Within is equivalent to ST_Contains in that all geometries must be completely within the interior & not intersect the boundary. Neither of these two can be used in round earth (SRID 4326) calculations. There are other methods that cannot be used with the 4326 system. Please see the Spatial Reference Guide for more information on the individual methods. Syntax overview... - first two queries use ST_Within function to return a population within a polygon - first one uses data in a Cartesian system & the ST_Within function - second uses WGS Planar & the ST_Within function - third uses WGS Planar & ST_Contains - fourth uses WGS 84 Spheroid / 4326 & ST_CoveredBy Results… - First 3 provide accurate results - ST_Within provides the best speed - SRS 4326 is by far the slowest and results are not what we expect Given the performance results from this simple test it would seem to make sense to use SRS 1000004326 / WGS 84 Planar or SRS 0 / Cartesian for storing Spatial data in HANA. However, we've seen with SRS 0, for obvious reasons, that the distance calculations are not accurate for LON LAT points. NOTE: The differences and results of the calculations are not unique to SAP HANA as they are typical of the spatial systems used. In the next steps we'll see how to effectively deal with having best performance, the greatest accuracy for distance calculations, as well as the greatest accuracy for other methods such as ST_Within.

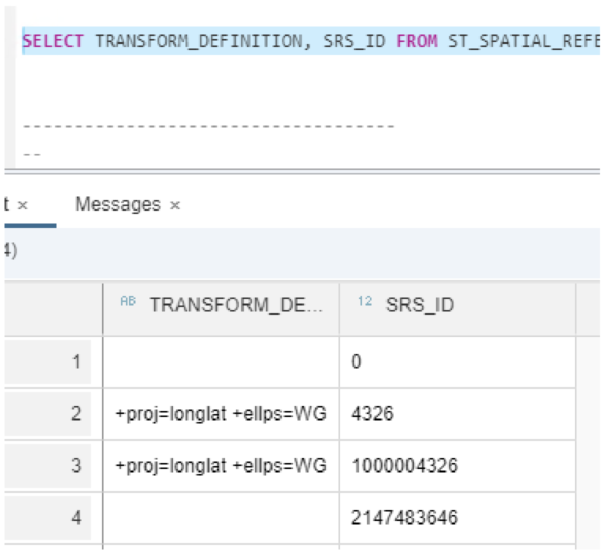
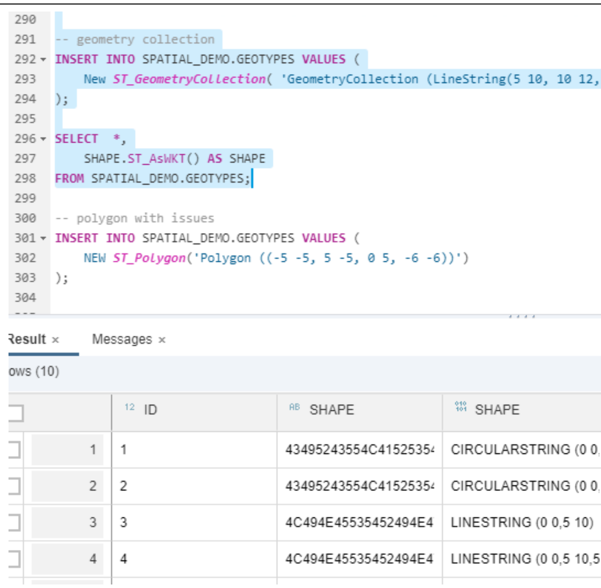
	What to do	What you should see	Notes
11	Select and run lines 219 to 228.		Prior to SPS10 of SAP HANA there was no method to convert data between two different Spatial Reference Systems (SRS). In theory this would mean that data would have to be stored in several different columns, each with a different SRS. The CENSUS_GEO table that we created had such a structure. However, this meant taking up extra database storage for what is essentially the same underlying data. New to SPS10 of HANA is the ST_Transform method that allows data to be converted between different SRS and thus eliminate the need to have multiple columns of the same geometry with different SRS. This is not a recommended complete solution though for performance reasons and should be used sparingly. The recommended solution would be to store the data in the appropriate system type where you can take advantage of HANA Spatial performance. In this syntax we've converted data that is stored in WGS 84 Planar (flat earth) / 1000004326 to WGS 84 Spheroid / 4326. This is done on the fly / in the result set only and does not alter the data stored in HANA.

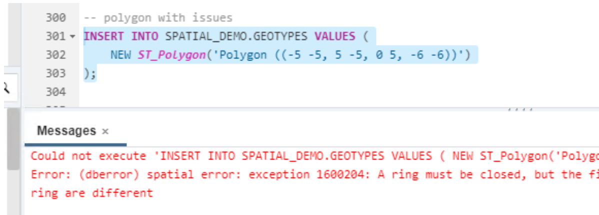
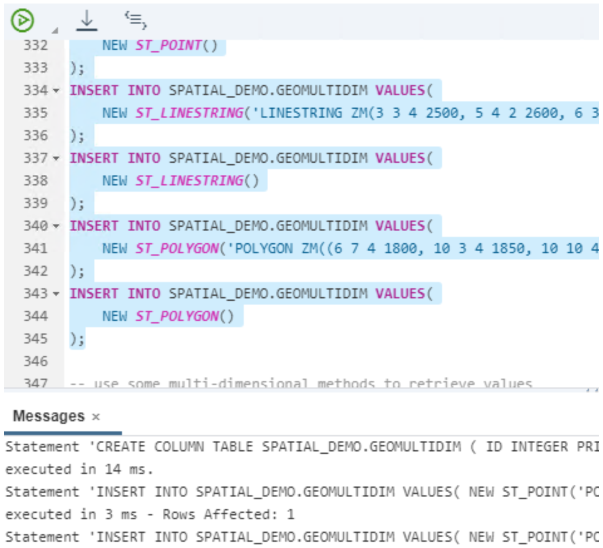
<table>
<tr><th></th><th>What to do</th><th>What you should see</th><th>Notes</th></tr>
<tr><td>12</td><td>Uncomment lines 223 to 225 only. Select and rerun lines 219 to 228.</td><td>
</td><td>In lines 223 to 225 we're transforming data that is stored in WGS 84 Planar so that it is in WGS 84 Spheroid. We then use this transformed data in a distance calculation with the SAP Vancouver office. The Vancouver office data is already constructed as WGS 84 Spheroid / 4326.

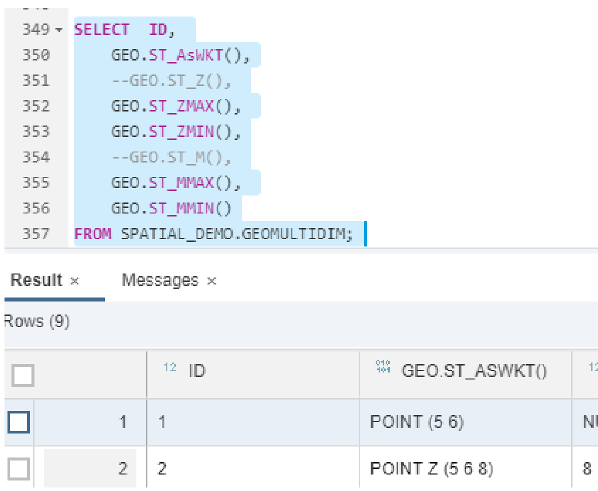
We can see how we can store data as WGS Planar and then use the ST_Transform method so that accurate distances can be calculated when the points are transformed in HANA SQL to WGS Spheroid.

By storing the data in WGS Planar we are then utilizing the SRS that has the best performance as well as having access to the most spatial methods / Spatial SQL functions. As mentioned earlier this should not be a complete solution as using the transform function should be done sparingly.</td></tr>
<tr><td>13</td><td>Uncomment line 226 and rerun lines 219 to 228.</td><td>
</td><td>In line 225 we're trying to transform a geometry stored in SRS 0 / Cartesian to SRS 4326. However an error is returned. In the next step we'll find out why.

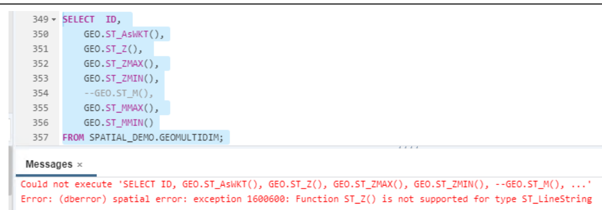
<code>Could not execute 'SELECT TOP 10 "ROWID", LONLAT_POINT_1000004326.ST_SRID() AS SRID, ...' in 143 ms 412 µs .
SAP DBTech JDBC: [669]: spatial error: exception 1610047: Invalid transform definition ''. at function st_transform()</code></td></tr>
</table>

	What to do	What you should see	Notes
14	Select and run line 232.		There are two main rules to keep in mind when using ST_Transform. The first rule is that the SRS must exist in your HANA system. The query that we just ran shows that we have 4 systems including the ones we've been using: 0, 4326, & 1000004326. The second rule is that the original system and the target system must have the same datum. If you look at the TRANSFORM_DEFINITION you'll see that the 4326 and the planar system are both WGS84 for the datum. SRS 0 does not have a TRANSFORM_DEFINITION and therefore one cannot transform between 0 & 4326 or 0 & 1000004326. Given the inability to transform from SRS 0 to other systems and vice versa, storing Spatial data in SRS 1000004326 remains the best option. However, you may encounter existing data that is already stored as SRS 0. In these cases if you require accurate distance calculations then altering that table to include a duplicate of the geometry data but in SRS 1000004326 may be the best solution.
15	Select and run lines 243 to 298.		We've created a new table that has an ST_Geometry type column. This column, named "SHAPE", must the ST_Geometry type as opposed to a point (ST_POINT) column as later we inserted a bunch of records using different geometry types. Once again we're using the ST_AsWKT() method to return a readable output for the SHAPE column. Go back to the SQL in those lines and look at the different constructors for these different geometry types.

	What to do	What you should see	Notes
16	Select and run lines 301 to 303.		We're trying to insert another polygon type but an error is returned. Given the error, what is the issue with this syntax? A polygon (ring) must be closed. To close a polygon the syntax must go back to the starting point of the ring. I.e. in this case the constructor must be as follows where the final point is the same as the starting point of "`-5 -5`". `NEW ST_Polygon('Polygon ((-5 -5, 5 -5, 0 5, -6 -6, -5 -5))')`
17	Select and run lines 314 to 345.		New for SPS10 of HANA 1 was support for multi-dimensional data. Multi-dimensional data allows one to store an additional Z dimension in geometries. A measure, M, can also be stored. The Z dimension does not necessarily have to be a spatial dimension as it can be used for storing other measures such as time. Note that the spatial column is created as ST_Geometry which is a requirement for multi-dimensional data. In the constructors for the new geometries: - lines 319 to 321 is the 2 dimensional method - lines 322 to 324 is a constructor for adding a third dimension - lines 325 to 327 is a constructor for adding a measure - lines 328 to 330 adds both a third dimension and a measure This shows us that multi-dimensional data can be mixed with two-dimensional data. We've looked at spatial types point, line, string, and polygon. Other Supported Spatial Types for Multi-Dimensional Data are: ST_MultiPoint, ST_MultiLineString, ST_MultiPolygon, and ST_GeometryCollection The only geometry type that is not supported (as of SPS10 of HANA 1) is circular string.

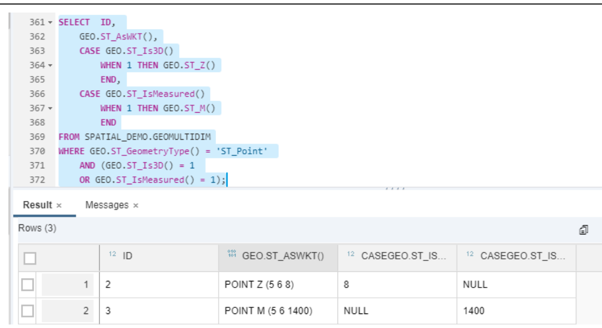
<table>
<tr><th></th><th>What to do</th><th>What you should see</th><th>Notes</th></tr>
<tr><td>18</td><td>Select and run lines 359 to 357.</td><td>


</td><td>There are several other methods that accompany multi-dimensional data. ST_ZMax as an example will return the maximum Z value of a geometry...note that this is not an aggregate function as it's a row-cell level function.

When ZMax is used on a 3-dimensional point it returns the single Z value…the same as ZMin. When these functions are used on 2-dimensional geometries, a NULL / ? is returned.</td></tr>
<tr><td>19</td><td>Uncomment line 351 and run lines 349 to 357 again.</td><td>


</td><td>What does this error mean? The ST_Z method can only be used with geometries that have a Z dimension or value.

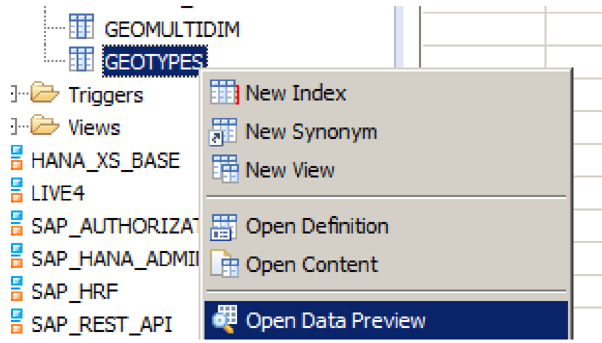
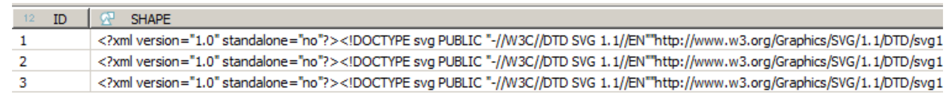
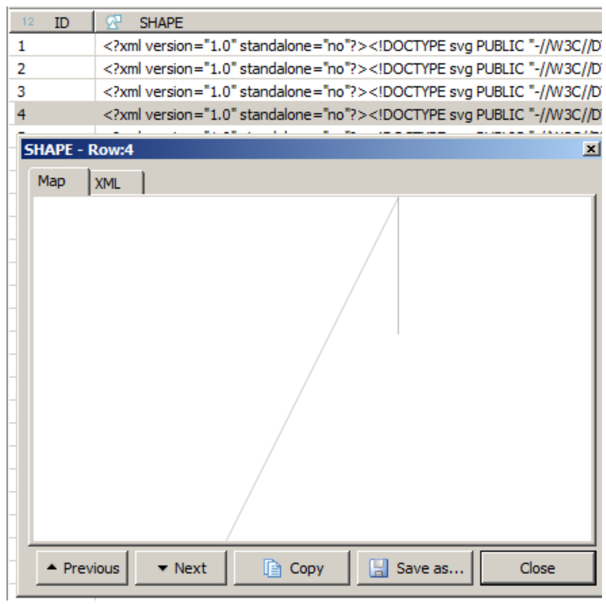
If you were to put the comment back in line 351, uncomment line 354, and rerun the query you would get a similar error. The ST_M method can only be used with geometries that have a measure value.

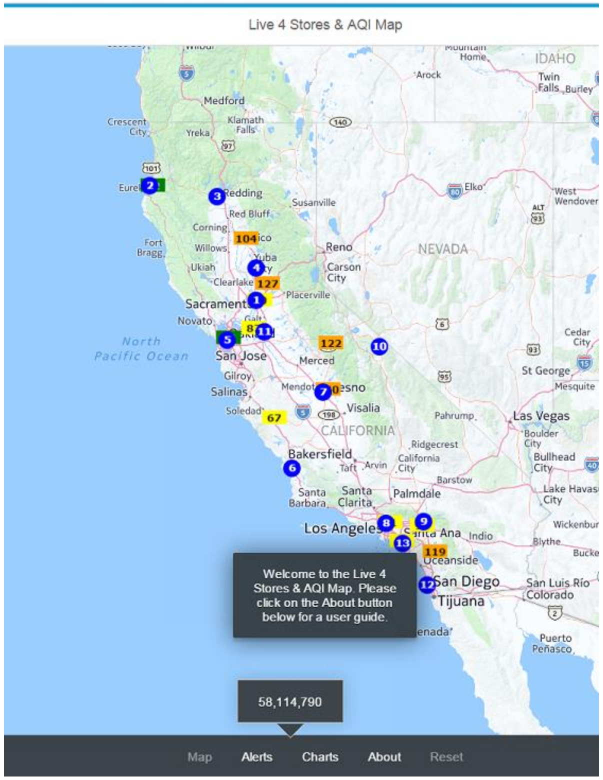
In the next step we will see how to avoid these errors.</td></tr>
<tr><td>20</td><td>Select and run lines 361 to 372.</td><td>


</td><td>Another set of functions for multi-dimensional data are the ST_IsMeasured function and the ST_Is3D function. These are Boolean functions which return a 1 or a 0.

In this query we want to bring back only records which have a measure or are 3d. Here we use case statements that utilize the aforementioned functions to return a measure or 3rd dimension when applicable or a NULL / ? value when those values are not applicable.

Another method to avoid the error in the previous step is to use only the max or min functions (e.g. ST_ZMax) as opposed to using the ST_Z or ST_M functions.</td></tr>
</table>

	What to do	What you should see	Notes
21	In your SPATIAL schema Tables, right click on the GEOTYPES table and choose Open Data Preview.		 New in SPS10 was a spatial preview in HANA Studio. In previous versions when one previewed (using Open Data Preview) a spatial column, the output was in a binary format. Now the output is in XML with an SVG document type.
	Double click on any of the SHAPE column records.		The SHAPE file is now shown as an SVG via the spatial preview in HANA Studio.

	What to do	What you should see	Notes
22	Relax as you're done.		For lots of videos on lots of topics on SAP HANA, please visit the SAP HANA Academy's site here. Be sure to check out the Live 4 / ERP Agility course created by the HANA Academy where you'll use spatial queries as well as utilize the Here Maps API in a SAPUI5 application to create a mapping application. You'll also see how to install some of the additional spatial content, such as zip-code tables, available for licensed HANA users.