IBM® Netezza® Analytics

Release 3.3.x.0

Spatial ESRI Package Developer's Guide

Revised: Oct. 05, 2017

Part Number 00J2396-04 Rev. 8

Note: Before using this information and the product that it supports, read the information in Notices and Trademarks on page 25.

Contents

List of Tables

List of Figures

Preface

Audience for This Guide

The *Netezza Spatial ESRI Package User's Guide* is for users who wish to use the Spatial ESRI features of the IBM Netezza Analytics Package on their IBM Netezza systems. You should be very familiar with spatial analysis and the OpenGIS standards, as well as the basic operation and concepts of the Netezza system.

Purpose of This Guide

The Netezza Spatial ESRI Package provides spatial analysis functions that can be used in queries that run on the Netezza appliance. This guide provides a reference to the spatial ESRI functions and the aggregate (ST_GrandMBR) included with the Netezza Spatial ESRI Package. While the guide describes the Netezza-specific aspects of the Spatial ESRI Package, it does not provide a broad general discussion of spatial or geospatial analysis and concepts.

Symbols and Conventions

Note on Terminology: The terms *User-Defined Analytic Process (UDAP)* and *Analytic Executable (AE)* are synonymous.

The following conventions apply:

- Italics for emphasis on terms and user-defined values, such as user input.
- Upper case for SQL commands, for example, INSERT or DELETE.
- Bold for command line input, for example, **`nzsystem stop`**.
- Bold to denote parameter names, argument names, or other named references.
- Angle brackets (< >) to indicate a placeholder (variable) that should be replaced with actual text, for example, `inza-<release_number>.zip`.
- A single backslash ("\") at the end of a line of code to denote a line continuation. Omit the backslash when using the code at the command line, in a SQL command, or in a file.
- When referencing a sequence of menu and submenu selections, the ">" character denotes the different menu options, for example, **Menu Name > Submenu Name > Selection**.

If You Need Help

If you are having trouble using the IBM Netezza appliance, IBM Netezza Analytics or any of its components:

1. Retry the action, carefully following the instructions in the documentation.
2. Go to the IBM Support Portal at: http://www.ibm.com/support. Log in using your IBM ID and password. You can search the Support Portal for solutions. To submit a support request, click the '**Service Requests & PMRs**' tab.
3. If you have an active service contract maintenance agreement with IBM, you may contact customer support teams via telephone. For individual countries, please visit the Technical Support section of the IBM Directory of worldwide contacts (http://www14.software.ibm.com/webapp/set2/sas/f/handbook/contacts.html#phone)

Comments on the Documentation

We welcome any questions, comments, or suggestions that you have for the IBM Netezza documentation. Please send us an e-mail message at netezza-doc@wwpdl.vnet.ibm.com and include the following information:

- The name and version of the manual that you are using
- Any comments that you have about the manual
- Your name, address, and phone number

We appreciate your comments.

CHAPTER 1

Netezza Spatial ESRI Package Overview

About the Netezza Spatial ESRI Package

The Netezza Spatial ESRI Package package provides a set of spatial and geospatial analytic functions for Netezza data warehouse appliances. These functions provide the ability to analyze distance, space, shape, and intersection questions for the data on the Netezza system.

The Netezza Spatial ESRI Package functions follow the Open Geospatial Consortium, Inc. (OGC) OpenGIS standards as documented in the *OpenGIS Implementation Specification for Geographic Information - Simple Feature Access - Part 1: Common Architecture 1.2.0* and the ISO19107 & ISO13249-3:2006 – SQL Multimedia and Application Packages – Part 3 – Spatial 3rd Ed.

While the OpenGIS framework is geared toward user-defined methods on custom datatypes, Netezza uses a more standard ANSI SQL function style of syntax, where user-defined functions (UDFs) are used instead with the first parameter representing the object for which the method is to be applied.

For example, to find the area of an ST_Polygon object in column A of table **counties** under OpenGIS, a SQL query such as the following may be used:

```
SELECT A.ST_Area() FROM counties;
```

Using the Netezza Spatial ESRI Package, the equivalent SQL is written as:

```
SELECT ST_Area(A) FROM counties;
```

Installation and Administration

For complete information on installation and administration of the Netezza Spatial ESRI Package, refer to the *IBM Netezza Analytics Installation and Administration Guide*.

Important: Depending on the selections that you make during the installation, several new databases are created, one of which is the INZA database. You must not use this INZA database for user data.

Changing from nzspatial to nzspatial_esri

When upgrading from nzspatial to nzspatial_esri, the **convert.sh** script is available to help migrate data to the ESRI format. Because nzspatial supports two non-standard SRIDs–1111 (sphere) and 1234 (Cartesian)– the script will first normalize the SRIDs (if your existing data is not in WGS 84 format) and then convert the data and create a new table. The script makes use of the ST_CONVERT function, so it is not necessary to call that function separately. The script is located in /nz/export/ae/products/netezza/nzspatial_esri/<version>/scripts/convert.sh.

Note: It is always recommended to make a copy of your database before running this script on the data.

The following describes the script.

The convert.sh Script

The **convert.sh** script converts a table with a VARCHAR/WKB geometry column to a new table. The column names remain the same, but the new table data format is ST_GEOMETRY/ESRI binary geometry data. The script also normalizes the two non-standard SRIDs in the well known binary (WKB) database. SRID 1111 is converted to 4269 (by default) and SRID 1234 to 27700 (by default) but the exact SRID can be specified. The result is a new table, either in the same database or a specified database.

To successfully use the convert.sh script, the following must be true:

- you must execute the script as **user nz**
- nzspatial functions exist
- nzspatial_esri functions exist
- the destination database is INZA-initialized
- the database, table, and geometry column specified must match how it actually appears.

Syntax

```
convert.sh <nzspatial db> <nzspatial table> <geometry column> <nzspatial
function database> <nzspatial_esri function database> <nzspatial_esri db>
<nzspatial_esri table> <new sphere srid> <new cartesian srid>
```

The following table describes each parameter:

Table 1: Parameters of the convert.sh script

Parameter	Description
<nzspatial db>	*Required*. The source database name containing the nzspatial table data.
<nzspatial table>	*Required*. The source table name containing nzspatial data.

Parameter	Description
<geometry column>	*Required*. The geometry column name.
<nzspatial function database>	*Required*. The name of the database containing the nzspatial functions.
<nzspatial_esri function database>	*Required*. The name of the database name containing the nzspatial_esri functions.
<nzspatial_esri db>	*Optional*. The destination database name containing the newly created nzspatial_esri table data. Defaults to “<nzspatial db>”.
<nzspatial_esri table>	*Optional*. The destination table name containing the newly created nzspatial_esri data. Defaults to "<nzspatial_table>_tmp".
<new sphere srid>	*Optional*. The SRID to assign to the newly created nzspatial_esri data that was previously assigned "sphere/1111". Defaults to 4269.
<new cartesian srid>	*Optional*. The SRID to assign to newly created nzspatial_esri data that was previously assigned "cartesian/1234". Defaults to 27700.

CHAPTER 2
Getting Started with Geometric Analysis

Spatial Concepts

The Netezza Spatial ESRI package contains functions and capabilities that allow you to process queries about geometric features or geographical data using the data contained ion the NPS. An example of geographical data could include such items as:

- The location of a store, restaurant, a wireless service tower, national park, or other landmark
- A plot or area of land, such as an office park, a county or precinct, or a wireless coverage zone
- A running feature such as a street, river, railway line, tunnel, or power line

The combination of spatial information with relational database information, allows powerful interpretations as well as images of the data correlations. For example:

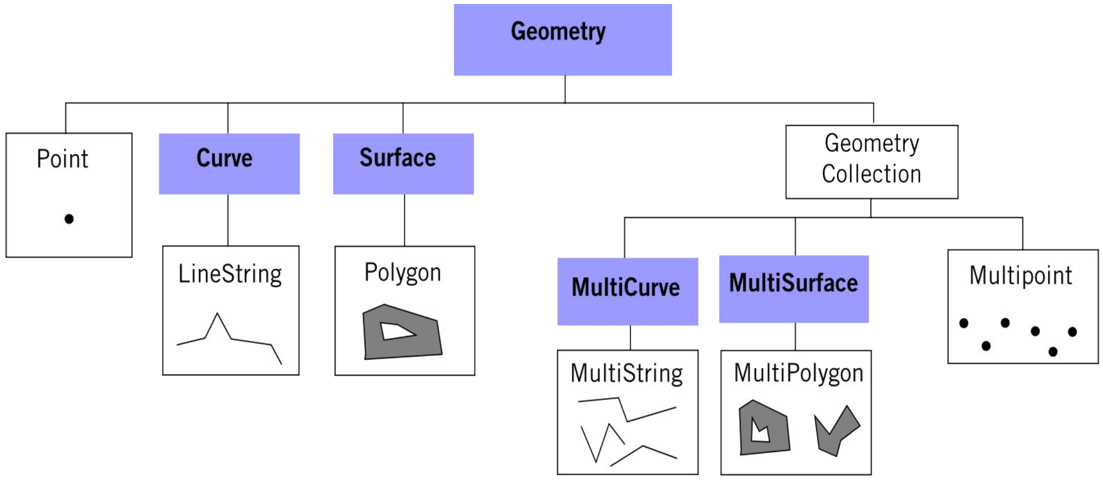
- Identify the number of wireless calls that occur in a particular area to improve the panning process for the addition of new towers for better wireless service
- Map the location of stores and calculate the distance between customer addresses and the store location to plan advertising coverage
- Identify an aquifer area and plan a buffer around it to calculate the impact and cost of a fence or enclosure protecting the water zone from unauthorized access

Spatial data typically originates from three sources: it can be derived from business data, calculated using spatial functions, or imported from external sources or databases.

Geometry Types

There are two main categories of geometry types – instantiated and abstract – categorized based on whether a visual rendering can be created. Instantiated types can be rendered visually while abstract, or non-instantiated, types cannot be rendered visually. See Figure 1: Geometry types and relationships for geometry type examples.

Figure 1: Geometry types and relationships



In the figure above, the boxes that are shaded blue, specifically Geometry, Curve, Surface, MultiCurve, and MultiSurface, are non-instantiable or abstract types that cannot be rendered visually. They define a subtype or grouping of object types. As instantiable types, Point, LineString, Polygon, Geometry Collection, MultiString, MultiPolygon, and MultiPoint can be rendered visually in mapping or image applications. These geometric types are often used to represent various geographic features:

- Points can represent a specific location, such as a city, an intersection of two streets, a radio tower, or a building.
- LineStrings can represent features such as a street, trail, route, river, or power line.
- Polygons represent areas or parcels, such as a university campus, a homeowner's property, a park, a floodplain, a service coverage area, or a floor plan.

For a complete description of the geometry types, refer to the OpenGIS standard specification.

Geometric Properties

Geometric types have coordinate and dimension properties. Coordinates define location as well as shape and size, while dimension specifies whether an object has length, width, and/or height. For example, points have four possible ordinate values:

- **X**—left/right
- **Y**—up/down
- **Z**—altitude/depth
- **M**—a measure associated with the object, such as a distance along a linestring from the start point, a flow rate for a pipe, or an average speed for a particular area of roadway

There are four possible dimensions for each geometry object:

- **-1**—an empty object
- **0**—a point type
- **1**—a line string
- **2**—a polygon that has an area larger than 0

Note: The Netezza Spatial ESRI Package supports vector objects and spatial operators, as defined in the OpenGIS standard. The package does not support 3D geocodings or raster data/functions.

Simple and Non-Simple Geometries

As defined in the OpenGIS standard, a simple geometry is one that does not have any "anomalous" geometric points, such as self intersection or self tangency. Each geometric type defines its simple and non-simple aspects. Some examples of non-simple geometries include: a polygon with vertices inside the area of the polygon itself; a linestring that intersects itself; a multipoint that has two points with equal coordinates; a polygon with an interior ring that touches the polygon's boundary. See Figure 2.

Figure 2: Simple and Non-Simple Geometries

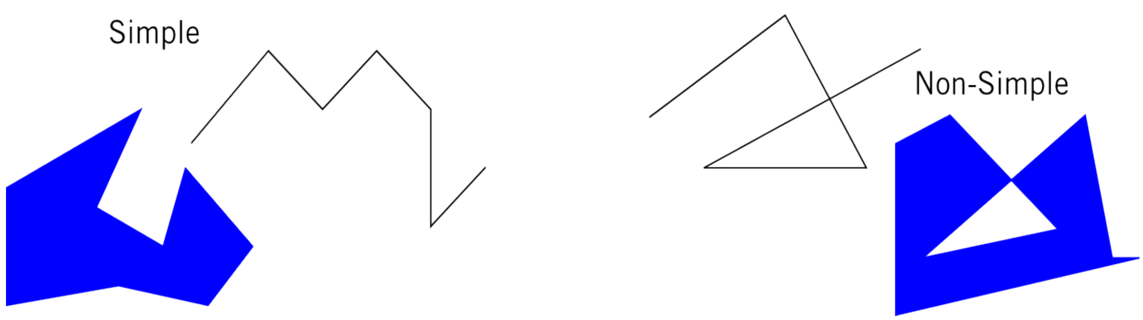


Figure 2 shows some examples of simple and non-simple geometries. The two geometries on the left are simple geometries that do not intersect within themselves. On the right side, both the linestring and the polygon have self-intersecting lines, and thus are non-simple.

The Netezza Spatial ESRI Package supports simple geometries and non-simple geometries. Thus, you can insert or load simple geometric object definitions and non-simple geometric object definitions.

Figure 3: Sample geometry definition for a simple linestring shows a sample SQL query to add the linestring object illustrated in the grid to a table named **geomtable**. The query successfully adds the linestring object.

Figure 3: Sample geometry definition for a simple linestring

```
CREATE TABLE geomtable(geoms ST_GEOMETRY(500));
CREATE TABLE

INSERT INTO geomtable VALUES
(inza..ST_WKTToSQL('LineString(1 1, 2 3, 3 4,
4 2, 5 3)'));
INSERT 0 1
```

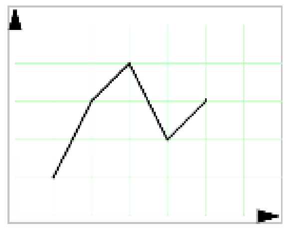
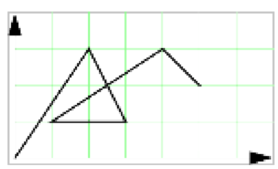


Figure 4: Sample geometry definition for a non-simple linestring shows a sample query that defines the non-simple linestring object illustrated in the grid.

Figure 4: Sample geometry definition for a non-simple linestring

```
INSERT INTO geomtable VALUES
(inza..ST_WKTToSQL('LineString(0 0, 2 3, 1 3,
1 1, 4 3, 2 5)', 4326));
ERROR:  Geometry is not simple

INSERT INTO geomtable VALUES (inza..ST_WKTToSQL('LineString(0 0, 2
3, 1 3,1 1, 4 3, 2 5)', 4326, true));
INSERT 0 1
```



Note that the 1st SQL query returns an error. The 2nd SQL query has the exact same non simple geometry but is successful since the skipSimpleTest is set to "true".

Note that loading of non-simple geometric data can lead to unexpected, incorrect results when analyzing the non-simple geometries for values such as distance, area, contains, or intersections. To load geometric data from other sources to the Netezza database, best practices are available for dealing with potential non-simple geometries in the data. For more information, see the Loading Geometric Data into the Netezza Database section.

Spatial ESRI Package Geometry Types

In the Netezza implementation, geometry types are defined by a new fundamental type called ST_GEOMETRY. An ST_GEOMETRY data field can represent a spatial object such as a point, linestring, or polygon.

The subtypes of ST_Geometry are as follows:

- ST_Point
- ST_Curve (non-instantiable)
- ST_Linestring
- ST_Surface (non-instantiable)
- ST_Polygon
- ST_MultiCurve (non-instantiable)
- ST_Multipoint
- ST_MultiLineString
- ST_MultiSurface (non-instantiable)
- ST_MultiPolygon
- ST_GeomCollection

The Spatial ESRI package does *not* support the following two subtypes of the OpenGIS standard:

- ST_Polyhedral
- ST_Text

In the OpenGIS standard, each geometry type has a defined integer value. Table 2 shows the integer code values and the number of coordinates for each type. These codes are used in some of the spatial functions described in the *Netezza Spatial ESRI Reference Guide*.

Table 2: OpenGIS standard geometry type values

Code	Geometry Types	Coordinates
0	GEOMETRY	X Y
1	POINT	X Y
2	LINESTRING	X Y
3	POLYGON	X Y
4	MULTIPOINT	X Y
5	MULTILINESTRING	X Y
6	MULTIPOLYGON	X Y
7	GEOMCOLLECTION	X Y

Code	Geometry Types	Coordinates
13	CURVE	X Y
14	SURFACE	X Y
15	POLYHEDRALSURFACE	X Y
1000	GEOMETRYZ	X Y Z
1001	POINTZ	X Y Z
1002	LINESTRINGZ	X Y Z
1003	POLYGONZ	X Y Z
1004	MULTIPOINTZ	X Y Z
1005	MULTILINESTRINGZ	X Y Z
1006	MULTIPOLYGONZ	X Y Z
1007	GEOMCOLLECTIONZ	X Y Z
1013	CURVEZ	X Y Z
1014	SURFACEZ	X Y Z
1015	POLYHEDRALSURFACEZ	X Y Z
2000	GEOMETRYM	X Y M
2001	POINTM	X Y M
2002	LINESTRINGM	X Y M
2003	POLYGONM	X Y M
2004	MULTIPOINTM	X Y M
2005	MULTILINESTRINGM	X Y M
2005	MULTILINESTRINGM	X Y M
2006	MULTIPOLYGONM	X Y M
2007	GEOMCOLLECTIONM	X Y M

Code	Geometry Types	Coordinates
2013	CURVEM	X Y M
2014	SURFACEM	X Y M
2015	POLYHEDRALSURFACEM	X Y M
3000	GEOMETRYZM	X Y Z M
3001	POINTZM	X Y Z M
3002	LINESTRINGZM	X Y Z M
3003	POLYGONZM	X Y Z M
3004	MULTIPOINTZM	X Y Z M
3005	MULTILINESTRINGZM	X Y Z M
3006	MULTIPOLYGONZM	X Y Z M
3007	GEOMCOLLECTIONZM	X Y Z M
3013	CURVEZM	X Y Z M
3014	SURFACEZM	X Y Z M
3015	POLYHEDRALSURFACEZM	X Y Z M

Netezza Spatial ESRI Package Functions

This section provides an overview of the Spatial ESRI package functions. The spatial functions typically perform operations that fall into these categories:

- Geometric information functions, which return information about a geometric object
- Conversion (or constructor) functions, which convert an object into another representation
- Comparison functions, which evaluate whether two or more objects touch, overlap, or otherwise intersect or connect
- Geometric object manipulation functions, which can set coordinate values or derive new objects such as centroids, buffers, bounding regions, and so on
- Distance and area functions, which evaluate objects for measurements such as area, distance, and length

Table 3 categorizes the spatial functions by the type of operations they perform. The functions and their arguments are described in the *Netezza Spatial ESRI Reference Guide*.

Table 3: Spatial functions and operation types

Operation Type	Functions			
Geometric Information	ST_CoordDim	ST_Is3D	ST_MaxX	ST_NumInteriorRing
	ST_Dimension	ST_IsClosed	ST_MaxY	ST_NumPoints
	ST_EndPoint	ST_IsEmpty	ST_MaxZ	ST_SRID
	ST_GeomFromText	ST_IsMeasured	ST_MinM	ST_StartPoint
	ST_GeomFromWKB	ST_IsRing	ST_MinX	ST_X
	ST_GeometryN	ST_IsSimple	ST_MinY	ST_Y
	ST_GeometryType	ST_M	ST_MinZ	ST_Z
	ST_GeometryTypeId	ST_MaxM	ST_NumGeometries	
Conversion Functions	ST_AsBinary	ST_AsKML[1]	ST_Tranform	ST_WKTToSQL
	ST_AsText	ST_Convert	ST_WKBToSQL	
Comparison Functions	ST_Contains	ST_DWithin	ST_Intersects	ST_Touches
	ST_Crosses	ST_MBRIntersects	ST_Overlaps	ST_Within
	ST_Disjoint	ST_Equals	ST_Relate	
Object Manipulation Functions	ST_Buffer	ST_Envelope	ST_MBR	ST_X
	ST_Boundary	ST_ExteriorRing	ST_Point	ST_Y
	ST_Centroid	ST_InteriorRingN	ST_PointN	ST_Z
	ST_ConvexHull	ST_Intersection	ST_SRID	
	ST_Difference	ST_M	ST_SymDifference	
Measurement and Distance Functions	ST_Area	ST_Distance	ST_Length	ST_Perimeter

Table 4 lists the functions and aggregates that can be used on each geometry type.

1 For more information on the Keyhole Markup Language (KML) used in the ST_AsKML function, see *http://code.google.com/apis/kml/documentation/kmlreference.html*.

Table 4: Spatial Functions and Operation Types

Geometry Type	Functions and Aggregates Supported
ST_Geometry	ST_AsBinary, ST_AsText, ST_Boundary, ST_Buffer, ST_Contains, ST_ConvexHull, ST_CoordDim, ST_Crosses, ST_Difference, ST_Dimension, ST_Disjoint, ST_Distance, ST_DWithin, ST_Envelope, ST_Equals, ST_GeometryType, ST_GeometryTypeId, ST_Intersection, ST_Intersects, ST_Is3D, ST_IsEmpty, ST_IsMeasured, ST_IsSimple, ST_Length, ST_MBR, ST_MaxM, ST_MaxX, ST_MaxY, ST_MaxZ, ST_MinM, ST_MinX, ST_MinY, ST_MinZ, ST_Overlaps, ST_Perimeter, ST_Relate, ST_SRID, ST_SymDifference, ST_Touches, ST_Union, ST_Within, ST_WKBToSQL, ST_WKTToSQL
ST_Point	ST_M ST_Point ST_X ST_Y ST_Z
ST_LineString	ST_EndPoint ST_IsClosed ST_IsRing ST_NumPoints ST_PointN ST_StartPoint
ST_Surface	ST_Area ST_Centroid
ST_Polygon	ST_ExteriorRing ST_InteriorRingN ST_NumInteriorRing
ST_GeomCollection	ST_GeometryN ST_NumGeometries
ST_MultiPoint	All the functions supported by ST_GeomCollection and ST_Geometry
ST_MultiLineString	ST_IsClosed ST_IsRing
ST_MultiPolygon	ST_Area ST_Centroid

Getting Started with Geometric Analysis

To get started with the Netezza Spatial ESRI Package, spatial data is first loaded into a Netezza database. A table is created with a geometry column as a ST_GEOMETRY column.

```
CREATE TABLE PointData (PointID integer, the_geom ST_GEOMETRY(200));
```

This command creates a table suitable for loading point data. To insert a point to the table, commands similar to the following examples can be used:

```
INSERT INTO PointData VALUES (1, inza..ST_Point(3423, 4356));
INSERT INTO PointData VALUES (1, inza..ST_WKTToSQL('Point (3423 4356)'));
```

Note that only one of these INSERT commands is needed to insert the point into the PointData table. Executing both INSERT commands results in two points with the same PointID. The column the_geom can be used anywhere that the datatype ST_Geometry (or the subclass ST_Point) is shown in the Spatial ESRI API functions, documented the *Netezza Spatial ESRI Reference Guide*.

As an example, the following command displays the geometry type of the data in the the_geom column:

```
SELECT inza..ST_GeometryType(the_geom) FROM PointData;
ST_GEOMETRYTYPE
---------------------
ST_Point
ST_Point
(2 rows)
```

To create a table that contains polygons, use the following command:

```
CREATE TABLE Polys (PolyID integer, the_geom ST_GEOMETRY(200));
```

To add polygons to the table, commands similar to the following can be used. These commands define the points which are the vertices of two square polygons:

```
INSERT INTO Polys VALUES (1, inza..ST_WKTToSQL('Polygon ((1000 1000, 1000
5000, 4000 5000, 4000 1000, 1000 1000))'));
INSERT INTO Polys VALUES (2, inza..ST_WKTToSQL('Polygon ((100 100, 100
500, 400 500, 400 100, 100 100))'));
```

With the polygons defined in the Polys table, basic point-in-polygon queries can be performed by joining through the ST_Intersects() function, as follows:

```
SELECT PointID, PolyID FROM PointData AS a, Polys AS b WHERE
inza..ST_Intersects(a.the_geom, b.the_geom);
 POINTID | POLYID
---------+--------
       1 | 1
       1 | 1
(2 rows)
```

About the Netezza Spatial ESRI Data Representation

When you create a geometric object or load existing geometry data into the Netezza, the data is saved in an internal format referred to as the Netezza Spatial ESRI Data representation. For example, the following commands create a small table named geoms and add a polygon and linestring object to the table:

```
CREATE TABLE geoms (PolyID INTEGER, the_geom ST_GEOMETRY(64000));
CREATE TABLE

INSERT INTO geoms VALUES (1, inza..ST_WKTtoSQL('Polygon ((1 1, 1 4, 3.5
2.5, 6 4, 6 1, 1 1))'));
INSERT 0 1

INSERT INTO geoms VALUES (2, inza..ST_WKTtoSQL('Linestring (1 1, 1 4, 6
4, 6 1)'));
INSERT 0 1
```

If a standard SELECT * FROM <table> query is used to view the spatial data, the geometric data does not appear in detail. To display the table contents in more readable form, use one of the Spatial ESRI functions such as ST_AsText() to render the spatial data in text format:

```
SELECT inza..ST_AsText(the_geom) FROM geoms;
                               ST_ASTEXT
-------------------------------------------------------------------------
 POLYGON ((1 1, 6 1, 6 4, 3.5 2.5, 1 4, 1 1))
 LINESTRING (1 1, 1 4, 6 4, 6 1)
(2 rows)
```

As another example, the following query outputs the spatial data in Keyhole Markup Language (KML):

```
SELECT inza..ST_AsKML(the_geom) FROM geoms;
                               ST_ASKML
---------------------------------------------------------------
 <Polygon><outerBoundaryIs><LinearRing><coordinates>1.0000000000000000,1.
0000000000000000 1.0000000000000000,4.0000000000000000
3.5000000000000000,2.5000000000000000
6.0000000000000000,4.0000000000000000
6.0000000000000000,1.0000000000000000
1.0000000000000000,1.0000000000000000</coordinates></LinearRing></outerBo
undaryIs></Polygon>
 <LineString><coordinates>1.0000000000000000,1.0000000000000000
1.0000000000000000,4.0000000000000000
6.0000000000000000,4.0000000000000000
6.0000000000000000,1.0000000000000000</coordinates></LineString>
(2 rows)
```

Understanding Geometric Data as ST_GEOMETRY

With the Netezza Spatial ESRI Package, geometric data is saved in a ST_GEOMETRY datatype column. In the Netezza database, a ST_GEOMETRY column has a maximum size of 64,000 bytes, and a database row has a maximum size of 65,535 bytes. While a geometry object can often be defined completely within one 64,000-byte ST_GEOMETRY field, some geometry object definitions could exceed the space available in one ST_GEOMETRY column, for example polygons with thousands or millions or vertices, or linestrings/multipoints with thousands of points.

Typically, a polygon that has a single ring—the outer ring--and uses XY ordinates for vertices can have up to 3990 vertices before reaching the ST_GEOMETRY column limit. The maximum number of vertices in one ST_GEOMETRY column decreases if a polygon has interior rings (holes) or if its vertices use XYZ, XYM, or XYZM ordinates.

For spatial objects larger than can be saved in one ST_GEOMETRY column, there are methods that can be uses to save those objects in the Netezza database, as described in the Loading Geometric Data into the Netezza Database section.

Although the spatial data is in a ST_GEOMETRY column, never attempt to manipulate the data in the ST_GEOMETRY fields using Netezza string functions or other operators other than Netezza Spatial ESRI Package functions. Since the spatial data is saved in an internal format as described in the About the Netezza Spatial ESRI Data Representation section, changes to the ST_GEOMETRY field corrupt the spatial data in the affected columns.

Loading Geometric Data into the Netezza Database

For geospatial data loaded from other databases or third-party spatial sources into the Netezza database, it is possible that the objects are too large to store in one ST_GEOMETRY column. It is also possible that the data could contain non-simple geometry types, or use characters or other values not supported by the Netezza loading processes.

As a best practice, Netezza recommends that you use the Feature Manipulation Engine (FME) Workbench application, which is a product of Safe Software Inc. The FME Workbench application helps prepare data sets and load them into the Netezza database.

Note: For more information about the use of the Safe Software Inc. FME Workbench product, refer to the documentation from the vendor and the online help that is available from that application.

The FME Workbench application helps ensure that the data loaded into the Netezza database is properly prepared. For example, the application can do the following:

- Detect polygons with greater than 3990 vertices and take a user-specified action such as “chop” the geometry into smaller geometries, skip/ignore the geometry, fail (or stop) the load, or generalize (or smooth) the geometry to make it small enough to fit within the ST_GEOMETRY field.
- Detect non-simple geometries, which by default are not supported by the Netezza Spatial ESRI package, and either filter them out or “buffer” them to transform them into simple geometries.
- Load binary spatial data while detecting and escaping known binary values which are not supported by the Netezza loading processes.

Using Spatial ESRI in non-INZA User Databases

When Netezza Analytics is installed, all spatial functions and stored procedures are registered in the INZA database by default. However, you can register nzspatial_esri functions and stored procedures also into non-INZA user databases. Thus, you have multiple versions of nzspatial_esri available in the system, for example, to test a newer version of nzspatial_esri.

To register nzspatial_esri functions and stored procedures into user databases, do the following steps:

1. Create the new user database by entering the following command, where <**newdb**> is the name of the user database that you want to create, and <**version number**> is the number of your Netezza Analytics version.

```
% nzsql -c "create database <newdb>;"
CREATE DATABASE
```

2. Register nzspatial_esri by entering the following command:

```
% nzcm -d <newdb> -r nzspatial_esri
```

The following messages are displayed:

```
Registering: nzspatial_esri
Netezza Spatial (ESRI)was successfully registered on <newdb>.
Registration of nzspatial_esri completed on '<newdb>'.
Log file: /nz/var/log/nzcm.20131126.13_50_34.3856.log
```

3. Run the create_inza_db.sh script by entering the following command:

```
% /nz/export/ae/utilities/bin/create_inza_db.sh <newdb>
```

The output from the create_inza_db.sh script is as follows:

```
CREATE GROUP
CREATE GROUP
CREATE GROUP
GRANT
GRANT
GRANT
GRANT
GRANT
GRANT
GRANT
GRANT
GRANT
REVOKE
ALTER GROUP
ALTER GROUP
nzspatial_esri registered in <newdb>
Setting up nzspatial_esri
Using newdb database version <version number>
                    ST_INITIALIZE
---------------------------------------------------
The metadata objects are successfully initialized.
(1 row)

GRANT
GRANT
GRANT
```

Within the user database, spatial_esri functions and stored procedures are registered as follows:

- In the INZA schema, when the full schema support NPS feature is enabled
- In the default schema, when the full schema support NPS feature disabled

APPENDIX A
Notices and Trademarks

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. (enter the year or years). All rights reserved.

Trademarks

IBM, the IBM logo, ibm.com and Netezza are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at ibm.com/legal/copytrade.shtml.

The following terms are trademarks or registered trademarks of other companies:

Adobe is a registered trademark of Adobe Systems Incorporated in the United States, and/or other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

NEC is a registered trademark of NEC Corporation.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Red Hat is a trademark or registered trademark of Red Hat, Inc. in the United States and/or other countries.

D-CC, D-C++, Diab+, FastJ, pSOS+, SingleStep, Tornado, VxWorks, Wind River, and the Wind River logo are trademarks, registered trademarks, or service marks of Wind River Systems, Inc. Tornado patent pending.

APC and the APC logo are trademarks or registered trademarks of American Power Conversion Corporation.

Other company, product or service names may be trademarks or service marks of others.

Regulatory and Compliance

Regulatory Notices

Install the NPS system in a restricted-access location. Ensure that only those trained to operate or service the equipment have physical access to it. Install each AC power outlet near the NPS rack that plugs into it, and keep it freely accessible. Provide approved circuit breakers on all power sources.

Product may be powered by redundant power sources. Disconnect ALL power sources before servicing. High leakage current. Earth connection essential before connecting supply. Courant de fuite élevé. Raccordement à la terre indispensable avant le raccordement au réseau.

Homologation Statement

This product may not be certified in your country for connection by any means whatsoever to interfaces of public telecommunications networks. Further certification may be required by law prior to making any such connection. Contact an IBM representative or reseller for any questions.

FCC - Industry Canada Statement

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio-frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case users will be required to correct the interference at their own expense.

This Class A digital apparatus meets all requirements of the Canadian Interference-Causing Equipment Regulations.

Cet appareil numérique de la classe A respecte toutes les exigences du Règlement sur le matériel brouilleur du Canada.

CE Statement (Europe)

This product complies with the European Low Voltage Directive 73/23/EEC and EMC Directive 89/336/EEC as amended by European Directive 93/68/EEC.

Warning: This is a class A product. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.

VCCI Statement

この装置は、情報処理装置等電波障害自主規制協議会 （VCCI） の基準に基づくクラス A 情報技術装置です。この装置を家庭環境で使用すると電波妨害を引き起越すことがあります。この場合には使用者が適切な対策を講ずるう要求されることがあります。