

# OpenText Gupta SQLBase

## **Release Notes**

12.1.1

Product Released: 2017-10-04

Release Notes Revised: 2017-10-05

 **Notes:**



**Caution**

Cautions help you avoid irreversible problems. Read this information carefully and follow all instructions.



**Important**

Important notes help you avoid major problems.



**Note:** Notes provide additional information about a task.



**Tip:** Tips offer you quicker or easier ways of performing a task.

# Contents

<b>1</b>	<b>Introduction .....</b>	<b>3</b>
1.1	Release Notes revision history .....	3
<b>2</b>	<b>About Gupta SQLBase .....</b>	<b>3</b>
2.1	New features.....	3
2.1.1	SQLTalk Plus .....	3
2.1.2	Support for Volume Shadow Copy Service .....	4
2.1.3	@FORMAT function to create formatted string from number.....	5
2.1.4	LIMIT clause to limit the number of records returned from a SELECT statement .....	7
2.1.5	Encrypted server connection password .....	8
2.1.6	.NET Data Provider API additions .....	8
2.1.7	DDEX Provider support for Visual Studio 2017 .....	8
2.1.8	Automatic database conversion .....	9
2.2	Discontinued and deprecated features.....	10
<b>3</b>	<b>Packaging and documentation.....</b>	<b>10</b>
3.1	Packaging and delivery information .....	10
3.2	Related documentation.....	10
3.3	Documentation errata .....	10
<b>4</b>	<b>Supported environments and compatibility.....</b>	<b>11</b>
4.1	Supported systems.....	11
4.2	OpenText product compatibility .....	11
4.3	Language support.....	12
<b>5</b>	<b>Installation and upgrade notes.....</b>	<b>12</b>
5.1	Installation notes.....	12
5.2	Upgrade notes .....	13
<b>6</b>	<b>Patches .....</b>	<b>13</b>
<b>7</b>	<b>Fixed issues.....</b>	<b>14</b>
<b>8</b>	<b>Known issues .....</b>	<b>16</b>
<b>9</b>	<b>Contact information .....</b>	<b>18</b>

# 1 Introduction

These Release Notes provide an overview of Gupta SQLBase 12.1, including new features, delivery information, and supported platforms. OpenText recommends that you read these Release Notes in conjunction with the documentation included with the software package. If any conflicts exist, the Release Notes supersede the other documentation.

We also recommend that you check OpenText My Support (<https://support.opentext.com>) for any patches or documentation updates that may have been posted after the initial release of this product.

## 1.1 Release Notes revision history

Revision date	Sections revised	Description of revisions
2017-06-30	First release.	All new content.
2017-10-02	Second release	Updates for 12.1.1

# 2 About Gupta SQLBase

This section provides an overview of Gupta SQLBase 12.1 & above

SQLBase is a SQL relational database management system that includes a database server, database tools, and multiple front-end client interfaces. It is available to run on Microsoft Windows and Linux.

## 2.1 New features

Gupta SQLBase 12.1 includes the following new features.

- SQLTalk Plus
- Support for Microsoft Volume Shadow Copy Service
- @FORMAT function to create formatted string from number
- LIMIT clause to limit the number of records returned from a SELECT statement
- Encrypted server connection password
- .NET Data Provider API additions
- DDEX Provider support for Visual Studio 2017
- Automatic database conversion

Gupta SQLBase 12.1.1 includes the following new features.

- SQLTalk Plus can import CSV file data into tables
- SQLTalk Plus has an explorer style database administration interface (experimental)

### 2.1.1 SQLTalk Plus

SQLTalk Plus is an interactive user interface for SQLBase. SQLTalk Plus is a lot like SQLTalk, as it has the ability to enter and execute SQL queries.

It has additional capabilities that will continue to be added and enhanced with future releases. Here are some of the positive benefits of SQLTalk Plus:

- Has a modern user interface, including a ribbon bar, to make it easier to learn and use.
- Uses the SQLBase ADO.NET interface and can use ADO.NET to communicate with database servers from other vendors.
- Has a grid display option that allows you to edit table data.
- Has syntax highlighting for SQL Queries.

SQLTalk Plus performs almost all the everyday functionality of SQLTalk.

For SQLBase 12.1.1 new functionality has been added.

- SQLTalk Plus now has the option to import a CSV file into a new table.  
This is a wizard interface that will create a new table to import the data to. It offers column customization to control the creation of the destination table.
- SQLTalk Plus has an experimental explorer style interface for database administration.
  - It supports several features via UI, including LOAD/UNLOAD, and a VIEW and Stored Procedure editor.
  - It can be enabled in the advanced options.
  - This is an experimental feature, and still a work-in-progress.

## 2.1.2 Support for Volume Shadow Copy Service

SQLBase now supports Microsoft's Volume Shadow Copy Service on Microsoft Windows. This allows the system administrator to perform a backup of an entire system volume with SQLBase running. At the time of the backup, a snapshot of the system volume is taken that includes a consistent copy of the database and log files which preserves all the committed data to the time the data is backed up.

The SQLBase server registers itself as a VSS Writer and cooperates with the Volume Shadow Copy Service to allow a snapshot of the SQLBase data to be preserved, without being forced to stop the SQLBase service and restart it again later.

The following configuration is required for SQLBase to be able to participate in Volume Shadow Copy Service operations:

- The SQLBase Server (dbntsrv.exe) must be running in elevated mode (i.e. with Administrator privileges or as an unattended service).
- On 64-bit Windows, you must be running the 64-bit server. A 32-bit server running on 64-bit Windows cannot participate in Volume Shadow Copy Service operations.

Both of these requirements are imposed by Windows on processes that participate in Volume Shadow Copy Service operations.

To determine whether the server is correctly configured to participate in Volume Shadow Copy Service operations, run the following command from an elevated (Administrator) command prompt:

```
vssadmin list writers
```

The output of this command should contain a section with "Writer name: 'SQLBase\_servername'" where *servername* is the server name specified in the sql.ini configuration file.

### 2.1.3 @FORMAT function to create formatted string from number

@FORMAT

@FORMAT(number, format[, decimal[, triad[, currency]])

This function will format a numeric value number using a specified format string format, optionally overriding the default decimal separator, triad separator and/or currency symbol.

If any of decimal, triad and/or currency are not specified or are specified as NULL, the values corresponding to the locale the server is running in will be used. Use NULL if, for example, you wish to override the currency symbol but use the locale's decimal and triad separators:

@FORMAT(12.54, '\$###,##0.00', NULL, NULL, 'Yen')

The following table describes the format expression characters:

Format Character	Description
\	<b>Escape character.</b> The '\ ' makes the following character be output as-is without being interpreted as a format character. For example, to place an explicit '#' character in the output use '\# '.
0	<b>Zero fill.</b> If the number to format includes a digit in the position represented by the '0' character in the format, the digit will be output. Otherwise, a '0' character will be output. Any '#' or '_' characters to the right of a '0' character on the left hand side of the decimal or any '#' or '_' characters to the left of a '0' character on the right hand side of the decimal will be interpreted as a '0' character. For example, the format '0_###.#_0' would be interpreted as '0000.000'.
—	<b>Space fill.</b> If the number to format includes a digit in the position represented by the '_' character in the format, the digit will be output. Otherwise, a space character will be output. Use this character to format a fixed-width number with no leading/trailing zeros. If a triad separator would follow a position that is output as a space, the triad separator will also be output as a space. Any '#' characters to the right of a '_' on the left-hand side of the decimal or any '#' characters to the left of a '_' on the right hand side of the decimal will be interpreted as '_' assuming they are not being interpreted as '0' (see description of the '0' formatting character). For example, the format '#_#0_##.#_0#_#' would be interpreted as '#_0000.000_# '.
#	<b>Zero suppression.</b> If the number to format includes a digit in the position represented by the '#' character in the format, the digit will be output. Otherwise, no characters will be output. Any triad separators that would have been output after a digit that is suppressed will also be suppressed.

,	<p><b>Triad (thousands) separator.</b> Any ‘,’ format character after the first ‘0’, ‘_’, or ‘#’ format character on the left-hand side of the decimal point specifies that the number should be displayed with triad (thousands) separators.</p> <p>All required triad separators will be inserted in the correct positions depending on the number of digits output to the left of the decimal point.</p> <p>The separator used will depend on the server locale.</p> <p>Any ‘,’ format characters after the first will be ignored.</p>														
.	<p><b>Decimal point.</b> The first ‘.’ format character will be interpreted as the location of the decimal point. It will be replaced with the decimal separator specific to the locale the server is running in.</p> <p>If there are no digits to display after the decimal separator, there will be no decimal separator output.</p> <p>Any ‘.’ format characters after the first will be ignored.</p>														
%	<p><b>Percentage.</b> Multiplies the value by 100 and adds the ‘%’ character. If a format has more than one ‘%’ character, the value will be multiplied by 100 for each ‘%’ character in the format.</p>														
\$	<p><b>Fixed Currency symbol.</b> The first ‘\$’ symbol in the format will be replaced with the server locale’s currency symbol. Any other ‘\$’ symbols will be ignored.</p>														
\$ \$	<p><b>Floating Currency symbol.</b> A ‘\$ \$’ sequence before the first digit specifier (‘0’, ‘_’, or ‘#’) specifies a floating currency symbol. The locale-specific currency symbol will be placed immediately before the first displayed digit (or the decimal point) in the formatted number. You can use the ‘\$ \$’ sequence combined with the ‘_’ format character to create a fixed-width format with a currency symbol that appears immediately before the first actual digit. For example, the format ‘(\$ \$ ____, __0.00)’ will display the following results:</p> <table border="1" data-bbox="400 1093 1396 1335"> <thead> <tr> <th>Input value</th> <th>Result</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>( \$0.00)</td> </tr> <tr> <td>1.23</td> <td>( \$1.23)</td> </tr> <tr> <td>123.45</td> <td>( \$123.45)</td> </tr> <tr> <td>1234.56</td> <td>( \$1,234.56)</td> </tr> <tr> <td>123456.78</td> <td>(\$123,456.78)</td> </tr> <tr> <td>1234567.89</td> <td>(\$1,234,567.89)</td> </tr> </tbody> </table> <p>Note that the last case extends past the format due to there being more digits than are specified in the format.</p> <p>If a ‘\$ \$’ sequence occurs after the first digit specifier (‘0’, ‘_’, or ‘#’) it will be interpreted as a fixed currency symbol.</p> <p>If a ‘\$ \$’ occurs after a fixed currency symbol it will be ignored.</p>	Input value	Result	0	( \$0.00)	1.23	( \$1.23)	123.45	( \$123.45)	1234.56	( \$1,234.56)	123456.78	(\$123,456.78)	1234567.89	(\$1,234,567.89)
Input value	Result														
0	( \$0.00)														
1.23	( \$1.23)														
123.45	( \$123.45)														
1234.56	( \$1,234.56)														
123456.78	(\$123,456.78)														
1234567.89	(\$1,234,567.89)														
<b>E0</b> <b>e0</b> <b>E+0</b> <b>e+0</b> <b>E-0</b> <b>e-0</b>	<p><b>Scientific notation (exponential).</b> The first occurrence of ‘E’, ‘e’, ‘E-’, ‘E+’, ‘e-’, or ‘e+’ followed immediately by one or more consecutive ‘0’ characters specifies that the number should be displayed in scientific notation.</p> <p>The number will be scaled appropriately to fill all digits specified to the left of the decimal point.</p> <p>The number of consecutive ‘0’ characters to the right of the ‘E’, ‘e’, ‘E-’, ‘E+’, ‘e-’, or ‘e+’ specifies how many digits will be displayed for the exponent.</p>														
;	<p><b>Separator.</b> Separates positive, negative and zero formats. For example, ‘##0.00; -##0.00; 0’.</p> <p>If there is no zero format, zero will be formatted using the positive format.</p> <p>If there is no positive format, negative numbers will be formatted using the positive format and a ‘-’ character will be inserted at the beginning of the formatted value.</p>														

If the number has more digits to the right of the decimal point than there are '0', '\_', or '#' characters to the right of the format, the formatted number will be rounded to as many decimal places as there are '0', '\_', or '#' characters to the right of the decimal point.

If the number has more digits to the left of the decimal point than there are '0', '\_', or '#' characters to the left of the decimal point in the format, the formatted number will include the extra digits and triad separators will be inserted as needed.

### 2.1.3.1 Examples

Number	Format	Result
123.45	0000.0000	0123.4500
123.45	[####.####]	[123.45]
123.45	[____.____]	[ 123.45 ]
1234.56	[\$\$ __, __0.00]	[ \$1,234.56]
1234.56	[\$ __, __0.00]	[\$ 1,234.56]
1234.56	##.##E00	12.35E02
.123	###.##%	12.3%
123.4567	#.##	123.46
123.456	#.##; (#.##); ZERO	123.46
-123.46	#.##; (#.##); ZERO	(123.46)
0	#.##; (#.##); ZERO	ZERO

### 2.1.3.2 Samples

```
-- Test overriding triad, decimal, currency
select count(*), @format(1234567.12, '$$ __, __0.00]', ', ', '.', 'Eur ')
  from formatflt

COUNT(*) @FORMAT(1234567.12, '$$ __, __0.00]', ', ', '.', 'EUR ')
=====
      8 [ Eur 1.234.567,12]

select count(*), @format(1234567.12, '$$ __, __0.00]', ', ', '.', NULL)
  from formatflt

COUNT(*) @FORMAT(1234567.12, '$$ __, __0.00]', ', ', '.', NULL)
=====
      8 [ $1.234.567,12]
```

## 2.1.4 LIMIT clause to limit the number of records returned from a SELECT statement

A new LIMIT clause has been added to the SELECT statement syntax that allows limiting the number of records returned by the select. The syntax of the LIMIT clause is as follows:

```
SELECT ... LIMIT number
```

The following rules apply to the LIMIT clause:



- The LIMIT clause must be at the end of the SELECT statement. If there is an INTO clause, the LIMIT clause must come after it.
- Only one LIMIT clause is allowed in a SELECT statement and it must be on the outermost SELECT. Subselects may not contain a LIMIT clause.
- In the case of UNION SELECTs, the LIMIT clause must be at the end of the complete list of SELECTs and applies to the whole group of SELECTs.
- The LIMIT clause is not allowed in a view definition.
- The LIMIT clause is part of the SELECT definition and as such may be stored and executed with the STORE and EXECUTE commands.
- The LIMIT clause may be used in an INSERT INTO ... SELECT FROM ... statement to limit the number of rows inserted.
- The *number* value in the LIMIT clause must be an integer constant.

Example:

```
SELECT emp_id
FROM employee
WHERE emp_age > 40
LIMIT 10;
```

### 2.1.5 Encrypted server connection password

Server password can be changed in the console and if set in the UI, will now be stored encrypted in the SQL.INI.

### 2.1.6 .NET Data Provider API additions

The .NET Data Provider has several new additions for developers:

- Command cancellation is now supported, using `SQLBaseCommand.Cancel()`
- “Persist Security Info” connection string option added, for controlling security of connection strings. If set to false, login information (password) is removed after connecting.
  - Note that by default, the security information will be removed. This is a change from previous versions.
- `SQLBaseException` now has Reason and Remedy split out from main message and additionally available via `DbException` using `Data[“Reason”]` and `Data[“Remedy”]`
- Added Fetching, Executing, and Connecting state flags to the `SQLBaseConnection.State` property, which are added and removed appropriately.

Additional changes:

- SQL.INI is no longer automatically searched. Use the “ini=path/to/sql.ini” connection string option to use a SQL.INI in the connection.
- `SQLBaseTransaction` is not user-instantiable. Use `SQLBaseConnection.BeginTransaction()` to return an instance.

### 2.1.7 DDEX Provider support for Visual Studio 2017

- Support for Visual Studio 2017

- Removed support for Visual Studio 2010

## 2.1.8 Automatic database conversion

Added support for automatic conversion of existing SQLBase databases starting with SQLBase version 8.5. At the first connect to an existing database, SQLBase 12.1 starts an automatic database conversion process by unloading the database and re-loading the data into a new SQLBase 12.1 database.

To prepare for automatic database conversion, please make sure there is enough disk space on the same drive as your database to make a full duplicate copy and save an unload file.

While the older version of the server is running, DEINSTALL your databases to make sure that each database is wholly contained. This means that the database .DBS file can be read without the presence of .LOG files that are used for rollback and rollforward.

After installation of this version of SQLBase, please edit the sql.ini file to enable or disable automatic database conversion for the server.

```
[dbntsrv]
; Enable automatic database conversion (0 – Disable , 1 – Enable)
AutoConvert=1
```

With AutoConvert enabled, the server will accept a connection request and check to see if a database conversion is required. If conversion is required, it will attempt to unload the older version of the database. It will then use the current version to load a new database file. If successful, it will save a copy of the original database and unload file in a sub-directory where the database file is located. If the conversion is unsuccessful, it will leave a temporary sub-directory containing the files used in its attempt. That sub-directory, in the location of the database file, can be removed when no longer wanted.

To avoid having the server make multiple conversion attempts on a database, it only makes one attempt while running. If you need multiple attempts, you must restart the server for each attempt.

The newly installed sql.ini contains more information on the autoconvert option. It contains information as follows:

```
;
; [dbntsrv]
;
; Automatic database conversion
;
; AUTOCONVERT=[0|1]
; autoconvert=[0|1] - Auto conversion is 0=Off or 1=On (default=0)
;
; For AutoConvert to work, SBHelper*.EXE helper programs must be
; present.
;
;autoconvert=0
```

Automatic database conversion does not operate on encrypted databases. It is also only available on Windows at this time.

## **2.2 Discontinued and deprecated features**

The following features have been discontinued in this release:

- Support for Microsoft Transaction Server
  - SQLBrm.exe
- Microsoft Management Console Plugin
  - SQLBmmc\*.msc

## **3 Packaging and documentation**

Downloads and documentation for Gupta SQLBase are available on OpenText My Support - <https://support.opentext.com>.

### **3.1 Packaging and delivery information**

The software and documentation for Gupta SQLBase includes:

- Windows 32-bit Install
- Windows 64-bit Install
- Redhat Linux 6 32-bit Install
- Redhat Linux 6 64-bit Install
- Redhat Linux 7 64-bit Install
- Windows Embedded Deployment Pack
- Linux Embedded Deployment Pack

### **3.2 Related documentation**

For additional information about Gupta SQLBase, or for supplemental information about related products, refer to the following documents, which are available on OpenText My Support - <https://support.opentext.com>.

### **3.3 Documentation errata**

The SQLBase 12 documentation covers SQLBase 12.1. There will be new documentation for SQLTalk Plus, which is not yet available yet.

The “SQLBase 12 New Features” guide is available online and is included in the installation as the file “SQLBase New Features.pdf”.

## 4 Supported environments and compatibility

This section provides details about supported platforms, systems, and versions.

SQLBase 12.1 on Microsoft Windows is built using Visual Studio 2017, without XP support. Between the platform toolset used and the Windows system features used, Windows 7 SP1 is now the minimum system SQLBase supports. Windows XP and Windows Vista are no longer supported.

Windows Server 2016 is supported.

For Linux, SQLBase is built on Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 7. On Red Hat Enterprise Linux 6, both 32-bit and 64-bit distributions are available. On Red Hat Enterprise Linux 7, SQLBase is 64-bit only.

### 4.1 Supported systems

SQLBase is available to run on the following systems:


- Windows 7 SP1
- Windows 8 & 8.1
- Windows 10
- Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2016
- Red Hat Enterprise Linux 6 (6.9 and above)
- Red Hat Enterprise Linux 7 (7.3 and above)

The Visual Studio plugin supports:

- Visual Studio 2012
- Visual Studio 2015
- Visual Studio 2016

### 4.2 OpenText product compatibility

The section provides details about which versions of other OpenText products are compatible with this release of Gupta SQLBase 12.1.

 **Note:** For the latest compatibility information for OpenText products, refer to the Compatibility Matrix (<https://knowledge.opentext.com/go/matrix>) on OpenText My Support.

Product name	Version	Notes
OpenText Business Intelligence (OTBI)	11.0	
OpenText Integration Center (OTIC)	16.0	
OpenText Gupta Report Builder	6.3	

OpenText Gupta Report Builder	7.0	
OpenText Gupta Team Developer	7.0	
OpenText Gupta Team Developer	6.3	
OpenText Gupta TD Mobile	2.0	
OpenText Gupta TD Mobile	2.1	

### 4.3 Language support

Gupta SQLBase is currently localized in the following languages. Additional languages may be available in future releases.

Component	Languages							
	EN	DE	JA	FR	IT	ZH	ES	RU
Gupta SQLBase Server	B	UI	UI					
Gupta SQLBase Command Center	B							
SQLTalk	B							
SQLBase Plus	B		UI					

UI = user interface only

B = both user interface and online help

## 5 Installation and upgrade notes

This section provides additional installation and upgrade information, including related or third-party product information and any required critical patches.

### 5.1 Installation notes

Before you install Gupta SQLBase, review these additional installation notes and verify related product or third-party product requirements.

On Red Hat Enterprise Linux, the SQLBase distribution does not include certain libraries that are independently available through the package manager. These packages are:

- OpenSSL
- wxWidgets
- ICU – International Components for Unicode
- Java 1.8

These can be installed via the yum package manager. You may have to allow Linux to install its updates to enable more recent versions of the packages to be installed.

The wxWidgets package requires installation of the Extra Packages for Enterprise Linux repository. The following commands will install all of the above requirements:

```
# yum -y install epel-release
# yum repolist
# yum install wxGTK3 java-1.8.0-openjdk libicu openssl
```

If you are installing the 32-bit RPMs on a 64-bit Red Hat Enterprise Linux 6, the 32-bit versions of ICU, OpenSSL and wxWidgets are required:

```
# yum install wxGTK3.i686 libicu.i686 openssl.i686
```

## 5.2 Upgrade notes

Before you upgrade, review these instructions.

If you upgrade from an older version of SQLBase and you would like your databases to migrate forward, UNINSTALL your databases while running your older version of the server. This causes the .LOG files to be rolled into the .DBS, leaving only a .DBS file.

SQLBase 12.1 can read and upgrade .DBS database files from version 8.5 to the current version, but it can't convert .LOG files that go with the .DBS.

Another reason to uninstall the database files is that you might be migrating from the SQLBase 32-bit to SQLBase 64-bit. There is another database conversion required when converting between 32-bit and 64-bit.

This is why you'll want to UNINSTALL the database files before the upgrade to SQLBase 12.1.

This is a good time to also backup your files to another location.

## 6 Patches

A *patch* is a piece of software that is designed to fix or improve a computer program or its supporting data. These may include repairs to security vulnerabilities or resolution of bugs, and may also improve usability or performance. On OpenText My Support you will find two general types of patches.

*Hotfixes* are also known as quick-fixes or bug fixes. *Updates* are also known as service packs or service releases.

## 7 Fixed issues

This section provides information about past issues that have been fixed in this release.

These are the issues fixed in SQLBase 12.1.1

Issue number	Issue description
	Various SQLTalk Plus fixes including those shown below
SQLB-2609	SELECT from table throws a 1301 MFE IMO error
SQLB-2608	Execute of create table statement crashes the session
SQLB-2607	Loading a file with embedded commit fails
SQLB-2606	SET/SHOW RECOVERY not implemented
SQLB-2605	Cannot do a SET SERVER to create a new database
SQLB-2600	Improvements to error listing and tooltips
SQLB-2594	Connecting, disconnecting and connecting again throws error
	Other issues
SQLB-2176	A quoted \$LONG or \$BLOB cannot be used as data in an insert statement with SQLTalk
SQLB-2624	Index corruption when mixing characters with country.sql character equivalents
SQLB-2599	Creating a role when a session has two connections hangs the creation
SQLB-2589	CHECK DATABASE caused crash. Source of corruption fixed
SQLB-2552	Datetime parsing was incorrect when the query is over 32K in size. There was an overflow of a short length field.
SQLB-2595	Doing an install of just the client and utilities misses gptutilcom.dll
SQLB-2604	ODBC driver setup doesn't allow modification and doesn't allow a new setup

SQLB-2636	<p>Fixed the autocommit state not being properly restored after an explicit SQLBaseTransaction.Commit or Rollback. The SQLBaseCommand.Transaction property is also set to null now.</p> <p>Applications reusing SQLBaseCommand objects after an explicit transaction ends may notice a change in behavior.</p>
SQLB-2630	<p>"GetColumnSchemaRowset result set column DATA_TYPE has changed from a string to a UInt16. The value conforms to the OLEDB DataTypeEnum. See: <a href="https://docs.microsoft.com/en-us/sql/ado/reference/ado-api/datatypeenum">https://docs.microsoft.com/en-us/sql/ado/reference/ado-api/datatypeenum</a>.</p> <p>The SQLBase column type is available as a numeric value from the COLUMN_TYPE column (Gupta.SQLBase.Data.SQLBaseType).</p> <p>GetTypeSchemaRowset result set column DATA_TYPE also changed, as above. In addition, it now supports string restrictions (on DATA_TYPE, BEST_MATCH, and TYPE_NAME columns, in that order)."</p>
SQLB-2629	<p>Getting AutoIncrement values from the .NET provider doesn't work</p> <p>The ordinal of the INTERNAL_LENGTH column returned from SQLBaseConnection.GetSchema("Columns") has been increased by one to accommodate the new AUTO_INCREMENT_NEXT column.</p>
SQLB-2611	Changing the cmdtimeout default changes the server's sql.ini file
SQLB-2615	DDEX Provider doesn't save login info
SQLB-2613	Change AUTOLOCKTABLE server setting default from 0 (NEVER) to 1 (SMART)

These are the issues fixed in SQLBase 12.1

Issue number	Issue description
SQLB-2583	.NET provider not returning non-zero values for last three digits of microsecond time
SQLB-2578	An empty string passed as the replacement string for @REPLACE now returns an expected result.
SQLB-2565	Intermittent 'Invalid Cursor' SQL errors from .NET provider
SQLB-2563	Server crash apparently due to very long database name passed in connect string.
SQLB-2558	.NET Provider reports a timeout error



SQLB-2556	Customer reported database crashes and corruptions. Server stress tests were run with added internal diagnostics. Problems were found and fixed, mostly related to index management.
SQLB-2554	Customer reported server hang The 32-bit server was running out of reserved memory address space. The size of the memory allocated to thread stacks was reduced.
SQLB-2553	EDP built installer ignores destination install directory on Windows 10
SQLB-2551	Fix multithreading issues found by using additional runtime diagnostics
SQLB-2548	Server crash on certain SQL statement. Fixed a problem with statement execution.
SQLB-2535	Increase security by encrypting passwords in the .NET provider
SQLB-2520	Unexpected rows returned with two left outer join statements and IN clause
SQLB-2507	Certain joins over three tables chooses the wrong execution plan
SQLB-2506	Certain case of outer join is slower than in SQLBase 8.5
SQLB-2236	SQLBase crashes when more than 255 databases are installed
SQLB-961	Case of distinct clause used in a correlated subquery returns no rows

## 8 Known issues

The following known issues exist in this release.

Issue number	Issue description



## 9 Contact information

OpenText Corporation  
275 Frank Tompa Drive  
Waterloo, Ontario  
Canada, N2L 0A1

OpenText My Support: <https://support.opentext.com>

For more information, visit [www.opentext.com](http://www.opentext.com)

**Copyright © 2017 Open Text. All Rights Reserved.**

Trademarks owned by Open Text. The list of trademarks is not exhaustive of other trademarks, registered trademarks, product names, company names, brands and service names mentioned herein are property of Open Text or other respective owners.