

Release Notes

MapInfo SpatialWare Version 4.6 for Microsoft SQL Server

These release notes contain information about the SpatialWare v. 4.6 release. These notes are specific to the Microsoft SQL Server implementation.

Note: This is the most up-to-date version of the release notes. It has an additional section, Errata that is not in the printed version provided with the product.

Contents:

♦ Introduction	2
♦ What's New?	2
♦ User Defined Functions	5
♦ Text Object Support	6
♦ Performance	6
♦ Replication	7
♦ Transforming Coordinate Systems	8
♦ Specifications	9
♦ ODBC and OLE	9
♦ Installation	10
♦ Known Issues	11
♦ Usage Notes	13
♦ Errata	14
♦ Copyright	14

Introduction

SpatialWare[®] allows users to store, access, manage, and manipulate spatial data as a standard part of their business data. Users may query both spatial and non-spatial data within a single SQL Server query. Spatial data may be a part of a SQL Server database for insert, update, or delete. All of the strengths of SQL Server as a relational database are extended to spatial data using SpatialWare. Spatial data may, for example, be directly managed and edited by multiple users. SpatialWare is accessible via standard SQL Server client interfaces, and other MapInfo products are compatible for use in a client-server setup. Spatial queries can be automatically executed using database triggers or scheduled jobs. Spatial data is stored as a regular column in the SQL Server database and an efficient form of spatial indexing is available with SpatialWare.

What's New?

There are several new features provided with this release. See the Installation and Setup Guide for more detailed information about these topics.

Documentation

A new section, Aggregate Functions, has been added to the online product documentation. This section explains the new spatial aggregate functions in SpatialWare for SQL Server 4.6. Another section, User Defined Functions, has also been added to describe additional new functionality and usage.

Copies of the Installation and Setup Guide and Release Notes are provided as PDF files on your SpatialWare CD. You can view them with the latest copy of Adobe Reader (free from the Adobe site at www.adobe.com), or from within your Internet browser (either Netscape, or Internet Explorer).

Functionality

New functions have been added to this release. These are the spatial aggregate functions which work across rows in a table. An aggregate function returns a single row result. Aggregate functions take spatial objects, usually a spatial column of type ST_Spatial, as input and return a single spatial object according to the logic of the aggregation request. These functions are described in the online documentation.

hg_morph_out

The MapInfo Professional 7.0 and MapX 5.0 clients can now perform the hg_morph_out operation automatically. When using these particular versions of the clients, there is no need to use the hg_morph_out syntax.

Demos for SpatialWare

New demonstrations have been created to illustrate both interesting and automated examples of combining SpatialWare functionality with SQL Server database tools.

- The demonstrations provide sample MapBasic and T-SQL code, together with examples of Spatial SQL queries.
- Database tools such as triggers and table management are used.
- Interfaces to Microsoft Word and Microsoft Excel are described and walked through with a simple example.
- All steps from Installation to Running the Examples are described.

The demos can be found at C:\Program Files\MapInfo\sw_4.6_s\demo\Client. See the files SQLServerScenario_4.6.doc for an interactive demo, and stored_proc_examples.sql for T-SQL samples.

Find_nearest function

Responding to requests for a faster find nearest function, a new form of the find nearest function, with significantly improved performance, has been implemented. This function returns the 'n' nearest neighbors to a given search, location, or geometry, where 'n' is a user supplied parameter. For more information refer to the find nearest topic under User Defined Functions in the online documentation.

Installation

For this release MapInfo is providing local license key locations. This will minimize your wait time for a license file, because you can e-mail your regional MapInfo office for the license key. Specific e-mail addresses are provided during the install procedure. Please read these notes carefully when installing SpatialWare.

OGC Data Loading

SpatialWare supports data extract and import in both the OGC Well Known Binary (WKB) and OGC Well Known Text (WKT) formats. MapInfo's commitment to open standards helps ensure inter-operability between MapInfo products and the products of other independent software vendors. See documentation on the following functions for guidelines: HG_AsText, HG_AsBinary, HG_GeometryFromBinary, HG_GeometryFromText.

OLE-DB Access

OLE-DB can be used for client programming work with SpatialWare 4.6. Tips and simple examples of its use are now provided in the Installation and Setup Guide.

Replication Support

As only SQL Server tools are involved in a setup that uses SQL Server replication, the behavior of spatial data in such a scenario is outside the control of SpatialWare. However, as we are frequently asked for help on this topic, outline notes and tips have been provided in this document in the section on Replication on page 7.

Simplified Spatializing

Two new utilities have been provided to improve your spatialize procedures:

- `sp_spatialize_db` – This is a new stored procedure, implemented to provide a simplified way of spatializing your databases.
- `sw_spatialize_tab.mbx` – This client side MapBasic utility helps with spatializing your tables. It can be installed on a client machine which has MapInfo client software.

sw_MI_import and EasyLoader

The `sw_MI_import` utility is not upgraded with this release of SpatialWare because the speed of the new EasyLoader release (7.0) is now the same as `sw_MI_import`. The EasyLoader utility is now supplied on the SpatialWare CD to provide a commonly distributed data loading utility, instead of keeping two separate utilities.

The `sw_MI_import` utility has not been upgraded to support collection type geometries. It is still supplied with the same functionality as the last release as an alternate utility, or for prior users. EasyLoader can be used from the server machine or from networked client machines, and has both a graphical user interface, and a command-line option (useful for bulk or batch data loading). Refer to the EasyLoader chapter in the Installation and Setup Guide for details on installing, copying and using the EasyLoader application.

SW_MEMBER IDENTITY

SpatialWare has been successfully integrated with the SQL Server IDENTITY feature for key columns, allowing the system to automatically increment the `SW_MEMBER` key fields. This means newly generated features no longer require the user to supply an explicit value.

The MapInfo Professional 6.5 patch and 7.0 version clients can now handle the IDENTITY property for a column in a SQL Server table. This means new features can be created much more simply.

New tables in SQL Server are now created by default with the `SW_MEMBER` column having the IDENTITY property. Client applications therefore no longer have to supply an explicit value for the `SW_MEMBER` field when creating a new feature or row on the client. When the new row(s) are saved, a new incremental value is assigned to the `SW_MEMBER` field for that row, ensuring that the unique constraint on the `SW_MEMBER` field is respected. This removes a step from the process of creating new features on the client.

Users of MapInfo Professional 6.5 who have not obtained the most recent patch should do so to make use of the IDENTITY functionality.

Users wishing to supply an explicit value at the client can override the default and not use the IDENTITY property if they so wish.

User Defined Functions

New optimized forms of spatial predicate functions have been implemented for this release, and are made available as User Defined Functions (UDFs) in SQL Server. This allows the functions to be used without the need for the *exec sp_spatial_query* syntax, so more general usage is possible. For example, they can be combined with ORDER BY syntax.

UDFs can significantly speed up spatial operations if table valued functions are used for spatial indexing. MapInfo client applications will automatically use the new function forms when appropriate. The UDFs are available for the Expert SQL Query Builder, but be aware that the syntax is more complicated than the traditional *sp_spatial_query()* form. For details refer to the online documentation on User Defined Functions.

A full set of examples for these new UDFs is given in the online documentation.

UDF Syntax

The syntax shown in the examples in the online documentation for the spatial functions uses *sp_spatial_query*. When using UDFs, this example syntax needs to be changed to get the correct form for the equivalent UDF. The *sp_spatial_query* syntax and quotation marks ' ' should be removed, and where necessary *dbo* should be added. For example:

```
exec sp_spatial_query '
  select HG_Version() from mapinfo.mapinfo_mapcatalog
'
```

would need to be changed to the following for UDF use:

```
select dbo.HG_Version() from mapinfo.mapinfo_mapcatalog
```

Spatial Predicate UDF Syntax

The following example shows the changes in syntax that need to be made to use a spatial predicate function in UDF format. As it is a spatial predicate, a '1' should be added.

```
exec sp_spatial_query '
  select a.sw_member, b.sw_member
  from pubbldg a, flood100 b
  where ST_Contains(a.sw_geometry, b.sw_geometry)
'
```

becomes the following as a UDF:

```
select a.sw_member, b.sw_member
  from pubbldg a, flood100 b
  where dbo.ST_Contains(a.sw_geometry, b.sw_geometry) = 1
```

This is the general format spatial predicates take for UDF usage.

HG_Box() and ST_Point()

Two constructor functions, ST_Point() and HG_Box(), have been created in UDF form for this release.

Note: Only the constructor functions that return small geometries that fit inside a limited length varbinary can be used.

As these constructor functions return varbinary (as opposed to ASCII) they can be used to generate new geometries without the overhead of converting to or from ASCII text format. The use of these functions for SQL-based data loading is recommended where possible. For example the following original statement:

```
update table1 set sw_geometry = 'ST_Point(  
    ' + cast (longitude as varchar(40)) + ',  
    ' + cast (latitude as varchar(40)) + '  
    )'
```

can be replaced with the following:

```
update table1 set sw_geometry = dbo.ST_Point(longitude, latitude)]
```

Text Object Support

Support for the Text Object type used by MapInfo client products has been implemented between SpatialWare v. 4.6 for SQL Server and the latest MapInfo client product releases. This type is used by client applications for placing positioned text labels as if they were individual rows in a spatial table. The geometry column holds the information relating to text placement and style. The internal structure of the text object is not exposed for direct manipulation.

Performance

Download speeds from a client machine or application are much improved in this version of SpatialWare. For example, the retrieval speed for node objects is an order of magnitude higher. In addition the EasyLoader application is now much faster for loading data.

Speeds for download may vary, but it is possible to provide a tool to help estimate performance speeds for a client-server retrieval. The speed is generally governed by the volume of binary data being transferred. Spatial data retrieval generally runs at approximately 0.5 MB per second. To estimate the size of spatial data in a table, run the following commands in SQL Server:

```
select avg(datalength(sw_geometry)) from <tablename>  
select count(*) from <tablename>
```

Multiplying the two gives you the volume of binary data.

Replication

There are three main types of replication in SQL Server: snapshot, transactional, and merge.

Replication introduces additional tables into the SQL Server database, so you might see additional tables listed in client applications as a result of replication management.

Unless you want to replicate the MAPINFO_MAPCATALOG table, spatial tables will need to be made mappable at each subscriber database.

Automated Identity Management

It is recommended that this feature be used where ranges of identity values are assigned to different publishers and subscribers for merge replication. With this model it is possible to update geometries at both Subscribers and Publishers and have them propagated to the others.

IDENTITY

There are several issues in replication centered around the IDENTITY property. For further understanding on this topic, see the SQL Server material on Managing Identity Values.

Immediate Updating Mode

A variation of transactional replication, Immediate Updating, allows updates from Subscribers as well as Publishers for transactional and snapshot replication.

In Immediate Updating mode, the Subscriber cannot update or insert text or image values. This SQL Server restriction means Subscribers cannot update geometries unless Merge replication is used. However, geometries can still be replicated from Publishers to Subscribers in either Transactional or Snapshot replication.

In accordance with SQL Server guidelines, for immediate updating with a transactional replication model, the IDENTITY property should not be specified on the sw_member column at the subscribers. SQL Server recommends they be integer columns with a default value of zero. However, remember that you cannot update geometries at a subscriber with transactional replication due to the restrictions on binary types.

Replicating a Database

For snapshot replication a whole database can be replicated. For merge and transactional types, replication takes place at the table level, but not all of the tables in the spatialized database are replicable (no primary key). The spatial index should not be replicated for transactional as transactional replication is only possible for tables with a primary key. Instead, replicate the base table, spatialize the Subscriber database, and then re-create the spatial index. For merge replication, spatialize the subscriber databases, and then spatialize the individual tables.

ROWGUIDCOL

For transactional and merge replication the wizards create an additional column called ROWGUIDCOL in each table being replicated. SQL Server uses this for internal replication management. This column is a special type unrecognized by MapInfo client applications. To view tables in MapInfo client applications you should use the Column Filter option to filter out the ROWGUIDCOL column from the selected table. Having done this it should be possible to continue using the table largely ignoring the ROWGUIDCOL column.

Transforming Coordinate Systems

A lot of questions are asked about the behavior of units and transforming coordinate systems so two new examples have been provided for this release.

Distances returned from spatial functions are always in the units corresponding to the projection. If you want to obtain a measurement result, such as length or area, it is possible to transform the data to another projection using HG_CSTransform(), then use the measurement function (which returns the distance in the current projection system) to return the measurement value in the units appropriate to the new projection.

Example 1 – HG_CSTransform()

The following query takes data in the Longitude/Latitude coordinate system, re-projects it into the Mercator projection, then asks for the distance between two of the points. This distance is in meters, which is the base unit for the Mercator projection.

```
exec sp_spatial_query '
select hg_distance (HG_CSTransform(a.sw_geometry, c.srtext,
d.srtext),HG_CSTransform(b.sw_geometry, c.srtext, d.srtext) )
from Landmark1000 a,Landmark1000 b,master..HG_SpatialRef c,
master..HG_SpatialRef d
where a.sw_member = 1 and b.sw_member = 100 and
c.cs_name = 'Longitude / Latitude'
and d.cs_name = 'Mercator''
```

Example 2 – HG_SphericalDist()

This example is different from Example 1 because it uses the spherical distance for the calculation. The spherical distance calculates the distance around the sphere directly, whereas the technique used in Example 1 transforms the data and then calculates the distance in the new projection.

```
exec sp_spatial_query 'select HG_SphericalDist(a.sw_geometry,
b.sw_geometry)from Landmark1000 a,Landmark1000 b where a.sw_member = 372 and
b.sw_member = 995'
```

Note: The spherical distance technique is only appropriate for data in the Longitude/Latitude projection.

Specifications

SpatialWare 4.6 has been thoroughly tested with Microsoft SQL Server 2000 on the following platforms:

- Windows XP Professional.
- Windows 2000 (Service Pack 1), 2000 Advanced Server, 2000 Server.
- Windows NT 4.0 (Service Pack 6A), NT 4.0 Server.

For system requirements for your installation, please refer to your SpatialWare 4.6 Installation and Setup Guide.

You will need Netscape Navigator 4.5 or higher or Microsoft Internet Explorer 5 or higher (the online guide uses Frames) to view the online User's Guide that installs with SpatialWare. To view the documentation, simply point your browser to the index.htm page under the docs subdirectory where SpatialWare is installed. For example:

```
file:///C:/Program Files/MapInfo/sw_4.6_s/doc/index.htm
```

Client Compatibility

For notes relating to the use of client software with SpatialWare for SQL Server, please refer to Known Issues on page 11.

The following client applications are certified with SpatialWare 4.6 for SQL Server:

- MapInfo Professional 6.5. We strongly advise upgrading to MapInfo Professional 7.0 to take advantage of the significant performance enhancements for SpatialWare.
- MapInfo Professional 7.0
- MapX 5.0
- MapXtreme NT 2.5 (with the MapX 4.5.1 patch applied)
- EasyLoader 7.0.4 and above.

EasyLoader should be run using the MapInfo SQL Server ODBC driver. Some specific problems have been found when running with the SQL Server 2000 ODBC library.

For clients to work with SpatialWare for SQL Server, it is recommended that quoted identifiers be turned on. They can be turned on for the whole server, or for a specific database.

ODBC and OLE

Drivers

There are two main choices for the SQL Server ODBC driver to use with client applications: MapInfo SQL Server ODBC driver version 4.0, and Microsoft SQL Server ODBC Driver. MapInfo client products are certified with the MapInfo driver, which has proved to be slightly more robust in

testing. It is recommended that users of MapInfo client products use the MapInfo driver, which is installed automatically with MapInfo DBMS support from the MapInfo client installer. The Microsoft driver may also be used, as the differences in behavior are generally obscure.

Transaction Log

The transaction log is found in the tempdb database. In certain situations using the ODBC/OLE interface (such as where a stored procedure is interrupted during execution) the transaction log can grow at an uncontrollable rate. The following example uses the stored procedure `sp_spatial_query` as an illustration.

The `sp_spatial_query` procedure returns rows of data from inside an explicit transaction. In this example, ODBC is used to allocate a statement handle, and this handle is used to execute the stored procedure. If all the rows are not fetched before the statement handle is freed, in some cases (depending on internal conditions), a SQL Server Attention Event will occur, and the remainder of the stored procedure will not execute. In this case the commit does not happen, causing the ODBC program to continue execution with its transaction nesting level incremented by one (1). This means the transaction log is not truncated.

There are two solutions for this problem:

- The first solution is to fetch all rows from the stored procedure before freeing the statement handle.
- The second solution is to fetch a certain number of rows, then call `SQLMoreResults` before freeing the statement handle.

Installation

This section discusses issues that may occur during the installation process. Please follow the upgrade instructions provided in the Installation and Setup Guide to update database contents when moving from SpatialWare for SQL Server version 4.5 to 4.6.

This section discusses the following topics:

- Remote Installation
- Database Settings
- License Protection
- Error Message: "This string is too long."

Remote Installation

If you are installing SpatialWare from a location that is remote to SQL Server, then a SQL Server client must be installed at that location. SpatialWare setup requires that some SQL Server files be present in order to execute the installation correctly.

There have been changes made to the remote install procedure in SpatialWare 4.6 so that no manual intervention is necessary during the installation process. If you use this option take careful note of the privileges required for a remote installation.

Database Settings

During the installation you have the option of selecting a database to spatially enable for use with SpatialWare. This database must have the "Recursive triggers" setting turned OFF and the QUOTED_IDENTIFIER setting turned ON to work successfully with your SpatialWare installation.

License Protection

If you are upgrading to SpatialWare 4.6, you will need a new license file. SpatialWare 4.3 and 4.5 licenses are not compatible with version 4.6. All SpatialWare installations are license protected. You will be asked during the installation process to obtain a license file. You will need the installation machine's hostid and hostname to obtain a license file from SpatialWare Customer Support. This information is provided during the installation process. Your SpatialWare Installation and Setup Guide provides more detailed information about how to obtain a license file.

Error Message: "This string is too long."

The message, "This string is too long", may appear during installation instead of a failure message if your PATH environment variable is longer than 255 characters. To remedy the problem, you will have to reduce the length of your PATH.

Known Issues

There are a number of minor issues or tips that you should be aware of when using a MapInfo client with your spatialized SQL Server database. This section discusses the following topics:

- DateTime Functionality
- EasyLoader
- Geometry Column Name
- Geometry Sizes
- HG_Center_In
- Naming Objects
- Use of Square Brackets in the Expert Dialog

DateTime Functionality

There are two issues concerning DateTime functionality.

MapBasic

The MapBasic command `server_columninfo`, used in MapBasic script writing, will return an incorrect type for a DateTime column in a SQL Server table (the type float is returned instead).

Column Restriction

MapInfo clients may only display the Data portion of a SQL Server DateTime field. The time portion (if present) is not displayed. An update to a DateTime field from a client application will clear the time portion of the field, so care should be used.

EasyLoader

Per-Row Symbology

There is a particular problem with the SQL Server Windows 2000 ODBC driver that affects the per-row symbology capabilities of EasyLoader. As noted elsewhere, we recommend that the MapInfo SQL Server ODBC driver be used instead of the Microsoft Windows 2000 SQL Server ODBC driver, where these capabilities of the EasyLoader application are required.

Loading Data

It is recommended that EasyLoader is used for loading data into spatial databases, rather than the sw_MI_import utility. The sw_MI_import utility is still released, but for remote installs with SQL Server, you need to modify your PATH environment setting manually, by adding %SW_ROOT%/bin and %SW_ROOT%/shlib.

Note: The sw_MI_import utility does not provide total support for text objects, or collection geometry types.

Geometry Column Name

MapInfo client applications currently assume that the name of the geometry column is SW_GEOMETRY. This is the name that the EasyLoader and sw_MI_import applications use to create the column.

Geometry Sizes

Links between client applications and the server database have been tested with single geometries up to 250,000 nodes in size.

HG_Center_In

You should not use the empty geometry ST_Spatial() as the second parameter to HG_Center_In or HG_Center_In_3D. If you do this, the function fails and may cause subsequent queries to fail.

Naming Objects

Although SQL Server allows object names of up to 128 characters, your spatial table and column names must be no more than 32 characters in length in order to use them with MapInfo client software.

Use of Square Brackets in the Expert Dialog

MapInfo Professional will not allow the use of square brackets in the Expert dialog. As QUOTED_IDENTIFIER should be set to ON, you can use double quotes instead of square brackets. You need to delimit identifiers if they do not comply with the format of regular SQL Server identifiers.

Usage Notes

This section provides notes on the following topics:

- Case Sensitivity
- Identity
- International Support
- Non-Updateable Column Types

Case Sensitivity

To spatialize a case-insensitive database you should first run the following command on that database through the Query Analyzer. This creates a case-sensitive form of the Spatial Type:

```
exec sp_addtype ST_Spatial, 'image', 'NULL'
```

The sample databases are supplied in case-insensitive form.

Identity

The sample databases Asia, Namerica, World and Europe do not have the identity property set, as it is anticipated that the data will not be edited.

International Support

To view table names, field values and column names in other character sets you should take note of the following:

1. The collation order in the SQL Server database should be set to this character set. See SQL Server documentation.
2. You should run on a machine, both client and server, that has the operating system running in the character set selected, and that machine should have the character set as its International setting through the control panel.

Non-Updateable Column Types

The following column types cannot be updated from SpatialWare into a SQL Server database:

- VarBinary
- Uniqueidentifier
- DateTime (This can be updated, but the time information is lost if present.)
- Ntext

- binary
- Sql_variant
- TimeStamp

Errata

This section describes omissions or errors in the SpatialWare 4.6 documentation.

SQL Server STR Function

In the online documentation, the following paragraph was omitted from the beginning of the section about XY Columns. This section can be found in Concepts > User Defined Functions.

These examples use the SQL Server STR function, with only one parameter. This can cause loss of precision in the data. The full syntax is:

```
STR(float_expression , length , decimal)
```

where:

length is total length

decimal is length to the right of the decimal point

Suitable values for length and decimal should be chosen to preserve the required precision, and STR should be called with these values.

SpatialWare on the Network

This addresses the situation when SpatialWare 4.6 is installed on a network computer. If the machine is disconnected from the network, SpatialWare will not work. This is due to an issue with FlexLM Protection.

Copyright

Information in this document is subject to change without notice and does not represent a commitment on the part of the vendor or its representatives. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, without the written permission of MapInfo Corporation, One Global View, Troy, New York 12180–8399.

©1992–2003 MapInfo Corporation. All rights reserved. MapInfo Help ©1992–2003 MapInfo Corporation. All rights reserved.

Release Notes

MapInfo, MapInfo Professional, the MapInfo "Rainbow" logo, MapXtreme, MapInfo MapX, MapXtend, StreetPro, TargetPro, and SpatialWare are trademarks of MapInfo Corporation and its affiliated companies. Other marks are the property of their respective owners.

