SYBASE iAnywhere

UltraLite® .NET 编程

2009 年 2 月

11.0.1 版

版权和商标

版权所有 © 2009 iAnywhere Solutions, Inc. 部分版权所有 © 2009 Sybase, Inc. 保留所有权利。

本文档按原样提供，并不做任何形式的担保或承担任何责任（除非在您与 iAnywhere 达成的书面协议中另行规定）。

对本文档（全部或部分）的使用、打印、复制和分发须符合下列条件：1) 必须在整个或部分文档的所有副本中保留此声明和所有其它所有权声明，2) 不得修改本文档，3) 不得以任何形式表明您或 iAnywhere 之外的任何人是本文档的作者或提供者。

iAnywhere®、Sybase® 以及在 http://www.sybase.com/detail?id=1011207 上所列出的商标均为 Sybase, Inc. 或其子公司的商标。® 表示在美国注册。

文中提及的所有其它公司和产品名可能是与其相关的各个公司的商标。

目录

关于本手册 vii

关于 SQL Anywhere 文档 viii

UltraLite.NET 简介 1

UltraLite 和 .NET 2
系统要求和支持的平台 3
UltraLite.NET 体系结构 4

SQL Anywhere Explorer for UltraLite 5

SQL Anywhere Explorer 简介 6
在 Ultralite 项目中使用 SQL Anywhere Explorer 7

了解 UltraLite.NET 开发 11

在 Visual Studio .NET 中使用 SQL Anywhere 工具 12
连接到数据库 13
加密和模糊处理 15
使用 SQL 访问和操作数据 16
使用 Table API 访问和操作数据 20
管理事务 25
访问模式信息 26
处理错误 27
验证用户 28
UltraLite 应用程序中的同步 29

教程：构建 UltraLite.NET 应用程序 31

UltraLite.NET 开发教程简介 32
第 1 课：创建 Visual Studio 项目 33
第 2 课：创建 UltraLite 数据库 36
第 3 课：连接到数据库 37

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

第 4 课：插入、更新和删除数据 39
第 5 课：构建和部署应用程序 43
C# 教程的代码列表 45
Visual Basic 教程的代码列表 48

UltraLite .NET 2.0 API 参考 51

ULActiveSyncListener 接口 53
ULAuthStatusCode 枚举 56
ULBulkCopy 类 57
ULBulkCopyColumnMapping 类 69
ULBulkCopyColumnMappingCollection 类 76
ULBulkCopyOptions 枚举 85
ULCommand 类 86
ULCommandBuilder 类 121
ULConnection 类 131
ULConnectionParms 类 177
ULConnectionParms.UnusedEventHandler 委派 188
ULConnectionStringBuilder 类 189
ULCreateParms 类 206
ULCursorSchema 类 217
ULDataAdapter 类 226
ULDatabaseManager 类 238
ULDatabaseSchema 类 247
ULDataReader 类 257
ULDateOrder 枚举 296
ULDbType 枚举 297
ULDBValid 枚举 301
ULException 类 302
ULFactory 类 306
ULFileTransfer 类 312
ULFileTransferProgressData 类 327
ULFileTransferProgressListener 接口 330
ULIndexSchema 类 332
ULInfoMessageEventArgs 类 341

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULInfoMessageEventHandler 委派 344
ULMetaDataCollectionNames 类 345
ULParameter 类 354
ULParameterCollection 类 371
ULPublicationSchema 类 391
ULResultSet 类 394
ULResultSetSchema 类 421
ULRowsCopiedEventArgs 类 424
ULRowsCopiedEventHandler 委派 427
ULRowUpdatedEventArgs 类 428
ULRowUpdatedEventHandler 委派 432
ULRowUpdatingEventArgs 类 433
ULRowUpdatingEventHandler 委派 436
ULRuntimeType 枚举 437
ULServerSyncListener 接口 438
ULSQLCode 枚举 441
ULSqlPassthroughProgressListener 接口 453
ULSqlProgressData 类 455
ULSqlProgressState 枚举 457
ULStreamErrorCode 枚举 458
ULStreamType 枚举 476
ULSyncParms 类 478
ULSyncProgressData 类 491
ULSyncProgressListener 接口 501
ULSyncProgressState 枚举 503
ULSyncResult 类 505
ULTable 类 511
ULTableSchema 类 533
ULTransaction 类 547

索引 551

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

关于本手册

主题

本手册介绍 UltraLite.NET。利用 UltraLite.NET，您可以开发数据库应用程序，并将它们部署到计算机、手持式设备、移动设备或嵌入式设备。

读者

本手册面向希望利用 UltraLite 关系数据库的性能、资源效率、可靠性和安全性进行数据存储和同步的 .NET 应用程序开发人员。

关于 SQL Anywhere 文档

完整的 SQL Anywhere 文档以四种形式提供，但所包含信息均相同。

- **HTML 帮助** 联机帮助文档包含完整的 SQL Anywhere 文档，其中包括手册和 SQL Anywhere 工具的上下文相关帮助。

  如果使用 Microsoft Windows 操作系统，则联机帮助文档以 HTML 帮助 (CHM) 格式提供。若要访问此文档，请选择 [开始] » [程序] » [**SQL Anywhere 11**] » [文档] » [联机手册]。

  管理工具使用同一联机文档来实现帮助功能。

- **Eclipse** 在 Unix 平台上以 Eclipse 格式提供完整的联机帮助。要访问文档，请从 SQL Anywhere 11 安装的 *bin32* 或 *bin64* 目录下运行 *sadoc*。

- **DocCommentXchange** DocCommentXchange 是一个用于访问和讨论 SQL Anywhere 文档的社区。

  使用 DocCommentXchange 可以执行以下任务：

  - 查看文档
  - 检查是否有用户对文档各部分所做出的阐明
  - 提供建议和修正意见以在将来的版本中为所有用户改进文档

  访问 http://dcx.sybase.com。

- **PDF** 整套 SQL Anywhere 手册会以一组 Portable Document Format (PDF) 文件的形式提供。您必须有 PDF 阅读器才能查看信息。要下载 Adobe Reader，请访问 http://get.adobe.com/reader/。

  若要在 Microsoft Windows 操作系统上访问 PDF 文档，请选择 [开始] » [程序] » [**SQL Anywhere 11**] » 文档 » [联机手册 - PDF 格式]。

  要在 Unix 操作系统上访问 PDF 文档，请使用 Web 浏览器打开 *install-dir/documentation/zh/pdf/index.html*。

关于文档集中的手册

SQL Anywhere 文档由以下手册组成：

- **SQL Anywhere 11 - 简介** 本手册介绍 SQL Anywhere 11，一个提供数据管理和数据交换技术的综合数据包，通过它可以为服务器环境、台式机环境、移动环境以及远程办公环境快速开发由数据库驱动的应用程序。
- **SQL Anywhere 11 - 更改和升级** 本手册介绍 SQL Anywhere 11 以及该软件以前版本中的新功能。
- **SQL Anywhere 服务器 - 数据库管理** 本手册介绍如何运行、管理及配置 SQL Anywhere 数据库。它介绍了数据库连接、数据库服务器、数据库文件、备份过程、安全性、高可用性、使用复制服务器进行复制以及管理实用程序和选项。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

- **SQL Anywhere 服务器 - 编程** 本手册介绍如何使用 C、C++、Java、PHP、Perl、Python 和 .NET 编程语言（例如 Visual Basic 和 Visual C#）建立和部署数据库应用程序。其中介绍了各种编程接口，如 ADO.NET 和 ODBC。
- **SQL Anywhere 服务器 - SQL 参考** 本手册提供了系统过程和目录（系统表和视图）的参考信息。也介绍了 SQL 语言（搜索条件、语法、数据类型和函数）的 SQL Anywhere 实现。
- **SQL Anywhere 服务器 - SQL 的用法** 本手册介绍如何设计和创建数据库；如何导入、导出和修改数据；如何检索数据以及如何建立存储过程和触发器。
- **MobiLink - 入门** 本手册介绍基于会话的关系数据库同步系统 MobiLink。MobiLink 技术支持双向复制并且非常适用于移动计算环境。
- **MobiLink - 客户端管理** 本手册介绍如何设置、配置和同步 MobiLink 客户端。MobiLink 客户端可以是 SQL Anywhere 或者 UltraLite 数据库。本手册同时也介绍了 Dbmlsync API，通过它可无缝地将同步集成到 C++ 或 .NET 客户端应用程序中。
- **MobiLink - 服务器管理** 本手册说明如何设置和管理 MobiLink 应用程序。
- **MobiLink - 服务器启动的同步** 本手册介绍 MobiLink 服务器启动的同步，这种功能允许 MobiLink 服务器启动同步或在远程设备上进行操作。
- **QAnywhere** 本手册介绍 QAnywhere，一个用于移动、无线、台式机和膝上型客户端的消息传递平台。
- **SQL Remote** 本手册介绍用于移动计算的 SQL Remote 数据复制系统，此系统支持使用电子邮件或文件传输等间接链接共享 SQL Anywhere 统一数据库和多个 SQL Anywhere 远程数据库之间的数据。
- **UltraLite - 数据库管理和参考** 本手册介绍适用于小型设备的 UltraLite 数据库系统。
- **UltraLite - C 及 C++ 编程** 本手册介绍 UltraLite C 和 C++ 编程接口。利用 UltraLite，可以开发数据库应用程序，并将它们部署到手持式设备、移动设备或嵌入式设备。
- **UltraLite - M-Business Anywhere 编程** 本手册介绍 UltraLite for M-Business Anywhere。利用 UltraLite for M-Business Anywhere，用户可以开发基于 Web 的数据库应用程序，并将它们部署到运行 Palm OS、Windows Mobile 或 Windows 的手持式设备、移动设备或嵌入式设备。
- **UltraLite - .NET 编程** 本手册介绍 UltraLite.NET。利用 UltraLite.NET，您可以开发数据库应用程序，并将它们部署到计算机、手持式设备、移动设备或嵌入式设备。
- **UltraLiteJ** 本手册介绍 UltraLiteJ。利用 UltraLiteJ，可以在支持 Java 的环境中开发和部署数据库应用程序。UltraLiteJ 支持 BlackBerry 智能手机和 Java SE 环境。UltraLiteJ 基于 iAnywhere UltraLite 数据库产品。
- **错误消息** 本手册提供了 SQL Anywhere 错误消息及其诊断信息的完整列表。

文档约定

本节列出了本文档中使用的约定。

操作系统

SQL Anywhere 可以在各种平台上运行。在大多数情况下，该软件在所有平台上的行为都是相同的，但也有变动或限制。这些变动或限制通常基于基础操作系统（Windows、Unix），很少基于特定变型（AIX、Windows Mobile）或版本。

为了简化对操作系统的提及，本文档按如下方式对支持的操作系统进行分组：

- **Windows** Microsoft Windows 系列包括 Windows Vista 和 Windows XP（主要用于服务器、台式计算机和膝上型计算机），以及 Windows Mobile（用于移动设备）。

  除非另外指定，否则当本文档提及 Windows 时，是指所有基于 Windows 的平台，包括 Windows Mobile。

- **Unix** 除非另外指定，否则当本文档提及 Unix 时，是指所有基于 Unix 的平台，包括 Linux 和 Mac OS X。

目录和文件名

大部分情况下，对目录和文件名的引用在所有支持的平台上都是类似的，只需在不同形式之间进行简单的转换。这时需使用 Windows 约定。在细节更为复杂的情况下，文档显示所有相关形式。

下面是文档编写中用于简化目录和文件名的约定：

- **大写和小写目录名** 在 Windows 和 Unix 上，目录和文件名可以包括大写和小写字母。创建目录和文件时，文件系统会保留字母大小写。

  在 Windows 上，对目录和文件的提及*不区分大小写*。混合使用大小写的目录和文件名很常见，但使用所有小写字母来提及目录和文件的形式也很常见。SQL Anywhere 安装包含诸如 *Bin32* 和 *Documentation* 的目录。

  在 Unix 上，对目录和文件的提及*区分大小写*。混合使用大小写的目录和文件名不常见。大多数的目录和文件名全部使用小写字母。SQL Anywhere 安装包含诸如 *bin32* 和 *documentation* 的目录。

  本文档采用 Windows 形式的目录名。大多数情况下，在 Unix 上可以将大小写混合形式的目录名转换成小写字母的等效目录名。

- **分隔目录和文件名的斜线** 文档使用反斜线作为目录分隔符。例如，PDF 格式的文档位于 *install-dir\Documentation\zh\PDF*（Windows 形式）。

  在 Unix 上，用正斜线替换反斜线。PDF 文档位于 *install-dir/documentation/zh/pdf* 下。

- **可执行文件** 文档使用 Windows 约定显示可执行文件名（带有诸如 *.exe* 或 *.bat* 后缀）。在 Unix 上，可执行文件名没有后缀。

  例如，在 Windows 上，网络数据库服务器是 *dbsrv11.exe*。在 Unix 上是 *dbsrv11*。

- ***install-dir*** 在安装过程中，选择 SQL Anywhere 的安装位置。创建环境变量 SQLANY11，用来表示此位置。文档中以 *install-dir* 表示此位置。

  例如，本文档将此文件表示为 *install-dir*\readme.txt。在 Windows 上，这等同于 *%SQLANY11%\readme.txt*。在 Unix 上，这等同于 *$SQLANY11/readme.txt* 或 *${SQLANY11}/readme.txt*。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

有关 *install-dir* 缺省位置的详细信息，请参见“SQLANY11 环境变量”一节 《SQL Anywhere 服务器 - 数据库管理》。

- ***samples-dir*** 在安装过程中，选择 SQL Anywhere 随附的示例的安装位置。创建环境变量 SQLANYSAMP11，用来表示此位置。文档中以 *samples-dir* 表示此位置。

  要在 *samples-dir* 中打开 Windows 资源管理器窗口，请在 [**开始**] 菜单中，选择 [**程序**] » [**SQL Anywhere 11**] » [**示例应用程序和项目**]。

  有关 *samples-dir* 缺省位置的详细信息，请参见“SQLANYSAMP11 环境变量”一节 《SQL Anywhere 服务器 - 数据库管理》。

命令提示符和命令 shell 语法

大多数操作系统都提供一种或多种使用命令 shell 或命令提示符来输入命令和参数的方法。Windows 命令提示符包括 Command Prompt（DOS 提示符）和 4NT。Unix 命令 shell 包括 Korn shell 和 bash。每个 shell 都具有一些功能，其能力不仅仅局限于简单命令。这些功能通过特殊字符来驱动。特殊字符和功能随 shell 的不同而不同。如果没有正确使用这些特殊字符，通常会导致语法错误或意外行为。

本文档以普通形式提供命令行示例。如果这些示例中包含 shell 的特殊字符，则命令需要根据特定 shell 进行修改。修改方法不在本文档所述范围之内，但通常是在包含这些特殊字符的参数两旁加上引号，或是在特殊字符前面使用转义字符。

下面是命令行语法的一些示例，不同的平台可能会有不同的形式：

- **括号和大括号** 有些命令行选项需要一个参数，该参数将以列表形式接受详细的值指定。该列表通常用括号或大括号括起来。本文档使用括号。例如：

  ```
  -x tcpip(host=127.0.0.1)
  ```

  如果括号导致出现语法问题，用大括号替代：

  ```
  -x tcpip{host=127.0.0.1}
  ```

  如果两种形式都将产生语法问题，应按照 shell 的要求，用引号将整个参数括起来：

  ```
  -x "tcpip(host=127.0.0.1)"
  ```

- **引号** 如果必须在参数值中指定引号，该引号可能会与用于括参数的引号的传统用法发生冲突。例如，要指定值中包含双引号的加密密钥，则可能必须用引号括起密钥，然后转义嵌入的引号：

  ```
  -ek "my \"secret\" key"
  ```

  在许多 shell 中，密钥的值为 `my "secret" key`。

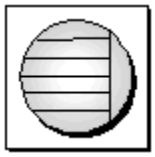
- **环境变量** 本文档介绍设置环境变量。在 Windows shell 中，环境变量使用语法 *%ENVVAR%* 来指定。在 Unix shell 中，环境变量使用语法 *$ENVVAR* 或 *${ENVVAR}* 来指定。

图标

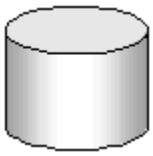
本文档中使用了下列图标。

- 客户端应用程序。

- 数据库服务器，如 Sybase SQL Anywhere。



- 数据库。在某些高水平的图中，可以使用此图标表示数据库和管理该数据库的数据库服务器。



- 复制或同步中间件。用于帮助在数据库之间共享数据。例如 MobiLink 服务器和 SQL Remote 消息代理。

- 编程接口。

联系文档小组

我们欢迎您就本帮助文档提出意见、建议和反馈信息。

要提交意见和建议，请发送电子邮件到 SQL Anywhere 文档小组，地址为 iasdoc@sybase.com。虽然我们不对这些电子邮件进行回复，但您的反馈会帮助我们改进文档，因此我们真诚地欢迎您提出宝贵的意见和建议。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

DocCommentXchange

也可以使用 DocCommentXchange 将意见或建议直接置于帮助主题中。DocCommentXchange (DCX) 是一个用于访问和讨论 SQL Anywhere 文档的社区。使用 DocCommentXchange 可以执行以下任务：

- 查看文档
- 检查是否有用户对文档各部分所做出的阐明
- 提供建议和修正意见以在将来的版本中为所有用户改进文档

访问 http://dcx.sybase.com。

查找详细信息并请求技术支持

附加信息和资源可从 Sybase iAnywhere 开发人员社区获得，网址是 http://www.sybase.com/developer/library/sql-anywhere-techcorner。

如果您有问题或是需要帮助，可将邮件发布到下面所列的 Sybase iAnywhere 新闻组。

当您向这些新闻组发布邮件时，请务必提供问题的详细信息，包括 SQL Anywhere 版本的内部版本号。可以通过运行以下命令找到此信息：**dbeng11 -v**。 **dbeng11 -v**.

新闻组位于 *forums.sybase.com* 新闻服务器上。

这些新闻组包括：

- sybase.public.sqlanywhere.general
- sybase.public.sqlanywhere.linux
- sybase.public.sqlanywhere.mobilink
- sybase.public.sqlanywhere.product_futures_discussion
- sybase.public.sqlanywhere.replication
- sybase.public.sqlanywhere.ultralite
- ianywhere.public.sqlanywhere.qanywhere

有关 Web 开发问题，请访问 http://groups.google.com/group/sql-anywhere-web-development。

新闻组免责声明

iAnywhere Solutions 没有义务为其新闻组提供解决方案、信息或建议，除提供系统操作员监控服务和确保新闻组的运行和可用性外，iAnywhere Solutions 也没有义务提供任何其它服务。

如果时间允许，iAnywhere 技术顾问以及其他员工也会对新闻组服务提供帮助。他们是在自愿的基础上提供帮助的，所以可能无法定期提供解决方案和信息。他们可以提供多少帮助取决于他们的工作量。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

UltraLite.NET 简介

目录

UltraLite 和 .NET 2
系统要求和支持的平台 3
UltraLite.NET 体系结构 4

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

UltraLite 和 .NET

可将 UltraLite.NET 应用程序部署到 Windows Mobile 和 Windows。如果部署到 Windows Mobile，则 UltraLite.NET 需要 .NET Compact Framework。如果部署到 Windows，则需要 .NET Framework。UltraLite.NET 还支持 ActiveSync 同步。

.NET Compact Framework 是适用于 Windows Mobile 的 Microsoft .NET 运行时组件。它支持多种编程语言。通过使用 UltraLite.NET，您可以使用 Visual Basic.NET 或 C# 构建应用程序。

UltraLite.NET 提供以下命名空间：

- **iAnywhere.Data.UltraLite** 此命名空间向 UltraLite 提供 ADO.NET 接口。它的优点是能够在工业标准模型上进行构建，并提供了到 SQL Anywhere ADO.NET 接口（非常相似）的迁移路径。

iAnywhere.Data.UltraLite 命名空间 (.NET 2.0)

本章介绍用于 .NET Framework 2.0 和 .NET Compact Framework 2.0 的 UltraLite .NET 数据提供程序的 API。

在本 API 参考中，用 **UL Ext.:** 标识了 UltraLite.NET 扩展，此扩展在用于 ADO.NET 的 SQL Anywhere 数据提供程序中是不可用的。

要使用 UltraLite.NET 的 UltraLite 引擎运行库，请在使用其它任何 UltraLite.NET API 之前将“RuntimeType 属性”一节第 239 页设置为适当的值。

应用程序必须先打开一个连接，然后才能对数据库执行操作。使用“ULConnection 类”一节第 131 页打开连接。

iAnywhere.Data.UltraLite 程序集使用名为 **iAnywhere.Data.UltraLite.resources** 的附属资源程序集。主程序集按如下语言顺序搜索此资源程序集：CultureInfo.CurrentUICulture，然后是 CultureInfo.CurrentCulture，最后是语言 **EN**。

本章中许多属性和方法与 OLE DB (System.Data.OleDb) 的 .NET Framework 数据提供程序非常类似。从 Microsoft .NET Framework 文档中可以找到更多信息及示例。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

系统要求和支持的平台

开发平台

要使用 UltraLite.NET 来开发应用程序，需要以下各项：

- 受支持的 Microsoft Windows 桌面操作系统版本。
- Microsoft Visual Studio 2005 或 Visual Studio 2008。
- 对于 Windows Mobile 设备，需要 .NET Compact Framework 2 或更高版本。

目标平台

UltraLite.NET 支持以下目标平台：

- Microsoft .NET Compact Framework 2.0 和 .NET Framework 2.0。
- 对于 Windows Mobile 设备，需要 Microsoft .NET Compact Framework 2 或更高版本。
- Windows 上的 Microsoft .NET Compact Framework 2 或更高版本。

除应用程序代码和 .NET Compact Framework 外，您还必须在 Windows Mobile 设备或运行 Windows 的计算机上部署以下文件：

- **iAnywhere.Data.UltraLite.dll** 包含 iAnywhere.Data.UltraLite ADO.NET 命名空间的程序集。仅当应用程序使用 iAnywhere.Data.UltraLite 命名空间时才需要此文件。
- **iAnywhere.Data.UltraLite.resources.dll** iAnywhere.Data.UltraLite ADO.NET 命名空间需要的资源。仅当应用程序使用 iAnywhere.Data.UltraLite 命名空间时才需要此文件。
- **ulnet11.dll** 此文件包含由单个客户端所使用的 UltraLite 运行库。对于每个目标平台，都提供了单独的运行库版本。
- **ulnetclient11.dll** 此文件包含 UltraLite 引擎的接口，用于允许多个客户端访问引擎。

有关 UltraLite 支持的平台的详细信息，请参见 http://www.sybase.com/detail?id=1062617。

UltraLite.NET 体系结构

将 UltraLite.NET 命名空间命名为 iAnywhere.Data.UltraLite（ADO.NET 接口）。

下列介绍了一些比较常用的 iAnywhere.Data.UltraLite ADO.NET 命名空间高层次类：

- **ULConnection** 每个 ULConnection 对象都表示一个指向 UltraLite 数据库的连接。您可以创建一个或多个 ULConnection 对象。
- **ULTable** 每个 ULTable 对象都提供对单个表中的数据的访问。
- **ULCommand 对象** 每个 ULCommand 对象都保存要对数据库执行的 SQL 语句。
- **ULDataReader 对象** 每个 ULDataReader 对象都保存单个查询的结果集。
- **ULSyncParms** 使用 ULSyncParms 对象可将 UltraLite 数据库与 MobiLink 服务器同步。

有关 ADO.NET 接口的详细信息，请参见“UltraLite .NET 2.0 API 参考”第 51 页。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

SQL Anywhere Explorer for UltraLite

目录

SQL Anywhere Explorer 简介 **6**
在 Ultralite 项目中使用 SQL Anywhere Explorer **7**

SQL Anywhere Explorer 简介

SQL Anywhere Explorer 是能够从 Visual Studio 连接到 SQL Anywhere 和 UltraLite 数据库的组件。

有关将 SQL Anywhere Explorer 与 SQL Anywhere 数据库配合使用的详细信息，请参见“SQL Anywhere Explorer”《SQL Anywhere 服务器 - 编程》。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

在 Ultralite 项目中使用 SQL Anywhere Explorer

在 Visual Studio 中，可以使用 SQL Anywhere Explorer 创建与 UltraLite 数据库的连接。在连接到数据库后，您可以：

- 浏览表和表数据
- 设计程序打开与 UltraLite 数据库的连接，或者检索和操作数据
- 将数据库对象拖放到 C# 或 Visual Basic 代码或窗体，使 IDE 自动生成引用所选对象的代码

还可以在 [**Tools**] 菜单中选择相应命令，从 Visual Studio 打开 Sybase Central 和 Interactive SQL。

安装说明

如果在已装有 Visual Studio 的 Windows 计算机上安装 UltraLite 软件，安装过程会检测到 Visual Studio 的存在，并运行必要的集成步骤。如果在安装 UltraLite 之后安装 Visual Studio，或安装新版本的 Visual Studio，必须在命令提示符处完成将 UltraLite 与 Visual Studio 集成的过程，操作步骤如下：

- 确保 Visual Studio 未运行。
- 在命令提示符处运行 *install-dir\UltraLite\UltraLite.NET\Assembly\v2\installULNet.exe*。

在 Visual Studio 中使用 UltraLite 数据库连接

使用 SQL Anywhere Explorer，在 [**Data Connections**] 节点显示 UltraLite 数据库连接。必须创建数据连接才能查看表中的数据。

您可以在 SQL Anywhere Explorer 中列出数据库表并扩展各个表以列出它们的列。Visual Studio [**Properties**] 窗格中会显示在 [SQL Anywhere Explorer] 窗口中选择的对象的属性。

♦ 在 Visual Studio 中添加 UltraLite 数据库连接

1. 选择 [**View**] » [**SQL Anywhere Explorer**]，打开 SQL Anywhere Explorer。
2. 在 [SQL Anywhere Explorer] 窗口中右击 [**Data Connections**]，然后选择 [**Add Connection**]。
3. 选择 [**UltraLite**]，然后单击 [**OK**]。
4. 输入连接到数据库所需的适当值。
5. 单击 [**OK**]。

   这样就建立了一个到数据库的连接，且该连接被添加到 [**Data Connections**] 列表。

♦ 从 Visual Studio 中删除 UltraLite 数据库连接

1. 选择 [**View**] » [**SQL Anywhere Explorer**]，打开 SQL Anywhere Explorer。
2. 在 [SQL Anywhere Explorer] 窗口中右击要删除的 UltraLite 数据连接，然后选择 [**Delete**]。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

这样该连接即从 [SQL Anywhere Explorer] 窗口删除。

配置 SQL Anywhere Explorer

Visual Studio [**Options**] 窗口包括可用来配置 SQL Anywhere Explorer 的设置。一些选项是常规选项，一些选项只能在 UltraLite 中使用。

♦ 访问 SQL Anywhere Explorer 选项

1. 从 Visual Studio [**Tools**] 菜单中选择 [**Options**]。
2. 在 [**Options**] 窗口的左窗格中展开 [**SQL Anywhere**]。
3. 单击 [**General**] 可按需要配置 SQL Anywhere Explorer 常规选项。

   限制发送到输出窗口的查询结果　指定 [**Output**] 窗口中显示的行数。缺省值是 500。

   在服务器浏览器中显示系统对象　如果要在 Microsoft Server Explorer 中查看系统对象，请选中此选项。这不是 SQL Anywhere Explorer 选项，而是 Server Explorer 选项。系统对象包括归 "dbo" 用户所有的对象。

   排序对象　选择根据对象名称或对象所有者名称在 [for UltraLite Explorer] 窗口中对对象进行排序。

   将表或视图放到设计器时生成 UI 代码　为您拖放到 Windows 窗体设计器的表或视图生成代码。

   生成用于数据适配器的 Insert、Update 和 Delete 命令　将表或视图拖放到 C# 或 Visual Basic 文档时，生成用于数据适配器的 INSERT、UPDATE 和 DELETE 命令。

   生成数据适配器的表映射　将表拖放到 C# 或 Visual Basic 文档时，生成数据适配器的表映射。
4. 单击 UltraLite 配置用于 UltraLite 的特定选项。

   将表放到代码中时生成　为拖放到应用程序代码中的表生成特定类型的代码。选择以下各项之一：

   - ULResultSet 表示一个可编辑结果集，您可在其上执行定位的更新和删除。
   - ULDataReader 表示一个只读结果集。
   - ULTable 表示用于在表中存储、删除、更新和读取数据的代码。
   - ULDataAdapter 提供了一种脱机处理表数据的方法。

使用 SQL Anywhere Explorer 添加 UltraLite 数据库对象

在 Visual Studio 中，当您将某些数据库对象从 SQL Anywhere Explorer 拖放到 Visual Studio 设计器时，IDE 自动创建引用所选对象的新组件。可以通过在 Visual Studio 中选择 [**Tools**] » [**Options**]并打开 SQL Anywhere 节点，配置拖放操作的设置。

下表列出了可以从 SQL Anywhere Explorer 拖放的 UltraLite 对象，并介绍了将对象拖放到 Visual Studio 窗体设计器或代码编辑器时创建的组件。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

项	结果
数据连接	创建数据连接。
表	创建适配器。

♦ **使用 SQL Anywhere Explorer 创建新 UltraLite 数据组件**

1. 打开想要添加数据组件的窗体或类。
2. 在 SQL Anywhere Explorer 中，选择要使用的 UltraLite 对象。
3. 将对象从 SQL Anywhere Explorer 拖放到窗体设计器或代码编辑器。

使用 SQL Anywhere Explorer 处理 UltraLite 表

使用 SQL Anywhere Explorer，可在 Visual Studio 中查看数据库中 UltraLite 表的属性和数据。

♦ **在 Visual Studio 中查看表**

1. 使用 SQL Anywhere Explorer 连接到 UltraLite 数据库。
2. 在 [SQL Anywhere Explorer] 对话框中展开 UltraLite 数据库，然后将表展开。
3. 右击表或，然后选择 [**Retrieve Data**]。

   所选的表中的数据显示在 Visual Studio 的 [**Output**] 窗口中。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

了解 UltraLite.NET 开发

目录

在 Visual Studio .NET 中使用 SQL Anywhere 工具 12
连接到数据库 13
加密和模糊处理 15
使用 SQL 访问和操作数据 16
使用 Table API 访问和操作数据 20
管理事务 25
访问模式信息 26
处理错误 27
验证用户 28
UltraLite 应用程序中的同步 29

在 Visual Studio .NET 中使用 SQL Anywhere 工具

有些 SQL Anywhere 工具已整合到 Visual Studio 2005 和 Visual Studio 2008 中，以便为 SQL Anywhere 和 UltraLite 数据库创建一个集成式数据库开发环境。

- 您可以使用 SQL Anywhere Explorer 查看表及这些表中的数据，也可以设计程序来打开与该数据库的连接以便进行数据操作和检索。从 [**视图**] 菜单中可以打开 SQL Anywhere Explorer。
- 不必退出 Visual Studio .NET 便可打开 Sybase Central 和 Interactive SQL。从 [**工具**] 菜单中可以打开 Sybase Central 和 Interactive SQL。

注意
对于 UltraLite 开发，您无法为视图添加适配器，也无法为存储过程添加命令；这些功能仅适用于 SQL Anywhere 数据库。

另请参见

- “在 Visual Studio 中使用 UltraLite 数据库连接”一节第 7 页
- “使用 SQL Anywhere Explorer 添加 UltraLite 数据库对象”一节第 8 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

连接到数据库

UltraLite 应用程序必须先连接到数据库，然后才能对数据库中的数据进行操作。本节介绍如何连接到 UltraLite 数据库。

使用 ULConnection 对象

ULConnection 对象的以下属性控制着全局应用程序行为。

- **提交行为** 缺省情况下，UltraLite.NET 应用程序处于 AutoCommit 模式。每个 Insert、Update 或 Delete 语句都会立即提交给数据库。可以使用 ULConnection.BeginTransaction 在应用程序中定义事务的起点。请参见“管理事务”一节第 25 页。
- **用户验证** 您可以使用授予或撤消连接权限的方法更改应用程序的用户 ID 缺省值 DBA 和口令缺省值 sql。每个 UltraLite 数据库最多可以定义四个用户 ID。请参见“验证用户”一节第 28 页。
- **同步** 可以通过 Connection 对象访问用于控制同步的一组对象。请参见“UltraLite 应用程序中的同步”一节第 29 页。
- **表** 使用 Connection 对象的方法可以访问 UltraLite 表。请参见“使用 Table API 访问和操作数据”一节第 20 页。
- **命令** 系统提供了一组对象来处理动态 SQL 语句的执行以及浏览结果集。请参见“使用 SQL 访问和操作数据”一节第 16 页。

请参见“ULConnection 类”一节第 131 页。

多线程应用程序

每个 ULConnection 以及从中创建的所有对象都应该在单个线程中使用。如果您的应用程序需要使用多个线程访问 UltraLite 数据库，则每个线程都需要一个单独的连接。例如，如果您将应用程序设计为在一个单独的线程中执行同步，则必须使用单独的连接来实现同步，而且必须从该线程中打开连接。

♦ 连接到 UltraLite 数据库

1. 声明一个 ULConnection 对象。

   大多数应用程序都使用一个与 UltraLite 数据库的连接，并使该连接保持打开状态。只有多线程数据访问才需要多个连接。因此，通常最好将 ULConnection 对象声明为应用程序范围内的全局对象。

   ```
   ULConnection conn;
   ```

2. 打开与现有数据库的连接。

   UltraLite 应用程序必须部署初始数据库文件，或者应用程序必须包含用来创建数据库文件的代码。可以使用 Sybase Central 或 UltraLite 提供的命令行实用程序来创建初始数据库文件。

   您可以使用连接字符串的形式或者使用 ULConnectionParms 对象来指定连接参数。下例说明了如何使用 ULConnectionParms 对象来连接名为 *mydata.udb* 的 UltraLite 数据库。

```
ULConnectionParms parms = new ULConnectionParms();
parms.DatabaseOnDesktop = "mydata.udb";
conn = new ULConnection( parms.ToString() );
conn.Open();
```

C# 代码示例
本章的代码示例是用 Microsoft C# 编写的。如果您使用的是其它某种受支持的开发工具，则必须相应地修改指令。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

加密和模糊处理

缺省情况下，新 UltraLite 数据库中的数据并未加密。通过指定适当的数据库创建参数，可以创建采用高度加密或简单模糊处理的数据库。模糊处理是一种非常弱的无密钥加密形式，它只能用于阻止随意查看数据库中的数据（例如，采用一种低级的文件检查实用程序）。

加密

要创建采用高度加密的数据库，请在使用 Sybase Central 创建数据库时指定加密密钥，如果是通过调用 ULCreateDatabase 或使用 ulcreate 实用程序创建数据库，请在创建参数中指定加密密钥。为了使加密密钥生效，它应该包含字符、数字和特殊符号的组合。使用长加密密钥可降低他人猜中密钥的几率。

数据库加密后，将无法恢复加密密钥。除非指定了正确的加密密钥，否则会彻底失去对数据库的访问。应将加密密钥视为敏感信息并妥善归档。

请参见“UltraLite DBKEY 连接参数”一节 《UltraLite - 数据库管理和参考》。

通过使用 Connection.ChangeEncryptionKey 方法应用新的加密密钥，可以更改现有 UltraLite 数据库的加密密钥。

请参见“ULConnection 类”一节第 131 页和“ULConnectionParms 类”一节第 177 页。

对数据库进行了加密后，与数据库的连接必须指定正确的加密密钥；否则连接失败。

模糊处理

要对数据库进行模糊处理，请指定 **obfuscate=y** 作为一个数据库创建参数。有关数据库加密和模糊处理参数的详细信息，请参见“为 UltraLite 选择数据库创建参数”一节 《UltraLite - 数据库管理和参考》。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

使用 SQL 访问和操作数据

UltraLite 应用程序可以使用 SQL 语句或 Table API 访问表数据。本节介绍如何使用 SQL 语句访问数据。

有关使用 Table API 的信息，请参见“使用 Table API 访问和操作数据”一节第 20 页。

本节讲解如何使用 SQL 执行以下任务：

- 插入、删除和更新行。
- 执行查询及检索结果集中的行。
- 滚动浏览结果集中的行。

本节不介绍 SQL 语言本身。有关 SQL 功能的详细信息，请参见“SQL 语句” 《SQL Anywhere 服务器 - SQL 参考》。

数据操作：INSERT、UPDATE 和 DELETE

使用 UltraLite 可执行 SQL 数据操作语言操作。可使用 ULCommand.ExecuteNonQuery 方法来执行这些操作。

请参见“ULCommand 类”一节第 86 页。

在 SQL 语句中，参数的占位符由 ? 字符表示。对于任何 INSERT、UPDATE 或 DELETE 语句，每个 ? 都是根据其在命令参数集合中的序号位置引用的。例如，第一个 ? 引用为 0，第二个引用为 1。

♦ 插入一行

1. 声明一个 ULCommand。

   ```
   ULCommand cmd;
   ```

2. 将一条 SQL 语句指派给 ULCommand 对象。

   ```
   cmd = conn.CreateCommand();
   cmd.Command = "INSERT INTO MyTable(MyColumn) values (?)";
   ```

3. 为该语句指派输入参数值。

   以下代码显示一个字符串参数。

   ```
   String newValue;
   // assign value
   cmd.Parameters.add("", newValue);
   ```

4. 执行该语句。

   返回值表示受该语句影响的行数。

   ```
   int rowsInserted = cmd.ExecuteNonQuery();
   ```

5. 如果正在使用显式事务，请提交更改。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
myTransaction.Commit();
```

♦ **更新一行**

1. 声明一个 ULCommand。

```
ULCommand cmd;
```

2. 将一条语句指派给 ULCommand 对象。

```
cmd = conn.CreateCommand();
cmd.Command = "UPDATE MyTable SET MyColumn1 = ? WHERE MyColumn2 = ?";
```

3. 为该语句指派输入参数值。

```
String newValue;
String oldValue;
// assign values
cmd.Parameters.add("", newValue);
cmd.Parameters.add("", oldValue);
```

4. 执行该语句。

```
int rowsUpdated = cmd.ExecuteNonQuery();
```

5. 如果正在使用显式事务，请提交更改。

```
myTransaction.Commit();
```

♦ **删除一行**

1. 声明一个 ULCommand。

```
ULCommand cmd;
```

2. 将一条语句指派给 ULCommand 对象。

```
cmd = conn.CreateCommand();
cmd.Command = "DELETE FROM MyTable WHERE MyColumn = ?";
```

3. 为该语句指派输入参数值。

```
String deleteValue;
// assign value
cmd.Parameters.add("", deleteValue);
```

4. 执行该语句。

```
int rowsDeleted = cmd.ExecuteNonQuery();
```

5. 如果正在使用显式事务，请提交更改。

```
myTransaction.Commit();
```

数据检索：SELECT

使用 SELECT 语句可从数据库中检索信息。本节介绍如何执行 SELECT 语句以及如何处理该语句返回的结果集。

♦ **执行 SELECT 语句**

1. 声明一个 ULCommand 对象，用于保存查询。

```
ULCommand cmd;
```

2. 将一条语句指派给该对象。

```
cmd = conn.CreateCommand();
cmd.Command = "SELECT MyColumn FROM MyTable";
```

3. 执行该语句。

   查询结果可以以其中一种对象类型的形式返回。在本示例中，使用了 ULDataReader 对象。在以下代码中，SELECT 语句的结果中包含一个字符串，该字符串被输出到命令提示符处。

```
ULDataReader customerNames = prepStmt.ExecuteReader();
int fc = customerNames.GetFieldCount();
while( customerNames.MoveNext() ) {
  for ( int i = 0; i < fc; i++ ) {
    System.Console.Write(customerNames.GetString( i ) + " " );
  }
  System.Console.WriteLine();
}
```

浏览 SQL 结果集

可以使用与 ULDataReader 对象相关联的方法浏览结果集。

结果集对象提供了以下方法来浏览结果集：

- **MoveAfterLast** 移至最后一行之后的位置。
- **MoveBeforeFirst** 移至第一行之前的位置。
- **MoveFirst** 移至第一行。
- **MoveLast** 移至最后一行。
- **MoveNext** 移至下一行。
- **MovePrevious** 移至上一行。
- **MoveRelative(offset)** 根据指定的偏移值，相对于当前行移动一定的行数。如果偏移值为正，则相对于游标在结果集中的当前位置在结果集中向前移动，如果偏移值为负，则在结果集中向后移动。如果偏移值为零，则不移动游标，但可以重新填充行缓冲区。

结果集模式说明

通过 ULDataReader.GetSchemaTable 方法和 ULDataReader.Schema 属性可以检索结果集的相关信息，如列名、总列数、列小数位数、列大小以及列 SQL 类型。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

示例

以下示例演示如何使用 ULDataReader.Schema 和 ResultSet.Schema 属性在命令提示符处显示模式信息。

```
for ( int i = 0; i < MyResultSet.Schema.GetColumnCount(); i++ ) {
      System.Console.WriteLine( MyResultSet.Schema.GetColumnName(i)
                                + " "
                                + MyResultSet.Schema.GetColumnSQLType(i) );
}
```

使用 Table API 访问和操作数据

UltraLite 应用程序可以使用 SQL 语句或 Table API 访问表数据。本节介绍如何使用 Table API 访问数据。

有关 SQL 的详细信息，请参见“使用 SQL 访问和操作数据”一节第 16 页。

本节讲解如何使用 Table API 执行以下任务：

- 滚动浏览表中的行。
- 访问当前行中的值。
- 使用查找和查寻方法来查找表中的行。
- 插入、删除和更新行。

浏览表中的行

UltraLite.NET 为您提供了若干种方法在表中进行导航以执行多种导航任务。

Table 对象提供了以下方法来浏览表。

- **MoveAfterLast** 移至最后一行之后的位置。
- **MoveBeforeFirst** 移至第一行之前的位置。
- **MoveFirst** 移至第一行。
- **MoveLast** 移至最后一行。
- **MoveNext** 移至下一行。
- **MovePrevious** 移至上一行。
- **MoveRelative(offset)** 根据指定的偏移值，相对于当前行移动一定的行数。如果偏移值为正，则相对于游标在表中的当前位置在表中向前移动，如果偏移值为负，则在表中向后移动。如果偏移值为零，则不移动游标，但可以重新填充行缓冲区。

示例

以下代码打开 MyTable 表并显示每一行的 MyColumn 列的值。

```
ULTable t = conn.ExecuteTable( "MyTable" );
int colID = t.GetOrdinal( "MyColumn" );
while ( t.MoveNext() ){
   System.Console.WriteLine( t.GetString( colID ) );
}
```

打开表对象时，应用程序可以访问表中的行。缺省情况下，这些行按主键值排序，但可在打开表时指定一个索引，以便以特定的顺序访问行。

示例

以下代码移动到按 ix_col 索引排序的 MyTable 表的第一行。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
ULTable t = conn.ExecuteTable( "MyTable", "ix_col" );
t.MoveFirst();
```

请参见“ULTable 类”一节第 511 页和“ULTableSchema 类”一节第 533 页。

使用 UltraLite 模式

UltraLite 模式确定缓冲区中的值将用于何种用途。除缺省模式外，UltraLite 还具有以下四种操作模式。

- **插入模式** 调用 insert 方法时，将缓冲区中的数据作为新行添加到表中。
- **更新模式** 调用 update 方法时，缓冲区中的数据将替换当前行。
- **查找模式** 在调用某一种查找方法时，用于定位其值与缓冲区中的数据完全匹配的行。
- **查寻模式** 在调用某一种查寻方法时，用于定位其值与缓冲区中的数据匹配或大于缓冲区中的数据的行。

访问当前行中的值

Table 对象始终位于以下位置之一：

- 在表的第一行之前。
- 在表的某一行上。
- 在表的最后一行之后。

如果 Table 对象位于某一行上，可以在适合该数据类型的一组方法中选择一种方法来检索或修改各列的值。

检索列值

Table 对象提供了一组用于检索列值的方法。这些方法都将列 ID 作为参数。

示例

以下代码检索 lname 列的值，该值是字符串。

```
int lname = t.GetOrdinal( "lname" );
string lastname = t.GetString( lname );
```

以下代码检索 cust_id 列的值，该值是一个整数。

```
int cust_id = t.GetOrdinal( "cust_id" );
int id = t.GetInt( cust_id );
```

修改列值

除用于检索值的方法外，还有用于设置值的方法。这些方法将列 ID 和值作为参数。

示例

例如，以下代码将 lname 列的值设置为 Kaminski。

```
t.SetString( lname, "Kaminski" );
```

通过将值指派给这些属性，您不用在数据库中变更数据的值。即使是在表的第一行之前或最后一行之后，您也可以为属性指派值。但是，如果当前行位于这些位置之一，那么尝试访问数据（例如，将属性指派给变量）就会出现错误。

```
// This code is incorrect
t.MoveBeforeFirst();
id = t.GetInt( cust_id );
```

转换值

所选方法必须与要指派的数据类型匹配。UltraLite 自动转换兼容的数据库数据类型，这样您就能使用 getString 方法将一个整数值读取到字符串变量中，依此类推。请参见“显式转换数据类型”一节 《UltraLite - 数据库管理和参考》。

使用查找和查寻搜索行

UltraLite 具有几种用于处理数据的操作模式。其中，查找和查寻两种模式用于搜索。Table 对象具有对应于这些模式的方法，用于定位表中的特定行。

注意
使用查找方法和查寻方法搜索的列必须在用于打开该表的索引中。

- **查找方法** 按照打开 Table 对象时指定的排序顺序，移动到与指定的搜索值完全匹配的第一行。如果找不到搜索值，则应用程序将定位到第一行之前或最后一行之后。
- **查寻方法** 按照打开 Table 对象时指定的排序顺序，移动到与指定的搜索值匹配或大于指定的搜索值的第一行。

♦ 搜索行

1. 进入查找或查寻模式。

   通过对表对象调用方法可进入该模式。例如，以下代码将进入查找模式。

   ```
   t.FindBegin();
   ```

2. 设置搜索值。

   可以通过设置当前行中的值来完成设置。设置这些值仅影响保存当前行的缓冲区，而不会影响数据库。例如，以下代码将缓冲区中的值设置为 Kaminski。

   ```
   int lname = t.GetOrdinal( "lname" );
   t.SetString( lname, "Kaminski" );
   ```

3. 搜索行。

   使用正确的方法来执行搜索。例如，以下指令在当前索引中查找与指定值完全匹配的第一行。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

对于多列索引，将始终使用第一列的值，但可以忽略其它列。

```
tCustomer.FindFirst();
```

4. 搜索该行的下一个实例。

   使用正确的方法来执行搜索。对于查找操作，FindNext 查找索引中参数的下一个实例。对于查寻操作，MoveNext 查找下一个实例。

请参见“ULTable 类”一节第 511 页。

更新行

以下过程介绍如何更新一行。

♦ **更新一行**

1. 移动到想要更新的行。

   可以通过滚动浏览表或使用查找或查寻方法在表中进行搜索来移动到某一行。

2. 进入更新模式。

   例如，以下指令在表 t 上进入更新模式。

   ```
   t.BeginUpdate();
   ```

3. 为要更新的行设置新值。

   例如，以下指令将缓冲区中的 id 列设置为 3。

   ```
   t.SetInt( id , 3);
   ```

4. 执行更新。

   ```
   t.Update();
   ```

完成更新操作后，当前行就是已更新的行。如果更改了在打开 Table 对象时指定的索引中的列的值，则当前行不确定。

缺省情况下，UltraLite.NET 在 AutoCommit 模式下运行，所以，更新会立即应用到永久存储中的行。如果已经禁用了 AutoCommit 模式，那么只有在执行提交操作后才会应用更新。请参见“管理事务”一节第 25 页。

小心
您无法更新行的主键值：而是删除该行并添加新行。

插入行

插入行的步骤与更新行的步骤非常相似，区别在于无需在执行插入操作之前在表中定位行。在表中插入行的顺序无关紧要。

示例

以下代码会插入一个新行。

```
t.InsertBegin();
t.SetInt( id, 3 );
t.SetString( lname, "Carlo" );
t.Insert();
```

如果没有设置其中一列的值，并且该列有缺省值，则使用该缺省值。如果该列没有缺省值，将使用以下条目之一：

- 对于可为空的列，添加空值。
- 对于禁止使用空值的数字列，添加 0。
- 对于禁止使用空值的字符列，添加空字符串。
- 要显式将一个值设置为空值，可使用 setDBNull 方法。

对于更新操作，在执行提交操作后，插入将应用到永久存储中的数据库。在 AutoCommit 模式中，执行插入方法本身就包含了提交操作。

删除行

删除行的步骤比插入或更新行的步骤更简单。没有与插入或更新模式对应的删除模式。

以下过程删除一行。

♦ 删除一行

1. 移到想要删除的行。
2. 执行 Table.Delete 方法。

```
t.Delete();
```


Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

管理事务

UltraLite 提供了事务处理机制以确保数据库中数据的完整性。事务是一个逻辑工作单元。要么执行整个事务，要么不执行事务中的任何语句。

缺省情况下，UltraLite.NET 在 AutoCommit 模式下运行，所以，每个插入、更新或删除都会作为一个独立事务来执行。一旦操作完成后，也就完成了对数据库的更改。

要使用多语句事务，必须通过调用 ULConnection.BeginTransaction 创建一个 ULTransaction 类对象。例如，如果应用程序在两个帐户之间转移资金，必须使用不同的操作，在减少汇出帐户金额的同时增加汇入帐户的金额，否则两个帐户都保持不变。

如果连接已执行了一个有效的事务，则必须执行 ULTransaction.Commit 语句以完成该事务，并向数据库提交更改。如果即将放弃更新集，则执行 ULTransaction.Rollback 语句取消和回退事务的所有操作。提交或回退一个事务后，连接将恢复为 AutoCommit 模式，直至下一次调用 ULConnection.BeginTransaction。

例如，以下代码段说明如何设置涉及多种操作的事务（避免缺省自动提交行为）：

```
// Assuming an already open connection named  conn
ULTransaction txn = conn.BeginTransaction(IsolationLevel.ReadUncommitted);
// Carry out transaction operations here
txn.Commit();
```

UltraLite 隔离级别
UltraLite 只支持 IsolationLevel 枚举的 IsolationLevel.ReadUncommitted 成员。

某些 SQL 语句（尤其是更改数据库结构的语句）会导致所有待执行的事务被提交。运行中自动提交事务的 SQL 语句示例为：CREATE TABLE 和 ALTER TABLE。

请参见“ULConnection 类”一节第 131 页和“ULTransaction 类”一节第 547 页。

访问模式信息

Table API 中的对象表示表、列、索引和同步发布。每个对象都有一个 Schema 属性，用于访问与该对象结构相关的信息。

您无法通过 API 修改模式。只能检索关于模式的信息。

您可以访问以下模式对象和信息：

- **DatabaseSchema** 提供数据库中表的数量和名称，以及日期和时间格式等全局属性。

  若要获取 ULDatabaseSchema 对象，请访问 ULConnection.Schema。

  请参见“ULConnection 类”一节第 131 页。

- **TableSchema** 表的列和索引的数量和名称。

  若要获取 ULTableSchema 对象，请访问 ULTable.Schema。

- **IndexSchema** 索引中列的相关信息。由于索引没有与其直接关联的数据，因此没有单独的 Index 类，而只有一个 ULIndexSchema 类。

  若要获取 ULIndexSchema 对象，请调用 ULTableSchema.GetIndex、ULTableSchema.GetOptimalIndex 或 ULTableSchema.GetPrimaryKey 方法。

- **PublicationSchema** 发布中包含的表和列的列表。发布包括 PublicationSchema 对象，但不包括 Publication 对象。

  若要获取 ULPublicationSchema 对象，请调用 ULDatabaseSchema.GetPublicationSchema 方法。

请参见“ULTableSchema 类”一节第 533 页。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

处理错误

您可以使用标准的 .NET 错误处理功能来处理错误。大多数 UltraLite 方法都会抛出 ULException 错误。可以使用 ULException.NativeError 来检索指派给此错误的 ULSQLCode 值。ULException 具有 Message 属性，使用该属性可以获取错误的描述性文本。ULSQLCode 错误是表示错误类型的负数。

有关错误代码的列表，请参见错误消息。

同步之后，可以使用连接的 SyncResult 属性来获取更详细的错误信息。例如，以下示例说明了用于报告在同步过程中发生的错误的可行方法：

```
public void Sync()
  {
         try
         {
            _conn.Synchronize( this );
            _inSync = false;
         }
         catch( ULException uEx ){
            if( uEx.NativeError == ULSQLCode.SQLE_COMMUNICATIONS_ERROR )
            {
               MessageBox.Show(
                     "StreamErrorCode = " +
                      _conn.SyncResult.StreamErrorCode.ToString() +
                      "\r\n"
                  + "StreamErrorContext = " +
                     _conn.SyncResult.StreamErrorContext + "\r\n"
                  + "StreamErrorID = " +
                     _conn.SyncResult.StreamErrorID + "\r\n"
                  + "StreamErrorSystem = " +
                     _conn.SyncResult.StreamErrorSystem + "\r\n"
                  );
            }
            else
            {
               MessageBox.Show(uEx.Message);
            }
         }
         catch(System.Exception ex )
         {
            MessageBox.Show(ex.Message);
         }
  }
```

另请参见

- “ULSyncProgressListener 接口”一节第 501 页
- “ULSyncResult 类”一节第 505 页

验证用户

必须从现有连接添加新用户。由于所有 UltraLite 数据库都是用缺省用户 ID DBA 和缺省口令 sql 创建的，因此必须首先以此用户的身份进行连接。

用户 ID 不能直接更改。但可添加新的用户 ID，然后删除现有的用户 ID。对于每个 UltraLite 数据库，UltraLite 最多支持四个用户 ID。

请参见“验证用户”一节第 28 页。

♦ 添加用户或更改现有用户的口令

1. 使用现有用户的用户 ID 和口令连接到数据库。
2. 使用 ULConnection.GrantConnectTo 方法，以设定所需口令的方式授予用户对数据库的访问权限。

   无论要添加新用户还是要更改现有用户的口令，此过程均相同。

   请参见“ULConnection 类”一节第 131 页。

♦ 删除现有用户

1. 使用现有用户的用户 ID 和口令连接到数据库。
2. 使用 Connection.RevokeConnectFrom 方法删除现有的用户。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

UltraLite 应用程序中的同步

您可以使 UltraLite 数据库与中央统一数据库同步。同步需要使用 SQL Anywhere 附带的 MobiLink 同步软件。

本节简要介绍了同步，并说明了 UltraLite.NET 用户特别感兴趣的一些功能。

有关同步的详细信息，请参见“UltraLite 客户端” 《UltraLite - 数据库管理和参考》。

还可以参见 CustDB 示例应用程序中的同步工作示例。有关详细信息，请参见 SQL Anywhere 11 安装目录下的 *Samples\UltraLite.NET\CustDB* 子目录。

UltraLite.NET 支持 TCP/IP、HTTP、HTTPS 和 TLS（传送层安全性）同步。同步由 UltraLite 应用程序启动。在所有情况中，都可以使用 SyncParms 对象的属性来控制同步。

需要单独授予许可的组成部分
ECC 加密和 FIPS 认证的加密需要单独的许可。所有高度加密技术受出口法规约束。

请参见“单独授权的组件”一节 《SQL Anywhere 11 - 简介》。

在 C# 应用程序中启动同步

以下代码说明了如何在使用 C# 编写的应用程序中启动同步。

```
private void Sync()
{
    // Sync
    try
    {
        // setup to synchronize a publication named "high_priority"
        conn.SyncParms.Publications = "high_priority";

        // Set the synchronization parameters
        conn.SyncParms.Version     = "Version1";
        conn.SyncParms.StreamParms = "";
        conn.SyncParms.Stream      = ULStreamType.TCPIP;
        conn.SyncParms.UserName    = "51";
        conn.Synchronize();
    }
    catch (System.Exception t)
        {
          MessageBox.Show("Exception: " + t.Message);
        }
}
```

向应用程序添加 ActiveSync 同步

本节介绍了如何向 UltraLite.NET 应用程序添加 ActiveSync 同步，以及如何在最终用户的计算机上注册应用程序以与 ActiveSync 配合使用。

ActiveSync 同步只能由 ActiveSync 启动。当设备放置在底座中或者从 ActiveSync 窗口选择了 [同步] 命令时，ActiveSync 会启动同步操作。

当 ActiveSync 启动同步时，ActiveSync 的 MobiLink 提供程序会启动 UltraLite 应用程序（如果它尚未运行）并向其发送消息。应用程序必须执行 ULActiveSyncListener 对象以接收和处理来自 MobiLink 提供程序的消息。应用程序必须使用 SetActiveSyncListener 方法指定监听器对象，其中，**MyAppClassName** 是应用程序的唯一 Windows 类名。

```
dbMgr.SetActiveSyncListener( "MyAppClassName", listener );
```

有关详细信息（包括示例代码），请参见“ULActiveSyncListener 接口”一节第 53 页。

当 UltraLite 收到 ActiveSync 消息时，它将在另一个线程上调用指定监听器的 ActiveSyncInvoked 方法。为避免多线程问题，ActiveSyncInvoked 方法应将事件发布到用户界面。

如果应用程序是多线程的，请使用单独的连接并利用 **lock** 关键字（在 C# 中）或 **SyncLock** 关键字（在 Visual Basic .NET 中）来访问与应用程序的其余部分共享的任何对象。ActiveSyncInvoked 方法应为其连接的 SyncParms.Stream 指定一个 ULStreamType.ACTIVE_SYNC，然后调用 ULConnection.Synchronize。

注册应用程序时，请设置以下参数：

- **类名** 该应用程序所使用的类名与 Connection.SetActiveSyncListener 方法相同。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

教程：构建 UltraLite.NET 应用程序

目录

UltraLite.NET 开发教程简介 32
第 1 课：创建 Visual Studio 项目 33
第 2 课：创建 UltraLite 数据库 36
第 3 课：连接到数据库 37
第 4 课：插入、更新和删除数据 39
第 5 课：构建和部署应用程序 43
C# 教程的代码列表 45
Visual Basic 教程的代码列表 48

UltraLite.NET 开发教程简介

本教程指导您使用 Microsoft Visual Studio 完成构建 UltraLite 应用程序的过程。它使用由 iAnywhere.Data.UltraLite 命名空间提供的 ADO.NET 接口。

本教程包含 Visual Basic 应用程序和 Visual C# 应用程序的代码。

能力和经验

本教程假定您具备以下条件：

- 熟悉 C# 编程语言或 Visual Basic 编程语言。
- 您的计算机上安装了 Microsoft Visual Studio，并且您熟悉 Visual Studio 的使用。本教程已使用 Visual Studio 2008 进行测试，涉及到的 Visual Studio 操作或过程可能会与其它版本的 Visual Studio 稍有不同。
- 您知道如何使用 Sybase Central 的 UltraLite 插件创建 UltraLite 数据库。

  请参见“使用 [创建数据库向导] 创建数据库”一节 《UltraLite - 数据库管理和参考》。

目标

本教程旨在帮助您掌握和熟悉在 Visual Studio 环境中开发 UltraLite 应用程序的过程。

安装说明

如果在已装有 Visual Studio 的 Windows 计算机上安装 UltraLite 软件，UltraLite 安装过程会检测到 Visual Studio 的存在，并运行必要的集成步骤。如果在安装 UltraLite 之后安装 Visual Studio，或安装新版本的 Visual Studio，则必须在命令提示符处手工执行 UltraLite 与 Visual Studio 的集成过程，操作步骤如下：

- 确保 Visual Studio 未运行。
- 对于 Visual Studio 2005 或更高版本，请从名为 *install-dir\UltraLite\UltraLite.NET\Assembly\v2* 的文件夹运行 *installULNet.exe*。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

第 1 课：创建 Visual Studio 项目

以下过程将创建并配置一个新的 Visual Studio 应用程序。您可以选择 Visual Basic 或 C# 作为编程语言。

本教程假定：设计 C# 应用程序时，文件在 *C:\tutorialotnet\CSApp* 目录中；设计 Visual Basic 应用程序时，文件在 *C:\tutorialotnet\VBApp* 目录中。如果您选择使用具有不同名称的目录，请在整个教程中都使用该目录。

♦ 创建一个 Visual Studio 项目

1. 创建一个 Visual Studio 项目。
   - 从 Visual Studio 的 [**File**] 菜单，选择 [**New**] » [**Project**]。
   - 随即出现 [**New Project**] 窗口。在左侧窗格中，展开 [**Visual Basic**] 文件夹或 [**Visual C#**] 文件夹。选择 [**Smart Device**] 作为项目类型。

     在右侧窗格中，选择 [**Smart Device Project**]，并根据您使用的编程语言是 Visual Basic 还是 C#，将项目命名为 **VBApp** 或 **CSApp**。
   - 输入 *C:\tutorialotnet* 作为位置，然后单击 [**OK**]。
   - 选择 [**Windows Mobile 5.0 Pocket PC SDK**] 作为目标平台。单击 [**OK**]。
2. 向项目中添加引用。
   - 将 iAnywhere.Data.UltraLite 程序集和相关的资源添加到项目中。
     a. 从 [**Project**] 菜单中，选择 [**Add Reference**]。
     b. 从可用引用的列表中，选择 [**iAnywhere.Data.UltraLite (CE)**]。单击 [**Select**] 将其添加到所选组件的列表中。

        如果此引用未在列表中出现，请单击 [**Browse**]，在 SQL Anywhere 安装目录的 *UltraLite\UltraLite.NET\ce\Assembly\v2* 子目录中查找此引用。选择 *iAnywhere.Data.UltraLite.dll* 并单击 [**OK**]。
     c. 从可用引用的列表中，选择 [**iAnywhere.Data.UltraLite (CE) EN**]。单击 [**Select**] 将其添加到所选组件的列表中。

        如果此引用未在列表中出现，请单击 [**Browse**]，在 SQL Anywhere 安装目录的 *UltraLite\UltraLite.NET\ce\xx* 子目录中查找此引用，其中，xx 是所需语言的两个字母缩写（例如，使用 **en** 代表英语）。选择 *iAnywhere.Data.UltraLite.resources.dll* 并单击 [**Open**]。
     d. 单击 [**OK**] 将该程序集和资源添加到项目中。
   - 将 UltraLite 组件链接到项目。

     在此步骤中，请确保添加了指向该组件的链接，且没有打开该组件。
     a. 从 [**Project**] 菜单中，选择 [**Add Existing Item**]，并浏览到 SQL Anywhere 安装目录的子目录 *UltraLite\UltraLite.NET\ce*。
     b. 在 [**Files of Type**] 列表中，选择 [**Executable Files**]。

c. 打开与所使用的 Windows Mobile 设备的处理器相对应的文件夹。对于 Visual Studio 2005 和更高版本，打开 *arm.50* 文件夹。选择 *ulnet11.dll*；单击 [**Add**] 按钮上的箭头并选择 [**Add as Link**]。

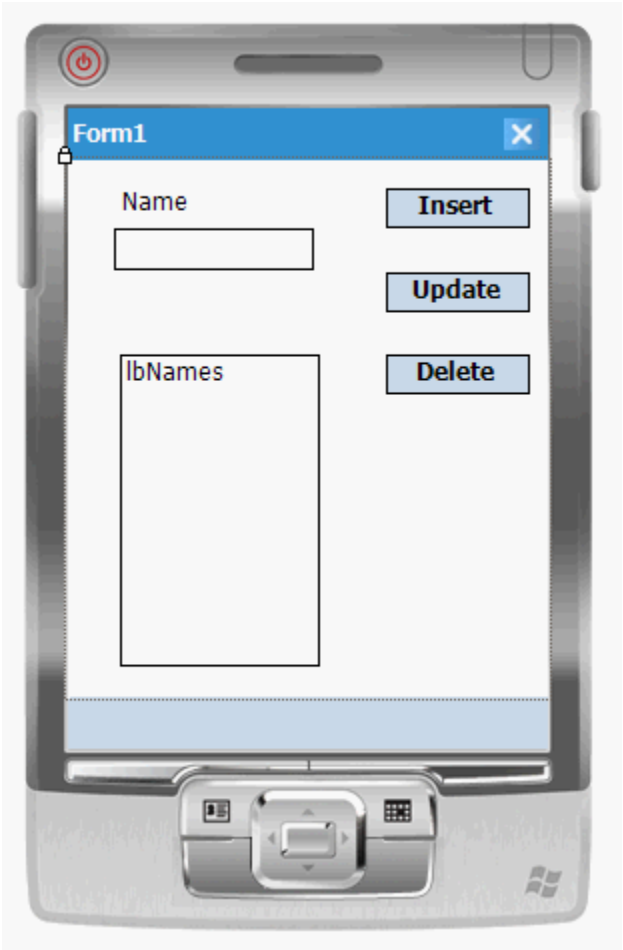
3. 为应用程序创建一个窗体。

   如果当前未显示 Visual Studio 工具箱面板，请从主菜单中选择 [**View**] » [**Toolbox**]。为窗体添加以下可视组件，方法是从工具箱中选择对象并将其拖放到窗体中的所需位置上。

Type	设计 - 名称	外观 - 文本
Button	btnInsert	Insert
Button	btnUpdate	Update
Button	btnDelete	Delete
TextBox	txtName	（没有文本）
ListBox	lbNames	（没有文本）
Label	laName	Name

   窗体看起来应如下图所示：

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1



4. 生成并部署您的解决方案。

   生成并部署解决方案可确认您已正确地配置了 Visual Studio 项目。

   a. 从 [**Build**] 菜单中，选择 [**Build Solution**]。确认项目成功生成。如果要生成 Visual Basic 应用程序，可忽略可能出现的以下警告：

      ```
      Referenced assembly 'iAnywhere.Data.UltraLite.resources' is a
      localized satellite assembly
      ```

   b. 从 [**Debug**] 菜单中，选择 [**Start Debugging**]。

      此操作将在设备或模拟器上部署应用程序并将其启动。应用程序被部署到模拟器或设备位置：*\Program Files\VBApp* 或 *\Program Files\CSApp*，视您的项目名称而定。

      部署可能需要一些时间。

   c. 确认应用程序已部署到模拟器或您的目标设备，并且您所设计的窗体 ([**Form1**]) 可以正确显示。

   d. 关闭模拟器或目标设备上的应用程序。

第 2 课：创建 UltraLite 数据库

以下过程（在台式机上执行）使用 Sybase Central 创建一个 UltraLite 数据库。

请参见“使用 [创建数据库向导] 创建数据库”一节 《UltraLite - 数据库管理和参考》。

♦ 创建数据库

1. 从 [开始] 菜单中，选择 [程序] » [SQL Anywhere 11] » [Sybase Central]。
2. 使用 Sybase Central 的 UltraLite 插件，在应用程序所在的同一目录下创建一个数据库。一般情况下，Sybase Central 所提供的缺省数据库特性是合适的。请注意以下特性：
   - **数据库文件的名称** *VBApp.udb* 或 *CSApp.udb*，视应用程序类型而定。
   - **归类序列** 使用缺省归类。
   - **使用区分大小写字符串比较** 不应开启此选项。
   - **表名** 键入 **Names**。
   - **列** 在 **Names** 表中创建具有以下属性的列：

列名	数据类型（大小）	空值	唯一	缺省值
ID	integer	否	是（主键）	全局自动增量
Name	varchar(30)	否	否	无

   - **主键** 将 ID 列指定为主键。
3. 退出 Sybase Central 并验证已在所需的目录中创建了数据库文件。
4. 将初始化的（空的）数据库文件与 Visual Studio 项目链接，以将数据库文件及应用程序代码部署到设备：
   - 从 [**Visual Studio**] 菜单，选择 [**Project**] » [**Add Existing Item**]。
   - 确保将 [**Objects of Type**] 设置为 [**All Files**]。浏览到创建数据库文件的目录，选择文件 *VBApp.udb* 或 *CSApp.udb*，这要视应用程序类型而定。
   - 单击 [**Add**] 按钮上的箭头，然后单击 [**Add As Link**]。
   - 在解决方案资源管理器框架中，右击刚刚添加到项目中的数据库文件名，然后选择 [**Properties**]。

     在属性窗格中，将 [**Build Action**] 属性设置为 [**Content**]；将 [**Copy to Output Directory**] 属性设置为 [**Copy always**]。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

第 3 课：连接到数据库

以下过程向 UltraLite.NET 应用程序中添加一个控件，通过该控件与 UltraLite 数据库建立连接。

♦ 向应用程序中添加 UltraLite 连接

1. 双击窗体以打开源文件（*Form1.cs* 或 *Form1.vb*）。
2. 添加用于导入 iAnywhere.Data.UltraLite 命名空间的代码。

   添加以下语句，作为该文件的第一行。

   ```
   //Visual C#
   using iAnywhere.Data.UltraLite;

   'Visual Basic
   Imports iAnywhere.Data.UltraLite
   ```

3. 向窗体声明中添加全局变量。

   对于 Visual C#，在描述窗体组件的代码和第一个方法声明之间添加以下代码。

   ```
   //Visual C#
   private ULConnection Conn;
   private int[] ids;
   ```

   对于 Visual Basic，在 Form1 类的开始位置添加以下代码。

   ```
   'Visual Basic
   Dim Conn As ULConnection
   Dim ids() As Integer
   ```

   这些变量的用法如下：

   - **ULConnection** Connection 对象是对数据库连接执行的所有操作的根对象。
   - **ids** ids 数组用于保存执行查询后返回的 ID 列值。

     虽然 ListBox 控件本身允许访问顺序编号，但如果删除了行，这些编号就会与 ID 列值不同。因此，ID 列值必须单独存储。

4. 双击窗体的空白处，创建一个 Form1_Load 方法。

   此方法执行以下任务：

   - 使用在 uLConnectionParms1 控件中设置的连接参数打开到数据库的连接。
   - 调用 RefreshListBox 方法（在本教程的稍后部分进行定义）。
   - 打印（显示）错误消息（如果出现错误）。对于 SQL Anywhere 错误，该代码还会打印错误代码。请参见错误消息。

   对于 C#，请将以下代码添加到 Form1_Load 方法中。

   ```
   //Visual C#
   try {
       String ConnString = "dbf=\\Program Files\\CSApp\\CSApp.udb";
       Conn = new ULConnection( ConnString );
       Conn.Open();
       Conn.DatabaseID = 1;
   ```

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
    RefreshListBox();
}
catch ( System.Exception t ) {
    MessageBox.Show( "Exception: " + t.Message);
}
```

对于 Visual Basic，请将以下代码添加到 Form1_Load 方法中。

```
'Visual Basic
Try
    Dim ConnString as String = "dbf=\Program Files\VBApp\VBApp.udb"
    Conn = New ULConnection( ConnString )
    Conn.Open()
    Conn.DatabaseID = 1
    RefreshListBox()
Catch
    MsgBox("Exception: " + err.Description)
End Try
```

5. 生成项目。

   从 [**Build**] 菜单中，选择 [**Build Solution**]。在此阶段，您可能会收到一条报告的错误；例如，在 C# 中：`error CS0103: The name 'RefreshListBox' does not exist in the class or namespace 'CSApp.Form1'`，因为 RefreshListBox 还未进行声明。下一课将添加该函数。

   如果看到其它错误，则必须先纠正它们，然后再继续。检查常见的错误，比如 C# 中的大小写一致性问题。例如，**UltraLite** 和 **ULConnection** 的大小写必须完全一致。在 Visual Basic 中，如第 3 课所述包括 **Imports iAnywhere.Data.Ultralite** 语句十分重要。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

第 4 课：插入、更新和删除数据

在这节课中，您将向应用程序中添加代码以修改数据库中的数据。以下过程使用动态 SQL。也可以使用 Table API 执行同样的技术。

请参见“使用 Table API 访问和操作数据”一节第 20 页。

以下过程创建用于维护列表框的支持方法。在接下来的过程中创建的数据操作方法将用到此方法。

♦ 添加维护列表框的代码

1. 右击窗体，然后选择 [**View Code**]。
2. 添加一个 Form1 类的方法，用于更新及填充列表框。此方法执行以下任务：
   - 清除列表框。
   - 实例化一个 ULCommand 对象并为其指派一个 SELECT 查询，该查询返回数据库中 Names 表的数据。
   - 执行查询，返回结果集作为 ULDataReader。
   - 实例化一个整数数组，其长度与结果集中的行数相同。
   - 用 ULDataReader 返回的名称填充列表框，并用 ULDataReader 返回的 ID 填充整数数组。
   - 关闭 ULDataReader。
   - 如果出现错误，则输出错误消息。对于 SQL 错误，该代码还会输出错误代码。

     请参见错误消息。

   对于 C#，向应用程序中添加以下代码，创建 Form1 类的方法。

```
//Visual C#
private void RefreshListBox(){
    try{
        long NumRows;
        int I = 0;
        lbNames.Items.Clear();
        using( ULCommand cmd = Conn.CreateCommand() ){
            cmd.CommandText = "SELECT ID, Name FROM Names";
            using( ULDataReader dr = cmd.ExecuteReader()){
                dr.MoveBeforeFirst();
                NumRows = dr.RowCount;
                ids = new int[ NumRows ];
                while (dr.MoveNext())
                {
                    lbNames.Items.Add(
                    dr.GetString(1));
                    ids[ I ] = dr.GetInt32(0);
                    I++;
                }
            }
            txtName.Text = " ";
        }
    }
    catch( Exception err ){
        MessageBox.Show(
        "Exception in RefreshListBox: " + err.Message );
```

```
    }
}
```

对于 Visual Basic，请将以下代码作为 Form1 类的一个方法添加到应用程序中。

```
'Visual Basic
Private Sub RefreshListBox()
    Try
        Dim cmd As ULCommand = Conn.CreateCommand()
        Dim I As Integer = 0
        lbNames.Items.Clear()
        cmd.CommandText = "SELECT ID, Name FROM Names"
        Dim dr As ULDataReader = cmd.ExecuteReader()
        ReDim ids(dr.RowCount)
        While (dr.MoveNext)
            lbNames.Items.Add(dr.GetString(1))
            ids(I) = dr.GetInt32(0)
            I = I + 1
        End While
        dr.Close()
        txtName.Text = " "
    Catch ex As Exception
        MsgBox(ex.ToString)
    End Try
End Sub
```

3. 生成项目。

   生成项目时不应产生错误。

♦ 实现 INSERT、UPDATE 和 DELETE

1. 在窗体设计选项卡上，双击 [**Insert**] 以创建 btnInsert_Click 方法。此方法执行以下任务：
   - 实例化一个 ULCommand 对象，并为其指派一个 INSERT 语句，该语句会将文本框中的值插入数据库。
   - 执行该语句。
   - 释放 ULCommand 对象。
   - 刷新列表框。
   - 如果出现错误，则输出错误消息。对于 SQL 错误，该代码还会输出错误代码。

     请参见错误消息。

   对于 C#，请将以下代码添加到 btnInsert_Click 方法中。

```
//Visual C#
try {
    long RowsInserted;
    using( ULCommand cmd = Conn.CreateCommand() ) {
        cmd.CommandText =
            "INSERT INTO Names(name) VALUES (?)";
        cmd.Parameters.Add("", txtName.Text);
        RowsInserted = cmd.ExecuteNonQuery();
    }
    RefreshListBox();
}
catch( Exception err ) {
    MessageBox.Show("Exception: " + err.Message );
}
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

对于 Visual Basic，请将以下代码添加到 btnInsert_Click 方法中。

```
'Visual Basic
Try
    Dim RowsInserted As Long
    Dim cmd As ULCommand = Conn.CreateCommand()
    cmd.CommandText = "INSERT INTO Names(name) VALUES (?)"
    cmd.Parameters.Add("", txtName.Text)
    RowsInserted = cmd.ExecuteNonQuery()
    cmd.Dispose()
    RefreshListBox()
Catch
    MsgBox("Exception: " + Err.Description)
End Try
```

2. 在窗体设计选项卡上，双击 [**Update**] 以创建 btnUpdate_Click 方法。此方法执行以下任务：
   - 实例化一个 ULCommand 对象，并为其指派一个 UPDATE 语句，该语句会根据关联的 ID 将文本框中的值插入数据库。
   - 执行该语句。
   - 释放 ULCommand 对象。
   - 刷新列表框。
   - 如果出现错误，则输出错误消息。对于 SQL 错误，该代码还会输出错误代码。

     请参见错误消息。

   对于 C#，请将以下代码添加到 btnUpdate_Click 方法中。

```
//Visual C#
try {
    long RowsUpdated;
    int updateID = ids[ lbNames.SelectedIndex ];
    using( ULCommand cmd = Conn.CreateCommand() ){
        cmd.CommandText =
            "UPDATE Names SET name = ? WHERE id = ?" ;
        cmd.Parameters.Add("", txtName.Text );
        cmd.Parameters.Add("", updateID);
        RowsUpdated = cmd.ExecuteNonQuery();
    }
    RefreshListBox();
}
catch( Exception err ) {
    MessageBox.Show(
        "Exception: " + err.Message);
}
```

对于 Visual Basic，请将以下代码添加到 btnUpdate_Click 方法中。

```
'Visual Basic
Try
    Dim RowsUpdated As Long
    Dim updateID As Integer = ids(lbNames.SelectedIndex)
    Dim cmd As ULCommand = Conn.CreateCommand()
    cmd.CommandText = "UPDATE Names SET name = ? WHERE id = ?"
    cmd.Parameters.Add("", txtName.Text)
    cmd.Parameters.Add("", updateID)
    RowsUpdated = cmd.ExecuteNonQuery()
    cmd.Dispose()
    RefreshListBox()
```

```
Catch
    MsgBox("Exception: " + Err.Description)
End Try
```

3. 在窗体设计选项卡上，双击 [**Delete**] 以创建 btnDelete_Click 方法。添加代码以执行以下任务：
   - 实例化一个 ULCommand 对象，并为其指派一个 DELETE 语句。该 DELETE 语句会根据整数数组 ID 中的关联 ID，从数据库中删除选定的行。
   - 执行该语句。
   - 释放 ULCommand 对象。
   - 刷新列表框。
   - 如果出现错误，则显示错误消息。对于 SQL 错误，该代码还会显示错误代码。

     请参见错误消息。

   对于 C#，请将以下代码添加到 btnDelete_Click 方法中。

```
//Visual C#
try{
    long RowsDeleted;
    int deleteID = ids[lbNames.SelectedIndex];
    using( ULCommand cmd = Conn.CreateCommand() ){
        cmd.CommandText =
            "DELETE From Names WHERE id = ?" ;
        cmd.Parameters.Add("", deleteID);
        RowsDeleted = cmd.ExecuteNonQuery ();
    }
    RefreshListBox();
}
catch( Exception err ) {
    MessageBox.Show("Exception: " + err.Message );
}
```

   对于 Visual Basic，请将以下代码添加到 btnDelete_Click 方法中。

```
'Visual Basic
Try
    Dim RowsDeleted As Long
    Dim deleteID As Integer = ids(lbNames.SelectedIndex)
    Dim cmd As ULCommand = Conn.CreateCommand()
    cmd.CommandText = "DELETE From Names WHERE id = ?"
    cmd.Parameters.Add("", deleteID)
    RowsDeleted = cmd.ExecuteNonQuery()
    cmd.Dispose()
    RefreshListBox()
Catch
    MsgBox("Exception: " + Err.Description)
End Try
```

4. 生成应用程序，确认其编译正确。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

第 5 课：构建和部署应用程序

在以下过程中，您将生成应用程序，并将其部署到一个远程设备或模拟器。

♦ 部署应用程序

1. 生成解决方案。

   确保生成应用程序时没有出错。

2. 选择部署目标。

   部署目标必须与应用程序中所包含的 *ulnet11.dll* 的版本匹配。

3. 选择 [**Debug**] » [**Start**]。

   这将生成一个包含应用程序的可执行文件，并将其部署到模拟器。此过程可能需要一些时间，尤其是在运行应用程序之前必须部署 .NET Compact Framework 时。

部署疑难解答项目清单

如果报告了错误，可以利用下面的项目清单来检查部署是否完全成功：

- 确认应用程序已部署到 *\Program Files\appname* 中，其中，*appname* 是您在第 1 课中为应用程序指定的名称（CSApp 或 VBApp）。
- 确认在应用程序代码中使用了正确的数据库文件路径。请参见“第 3 课：连接到数据库”一节第 37 页。
- 确认在向项目中添加数据库文件时选择了 [Link File]，并将 [Build Action] 设置为 [Content Only]，将 [Copy to Output Directory] 设置为 [Copy Always]。如果没有正确设置这些选项，文件就不会部署到设备。
- 确保为目标平台添加了正确版本的 *ulnet11.dll* 引用，或者运行了 Windows Mobile 安装程序。对于 Windows Mobile 5.0 之前的 Windows Mobile 版本，如果在模拟器和实际设备之间切换，则必须更改所使用的库的版本。请参见“第 1 课：创建 Visual Studio 项目”一节第 33 页。
- 您可能希望在退出模拟器时不保存模拟器状态。重新部署应用程序可将所有需要的文件复制到模拟器，并确保没有版本问题。

♦ 测试应用程序

1. 在数据库中插入数据：

   在文本框中输入一个名称，然后单击 [**Insert**]。现在，该名称应显示在列表框中。

2. 更新数据库中的数据：

   从列表框中选择一个名称。在文本框中输入一个新名称。单击 [**Update**]。现在，列表框中原名称的位置上应该显示新名称。

3. 从数据库中删除数据：

   从列表中选择一个名称。单击 [**Delete**]。列表中不再显示该名称。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

本教程结束。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

C# 教程的代码列表

以下是前几节中介绍的教程程序的完整代码。

```
using iAnywhere.Data.UltraLite;
using System;
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace CSApp
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private ULConnection Conn;
        private int[] ids;

        private void Form1_Load(object sender, EventArgs e)
        {
            try
            {
                String ConnString = "dbf=\\Program Files\\CSApp\\CSApp.udb";
                Conn = new ULConnection(ConnString);
                Conn.Open();
                Conn.DatabaseID = 1;
                RefreshListBox();
            }
            catch (System.Exception t)
            {
                MessageBox.Show("Exception: " + t.Message);
            }
        }
        private void RefreshListBox()
        {
            try
            {
                long NumRows;
                int I = 0;
                lbNames.Items.Clear();
                using (ULCommand cmd = Conn.CreateCommand())
                {
                    cmd.CommandText = "SELECT ID, Name FROM Names";
                    using (ULDataReader dr = cmd.ExecuteReader())
                    {
                        dr.MoveBeforeFirst();
                        NumRows = dr.RowCount;
                        ids = new int[NumRows];
                        while (dr.MoveNext())
                        {
                            lbNames.Items.Add(
                            dr.GetString(1));
                            ids[i] = dr.GetInt32(0);
                            I++;
                        }
```

```
                }
                txtName.Text = " ";
            }
        }
        catch (Exception err)
        {
            MessageBox.Show(
            "Exception in RefreshListBox: " + err.Message);
        }
    }

    private void btnInsert_Click(object sender, EventArgs e)
    {
        try
        {
            long RowsInserted;
            using (ULCommand cmd = Conn.CreateCommand())
            {
                cmd.CommandText =
                    "INSERT INTO Names(name) VALUES (?)";
                cmd.Parameters.Add("", txtName.Text);
                RowsInserted = cmd.ExecuteNonQuery();
            }
            RefreshListBox();
        }
        catch (Exception err)
        {
            MessageBox.Show("Exception: " + err.Message);
        }
    }

    private void btnUpdate_Click(object sender, EventArgs e)
    {
        try
        {
            long RowsUpdated;
            int updateID = ids[lbNames.SelectedIndex];
            using (ULCommand cmd = Conn.CreateCommand())
            {
                cmd.CommandText =
                    "UPDATE Names SET name = ? WHERE id = ?";
                cmd.Parameters.Add("", txtName.Text);
                cmd.Parameters.Add("", updateID);
                RowsUpdated = cmd.ExecuteNonQuery();
            }
            RefreshListBox();
        }
        catch (Exception err)
        {
            MessageBox.Show(
                "Exception: " + err.Message);
        }
    }

    private void btnDelete_Click(object sender, EventArgs e)
    {
        try
        {
            long RowsDeleted;
            int deleteID = ids[lbNames.SelectedIndex];
            using (ULCommand cmd = Conn.CreateCommand())
            {
                cmd.CommandText =
                    "DELETE From Names WHERE id = ?";
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
                    cmd.Parameters.Add("", deleteID);
                    RowsDeleted = cmd.ExecuteNonQuery();
                }
                RefreshListBox();
            }
            catch (Exception err)
            {
                MessageBox.Show("Exception: " + err.Message);
            }
        }
    }
}
```

Visual Basic 教程的代码列表

```
Imports iAnywhere.Data.UltraLite
Public Class Form1
    Dim Conn As ULConnection
    Dim ids() As Integer
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
      System.EventArgs) Handles MyBase.Load
        Try
            Dim ConnString As String = "dbf=\Program Files\VBApp\VBApp.udb"
            Conn = New ULConnection(ConnString)
            Conn.Open()
            Conn.DatabaseID = 1
            RefreshListBox()
        Catch
            MsgBox("Exception: " + Err.Description)
        End Try
    End Sub
    Private Sub RefreshListBox()
        Try
            Dim cmd As ULCommand = Conn.CreateCommand()
            Dim I As Integer = 0
            lbNames.Items.Clear()
            cmd.CommandText = "SELECT ID, Name FROM Names"
            Dim dr As ULDataReader = cmd.ExecuteReader()
            ReDim ids(dr.RowCount)
            While (dr.MoveNext)
                lbNames.Items.Add(dr.GetString(1))
                ids(I) = dr.GetInt32(0)
                I = I + 1
            End While
            dr.Close()
            txtName.Text = " "
        Catch ex As Exception
            MsgBox(ex.ToString)
        End Try
    End Sub

    Private Sub btnInsert_Click(ByVal sender As System.Object, ByVal e As
      System.EventArgs) Handles btnInsert.Click
        Try
            Dim RowsInserted As Long
            Dim cmd As ULCommand = Conn.CreateCommand()
            cmd.CommandText = "INSERT INTO Names(name) VALUES (?)"
            cmd.Parameters.Add("", txtName.Text)
            RowsInserted = cmd.ExecuteNonQuery()
            cmd.Dispose()
            RefreshListBox()
        Catch
            MsgBox("Exception: " + Err.Description)
        End Try
    End Sub

    Private Sub btnUpdate_Click(ByVal sender As System.Object, ByVal e As
      System.EventArgs) Handles btnUpdate.Click
        Try
            Dim RowsUpdated As Long
            Dim updateID As Integer = ids(lbNames.SelectedIndex)
            Dim cmd As ULCommand = Conn.CreateCommand()
            cmd.CommandText = "UPDATE Names SET name = ? WHERE id = ?"
            cmd.Parameters.Add("", txtName.Text)
            cmd.Parameters.Add("", updateID)
```


Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
            RowsUpdated = cmd.ExecuteNonQuery()
            cmd.Dispose()
            RefreshListBox()
        Catch
            MsgBox("Exception: " + Err.Description)
        End Try
    End Sub

    Private Sub btnDelete_Click(ByVal sender As System.Object, ByVal e As
      System.EventArgs) Handles btnDelete.Click
        Try
            Dim RowsDeleted As Long
            Dim deleteID As Integer = ids(lbNames.SelectedIndex)
            Dim cmd As ULCommand = Conn.CreateCommand()
            cmd.CommandText = "DELETE From Names WHERE id = ?"
            cmd.Parameters.Add("", deleteID)
            RowsDeleted = cmd.ExecuteNonQuery()
            cmd.Dispose()
            RefreshListBox()
        Catch
            MsgBox("Exception: " + Err.Description)
        End Try
    End Sub
End Class
```

UltraLite .NET 2.0 API 参考

目录

ULActiveSyncListener 接口 53
ULAuthStatusCode 枚举 56
ULBulkCopy 类 57
ULBulkCopyColumnMapping 类 69
ULBulkCopyColumnMappingCollection 类 76
ULBulkCopyOptions 枚举 85
ULCommand 类 86
ULCommandBuilder 类 121
ULConnection 类 131
ULConnectionParms 类 177
ULConnectionParms.UnusedEventHandler 委派 188
ULConnectionStringBuilder 类 189
ULCreateParms 类 206
ULCursorSchema 类 217
ULDataAdapter 类 226
ULDatabaseManager 类 238
ULDatabaseSchema 类 247
ULDataReader 类 257
ULDateOrder 枚举 296
ULDbType 枚举 297
ULDBValid 枚举 301
ULException 类 302
ULFactory 类 306
ULFileTransfer 类 312
ULFileTransferProgressData 类 327
ULFileTransferProgressListener 接口 330
ULIndexSchema 类 332
ULInfoMessageEventArgs 类 341
ULInfoMessageEventHandler 委派 344
ULMetaDataCollectionNames 类 345
ULParameter 类 354

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULParameterCollection 类 **371**
ULPublicationSchema 类 **391**
ULResultSet 类 **394**
ULResultSetSchema 类 **421**
ULRowsCopiedEventArgs 类 **424**
ULRowsCopiedEventHandler 委派 **427**
ULRowUpdatedEventArgs 类 **428**
ULRowUpdatedEventHandler 委派 **432**
ULRowUpdatingEventArgs 类 **433**
ULRowUpdatingEventHandler 委派 **436**
ULRuntimeType 枚举 **437**
ULServerSyncListener 接口 **438**
ULSQLCode 枚举 **441**
ULSqlPassthroughProgressListener 接口 **453**
ULSqlProgressData 类 **455**
ULSqlProgressState 枚举 **457**
ULStreamErrorCode 枚举 **458**
ULStreamType 枚举 **476**
ULSyncParms 类 **478**
ULSyncProgressData 类 **491**
ULSyncProgressListener 接口 **501**
ULSyncProgressState 枚举 **503**
ULSyncResult 类 **505**
ULTable 类 **511**
ULTableSchema 类 **533**
ULTransaction 类 **547**

命名空间

iAnywhere.Data.UltraLite 命名空间

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULActiveSyncListener 接口

UL Ext.: 用于接收 ActiveSync 事件的监听器接口。

语法

Visual Basic
Public Interface **ULActiveSyncListener**

C#
public interface **ULActiveSyncListener**

另请参见

- “ULActiveSyncListener 成员”一节第 53 页

ULActiveSyncListener 成员

公共方法

成员名称	说明
“ActiveSyncInvoked 方法”一节第 53 页	在 ActiveSync 的 MobiLink 提供程序调用应用程序执行同步时被调用。

另请参见

- “ULActiveSyncListener 接口”一节第 53 页

ActiveSyncInvoked 方法

在 ActiveSync 的 MobiLink 提供程序调用应用程序执行同步时被调用。

语法

Visual Basic
```
Public Sub ActiveSyncInvoked( _
   ByVal launchedByProvider As Boolean _
)
```

C#
```
public void ActiveSyncInvoked(
   bool launchedByProvider
);
```

参数

- **launchedByProvider** 如果是由 MobiLink 提供程序启动应用程序来执行 ActiveSync 同步，则为 true。应用程序在完成同步后必须自行关闭。如果当用于 ActiveSync 的 MobiLink 提供程序调用应用程序时，该应用程序已经在运行，则为 false。

注释

此方法由单独的线程调用。为避免多线程问题，它应向 UI 发布一个事件。如果您正在使用多线程，建议您使用一个单独的连接，并使用 lock 关键字来访问与应用程序的其余部分共享的所有对象。

完成同步后，应用程序应调用 ULDatabaseManager.SignalSyncIsComplete() 向 ActiveSync 的 MobiLink 提供程序发出信号。

示例

以下代码段演示了如何在 UI 线程中接收 ActiveSync 请求并执行同步。

```
' Visual Basic
Imports iAnywhere.Data.UltraLite

Public Class MainWindow
  Inherits System.Windows.Forms.Form
  Implements ULActiveSyncListener
  Private conn As ULConnection

  Public Sub New(ByVal args() As String)

    MyBase.New()

    'This call is required by the Windows Form Designer.
    InitializeComponent()

    'Add any initialization after the InitializeComponent() call
    ULConnection.DatabaseManager.SetActiveSyncListener( _
        "myCompany.myapp", Me _
      )
    'Create Connection
    ...
  End Sub

  Protected Overrides Sub OnClosing( _
        ByVal e As System.ComponentModel.CancelEventArgs _
      )
    ULConnection.DatabaseManager.SetActiveSyncListener( _
        Nothing, Nothing _
      )
    MyBase.OnClosing(e)
  End Sub

  Public Sub ActiveSyncInvoked( _
        ByVal launchedByProvider As Boolean _
      ) Implements ULActiveSyncListener.ActiveSyncInvoked
    Me.Invoke(New EventHandler(AddressOf Me.ActiveSyncAction))
  End Sub

  Public Sub ActiveSyncAction( _
        ByVal sender As Object, ByVal e As EventArgs _
      )
    ' Do active sync
    conn.Synchronize()
    ULConnection.DatabaseManager.SignalSyncIsComplete()
  End Sub
End Class


// C#
using iAnywhere.Data.UltraLite;
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
public class Form1 : System.Windows.Forms.Form, ULActiveSyncListener
{
  private System.Windows.Forms.MainMenu mainMenu1;
  private ULConnection conn;

  public Form1()
  {
    //
    // Required for Windows Form Designer support
    //
    InitializeComponent();

    //
    // TODO: Add any constructor code after
    // InitializeComponent call
    //
    ULConnection.DatabaseManager.SetActiveSyncListener(
        "myCompany.myapp", this
      );
    // Create connection
    ...
  }

  protected override void Dispose( bool disposing )
  {
    base.Dispose( disposing );
  }

  protected override void OnClosing(
      System.ComponentModel.CancelEventArgs e
    )
  {
    ULConnection.DatabaseManager.SetActiveSyncListener(
        null, null
      );
    base.OnClosing(e);
  }

  public void ActiveSyncInvoked(bool launchedByProvider)
  {
    this.Invoke( new EventHandler( ActiveSyncHandler ) );
  }

  internal void ActiveSyncHandler(object sender, EventArgs e)
  {
    conn.Synchronize();
    ULConnection.DatabaseManager.SignalSyncIsComplete();
  }
}
```

另请参见

- “ULActiveSyncListener 接口”一节第 53 页
- “ULActiveSyncListener 成员”一节第 53 页
- “SignalSyncIsComplete 方法”一节第 245 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULAuthStatusCode 枚举

UL Ext.: 枚举在 MobiLink 用户验证过程中可能报告的状态码。

语法

Visual Basic
Public Enum **ULAuthStatusCode**

C#
public enum **ULAuthStatusCode**

成员

成员名称	说明	值
EXPIRED	用户 ID 或口令已到期 - 授权失败 (EXPIRED = 3)。	3
IN_USE	用户 ID 已在使用 - 授权失败 (IN_USE = 5)。	5
INVALID	用户 ID 或口令不正确 - 授权失败 (INVALID = 4)。	4
UNKNOWN	授权状态未知，原因可能是连接尚未执行同步 (UNKNOWN = 0)。	0
VALID	用户 ID 和口令在同步时有效 (VALID = 1)。	1
VALID_BUT_EXPIRES_SOON	用户 ID 和口令在同步时有效，但很快就要到期 (VALID_BUT_EXPIRES_SOON = 2)。	2

另请参见

- “AuthStatus 属性”一节第 506 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULBulkCopy 类

以有效率的方式将另一数据源中的数据批量装载到 UltraLite 表中。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULBulkCopy**
  Implements IDisposable

C#
public sealed class **ULBulkCopy** : IDisposable

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopy 类。

另请参见

- “ULBulkCopy 成员”一节第 57 页

ULBulkCopy 成员

公共构造函数

成员名称	说明
“ULBulkCopy 构造函数”一节 第 58 页	用指定的 ULConnection 初始化 ULBulkCopy 对象。

公共属性

成员名称	说明
“BatchSize 属性”一节 第 61 页	获取或设置每个批处理中的行数。每个批处理结束时，该批处理中的行将发送到服务器。
“BulkCopyTimeout 属性”一节 第 62 页	获取或设置操作在超时之前完成所需的秒数。
“ColumnMappings 属性”一节 第 62 页	返回ULBulkCopyColumnMapping 项的集合。列映射定义数据源中的列与目标中的列之间的关系。
“DestinationTableName 属性” 一节第 63 页	获取或设置服务器上目标表的名称。
“NotifyAfter 属性”一节 第 63 页	指定要在生成通知事件之前处理的行数。

公共方法

成员名称	说明
“Close 方法”一节第 64 页	关闭 ULBulkCopy 实例。
“Dispose 方法”一节第 64 页	终止 ULBulkCopy 实例。
“WriteToServer 方法”一节第 65 页	将所提供的 System.Data.DataRow 对象的数组中的所有行复制到由 ULBulkCopy 对象的 DestinationTableName 字段所指定的目标表中。

公共事件

成员名称	说明
“ULRowsCopied 事件”一节第 67 页	每次处理完 NotifyAfter 所指定的行数时便会发生此事件。

另请参见

- “ULBulkCopy 类”一节第 57 页

ULBulkCopy 构造函数

用指定的 ULConnection 初始化 ULBulkCopy 对象。

另请参见

- “ULConnection 类”一节第 131 页

ULBulkCopy(ULConnection) 构造函数

语法

Visual Basic

```
Public Sub New( _
  ByVal connection As ULConnection _
)
```

C#

```
public  ULBulkCopy(
  ULConnection connection
);
```

参数

- **connection**　将用于执行批量复制操作且已打开的 ULConnection。如果连接未打开，则 WriteToServer 中会抛出异常。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopy 类。

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页
- “ULBulkCopy 构造函数”一节第 58 页
- “ULConnection 类”一节第 131 页

ULBulkCopy(String) 构造函数

用指定的连接字符串初始化 ULBulkCopy 对象。

语法

Visual Basic

```
Public Sub New( _
  ByVal connectionString As String _
)
```

C#

```
public  ULBulkCopy(
  string  connectionString
);
```

参数

- **connectionString** 一个字符串，用于定义将打开以供 ULBulkCopy 实例使用的连接。连接字符串是以分号分隔的 "关键字=值" 对的列表。

  有关参数的列表，请参见“ConnectionString 属性”一节第 137 页。

注释

此语法在 WriteToServer 期间使用 connectionString 打开连接。连接在 WriteToServer 结束时关闭。

可以使用 ULConnectionParms 对象提供连接字符串。

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopy 类。

Implements: System.IDisposable

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页
- “ULBulkCopy 构造函数”一节第 58 页
- “ULConnectionParms 类”一节第 177 页
- IDisposable

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULBulkCopy(String, ULBulkCopyOptions) 构造函数

用指定的连接字符串和复制选项初始化 ULBulkCopy 对象。

语法

Visual Basic
```
Public Sub New( _
  ByVal connectionString As String, _
  ByVal copyOptions As ULBulkCopyOptions _
)
```

C#
```
public  ULBulkCopy(
  string  connectionString,
  ULBulkCopyOptions copyOptions
);
```

参数

- **connectionString** 一个字符串，用于定义将打开以供 ULBulkCopy 实例使用的连接。连接字符串是以分号分隔的 "关键字=值" 对的列表。

  有关参数的列表，请参见“ConnectionString 属性”一节第 137 页。

- **copyOptions** ULBulkCopyOptions 枚举中的值组合，用于确定如何将数据源行复制到目标表中。

注释

此语法在 WriteToServer 期间使用 connectionString 打开连接。连接在 WriteToServer 结束时关闭。

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopy 类。

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页
- “ULBulkCopy 构造函数”一节第 58 页
- “ULBulkCopyOptions 枚举”一节第 85 页

ULBulkCopy(ULConnection, ULBulkCopyOptions, ULTransaction) 构造函数

用指定的 ULConnection、复制选项和 ULTransaction 初始化 ULBulkCopy 对象。

语法

Visual Basic
```
Public Sub New( _
  ByVal connection As ULConnection, _
  ByVal copyOptions As ULBulkCopyOptions, _
  ByVal externalTransaction As ULTransaction _
)
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

C#
```
public ULBulkCopy(
  ULConnection connection,
  ULBulkCopyOptions copyOptions,
  ULTransaction externalTransaction
);
```

参数

- **connection** 将用于执行批量复制操作且已打开的 ULConnection。如果连接未打开，则 WriteToServer 中会抛出异常。
- **copyOptions** ULBulkCopyOptions 枚举中的值组合，用于确定如何将数据源行复制到目标表中。
- **externalTransaction** 一个现有的 ULTransaction 实例，批量复制将在该实例下进行。如果 externalTransaction 不为空值引用（在 Visual Basic 中是 Nothing），则批量复制操作将在其内部进行。如果同时指定 externalTransaction 和 ULBulkCopyOptions.UseInternalTransaction 选项，则会发生错误。

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopy 类。

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页
- “ULBulkCopy 构造函数”一节第 58 页
- “ULConnection 类”一节第 131 页
- “ULTransaction 类”一节第 547 页

BatchSize 属性

获取或设置每个批处理中的行数。每个批处理结束时，该批处理中的行将发送到服务器。

语法

Visual Basic
```
Public Property BatchSize As Integer
```

C#
```
public int BatchSize { get; set; }
```

属性值

每个批处理中的行数。缺省值为 0。

注释

将此属性设置为零会使所有行都通过一个批处理进行发送。

将它设置为小于零是错误的。

如果在批处理执行期间更改了此值，当前批处理仍将使用更改前的值完成，但所有后续批处理都将使用新值。

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页

BulkCopyTimeout 属性

获取或设置操作在超时之前完成所需的秒数。

语法

Visual Basic
Public Property **BulkCopyTimeout** As Integer

C#
public int **BulkCopyTimeout** { get; set; }

属性值

缺省值为 30 秒。

注释

值为零表示没有限制。应该避免使用该值，因为这可能会导致无限期的等待。

如果操作超时，将回退当前事务中的所有行，并抛出 SAException。

将它设置为小于零是错误的。

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页

ColumnMappings 属性

返回 ULBulkCopyColumnMapping 项的集合。列映射定义数据源中的列与目标中的列之间的关系。

语法

Visual Basic
Public Readonly Property **ColumnMappings** As ULBulkCopyColumnMappingCollection

C#
public ULBulkCopyColumnMappingCollection **ColumnMappings** { get;}

属性值

缺省情况下它是一个空集合。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

WriteToServer 执行期间无法对该属性进行修改。

如果执行 WriteToServer 时 ColumnMappings 为空，则数据源中的第一列将映射到目标中的第一列，其第二列将映射到目标的第二列，以此类推。这种映射的发生条件是：列类型可以转换、目标的列数至少与数据源的列数一样多且目标的任何额外列都可以为空。

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页
- “ULBulkCopyColumnMapping 类”一节第 69 页

DestinationTableName 属性

获取或设置服务器上目标表的名称。

语法

Visual Basic
Public Property **DestinationTableName** As String

C#
public string **DestinationTableName** { get; set; }

属性值

缺省值为空值引用（在 Visual Basic 中是 Nothing）。

注释

如果在 WriteToServer 执行期间更改了该值，这种更改不会产生任何影响。

如果在调用 WriteToServer 之前未设置该值，将会引发 InvalidOperationException。

将该值设置为空值（在 Visual Basic 中是 Nothing）或空字符串是错误的。

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页

NotifyAfter 属性

指定要在生成通知事件之前处理的行数。

语法

Visual Basic
Public Property **NotifyAfter** As Integer

C#
public int **NotifyAfter** { get; set; }

属性值

一个整数，表示要在生成通知事件之前处理的行数；如果尚未设置该属性，则为零。

注释

WriteToServer 执行期间对 NotifyAfter 进行的更改到下次通知后才会生效。

将它设置为小于零是错误的。

NotifyAfter 与 BulkCopyTimeOut 的值相互排斥，因此，即使未将行发送到数据库或未提交行，也可以触发事件。

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页
- “BulkCopyTimeout 属性”一节第 62 页

Close 方法

关闭 ULBulkCopy 实例。

语法

Visual Basic
Public Sub **Close()**

C#
public void **Close();**

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页

Dispose 方法

终止 ULBulkCopy 实例。

语法

Visual Basic
NotOverridable Public Sub **Dispose()**

C#
public void **Dispose();**

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页

WriteToServer 方法

将所提供的 System.Data.DataRow 对象的数组中的所有行复制到由 ULBulkCopy 对象的 DestinationTableName 字段所指定的目标表中。

WriteToServer(DataRow[]) 方法

将所提供的 System.Data.DataRow 对象的数组中的所有行复制到由 ULBulkCopy 对象的 DestinationTableName 字段所指定的目标表中。

语法

Visual Basic

```
Public Sub WriteToServer( _
  ByVal rows As DataRow() _
)
```

C#

```
public void WriteToServer(
  DataRow[] rows
);
```

参数

- **rows** 将要复制到目标表中的 System.Data.DataRow 对象的数组。

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页
- “WriteToServer 方法”一节第 65 页
- DataRow
- “DestinationTableName 属性”一节第 63 页

WriteToServer(DataTable) 方法

将所提供的 System.Data.DataTable 中的所有行复制到由 ULBulkCopy 对象的 DestinationTableName 所指定的目标表中。

语法

Visual Basic

```
Public Sub WriteToServer( _
  ByVal table As DataTable _
)
```

C#
```
public void WriteToServer(
  DataTable table
);
```

参数

- **table** 一个 System.Data.DataTable，它的行将复制到目标表中。

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页
- “WriteToServer 方法”一节第 65 页
- “DestinationTableName 属性”一节第 63 页
- DataTable

WriteToServer(IDataReader) 方法

将所提供的 System.Data.IDataReader 中的所有行复制到由 ULBulkCopy 对象的 DestinationTableName 所指定的目标表中。

语法

Visual Basic
```
Public Sub WriteToServer( _
  ByVal reader As IDataReader _
)
```

C#
```
public void WriteToServer(
  IDataReader reader
);
```

参数

- **reader** 一个 System.Data.IDataReader，它的行将复制到目标表中。

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页
- “WriteToServer 方法”一节第 65 页
- IDataReader
- “DestinationTableName 属性”一节第 63 页

WriteToServer(DataTable, DataRowState) 方法

将所提供的 System.Data.DataTable 中具有指定行状态的所有行复制到由 ULBulkCopy 对象的 DestinationTableName 所指定的目标表中。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
```
Public Sub WriteToServer( _
  ByVal table As DataTable, _
  ByVal rowState As DataRowState _
)
```

C#
```
public void WriteToServer(
  DataTable table,
  DataRowState rowState
);
```

参数

- **table** 一个 System.Data.DataTable，它的行将复制到目标表中。
- **rowState** System.Data.DataRowState 枚举中的一个值。只有与行状态匹配的行才会复制到目标表中。

注释

如果指定了 rowState，则仅复制那些具有相同行状态的行。

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页
- “WriteToServer 方法”一节第 65 页
- “DestinationTableName 属性”一节第 63 页
- DataTable
- DataRowState

ULRowsCopied 事件

每次处理完 NotifyAfter 所指定的行数时便会发生此事件。

语法

Visual Basic
```
Public Event ULRowsCopied As ULRowsCopiedEventHandler
```

C#
```
public event ULRowsCopiedEventHandler ULRowsCopied;
```

注释

接收到 ULRowsCopied 事件并不意味着已提交了任何行。无法通过此事件调用 Close 方法。

另请参见

- “ULBulkCopy 类”一节第 57 页
- “ULBulkCopy 成员”一节第 57 页
- “NotifyAfter 属性”一节第 63 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULBulkCopyColumnMapping 类

定义 ULBulkCopy 实例的数据源中的列与该实例的目标表中的列之间的映射。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULBulkCopyColumnMapping**

C#
public sealed class **ULBulkCopyColumnMapping**

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopyColumnMapping 类。

另请参见

- “ULBulkCopyColumnMapping 成员”一节第 69 页
- “ULBulkCopy 类”一节第 57 页

ULBulkCopyColumnMapping 成员

公共构造函数

成员名称	说明
“ULBulkCopyColumnMapping 构造函数”一节第 70 页	初始化一个新的“ULBulkCopyColumnMapping 类”一节第 69 页实例。

公共属性

成员名称	说明
“DestinationColumn 属性”一节第 73 页	指定目标数据库表中要映射到的列的名称。
“DestinationOrdinal 属性”一节第 73 页	指定目标数据库表中要映射到的列的顺序值。
“SourceColumn 属性”一节第 74 页	指定数据源中要映射的列的名称。
“SourceOrdinal 属性”一节第 75 页	指定源列在数据源内的顺序位置。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULBulkCopyColumnMapping 类”一节第 69 页
- “ULBulkCopy 类”一节第 57 页

ULBulkCopyColumnMapping 构造函数

初始化一个新的“ULBulkCopyColumnMapping 类”一节第 69 页实例。

ULBulkCopyColumnMapping() 构造函数

创建新的列映射。

语法

Visual Basic
```
Public Sub New()
```

C#
```
public  ULBulkCopyColumnMapping();
```

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopyColumnMapping 类。

另请参见

- “ULBulkCopyColumnMapping 类”一节第 69 页
- “ULBulkCopyColumnMapping 成员”一节第 69 页
- “ULBulkCopyColumnMapping 构造函数”一节第 70 页

ULBulkCopyColumnMapping(Int32, Int32) 构造函数

通过使用列顺序号或列名引用源列和目标列来创建新的列映射。

语法

Visual Basic
```
Public Sub New( _
  ByVal sourceColumnOrdinal As Integer, _
  ByVal destinationColumnOrdinal As Integer _
)
```

C#
```
public  ULBulkCopyColumnMapping(
  int sourceColumnOrdinal,
  int destinationColumnOrdinal
);
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

参数

- **sourceColumnOrdinal** 源列在数据源内的顺序位置。数据源中第一列的顺序位置为零。
- **destinationColumnOrdinal** 目标列在目标表内的顺序位置。表中第一列的顺序位置为零。

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopyColumnMapping 类。

另请参见

- “ULBulkCopyColumnMapping 类”一节第 69 页
- “ULBulkCopyColumnMapping 成员”一节第 69 页
- “ULBulkCopyColumnMapping 构造函数”一节第 70 页

ULBulkCopyColumnMapping(Int32, String) 构造函数

通过使用列顺序号引用源列以及使用列名引用目标列来创建新的列映射。

语法

Visual Basic

```
Public Sub New( _
  ByVal sourceColumnOrdinal As Integer, _
  ByVal destinationColumn As String _
)
```

C#

```
public  ULBulkCopyColumnMapping(
  int sourceColumnOrdinal,
  string  destinationColumn
);
```

参数

- **sourceColumnOrdinal** 源列在数据源内的顺序位置。数据源中第一列的顺序位置为零。
- **destinationColumn** 目标列在目标表中的名称。

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopyColumnMapping 类。

另请参见

- “ULBulkCopyColumnMapping 类”一节第 69 页
- “ULBulkCopyColumnMapping 成员”一节第 69 页
- “ULBulkCopyColumnMapping 构造函数”一节第 70 页

ULBulkCopyColumnMapping(String, Int32) 构造函数

通过使用列名引用源列以及使用列顺序号引用目标列来创建新的列映射。

语法

Visual Basic

```
Public Sub New( _
  ByVal sourceColumn As String, _
  ByVal destinationColumnOrdinal As Integer _
)
```

C#

```
public ULBulkCopyColumnMapping(
  string sourceColumn,
  int destinationColumnOrdinal
);
```

参数

- **sourceColumn** 源列在数据源中的名称。
- **destinationColumnOrdinal** 目标列在目标表内的顺序位置。表中第一列的顺序位置为零。

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopyColumnMapping 类。

另请参见

- “ULBulkCopyColumnMapping 类”一节第 69 页
- “ULBulkCopyColumnMapping 成员”一节第 69 页
- “ULBulkCopyColumnMapping 构造函数”一节第 70 页

ULBulkCopyColumnMapping(String, String) 构造函数

通过使用列名引用源列和目标列来创建新的列映射。

语法

Visual Basic

```
Public Sub New( _
  ByVal sourceColumn As String, _
  ByVal destinationColumn As String _
)
```

C#

```
public ULBulkCopyColumnMapping(
  string sourceColumn,
  string destinationColumn
);
```

参数

- **sourceColumn** 源列在数据源内的名称。
- **destinationColumn** 目标列在目标表内的名称。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopyColumnMapping 类。

另请参见

- “ULBulkCopyColumnMapping 类”一节第 69 页
- “ULBulkCopyColumnMapping 成员”一节第 69 页
- “ULBulkCopyColumnMapping 构造函数”一节第 70 页

DestinationColumn 属性

指定目标数据库表中要映射到的列的名称。

语法

Visual Basic
Public Property **DestinationColumn** As String

C#
public string **DestinationColumn** { get; set; }

属性值

指定目标表中列名称的字符串；如果 DestinationOrdinal 有优先级，则该值为空值引用（在 Visual Basic 中是 Nothing）。

注释

DestinationColumn 与 DestinationOrdinal 属性相互排斥。设置时间较近者优先。

设置 DestinationColumn 属性会使 DestinationOrdinal 属性被设置为 -1；设置 DestinationOrdinal 属性会使 DestinationColumn 属性被设置为空值引用（在 Visual Basic 中是 Nothing）。

将 DestinationColumn 设置为空值或空字符串是错误的。

另请参见

- “ULBulkCopyColumnMapping 类”一节第 69 页
- “ULBulkCopyColumnMapping 成员”一节第 69 页
- “DestinationOrdinal 属性”一节第 73 页
- “DestinationOrdinal 属性”一节第 73 页

DestinationOrdinal 属性

指定目标数据库表中要映射到的列的顺序值。

语法

Visual Basic
Public Property **DestinationOrdinal** As Integer

C#
public int **DestinationOrdinal** { get; set; }

属性值

一个整数，指定目标表中要映射到的列的顺序号，未设置该属性时值为 -1。

注释

DestinationColumn 与 DestinationOrdinal 属性相互排斥。设置时间较近者优先。

设置 DestinationColumn 属性会使 DestinationOrdinal 属性被设置为 -1；设置 DestinationOrdinal 属性会使 DestinationColumn 属性被设置为空值引用（在 Visual Basic 中是 Nothing）。

另请参见

- “ULBulkCopyColumnMapping 类”一节第 69 页
- “ULBulkCopyColumnMapping 成员”一节第 69 页
- “DestinationColumn 属性”一节第 73 页
- “DestinationColumn 属性”一节第 73 页

SourceColumn 属性

指定数据源中要映射的列的名称。

语法

Visual Basic
Public Property **SourceColumn** As String

C#
public string **SourceColumn** { get; set; }

属性值

指定数据源中列名称的字符串；如果 SourceOrdinal 有优先级，则该值为空值引用（在 Visual Basic 中是 Nothing）。

注释

SourceColumn 与 SourceOrdinal 属性相互排斥。最近设置的值优先。

设置 SourceColumn 属性会使 SourceOrdinal 属性被设置为 -1；设置 SourceOrdinal 属性会使 SourceColumn 属性被设置为空值引用（在 Visual Basic 中是 Nothing）。

将 SourceColumn 设置为空值或空字符串是错误的。

另请参见

- “ULBulkCopyColumnMapping 类”一节第 69 页
- “ULBulkCopyColumnMapping 成员”一节第 69 页
- “SourceOrdinal 属性”一节第 75 页
- “SourceOrdinal 属性”一节第 75 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

SourceOrdinal 属性

指定源列在数据源内的顺序位置。

语法

Visual Basic
Public Property **SourceOrdinal** As Integer

C#
public int **SourceOrdinal** { get; set; }

属性值

一个整数，指定列在数据源中的顺序号，未设置该属性时值为 -1。

注释

SourceColumn 与 SourceOrdinal 属性相互排斥。最近设置的值优先。

设置 SourceColumn 属性会使 SourceOrdinal 属性被设置为 -1；设置 SourceOrdinal 属性会使 SourceColumn 属性被设置为空值引用（在 Visual Basic 中是 Nothing）。

另请参见

- “ULBulkCopyColumnMapping 类”一节第 69 页
- “ULBulkCopyColumnMapping 成员”一节第 69 页
- “SourceColumn 属性”一节第 74 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULBulkCopyColumnMappingCollection 类

从 System.Collections.CollectionBase 继承的 ULBulkCopyColumnMapping 对象的集合。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULBulkCopyColumnMappingCollection**
  Inherits CollectionBase

C#
public sealed class **ULBulkCopyColumnMappingCollection**: CollectionBase

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopyColumnMappingCollection 类。

另请参见

- “ULBulkCopyColumnMappingCollection 成员”一节第 76 页
- “ULBulkCopyColumnMapping 类”一节第 69 页

ULBulkCopyColumnMappingCollection 成员

公共属性

成员名称	说明
Capacity（继承自 CollectionBase）	获取或设置 CollectionBase 可以包含的元素数量。
Count（继承自 CollectionBase）	获取包含在 CollectionBase 实例中的元素的数量。此属性不能被覆盖。
“Item 属性”一节第 77 页	获取指定索引处的 ULBulkCopyColumnMapping 对象。

公共方法

成员名称	说明
“Add 方法”一节第 78 页	将指定的 ULBulkCopyColumnMapping 添加到集合中。
Clear（继承自 CollectionBase）	删除 CollectionBase 实例的所有对象。此属性不能被覆盖。
“Contains 方法”一节第 81 页	返回集合中是否存在指定的 ULBulkCopyColumnMapping 对象。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“CopyTo 方法”一节 第 82 页	将 ULBulkCopyColumnMappingCollection 的元素复制到 ULBulkCopyColumnMapping 项的数组中（从特定索引处开始）。
GetEnumerator（继承自 CollectionBase）	返回迭代通过 CollectionBase 实例的枚举器。
“IndexOf 方法”一节 第 82 页	返回集合内指定 ULBulkCopyColumnMapping 的索引。
“Remove 方法”一节 第 83 页	从 ULBulkCopyColumnMappingCollection 中删除指定的 ULBulkCopyColumnMapping 元素。
“RemoveAt 方法”一节 第 84 页	从集合中删除指定索引处的映射。

另请参见

- “ULBulkCopyColumnMappingCollection 类”一节第 76 页
- “ULBulkCopyColumnMapping 类”一节第 69 页

Item 属性

获取指定索引处的 ULBulkCopyColumnMapping 对象。

语法

Visual Basic

```
Public Readonly Property Item( _
   ByVal index As Integer _
) As ULBulkCopyColumnMapping
```

C#

```
public ULBulkCopyColumnMapping this[
   int index
] { get;}
```

参数

- **index** 要查找的 ULBulkCopyColumnMapping 对象的索引（从零开始）。

属性值

返回 ULBulkCopyColumnMapping 对象。

另请参见

- “ULBulkCopyColumnMappingCollection 类”一节第 76 页
- “ULBulkCopyColumnMappingCollection 成员”一节第 76 页
- “ULBulkCopyColumnMapping 类”一节第 69 页

Add 方法

将指定的 ULBulkCopyColumnMapping 添加到集合中。

Add(ULBulkCopyColumnMapping) 方法

将指定的 ULBulkCopyColumnMapping 添加到集合中。

语法

Visual Basic

```
Public Function Add( _
  ByVal bulkCopyColumnMapping As ULBulkCopyColumnMapping _
) As ULBulkCopyColumnMapping
```

C#

```
public ULBulkCopyColumnMapping Add(
  ULBulkCopyColumnMapping bulkCopyColumnMapping
);
```

参数

- **bulkCopyColumnMapping** 对将要添加到集合中的映射进行说明的 ULBulkCopyColumnMapping 对象。

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopyColumnMappingCollection 类。

另请参见

- “ULBulkCopyColumnMappingCollection 类”一节第 76 页
- “ULBulkCopyColumnMappingCollection 成员”一节第 76 页
- “Add 方法”一节第 78 页
- “ULBulkCopyColumnMapping 类”一节第 69 页

Add(Int32, Int32) 方法

通过使用顺序号指定源列和目标列来创建新的 ULBulkCopyColumnMapping 实例，并将该映射添加到集合中。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic

```
Public Function Add( _
  ByVal sourceColumnOrdinal As Integer, _
  ByVal destinationColumnOrdinal As Integer _
) As ULBulkCopyColumnMapping
```

C#

```
public ULBulkCopyColumnMapping Add(
  int sourceColumnOrdinal,
  int destinationColumnOrdinal
);
```

参数

- **sourceColumnOrdinal** 源列在数据源内的顺序位置。数据源中第一列的顺序位置为零。
- **destinationColumnOrdinal** 目标列在目标表内的顺序位置。表中第一列的顺序位置为零。

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopyColumnMappingCollection 类。

另请参见

- “ULBulkCopyColumnMappingCollection 类”一节第 76 页
- “ULBulkCopyColumnMappingCollection 成员”一节第 76 页
- “Add 方法”一节第 78 页
- “ULBulkCopyColumnMapping 类”一节第 69 页

Add(Int32, String) 方法

通过使用列顺序号引用源列和使用列名引用目标列来创建新的 ULBulkCopyColumnMapping，并将该映射添加到集合中。

语法

Visual Basic

```
Public Function Add( _
  ByVal sourceColumnOrdinal As Integer, _
  ByVal destinationColumn As String _
) As ULBulkCopyColumnMapping
```

C#

```
public ULBulkCopyColumnMapping Add(
  int sourceColumnOrdinal,
  string  destinationColumn
);
```

参数

- **sourceColumnOrdinal** 源列在数据源内的顺序位置。数据源中第一列的顺序位置为零。
- **destinationColumn** 目标列在目标表内的名称。

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopyColumnMappingCollection 类。

另请参见

- “ULBulkCopyColumnMappingCollection 类”一节第 76 页
- “ULBulkCopyColumnMappingCollection 成员”一节第 76 页
- “Add 方法”一节第 78 页
- “ULBulkCopyColumnMapping 类”一节第 69 页

Add(String, Int32) 方法

通过使用列名引用源列和使用列顺序号引用目标列来创建新的 ULBulkCopyColumnMapping，并将该映射添加到集合中。

通过使用列顺序号或列名引用源列和目标列来创建新的列映射。

语法

Visual Basic

```
Public Function Add( _
   ByVal sourceColumn As String, _
   ByVal destinationColumnOrdinal As Integer _
) As ULBulkCopyColumnMapping
```

C#

```
public ULBulkCopyColumnMapping Add(
   string  sourceColumn,
   int destinationColumnOrdinal
);
```

参数

- **sourceColumn** 源列在数据源内的名称。
- **destinationColumnOrdinal** 目标列在目标表内的顺序位置。表中第一列的顺序位置为零。

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopyColumnMappingCollection 类。

另请参见

- “ULBulkCopyColumnMappingCollection 类”一节第 76 页
- “ULBulkCopyColumnMappingCollection 成员”一节第 76 页
- “Add 方法”一节第 78 页
- “ULBulkCopyColumnMapping 类”一节第 69 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

Add(String, String) 方法

通过使用列名指定源列和目标列来创建新的 ULBulkCopyColumnMapping，并将该映射添加到集合中。

语法

Visual Basic
```
Public Function Add( _
  ByVal sourceColumn As String, _
  ByVal destinationColumn As String _
) As ULBulkCopyColumnMapping
```

C#
```
public ULBulkCopyColumnMapping Add(
  string  sourceColumn,
  string  destinationColumn
);
```

参数

- **sourceColumn** 源列在数据源内的名称。
- **destinationColumn** 目标列在目标表内的名称。

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopyColumnMappingCollection 类。

另请参见

- “ULBulkCopyColumnMappingCollection 类”一节第 76 页
- “ULBulkCopyColumnMappingCollection 成员”一节第 76 页
- “Add 方法”一节第 78 页
- “ULBulkCopyColumnMapping 类”一节第 69 页

Contains 方法

返回集合中是否存在指定的 ULBulkCopyColumnMapping 对象。

语法

Visual Basic
```
Public Function Contains( _
  ByVal value As ULBulkCopyColumnMapping _
) As Boolean
```

C#
```
public bool Contains(
  ULBulkCopyColumnMapping value
);
```

参数

- **value** 有效的 ULBulkCopyColumnMapping 对象。

返回值

如果集合中存在指定的映射，则为 true；否则为 false。

另请参见

- “ULBulkCopyColumnMappingCollection 类”一节第 76 页
- “ULBulkCopyColumnMappingCollection 成员”一节第 76 页
- “ULBulkCopyColumnMapping 类”一节第 69 页

CopyTo 方法

将 ULBulkCopyColumnMappingCollection 的元素复制到 ULBulkCopyColumnMapping 项的数组中（从特定索引处开始）。

语法

Visual Basic

```
Public Sub CopyTo( _
  ByVal array As ULBulkCopyColumnMapping(), _
  ByVal index As Integer _
)
```

C#

```
public void CopyTo(
  ULBulkCopyColumnMapping[] array,
  int index
);
```

参数

- **array** 一维 ULBulkCopyColumnMapping 数组，从 ULBulkCopyColumnMappingCollection 中复制的元素以其为目标。该数组必须具有从零开始的索引。
- **index** 数组中复制开始时所在的从零开始索引。

另请参见

- “ULBulkCopyColumnMappingCollection 类”一节第 76 页
- “ULBulkCopyColumnMappingCollection 成员”一节第 76 页
- “ULBulkCopyColumnMapping 类”一节第 69 页

IndexOf 方法

返回集合内指定 ULBulkCopyColumnMapping 的索引。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic

```
Public Function IndexOf( _
  ByVal value As ULBulkCopyColumnMapping _
) As Integer
```

C#

```
public int IndexOf(
  ULBulkCopyColumnMapping value
);
```

参数

- **value** 要搜索的 ULBulkCopyColumnMapping 对象。

返回值

返回列映射的从零开始索引；如果未在集合中找到列映射，则返回 -1。

另请参见

- “ULBulkCopyColumnMappingCollection 类”一节第 76 页
- “ULBulkCopyColumnMappingCollection 成员”一节第 76 页
- “ULBulkCopyColumnMapping 类”一节第 69 页

Remove 方法

从 ULBulkCopyColumnMappingCollection 中删除指定的 ULBulkCopyColumnMapping 元素。

语法

Visual Basic

```
Public Sub Remove( _
  ByVal value As ULBulkCopyColumnMapping _
)
```

C#

```
public void Remove(
  ULBulkCopyColumnMapping value
);
```

参数

- **value** 要从集合中删除的 ULBulkCopyColumnMapping 对象。

另请参见

- “ULBulkCopyColumnMappingCollection 类”一节第 76 页
- “ULBulkCopyColumnMappingCollection 成员”一节第 76 页
- “ULBulkCopyColumnMapping 类”一节第 69 页

RemoveAt 方法

从集合中删除指定索引处的映射。

语法

Visual Basic

```
Public Sub RemoveAt( _
  ByVal index As Integer _
)
```

C#

```
public void RemoveAt(
  int index
);
```

参数

- **index** 要从集合中删除的 ULBulkCopyColumnMapping 对象的从零开始索引。

另请参见

- “ULBulkCopyColumnMappingCollection 类”一节第 76 页
- “ULBulkCopyColumnMappingCollection 成员”一节第 76 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULBulkCopyOptions 枚举

用于指定要与 ULBulkCopy 类实例配合使用的一个或多个选项的逐位标志。

语法

Visual Basic
Public Enum **ULBulkCopyOptions**

C#
public enum **ULBulkCopyOptions**

注释

构造 ULBulkCopy 实例时，可以使用 ULBulkCopyOptions 枚举来指定 WriteToServer 方法的行为方式。

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULBulkCopyOptions 类。

成员

成员名称	说明	值
缺省值	仅指定此值会造成使用缺省行为。	0
KeepIdentity	指定此值时将保留要复制到标识列中的源值。缺省情况下，将在目标表中生成新标识值。	1
UseInternalTransaction	指定此值时每个批量复制操作批处理都在事务内执行。未指定此值时，将不使用事务。如果指定此选项，同时为构造函数提供了 ULTransaction 对象，则会发生 System.ArgumentException。	2

另请参见

- “ULBulkCopy 类”一节第 57 页

ULCommand 类

表示具有或不具有 IN 参数的预编译 SQL 语句或查询。此对象可用于多次有效地执行语句或查询。此类无法继承。

语法

Visual Basic

```
Public NotInheritable Class ULCommand
  Inherits DbCommand
  Implements ICloneable
```

C#

```
public sealed class ULCommand: DbCommand,
  ICloneable
```

注释

可以直接创建 ULCommand 对象，或使用 ULConnection.CreateCommand 方法创建。此方法可以确保命令在给定的连接上具有执行语句所需的正确事务。

在提交或回退当前事务后，必须重置 ULCommand.Transaction 方法。

ULCommand 使用下列方法在 UltraLite.NET 数据库中执行命令：

方法	说明
ULCommand.ExecuteNonQuery	执行一条不返回结果集的语句，如 SQL INSERT、DELETE 或 UPDATE 语句。
ULCommand.ExecuteReader()	执行 SQL SELECT 语句并在 ULDataReader 中返回结果集。使用此方法可创建只读结果集。
ULCommand.ExecuteResultSet()	**UL Ext.:** 执行 SQL SELECT 语句并在 ULResultSet 中返回结果集。使用此方法可创建可变结果集。
ULCommand.ExecuteScalar	执行 SQL SELECT 语句并返回单个值。
ULCommand.ExecuteTable()	**UL Ext.:** 在 ULTable 中检索数据库表以便直接进行操作。ULCommand.CommandText 被解释为表的名称，而 ULCommand.IndexName 可用于指定表的排序顺序。ULCommand.CommandType 必须为 System.Data.CommandType.TableDirect。

您可重置大多数属性（包括 ULCommand.CommandText）并重新使用 ULCommand 对象。

出于资源管理方面的原因，建议在使用完命令后显式终止这些命令。在 C# 中，您可以使用 using 语句来自动调用 System.ComponentModel.Component.Dispose() 方法或显式调用 System.ComponentModel.Component.Dispose() 方法。在 Visual Basic 中，您将始终显式调用 System.ComponentModel.Component.Dispose() 方法。

Inherits: System.Data.Common.DbCommand

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

Implements: System.Data.IDbCommand、System.IDisposable

另请参见

- “ULCommand 成员”一节第 87 页
- “CreateCommand 方法”一节第 151 页
- “Transaction 属性”一节第 98 页
- “ExecuteNonQuery 方法”一节第 111 页
- “ExecuteReader() 方法”一节第 112 页
- “ExecuteResultSet() 方法”一节第 115 页
- “ExecuteScalar 方法”一节第 117 页
- “ExecuteTable() 方法”一节第 118 页
- “ULTable 类”一节第 511 页
- “CommandText 属性”一节第 93 页
- CommandType.TableDirect
- Component.Dispose
- “CommandType 属性”一节第 94 页
- “IndexName 属性”一节第 96 页
- DbCommand
- IDbCommand
- IDisposable
- “ULResultSet 类”一节第 394 页
- “ULDataReader 类”一节第 257 页

ULCommand 成员

公共构造函数

成员名称	说明
“ULCommand 构造函数”一节第 89 页	初始化一个新的“ULCommand 类”一节第 86 页实例。

公共属性

成员名称	说明
“CommandText 属性”一节第 93 页	指定 SQL 语句的文本或表的名称（此时 ULCommand.CommandType 为 System.Data.CommandType.TableDirect）。如果是参数化语句，请使用问号 (?) 占位符来传递参数。
“CommandTimeout 属性”一节第 94 页	UltraLite.NET 不支持此功能。
“CommandType 属性”一节第 94 页	指定要执行的命令的类型。

成员名称	说明
“Connection 属性”一节第 95 页	在其上执行 ULCommand 对象的连接对象。
“DesignTimeVisible 属性”一节第 96 页	指示 ULCommand 是否应在自定义的 [Windows 窗体设计器] 控件中可见。
“IndexName 属性”一节第 96 页	**UL Ext.:** 指定当 ULCommand.CommandType 为 System.Data.CommandType.TableDirect 时打开（排序）表所使用的索引的名称。
“Parameters 属性”一节第 97 页	指定当前语句的参数。
“Plan 属性”一节第 97 页	**UL Ext.:** 返回 UltraLite.NET 用于执行查询的访问计划。此属性主要供开发期间使用。
“Transaction 属性”一节第 98 页	指定 ULCommand 执行时所处的 ULTransaction。
“UpdatedRowSource 属性”一节第 99 页	指定当 ULDataAdapter 的 Update 方法使用命令结果时，如何将命令结果应用于 DataRow。

公共方法

成员名称	说明
“BeginExecuteNonQuery 方法”一节第 99 页	在已知回调过程和状态信息的情况下，启动由此 ULCommand 说明的 SQL 语句的异步执行。
“BeginExecuteReader 方法”一节第 100 页	启动由此 ULCommand 说明的 SQL 语句的异步执行，并检索结果集。
“Cancel 方法”一节第 104 页	UltraLite.NET 不支持此方法。
“CreateParameter 方法”一节第 104 页	提供 ULParameter 对象来为 ULCommand 对象提供参数。
“EndExecuteNonQuery 方法”一节第 105 页	完成 SQL 语句的异步执行。
“EndExecuteReader 方法”一节第 108 页	完成 SQL 语句的异步执行，并返回所请求的 ULDataReader。
“ExecuteNonQuery 方法”一节第 111 页	执行一条不返回结果集的语句，如 SQL INSERT、DELETE 或 UPDATE 语句。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“ExecuteReader 方法”一节第 112 页	执行 SQL SELECT 语句并返回结果集。
“ExecuteResultSet 方法”一节第 114 页	**UL Ext.:** 执行 SQL SELECT 语句并以 ULResultSet 形式返回结果集。
“ExecuteScalar 方法”一节第 117 页	执行 SQL SELECT 语句并返回单个值。
“ExecuteTable 方法”一节第 118 页	**UL Ext.:** 在 ULTable 中检索数据库表以便直接进行操作。ULCommand.CommandText 被解释为表的名称，而 ULCommand.IndexName 可用于指定表的排序顺序。
“Prepare 方法”一节第 120 页	预编译并存储此命令的 SQL 语句。

另请参见

- “ULCommand 类”一节第 86 页
- “CreateCommand 方法”一节第 151 页
- “Transaction 属性”一节第 98 页
- “ExecuteNonQuery 方法”一节第 111 页
- “ExecuteReader() 方法”一节第 112 页
- “ExecuteResultSet() 方法”一节第 115 页
- “ExecuteScalar 方法”一节第 117 页
- “ExecuteTable() 方法”一节第 118 页
- “ULTable 类”一节第 511 页
- “CommandText 属性”一节第 93 页
- CommandType.TableDirect
- Component.Dispose
- “CommandType 属性”一节第 94 页
- “IndexName 属性”一节第 96 页
- DbCommand
- IDbCommand
- IDisposable
- “ULResultSet 类”一节第 394 页
- “ULDataReader 类”一节第 257 页

ULCommand 构造函数

初始化一个新的“ULCommand 类”一节第 86 页实例。

ULCommand() 构造函数

初始化 ULCommand 对象。

语法

Visual Basic
Public Sub **New()**

C#
public **ULCommand();**

注释

执行语句前需要先设置 ULCommand 对象的 ULCommand.CommandText、ULCommand.Connection 和 ULCommand.Transaction 属性。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ULCommand 构造函数”一节第 89 页
- “CreateCommand 方法”一节第 151 页
- “ULCommand(String) 构造函数”一节第 90 页
- “ULCommand(String, ULConnection) 构造函数”一节第 91 页
- “ULCommand(String, ULConnection, ULTransaction) 构造函数”一节第 92 页
- “CommandText 属性”一节第 93 页
- “Connection 属性”一节第 95 页
- “Transaction 属性”一节第 98 页

ULCommand(String) 构造函数

用指定的命令文本初始化 ULCommand 对象。

语法

Visual Basic
```
Public Sub New( _
  ByVal cmdText As String _
)
```

C#
```
public ULCommand(
  string cmdText
);
```

参数

- **cmdText** SQL 语句的文本，或者当 ULCommand.CommandType 为 System.Data.CommandType.TableDirect 时，该参数值为表的名称。如果是参数化语句，请使用问号 (?) 占位符来传递参数。

注释

执行语句前需要先设置 ULCommand 对象的 ULCommand.Connection 和 ULCommand.Transaction。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ULCommand 构造函数”一节第 89 页
- “CreateCommand 方法”一节第 151 页
- “ULCommand() 构造函数”一节第 89 页
- “ULCommand(String, ULConnection) 构造函数”一节第 91 页
- “ULCommand(String, ULConnection, ULTransaction) 构造函数”一节第 92 页
- “Connection 属性”一节第 95 页
- “Transaction 属性”一节第 98 页
- “CommandType 属性”一节第 94 页
- CommandType.TableDirect

ULCommand(String, ULConnection) 构造函数

用指定的命令文本和连接初始化 ULCommand 对象。

语法

Visual Basic

```
Public Sub New( _
  ByVal cmdText As String, _
  ByVal connection As ULConnection _
)
```

C#

```
public  ULCommand(
  string  cmdText,
  ULConnection connection
);
```

参数

- **cmdText**　SQL 语句的文本，或者当 ULCommand.CommandType 为 System.Data.CommandType.TableDirect 时，该参数值为表的名称。如果是参数化语句，请使用问号 (?) 占位符来传递参数。
- **connection**　表示当前连接的 ULConnection 对象。

注释

执行语句前可能需要先设置 ULCommand 对象的 ULCommand.Transaction。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ULCommand 构造函数”一节第 89 页
- “CreateCommand 方法”一节第 151 页
- “ULCommand() 构造函数”一节第 89 页
- “ULCommand(String) 构造函数”一节第 90 页
- “ULCommand(String, ULConnection, ULTransaction) 构造函数”一节第 92 页
- “Transaction 属性”一节第 98 页
- “CommandType 属性”一节第 94 页
- CommandType.TableDirect
- “ULConnection 类”一节第 131 页

ULCommand(String, ULConnection, ULTransaction) 构造函数

用指定的命令文本、连接和事务初始化 ULCommand 对象。

语法

Visual Basic

```
Public Sub New( _
  ByVal cmdText As String, _
  ByVal connection As ULConnection, _
  ByVal transaction As ULTransaction _
)
```

C#

```
public  ULCommand(
  string  cmdText,
  ULConnection connection,
  ULTransaction transaction
);
```

参数

- **cmdText**　SQL 语句的文本，或者当 ULCommand.CommandType 为 System.Data.CommandType.TableDirect 时，该参数值为表的名称。如果是参数化语句，请使用问号 (?) 占位符来传递参数。
- **connection**　表示当前连接的 ULConnection 对象。
- **事务**　ULCommand 执行时所在的 ULTransaction。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ULCommand 构造函数”一节第 89 页
- “CreateCommand 方法”一节第 151 页
- “ULCommand() 构造函数”一节第 89 页
- “ULCommand(String) 构造函数”一节第 90 页
- “ULCommand(String, ULConnection) 构造函数”一节第 91 页
- “CommandType 属性”一节第 94 页
- CommandType.TableDirect
- “ULTransaction 类”一节第 547 页

CommandText 属性

指定 SQL 语句的文本或表的名称（此时 ULCommand.CommandType 为 System.Data.CommandType.TableDirect）。如果是参数化语句，请使用问号 (?) 占位符来传递参数。

语法

Visual Basic
Public Overrides Property **CommandText** As String

C#
public override string **CommandText** { get; set; }

属性值

指定 SQL 语句的文本或表的名称的字符串。缺省值为空字符串（无效命令）。

注释

出于性能原因，SELECT 语句在缺省情况下标记为只读。如果要将该查询用于执行更新，则此语句必须以 "FOR UPDATE" 作为结尾。

示例

以下示例演示了如何使用参数化占位符：

```
' Visual Basic
myCmd.CommandText = "SELECT * FROM Customers WHERE CustomerID = ?"

// C#
myCmd.CommandText = "SELECT * FROM Customers WHERE CustomerID = ?";
```

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ExecuteNonQuery 方法”一节第 111 页
- “ExecuteReader() 方法”一节第 112 页
- “ExecuteResultSet() 方法”一节第 115 页
- “ExecuteScalar 方法”一节第 117 页
- “ExecuteTable() 方法”一节第 118 页
- “CommandType 属性”一节第 94 页
- CommandType.TableDirect

CommandTimeout 属性

UltraLite.NET 不支持此功能。

语法

Visual Basic
Public Overrides Property **CommandTimeout** As Integer

C#
public override int **CommandTimeout** { get; set; }

属性值

值始终为 0。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页

CommandType 属性

指定要执行的命令的类型。

语法

Visual Basic
Public Overrides Property **CommandType** As CommandType

C#
public override CommandType **CommandType** { get; set; }

属性值

System.Data.CommandType 值之一。缺省值为 System.Data.CommandType.Text。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

所支持的命令类型如下：

- System.Data.CommandType.TableDirect - **UL Ext.:** 指定此 CommandType 时，ULCommand.CommandText 必须是数据库表的名称。您还可以使用 ULCommand.IndexName 指定用于打开（排序）表的索引。使用 ULCommand.ExecuteTable() 或 ULCommand.ExecuteReader() 访问表。
- System.Data.CommandType.Text - 指定此 CommandType 时，ULCommand.CommandText 必须是 SQL 语句或查询。使用 ULCommand.ExecuteNonQuery 执行非查询 SQL 语句并使用 ULCommand.ExecuteReader() 或 ULCommand.ExecuteScalar 执行查询。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- CommandType
- CommandType.Text
- CommandType.TableDirect
- “CommandText 属性”一节第 93 页
- “IndexName 属性”一节第 96 页
- “ExecuteTable() 方法”一节第 118 页
- “ExecuteReader() 方法”一节第 112 页
- CommandType.Text
- “CommandText 属性”一节第 93 页
- “ExecuteNonQuery 方法”一节第 111 页
- “ExecuteReader() 方法”一节第 112 页
- “ExecuteScalar 方法”一节第 117 页

Connection 属性

在其上执行 ULCommand 对象的连接对象。

语法

Visual Basic
Public Property **Connection** As ULConnection

C#
public ULConnection **Connection** { get; set; }

属性值

在其上执行命令的 ULConnection 对象。

注释

执行 ULCommand 对象前必须先打开一个连接。

缺省值为空值引用（在 Visual Basic 中是 Nothing）。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

这是 System.Data.IDbCommand.Connection 和 System.Data.Common.DbCommand.Connection 的强类型版本。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ULConnection 类”一节第 131 页
- IDbCommand.Connection
- DbCommand.Connection

DesignTimeVisible 属性

指示 ULCommand 是否应在自定义的 [Windows 窗体设计器] 控件中可见。

语法

Visual Basic
Public Overrides Property **DesignTimeVisible** As Boolean

C#
public override bool **DesignTimeVisible** { get; set; }

属性值

如果应使此 ULCommand 实例可见，则为 true；否则为 false。缺省值为 false。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页

IndexName 属性

UL Ext.: 指定当 ULCommand.CommandType 为 System.Data.CommandType.TableDirect 时打开（排序）表所使用的索引的名称。

语法

Visual Basic
Public Property **IndexName** As String

C#
public string **IndexName** { get; set; }

属性值

用于指定索引名称的字符串。缺省值为空值引用（在 Visual Basic 中是 Nothing），表示用表的主键打开表。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ExecuteTable() 方法”一节第 118 页
- “ExecuteReader() 方法”一节第 112 页
- “CommandType 属性”一节第 94 页
- CommandType.TableDirect

Parameters 属性

指定当前语句的参数。

语法

Visual Basic
Public Readonly Property **Parameters** As ULParameterCollection

C#
public ULParameterCollection **Parameters** { get;}

属性值

保存 SQL 语句参数的 ULParameterCollection。缺省值为空集合。

注释

在 ULCommand.CommandText 中使用问号表示参数。集合中参数的指定顺序与问号占位符相同。例如，集合中的第一个参数对应 SQL 语句中的第一个问号，集合中的第二个参数对应 SQL 语句中的第二个问号，依此类推。ULCommand.CommandText 中问号的个数必须至少与此集合中参数的个数相同。

这是 System.Data.IDbCommand.Parameters 和 System.Data.Common.DbCommand.Parameters 的强类型版本。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ULParameterCollection 类”一节第 371 页
- “ULParameter 类”一节第 354 页
- IDbCommand.Connection
- DbCommand.Connection
- “CommandText 属性”一节第 93 页

Plan 属性

UL Ext.: 返回 UltraLite.NET 用于执行查询的访问计划。此属性主要供开发期间使用。

语法

Visual Basic
Public Readonly Property **Plan** As String

C#
public string **Plan** { get;}

属性值

字符串，其中包含对查询执行计划的基于文本的描述。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页

Transaction 属性

指定 ULCommand 执行时所处的 ULTransaction。

语法

Visual Basic
Public Property **Transaction** As ULTransaction

C#
public ULTransaction **Transaction** { get; set; }

属性值

ULCommand 执行时所在的 ULTransaction。它应该是由 ULCommand.Connection 所指定的连接的当前事务。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

注释

如果事务提交或回退后重用命令，则需要重置该属性。

这是 System.Data.IDbCommand.Transaction 和 System.Data.Common.DbCommand.Transaction 的强类型版本。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “BeginTransaction() 方法”一节第 144 页
- “ULTransaction 类”一节第 547 页
- “Connection 属性”一节第 95 页
- IDbCommand.Transaction
- DbCommand.Transaction

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

UpdatedRowSource 属性

指定当 ULDataAdapter 的 Update 方法使用命令结果时，如何将命令结果应用于 DataRow。

语法

Visual Basic
Public Overrides Property **UpdatedRowSource** As UpdateRowSource

C#
public override UpdateRowSource **UpdatedRowSource** { get; set; }

属性值

System.Data.UpdateRowSource 值之一。缺省值为 System.Data.UpdateRowSource.Both。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- UpdateRowSource
- UpdateRowSource.Both

BeginExecuteNonQuery 方法

在已知回调过程和状态信息的情况下，启动由此 ULCommand 说明的 SQL 语句的异步执行。

BeginExecuteNonQuery() 方法

语法

Visual Basic
Public Function **BeginExecuteNonQuery()** As IAsyncResult

C#
public IAsyncResult **BeginExecuteNonQuery();**

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “BeginExecuteNonQuery 方法”一节第 99 页

BeginExecuteNonQuery(AsyncCallback, Object) 方法

在已知回调过程和状态信息的情况下，启动由此 ULCommand 说明的 SQL 语句的异步执行。

语法

Visual Basic
```
Public Function BeginExecuteNonQuery( _
  ByVal callback As AsyncCallback, _
  ByVal stateObject As Object _
) As IAsyncResult
```

C#
```
public IAsyncResult BeginExecuteNonQuery(
  AsyncCallback callback,
  object stateObject
);
```

参数

- **callback** 命令执行完成时调用的 System.AsyncCallback 委派。传递空值（在 Microsoft Visual Basic 中是 Nothing）表示不需要回调。
- **stateObject** 传递给回调过程的用户定义状态对象。使用 System.IAsyncResult.AsyncState 属性从回调过程中检索此对象。

返回值

可只用于进行轮询、只用于等待结果或返回轮询和等待结果这两者的 System.IAsyncResult；调用 EndExecuteNonQuery(IAsyncResult)（返回受影响的行数）时也需要此值。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “BeginExecuteNonQuery 方法”一节第 99 页
- “EndExecuteNonQuery 方法”一节第 105 页
- IAsyncResult.AsyncState
- AsyncCallback
- IAsyncResult

BeginExecuteReader 方法

启动由此 ULCommand 说明的 SQL 语句的异步执行，并检索结果集。

BeginExecuteReader() 方法

启动由此 ULCommand 说明的 SQL 语句的异步执行，并检索结果集。

语法

Visual Basic
```
Public Function BeginExecuteReader() As IAsyncResult
```

C#
```
public IAsyncResult BeginExecuteReader();
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

返回值

可只用于进行轮询、只用于等待结果或返回轮询和等待结果这两者的 System.IAsyncResult；调用 EndExecuteReader(IAsyncResult)（返回可用于检索返回行的 ULDataReader 实例）时也需要此值。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “BeginExecuteReader 方法”一节第 100 页
- “EndExecuteReader 方法”一节第 108 页
- IAsyncResult
- “ULDataReader 类”一节第 257 页

BeginExecuteReader(CommandBehavior) 方法

使用 CommandBehavior 值之一，启动由此 ULCommand 说明的 SQL 语句的异步执行，并检索结果集。

语法

Visual Basic

```
Public Function BeginExecuteReader( _
  ByVal cmdBehavior As CommandBehavior _
) As IAsyncResult
```

C#

```
public IAsyncResult BeginExecuteReader(
  CommandBehavior cmdBehavior
);
```

参数

- **cmdBehavior** 用于描述查询结果及其对连接影响的 System.Data.CommandBehavior 标志的逐位组合。UltraLite.NET 仅支持 System.Data.CommandBehavior.Default、System.Data.CommandBehavior.CloseConnection 和 System.Data.CommandBehavior.SchemaOnly 标志。

返回值

可只用于进行轮询、只用于等待结果或返回轮询和等待结果这两者的 System.IAsyncResult；调用 EndExecuteReader(IAsyncResult)（返回可用于检索返回行的 ULDataReader 实例）时也需要此值。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “BeginExecuteReader 方法”一节第 100 页
- “EndExecuteReader 方法”一节第 108 页
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- IAsyncResult
- “EndExecuteReader 方法”一节第 108 页
- “ULDataReader 类”一节第 257 页

BeginExecuteReader(AsyncCallback, Object) 方法

在已知回调过程和状态信息的情况下，启动由此 ULCommand 说明的 SQL 语句的异步执行并检索结果集。

语法

Visual Basic

```
Public Function BeginExecuteReader( _
  ByVal callback As AsyncCallback, _
  ByVal stateObject As Object _
) As IAsyncResult
```

C#

```
public IAsyncResult BeginExecuteReader(
  AsyncCallback callback,
  object stateObject
);
```

参数

- **callback** 命令执行完成时调用的 System.AsyncCallback 委派。传递空值（在 Microsoft Visual Basic 中是 Nothing）表示不需要回调。
- **stateObject** 传递给回调过程的用户定义状态对象。使用 System.IAsyncResult.AsyncState 属性从回调过程中检索此对象。

返回值

可只用于进行轮询、只用于等待结果或返回轮询和等待结果这两者的 System.IAsyncResult；调用 EndExecuteReader(IAsyncResult)（返回可用于检索返回行的 ULDataReader 实例）时也需要此值。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “BeginExecuteReader 方法”一节第 100 页
- “EndExecuteReader 方法”一节第 108 页
- AsyncCallback
- IAsyncResult.AsyncState
- IAsyncResult.AsyncState
- “EndExecuteReader 方法”一节第 108 页
- IAsyncResult
- “ULDataReader 类”一节第 257 页

BeginExecuteReader(AsyncCallback, Object, CommandBehavior) 方法

在已知回调过程和状态信息的情况下，使用 CommandBehavior 值之一启动由此 ULCommand 说明的 SQL 语句的异步执行并检索结果集。

语法

Visual Basic

```
Public Function BeginExecuteReader( _
  ByVal callback As AsyncCallback, _
  ByVal stateObject As Object, _
  ByVal cmdBehavior As CommandBehavior _
) As IAsyncResult
```

C#

```
public IAsyncResult BeginExecuteReader(
  AsyncCallback callback,
  object stateObject,
  CommandBehavior cmdBehavior
);
```

参数

- **callback** 命令执行完成时调用的 System.AsyncCallback 委派。传递空值（在 Microsoft Visual Basic 中是 Nothing）表示不需要回调。
- **stateObject** 传递给回调过程的用户定义状态对象。使用 System.IAsyncResult.AsyncState 属性从回调过程中检索此对象。
- **cmdBehavior** 用于描述查询结果及其对连接影响的 System.Data.CommandBehavior 标志的逐位组合。UltraLite.NET 仅支持 System.Data.CommandBehavior.Default、System.Data.CommandBehavior.CloseConnection 和 System.Data.CommandBehavior.SchemaOnly 标志。

返回值

可只用于进行轮询、只用于等待结果或返回轮询和等待结果这两者的 System.IAsyncResult；调用 EndExecuteReader(IAsyncResult)（返回可用于检索返回行的 ULDataReader 实例）时也需要此值。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “BeginExecuteReader 方法”一节第 100 页
- AsyncCallback
- IAsyncResult.AsyncState
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- IAsyncResult
- “EndExecuteReader 方法”一节第 108 页
- “ULDataReader 类”一节第 257 页

Cancel 方法

UltraLite.NET 不支持此方法。

语法

Visual Basic
Public Overrides Sub **Cancel()**

C#
public override void **Cancel();**

注释

此方法不执行任何操作。UltraLite.NET 命令在执行时不能被中断。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页

CreateParameter 方法

提供 ULParameter 对象来为 ULCommand 对象提供参数。

语法

Visual Basic
Public Function **CreateParameter()** As ULParameter

C#
public ULParameter **CreateParameter();**

返回值

ULParameter 对象形式的新参数。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

某些 SQL 语句可以带参数，这些参数在语句文本中以问号 (?) 表示。CreateParameter 方法提供一个 ULParameter 对象。您可以设置 ULParameter 中的属性来为该参数指定值。

这是 System.Data.IDbCommand.CreateParameter 和 System.Data.Common.DbCommand.CreateParameter 的强类型版本。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ULParameter 类”一节第 354 页
- IDbCommand.CreateParameter
- DbCommand.CreateParameter

EndExecuteNonQuery 方法

完成 SQL 语句的异步执行。

语法

Visual Basic

```
Public Function EndExecuteNonQuery( _
  ByVal asyncResult As IAsyncResult _
) As Integer
```

C#

```
public int EndExecuteNonQuery(
  IAsyncResult asyncResult
);
```

参数

- **asyncResult** 调用 BeginExecuteNonQuery 而返回的 System.IAsyncResult。

返回值

受影响的行数（与 ExecuteNonQuery 的行为相同）。

注释

每次调用 BeginExecuteNonQuery 时都必须调用一次 EndExecuteNonQuery。必须在 BeginExecuteNonQuery 返回之后执行该调用。ADO.NET 不是线程安全的；由您负责来确保 BeginExecuteNonQuery 已返回。传递给 EndExecuteNonQuery 的 System.IAsyncResult 必须与从即将完成的 BeginExecuteNonQuery 调用返回的 System.IAsyncResult 相同。通过调用 EndExecuteNonQuery 来结束对 BeginExecuteReader 的调用是错误的，反之亦然。

如果在执行命令时出现错误，则调用 EndExecuteNonQuery 时会抛出异常。

等待执行完成有四种方法：

Call EndExecuteNonQuery 命令完成后再调用 EndExecuteNonQuery 块。例如：

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "UPDATE Departments" _
    + " SET DepartmentName = 'Engineering'" _
    + " WHERE DepartmentID=100", _
    conn _
  )
Dim res As IAsyncResult res = _
  cmd.BeginExecuteNonQuery()
' perform other work
' this will block until the command completes
Dim rowCount As Integer = _
  cmd.EndExecuteNonQuery( res )

// C#
ULCommand cmd = new ULCommand(
    "UPDATE Departments"
    + " SET DepartmentName = 'Engineering'"
    + " WHERE DepartmentID=100",
    conn
  );
IAsyncResult res = cmd.BeginExecuteNonQuery();
// perform other work
// this will block until the command completes
int rowCount = cmd.EndExecuteNonQuery( res );
```

Poll the IsCompleted property of the IAsyncResult 可以轮询 IAsyncResult 的 IsCompleted 属性。例如：

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "UPDATE Departments" _
    + " SET DepartmentName = 'Engineering'" _
    + " WHERE DepartmentID=100", _
    conn _
  )
Dim res As IAsyncResult res = _
  cmd.BeginExecuteNonQuery()
While( !res.IsCompleted )
  ' do other work
End While
' this will block until the command completes
Dim rowCount As Integer = _
  cmd.EndExecuteNonQuery( res )

// C#
ULCommand cmd = new ULCommand(
    "UPDATE Departments"
    + " SET DepartmentName = 'Engineering'"
    + " WHERE DepartmentID=100",
    conn
  );
IAsyncResult res = cmd.BeginExecuteNonQuery();
while( !res.IsCompleted ) {
  // do other work
}
// this will block until the command completes
int rowCount = cmd.EndExecuteNonQuery( res );
```

Use the IAsyncResult.AsyncWaitHandle property to get a synchronization object 可以使用 IAsyncResult.AsyncWaitHandle 属性获取同步对象，并进行等待。例如：

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "UPDATE Departments" _
    + " SET DepartmentName = 'Engineering'" _
    + " WHERE DepartmentID=100", _
    conn _
  )
Dim res As IAsyncResult res = _
  cmd.BeginExecuteNonQuery()
' perform other work
Dim wh As WaitHandle = res.AsyncWaitHandle
wh.WaitOne()
' this will not block because the command is finished
Dim rowCount As Integer = _
  cmd.EndExecuteNonQuery( res )

// C#
ULCommand cmd = new ULCommand(
    "UPDATE Departments"
    + " SET DepartmentName = 'Engineering'"
    + " WHERE DepartmentID=100",
    conn
  );
IAsyncResult res = cmd.BeginExecuteNonQuery();
// perform other work
WaitHandle wh = res.AsyncWaitHandle;
wh.WaitOne();
// this will not block because the command is finished
int rowCount = cmd.EndExecuteNonQuery( res );
```

Specify a callback function when calling BeginExecuteNonQuery 可在调用 BeginExecuteNonQuery 时指定回调函数。例如：

```
' Visual Basic
Private Sub callbackFunction(ByVal ar As IAsyncResult)
  Dim cmd As ULCommand = _
    CType(ar.AsyncState, ULCommand)
  ' this won't block since the command has completed
  Dim rowCount As Integer = _
    cmd.EndExecuteNonQuery( res )
End Sub


' elsewhere in the code
Private Sub DoStuff()
  Dim cmd As ULCommand = new ULCommand( _
      "UPDATE Departments" _
      + " SET DepartmentName = 'Engineering'" _
      + " WHERE DepartmentID=100", _
      conn _
    )
  Dim res As IAsyncResult = _
    cmd.BeginExecuteNonQuery( _
        callbackFunction, cmd _
      )
  ' perform other work.  The callback function
  ' will be called when the command completes
End Sub


// C#
```

```
private void callbackFunction( IAsyncResult ar )
{
  ULCommand cmd = (ULCommand) ar.AsyncState;
  // this won't block since the command has completed
  int rowCount = cmd.EndExecuteNonQuery();
}


// elsewhere in the code
private void DoStuff()
{
  ULCommand cmd = new ULCommand(
      "UPDATE Departments"
      + " SET DepartmentName = 'Engineering'"
      + " WHERE DepartmentID=100",
      conn
    );
  IAsyncResult res = cmd.BeginExecuteNonQuery(
      callbackFunction, cmd
    );
  // perform other work.  The callback function
  // will be called when the command completes
}
```

回调函数在单独的线程中执行，因此与在线程化程序中更新用户界面有关的常见告诫也适用。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “BeginExecuteNonQuery() 方法”一节第 99 页
- IAsyncResult

EndExecuteReader 方法

完成 SQL 语句的异步执行，并返回所请求的 ULDataReader。

语法

Visual Basic

```
Public Function EndExecuteReader( _
  ByVal asyncResult As IAsyncResult _
) As ULDataReader
```

C#

```
public ULDataReader EndExecuteReader(
  IAsyncResult asyncResult
);
```

参数

- **asyncResult** 调用 BeginExecuteReader 而返回的 System.IAsyncResult。

返回值

可用于检索所请求行的 ULDataReader 对象（与 ExecuteReader 的行为相同）。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

每次调用 BeginExecuteReader 时都必须调用一次 EndExecuteReader。必须在 BeginExecuteReader 返回之后执行该调用。ADO.NET 不是线程安全的；由您负责来确保 BeginExecuteReader 已返回。传递给 EndExecuteReader 的 System.IAsyncResult 必须与从即将完成的 BeginExecuteReader 调用返回的 System.IAsyncResult 相同。通过调用 EndExecuteReader 来结束对 BeginExecuteNonQuery 的调用是错误的，反之亦然。

如果在执行命令时出现错误，则调用 EndExecuteReader 时会抛出异常。

等待执行完成有四种方法：

Call EndExecuteReader 在命令完成后再调用 EndExecuteReader 块。例如：

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "SELECT * FROM Departments", conn _
  )
Dim res As IAsyncResult res = _
  cmd.BeginExecuteReader()
' perform other work
' this will block until the command completes
Dim reader As ULDataReader = _
  cmd.EndExecuteReader( res )

// C#
ULCommand cmd = new ULCommand(
    "SELECT * FROM Departments", conn
  );
IAsyncResult res = cmd.BeginExecuteReader();
// perform other work
// this will block until the command completes
ULDataReader reader = cmd.EndExecuteReader( res );
```

Poll the IsCompleted property of the IAsyncResult 可以轮询 IAsyncResult 的 IsCompleted 属性。例如：

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "SELECT * FROM Departments", conn _
  )
Dim res As IAsyncResult res = _
  cmd.BeginExecuteReader()
While( !res.IsCompleted )
  ' do other work
End While
' this will block until the command completes
Dim reader As ULDataReader = _
  cmd.EndExecuteReader( res )

// C#
ULCommand cmd = new ULCommand(
    "SELECT * FROM Departments", conn
  );
IAsyncResult res = cmd.BeginExecuteReader();
while( !res.IsCompleted ) {
  // do other work
}
// this will block until the command completes
ULDataReader reader = cmd.EndExecuteReader( res );
```

Use the IAsyncResult.AsyncWaitHandle property to get a synchronization object 可以使用 IAsyncResult.AsyncWaitHandle 属性获取同步对象，并进行等待。例如：

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "SELECT * FROM Departments", conn _
  )
Dim res As IAsyncResult res = _
  cmd.BeginExecuteReader()
' perform other work
Dim wh As WaitHandle = res.AsyncWaitHandle
wh.WaitOne()
' this will not block because the command is finished
Dim reader As ULDataReader = _
  cmd.EndExecuteReader( res )

// C#
ULCommand cmd = new ULCommand(
    "SELECT * FROM Departments", conn
  );
IAsyncResult res = cmd.BeginExecuteReader();
// perform other work
WaitHandle wh = res.AsyncWaitHandle;
wh.WaitOne();
// this will not block because the command is finished
ULDataReader reader = cmd.EndExecuteReader( res );
```

Specify a callback function when calling BeginExecuteReader 可在调用 BeginExecuteReader 时指定回调函数。例如：

```
' Visual Basic
Private Sub callbackFunction(ByVal ar As IAsyncResult)
  Dim cmd As ULCommand = _
    CType(ar.AsyncState, ULCommand)
  ' this won't block since the command has completed
  Dim reader As ULDataReader = cmd.EndExecuteReader()
End Sub


' elsewhere in the code
Private Sub DoStuff()
  Dim cmd As ULCommand = new ULCommand( _
      "SELECT * FROM Departments", conn _
    )
  Dim res As IAsyncResult = _
    cmd.BeginExecuteReader( _
        callbackFunction, cmd _
      )
  ' perform other work.  The callback function
  ' will be called when the command completes
End Sub


// C#
private void callbackFunction( IAsyncResult ar )
{
  ULCommand cmd = (ULCommand) ar.AsyncState;
  // this won't block since the command has completed
  ULDataReader reader = cmd.EndExecuteReader();
}
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
// elsewhere in the code
private void DoStuff()
{
  ULCommand cmd = new ULCommand(
      "SELECT * FROM Departments", conn
    );
  IAsyncResult res = cmd.BeginExecuteReader(
      callbackFunction, cmd
    );
  // perform other work.  The callback function
  // will be called when the command completes
}
```

回调函数在单独的线程中执行，因此与在线程化程序中更新用户界面有关的常见告诫也适用。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “BeginExecuteReader() 方法”一节第 100 页
- “ULDataReader 类”一节第 257 页
- IAsyncResult

ExecuteNonQuery 方法

执行一条不返回结果集的语句，如 SQL INSERT、DELETE 或 UPDATE 语句。

语法

Visual Basic
Public Overrides Function **ExecuteNonQuery()** As Integer

C#
public override int **ExecuteNonQuery();**

返回值

受影响的行数。

注释

该语句就是当前的 ULCommand 对象，它具有必需的 ULCommand.CommandText 和 ULCommand.Parameters。

如果是 UPDATE、INSERT 和 DELETE 语句，返回值将是受该命令影响的行数。对于所有其它类型的语句和回退，返回值将是 -1。

ULCommand.CommandType 不能为 System.Data.CommandType.TableDirect。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “CommandText 属性”一节第 93 页
- “Parameters 属性”一节第 97 页
- “CommandType 属性”一节第 94 页
- CommandType.TableDirect
- “Connection 属性”一节第 95 页
- “Transaction 属性”一节第 98 页
- “CommandText 属性”一节第 93 页

ExecuteReader 方法

执行 SQL SELECT 语句并返回结果集。

ExecuteReader() 方法

执行 SQL SELECT 语句并返回结果集。

语法

Visual Basic
Public Function **ExecuteReader()** As ULDataReader

C#
public ULDataReader **ExecuteReader();**

返回值

ULDataReader 对象形式的结果集。

注释

该语句就是当前的 ULCommand 对象，它具有必需的 ULCommand.CommandText 和任何 ULCommand.Parameters。ULDataReader 对象为只读结果集。要获得可编辑的结果集，请使用 ULCommand.ExecuteResultSet()、ULCommand.ExecuteTable() 或 ULDataAdapter。

如果 ULCommand.CommandType 为 System.Data.CommandType.TableDirect，则 ExecuteReader 会执行 ULCommand.ExecuteTable() 并返回一个转换为 ULDataReader 的 ULTable。

出于性能原因，SELECT 语句在缺省情况下标记为只读。如果要将该查询用于执行更新，则此语句必须以 "FOR UPDATE" 作为结尾。

这是 System.Data.IDbCommand.ExecuteReader() 和 System.Data.Common.DbCommand.ExecuteReader() 的强类型版本。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ExecuteReader 方法”一节第 112 页
- “ExecuteReader(CommandBehavior) 方法”一节第 113 页
- “CommandText 属性”一节第 93 页
- “Parameters 属性”一节第 97 页
- “ULDataReader 类”一节第 257 页
- “ExecuteResultSet() 方法”一节第 115 页
- “ExecuteTable() 方法”一节第 118 页
- “ULDataAdapter 类”一节第 226 页
- “CommandType 属性”一节第 94 页
- CommandType.TableDirect
- “ExecuteTable() 方法”一节第 118 页
- “ULTable 类”一节第 511 页
- “ULDataReader 类”一节第 257 页
- IDbCommand.ExecuteReader
- DbCommand.ExecuteReader
- “Connection 属性”一节第 95 页
- “Transaction 属性”一节第 98 页

ExecuteReader(CommandBehavior) 方法

执行具有指定命令行为的 SQL SELECT 语句并返回结果集。

语法

Visual Basic

```
Public Function ExecuteReader( _
  ByVal cmdBehavior As CommandBehavior _
) As ULDataReader
```

C#

```
public ULDataReader ExecuteReader(
  CommandBehavior cmdBehavior
);
```

参数

- **cmdBehavior** 用于描述查询结果及其对连接影响的 System.Data.CommandBehavior 标志的逐位组合。UltraLite.NET 仅支持 System.Data.CommandBehavior.Default、System.Data.CommandBehavior.CloseConnection 和 System.Data.CommandBehavior.SchemaOnly 标志。

返回值

ULDataReader 对象形式的结果集。

注释

该语句就是当前的 ULCommand 对象，它具有必需的 ULCommand.CommandText 和任何 ULCommand.Parameters。ULDataReader 对象为只读结果集。要获得可编辑的结果集，请使用 ULCommand.ExecuteResultSet(CommandBehavior)、ULCommand.ExecuteTable(CommandBehavior) 或 ULDataAdapter。

如果 ULCommand.CommandType 为 System.Data.CommandType.TableDirect，则 ExecuteReader 会执行 ULCommand.ExecuteTable(CommandBehavior) 并返回一个转换为 ULDataReader 的 ULTable。

出于性能原因，SELECT 语句在缺省情况下标记为只读。如果要将该查询用于执行更新，则此语句必须以 "FOR UPDATE" 作为结尾。

这是 System.Data.IDbCommand.ExecuteReader(System.Data.CommandBehavior) 和 System.Data.Common.DbCommand.ExecuteReader(System.Data.CommandBehavior) 的强类型版本。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ExecuteReader 方法”一节第 112 页
- “ExecuteReader() 方法”一节第 112 页
- “CommandText 属性”一节第 93 页
- “Parameters 属性”一节第 97 页
- “ULDataReader 类”一节第 257 页
- “ExecuteResultSet(CommandBehavior) 方法”一节第 116 页
- “ExecuteTable(CommandBehavior) 方法”一节第 119 页
- “ULDataAdapter 类”一节第 226 页
- “CommandType 属性”一节第 94 页
- CommandType.TableDirect
- “ExecuteTable(CommandBehavior) 方法”一节第 119 页
- “ULTable 类”一节第 511 页
- “ULDataReader 类”一节第 257 页
- IDbCommand.ExecuteReader
- DbCommand.ExecuteReader
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- “Connection 属性”一节第 95 页
- “Transaction 属性”一节第 98 页
- “CommandText 属性”一节第 93 页

ExecuteResultSet 方法

UL Ext.: 执行 SQL SELECT 语句并以 ULResultSet 形式返回结果集。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ExecuteResultSet() 方法

UL Ext.: 执行 SQL SELECT 语句并以 ULResultSet 形式返回结果集。

语法

Visual Basic
Public Function **ExecuteResultSet()** As ULResultSet

C#
public ULResultSet **ExecuteResultSet();**

返回值

ULResultSet 对象形式的结果集。

注释

该语句就是当前的 ULCommand 对象，它具有必需的 ULCommand.CommandText 和任何 ULCommand.Parameters。ULResultSet 对象为可编辑的结果集，您可在其上执行定位更新和删除。要获得完全可编辑的结果集，请使用 ULCommand.ExecuteTable() 或 ULDataAdapter。

如果 ULCommand.CommandType 为 System.Data.CommandType.TableDirect，则 ExecuteReader 会执行 ULCommand.ExecuteTable() 并返回一个转换为 ULResultSet 的 ULTable。

ULCommand.ExecuteResultSet 支持通过动态 SQL 进行的定位更新和删除。

示例

```
cmd.CommandText = "SELECT id, season, price FROM OurProducts";
ULResultSet rs = cmd.ExecuteResultSet();
while( rs.Read() ) {
    string season = rs.GetString( 1 );
    double price = rs.GetDouble( 2 );
    if( season.Equals( "summer" ) && price < 100.0 ) {
        rs.SetDouble( 2, price * .5 );
        rs.Update();
    }
    if( season.Equals( "discontinued" ) ) {
        rs.Delete();
    }
}
rs.Close();
```

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ExecuteResultSet 方法”一节第 114 页
- “ExecuteResultSet(CommandBehavior) 方法”一节第 116 页
- “ULCommand 类”一节第 86 页
- “CommandText 属性”一节第 93 页
- “Parameters 属性”一节第 97 页
- “CommandType 属性”一节第 94 页
- CommandType.TableDirect
- “ULResultSet 类”一节第 394 页
- “ULTable 类”一节第 511 页

ExecuteResultSet(CommandBehavior) 方法

UL Ext.: 执行具有指定命令行为的 SQL SELECT 语句并以 ULResultSet 形式返回结果集。

语法

Visual Basic

```
Public Function ExecuteResultSet( _
  ByVal cmdBehavior As CommandBehavior _
) As ULResultSet
```

C#

```
public ULResultSet ExecuteResultSet(
  CommandBehavior cmdBehavior
);
```

参数

- **cmdBehavior** 用于描述查询结果及其对连接影响的 System.Data.CommandBehavior 标志的逐位组合。UltraLite.NET 仅支持 System.Data.CommandBehavior.Default、System.Data.CommandBehavior.CloseConnection 和 System.Data.CommandBehavior.SchemaOnly 标志。

返回值

ULResultSet 对象形式的结果集。

注释

该语句就是当前的 ULCommand 对象，它具有必需的 ULCommand.CommandText 和任何 ULCommand.Parameters。ULResultSet 对象为可编辑的结果集，您可在其上执行定位更新和删除。要获得完全可编辑的结果集，请使用 ULCommand.ExecuteTable(CommandBehavior) 或 ULDataAdapter。

如果 ULCommand.CommandType 为 System.Data.CommandType.TableDirect，则 ExecuteReader 会执行 ULCommand.ExecuteTable(CommandBehavior) 并返回一个转换为 ULResultSet 的 ULTable。

ULCommand.ExecuteResultSet 支持通过动态 SQL 进行的定位更新和删除。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ExecuteResultSet 方法”一节第 114 页
- “ExecuteReader() 方法”一节第 112 页
- “ULResultSet 类”一节第 394 页
- “CommandText 属性”一节第 93 页
- “Parameters 属性”一节第 97 页
- “ExecuteTable(CommandBehavior) 方法”一节第 119 页
- “ULDataAdapter 类”一节第 226 页
- “CommandType 属性”一节第 94 页
- CommandType.TableDirect
- “ExecuteTable(CommandBehavior) 方法”一节第 119 页
- “ULTable 类”一节第 511 页
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- “Connection 属性”一节第 95 页
- “Transaction 属性”一节第 98 页

ExecuteScalar 方法

执行 SQL SELECT 语句并返回单个值。

语法

Visual Basic
Public Overrides Function **ExecuteScalar()** As Object

C#
public override object **ExecuteScalar();**

返回值

结果集中第一行的第一列，如果结果集为空，则为空值引用（在 Visual Basic 中是 Nothing）。

注释

该语句就是当前的 ULCommand 对象，它具有必需的 ULCommand.CommandText 和任何 ULCommand.Parameters。

如果在返回多个行和列的查询上调用该方法，则只会返回第一行的第一列。

如果 ULCommand.CommandType 为 System.Data.CommandType.TableDirect，则 ExecuteScalar 会执行 ULCommand.ExecuteTable() 并返回第一行的第一列。

出于性能原因，SELECT 语句在缺省情况下标记为只读。如果要将该查询用于执行更新，则此语句必须以 "FOR UPDATE" 作为结尾。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “Connection 属性”一节第 95 页
- “Transaction 属性”一节第 98 页
- “CommandText 属性”一节第 93 页
- “IndexName 属性”一节第 96 页
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- “CommandType 属性”一节第 94 页
- “Parameters 属性”一节第 97 页
- CommandType.TableDirect

ExecuteTable 方法

UL Ext.: 在 ULTable 中检索数据库表以便直接进行操作。ULCommand.CommandText 被解释为表的名称，而 ULCommand.IndexName 可用于指定表的排序顺序。

ExecuteTable() 方法

UL Ext.: 在 ULTable 中检索数据库表以便直接进行操作。ULCommand.CommandText 被解释为表的名称，而 ULCommand.IndexName 可用于指定表的排序顺序。

语法

Visual Basic
Public Function **ExecuteTable()** As ULTable

C#
public ULTable **ExecuteTable();**

返回值

ULTable 对象形式的表。

注释

必须将 ULCommand.CommandType 设置为 System.Data.CommandType.TableDirect。

如果 ULCommand.IndexName 为空值引用（在 Visual Basic 中是 Nothing），则使用主键打开表。否则，使用 ULCommand.IndexName 值打开表，该值是作为排序依据的索引的名称。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ExecuteTable 方法”一节第 118 页
- “ExecuteTable(CommandBehavior) 方法”一节第 119 页
- “Connection 属性”一节第 95 页
- “Transaction 属性”一节第 98 页
- “CommandText 属性”一节第 93 页
- “IndexName 属性”一节第 96 页
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- “CommandType 属性”一节第 94 页
- CommandType.TableDirect
- “ULTable 类”一节第 511 页

ExecuteTable(CommandBehavior) 方法

UL Ext.: 以指定的命令行为检索数据库表以便直接进行操作。ULCommand.CommandText 被解释为表的名称，而 ULCommand.IndexName 可用于指定表的排序顺序。

语法

Visual Basic

```
Public Function ExecuteTable( _
  ByVal cmdBehavior As CommandBehavior _
) As ULTable
```

C#

```
public ULTable ExecuteTable(
  CommandBehavior cmdBehavior
);
```

参数

- **cmdBehavior** 用于描述查询结果及其对连接影响的 System.Data.CommandBehavior 标志的逐位组合。UltraLite.NET 仅支持 System.Data.CommandBehavior.Default、System.Data.CommandBehavior.CloseConnection 和 System.Data.CommandBehavior.SchemaOnly 标志。

返回值

ULTable 对象形式的表。

注释

必须将 ULCommand.CommandType 设置为 System.Data.CommandType.TableDirect。

如果 ULCommand.IndexName 为空值引用（在 Visual Basic 中是 Nothing），则使用主键打开表。否则，使用 ULCommand.IndexName 值打开表，该值是作为排序依据的索引的名称。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “ExecuteTable 方法”一节第 118 页
- “ExecuteTable() 方法”一节第 118 页
- “Connection 属性”一节第 95 页
- “Transaction 属性”一节第 98 页
- “CommandText 属性”一节第 93 页
- “IndexName 属性”一节第 96 页
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- “CommandType 属性”一节第 94 页
- CommandType.TableDirect

Prepare 方法

预编译并存储此命令的 SQL 语句。

语法

Visual Basic
Public Overrides Sub **Prepare()**

C#
public override void **Prepare();**

注释

当只有参数值发生更改时，对语句进行预编译可以有效地重复使用语句。更改此命令中的任何其它属性都会使得准备的语句无效。

UltraLite.NET 不需要显式准备语句，因为所有未预先准备的命令都会在调用各种 Execute 方法时准备好。

另请参见

- “ULCommand 类”一节第 86 页
- “ULCommand 成员”一节第 87 页
- “Connection 属性”一节第 95 页
- “Transaction 属性”一节第 98 页
- “CommandText 属性”一节第 93 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULCommandBuilder 类

将自动生成单表命令，这些命令用于使您对 System.Data.DataSet 所做的更改与关联数据库保持一致。

语法

Visual Basic
Public Class **ULCommandBuilder**
  Inherits DbCommandBuilder

C#
public class **ULCommandBuilder**: DbCommandBuilder

注释

ULDataAdapter 不会自动生成用于使您对 System.Data.DataSet 所做的更改与关联数据源保持一致的 SQL 语句。但是，如果您设置了 ULDataAdapter 的 SelectCommand 属性，则可创建一个 ULCommandBuilder 对象以自动生成 SQL 语句来实现单表更新。随后，ULCommandBuilder 会生成您未设置的任何其它 SQL 语句。

Inherits: System.ComponentModel.Component

Implements: System.IDisposable

示例

以下示例使用 ULCommand、ULDataAdapter 以及 ULConnection，从数据源中选择行。在本例中，传递了一个连接字符串、一个作为 SQL SELECT 语句的查询字符串，以及一个作为数据库表名称的字符串。该示例随后会创建一个 ULCommandBuilder。

```
' Visual Basic
Public Shared Function SelectULRows(ByVal connectionString As String, _
    ByVal queryString As String, ByVal tableName As String)

    Dim connection As ULConnection = New ULConnection(connectionString)
    Dim adapter As ULDataAdapter = New ULDataAdapter()

       adapter.SelectCommand = New ULCommand(queryString, connection)

       Dim builder As ULCommandBuilder = New ULCommandBuilder(adapter)

    connection.Open()

    Dim dataSet As DataSet = New DataSet()
    adapter.Fill(dataSet, tableName)

    'code to modify data in DataSet here

    'Without the ULCommandBuilder this line would fail
    adapter.Update(dataSet, tableName)

    Return dataSet

End Function

// C#
```

```
public static DataSet SelectULRows(string connectionString,
    string queryString, string tableName)
{
    using (ULConnection connection = new ULConnection(connectionString))
    {
        ULDataAdapter adapter = new ULDataAdapter();
        adapter.SelectCommand = new ULCommand(queryString, connection);
        ULCommandBuilder builder = new ULCommandBuilder(adapter);

        connection.Open();

        DataSet dataSet = new DataSet();
        adapter.Fill(dataSet, tableName);

           //code to modify data in DataSet here

        //Without the ULCommandBuilder this line would fail
        adapter.Update(dataSet, tableName);

        return dataSet;
    }
}
```

另请参见

- “ULCommandBuilder 成员”一节第 122 页
- DataSet
- “ULDataAdapter 类”一节第 226 页
- Component
- IDisposable
- “ULCommand 类”一节第 86 页
- “ULConnection 类”一节第 131 页

ULCommandBuilder 成员

公共构造函数

成员名称	说明
“ULCommandBuilder 构造函数”一节第 124 页	初始化一个新的“ULCommandBuilder 类”一节第 121 页实例。

公共属性

成员名称	说明
CatalogLocation（继承自 DbCommandBuilder）	设置或获取 DbCommandBuilder 实例的 CatalogLocation。
CatalogSeparator（继承自 DbCommandBuilder）	设置或获取用作 DbCommandBuilder 实例的目录分隔符的字符串。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
ConflictOption（继承自 DbCommandBuilder）	指定 DbCommandBuilder 即将使用哪一个 ConflictOption。
“DataAdapter 属性”一节 第 125 页	获取或设置会为其自动生成 SQL 语句的 ULDataAdapter 对象。
QuotePrefix（继承自 DbCommandBuilder）	获取或设置要在指定名称包含空格或保留标识之类字符的数据库对象（例如，表或列）时使用的一个或多个起始字符。
QuoteSuffix（继承自 DbCommandBuilder）	获取或设置要在指定名称包含空格或保留标识之类字符的数据库对象（例如，表或列）时使用的一个或多个起始字符。
SchemaSeparator（继承自 DbCommandBuilder）	获取和设置要用作模式标识符与其它任何标识符之间分隔符的字符。
SetAllValues（继承自 DbCommandBuilder）	指定在更新语句中是包含所有列值，还是只包含更改的列值。

公共方法

成员名称	说明
“GetDeleteCommand 方法”一节第 125 页	获取在数据源中执行删除所需的自动生成的 DbCommand 对象。
“GetInsertCommand 方法”一节第 127 页	获取在数据源中执行插入所需的自动生成的 DbCommand 对象。
“GetUpdateCommand 方法”一节第 128 页	获取在数据源中执行更新所需的自动生成的 DbCommand 对象。
QuoteIdentifier（继承自 DbCommandBuilder）	如果在正确的目录中有不带引号的标识符，返回标识符正确的加引号形式，包括正确转义标识符中的所有嵌入式引号。
RefreshSchema（继承自 DbCommandBuilder）	清除与此 DbCommandBuilder 关联的命令。
UnquoteIdentifier（继承自 DbCommandBuilder）	如果有带引号的标识符，返回该标识符的正确的不加引号形式，包括正确取消转义该标识符中的所有嵌入式引号。

另请参见

- “ULCommandBuilder 类”一节第 121 页
- DataSet
- “ULDataAdapter 类”一节第 226 页
- Component
- IDisposable
- “ULCommand 类”一节第 86 页
- “ULConnection 类”一节第 131 页

ULCommandBuilder 构造函数

初始化一个新的“ULCommandBuilder 类”一节第 121 页实例。

ULCommandBuilder() 构造函数

初始化 ULCommandBuilder 对象。

语法

Visual Basic

```
Public Sub New()
```

C#

```
public  ULCommandBuilder();
```

另请参见

- “ULCommandBuilder 类”一节第 121 页
- “ULCommandBuilder 成员”一节第 122 页
- “ULCommandBuilder 构造函数”一节第 124 页
- “ULCommandBuilder(ULDataAdapter) 构造函数”一节第 124 页

ULCommandBuilder(ULDataAdapter) 构造函数

用指定的 ULDataAdapter 对象初始化 ULCommandBuilder 对象。

语法

Visual Basic

```
Public Sub New( _
  ByVal adapter As ULDataAdapter _
)
```

C#

```
public  ULCommandBuilder(
  ULDataAdapter adapter
);
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

参数

- **adapter** ULDataAdapter 对象。

另请参见

- “ULCommandBuilder 类”一节第 121 页
- “ULCommandBuilder 成员”一节第 122 页
- “ULCommandBuilder 构造函数”一节第 124 页
- “ULDataAdapter 类”一节第 226 页

DataAdapter 属性

获取或设置会为其自动生成 SQL 语句的 ULDataAdapter 对象。

语法

Visual Basic
Public Property **DataAdapter** As ULDataAdapter

C#
public ULDataAdapter **DataAdapter** { get; set; }

属性值

ULDataAdapter 对象。

另请参见

- “ULCommandBuilder 类”一节第 121 页
- “ULCommandBuilder 成员”一节第 122 页
- “ULDataAdapter 类”一节第 226 页

GetDeleteCommand 方法

获取在数据源中执行删除所需的自动生成的 DbCommand 对象。

GetDeleteCommand(Boolean) 方法

获取自动生成的、执行数据库删除操作所需的 ULCommand 对象。

语法

Visual Basic
```
Public Function GetDeleteCommand( _
  ByVal useColumnsForParameterNames As Boolean _
) As ULCommand
```

C#
```
public ULCommand GetDeleteCommand(
```

```
  bool useColumnsForParameterNames
);
```

参数

- **useColumnsForParameterNames** 如果为 true，则在可能的情况下生成与列名匹配的参数名。如果为 false，则生成 @p1、@p2 等参数名。

返回值

自动生成的、执行删除操作所需的 ULCommand 对象。

注释

首先生成 SQL 语句后，如果应用程序以任何方式对 ULDataAdapter.SelectCommand 进行了更改，则它必须显式调用 DbCommandBuilder.RefreshSchema。否则，GetDeleteCommand 方法会继续使用来自上一条语句的信息，而该信息可能是不正确的。当应用程序调用 DbDataAdapter.Update(System.Data.DataSet) 或 GetDeleteCommand 方法时，会首先生成 SQL 语句。

另请参见

- “ULCommandBuilder 类”一节第 121 页
- “ULCommandBuilder 成员”一节第 122 页
- “GetDeleteCommand 方法”一节第 125 页
- “ULCommand 类”一节第 86 页
- DbCommandBuilder.RefreshSchema
- “SelectCommand 属性”一节第 233 页
- DbDataAdapter.Update
- DbCommandBuilder.DataAdapter

GetDeleteCommand() 方法

获取自动生成的、执行数据库删除操作所需的 ULCommand 对象。

语法

Visual Basic

```
Public Function GetDeleteCommand() As ULCommand
```

C#

```
public ULCommand GetDeleteCommand();
```

返回值

自动生成的、执行删除操作所需的 ULCommand 对象。

注释

首先生成 SQL 语句后，如果应用程序以任何方式对 ULDataAdapter.SelectCommand 进行了更改，则它必须显式调用 DbCommandBuilder.RefreshSchema。否则，GetDeleteCommand 方法会继续使用来自上一条语句的信息，而该信息可能是不正确的。当应用程序调用 DbDataAdapter.Update(System.Data.DataSet) 或 GetDeleteCommand 方法时，会首先生成 SQL 语句。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULCommandBuilder 类”一节第 121 页
- “ULCommandBuilder 成员”一节第 122 页
- “GetDeleteCommand 方法”一节第 125 页
- “ULCommand 类”一节第 86 页
- DbCommandBuilder.RefreshSchema
- “SelectCommand 属性”一节第 233 页
- DbDataAdapter.Update
- DbCommandBuilder.DataAdapter

GetInsertCommand 方法

获取在数据源中执行插入所需的自动生成的 DbCommand 对象。

GetInsertCommand(Boolean) 方法

获取自动生成的、执行数据库插入操作所需的 ULCommand 对象。

语法

Visual Basic

```
Public Function GetInsertCommand( _
  ByVal useColumnsForParameterNames As Boolean _
) As ULCommand
```

C#

```
public ULCommand GetInsertCommand(
  bool useColumnsForParameterNames
);
```

参数

- **useColumnsForParameterNames** 如果为 true，则在可能的情况下生成与列名匹配的参数名。如果为 false，则生成 @p1、@p2 等等。

返回值

自动生成的、执行插入操作所需的 ULCommand 对象。

注释

首先生成 SQL 语句后，如果应用程序以任何方式对 ULDataAdapter.SelectCommand 进行了更改，则它必须显式调用 DbCommandBuilder.RefreshSchema。否则，GetInsertCommand 方法会继续使用来自上一条语句的信息，而该信息可能是不正确的。当应用程序调用 DbDataAdapter.Update(System.Data.DataSet) 或 GetInsertCommand 方法时，会首先生成 SQL 语句。

另请参见

- “ULCommandBuilder 类”一节第 121 页
- “ULCommandBuilder 成员”一节第 122 页
- “GetInsertCommand 方法”一节第 127 页
- “ULCommand 类”一节第 86 页
- DbCommandBuilder.RefreshSchema
- “SelectCommand 属性”一节第 233 页
- DbDataAdapter.Update
- DbCommandBuilder.DataAdapter

GetInsertCommand() 方法

获取自动生成的、执行数据库插入操作所需的 ULCommand 对象。

语法

Visual Basic
Public Function **GetInsertCommand()** As ULCommand

C#
public ULCommand **GetInsertCommand();**

返回值

自动生成的、执行插入操作所需的 ULCommand 对象。

注释

首先生成 SQL 语句后，如果应用程序以任何方式对 ULDataAdapter.SelectCommand 进行了更改，则它必须显式调用 DbCommandBuilder.RefreshSchema。否则，GetInsertCommand 方法会继续使用来自上一条语句的信息，而该信息可能是不正确的。当应用程序调用 DbDataAdapter.Update(System.Data.DataSet) 或 GetInsertCommand 方法时，会首先生成 SQL 语句。

另请参见

- “ULCommandBuilder 类”一节第 121 页
- “ULCommandBuilder 成员”一节第 122 页
- “GetInsertCommand 方法”一节第 127 页
- “ULCommand 类”一节第 86 页
- DbCommandBuilder.RefreshSchema
- “SelectCommand 属性”一节第 233 页
- “ULCommand 类”一节第 86 页
- DbCommandBuilder.DataAdapter

GetUpdateCommand 方法

获取在数据源中执行更新所需的自动生成的 DbCommand 对象。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

GetUpdateCommand(Boolean) 方法

获取自动生成的、执行数据库更新操作所需的 ULCommand 对象。

语法

Visual Basic

```
Public Function GetUpdateCommand( _
  ByVal useColumnsForParameterNames As Boolean _
) As ULCommand
```

C#

```
public ULCommand GetUpdateCommand(
  bool useColumnsForParameterNames
);
```

参数

- **useColumnsForParameterNames** 如果为 true，则在可能的情况下生成与列名匹配的参数名。如果为 false，则生成 @p1、@p2 等等。

返回值

自动生成的、执行更新操作所需的 ULCommand 对象。

注释

首先生成 SQL 语句后，如果应用程序以任何方式对 ULDataAdapter.SelectCommand 进行了更改，则它必须显式调用 DbCommandBuilder.RefreshSchema。否则，GetUpdateCommand 方法会继续使用来自上一条语句的信息，而该信息可能是不正确的。当应用程序调用 DbDataAdapter.Update(System.Data.DataSet) 或 GetUpdateCommand 方法时，会首先生成 SQL 语句。

另请参见

- “ULCommandBuilder 类”一节第 121 页
- “ULCommandBuilder 成员”一节第 122 页
- “GetUpdateCommand 方法”一节第 128 页
- “ULCommand 类”一节第 86 页
- DbCommandBuilder.RefreshSchema
- “SelectCommand 属性”一节第 233 页
- DbDataAdapter.Update
- DbCommandBuilder.DataAdapter

GetUpdateCommand() 方法

获取自动生成的、执行数据库更新操作所需的 ULCommand 对象。

语法

Visual Basic

```
Public Function GetUpdateCommand() As ULCommand
```

C#
public ULCommand **GetUpdateCommand();**

返回值

自动生成的、执行更新操作所需的 ULCommand 对象。

注释

首先生成 SQL 语句后，如果应用程序以任何方式对 ULDataAdapter.SelectCommand 进行了更改，则它必须显式调用 DbCommandBuilder.RefreshSchema。否则，GetUpdateCommand 方法会继续使用来自上一条语句的信息，而该信息可能是不正确的。当应用程序调用 DbDataAdapter.Update(System.Data.DataSet) 或 GetUpdateCommand 方法时，会首先生成 SQL 语句。

另请参见

- “ULCommandBuilder 类”一节第 121 页
- “ULCommandBuilder 成员”一节第 122 页
- “GetUpdateCommand 方法”一节第 128 页
- “ULCommand 类”一节第 86 页
- DbCommandBuilder.RefreshSchema
- “SelectCommand 属性”一节第 233 页
- DbDataAdapter.Update
- DbCommandBuilder.DataAdapter

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULConnection 类

表示与 UltraLite.NET 数据库的连接。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULConnection**
  Inherits DbConnection

C#
public sealed class **ULConnection**: DbConnection

注释

要使用 UltraLite.NET 的 UltraLite 引擎运行时，请在使用任何其它 UltraLite.NET API 之前将 ULDatabaseManager.RuntimeType 设置为适当的值。

可使用 ULConnection.Open 打开与现有数据库的连接。

您必须打开一个连接，然后才可执行其它操作，而且在连接上完成所有操作之后，必须在应用程序终止前关闭该连接。此外，必须在关闭连接前关闭在该连接上打开的所有结果集和表。

可以使用已打开连接的 ULConnection.Schema 访问数据库的模式。

Implements: System.Data.IDbCommand、System.IDisposable

另请参见

- “ULConnection 成员”一节第 131 页
- “RuntimeType 属性”一节第 239 页
- “Open 方法”一节第 166 页
- “Schema 属性”一节第 142 页
- IDbConnection
- IDisposable

ULConnection 成员

公共构造函数

成员名称	说明
“ULConnection 构造函数”一节第 135 页	初始化一个新的“ULConnection 类”一节第 131 页实例。

公共字段

成员名称	说明
“INVALID_DATABASE_ID 字段”一节第 137 页	**UL Ext.:** 一个数据库 ID 常量，表示尚未设置 ULConnection.DatabaseID。此字段为常量并且是只读字段。

公共属性

成员名称	说明
“ConnectionString 属性”一节第 137 页	指定用于打开 UltraLite.NET 数据库连接的参数。可以使用 ULConnectionParms 对象提供连接字符串。
“ConnectionTimeout 属性”一节第 138 页	UltraLite.NET 不支持此功能。
“DataSource 属性”一节第 139 页	UltraLite.NET 不支持此功能。
“Database 属性”一节第 139 页	返回打开连接的数据库的名称。
“DatabaseID 属性”一节第 140 页	**UL Ext.:** 指定用于全局自动增量列的数据库 ID 值。
“DatabaseManager 属性”一节第 140 页	**UL Ext.:** 提供对单个 ULDatabaseManager 对象的访问。
“GlobalAutoIncrementUsage 属性”一节第 141 页	**UL Ext.:** 返回已经使用的可用全局自动增量值的百分比。
“LastIdentity 属性”一节第 141 页	**UL Ext.:** 返回最近使用的标识值。
“Schema 属性”一节第 142 页	**UL Ext.:** 提供对此连接相关联的当前数据库模式的访问。
“ServerVersion 属性”一节第 142 页	UltraLite.NET 不支持此功能。
“State 属性”一节第 143 页	返回连接的当前状态。
“SyncParms 属性”一节第 143 页	**UL Ext.:** 指定此连接的同步设置。
“SyncResult 属性”一节第 144 页	**UL Ext.:** 返回此连接上次的同步结果。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

公共方法

成员名称	说明
“BeginTransaction 方法”一节第 144 页	启动数据库事务。
“CancelGetNotification 方法”一节第 146 页	取消与给定名称匹配的所有队列上任何待执行 get-notification 调用。
“ChangeDatabase 方法”一节第 147 页	为打开的 ULConnection 更改当前数据库。
“ChangeEncryptionKey 方法”一节第 148 页	**UL Ext.:** 将数据库的加密密钥更改为指定的新密钥。
“ChangePassword 方法”一节第 148 页	将连接字符串中指示的用户口令更改为所提供的新口令。
“Close 方法”一节第 149 页	关闭数据库连接。
“CountUploadRows 方法”一节第 149 页	**UL Ext.:** 返回下次进行同步时需要上载的行数。
“CreateCommand 方法”一节第 151 页	创建并初始化与此连接及其当前事务关联的 ULCommand 对象。您可以使用 ULCommand 的属性来控制其行为。
“CreateNotificationQueue 方法”一节第 152 页	创建事件队列。
“DeclareEvent 方法”一节第 152 页	声明一个指定的事件。
“DestroyNotificationQueue 方法”一节第 153 页	取消事件队列。
EnlistTransaction（继承自 DbConnection）	在指定的事务中征用。
“ExecuteNextSQLPassthroughScript 方法”一节第 154 页	执行下一个待执行的 SQL 直通脚本。
“ExecuteSQLPassthroughScripts 方法”一节第 155 页	执行所有待执行的 SQL 直通脚本。
“ExecuteTable 方法”一节第 155 页	**UL Ext.:** 在 ULTable 中检索数据库表以便直接进行操作。使用表的主键打开（排序）该表。

成员名称	说明
“GetLastDownloadTime 方法”一节第 159 页	**UL Ext.:** 返回指定发布的最近一次下载的时间。
“GetNewUUID 方法”一节第 160 页	**UL Ext.:** 生成一个新的 UUID (System.Guid)。
“GetNotification 方法”一节第 161 页	用于通知或超时的块。返回事件名或空值。
“GetNotificationParameter 方法”一节第 162 页	获取刚刚由 GetNotification() 读取的事件的参数值。
“GetSQLPassthroughScriptCount 方法”一节第 162 页	获取待执行的 SQL 直通脚本的数量。
“GetSchema 方法”一节第 163 页	返回此 DbConnection 的数据源的模式信息。
“GrantConnectTo 方法”一节第 166 页	**UL Ext.:** 授权某个用户 ID 使用指定口令访问 UltraLite 数据库。
“Open 方法”一节第 166 页	使用先前指定的连接字符串打开与某个数据库的连接。
“RegisterForEvent 方法”一节第 167 页	注册一个队列以便从对象获取事件。
“ResetLastDownloadTime 方法”一节第 168 页	**UL Ext.:** 重置最近一次下载的时间。
“RevokeConnectFrom 方法”一节第 168 页	**UL Ext.:** 撤消指定的用户 ID 访问 UltraLite 数据库的权限。
“RollbackPartialDownload 方法”一节第 169 页	**UL Ext.:** 回退部分下载中对数据库的未完成更改。
“SendNotification 方法”一节第 169 页	将通知发送到匹配的队列。返回匹配队列的数量。
“StartSynchronizationDelete 方法”一节第 170 页	**UL Ext.:** 标记此连接所做的所有后续删除以进行同步。
“StopSynchronizationDelete 方法”一节第 171 页	**UL Ext.:** 防止同步删除操作。
“Synchronize 方法”一节第 171 页	**UL Ext.:** 使用当前的 ULConnection.SyncParms 同步数据库。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“TriggerEvent 方法”一节 第 173 页	触发一个事件。返回已发送通知的数量。
“ValidateDatabase 方法”一节 第 173 页	在当前数据库上执行校验。

公共事件

成员名称	说明
“InfoMessage 事件”一节 第 174 页	当 UltraLite.NET 发送有关此连接的警告或信息性消息时发生。
“StateChange 事件”一节 第 175 页	当此连接更改状态时发生。

另请参见

- “ULConnection 类”一节第 131 页
- “RuntimeType 属性”一节第 239 页
- “Open 方法”一节第 166 页
- “Schema 属性”一节第 142 页
- IDbConnection
- IDisposable

ULConnection 构造函数

初始化一个新的“ULConnection 类”一节第 131 页实例。

ULConnection() 构造函数

初始化 ULConnection 对象。对数据库进行任何操作前必须打开该连接。

语法

Visual Basic
Public Sub **New()**

C#
public **ULConnection();**

注释

要使用 UltraLite.NET 的 UltraLite 引擎运行时，请在使用任何其它 UltraLite.NET API 之前将 ULDatabaseManager.RuntimeType 设置为适当的值。

打开 ULConnection 对象之前需要先设置 ULConnection.ConnectionString。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ULConnection 构造函数”一节第 135 页
- “Open 方法”一节第 166 页
- “ConnectionString 属性”一节第 137 页

ULConnection(String) 构造函数

用指定的连接字符串初始化 ULConnection 对象。对数据库进行任何操作前必须打开该连接。

语法

Visual Basic

```
Public Sub New( _
  ByVal connectionString As String _
)
```

C#

```
public  ULConnection(
  string  connectionString
);
```

参数

- **connectionString**　UltraLite.NET 连接字符串。连接字符串是一个以分号分隔的 "关键字-值" 对的列表。

  有关参数的列表，请参见“ConnectionString 属性”一节第 137 页。

注释

要使用 UltraLite.NET 的 UltraLite 引擎运行时，请在使用任何其它 UltraLite.NET API 之前将 ULDatabaseManager.RuntimeType 设置为适当的值。

可以使用 ULConnectionParms 对象提供连接字符串。

示例

以下代码会创建并打开与 Windows Mobile 设备上现有数据库 \UltraLite\MyDatabase.udb 的连接。

```
' Visual Basic
Dim openParms As ULConnectionParms = New ULConnectionParms
openParms.DatabaseOnCE = "\UltraLite\MyDatabase.udb"
Dim conn As ULConnection = _
  New ULConnection( openParms.ToString() )
conn.Open()

// C#
ULConnectionParms openParms = new ULConnectionParms();
openParms.DatabaseOnCE = @"\UltraLite\MyDatabase.udb";
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
ULConnection conn = new ULConnection( openParms.ToString() );
conn.Open();
```

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ULConnection 构造函数”一节第 135 页
- “Open 方法”一节第 166 页
- “ULConnection() 构造函数”一节第 135 页
- “RuntimeType 属性”一节第 239 页
- “ULConnectionParms 类”一节第 177 页
- “ConnectionString 属性”一节第 137 页

INVALID_DATABASE_ID 字段

UL Ext.: 一个数据库 ID 常量，表示尚未设置 ULConnection.DatabaseID。此字段为常量并是只读字段。

语法

Visual Basic
Public Shared **INVALID_DATABASE_ID** As Long

C#
public const long **INVALID_DATABASE_ID**;

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “DatabaseID 属性”一节第 140 页

ConnectionString 属性

指定用于打开 UltraLite.NET 数据库连接的参数。可以使用 ULConnectionParms 对象提供连接字符串。

语法

Visual Basic
Public Overrides Property **ConnectionString** As String

C#
public override string **ConnectionString** { get; set; }

属性值

用于打开此连接的参数，以分号分隔的 "关键字-值" 对列表的形式表示。缺省值为空字符串（无效连接字符串）。

注释

UL Ext.: UltraLite.NET 使用的参数是 UltraLite 数据库所特有的，因此连接字符串与 SQL Anywhere 连接字符串不兼容。有关参数的列表，请参见“UltraLite 连接参数” 《UltraLite - 数据库管理和参考》。

可以用单引号 (') 字符或双引号 (") 字符将参数值括起来，前提是引用内容不包含同类型的引号字符。如果参数值包含分号 (;)、以引号开头或需要前导或尾随空格，则必须用引号将值括起来。

如果不用引号将参数值括起来，则请确保参数值不包含分号 (;)，且以单引号 (') 或双引号 (") 字符开头。值中的前导和尾随空格会被忽略。

缺省情况下会使用 UID=DBA 和 PWD=sql 打开连接。为使数据库更加安全，请更改用户 DBA 的口令，或者创建新的用户（使用 GrantConnectTo）并删除 DBA 用户（使用 RevokeConnectFrom）。

示例

以下代码会创建并打开与 Windows Mobile 设备上现有数据库 *\UltraLite\MyDatabase.udb* 的连接。

```
' Visual Basic
Dim openParms As ULConnectionParms = New ULConnectionParms
openParms.DatabaseOnCE = "\UltraLite\MyDatabase.udb"
Dim conn As ULConnection = New ULConnection
conn.ConnectionString = openParms.ToString()
conn.Open()

// C#
ULConnectionParms openParms = new ULConnectionParms();
openParms.DatabaseOnCE = @"\UltraLite\MyDatabase.udb";
ULConnection conn = new ULConnection();
conn.ConnectionString = openParms.ToString();
conn.Open();
```

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “Open 方法”一节第 166 页
- “ULConnectionParms 类”一节第 177 页
- “GrantConnectTo 方法”一节第 166 页

ConnectionTimeout 属性

UltraLite.NET 不支持此功能。

语法

Visual Basic
Public Overrides Readonly Property **ConnectionTimeout** As Integer

C#
public override int **ConnectionTimeout** { get;}

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

属性值

值始终为 0。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页

DataSource 属性

UltraLite.NET 不支持此功能。

语法

Visual Basic
Public Overrides Readonly Property **DataSource** As String

C#
public override string **DataSource** { get;}

属性值

值始终为空字符串。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页

Database 属性

返回打开连接的数据库的名称。

语法

Visual Basic
Public Overrides Readonly Property **Database** As String

C#
public override string **Database** { get;}

属性值

包含数据库名称的字符串。

注释

在 Windows Mobile 设备上，ULConnection 按照以下顺序查看连接字符串：dbn、ce_file。

在台式机上，ULConnection 按照以下顺序查看连接字符串：dbn、nt_file。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页

DatabaseID 属性

UL Ext.: 指定用于全局自动增量列的数据库 ID 值。

语法

Visual Basic
Public Property **DatabaseID** As Long

C#
public long **DatabaseID** { get; set; }

属性值

当前数据库的数据库 ID 值。

注释

数据库 ID 值必须在 [0,System.UInt32.MaxValue] 范围内。值 ULConnection.INVALID_DATABASE_ID 用于指示还没有为当前数据库设置数据库 ID。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “GetDatabaseProperty 方法”一节第 251 页
- “SetDatabaseOption 方法”一节第 255 页
- UInt32.MaxValue
- “INVALID_DATABASE_ID 字段”一节第 137 页

DatabaseManager 属性

UL Ext.: 提供对单个 ULDatabaseManager 对象的访问。

语法

Visual Basic
Public Shared Readonly Property **DatabaseManager** As ULDatabaseManager

C#
public const ULDatabaseManager **DatabaseManager** { get;}

属性值

对单个 ULDatabaseManager 对象的引用。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ULDatabaseManager 类”一节第 238 页

GlobalAutoIncrementUsage 属性

UL Ext.: 返回已经使用的可用全局自动增量值的百分比。

语法

Visual Basic
Public Readonly Property **GlobalAutoIncrementUsage** As Short

C#
public short **GlobalAutoIncrementUsage** { get;}

属性值

已经使用的可用全局自动增量值的百分比。它是 [0-100]（含 0 和 100）范围内的整数。

注释

如果百分比接近 100，则应用程序应使用 ULConnection.DatabaseID 为全局数据库 ID 设置一个新值。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ULDatabaseManager 类”一节第 238 页
- “DatabaseID 属性”一节第 140 页

LastIdentity 属性

UL Ext.: 返回最近使用的标识值。

语法

Visual Basic
Public Readonly Property **LastIdentity** As UInt64

C#
public ulong **LastIdentity** { get;}

属性值

最近使用的 unsigned long 类型标识值。

注释

最近使用的标识值。此属性等效于以下 SQL Anywhere 语句：

```
SELECT @@identity
```

LastIdentity 在全局自动增量列的上下文中尤为有用。

由于此属性仅允许您确定最近指派的缺省值，因此您应在执行插入语句后尽快检索此值，以避免得到虚假结果。

有时，单个插入语句可能会包括类型为全局自动增量的多个列。在这种情况下，LastIdentity 是生成的缺省值之一，但没有可靠的方法能确定该值来自哪一列。因此，在设计数据库和编写插入语句时，应避免这种情况。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页

Schema 属性

UL Ext.: 提供对此连接相关联的当前数据库模式的访问。

语法

Visual Basic
Public Readonly Property **Schema** As ULDatabaseSchema

C#
public ULDatabaseSchema **Schema** { get;}

属性值

对 ULDatabaseSchema 对象的引用，该对象表示此连接打开时所在数据库的模式。

注释

此属性仅在其连接打开期间才有效。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ULDatabaseSchema 类”一节第 247 页

ServerVersion 属性

UltraLite.NET 不支持此功能。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Overrides Readonly Property **ServerVersion** As String

C#
public override string **ServerVersion** { get;}

属性值

值始终为空字符串。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页

State 属性

返回连接的当前状态。

语法

Visual Basic
Public Overrides Readonly Property **State** As ConnectionState

C#
public override ConnectionState **State** { get;}

属性值

如果连接已打开，则为 System.Data.ConnectionState.Open；如果连接已关闭，则为 System.Data.ConnectionState.Closed。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “StateChange 事件”一节第 175 页
- ConnectionState.Open
- ConnectionState.Closed

SyncParms 属性

UL Ext.: 指定此连接的同步设置。

语法

Visual Basic
Public Readonly Property **SyncParms** As ULSyncParms

C#
public ULSyncParms **SyncParms** { get;}

属性值

对 ULSyncParms 对象的引用，该对象表示此连接用来进行同步的参数。对参数的修改将影响此连接上的下次同步。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “Synchronize() 方法”一节第 171 页
- “SyncResult 属性”一节第 144 页
- “ULSyncParms 类”一节第 478 页

SyncResult 属性

UL Ext.: 返回此连接上次的同步结果。

语法

Visual Basic
Public Readonly Property **SyncResult** As ULSyncResult

C#
public ULSyncResult **SyncResult** { get;}

属性值

对 ULSyncResult 对象的引用，该对象表示此连接上次的同步结果。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “Synchronize() 方法”一节第 171 页
- “SyncParms 属性”一节第 143 页
- “SyncResult 属性”一节第 144 页

BeginTransaction 方法

启动数据库事务。

BeginTransaction() 方法

返回事务对象。将与事务对象关联的命令作为单个事务执行。使用 ULTransaction.Commit 或 ULTransaction.Rollback 终止事务。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Function **BeginTransaction()** As ULTransaction

C#
public ULTransaction **BeginTransaction();**

返回值

一个表示新事务的 ULTransaction 对象。

注释

使用 IsolationLevel.ReadCommitted 创建事务。

要将命令与事务对象相关联，请使用 ULCommand.Transaction 属性。当前事务会自动与 ULConnection.CreateCommand 所创建的命令相关联。

缺省情况下，连接不使用事务，且所有命令均会在执行时自动提交。提交或回退当前事务后，连接即会恢复为自动提交模式和前一个隔离级别，直至下一次调用 BeginTransaction。

与 ADO.NET 的 IsolationLevel 文档相比，UltraLite 的每个隔离级别定义稍有不同。有关详细信息，请参见“UltraLite 隔离级别”一节 《UltraLite - 数据库管理和参考》。

这是 System.Data.IDbConnection.BeginTransaction() 和 System.Data.Common.DbConnection.BeginTransaction() 的强类型版本。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “BeginTransaction 方法”一节第 144 页
- “BeginTransaction(IsolationLevel) 方法”一节第 145 页
- “Commit 方法”一节第 549 页
- “Rollback 方法”一节第 550 页
- “Transaction 属性”一节第 98 页
- “CreateCommand 方法”一节第 151 页
- IDbConnection.BeginTransaction
- DbConnection.BeginTransaction

BeginTransaction(IsolationLevel) 方法

返回具有指定隔离级别的事务对象。将与事务对象关联的命令作为单个事务执行。使用 ULTransaction.Commit 或 ULTransaction.Rollback 终止事务。

语法

Visual Basic
```
Public Function BeginTransaction( _
  ByVal isolationLevel As IsolationLevel _
) As ULTransaction
```

C#
public ULTransaction **BeginTransaction(**
  IsolationLevel *isolationLevel*
);

参数

- **isolationLevel** 事务所需的隔离级别。UltraLite.NET 仅支持 System.Data.IsolationLevel.ReadUncommitted 和 ReadCommitted。

返回值

一个表示新事务的 ULTransaction 对象。

注释

要将命令与事务对象相关联，请使用 ULCommand.Transaction 属性。当前事务会自动与 ULConnection.CreateCommand 所创建的命令相关联。

缺省情况下，连接不使用事务，且所有命令均会在执行时自动提交。提交或回退当前事务后，连接即会恢复为自动提交模式和前一个隔离级别，直至下一次调用 BeginTransaction。

与 ADO.NET 的 IsolationLevel 文档相比，UltraLite 的每个隔离级别定义稍有不同。有关详细信息，请参见“UltraLite 隔离级别”一节 《UltraLite - 数据库管理和参考》。

这是 System.Data.IDbConnection.BeginTransaction(System.Data.IsolationLevel) 和 System.Data.Common.DbConnection.BeginTransaction(System.Data.IsolationLevel) 的强类型版本。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “BeginTransaction 方法”一节第 144 页
- “ULTransaction 类”一节第 547 页
- “BeginTransaction() 方法”一节第 144 页
- “Commit 方法”一节第 549 页
- “Rollback 方法”一节第 550 页
- “Transaction 属性”一节第 98 页
- “CreateCommand 方法”一节第 151 页
- IDbConnection.BeginTransaction
- DbConnection.BeginTransaction
- IsolationLevel

CancelGetNotification 方法

取消与给定名称匹配的所有队列上任何待执行 get-notification 调用。

语法

Visual Basic
Public Function **CancelGetNotification(_**

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
  ByVal queueName As String _
) As Integer

C#
public int CancelGetNotification(
  string  queueName
);
```

参数

- **queueName** 与队列名匹配的表达式。

注释

取消与给定名称匹配的所有队列上任何待执行 get-notification 调用。

返回受影响的队列数（不一定是受阻塞读取的数目）。

另请参见

- “CreateNotificationQueue 方法”一节第 152 页
- “DeclareEvent 方法”一节第 152 页
- “DestroyNotificationQueue 方法”一节第 153 页
- “DeclareEvent 方法”一节第 152 页
- “GetNotification 方法”一节第 161 页
- “RegisterForEvent 方法”一节第 167 页
- “SendNotification 方法”一节第 169 页
- “TriggerEvent 方法”一节第 173 页
- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “GetNotification 方法”一节第 161 页
- “ULException 类”一节第 302 页

ChangeDatabase 方法

为打开的 ULConnection 更改当前数据库。

语法

```
Visual Basic
Public Overrides Sub ChangeDatabase( _
  ByVal connectionString As String _
)

C#
public override void ChangeDatabase(
  string  connectionString
);
```

参数

- **connectionString** 用于打开新数据库连接的完整连接字符串。

注释

即使存在参数错误，也会关闭与当前数据库的连接。

UL Ext.:*connectionString* 为完整的连接字符串，而不是 dbn 或 dbf。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ConnectionString 属性”一节第 137 页

ChangeEncryptionKey 方法

UL Ext.: 将数据库的加密密钥更改为指定的新密钥。

语法

Visual Basic

```
Public Sub ChangeEncryptionKey( _
  ByVal newKey As String _
)
```

C#

```
public void ChangeEncryptionKey(
  string  newKey
);
```

参数

- **newKey**　数据库的新加密密钥。

注释

如果丢失了加密密钥，便无法打开数据库。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “EncryptionKey 属性”一节第 185 页

ChangePassword 方法

将连接字符串中指示的用户口令更改为所提供的新口令。

语法

Visual Basic

```
Public Shared Sub ChangePassword( _
  ByVal connectionString As String, _
  ByVal newPassword As String _
)
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

C#
```
public static void ChangePassword(
   string  connectionString,
   string  newPassword
);
```

参数

- **connectionString** 一个连接字符串，其中包含连接到指定数据库所需的足够信息。连接字符串可以包含用户 ID 和当前口令。
- **newPassword** 要设置的新口令。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页

Close 方法

关闭数据库连接。

语法

Visual Basic
```
Public Overrides Sub Close()
```

C#
```
public override void Close();
```

注释

Close 方法会回退所有待执行事务，然后关闭连接。应用程序可以多次调用 Close。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页

CountUploadRows 方法

UL Ext.: 返回下次进行同步时需要上载的行数。

CountUploadRows(String, UInt32) 方法

UL Ext.: 返回下次进行同步时需要上载的行数。

语法

Visual Basic
```
Public Function CountUploadRows( _
```

```
  ByVal pubs As String, _
  ByVal threshold As UInt32 _
) As UInt32

C#
public uint CountUploadRows(
  string  pubs,
  uint threshold
);
```

参数

- **pubs** 以逗号分隔的要检查行的发布的列表。
- **threshold** 要计数的最大行数，用于限制 CountUploadRows 所用的时间量。0 值对应于最大限制。值 1 决定了是否有需要同步的行。

返回值

需要从指定发布上载的行数。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “CountUploadRows 方法”一节第 149 页
- “CountUploadRows(String, Int64) 方法”一节第 150 页

CountUploadRows(String, Int64) 方法

UL Ext.: 返回下次进行同步时需要上载的行数。

语法

```
Visual Basic
Public Function CountUploadRows( _
  ByVal pubs As String, _
  ByVal threshold As Long _
) As Long

C#
public long CountUploadRows(
  string  pubs,
  long threshold
);
```

参数

- **pubs** 以逗号分隔的要检查行的发布的列表。
- **threshold** 要计数的最大行数，用于限制 CountUploadRows 所用的时间量。0 值对应于最大限制。值 1 决定了是否有需要同步的行。阈值必须在 [0,0x0fffffff] 范围内。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

返回值

需要从指定发布上载的行数。

注释

此方法是为本来不支持 System.UInt32 类型的语言提供的。如果应用程序支持，也可使用此方法的其它形式。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “CountUploadRows 方法”一节第 149 页
- “CountUploadRows(String, UInt32) 方法”一节第 149 页

CreateCommand 方法

创建并初始化与此连接及其当前事务关联的 ULCommand 对象。您可以使用 ULCommand 的属性来控制其行为。

语法

Visual Basic
Public Function **CreateCommand()** As ULCommand

C#
public ULCommand **CreateCommand();**

返回值

新的 ULCommand 对象。

注释

必须先设置 ULCommand.CommandText，才能执行该命令。

这是 System.Data.IDbConnection.CreateCommand 和 System.Data.Common.DbConnection.CreateCommand() 的强类型版本。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ULCommand 类”一节第 86 页
- “CommandText 属性”一节第 93 页
- IDbConnection.CreateCommand
- DbConnection.CreateCommand

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

CreateNotificationQueue 方法

创建事件队列。

语法

Visual Basic

```
Public Sub CreateNotificationQueue( _
  ByVal queueName As String, _
  ByVal parameters As String _
)
```

C#

```
public void CreateNotificationQueue(
  string  queueName,
  string  parameters
);
```

参数

- **queueName** 新队列的名称。
- **parameters** 创建参数；目前未使用，设置为 NULL。

注释

创建此连接的事件通知队列。队列名的使用范围仅限于每个连接，所以不同的连接可以创建具有相同名称的队列。发送事件通知时，数据库中具有匹配名称的所有队列都接收通知（单独实例）。名称不区分大小写。如果没指定任何队列，则调用 RegisterForEvent() 时按需为每个连接创建缺省队列。

另请参见

- “CancelGetNotification 方法”一节第 146 页
- “DeclareEvent 方法”一节第 152 页
- “DestroyNotificationQueue 方法”一节第 153 页
- “DeclareEvent 方法”一节第 152 页
- “GetNotification 方法”一节第 161 页
- “RegisterForEvent 方法”一节第 167 页
- “SendNotification 方法”一节第 169 页
- “TriggerEvent 方法”一节第 173 页
- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “DestroyNotificationQueue 方法”一节第 153 页
- “ULException 类”一节第 302 页

DeclareEvent 方法

声明一个指定的事件。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
```
Public Sub DeclareEvent( _
  ByVal eventName As String _
)
```

C#
```
public void DeclareEvent(
  string  eventName
);
```

参数

- **eventName**　事件名称。

注释

声明稍后可以进行注册和触发的事件。UltraLite 预定义一些由对数据库或环境的操作触发的系统事件。事件的名称必须是唯一的。名称不区分大小写。如果名称已被使用或无效则抛出错误。

另请参见

- “CancelGetNotification 方法”一节第 146 页
- “CreateNotificationQueue 方法”一节第 152 页
- “DestroyNotificationQueue 方法”一节第 153 页
- “DeclareEvent 方法”一节第 152 页
- “GetNotification 方法”一节第 161 页
- “RegisterForEvent 方法”一节第 167 页
- “SendNotification 方法”一节第 169 页
- “TriggerEvent 方法”一节第 173 页
- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “CreateNotificationQueue 方法”一节第 152 页
- “ULException 类”一节第 302 页

DestroyNotificationQueue 方法

取消事件队列。

语法

Visual Basic
```
Public Sub DestroyNotificationQueue( _
  ByVal queueName As String _
)
```

C#
```
public void DestroyNotificationQueue(
  string  queueName
);
```

参数

- **queueName** 队列的名称。

注释

取消给定的事件通知队列。如果未读通知仍然在队列中，则会发出警告。未读通知将被放弃。如果已创建连接的缺省事件队列，则在连接关闭时会将其取消。

另请参见

- “CancelGetNotification 方法”一节第 146 页
- “CreateNotificationQueue 方法”一节第 152 页
- “DeclareEvent 方法”一节第 152 页
- “GetNotification 方法”一节第 161 页
- “RegisterForEvent 方法”一节第 167 页
- “SendNotification 方法”一节第 169 页
- “TriggerEvent 方法”一节第 173 页
- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “CreateNotificationQueue 方法”一节第 152 页
- “ULException 类”一节第 302 页

ExecuteNextSQLPassthroughScript 方法

执行下一个待执行的 SQL 直通脚本。

语法

Visual Basic
Public Sub **ExecuteNextSQLPassthroughScript()**

C#
public void **ExecuteNextSQLPassthroughScript();**

注释

同步可能导致同时下载多个 SQL 直通脚本。

另请参见

- “ExecuteSQLPassthroughScripts 方法”一节第 155 页
- “GetSQLPassthroughScriptCount 方法”一节第 162 页
- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “GetSQLPassthroughScriptCount 方法”一节第 162 页
- “ExecuteSQLPassthroughScripts 方法”一节第 155 页
- “ULException 类”一节第 302 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ExecuteSQLPassthroughScripts 方法

执行所有待执行的 SQL 直通脚本。

语法

Visual Basic
Public Sub **ExecuteSQLPassthroughScripts()**

C#
public void **ExecuteSQLPassthroughScripts();**

注释

同步可能导致同时下载多个 SQL 直通脚本。此方法用于执行所有待执行的脚本。可使用 ULDatabaseManager.SetGlobalListener() 方法跟踪脚本进程。

另请参见

- “ExecuteNextSQLPassthroughScript 方法”一节第 154 页
- “GetSQLPassthroughScriptCount 方法”一节第 162 页
- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “GetSQLPassthroughScriptCount 方法”一节第 162 页
- “ExecuteNextSQLPassthroughScript 方法”一节第 154 页
- “ULException 类”一节第 302 页
- “SetGlobalListener 方法”一节第 243 页

ExecuteTable 方法

UL Ext.: 在 ULTable 中检索数据库表以便直接进行操作。使用表的主键打开（排序）该表。

ExecuteTable(String) 方法

UL Ext.: 在 ULTable 中检索数据库表以便直接进行操作。使用表的主键打开（排序）该表。

语法

Visual Basic
```
Public Function ExecuteTable( _
  ByVal tableName As String _
) As ULTable
```

C#
```
public ULTable ExecuteTable(
  string  tableName
);
```

参数

- **tableName**　要打开的表的名称。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

返回值

ULTable 对象形式的表。

注释

此方法为 ULCommand.ExecuteTable() 方法的快捷方式，它不需要 ULCommand 实例。提供此方法是为了帮助用户从较早的 UltraLite.NET 版本进行移植（它替换了 iAnywhere.UltraLite.Connection.GetTable() 和 iAnywhere.UltraLite.Table.Open()）。

示例

以下代码会使用表的主键打开名为 MyTable 的表。它假定已打开了一个名为 conn 的 ULConnection 实例。

```
' Visual Basic
Dim t As ULTable = conn.ExecuteTable("MyTable")
' The line above is equivalent to
' Dim cmd As ULCommand = conn.CreateCommand()
' cmd.CommandText = "MyTable"
' cmd.CommandType = CommandType.TableDirect
' Dim t As ULTable = cmd.ExecuteTable()
' cmd.Dispose()


// C#
ULTable t = conn.ExecuteTable("MyTable");
// The line above is equivalent to
// ULTable t;
// using(ULCommand cmd = conn.CreateCommand())
// {
//     cmd.CommandText = "MyTable";
//     cmd.CommandType = CommandType.TableDirect;
//     t = cmd.ExecuteTable();
// }
```

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ExecuteTable 方法”一节第 155 页
- “ExecuteTable(String, String) 方法”一节第 156 页
- “ExecuteTable(String, String, CommandBehavior) 方法”一节第 158 页
- “ULTable 类”一节第 511 页
- “ExecuteTable() 方法”一节第 118 页
- “ULCommand 类”一节第 86 页

ExecuteTable(String, String) 方法

UL Ext.: 在 ULTable 中检索数据库表以便直接进行操作。使用指定的索引打开（排序）该表。

语法

Visual Basic
Public Function **ExecuteTable(_**

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
  ByVal tableName As String, _
  ByVal indexName As String _
) As ULTable

C#
public ULTable ExecuteTable(
  string  tableName,
  string  indexName
);
```

参数

- **tableName** 要打开的表的名称。
- **indexName** 用于打开（排序）表的索引名称。

返回值

ULTable 对象形式的表。

注释

此方法为 ULCommand.ExecuteTable() 方法的快捷方式，它不需要 ULCommand 实例。提供此方法是为了帮助用户从较早的 UltraLite.NET 版本进行移植（它替换了 iAnywhere.UltraLite.Connection.GetTable() 和 iAnywhere.UltraLite.Table.Open()）。

示例

以下代码会使用名为 MyIndex 的索引打开名为 MyTable 的表。它假定已打开了一个名为 conn 的 ULConnection 实例。

```
' Visual Basic
Dim t As ULTable = conn.ExecuteTable("MyTable", "MyIndex")
' The line above is equivalent to
' Dim cmd As ULCommand = conn.CreateCommand()
' cmd.CommandText = "MyTable"
' cmd.IndexName = "MyIndex"
' cmd.CommandType = CommandType.TableDirect
' Dim t As ULTable = cmd.ExecuteTable()
' cmd.Dispose()


// C#
ULTable t = conn.ExecuteTable("MyTable", "MyIndex");
// The line above is equivalent to
// ULTable t;
// using(ULCommand cmd = conn.CreateCommand())
// {
//     cmd.CommandText = "MyTable";
//     cmd.IndexName = "MyIndex";
//     cmd.CommandType = CommandType.TableDirect;
//     t = cmd.ExecuteTable();
// }
```

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ExecuteTable 方法”一节第 155 页
- “ExecuteTable(String) 方法”一节第 155 页
- “ExecuteTable(String, String, CommandBehavior) 方法”一节第 158 页
- “ULTable 类”一节第 511 页
- “ExecuteTable() 方法”一节第 118 页
- “ULCommand 类”一节第 86 页

ExecuteTable(String, String, CommandBehavior) 方法

UL Ext.: 以指定的命令行为检索数据库表以便直接进行操作。使用指定的索引打开（排序）该表。

语法

Visual Basic

```
Public Function ExecuteTable( _
  ByVal tableName As String, _
  ByVal indexName As String, _
  ByVal cmdBehavior As CommandBehavior _
) As ULTable
```

C#

```
public ULTable ExecuteTable(
  string  tableName,
  string  indexName,
  CommandBehavior cmdBehavior
);
```

参数

- **tableName** 要打开的表的名称。
- **indexName** 用于打开（排序）表的索引名称。
- **cmdBehavior** 用于描述查询结果及其对连接影响的 System.Data.CommandBehavior 标志的逐位组合。UltraLite.NET 仅支持 System.Data.CommandBehavior.Default、System.Data.CommandBehavior.CloseConnection 和 System.Data.CommandBehavior.SchemaOnly 标志。

返回值

ULTable 对象形式的表。

注释

此方法为 ULCommand.ExecuteTable(System.Data.CommandBehavior) 的快捷方式，它不需要 ULCommand 实例。提供此方法是为了帮助用户从较早的 UltraLite.NET 版本进行移植（它替换了 iAnywhere.UltraLite.Connection.GetTable() 和 iAnywhere.UltraLite.Table.Open()）。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

示例

以下代码会使用名为 MyIndex 的索引打开名为 MyTable 的表。它假定已打开了一个名为 conn 的 ULConnection 实例。

```
' Visual Basic
Dim t As ULTable = conn.ExecuteTable( _
    "MyTable", "MyIndex", CommandBehavior.Default _
  )
' The line above is equivalent to
' Dim cmd As ULCommand = conn.CreateCommand()
' cmd.CommandText = "MyTable"
' cmd.IndexName = "MyIndex"
' cmd.CommandType = CommandType.TableDirect
' Dim t As ULTable = cmd.ExecuteTable(CommandBehavior.Default)
' cmd.Dispose()


// C#
ULTable t = conn.ExecuteTable(
    "MyTable", "MyIndex", CommandBehavior.Default
  );
// The line above is equivalent to
// ULTable t;
// using(ULCommand cmd = conn.CreateCommand())
// {
//     cmd.CommandText = "MyTable";
//     cmd.IndexName = "MyIndex";
//     cmd.CommandType = CommandType.TableDirect;
//     t = cmd.ExecuteTable(CommandBehavior.Default);
// }
```

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ExecuteTable 方法”一节第 155 页
- “ExecuteTable(String) 方法”一节第 155 页
- “ExecuteTable(String, String) 方法”一节第 156 页
- “ULTable 类”一节第 511 页
- “ExecuteTable(CommandBehavior) 方法”一节第 119 页
- “ULCommand 类”一节第 86 页
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly

GetLastDownloadTime 方法

UL Ext.: 返回指定发布的最近一次下载的时间。

语法

Visual Basic
Public Function **GetLastDownloadTime(_**

```
  ByVal publication As String _
) As Date

C#
public DateTime GetLastDownloadTime(
  string  publication
);
```

参数

- **发布** 待检查的发布。有关详细信息，请参见“ULPublicationSchema 类”一节第 391 页。

返回值

上次的下载时间戳。

注释

参数 *publication* 为待检查的发布名称。如果使用了特殊常量 ULPublicationSchema.SYNC_ALL_DB，则返回最近一次下载整个数据库的时间。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ResetLastDownloadTime 方法”一节第 168 页
- “SYNC_ALL_DB 字段”一节第 392 页

GetNewUUID 方法

UL Ext.: 生成一个新的 UUID (System.Guid)。

语法

```
Visual Basic
Public Function GetNewUUID() As Guid

C#
public Guid GetNewUUID();
```

返回值

System.Guid 形式的新 UUID。

注释

这里提供此方法是因为 .NET Compact Framework 中未包括它。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- Guid

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

GetNotification 方法

用于通知或超时的块。返回事件名或空值。

语法

Visual Basic

```
Public Function GetNotification( _
  ByVal queueName As String, _
  ByVal wait_ms As Integer _
) As String
```

C#

```
public string  GetNotification(
  string  queueName,
  int wait_ms
);
```

参数

- **queueName** 将对其等待的队列的名称。
- **wait_ms** 等待的时间，以毫秒为单位。用 System.Threading.Timeout.Infinite (-1) 表示无限期的等待。

注释

读取事件通知。此调用将阻塞直到收到通知或给定等待时期到期。要进行无限期的等待，对于 \p wait_ms 传递 System.Threading.Timeout.Infinite。要取消等待，则将另一个通知发送到给定队列或使用 CancelGetNotification。读取通知后，使用 ReadNotificationParameter() 来检索其它参数。

如果等待期过期或等待被取消则返回空值；否则，返回事件名。

另请参见

- “CancelGetNotification 方法”一节第 146 页
- “CreateNotificationQueue 方法”一节第 152 页
- “DeclareEvent 方法”一节第 152 页
- “DestroyNotificationQueue 方法”一节第 153 页
- “DeclareEvent 方法”一节第 152 页
- “RegisterForEvent 方法”一节第 167 页
- “SendNotification 方法”一节第 169 页
- “TriggerEvent 方法”一节第 173 页
- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “SendNotification 方法”一节第 169 页
- “GetNotificationParameter 方法”一节第 162 页
- “CancelGetNotification 方法”一节第 146 页
- “ULException 类”一节第 302 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

GetNotificationParameter 方法

获取刚刚由 GetNotification() 读取的事件的参数值。

语法

Visual Basic

```
Public Function GetNotificationParameter( _
  ByVal queueName As String, _
  ByVal parameterName As String _
) As String
```

C#

```
public string  GetNotificationParameter(
  string  queueName,
  string  parameterName
);
```

参数

- **queueName** 将对其等待的队列的名称。
- **parameterName** 其值应返回的参数的名称。

注释

获取刚刚由 ULGetNotification() 读取的事件通知的参数。只有给定队列中来自最近读取的通知的参数可用。

如果找到参数，则返回参数值，否则返回空值。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “GetNotification 方法”一节第 161 页
- “ULException 类”一节第 302 页

GetSQLPassthroughScriptCount 方法

获取待执行的 SQL 直通脚本的数量。

语法

Visual Basic

```
Public Function GetSQLPassthroughScriptCount() As Long
```

C#

```
public long GetSQLPassthroughScriptCount();
```

注释

同步可能导致同时下载多个 SQL 直通脚本。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ExecuteNextSQLPassthroughScript 方法”一节第 154 页
- “ExecuteSQLPassthroughScripts 方法”一节第 155 页
- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ExecuteNextSQLPassthroughScript 方法”一节第 154 页
- “ExecuteSQLPassthroughScripts 方法”一节第 155 页
- “ULException 类”一节第 302 页

GetSchema 方法

返回此 DbConnection 的数据源的模式信息。

GetSchema() 方法

返回所支持模式集合的列表。

语法

Visual Basic
Public Overrides Function **GetSchema()** As DataTable

C#
public override DataTable **GetSchema();**

注释

有关可用元数据的说明，请参见“GetSchema(String, String[]) 方法”一节第 164 页。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “GetSchema 方法”一节第 163 页

GetSchema(String) 方法

返回此 ULConnection 的指定元数据集合的信息。

语法

Visual Basic
Public Overrides Function **GetSchema(_**
  ByVal *collection* As String _
) As DataTable

C#
public override DataTable **GetSchema(**

```
  string  collection
);
```

参数

- **collection** 元数据集合的名称。如果未提供，则会使用 MetaDataCollections。

注释

有关可用元数据的说明，请参见“GetSchema(String, String[]) 方法”一节第 164 页。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “GetSchema 方法”一节第 163 页
- “ULConnection 类”一节第 131 页

GetSchema(String, String[]) 方法

返回此 ULConnection 的数据源模式信息，并且如果已指定，则使用指定的模式名称字符串和指定的限制值字符串数组。

语法

Visual Basic

```
Public Overrides Function GetSchema( _
  ByVal collection As String, _
  ByVal restrictions As String() _
) As DataTable
```

C#

```
public override DataTable GetSchema(
  string  collection,
  string [] restrictions
);
```

参数

- **collection** 元数据集合的名称。如果未提供，则会使用 MetaDataCollections。
- **restrictions** 所请求模式的一组限制值。

返回值

包含模式信息的 DataTable。

注释

此方法用于查询数据库以获取各种元数据。每种类型的元数据均被赋予了一个集合名称，必须传递该名称方可接收该数据。缺省集合名称为 MetaDataCollections。

通过不使用任何参数或使用模式集合名称 **MetaDataCollections** 调用 GetSchema 方法，可以查询 .NET 数据提供程序以确定所支持的模式集合列表。这会返回一个 DataTable，其中含有所支持

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

模式集合的列表 (CollectionName)、每个集合所支持的限制数 (NumberOfRestrictions)，以及它们所使用的标识符部分的个数 (NumberOfIdentifierParts)。

集合	元数据
Columns	返回有关数据库中所有列的信息。
DataSourceInformation	返回有关数据库提供程序的信息。
DataTypes	返回所支持数据类型的列表。
ForeignKeys	返回有关数据库中所有外键的信息。
IndexColumns	返回有关数据库中所有索引列的信息。
索引	返回有关数据库中所有索引的信息。
MetaDataCollections	返回所有集合名称的列表。
Publications	返回有关数据库中所有发布的信息。
ReservedWords	返回 UltraLite 所用保留字的列表。
Restrictions	返回有关 GetSchema 中所用限制的信息。
Tables	返回有关数据库中所有表的信息。

这些集合名称也可作为只读属性在 ULMetaDataCollectionNames 类中使用。

可通过在调用 GetSchema 时指定限制数组来过滤返回的结果。

通过以下调用可以查询每个集合可用的限制：

```
GetSchema( "Restrictions" )
```

如果集合需要四个限制，则限制参数必须为 NULL 或具有四个值的字符串。

要根据特定限制进行过滤，请将用以过滤的字符串置于其在数组中的适当位置，并将任何未使用的位置均保留为 NULL。例如，Tables 集合具有三个限制：Table、TableType、SyncType。

要过滤 Table 集合：

GetSchema("Tables", new string[ ] { "my_table", NULL, NULL }) 返回有关名为 my_table 的所有表的信息。

GetSchema("Tables", new string[ ] { NULL, "User", NULL }) 返回有关所有用户表的信息。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “GetSchema 方法”一节第 163 页
- “ULConnection 类”一节第 131 页
- “ULMetaDataCollectionNames 类”一节第 345 页

GrantConnectTo 方法

UL Ext.: 授权某个用户 ID 使用指定口令访问 UltraLite 数据库。

语法

Visual Basic

```
Public Sub GrantConnectTo( _
  ByVal uid As String, _
  ByVal pwd As String _
)
```

C#

```
public void GrantConnectTo(
  string  uid,
  string  pwd
);
```

参数

- **uid** 获得数据库访问权限的用户 ID。用户 ID 的最大长度为 16 个字符。
- **pwd** 与用户 ID 关联的口令。它的最大长度为 16 个字符。

注释

如果指定了现有的用户 ID，则此函数更新该用户的口令。UltraLite 最多支持 4 个用户。只有在连接打开时启用了用户验证的情况下，才会启用此方法。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “UserID 属性”一节第 186 页
- “Password 属性”一节第 185 页
- “ConnectionString 属性”一节第 137 页

Open 方法

使用先前指定的连接字符串打开与某个数据库的连接。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Overrides Sub **Open()**

C#
public override void **Open();**

注释

您应该在完成与该连接有关的工作后，显式关闭或释放该连接。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ConnectionString 属性”一节第 137 页
- “State 属性”一节第 143 页

RegisterForEvent 方法

注册一个队列以便从对象获取事件。

语法

Visual Basic
```
Public Sub RegisterForEvent( _
  ByVal eventName As String, _
  ByVal objectName As String, _
  ByVal queueName As String, _
  ByVal registerNotUnReg As Boolean _
)
```

C#
```
public void RegisterForEvent(
  string  eventName,
  string  objectName,
  string  queueName,
  bool registerNotUnReg
);
```

参数

- **eventName**　事件名称。
- **objectName**　应用事件的对象名称。例如，表名。
- **queueName**　要使用的事件队列名。
- **registerNotUnReg**　true 表示注册，false 表示注销。

注释

注册（队列）以便接收事件的通知。如果不提供队列名称，则表示采用缺省连接队列，如果需要，可以创建缺省连接队列。某些系统事件允许指定应用事件的对象名称。例如，TableModified 事件

可以指定表名称。与 SendNotification() 不同，仅特定的注册队列会收到事件的通知 - 不同连接中同名的其它队列则不会收到通知（除非它们也经过显式注册）。若队列或事件不存在，则抛出错误

另请参见

- “CancelGetNotification 方法”一节第 146 页
- “CreateNotificationQueue 方法”一节第 152 页
- “DeclareEvent 方法”一节第 152 页
- “DestroyNotificationQueue 方法”一节第 153 页
- “DeclareEvent 方法”一节第 152 页
- “GetNotification 方法”一节第 161 页
- “SendNotification 方法”一节第 169 页
- “TriggerEvent 方法”一节第 173 页
- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “DeclareEvent 方法”一节第 152 页
- “CreateNotificationQueue 方法”一节第 152 页
- “ULException 类”一节第 302 页

ResetLastDownloadTime 方法

UL Ext.: 重置最近一次下载的时间。

语法

Visual Basic

```
Public Sub ResetLastDownloadTime( _
  ByVal pubs As String _
)
```

C#

```
public void ResetLastDownloadTime(
  string  pubs
);
```

参数

- **pubs** 要重置的发布集。有关详细信息，请参见“ULPublicationSchema 类”一节第 391 页。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “GetLastDownloadTime 方法”一节第 159 页

RevokeConnectFrom 方法

UL Ext.: 撤消指定的用户 ID 访问 UltraLite 数据库的权限。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
```
Public Sub RevokeConnectFrom( _
  ByVal uid As String _
)
```

C#
```
public void RevokeConnectFrom(
  string  uid
);
```

参数

- **uid**　要撤消其数据库访问权限的用户 ID。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “GrantConnectTo 方法”一节第 166 页

RollbackPartialDownload 方法

UL Ext.: 回退部分下载中对数据库的未完成更改。

语法

Visual Basic
```
Public Sub RollbackPartialDownload()
```

C#
```
public void RollbackPartialDownload();
```

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “KeepPartialDownload 属性”一节第 482 页
- “ResumePartialDownload 属性”一节第 485 页

SendNotification 方法

将通知发送到匹配的队列。返回匹配队列的数量。

语法

Visual Basic
```
Public Function SendNotification( _
  ByVal queueName As String, _
  ByVal eventName As String, _
```

```
  ByVal parameters As String _
) As Integer
```

C#

```
public int SendNotification(
  string  queueName,
  string  eventName,
  string  parameters
);
```

参数

- **queueName** 要使用的事件队列名。
- **eventName** 事件名称。
- **parameters** 要传送的参数。

注释

将通知发送到与给定名称匹配的所有队列（包括当前连接中任何此类队列）。此调用不会阻塞。使用特殊队列名称 "*" 发送到所有队列。

返回已发送通知的数目（匹配队列的数目）。

另请参见

- “CancelGetNotification 方法”一节第 146 页
- “CreateNotificationQueue 方法”一节第 152 页
- “DeclareEvent 方法”一节第 152 页
- “DestroyNotificationQueue 方法”一节第 153 页
- “DeclareEvent 方法”一节第 152 页
- “GetNotification 方法”一节第 161 页
- “RegisterForEvent 方法”一节第 167 页
- “TriggerEvent 方法”一节第 173 页
- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “DeclareEvent 方法”一节第 152 页
- “RegisterForEvent 方法”一节第 167 页
- “ULException 类”一节第 302 页

StartSynchronizationDelete 方法

UL Ext.: 标记此连接所做的所有后续删除以进行同步。

语法

Visual Basic

```
Public Sub StartSynchronizationDelete()
```

C#

```
public void StartSynchronizationDelete();
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

调用此函数时，所有删除操作都会再次进行同步，从而导致从 UltraLite 数据库删除的行也会从统一数据库中删除。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “StopSynchronizationDelete 方法”一节第 171 页
- “Truncate 方法”一节第 531 页

StopSynchronizationDelete 方法

UL Ext.: 防止同步删除操作。

语法

Visual Basic
Public Sub **StopSynchronizationDelete()**

C#
public void **StopSynchronizationDelete();**

注释

当要删除 UltraLite 数据库中的旧信息以节省空间而不删除统一数据库中的相应信息时，此方法很有用。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “StartSynchronizationDelete 方法”一节第 170 页

Synchronize 方法

UL Ext.: 使用当前的 ULConnection.SyncParms 同步数据库。

Synchronize() 方法

UL Ext.: 使用当前的 ULConnection.SyncParms 同步数据库。

语法

Visual Basic
Public Sub **Synchronize()**

C#
public void **Synchronize();**

注释

在此连接的 ULConnection.SyncResult 中会报告详细的结果状态。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “Synchronize 方法”一节第 171 页
- “Synchronize(ULSyncProgressListener) 方法”一节第 172 页
- “SyncParms 属性”一节第 143 页
- “SyncResult 属性”一节第 144 页

Synchronize(ULSyncProgressListener) 方法

UL Ext.: 使用当前的 ULConnection.SyncParms 将数据库与发布到指定监听器的进度事件同步。

语法

Visual Basic

```
Public Sub Synchronize( _
  ByVal listener As ULSyncProgressListener _
)
```

C#

```
public void Synchronize(
  ULSyncProgressListener listener
);
```

参数

- **listener** 接收同步进度事件的对象。

注释

同步过程中的错误将作为 ULSyncProgressState.STATE_ERROR 事件发布，然后作为 ULExceptions 抛出。

在此连接的 ULConnection.SyncResult 中将会报告详细的结果状态。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “Synchronize 方法”一节第 171 页
- “ULSyncProgressListener 接口”一节第 501 页
- “Synchronize() 方法”一节第 171 页
- “SyncParms 属性”一节第 143 页
- “ULSyncProgressState 枚举”一节第 503 页
- “ULException 类”一节第 302 页
- “SyncResult 属性”一节第 144 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

TriggerEvent 方法

触发一个事件。返回已发送通知的数量。

语法

Visual Basic

```
Public Function TriggerEvent( _
  ByVal eventName As String, _
  ByVal parameters As String _
) As Integer
```

C#

```
public int TriggerEvent(
  string  eventName,
  string  parameters
);
```

参数

- **eventName** 待触发的事件名称。
- **parameters** 要传送的参数。

注释

触发事件（并将通知发送到所有已注册的队列）。

返回已发送事件通知的数目。

另请参见

- “CancelGetNotification 方法”一节第 146 页
- “CreateNotificationQueue 方法”一节第 152 页
- “DeclareEvent 方法”一节第 152 页
- “DestroyNotificationQueue 方法”一节第 153 页
- “DeclareEvent 方法”一节第 152 页
- “GetNotification 方法”一节第 161 页
- “RegisterForEvent 方法”一节第 167 页
- “SendNotification 方法”一节第 169 页
- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “DeclareEvent 方法”一节第 152 页
- “RegisterForEvent 方法”一节第 167 页
- “ULException 类”一节第 302 页

ValidateDatabase 方法

在当前数据库上执行校验。

语法

Visual Basic
```
Public Sub ValidateDatabase( _
  ByVal how As ULDBValid, _
  ByVal tableName As String _
)
```

C#
```
public void ValidateDatabase(
  ULDBValid how,
  string  tableName
);
```

参数

- **how** 如何校验数据库。有关详细信息，请参见“ULDBValid 枚举”一节第 301 页。
- **tableName** 如果为空值（在 Visual Basic 中为 Nothing），则校验整个数据库；否则，只校验指定的表。

示例

以下代码校验当前数据库

```
' Visual Basic
conn.ValidateDatabase( iAnywhere.Data.UltraLite.ULVF_INDEX, Nothing )


// C#
conn.ValidateDatabase( iAnywhere.Data.UltraLite.ULVF_INDEX, null )
```

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ValidateDatabase 方法”一节第 245 页
- “ULDBValid 枚举”一节第 301 页

InfoMessage 事件

当 UltraLite.NET 发送有关此连接的警告或信息性消息时发生。

语法

Visual Basic
```
Public Event InfoMessage As ULInfoMessageEventHandler
```

C#
```
public event ULInfoMessageEventHandler InfoMessage;
```

注释

要处理 UltraLite.NET 警告或信息性消息，必须创建一个 ULInfoMessageEventHandler 委派，并将其附加到此事件。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

示例

以下代码定义了一个信息性消息事件处理程序。

```
' Visual Basic
Private Sub MyInfoMessageHandler( _
      obj As Object, args As ULInfoMessageEventArgs _
    )
  System.Console.WriteLine( _
      "InfoMesageHandler: " + args.NativeError + ", " _
      + args.Message _
    )
End Sub

// C#
private void MyInfoMessageHandler(
      object obj, ULInfoMessageEventArgs args
    )
{
  System.Console.WriteLine(
      "InfoMesageHandler: " + args.NativeError + ", "
      + args.Message
    );
}
```

以下代码会将 MyInfoMessageHandler 添加至名为 conn 的连接。

```
' Visual Basic
AddHandler conn.InfoMessage, AddressOf MyInfoMessageHandler

// C#
conn.InfoMessage +=
  new ULInfoMessageEventHandler(MyInfoMessageHandler);
```

事件数据

- **NativeError** 与数据库返回的信息性消息或警告相对应的 SQL 代码。
- **Message** 数据库返回的信息性或警告消息字符串。
- **Source** 返回消息的 ADO.NET 数据提供程序的名称。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- “ULInfoMessageEventHandler 委派”一节第 344 页

StateChange 事件

当此连接更改状态时发生。

语法

Visual Basic
Public Overrides Event **StateChange** As StateChangeEventHandler

C#
public event override StateChangeEventHandler **StateChange**;

注释

要处理状态更改消息，必须创建一个 System.Data.StateChangeEventHandler 委派，并将其附加到此事件。

示例

以下代码定义了一个状态更改事件处理程序。

```
' Visual Basic
Private Sub MyStateHandler( _
      obj As Object, args As StateChangeEventArgs _
    )
  System.Console.WriteLine( _
      "StateHandler: " + args.OriginalState + " to " _
      + args.CurrentState _
    )
End Sub

// C#
private void MyStateHandler(
      object obj, StateChangeEventArgs args
    )
{
  System.Console.WriteLine(
      "StateHandler: " + args.OriginalState + " to "
      + args.CurrentState
    );
}
```

以下代码会将 MyStateHandler 添加至名为 conn 的连接。

```
' Visual Basic
AddHandler conn.StateChange, AddressOf MyStateHandler

// C#
conn.StateChange += new StateChangeEventHandler(MyStateHandler);
```

事件数据

- **CurrentState** 获取连接的新状态。触发事件时，连接对象将已处于新状态。
- **OriginalState** 获取连接的原始状态。

另请参见

- “ULConnection 类”一节第 131 页
- “ULConnection 成员”一节第 131 页
- StateChangeEventHandler

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULConnectionParms 类

UL Ext.: 生成用于打开与 UltraLite 数据库连接的连接字符串。常用连接参数为 ULConnectionParms 对象上的各个属性。

语法

Visual Basic
Public Class **ULConnectionParms**
  Inherits Component

C#
public class **ULConnectionParms**: Component

注释

ULConnectionParms 对象用于指定打开连接 (ULConnection.Open) 或丢弃数据库 (ULDatabaseManager.DropDatabase) 时所需的参数。

前导和尾随空格在所有的值中均会被忽略。值不得包含前导或尾随空格，或是分号 (;)，也不得以单引号 (') 或双引号 (") 开头。

构建连接字符串时，需要标识数据库并指定任何可选的连接设置。通过在 ULConnectionParms 对象上设置相应属性来提供所有连接参数后，即可使用 ULConnectionParms.ToString 创建连接字符串。结果字符串被 ULConnection(String) 构造函数用来创建新的 ULConnection，或用来设置现有 ULConnection 对象的 ULConnection.ConnectionString。

Identifying the database

每个实例均包含特定于平台的指向数据库的路径。仅使用与执行平台对应的值。例如，在下面的代码中，将在 Windows Mobile 上使用 \UltraLite\mydb1.udb 路径，而在其它平台上则使用 mydb2.db。

```
' Visual Basic
Dim dbName As ULConnectionParms = new ULConnectionParms
dbName.DatabaseOnCE = "\UltraLite\mydb1.udb"
dbName.DatabaseOnDesktop = "somedir\mydb2.udb"

// C#
ULConnectionParms dbName = new ULConnectionParms();
dbName.DatabaseOnCE = "\\UltraLite\\mydb1.udb";
dbName.DatabaseOnDesktop = @"somedir\mydb2.udb";
```

对于 UltraLite 数据库文件，建议使用扩展名 .udb。在 Windows Mobile 设备上，缺省数据库为 \UltraLiteDB\ulstore.udb。在其它 Windows 平台上，缺省数据库为 ulstore.udb。在 C# 中，必须对路径中的任何反斜线字符进行转义，或是在字符串文字两边加上 @。

如果使用多个数据库，则必须为每个数据库指定一个数据库名。有关详细信息，请参见“AdditionalParms 属性”一节第 180 页。

Optional connection settings

视应用程序的需要和数据库的创建方式而定，可能需要提供非缺省的 ULConnectionParms.UserID 和 ULConnectionParms.Password、数据库 ULConnectionParms.EncryptionKey 以及连接 ULConnectionParms.CacheSize。如果应用程序使用多个连接，则应为每个连接提供唯一的 ULConnectionParms.ConnectionName。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

数据库由一个已通过验证的用户 DBA 来创建，其初始口令为 sql。缺省情况下会使用用户 ID DBA 和口令 sql 打开连接。要禁用缺省用户，请使用 ULConnection.RevokeConnectFrom。要添加用户或更改用户口令，请使用 ULConnection.GrantConnectTo。

如果创建数据库时提供了加密密钥，那么随后与数据库的所有连接都必须使用同一个加密密钥。要更改数据库的加密密钥，请使用 ULConnection.ChangeEncryptionKey。

有关详细信息，请参见“UltraLite 连接参数”《UltraLite - 数据库管理和参考》。

另请参见

- “ULConnectionParms 成员”一节第 178 页
- “Open 方法”一节第 166 页
- “DropDatabase 方法”一节第 241 页
- “ToString 方法”一节第 186 页
- “ULConnection 类”一节第 131 页
- “ULConnection(String) 构造函数”一节第 136 页
- “ConnectionString 属性”一节第 137 页
- “UserID 属性”一节第 186 页
- “Password 属性”一节第 185 页
- “EncryptionKey 属性”一节第 185 页
- “CacheSize 属性”一节第 182 页
- “ConnectionName 属性”一节第 183 页
- “RevokeConnectFrom 方法”一节第 168 页
- “GrantConnectTo 方法”一节第 166 页
- “ChangeEncryptionKey 方法”一节第 148 页

ULConnectionParms 成员

公共构造函数

成员名称	说明
“ULConnectionParms 构造函数”一节第 180 页	用 ULConnectionParms 实例的缺省值初始化该实例。

公共属性

成员名称	说明
“AdditionalParms 属性”一节第 180 页	采用以分号分隔的 "名称=值" 对的列表形式指定附加参数。这些参数不太常用。
“CacheSize 属性”一节第 182 页	指定高速缓存的大小。
“ConnectionName 属性”一节第 183 页	指定连接名称。只有在创建与数据库的多个连接时才需要指定连接名称。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“DatabaseOnCE 属性”一节第 184 页	指定 UltraLite 数据库在 Windows Mobile 上的路径和文件名。
“DatabaseOnDesktop 属性”一节第 184 页	指定 UltraLite 数据库在 Windows 桌面平台上的路径和文件名。
“EncryptionKey 属性”一节第 185 页	指定用于加密数据库的密钥。
“Password 属性”一节第 185 页	为经过验证的用户指定口令。
“UserID 属性”一节第 186 页	为数据库指定一个经过验证的用户。

公共方法

成员名称	说明
“ToString 方法”一节第 186 页	返回此实例的字符串表示形式。

公共事件

成员名称	说明
“UnusedEvent 事件”一节第 187 页	未使用。

另请参见

- “ULConnectionParms 类”一节第 177 页
- “Open 方法”一节第 166 页
- “DropDatabase 方法”一节第 241 页
- “ToString 方法”一节第 186 页
- “ULConnection 类”一节第 131 页
- “ULConnection(String) 构造函数”一节第 136 页
- “ConnectionString 属性”一节第 137 页
- “UserID 属性”一节第 186 页
- “Password 属性”一节第 185 页
- “EncryptionKey 属性”一节第 185 页
- “CacheSize 属性”一节第 182 页
- “ConnectionName 属性”一节第 183 页
- “RevokeConnectFrom 方法”一节第 168 页
- “GrantConnectTo 方法”一节第 166 页
- “ChangeEncryptionKey 方法”一节第 148 页

ULConnectionParms 构造函数

用 ULConnectionParms 实例的缺省值初始化该实例。

语法

Visual Basic
Public Sub **New()**

C#
public **ULConnectionParms();**

另请参见

- “ULConnectionParms 类”一节第 177 页
- “ULConnectionParms 成员”一节第 178 页

AdditionalParms 属性

采用以分号分隔的 "名称=值" 对的列表形式指定附加参数。这是一些较不常用的参数。

语法

Visual Basic
Public Property **AdditionalParms** As String

C#
public string **AdditionalParms** { get; set; }

属性值

以分号分隔的 "关键字=值" 列表形式的附加参数。"关键字=值" 列表的值必须符合 ULConnection.ConnectionString 的规则。缺省值为空值引用（在 Visual Basic 中为 Nothing）。

注释

页面大小及保留大小参数值是以字节为单位指定的。使用后缀 k 或 K 来表示以千字节为单位；使用后缀 m 或 M 来表示以兆字节为单位。

附加参数有：

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

关键字	说明
dbn	标识需要与之建立连接的已装载数据库。 启动数据库时会为它指派一个数据库名，既可使用 dbn 参数显式指派，也可由 UltraLite 使用不带扩展名和路径的基本文件名进行指派。 打开连接时，UltraLite 会首先搜索具有匹配 dbn 且正在运行的数据库。如果未找到，则 UltraLite 会使用适当的数据库文件名参数（DatabaseOnCE 或 DatabaseOnDesktop）启动新的数据库。 如果应用程序（或 UltraLite 引擎）需要访问两个具有相同基本文件名的不同数据库，则必须使用此参数。 仅当使用 ULConnection.Open 打开连接时才会使用此参数。
ordered_table_scans	指定不含 ORDER BY 子句的 SQL 查询是否应在缺省情况下执行有序的表扫描。 自版本 10.0.1 起，当在 UltraLite 中使用动态 SQL 时，如果顺序对于执行查询并不重要，则 UltraLite 将会从数据库页直接访问行，而不是使用主键索引。这样会提高读取行的性能。要使用此优化，查询必须为只读且必须扫描所有的行。 当要求行以特定顺序排序时，应包括 ORDER BY 语句作为 SQL 查询的一部分。但一些应用程序会依赖于缺省情况下按主键顺序返回行的行为。在这种情况下，用户应将 ordered_table_scans 参数设置为 1（true、yes、on），以便在对表进行迭代时恢复为原来的行为。 当 ordered_table_scans 设置为 1（true、yes、on）且用户未指定 ORDER BY 子句时，或者，如果查询不会从索引中受益，则缺省情况下 UltraLite 将会使用主键。 以下参数字符串用于确保 UltraLite 采取与先前版本同样的行为。 `createParms.AdditionalParms = "ordered_table_scans=yes"` 缺省值为 0（false、off、no）。 仅当使用 ULConnection.Open 打开连接时才会使用此参数。 有关详细信息，请参见“UltraLite 连接参数”《UltraLite - 数据库管理和参考》。

关键字	说明
reserve_size	预留文件系统空间以供存储 UltraLite 持久数据。 reserve_size 参数使您不用插入任何数据就能预先分配 UltraLite 数据库所需的文件系统空间。预留文件系统空间可以略微提高性能，还可以防止发生内存不足的故障。缺省情况下，持久存储文件仅会在需要时随着应用程序更新数据库而增长。 请注意，reserve_size 会预留文件系统空间，这包括持久存储文件中的元数据，而不仅仅是原始数据。当根据数据库的数据量得出所需的文件系统空间时，必须考虑元数据开销和数据压缩。建议运行含有测试数据的数据库，并注意观察持久存储文件的大小。 reserve_size 参数预留空间的方法是：在启动时将持久存储文件增大到给定的预留大小，而不管该文件先前是否存在。决不会将文件截断。 以下参数字符串用于确保在启动时持久存储文件的大小至少为 2 MB。 `createParms.AdditionalParms = "reserve_size=2m"` 仅当使用 ULConnection.Open 打开连接时才会使用此参数。
start	指定位置，然后启动 UltraLite 引擎。 仅当要连接到当前未在运行的引擎时，才提供 StartLine (START) 连接参数。 仅当 UltraLite 引擎不在系统路径中时，位置才是必需的。 有关配合使用 UltraLite.NET 和 UltraLite 引擎的详细信息，请参见“RuntimeType 属性”一节第 239 页。

另请参见

- “ULConnectionParms 类”一节第 177 页
- “ULConnectionParms 成员”一节第 178 页
- “ConnectionString 属性”一节第 137 页
- “DatabaseOnCE 属性”一节第 184 页
- “DatabaseOnDesktop 属性”一节第 184 页
- “Open 方法”一节第 166 页

CacheSize 属性

指定高速缓存的大小。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Property **CacheSize** As String

C#
public string **CacheSize** { get; set; }

属性值

指定高速缓存大小的字符串。缺省值为空值引用（在 Visual Basic 中为 Nothing），表示使用缺省值 16 页。

注释

以字节为单位指定高速缓存大小的值。使用后缀 k 或 K 来表示以千字节为单位；使用后缀 m 或 M 来表示以兆字节为单位。

例如，以下代码会将高速缓存大小设置为 128 KB。

```
connParms.CacheSize = "128k"
```

如果未指定高速缓存大小或者指定的大小不正确，则会使用缺省大小。缺省高速缓存大小为 16 页。由于使用缺省页面大小 4 KB，因此缺省高速缓存大小为 64 KB。最小高速缓存大小与平台有关。

缺省高速缓存大小是保守值。如果测试显示需要提高性能，则应该增加高速缓存大小。如果将高速缓存大小增大到超过数据库本身的大小，不但不会提高性能，并且高速缓存过大可能会影响能够使用的其它应用程序的数目。

另请参见

- “UltraLite CACHE_SIZE 连接参数”一节 《UltraLite - 数据库管理和参考》
- “ULConnectionParms 类”一节第 177 页
- “ULConnectionParms 成员”一节第 178 页

ConnectionName 属性

指定连接名称。只有在创建与数据库的多个连接时才需要指定连接名称。

语法

Visual Basic
Public Property **ConnectionName** As String

C#
public string **ConnectionName** { get; set; }

属性值

指定连接名称的字符串。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

另请参见

- “ULConnectionParms 类”一节第 177 页
- “ULConnectionParms 成员”一节第 178 页

DatabaseOnCE 属性

指定 UltraLite 数据库在 Windows Mobile 上的路径和文件名。

语法

Visual Basic
Public Property **DatabaseOnCE** As String

C#
public string **DatabaseOnCE** { get; set; }

属性值

指定数据库完整路径的字符串。如果值为空值引用（在 Visual Basic 中为 Nothing），则会使用数据库 \UltraLiteDB\ulstore.udb。在 C# 中，必须对路径中的任何反斜线字符进行转义，或在字符串文字两边加上 @。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

另请参见

- “ULConnectionParms 类”一节第 177 页
- “ULConnectionParms 成员”一节第 178 页

DatabaseOnDesktop 属性

指定 UltraLite 数据库在 Windows 桌面平台上的路径和文件名。

语法

Visual Basic
Public Property **DatabaseOnDesktop** As String

C#
public string **DatabaseOnDesktop** { get; set; }

属性值

指定数据库绝对或相对路径的字符串。如果值为空值引用（在 Visual Basic 中为 Nothing），则会使用数据库 ulstore.udb。在 C# 中，必须对路径中的任何反斜线字符进行转义，或在字符串文字两边加上 @。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

另请参见

- “ULConnectionParms 类”一节第 177 页
- “ULConnectionParms 成员”一节第 178 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

EncryptionKey 属性

指定用于加密数据库的密钥。

语法

Visual Basic
Public Property **EncryptionKey** As String

C#
public string **EncryptionKey** { get; set; }

属性值

指定加密密钥的字符串。缺省值为空值引用（在 Visual Basic 中为 Nothing），表示没有加密。

注释

所有连接都必须使用在创建数据库时指定的相同密钥。丢失或忘记密钥会导致数据库完全无法访问。

与所有口令一样，最好选择不容易被猜到的密钥值。对密钥的长度没有任何限制，但通常密钥越长越好，因为较短的密钥比较长的密钥更容易被猜到。使用数字、字母和特殊字符的组合会减少他人猜中密钥的几率。

另请参见

- “ULConnectionParms 类”一节第 177 页
- “ULConnectionParms 成员”一节第 178 页
- “ChangeEncryptionKey 方法”一节第 148 页

Password 属性

为经过验证的用户指定口令。

语法

Visual Basic
Public Property **Password** As String

C#
public string **Password** { get; set; }

属性值

指定数据库用户 ID 的字符串。缺省值为空值引用（在 Visual Basic 中为 Nothing）。

注释

口令区分大小写。

创建数据库时，DBA 用户 ID 的口令设置为 sql。

另请参见

- “ULConnectionParms 类”一节第 177 页
- “ULConnectionParms 成员”一节第 178 页
- “UserID 属性”一节第 186 页

UserID 属性

为数据库指定一个经过验证的用户。

语法

Visual Basic
Public Property **UserID** As String

C#
public string **UserID** { get; set; }

属性值

指定数据库用户 ID 的字符串。缺省值为空值引用（在 Visual Basic 中为 Nothing）。

注释

用户 ID 不区分大小写。

数据库最初是以一个名为 DBA 的已验证用户身份创建的。

如果用户 ID 和口令均未提供，则会使用口令为 sql 的用户 DBA。为使数据库更加安全，请更改用户 DBA 的口令，或者创建新的用户（使用 ULConnection.GrantConnectTo）并删除 DBA 用户（使用 ULConnection.RevokeConnectFrom）。

另请参见

- “ULConnectionParms 类”一节第 177 页
- “ULConnectionParms 成员”一节第 178 页
- “Password 属性”一节第 185 页
- “GrantConnectTo 方法”一节第 166 页
- “RevokeConnectFrom 方法”一节第 168 页

ToString 方法

返回此实例的字符串表示形式。

语法

Visual Basic
Public Overrides Function **ToString()** As String

C#
public override string **ToString();**

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

返回值

以分号分隔的 "关键字=值" 对列表形式表示此实例的字符串。

另请参见

- “ULConnectionParms 类”一节第 177 页
- “ULConnectionParms 成员”一节第 178 页

UnusedEvent 事件

未使用。

语法

Visual Basic
Public Event **UnusedEvent** As ULConnectionParms.UnusedEventHandler

C#
public event ULConnectionParms.UnusedEventHandler **UnusedEvent**;

注释

提供此公共事件是为了修复一个 Visual Studio .NET 错误（与此类在 Visual Basic .NET 项目中的集成有关）。它没有实际用途。

另请参见

- “ULConnectionParms 类”一节第 177 页
- “ULConnectionParms 成员”一节第 178 页

ULConnectionParms.UnusedEventHandler 委派

UL Ext.: 未使用。

语法

Visual Basic

```
Public Delegate Sub ULConnectionParms.UnusedEventHandler( _
  ByVal sender As Object, _
  ByVal args As EventArgs _
)
```

C#

```
public delegate void ULConnectionParms.UnusedEventHandler(
  object sender,
  EventArgs args
);
```

注释

提供此公共委派是为了修复一个 Visual Studio .NET 错误（与此类在 Visual Basic .NET 项目中的集成有关）。它没有实际用途。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULConnectionStringBuilder 类

生成用于打开 UltraLite 数据库连接的连接字符串。常用连接参数为 ULConnectionStringBuilder 对象上的各个属性。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULConnectionStringBuilder**
  Inherits DbConnectionStringBuilder

C#
public sealed class **ULConnectionStringBuilder**: DbConnectionstring Builder

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULConnectionStringBuilder 类。

ULConnectionStringBuilder 对象用于指定打开连接 (ULConnection.Open) 或丢弃数据库 (ULDatabaseManager.DropDatabase) 时所需的参数。

前导和尾随空格在所有的值中均会被忽略。值不得包含前导或尾随空格，或是分号 (;)，也不得以单引号 (') 或双引号 (") 开头。

构建连接字符串时，需要标识数据库并指定任何可选的连接设置。通过在 ULConnectionStringBuilder 对象上设置相应属性来提供所有连接参数后，即可使用 System.Data.Common.DbConnectionStringBuilder.ConnectionString 创建连接字符串。结果字符串被 ULConnection(String) 构造函数用来创建新的 ULConnection，或用来设置现有 ULConnection 对象的 ULConnection.ConnectionString。

Identifying the database

每个实例均包含特定于平台的指向数据库的路径。仅使用与执行平台对应的值。例如，在下面的代码中，将在 Windows Mobile 上使用 \UltraLite\mydb1.udb 路径，而在其它平台上则使用 mydb2.db。

```
' Visual Basic
Dim dbName As ULConnectionStringBuilder = _
  new ULConnectionStringBuilder
dbName.DatabaseOnCE = "\UltraLite\mydb1.udb"
dbName.DatabaseOnDesktop = "somedir\mydb2.udb"

// C#
ULConnectionStringBuilder dbName = new ULConnectionStringBuilder();
dbName.DatabaseOnCE = "\\UltraLite\\mydb1.udb";
dbName.DatabaseOnDesktop = @"somedir\mydb2.udb";
```

对于 UltraLite 数据库文件，建议使用扩展名 .udb。在 Windows Mobile 设备上，缺省数据库为 \UltraLiteDB\ulstore.udb。在其它 Windows 平台上，缺省数据库为 ulstore.udb。在 C# 中，必须对路径中的任何反斜线字符进行转义，或是在字符串文字两边加上 @。

如果使用多个数据库，则必须为每个数据库指定一个数据库名。有关详细信息，请参见“DatabaseName 属性”一节第 196 页。

Optional connection settings

视应用程序的需要和数据库的创建方式而定，可能需要提供非缺省的 ULConnectionStringBuilder.UserID 和 ULConnectionStringBuilder.Password、数据库 ULConnectionStringBuilder.DatabaseKey 以及连接 ULConnectionStringBuilder.CacheSize。如果应用程序使用多个连接，则应为每个连接提供唯一的 ULConnectionStringBuilder.ConnectionName。

数据库由一个已通过验证的用户 DBA 来创建，其初始口令为 sql。缺省情况下会使用用户 ID DBA 和口令 sql 打开连接。要禁用缺省用户，请使用 ULConnection.RevokeConnectFrom。要添加用户或更改用户口令，请使用 ULConnection.GrantConnectTo。

如果创建数据库时提供了加密密钥，那么随后与数据库的所有连接都必须使用同一个加密密钥。要更改数据库的加密密钥，请使用 ULConnection.ChangeEncryptionKey。

有关详细信息，请参见“UltraLite 连接参数”《UltraLite - 数据库管理和参考》。

另请参见

- “ULConnectionStringBuilder 成员”一节第 190 页
- “Open 方法”一节第 166 页
- “DropDatabase 方法”一节第 241 页
- DbConnectionStringBuilder.ConnectionString
- “ULConnection 类”一节第 131 页
- “ULConnection(String) 构造函数”一节第 136 页
- “ConnectionString 属性”一节第 137 页
- “DatabaseName 属性”一节第 196 页
- “UserID 属性”一节第 201 页
- “Password 属性”一节第 199 页
- “DatabaseKey 属性”一节第 195 页
- “CacheSize 属性”一节第 194 页
- “ConnectionName 属性”一节第 195 页
- “RevokeConnectFrom 方法”一节第 168 页
- “GrantConnectTo 方法”一节第 166 页
- “ChangeEncryptionKey 方法”一节第 148 页

ULConnectionStringBuilder 成员

公共构造函数

成员名称	说明
“ULConnectionStringBuilder 构造函数”一节第 193 页	初始化一个新的“ULConnectionStringBuilder 类”一节第 189 页实例。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

公共属性

成员名称	说明
BrowsableConnectionString（继承自 DbConnectionStringBuilder）	获取或设置一个值，指示 DbConnectionStringBuilder.ConnectionString 在 Visual Studio 设计器中是否可见。
“CacheSize 属性”一节 第 194 页	**UL Ext.:** 指定高速缓存的大小。
“ConnectionName 属性”一节 第 195 页	指定连接名称。只有在创建与数据库的多个连接时才需要指定连接名称。
ConnectionString（继承自 DbConnectionStringBuilder）	获取或设置与 DbConnectionStringBuilder 相关联的连接字符串。
Count（继承自 DbConnectionStringBuilder）	获取 DbConnectionStringBuilder.ConnectionString 中当前包含的键数。
“DatabaseKey 属性”一节 第 195 页	指定用于加密数据库的密钥。
“DatabaseName 属性”一节 第 196 页	指定数据库的名称，或需要与之建立连接的已装载数据库的名称。
“DatabaseOnCE 属性”一节 第 196 页	**UL Ext.:** 指定 UltraLite 数据库在 Windows Mobile 上的路径和文件名。
“DatabaseOnDesktop 属性”一节第 197 页	**UL Ext.:** 指定 UltraLite 数据库在 Windows 桌面平台中的路径和文件名。
IsFixedSize（继承自 DbConnectionStringBuilder）	获取表示 DbConnectionStringBuilder 是否有固定大小的值。
IsReadOnly（继承自 DbConnectionStringBuilder）	获取表示 DbConnectionStringBuilder 是否为只读的值。
“Item 属性”一节第 197 页	指定所指定连接关键字的值。
Keys（继承自 DbConnectionStringBuilder）	获取包含 DbConnectionStringBuilder 中密钥的 ICollection。
“OrderedTableScans 属性”一节第 199 页	指定不含 ORDER BY 子句的 SQL 查询是否应在缺省情况下执行有序的表扫描。
“Password 属性”一节 第 199 页	为经过验证的用户指定口令。

成员名称	说明
“ReserveSize 属性”一节 第 200 页	**UL Ext.:** 指定为存储 UltraLite 持久数据而预留的文件系统空间.
“StartLine 属性”一节 第 201 页	指定位置，然后启动 UltraLite 引擎。
“UserID 属性”一节第 201 页	为数据库指定一个经过验证的用户。
Values（继承自 DbConnectionStringBuilder）	获取包含 DbConnectionStringBuilder 中的值的 ICollection。

公共方法

成员名称	说明
Add（继承自 DbConnectionStringBuilder）	将具有指定键和值的条目添加到 DbConnectionStringBuilder 中。
Clear（继承自 DbConnectionStringBuilder）	清除 DbConnectionStringBuilder 实例的内容。
“ContainsKey 方法”一节 第 202 页	确定 ULConnectionStringBuilder 对象是否包含特定关键字。
“EquivalentTo 方法”一节 第 202 页	将此 ULConnectionStringBuilder 对象中的连接信息与所提供 DbConnectionStringBuilder 对象中的连接信息进行比较。
“GetShortName 方法”一节 第 203 页	检索所提供关键字的简短版本。
“Remove 方法”一节 第 203 页	从 ULConnectionStringBuilder 实例中删除具有指定键的条目。
ShouldSerialize（继承自 DbConnectionStringBuilder）	表示此 DbConnectionStringBuilder 实例中是否存在指定的键。
ToString（继承自 DbConnectionStringBuilder）	返回与此 DbConnectionStringBuilder 关联的连接字符串。
“TryGetValue 方法”一节 第 204 页	从此 ULConnectionStringBuilder 中检索与所提供的键对应的值。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “Open 方法”一节第 166 页
- “DropDatabase 方法”一节第 241 页
- DbConnectionStringBuilder.ConnectionString
- “ULConnection 类”一节第 131 页
- “ULConnection(String) 构造函数”一节第 136 页
- “ConnectionString 属性”一节第 137 页
- “DatabaseName 属性”一节第 196 页
- “UserID 属性”一节第 201 页
- “Password 属性”一节第 199 页
- “DatabaseKey 属性”一节第 195 页
- “CacheSize 属性”一节第 194 页
- “ConnectionName 属性”一节第 195 页
- “RevokeConnectFrom 方法”一节第 168 页
- “GrantConnectTo 方法”一节第 166 页
- “ChangeEncryptionKey 方法”一节第 148 页

ULConnectionStringBuilder 构造函数

初始化一个新的“ULConnectionStringBuilder 类”一节第 189 页实例。

ULConnectionStringBuilder() 构造函数

用 ULConnectionStringBuilder 实例的缺省值初始化该实例。

语法

Visual Basic
Public Sub **New()**

C#
public **ULConnectionStringBuilder();**

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页
- “ULConnectionStringBuilder 构造函数”一节第 193 页

ULConnectionStringBuilder(String) 构造函数

用指定的连接字符串初始化 ULConnectionStringBuilder 实例。

语法

Visual Basic
```
Public Sub New( _
  ByVal connectionString As String _
)
```

C#
```
public  ULConnectionStringBuilder(
  string  connectionString
);
```

参数

- **connectionString**　UltraLite.NET 连接字符串。连接字符串是一个以分号分隔的 "关键字-值" 对的列表。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页
- “ULConnectionStringBuilder 构造函数”一节第 193 页

CacheSize 属性

UL Ext.: 指定高速缓存的大小。

语法

Visual Basic
```
Public Property CacheSize As String
```

C#
```
public string  CacheSize { get; set; }
```

属性值

指定高速缓存大小的字符串。缺省值为空值引用（在 Visual Basic 中为 Nothing），表示使用缺省值 16 页。

注释

以字节为单位指定高速缓存大小的值。使用后缀 k 或 K 来表示以千字节为单位；使用后缀 m 或 M 来表示以兆字节为单位。

例如，以下代码会将高速缓存大小设置为 128 KB。

```
connParms.CacheSize = "128k"
```

如果未指定高速缓存大小或者指定的大小不正确，则会使用缺省大小。缺省高速缓存大小为 16 页。由于使用缺省页面大小 4 KB，因此缺省高速缓存大小为 64 KB。最小高速缓存大小与平台有关。

缺省高速缓存大小是保守值。如果测试显示需要提高性能，则应该增加高速缓存大小。如果将高速缓存大小增大到超过数据库本身的大小，不但不会提高性能，并且高速缓存过大可能会影响能够使用的其它应用程序的数目。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “UltraLite CACHE_SIZE 连接参数”一节 《UltraLite - 数据库管理和参考》
- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页

ConnectionName 属性

指定连接名称。只有在创建与数据库的多个连接时才需要指定连接名称。

语法

Visual Basic
Public Property **ConnectionName** As String

C#
public string **ConnectionName** { get; set; }

属性值

指定连接名称的字符串。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页

DatabaseKey 属性

指定用于加密数据库的密钥。

语法

Visual Basic
Public Property **DatabaseKey** As String

C#
public string **DatabaseKey** { get; set; }

属性值

指定加密密钥的字符串。缺省值为空值引用（在 Visual Basic 中为 Nothing），表示没有加密。

注释

所有连接都必须使用在创建数据库时指定的相同密钥。丢失或忘记密钥会导致数据库完全无法访问。

与所有口令一样，最好选择不容易被猜到的密钥值。对密钥的长度没有任何限制，但通常密钥越长越好，因为较短的密钥比较长的密钥更容易被猜到。使用数字、字母和特殊字符的组合会减少他人猜中密钥的几率。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页
- “ChangeEncryptionKey 方法”一节第 148 页

DatabaseName 属性

指定数据库的名称，或需要与之建立连接的已装载数据库的名称。

语法

Visual Basic
Public Property **DatabaseName** As String

C#
public string **DatabaseName** { get; set; }

属性值

指定数据库名称的字符串。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

注释

启动数据库时会为它指派一个数据库名，既可使用 dbn 参数显式指派，也可由 UltraLite 使用不带扩展名和路径的基本文件名进行指派。

打开连接时，UltraLite 会首先搜索具有匹配 dbn 且正在运行的数据库。如果未找到，则 UltraLite 会使用适当的数据库文件名参数（DatabaseOnCE 或 DatabaseOnDesktop）启动新的数据库。

如果应用程序（或 UltraLite 引擎）需要访问两个具有相同基本文件名的不同数据库，则必须使用此参数。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页
- “DatabaseOnCE 属性”一节第 196 页
- “DatabaseOnDesktop 属性”一节第 197 页

DatabaseOnCE 属性

UL Ext.: 指定 UltraLite 数据库在 Windows Mobile 上的路径和文件名。

语法

Visual Basic
Public Property **DatabaseOnCE** As String

C#
public string **DatabaseOnCE** { get; set; }

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

属性值

指定数据库完整路径的字符串。如果值为空值引用（在 Visual Basic 中为 Nothing），则会使用数据库 \UltraLiteDB\ulstore.udb。在 C# 中，必须对路径中的任何反斜线字符进行转义，或在字符串文字两边加上 @。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页

DatabaseOnDesktop 属性

UL Ext.: 指定 UltraLite 数据库在 Windows 桌面平台中的路径和文件名。

语法

Visual Basic

```
Public Property DatabaseOnDesktop As String
```

C#

```
public string  DatabaseOnDesktop { get; set; }
```

属性值

指定数据库绝对或相对路径的字符串。如果值为空值引用（在 Visual Basic 中为 Nothing），则会使用数据库 ulstore.udb。在 C# 中，必须对路径中的任何反斜线字符进行转义，或在字符串文字两边加上 @。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页

Item 属性

指定所指定连接关键字的值。

语法

Visual Basic

```
Public Overrides Default Property Item( _
   ByVal keyword As String _
) As Object
```

C#

```
public override object this[
   string  keyword
] { get; set; }
```

参数

- **keyword** 连接关键字的名称。

属性值

表示指定连接关键字的值的对象。

注释

下表中对 ULConnectionStringBuilder 上的连接关键字及对应的属性进行了说明：

关键字	对应的属性
cache_size	“CacheSize 属性”一节第 194 页
ce_file	“DatabaseOnCE 属性”一节第 196 页
con	“ConnectionName 属性”一节第 195 页
dbkey	“DatabaseKey 属性”一节第 195 页
dbn	“DatabaseName 属性”一节第 196 页
nt_file	“DatabaseOnDesktop 属性”一节第 197 页
pwd	“Password 属性”一节第 199 页
reserve_size	“ReserveSize 属性”一节第 200 页
start	“StartLine 属性”一节第 201 页
uid	“UserID 属性”一节第 201 页

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页
- “CacheSize 属性”一节第 194 页
- “DatabaseOnCE 属性”一节第 196 页
- “ConnectionName 属性”一节第 195 页
- “DatabaseKey 属性”一节第 195 页
- “DatabaseName 属性”一节第 196 页
- “DatabaseOnDesktop 属性”一节第 197 页
- “Password 属性”一节第 199 页
- “ReserveSize 属性”一节第 200 页
- “StartLine 属性”一节第 201 页
- “UserID 属性”一节第 201 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

OrderedTableScans 属性

指定不含 ORDER BY 子句的 SQL 查询是否应在缺省情况下执行有序的表扫描。

语法

Visual Basic
Public Property **OrderedTableScans** As String

C#
public string **OrderedTableScans** { get; set; }

属性值

指定是否使用有序表扫描的布尔字符串。例如，true/false、yes/no、1/0，等等。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

注释

自版本 10.0.1 起，当在 UltraLite 中使用动态 SQL 时，如果顺序对于执行查询并不重要，则 UltraLite 将会从数据库页直接访问行，而不是使用主键索引。这样会提高读取行的性能。要使用此优化，查询必须为只读且必须扫描所有的行。

当要求行以特定顺序排序时，应包括 ORDER BY 语句作为 SQL 查询的一部分。但一些应用程序会依赖于缺省情况下按主键顺序返回行的行为。在这种情况下，用户应将 OrderedTableScans 参数设置为 1（true、yes、on），以便在对表进行迭代时恢复为原来的行为。

将 OrderedTableScans 设置为 1（true、yes、on）且用户未指定 ORDER BY 子句时，或者，如果查询不会从索引中受益，则缺省情况下 UltraLite 将会使用主键。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页

Password 属性

为经过验证的用户指定口令。

语法

Visual Basic
Public Property **Password** As String

C#
public string **Password** { get; set; }

属性值

指定数据库用户 ID 的字符串。缺省值为空值引用（在 Visual Basic 中为 Nothing）。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

口令区分大小写。

创建数据库时，DBA 用户 ID 的口令设置为 sql。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页
- “UserID 属性”一节第 201 页

ReserveSize 属性

UL Ext.: 指定为存储 UltraLite 持久数据而预留的文件系统空间.

语法

Visual Basic
Public Property **ReserveSize** As String

C#
public string **ReserveSize** { get; set; }

属性值

指定预留大小的字符串。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

注释

以字节为单位指定预留大小参数的值。使用后缀 k 或 K 来表示以千字节为单位；使用后缀 m 或 M 来表示以兆字节为单位。

reserve_size 参数使您不用插入任何数据就能预先分配 UltraLite 数据库所需的文件系统空间。预留文件系统空间可以略微提高性能，还可以防止发生内存不足的故障。缺省情况下，持久存储文件仅会在需要时随着应用程序更新数据库而增长。

请注意，reserve_size 会预留文件系统空间，这包括持久存储文件中的元数据，而不仅仅是原始数据。当根据数据库的数据量得出所需的文件系统空间时，必须考虑元数据开销和数据压缩。建议运行含有测试数据的数据库，并注意观察持久存储文件的大小。

reserve_size 参数预留空间的方法是：在启动时将持久存储文件增大到给定的预留大小，而不管该文件先前是否存在。决不会将文件截断。

以下参数字符串用于确保在启动时持久存储文件的大小至少为 2 MB。

```
connParms.ReserveSize = "2m"
```

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

StartLine 属性

指定位置，然后启动 UltraLite 引擎。

语法

Visual Basic
Public Property **StartLine** As String

C#
public string **StartLine** { get; set; }

属性值

指定 UltraLite 引擎可执行文件位置的字符串。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

注释

仅当要连接到当前未在运行的引擎时，才提供 StartLine (START) 连接参数。

有关配合使用 UltraLite.NET 和 UltraLite 引擎的详细信息，请参见“RuntimeType 属性”一节第 239 页。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页
- “RuntimeType 属性”一节第 239 页

UserID 属性

为数据库指定一个经过验证的用户。

语法

Visual Basic
Public Property **UserID** As String

C#
public string **UserID** { get; set; }

属性值

指定数据库用户 ID 的字符串。缺省值为空值引用（在 Visual Basic 中为 Nothing）。

注释

用户 ID 不区分大小写。

数据库最初是以一个名为 DBA 的已验证用户身份创建的。

如果用户 ID 和口令均未提供，则会使用口令为 sql 的用户 DBA。为使数据库更加安全，请更改用户 DBA 的口令，或者创建新的用户（使用 ULConnection.GrantConnectTo）并删除 DBA 用户（使用 ULConnection.RevokeConnectFrom）。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页
- “Password 属性”一节第 199 页
- “GrantConnectTo 方法”一节第 166 页
- “RevokeConnectFrom 方法”一节第 168 页

ContainsKey 方法

确定 ULConnectionStringBuilder 对象是否包含特定关键字。

语法

Visual Basic

```
Public Overrides Function ContainsKey( _
  ByVal keyword As String _
) As Boolean
```

C#

```
public override bool ContainsKey(
  string  keyword
);
```

参数

- **keyword**　连接关键字的名称。

返回值

如果此连接字符串构建器包含指定关键字的值，则为 true；否则返回 false。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页

EquivalentTo 方法

将此 ULConnectionStringBuilder 对象中的连接信息与所提供 DbConnectionStringBuilder 对象中的连接信息进行比较。

语法

Visual Basic

```
Public Overrides Function EquivalentTo( _
  ByVal connectionStringBuilder As DbConnectionStringBuilder _
) As Boolean
```

C#

```
public override bool EquivalentTo(
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
  DbConnectionstring Builder connectionStringBuilder
);
```

参数

- **connectionStringBuilder** 要与此 ULConnectionStringBuilder 对象进行比较的其它 DbConnectionStringBuilder 对象。

返回值

如果此对象等效于指定的 DbConnectionStringBuilder 对象，则为 true；否则返回 false。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页
- DbConnectionStringBuilder

GetShortName 方法

检索所提供关键字的简短版本。

语法

Visual Basic

```
Public Shared Function GetShortName( _
  ByVal keyword As String _
) As String
```

C#

```
public static string  GetShortName(
  string  keyword
);
```

参数

- **keyword** 要检索的项目的键。

返回值

如果识别关键字，则为所提供关键字的简短版本；否则为空值。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页

Remove 方法

从 ULConnectionStringBuilder 实例中删除具有指定键的条目。

语法

Visual Basic

```
Public Overrides Function Remove( _
  ByVal keyword As String _
) As Boolean
```

C#

```
public override bool Remove(
  string  keyword
);
```

参数

- **keyword** 连接关键字的名称。

返回值

如果连接字符串内存在过键且已被删除，则为 true；如果键不曾存在，则为 false。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页

TryGetValue 方法

从此 ULConnectionStringBuilder 中检索与所提供的键对应的值。

语法

Visual Basic

```
Public Overrides Function TryGetValue( _
  ByVal keyword As String, _
  ByVal value As Object _
) As Boolean
```

C#

```
public override bool TryGetValue(
  string  keyword,
  object value
);
```

参数

- **keyword** 要检索的项目的键。
- **value** 与键对应的值。

返回值

如果在连接字符串内找到了关键字，则为 true；否则为 false。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

利用 TryGetValue 方法，开发人员无需首先调用 ContainsKey 方法即可从 ULConnectionStringBuilder 安全地检索值。由于在调用 TryGetValue 时即使传入不存在的键也不会抛出异常，因此不必在检索其值之前查找键。以不存在的键调用 TryGetValue 会将空值（在 Visual Basic 中为 Nothing）置于 value 参数中。

另请参见

- “ULConnectionStringBuilder 类”一节第 189 页
- “ULConnectionStringBuilder 成员”一节第 190 页

ULCreateParms 类

UL Ext.: 构建用于创建 UltraLite 数据库的创建时选项字符串。

语法

Visual Basic
Public Class **ULCreateParms**

C#
public class **ULCreateParms**

注释

ULCreateParms 对象用于指定创建数据库 (ULDatabaseManager.CreateDatabase(string,byte[],string)) 时所需的参数。

前导和尾随空格在所有字符串值中均会被忽略。值不得包含前导或尾随空格，或是分号 (;)，也不得以单引号 (') 或双引号 (") 开头。

通过在 ULCreateParms 对象上设置相应属性提供了所有创建参数后，即可使用 ULCreateParms.ToString 创建一个创建参数字符串。随后便可将生成的字符串用作 ULDatabaseManager.CreateDatabase(string,byte[],string) 方法的 createParms 参数。

有关详细信息，请参见“UltraLite 连接参数”《UltraLite - 数据库管理和参考》。

示例

以下代码在 Windows Mobile 设备上创建数据库 \UltraLite\MyDatabase.udb。所创建的数据库区分大小写且使用 UTF8 字符集。

```
' Visual Basic
Dim createParms As ULCreateParms = New ULCreateParms
createParms.CaseSensitive = True
createParms.UTF8Encoding = True
Dim openParms As ULConnectionParms = New ULConnectionParms
openParms.DatabaseOnCE = "\UltraLite\MyDatabase.udb"
' Assumes file coll_1250LATIN2.vb is
' also compiled in the current project
ULConnection.DatabaseManager.CreateDatabase( _
    openParms.ToString(), _
    iAnywhere.UltraLite.Collations.Collation_1250LATIN2.Data, _
    createParms.ToString() _
  )
Dim conn As ULConnection = _
  New ULConnection( openParms.ToString() )
conn.Open()


// C#
ULCreateParms createParms = new ULCreateParms();
createParms.CaseSensitive = true;
createParms.UTF8Encoding = true;
ULConnectionParms openParms = new ULConnectionParms();
openParms.DatabaseOnCE = @"\UltraLite\MyDatabase.udb";
// Assumes file coll_1250LATIN2.vb is
// also compiled in the current project
ULConnection.DatabaseManager.CreateDatabase(
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
      openParms.ToString(),
      iAnywhere.UltraLite.Collations.Collation_1250LATIN2.Data,
      createParms.ToString()
    );
ULConnection conn = new ULConnection( openParms.ToString() );
conn.Open();
```

另请参见

- “ULCreateParms 成员”一节第 207 页
- “GetDatabaseProperty 方法”一节第 251 页
- “CreateDatabase 方法”一节第 240 页
- “ToString 方法”一节第 216 页

ULCreateParms 成员

公共构造函数

成员名称	说明
“ULCreateParms 构造函数”一节第 208 页	用 ULCreateParms 实例的缺省值初始化该实例。

公共属性

成员名称	说明
“CaseSensitive 属性”一节第 209 页	指定在比较字符串值时新数据库是否应区分大小写。
“ChecksumLevel 属性”一节第 209 页	指定为新数据库启用的数据库页校验和级别。
“DateFormat 属性”一节第 210 页	指定新数据库进行字符串转换所用的日期格式。
“DateOrder 属性”一节第 210 页	指定新数据库进行字符串转换所用的日期顺序。
“FIPS 属性”一节第 210 页	指定新数据库应使用 AES_FIPS 加密还是 AES 加密。
“MaxHashSize 属性”一节第 211 页	指定用于在新数据库中进行索引散列的缺省最大字节数。
“NearestCentury 属性”一节第 211 页	指定新数据库进行字符串转换所用的最接近的世纪值。
“Obfuscate 属性”一节第 212 页	指定新数据库是否应使用模糊处理（简单加密）。

成员名称	说明
“PageSize 属性”一节第 212 页	指定新数据库的页大小（以字节或千字节为单位）。
“Precision 属性”一节第 213 页	指定新数据库用于字符串转换的浮点精度。
“Scale 属性”一节第 213 页	指定在新数据库进行字符串转换期间，按最大 PRECISION 截断运算结果时小数点之后最少有几位数。
“TimeFormat 属性”一节第 214 页	指定新数据库进行字符串转换所用的时间格式。
“TimestampFormat 属性”一节第 214 页	指定新数据库进行字符串转换所用的时间戳格式。
“TimestampIncrement 属性”一节第 215 页	指定两个唯一时间戳值之间的最小差异，以微秒（百万分之一秒）为单位。
“UTF8Encoding 属性”一节第 215 页	指定新数据库应使用 UTF8 字符集还是与归类关联的字符集。

公共方法

成员名称	说明
“ToString 方法”一节第 216 页	返回此实例的字符串表示形式。

另请参见

- “ULCreateParms 类”一节第 206 页
- “GetDatabaseProperty 方法”一节第 251 页
- “CreateDatabase 方法”一节第 240 页
- “ToString 方法”一节第 216 页

ULCreateParms 构造函数

用 ULCreateParms 实例的缺省值初始化该实例。

语法

Visual Basic
Public Sub **New()**

C#
public **ULCreateParms();**

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页

CaseSensitive 属性

指定在比较字符串值时新数据库是否应区分大小写。

语法

Visual Basic
Public Property **CaseSensitive** As Boolean

C#
public bool **CaseSensitive** { get; set; }

属性值

如果数据库应区分大小写，则为 true；如果数据库不应区分大小写，则为 false。缺省值为 false。

注释

CaseSensitive 仅影响字符串数据的比较和排序方式。数据库标识符（如表名、列名、索引名和连接用户 ID）始终不区分大小写。连接口令和数据库加密密钥则始终区分大小写。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页

ChecksumLevel 属性

指定为新数据库启用的数据库页校验和级别。

语法

Visual Basic
Public Property **ChecksumLevel** As Integer

C#
public int **ChecksumLevel** { get; set; }

属性值

指定校验和级别的整数。有效值为 0、1 和 2。缺省值为 0。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

DateFormat 属性

指定新数据库进行字符串转换所用的日期格式。

语法

Visual Basic
Public Property **DateFormat** As String

C#
public string **DateFormat** { get; set; }

属性值

指定日期格式的字符串。如果值为空值引用（在 Visual Basic 中为 Nothing），则数据库将使用 "YYYY-MM-DD"。在 C# 中，必须对路径中的任何反斜线字符进行转义，或在字符串文字两边加上 @。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页

DateOrder 属性

指定新数据库进行字符串转换所用的日期顺序。

语法

Visual Basic
Public Property **DateOrder** As ULDateOrder

C#
public ULDateOrder **DateOrder** { get; set; }

属性值

一个 ULDateOrder 值，指明用于进行字符串转换的日期顺序。缺省值为 YMD。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页
- “ULDateOrder 枚举”一节第 296 页

FIPS 属性

指定新数据库应使用 AES_FIPS 加密还是 AES 加密。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Property **FIPS** As Boolean

C#
public bool **FIPS** { get; set; }

属性值

如果应使用 AES_FIPS 对数据库进行加密，则为 true；如果应使用 AES 对数据库进行加密，则为 false。缺省值为 false。

注释

必须通过在创建新数据库时为连接参数 EncryptionKey 提供值来开启加密。如果将 FIPS 设置为 true 而未提供加密密钥，则 ULDatabaseManager.CreateDatabase(string,byte[],string) 方法将会失败，并显示缺少加密密钥的错误。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页
- “EncryptionKey 属性”一节第 185 页
- “CreateDatabase 方法”一节第 240 页

MaxHashSize 属性

指定用于在新数据库中进行索引散列的缺省最大字节数。

语法

Visual Basic
Public Property **MaxHashSize** As Integer

C#
public int **MaxHashSize** { get; set; }

属性值

指定最大散列大小的整数。值必须在 [0.32] 范围内。缺省值为 8。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页

NearestCentury 属性

指定新数据库进行字符串转换所用的最接近的世纪值。

语法

Visual Basic
Public Property **NearestCentury** As Integer

C#
public int **NearestCentury** { get; set; }

属性值

指定最接近世纪值的整数。值必须在 [0.100] 范围内。缺省值为 50。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页

Obfuscate 属性

指定新数据库是否应使用模糊处理（简单加密）。

语法

Visual Basic
Public Property **Obfuscate** As Boolean

C#
public bool **Obfuscate** { get; set; }

属性值

如果应使用模糊处理对数据库进行加密，则为 true；否则为 false。缺省值为 false。

注释

如果开启了 FIPS 加密 (ULCreateParms.FIPS)，则会忽略此选项。如果在创建新数据库时开启了模糊处理并为连接参数 EncryptionKey (DBKEY) 提供了值，则会忽略加密密钥。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页
- “FIPS 属性”一节第 210 页

PageSize 属性

指定新数据库的页大小（以字节或千字节为单位）。

语法

Visual Basic
Public Property **PageSize** As Integer

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

C#
public int **PageSize** { get; set; }

属性值

指定页面大小（以字节为单位）的整数。有效值为 1024 (1K)、2048 (2K)、4096 (4K)、8192 (8K)、16384 (16K)。缺省值为 4096。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页

Precision 属性

指定新数据库用于字符串转换的浮点精度。

语法

Visual Basic
Public Property **Precision** As Integer

C#
public int **Precision** { get; set; }

属性值

指定精度的整数。值必须在 [1,127] 范围内。缺省值为 30。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页
- “Scale 属性”一节第 213 页

Scale 属性

指定在新数据库进行字符串转换期间，按最大 PRECISION 截断运算结果时小数点之后最少有几位数。

语法

Visual Basic
Public Property **Scale** As Integer

C#
public int **Scale** { get; set; }

属性值

指定小数位数的整数。值必须在 [0.127] 范围内。缺省值为 6。

注释

Scale 必须小于或等于 Precision。如果 Scale 大于 Precision，则会在创建数据库时出现错误。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页

TimeFormat 属性

指定新数据库进行字符串转换所用的时间格式。

语法

Visual Basic
Public Property **TimeFormat** As String

C#
public string **TimeFormat** { get; set; }

属性值

指定时间格式的字符串。如果值为空值引用（在 Visual Basic 中为 Nothing），则数据库将使用 "HH:NN:SS.SSS"。在 C# 中，必须对路径中的任何反斜线字符进行转义，或在字符串文字两边加上 @。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页

TimestampFormat 属性

指定新数据库进行字符串转换所用的时间戳格式。

语法

Visual Basic
Public Property **TimestampFormat** As String

C#
public string **TimestampFormat** { get; set; }

属性值

指定时间戳格式的字符串。如果值为空值引用（在 Visual Basic 中为 Nothing），则数据库将使用 "YYYY-MM-DD HH:NN:SS.SSS"。在 C# 中，必须对路径中的任何反斜线字符进行转义，或在字符串文字两边加上 @。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页

TimestampIncrement 属性

指定两个唯一时间戳值之间的最小差异，以微秒（百万分之一秒）为单位。

语法

Visual Basic
Public Property **TimestampIncrement** As Integer

C#
public int **TimestampIncrement** { get; set; }

属性值

指定时间戳增量的整数。值必须在 [1.60000000] 范围内。缺省值为 1。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页

UTF8Encoding 属性

指定新数据库应使用 UTF8 字符集还是与归类关联的字符集。

语法

Visual Basic
Public Property **UTF8Encoding** As Boolean

C#
public bool **UTF8Encoding** { get; set; }

属性值

如果数据库应使用 UTF8 字符集，则为 true；如果数据库应使用与归类关联的字符集，则为 false。缺省值为 false。

注释

如果要存储与归类相关联的字符集中不包含的字符，请选择使用 UTF8 字符集。例如，如果您想要进行 US 排序，需使用 1252LATIN1 归类创建数据库，但是如果您想要以本地拼写形式存储国际地址，则要将 UTF8Encoding 指定为 true。

对于 Symbian OS 设备上使用的数据库，必须将 UTF8Encoding 设置为 true。

对于 Palm OS 设备上使用的数据库，必须将 UTF8Encoding 保留为 false。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页

ToString 方法

返回此实例的字符串表示形式。

语法

Visual Basic
Public Overrides Function **ToString()** As String

C#
public override string **ToString();**

返回值

以分号分隔的 "关键字=值" 对列表形式表示此实例的字符串。

另请参见

- “ULCreateParms 类”一节第 206 页
- “ULCreateParms 成员”一节第 207 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULCursorSchema 类

UL Ext.: 表示 UltraLite.NET 游标的模式。此类属于抽象类，因此无法将其实例化。

语法

Visual Basic
MustInherit Public Class **ULCursorSchema**

C#
public abstract class **ULCursorSchema**

注释

此类是 ULTableSchema 类和 ULResultSetSchema 类的抽象基类。

Note to users porting from the iAnywhere.UltraLite namespace: 列 ID 从 0 开始，而不是像 iAnywhere.UltraLite 命名空间中那样从 1 开始。

另请参见

- “ULCursorSchema 成员”一节第 217 页
- “ULTableSchema 类”一节第 533 页
- “ULResultSetSchema 类”一节第 421 页

ULCursorSchema 成员

公共属性

成员名称	说明
“ColumnCount 属性”一节第 218 页	返回游标中的列数。
“IsOpen 属性”一节第 219 页	检查游标模式当前是否处于打开状态。
“Name 属性”一节第 219 页	返回游标的名称。

公共方法

成员名称	说明
“GetColumnID 方法”一节第 219 页	返回指定列的列 ID。
“GetColumnName 方法”一节第 220 页	返回由指定的列 ID 标识的列的名称。

成员名称	说明
“GetColumnPrecision 方法”一节第 221 页	如果列为数值列（SQL 类型 NUMERIC），则返回由指定的列 ID 标识的列的精度。
“GetColumnSQLName 方法”一节第 222 页	返回由指定的列 ID 标识的列的名称。
“GetColumnScale 方法”一节第 223 页	如果列为数值列（SQL 类型 NUMERIC），则返回由指定的列 ID 标识的列的小数位数。
“GetColumnSize 方法”一节第 223 页	如果列是指定大小的列（SQL 类型 BINARY 或 CHAR），则返回由指定的列 ID 标识的列的大小。
“GetColumnULDbType 方法”一节第 224 页	返回由指定的列 ID 标识的列的 UltraLite.NET 数据类型。
“GetSchemaTable 方法”一节第 224 页	返回描述 ULDataReader 的列模式的 System.Data.DataTable。

另请参见

- “ULCursorSchema 类”一节第 217 页
- “ULTableSchema 类”一节第 533 页
- “ULResultSetSchema 类”一节第 421 页

ColumnCount 属性

返回游标中的列数。

语法

Visual Basic
Public Readonly Property **ColumnCount** As Short

C#
public short **ColumnCount** { get;}

属性值

游标中的列数；如果游标模式已关闭，则为 0。

注释

列 ID 的范围是从 0 到 ColumnCount-1（含 0 和 ColumnCount-1）。

列 ID 和计数在模式升级过程中可能会发生变化。为了正确地标识列，请按名称访问它，或者在模式升级后刷新高速缓存中的 ID 和计数。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULCursorSchema 类”一节第 217 页
- “ULCursorSchema 成员”一节第 217 页

IsOpen 属性

检查游标模式当前是否处于打开状态。

语法

Visual Basic
Public Readonly Property **IsOpen** As Boolean

C#
public bool **IsOpen** { get;}

属性值

如果游标模式当前处于打开状态，则为 true；如果游标模式处于关闭状态，则为 false。

另请参见

- “ULCursorSchema 类”一节第 217 页
- “ULCursorSchema 成员”一节第 217 页

Name 属性

返回游标的名称。

语法

Visual Basic
Public Readonly Property **Name** As String

C#
public string **Name** { get;}

属性值

字符串形式的游标名称。

另请参见

- “ULCursorSchema 类”一节第 217 页
- “ULCursorSchema 成员”一节第 217 页

GetColumnID 方法

返回指定列的列 ID。

语法

Visual Basic
```
Public Function GetColumnID( _
  ByVal name As String _
) As Short
```

C#
```
public short GetColumnID(
  string  name
);
```

参数

- **name**　列的名称。

返回值

指定列的列 ID。

注释

列 ID 的范围是从 0 到 ColumnCount-1（含 0 和 ColumnCount-1）。

请注意，结果集中并非所有列都有名称，并且并非所有列名都是唯一的。如果没有使用别名，非计算列的名称将以其所属表的名称为前缀。例如，MyTable.ID 是查询 "SELECT ID FROM MyTable" 返回的结果集中仅有的一列的名称。

列 ID 和计数在模式升级过程中可能会发生变化。为了正确地标识列，请按名称访问它，或者在模式升级后刷新高速缓存中的 ID 和计数。

另请参见

- “ULCursorSchema 类”一节第 217 页
- “ULCursorSchema 成员”一节第 217 页
- “ColumnCount 属性”一节第 218 页
- “ColumnCount 属性”一节第 218 页

GetColumnName 方法

返回由指定的列 ID 标识的列的名称。

语法

Visual Basic
```
Public Function GetColumnName( _
  ByVal columnID As Integer _
) As String
```

C#
```
public string  GetColumnName(
  int columnID
);
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

参数

- **columnID** 列的 ID。值必须在 [0,ColumnCount-1] 范围内。

返回值

列的名称；如果该列没有名称，则为空值引用（在 Visual Basic 中是 Nothing）。如果该列在 SQL 查询中使用别名，则返回该别名。

注释

请注意，结果集中并非所有列都有名称，并且并非所有列名都是唯一的。如果没有使用别名，非计算列的名称将以其所属表的名称为前缀。例如，MyTable.ID 是查询 "SELECT ID FROM MyTable" 返回的结果集中仅有的一列的名称。

列 ID 和计数在模式升级过程中可能发生变化。为了正确地标识列，请按名称访问它，或者在模式升级后刷新高速缓存中的 ID 和计数。

另请参见

- “ULCursorSchema 类”一节第 217 页
- “ULCursorSchema 成员”一节第 217 页
- “ColumnCount 属性”一节第 218 页
- “ColumnCount 属性”一节第 218 页

GetColumnPrecision 方法

如果列为数值列（SQL 类型 NUMERIC），则返回由指定的列 ID 标识的列的精度。

语法

Visual Basic

```
Public Function GetColumnPrecision( _
   ByVal columnID As Integer _
) As Integer
```

C#

```
public int GetColumnPrecision(
   int columnID
);
```

参数

- **columnID** 列的 ID。值必须在 [0,ColumnCount-1] 范围内。

返回值

所指定数字列的精度。

另请参见

- “ULCursorSchema 类”一节第 217 页
- “ULCursorSchema 成员”一节第 217 页
- “GetColumnULDbType 方法”一节第 224 页
- “ColumnCount 属性”一节第 218 页

GetColumnSQLName 方法

返回由指定的列 ID 标识的列的名称。

语法

Visual Basic

```
Public Function GetColumnSQLName( _
  ByVal columnID As Integer _
) As String
```

C#

```
public string  GetColumnSQLName(
  int columnID
);
```

参数

- **columnID**　列的 ID。值必须在 [0,ColumnCount-1] 范围内。

返回值

列的名称；如果该列没有名称，则为空值引用（在 Visual Basic 中是 Nothing）。如果该列在 SQL 查询中使用别名，则返回该别名。

注释

请注意，结果集中并非所有列都有名称，并且并非所有列名都是唯一的。如果使用别名，则列的名称为别名。

GetColumnSQLName 方法与 GetColumnName 的不同之处在于：对于不使用别名的非计算列，GetColumnSQLName 始终只返回列的名称（而不会以表名作为前缀）。尽管此行为与其它 ADO.NET 提供程序的行为很相近，但很可能会产生不唯一的名称。

列 ID 和计数在模式升级过程中可能发生变化。为了正确地标识列，请按名称访问它，或者在模式升级后刷新高速缓存中的 ID 和计数。

另请参见

- “ULCursorSchema 类”一节第 217 页
- “ULCursorSchema 成员”一节第 217 页
- “ColumnCount 属性”一节第 218 页
- “GetColumnName 方法”一节第 220 页
- “ColumnCount 属性”一节第 218 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

GetColumnScale 方法

如果列为数值列（SQL 类型 NUMERIC），则返回由指定的列 ID 标识的列的小数位数。

语法

Visual Basic

```
Public Function GetColumnScale( _
  ByVal columnID As Integer _
) As Integer
```

C#

```
public int GetColumnScale(
  int columnID
);
```

参数

- **columnID** 列的 ID。值必须在 [0,ColumnCount-1] 范围内。

返回值

指定数字列的小数位数。

另请参见

- “ULCursorSchema 类”一节第 217 页
- “ULCursorSchema 成员”一节第 217 页
- “GetColumnULDbType 方法”一节第 224 页
- “ColumnCount 属性”一节第 218 页

GetColumnSize 方法

如果列是指定大小的列（SQL 类型 BINARY 或 CHAR），则返回由指定的列 ID 标识的列的大小。

语法

Visual Basic

```
Public Function GetColumnSize( _
  ByVal columnID As Integer _
) As Integer
```

C#

```
public int GetColumnSize(
  int columnID
);
```

参数

- **columnID** 列的 ID。值必须在 [0,ColumnCount-1] 范围内。

返回值

指定大小的列的大小。

另请参见

- “ULCursorSchema 类”一节第 217 页
- “ULCursorSchema 成员”一节第 217 页
- “GetColumnULDbType 方法”一节第 224 页
- “ColumnCount 属性”一节第 218 页

GetColumnULDbType 方法

返回由指定的列 ID 标识的列的 UltraLite.NET 数据类型。

语法

Visual Basic

```
Public Function GetColumnULDbType( _
  ByVal columnID As Integer _
) As ULDbType
```

C#

```
public ULDbType GetColumnULDbType(
  int columnID
);
```

参数

- **columnID** 列的 ID。值必须在 [0,ColumnCount-1] 范围内。

返回值

ULDbType 枚举的整数。

另请参见

- “ULCursorSchema 类”一节第 217 页
- “ULCursorSchema 成员”一节第 217 页
- “ColumnCount 属性”一节第 218 页
- “ColumnCount 属性”一节第 218 页
- “ULDbType 枚举”一节第 297 页

GetSchemaTable 方法

返回描述 ULDataReader 的列模式的 System.Data.DataTable。

语法

Visual Basic

```
Public Function GetSchemaTable() As DataTable
```

C#

```
public DataTable GetSchemaTable();
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

返回值

描述列模式的 System.Data.DataTable。

注释

有关详细信息，请参见“GetSchemaTable 方法”一节第 284 页。

另请参见

- “ULCursorSchema 类”一节第 217 页
- “ULCursorSchema 成员”一节第 217 页
- DataTable
- “ULDataReader 类”一节第 257 页

ULDataAdapter 类

表示用来填充 System.Data.DataSet 和更新数据库的一组命令和一个数据库连接。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULDataAdapter**
  Inherits DbDataAdapter

C#
public sealed class **ULDataAdapter**: DbDataAdapter

注释

System.Data.DataSet 提供了一种脱机处理数据的方法；也就是说，无需连接至 UltraLite 数据库。ULDataAdapter 提供了一些将 System.Data.DataSet 与一组 SQL 语句相关联的方法。

因为 UltraLite 为本地数据库且 MobiLink 具备冲突解决能力，所以 ULDataAdapter 的使用将受到限制。大多数情况下，ULDataReader 或 ULTable 会提供更有效的数据访问。

Inherits: System.Data.Common.DbDataAdapter

Implements: System.Data.IDbDataAdapter、System.Data.IDataAdapter、System.IDisposable

另请参见

- “ULDataAdapter 成员”一节第 226 页
- DataSet
- “ULDataReader 类”一节第 257 页
- “ULTable 类”一节第 511 页
- DbDataAdapter
- IDbDataAdapter
- IDataAdapter
- IDisposable

ULDataAdapter 成员

公共构造函数

成员名称	说明
“ULDataAdapter 构造函数”一节第 229 页	初始化一个新的“ULDataAdapter 类”一节第 226 页实例。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

公共属性

成员名称	说明
AcceptChangesDuringFill（继承自 DataAdapter）	获取或设置一个值，指示在任何 Fill 操作期间将 DataRow 添加到 DataTable 之后是否对其调用 DataRow.AcceptChanges。
AcceptChangesDuringUpdate（继承自 DataAdapter）	获取或设置在 DataAdapter.Update 期间是否调用 DataRow.AcceptChanges。
ContinueUpdateOnError（继承自 DataAdapter）	获取或设置一个值，指定在行更新过程中遇到错误时是否产生异常。
“DeleteCommand 属性”一节 第 231 页	指定当调用 DbDataAdapter.Update(System.Data.DataSet) 来删除数据库中与 System.Data.DataSet 中已删除的行对应的行时，针对数据库执行的 ULCommand 对象。
FillLoadOption（继承自 DataAdapter）	获取或设置决定适配器如何从 DbDataReader 填充 DataTable 的 LoadOption。
“InsertCommand 属性”一节 第 232 页	指定当调用 DbDataAdapter.Update(System.Data.DataSet) 以向数据库中插入与 System.Data.DataSet 中已插入的行对应的行时，针对数据库执行的 ULCommand 对象。
MissingMappingAction（继承自 DataAdapter）	确定当进来的数据没有匹配的表或列时所执行的操作。
MissingSchemaAction（继承自 DataAdapter）	决定当现有 DataSet 模式与进入的数据不匹配时采取何种动作。
ReturnProviderSpecificTypes（继承自 DataAdapter）	获取或设置 DataAdapter.Fill 方法应返回特定于提供程序的值还是返回公用的符合 CLS 标准的值。
“SelectCommand 属性”一节 第 233 页	指定在 System.Data.Common.DbDataAdapter.Fill(System.Data.DataSet) 或 System.Data.Common.DbDataAdapter.FillSchema(System.Data.DataSet,System.Data.SchemaType) 过程中使用的 ULCommand，用于从数据库获取要复制到 System.Data.DataSet 的结果集。
“TableMappings 属性”一节 第 234 页	返回提供源表和 System.Data.DataTable 之间主映射的集合
UpdateBatchSize（继承自 DbDataAdapter）	获取或设置一个值，此值可启用或禁用批处理支持，此值同时指定在批处理中可以执行的命令的数量。

成员名称	说明
“UpdateCommand 属性”一节 第 235 页	指定当调用 System.Data.Common.DbDataAdapter.Update(System.Data.DataSet) 来更新数据库中与 System.Data.DataSet 中已更新的行对应的行时，针对数据库执行的 ULCommand 对象。

公共方法

成员名称	说明
Fill（继承自 DbDataAdapter）	填充 DataSet 或 DataTable。
FillSchema（继承自 DbDataAdapter）	将 DataTable 添加到 DataSet 中并将模式配置为与数据源中的模式相匹配。
“GetFillParameters 方法”一节 第 235 页	返回执行 SELECT 语句时由用户设置的参数。
ResetFillLoadOption（继承自 DataAdapter）	将 DataAdapter.FillLoadOption 重置为其缺省状态，并使 DataAdapter.Fill 履行 DataAdapter.AcceptChangesDuringFill。
ShouldSerializeAcceptChangesDuringFill（继承自 DataAdapter）	决定是否应保持 DataAdapter.AcceptChangesDuringFill。
ShouldSerializeFillLoadOption（继承自 DataAdapter）	决定是否应保持 DataAdapter.FillLoadOption。
Update（继承自 DbDataAdapter）	针对 DataSet 中每个插入、更新或删除的行调用相应的 INSERT、UPDATE 或 DELETE 语句。

公共事件

成员名称	说明
FillError（继承自 DataAdapter）	填充操作过程中出现错误时返回。
“RowUpdated 事件”一节 第 236 页	更新过程中对数据源执行命令后发生。尝试进行更新时便会触发该事件。
“RowUpdating 事件”一节 第 237 页	更新过程中对数据源执行命令前发生。尝试进行更新时便会触发该事件。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULDataAdapter 类”一节第 226 页
- DataSet
- “ULDataReader 类”一节第 257 页
- “ULTable 类”一节第 511 页
- DbDataAdapter
- IDbDataAdapter
- IDataAdapter
- IDisposable

ULDataAdapter 构造函数

初始化一个新的“ULDataAdapter 类”一节第 226 页实例。

ULDataAdapter() 构造函数

初始化 ULDataAdapter 对象。

语法

Visual Basic
Public Sub **New()**

C#
public **ULDataAdapter();**

另请参见

- “ULDataAdapter 类”一节第 226 页
- “ULDataAdapter 成员”一节第 226 页
- “ULDataAdapter 构造函数”一节第 229 页
- “ULDataAdapter(ULCommand) 构造函数”一节第 229 页
- “ULDataAdapter(String, ULConnection) 构造函数”一节第 230 页
- “ULDataAdapter(String, String) 构造函数”一节第 231 页

ULDataAdapter(ULCommand) 构造函数

用指定的 SELECT 语句初始化 ULDataAdapter 对象。

语法

Visual Basic
Public Sub **New(_**
  ByVal *selectCommand* As ULCommand _
)

C#
public **ULDataAdapter(**

```
  ULCommand selectCommand
);
```

参数

- **selectCommand** System.Data.Common.DbDataAdapter.Fill(System.Data.DataSet) 中使用的 ULCommand 对象，用于从数据源中选择要放入 System.Data.DataSet 的记录。

另请参见

- “ULDataAdapter 类”一节第 226 页
- “ULDataAdapter 成员”一节第 226 页
- “ULDataAdapter 构造函数”一节第 229 页
- “ULDataAdapter() 构造函数”一节第 229 页
- “ULDataAdapter(String, ULConnection) 构造函数”一节第 230 页
- “ULDataAdapter(String, String) 构造函数”一节第 231 页
- “ULCommand 类”一节第 86 页
- DbDataAdapter.Fill
- DataSet

ULDataAdapter(String, ULConnection) 构造函数

用指定的 SELECT 语句和连接初始化 ULDataAdapter 对象。

语法

Visual Basic

```
Public Sub New( _
  ByVal selectCommandText As String, _
  ByVal selectConnection As ULConnection _
)
```

C#

```
public  ULDataAdapter(
  string  selectCommandText,
  ULConnection selectConnection
);
```

参数

- **selectCommandText** ULDataAdapter 的 ULDataAdapter.SelectCommand 使用的 SELECT 语句。
- **selectConnection** 定义到数据库的连接的 ULConnection 对象。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULDataAdapter 类”一节第 226 页
- “ULDataAdapter 成员”一节第 226 页
- “ULDataAdapter 构造函数”一节第 229 页
- “ULDataAdapter() 构造函数”一节第 229 页
- “ULDataAdapter(ULCommand) 构造函数”一节第 229 页
- “ULDataAdapter(String, String) 构造函数”一节第 231 页
- “SelectCommand 属性”一节第 233 页
- “ULConnection 类”一节第 131 页

ULDataAdapter(String, String) 构造函数

用指定的 SELECT 语句和连接字符串初始化 ULDataAdapter 对象。

语法

Visual Basic

```
Public Sub New( _
  ByVal selectCommandText As String, _
  ByVal selectConnectionString As String _
)
```

C#

```
public  ULDataAdapter(
  string  selectCommandText,
  string  selectConnectionString
);
```

参数

- **selectCommandText**　ULDataAdapter 的 ULDataAdapter.SelectCommand 使用的 SELECT 语句。
- **selectConnectionString**　用于 UltraLite.NET 数据库的连接字符串。

另请参见

- “ULDataAdapter 类”一节第 226 页
- “ULDataAdapter 成员”一节第 226 页
- “ULDataAdapter 构造函数”一节第 229 页
- “ULDataAdapter() 构造函数”一节第 229 页
- “ULDataAdapter(ULCommand) 构造函数”一节第 229 页
- “ULDataAdapter(String, ULConnection) 构造函数”一节第 230 页
- “SelectCommand 属性”一节第 233 页

DeleteCommand 属性

指定当调用 DbDataAdapter.Update(System.Data.DataSet) 来删除数据库中与 System.Data.DataSet 中已删除的行对应的行时，针对数据库执行的 ULCommand 对象。

语法

Visual Basic
Public Property **DeleteCommand** As ULCommand

C#
public ULCommand **DeleteCommand** { get; set; }

属性值

为删除数据库中与 System.Data.DataSet 中已删除的行对应的行而执行的 ULCommand 对象。

注释

将 DeleteCommand 指派给现有 ULCommand 对象时，不会复制 ULCommand 对象。DeleteCommand 会保留对现有 ULCommand 的引用。

这是 System.Data.IDbDataAdapter.DeleteCommand 和 System.Data.Common.DbDataAdapter.DeleteCommand 的强类型版本。

另请参见

- “ULDataAdapter 类”一节第 226 页
- “ULDataAdapter 成员”一节第 226 页
- “ULCommand 类”一节第 86 页
- DbDataAdapter.Update
- DataSet
- “ULCommand 类”一节第 86 页
- IDbDataAdapter.DeleteCommand
- DbDataAdapter.DeleteCommand

InsertCommand 属性

指定当调用 DbDataAdapter.Update(System.Data.DataSet) 以向数据库中插入与 System.Data.DataSet 中已插入的行对应的行时，针对数据库执行的 ULCommand 对象。

语法

Visual Basic
Public Property **InsertCommand** As ULCommand

C#
public ULCommand **InsertCommand** { get; set; }

属性值

为向数据库中插入与 System.Data.DataSet 中已插入的行对应的行而执行的 ULCommand 对象。

注释

将 InsertCommand 指派给现有 ULCommand 对象时，不会复制 ULCommand 对象。InsertCommand 会保留对现有 ULCommand 的引用。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

这是 System.Data.IDbDataAdapter.InsertCommand 和 System.Data.Common.DbDataAdapter.InsertCommand 的强类型版本。

另请参见

- “ULDataAdapter 类”一节第 226 页
- “ULDataAdapter 成员”一节第 226 页
- “ULCommand 类”一节第 86 页
- DbDataAdapter.Update
- DataSet
- “ULCommand 类”一节第 86 页
- IDbDataAdapter.InsertCommand
- DbDataAdapter.InsertCommand

SelectCommand 属性

指定在 System.Data.Common.DbDataAdapter.Fill(System.Data.DataSet) 或 System.Data.Common.DbDataAdapter.FillSchema(System.Data.DataSet,System.Data.SchemaType) 过程中使用的 ULCommand，用于从数据库获取要复制到 System.Data.DataSet 的结果集。

语法

Visual Basic
Public Property **SelectCommand** As ULCommand

C#
public ULCommand **SelectCommand** { get; set; }

属性值

为填充 System.Data.DataSet 而执行的 ULCommand 对象。

注释

将 SelectCommand 指派给现有 ULCommand 对象时，不会复制 ULCommand 对象。SelectCommand 会保留对现有 ULCommand 的引用。

如果 SelectCommand 不返回任何行，则不会将任何表添加到 System.Data.DataSet 中，也不会抛出任何异常。也可在 ULDataAdapter(ULCommand)、ULDataAdapter(String,ULConnection) 或 ULDataAdapter(String,String) 构造函数中指定 SELECT 语句。

这是 System.Data.IDbDataAdapter.SelectCommand 和 System.Data.Common.DbDataAdapter.SelectCommand 的强类型版本。

另请参见

- “ULDataAdapter 类”一节第 226 页
- “ULDataAdapter 成员”一节第 226 页
- “ULCommand 类”一节第 86 页
- DataSet
- DbDataAdapter.Fill
- DbDataAdapter.FillSchema
- “ULDataAdapter(ULCommand) 构造函数”一节第 229 页
- “ULDataAdapter(String, ULConnection) 构造函数”一节第 230 页
- “ULDataAdapter(String, String) 构造函数”一节第 231 页
- IDbDataAdapter.SelectCommand
- DbDataAdapter.SelectCommand

TableMappings 属性

返回提供源表和 System.Data.DataTable 之间主映射的集合

语法

Visual Basic
Public Readonly Property **TableMappings** As DataTableMappingCollection

C#
public DataTableMappingCollection **TableMappings** { get;}

属性值

提供源表和 System.Data.DataTables 之间主映射的 System.Data.Common.DataTableMapping 对象集合。缺省值为空集合。

注释

在使更改保持一致时，ULDataAdapter 会使用 System.Data.Common.DataTableMappingCollection 集合将数据源使用的列名与 System.Data.DataSet 使用的列名相关联。

这是 System.Data.IDataAdapter.TableMappings 的强类型版本。

另请参见

- “ULDataAdapter 类”一节第 226 页
- “ULDataAdapter 成员”一节第 226 页
- DataTable
- DataTableMapping
- DataTable
- DataTableMappingCollection
- DataSet
- IDataAdapter.TableMappings

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

UpdateCommand 属性

指定当调用 System.Data.Common.DbDataAdapter.Update(System.Data.DataSet) 来更新数据库中与 System.Data.DataSet 中已更新的行对应的行时，针对数据库执行的 ULCommand 对象。

语法

Visual Basic
Public Property **UpdateCommand** As ULCommand

C#
public ULCommand **UpdateCommand** { get; set; }

属性值

为更新数据库中与 System.Data.DataSet 中已更新的行对应的行而执行的 ULCommand 对象。

注释

将 UpdateCommand 指派给现有 ULCommand 对象时，不会复制 ULCommand 对象。UpdateCommand 会保留对现有 ULCommand 的引用。

如果执行此命令返回行，则这些行可能会合并到 System.Data.DataSet 中，具体视 ULCommand 对象的 ULCommand.UpdatedRowSource 设置而定。

这是 System.Data.IDbDataAdapter.UpdateCommand 和 System.Data.Common.DbDataAdapter.DeleteCommand 的强类型版本。

另请参见

- “ULDataAdapter 类”一节第 226 页
- “ULDataAdapter 成员”一节第 226 页
- “ULCommand 类”一节第 86 页
- DataSet
- DbDataAdapter.Update
- “UpdatedRowSource 属性”一节第 99 页
- IDbDataAdapter.UpdateCommand
- DbDataAdapter.DeleteCommand

GetFillParameters 方法

返回执行 SELECT 语句时由用户设置的参数。

语法

Visual Basic
Public Function **GetFillParameters()** As ULParameter

C#
public ULParameter **GetFillParameters();**

返回值

包含由用户设置的参数的 ULParameter 对象数组。

注释

这是 System.Data.Common.DbDataAdapter.GetFillParameters 的强类型版本。

另请参见

- “ULDataAdapter 类”一节第 226 页
- “ULDataAdapter 成员”一节第 226 页
- DbDataAdapter.GetFillParameters
- “ULParameter 类”一节第 354 页

RowUpdated 事件

更新过程中对数据源执行命令后发生。尝试进行更新时便会触发该事件。

语法

Visual Basic
Public Event **RowUpdated** As ULRowUpdatedEventHandler

C#
public event ULRowUpdatedEventHandler **RowUpdated**;

注释

要处理行已更新事件，您必须创建一个 ULRowUpdatedEventHandler 委派，并将其附加到此事件。

事件数据

- **Command** 返回调用 DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) 时所执行的 ULCommand。
- **RecordsAffected** 返回通过执行 SQL 语句所更改、插入或删除的行数。对于 SELECT 语句，此值为 -1。
- **Command** 获取调用 DbDataAdapter.Update 时执行的 IDbCommand。
- **Errors** 获取执行 RowUpdatedEventArgs.Command 时由 .NET Framework 数据提供程序生成的所有错误。
- **Row** 获取通过 DbDataAdapter.Update 发送的 DataRow。
- **RowCount** 获取一批更新记录中处理的行数。
- **StatementType** 获取所执行 SQL 语句的类型。
- **Status** 获取 RowUpdatedEventArgs.Command 的 UpdateStatus。
- **TableMapping** 获取通过 DbDataAdapter.Update 发送的 DataTableMapping。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULDataAdapter 类”一节第 226 页
- “ULDataAdapter 成员”一节第 226 页
- “ULRowUpdatedEventHandler 委派”一节第 432 页

RowUpdating 事件

更新过程中对数据源执行命令前发生。尝试进行更新时便会触发该事件。

语法

Visual Basic
Public Event **RowUpdating** As ULRowUpdatingEventHandler

C#
public event ULRowUpdatingEventHandler **RowUpdating**;

注释

要处理行正在更新事件，您必须创建一个 ULRowUpdatingEventHandler 委派，并将其附加到此事件。

事件数据

- **Command** 指定执行 DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) 时要执行的 ULCommand。
- **Command** 获取要在 DbDataAdapter.Update 操作期间执行的 IDbCommand。
- **Errors** 获取执行 RowUpdatedEventArgs.Command 时 .NET Framework 数据提供程序所产生的任何错误。
- **Row** 获取将作为插入、更新或删除操作的一部分发送给服务器的 DataRow。
- **StatementType** 获取要执行的 SQL 语句的类型。
- **Status** 获取或设置 RowUpdatedEventArgs.Command 的 UpdateStatus。
- **TableMapping** 获取要通过 DbDataAdapter.Update 发送的 DataTableMapping。

另请参见

- “ULDataAdapter 类”一节第 226 页
- “ULDataAdapter 成员”一节第 226 页
- “ULRowUpdatedEventHandler 委派”一节第 432 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULDatabaseManager 类

UL Ext.: 管理同步监听器和 UltraLite.NET 运行时类型。ULDatabaseManager 类也允许您删除 UltraLite.NET 数据库。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULDatabaseManager**

C#
public sealed class **ULDatabaseManager**

注释

此类为单个类，可通过 static（在 Visual Basic 为 Shared）ULConnection.DatabaseManager 访问它的唯一实例。

要使用 UltraLite.NET 的 UltraLite 引擎运行时，请在使用任何其它 UltraLite.NET API 之前将 ULDatabaseManager.RuntimeType 设置为适当的值。

示例

以下示例将选择 UltraLite 引擎运行时并创建一个连接。

```
' Visual Basic
ULDatabaseManager.RuntimeType = ULRuntimeType.UL_ENGINE_CLIENT
Dim conn As ULConnection = new ULConnection
' The RuntimeType is now locked

// C#
ULDatabaseManager.RuntimeType = ULRuntimeType.UL_ENGINE_CLIENT;
ULConnection conn = new ULConnection();
// The RuntimeType is now locked
```

另请参见

- “ULDatabaseManager 成员”一节第 238 页
- “DatabaseManager 属性”一节第 140 页
- “RuntimeType 属性”一节第 239 页

ULDatabaseManager 成员

公共属性

成员名称	说明
“RuntimeType 属性”一节第 239 页	指定 UltraLite.NET 运行时类型。使用任何其它 UltraLite.NET API 之前，必须选择运行时类型。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

公共方法

成员名称	说明
“CreateDatabase 方法”一节第 240 页	创建新 UltraLite 数据库。
“DropDatabase 方法”一节第 241 页	删除指定的数据库。 不能删除具有已打开连接的数据库。
“SetActiveSyncListener 方法”一节第 242 页	指定监听器对象，用以处理来自 ActiveSync 的 MobiLink 提供程序的 ActiveSync 调用。
“SetGlobalListener 方法”一节第 243 页	指定用于处理全局同步和 SQL 直通消息的监听器对象。
“SetServerSyncListener 方法”一节第 244 页	指定用于处理指定服务器同步消息的监听器对象。
“SignalSyncIsComplete 方法”一节第 245 页	用信号通知 ActiveSync 的 MobiLink 提供程序某个应用程序已完成同步。
“ValidateDatabase 方法”一节第 245 页	在数据库上执行低级校验和索引校验。

另请参见

- “ULDatabaseManager 类”一节第 238 页
- “DatabaseManager 属性”一节第 140 页
- “RuntimeType 属性”一节第 239 页

RuntimeType 属性

指定 UltraLite.NET 运行时类型。使用任何其它 UltraLite.NET API 之前，必须选择运行时类型。

语法

Visual Basic
Public Shared Property **RuntimeType** As ULRuntimeType

C#
public static ULRuntimeType **RuntimeType** { get; set; }

属性值

ULRuntimeType 值，表示非托管 UltraLite .NET 运行时的类型。

示例

以下示例将选择 UltraLite 引擎运行时并创建一个连接。

```
' Visual Basic
ULDatabaseManager.RuntimeType = ULRuntimeType.UL_ENGINE_CLIENT
Dim conn As ULConnection = new ULConnection
' The RuntimeType is now locked

// C#
ULDatabaseManager.RuntimeType = ULRuntimeType.UL_ENGINE_CLIENT;
ULConnection conn = new ULConnection();
// The RuntimeType is now locked
```

另请参见

- “ULDatabaseManager 类”一节第 238 页
- “ULDatabaseManager 成员”一节第 238 页
- “ULRuntimeType 枚举”一节第 437 页

CreateDatabase 方法

创建新 UltraLite 数据库。

语法

Visual Basic

```
Public Sub CreateDatabase( _
  ByVal connString As String, _
  ByVal collationData As Byte(), _
  ByVal createParms As String _
)
```

C#

```
public void CreateDatabase(
  string  connString,
  byte[] collationData,
  string  createParms
);
```

参数

- **connString**　用于标识数据库的参数，以分号分隔的 "关键字-值" 对列表形式表示。有关详细信息，请参见“ULConnectionParms 类”一节第 177 页。
- **collationData**　指定数据库存储和比较字符串方式的归类数据。
- **createParms**　用于配置新数据库的参数，以分号分隔的 "关键字-值" 对列表形式表示。有关详细信息，请参见“ULCreateParms 类”一节第 206 页。

注释

要指定归类，您必须首先包括 SQL Anywhere 安装目录下 UltraLite\Collations\cs（对于 C# 项目）或 UltraLite\Collations\vb.net（对于 Visual Basic 项目）子目录中的相应归类数据源文件。将其纳入到项目中后，请使用归类的 Data 属性向 CreateDatabase() 方法提供归类数据。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

示例

以下代码会在 Windows Mobile 设备上创建数据库 \UltraLite\MyDatabase.udb，然后打开与该数据库的连接。

```
' Visual Basic
Dim openParms As ULConnectionParms = New ULConnectionParms
openParms.DatabaseOnCE = "\UltraLite\MyDatabase.udb"
' Assumes file UltraLite\Collations\vb.net\coll_1250LATIN2.vb is
' also compiled in the current project
ULConnection.DatabaseManager.CreateDatabase( _
    openParms.ToString(), _
    iAnywhere.UltraLite.Collations.Collation_1250LATIN2.Data, _
    "" _
  )
Dim conn As ULConnection = _
  New ULConnection( openParms.ToString() )
conn.Open()


// C#
ULConnectionParms openParms = new ULConnectionParms();
openParms.DatabaseOnCE = @"\UltraLite\MyDatabase.udb";
// Assumes file UltraLite\Collations\cs\coll_1250LATIN2.cs is
// also compiled in the current project
ULConnection.DatabaseManager.CreateDatabase(
    openParms.ToString(),
    iAnywhere.UltraLite.Collations.Collation_1250LATIN2.Data,
    ""
  );
ULConnection conn = new ULConnection( openParms.ToString() );
conn.Open();
```

另请参见

- “ULDatabaseManager 类”一节第 238 页
- “ULDatabaseManager 成员”一节第 238 页
- “Open 方法”一节第 166 页

DropDatabase 方法

删除指定的数据库。

不能删除具有已打开连接的数据库。

语法

Visual Basic

```
Public Sub DropDatabase( _
  ByVal connString As String _
)
```

C#

```
public void DropDatabase(
  string  connString
);
```

参数

- **connString** 用于标识数据库的参数，以分号分隔的 "关键字-值" 对列表形式表示。有关详细信息，请参见“ULConnectionParms 类”一节第 177 页。

示例

以下代码会在 Windows Mobile 设备上创建数据库 \UltraLite\MyDatabase.udb，然后打开与该数据库的连接。

```
' Visual Basic
Dim connParms As ULConnectionParms = New ULConnectionParms
connParms.DatabaseOnCE = "\UltraLite\MyDatabase.udb"
ULConnection.DatabaseManager.DropDatabase( _
    connParms.ToString() _
  )


// C#
ULConnectionParms connParms = new ULConnectionParms();
connParms.DatabaseOnCE = @"\UltraLite\MyDatabase.udb";
ULConnection.DatabaseManager.DropDatabase(
    connParms.ToString()
  );
ULConnection conn = new ULConnection( openParms.ToString() );
conn.Open();
```

另请参见

- “ULDatabaseManager 类”一节第 238 页
- “ULDatabaseManager 成员”一节第 238 页
- “Open 方法”一节第 166 页

SetActiveSyncListener 方法

指定监听器对象，用以处理来自 ActiveSync 的 MobiLink 提供程序的 ActiveSync 调用。

语法

Visual Basic

```
Public Sub SetActiveSyncListener( _
  ByVal appClassName As String, _
  ByVal listener As ULActiveSyncListener _
)
```

C#

```
public void SetActiveSyncListener(
  string  appClassName,
  ULActiveSyncListener listener
);
```

参数

- **appClassName** 应用程序的唯一类名。这是在应用程序注册用于 ActiveSync 时所使用的类名。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

- **listener** ULActiveSyncListener 对象。使用空值（Visual Basic 中是 Nothing）删除以前的监听器。

注释

参数 *appClassName* 是用于标识应用程序的唯一标识符。应用程序一次只能使用一个 *appClassName*。因为监听器是用特定的 *appClassName* 注册的，而使用其它 *appClassName* 调用 SetServerSyncListener 或 SetActiveSyncListener 将会失败。

要删除 ActiveSync 监听器，请使用空值引用（在 Visual Basic 中为 Nothing）作为 *listener* 参数调用 SetActiveSyncListener。

要删除所有监听器，请使用空值引用（在 Visual Basic 中为 Nothing）作为所有参数调用 SetServerSyncListener。

应用程序在退出前应删除所有监听器。

示例

有关 SetActiveSyncListener 的示例，请参见“ActiveSyncInvoked 方法”一节第 53 页。

另请参见

- “ULDatabaseManager 类”一节第 238 页
- “ULDatabaseManager 成员”一节第 238 页
- “SetServerSyncListener 方法”一节第 244 页
- “SetActiveSyncListener 方法”一节第 242 页
- “ULActiveSyncListener 接口”一节第 53 页
- “ActiveSyncInvoked 方法”一节第 53 页

SetGlobalListener 方法

指定用于处理全局同步和 SQL 直通消息的监听器对象。

语法

Visual Basic

```
Public Sub SetGlobalListener( _
  ByVal syncListener As ULSyncProgressListener, _
  ByVal sqlListener As ULSqlPassthroughProgressListener _
)
```

C#

```
public void SetGlobalListener(
  ULSyncProgressListener syncListener,
  ULSqlPassthroughProgressListener sqlListener
);
```

参数

- **syncListener** 实现 SyncProgressed() 的 ULSyncProgressListener 对象，处理全局同步消息时调用此对象。

- **sqlListener** 实现 ScriptProgressed() 的 ULSqlPassthroughProgressListener 对象，执行每个 SQL 直通脚本时都调用此对象。

注释

当执行 SYNCHRONIZE profileName SQL 语句时，如果它的进程消息不为空值（在 Visual Basic 中不为 Nothing），则会将进程消息发送到 syncListener。

当连接上数据库时，可能有许多 SQL 脚本可用并会自动执行。同样，在随后进行的同步中还可以将脚本向下传递，使用 ULConnection.ExecuteSQLPassthroughScripts() 直接执行。在任何一种情况中，如果进程消息不为空值（在 Visual Basic 中不为 Nothing），则进程消息都会被发送到 sqlListener。

删除任何一个监听器都将在某个调用中的空值引用传递到 SetGlobalListener。从 11.0 版本开始，应用程序退出前不再需要删除监听器。

另请参见

- “ULDatabaseManager 类”一节第 238 页
- “ULDatabaseManager 成员”一节第 238 页
- “ULSyncProgressListener 接口”一节第 501 页
- “ULSqlPassthroughProgressListener 接口”一节第 453 页
- “ExecuteSQLPassthroughScripts 方法”一节第 155 页

SetServerSyncListener 方法

指定用于处理指定服务器同步消息的监听器对象。

语法

Visual Basic

```
Public Sub SetServerSyncListener( _
  ByVal messageName As String, _
  ByVal appClassName As String, _
  ByVal listener As ULServerSyncListener _
)
```

C#

```
public void SetServerSyncListener(
  string  messageName,
  string  appClassName,
  ULServerSyncListener listener
);
```

参数

- **messageName** 消息的名称。
- **appClassName** 应用程序的唯一类名。这是用来标识此应用程序的唯一标识符。
- **listener** ULServerSyncListener 对象。使用空值（Visual Basic 中是 Nothing）删除以前的监听器。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

参数 *appClassName* 是用于标识应用程序的唯一标识符。应用程序一次只能使用一个 *appClassName*。因为监听器是用特定的 *appClassName* 注册的，而使用其它 *appClassName* 调用 SetServerSyncListener 或 SetActiveSyncListener 将会失败。

要删除特定消息的监听器，请使用空值引用（在 Visual Basic 中为 Nothing）作为 *listener* 参数调用 SetServerSyncListener。

要删除所有监听器，请使用空值引用（在 Visual Basic 中为 Nothing）作为所有参数调用 SetServerSyncListener。

应用程序在退出前应删除所有监听器。

示例

有关 SetServerSyncListener 的示例，请参见“ServerSyncInvoked 方法”一节第 438 页。

另请参见

- “ULDatabaseManager 类”一节第 238 页
- “ULDatabaseManager 成员”一节第 238 页
- “SetServerSyncListener 方法”一节第 244 页
- “SetActiveSyncListener 方法”一节第 242 页
- “ServerSyncInvoked 方法”一节第 438 页

SignalSyncIsComplete 方法

用信号通知 ActiveSync 的 MobiLink 提供程序某个应用程序已完成同步。

语法

Visual Basic
Public Sub **SignalSyncIsComplete()**

C#
public void **SignalSyncIsComplete();**

示例

有关 SignalSyncIsComplete 的示例，请参见“ActiveSyncInvoked 方法”一节第 53 页。

另请参见

- “ULDatabaseManager 类”一节第 238 页
- “ULDatabaseManager 成员”一节第 238 页
- “SignalSyncIsComplete 方法”一节第 245 页

ValidateDatabase 方法

在数据库上执行低级校验和索引校验。

语法

Visual Basic

```
Public Sub ValidateDatabase( _
  ByVal start_parms As String, _
  ByVal how As ULDBValid _
)
```

C#

```
public void ValidateDatabase(
  string  start_parms,
  ULDBValid how
);
```

参数

- **start_parms** 用于标识数据库的参数，以分号分隔的 "关键字-值" 对列表形式表示。有关详细信息，请参见“ULConnectionParms 类”一节第 177 页。
- **how** 如何校验数据库。有关详细信息，请参见“ULDBValid 枚举”一节第 301 页。

示例

以下代码校验 Windows Mobile 下的数据库 \UltraLite\MyDatabase.udb 的索引

```
' Visual Basic
Dim openParms As ULConnectionParms = New ULConnectionParms
openParms.DatabaseOnCE = "\UltraLite\MyDatabase.udb"
ULConnection.DatabaseManager.ValidateDatabase( _
    openParms.ToString(), iAnywhere.Data.UltraLite.ULVF_INDEX )


// C#
ULConnectionParms openParms = new ULConnectionParms();
openParms.DatabaseOnCE = @"\UltraLite\MyDatabase.udb";
ULConnection.DatabaseManager.ValidateDatabase(
    openParms.ToString(), iAnywhere.Data.UltraLite.ULVF_INDEX );
```

另请参见

- “ULDatabaseManager 类”一节第 238 页
- “ULDatabaseManager 成员”一节第 238 页
- “ValidateDatabase 方法”一节第 173 页
- “ULDBValid 枚举”一节第 301 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULDatabaseSchema 类

UL Ext.: 表示 UltraLite.NET 数据库的模式。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULDatabaseSchema**

C#
public sealed class **ULDatabaseSchema**

注释

没有用于此类的构造函数。ULDatabaseSchema 对象以连接的 ULConnection.Schema 形式附加到连接上，并且只有在该连接处于打开状态时才有效。

另请参见

- “ULDatabaseSchema 成员”一节第 247 页
- “ULDatabaseSchema 类”一节第 247 页
- “Schema 属性”一节第 142 页

ULDatabaseSchema 成员

公共属性

成员名称	说明
“CollationName 属性”一节第 248 页	不建议使用此属性。请改用 GetDatabaseProperty("Collation")。数据库的归类序列名称。
“IsCaseSensitive 属性”一节第 248 页	检查数据库是否区分大小写。
“IsOpen 属性”一节第 249 页	数据库模式是否处于打开状态。
“PublicationCount 属性”一节第 250 页	数据库中发布的数目。
“TableCount 属性”一节第 250 页	数据库中表的数目。

公共方法

成员名称	说明
“GetDatabaseProperty 方法”一节第 251 页	返回指定数据库属性的值。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“GetPublicationName 方法”一节第 253 页	返回由指定的发布 ID 标识的发布的名称。
“GetPublicationSchema 方法”一节第 253 页	返回与指定发布对应的发布模式.
“GetTableName 方法”一节第 254 页	返回由指定的表 ID 所标识的表的名称。
“SetDatabaseOption 方法”一节第 255 页	设置指定数据库选项的值。

另请参见

- “ULDatabaseSchema 类”一节第 247 页
- “ULDatabaseSchema 类”一节第 247 页
- “Schema 属性”一节第 142 页

CollationName 属性

不建议使用此属性。请改用 GetDatabaseProperty("Collation")。数据库的归类序列名称。

语法

Visual Basic
Public Readonly Property **CollationName** As String

C#
public string **CollationName** { get;}

属性值

表示数据库归类序列的字符串。

注释

数据库归类序列将影响表和结果集索引的排序方式。

另请参见

- “ULDatabaseSchema 类”一节第 247 页
- “ULDatabaseSchema 成员”一节第 247 页
- “GetDatabaseProperty 方法”一节第 251 页

IsCaseSensitive 属性

检查数据库是否区分大小写。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Readonly Property **IsCaseSensitive** As Boolean

C#
public bool **IsCaseSensitive** { get;}

属性值

如果数据库区分大小写，则返回 true；如果数据库不区分大小写，则返回 false。

注释

数据库是否区分大小写将影响表索引和结果集索引的排序方式。是否区分大小写还会影响 ULConnectionParms.UserID 和 ULConnectionParms.Password 的验证方式。

另请参见

- “ULDatabaseSchema 类”一节第 247 页
- “ULDatabaseSchema 成员”一节第 247 页
- “GetDatabaseProperty 方法”一节第 251 页
- “UserID 属性”一节第 186 页
- “Password 属性”一节第 185 页

IsOpen 属性

数据库模式是否处于打开状态。

语法

Visual Basic
Public Readonly Property **IsOpen** As Boolean

C#
public bool **IsOpen** { get;}

属性值

如果此数据库模式当前处于打开状态，则返回 true；如果此数据库模式当前处于关闭状态，则返回 false。

注释

ULDatabaseSchema 对象只有在其附加到的连接已打开的情况下才打开。

另请参见

- “ULDatabaseSchema 类”一节第 247 页
- “ULDatabaseSchema 成员”一节第 247 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

PublicationCount 属性

数据库中发布的数目。

语法

Visual Basic
Public Readonly Property **PublicationCount** As Integer

C#
public int **PublicationCount** { get;}

属性值

数据库中发布的数目；如果连接未打开，则为 0。

注释

发布 ID 的范围是从 1 到 PublicationCount（含 1 和 PublicationCount）。

注意：发布 ID 和计数在模式升级过程中可能发生变化。为了正确地标识发布，请按名称访问它，或者在模式升级后刷新高速缓存中的 ID 和计数。

另请参见

- “ULDatabaseSchema 类”一节第 247 页
- “ULDatabaseSchema 成员”一节第 247 页
- “GetPublicationName 方法”一节第 253 页

TableCount 属性

数据库中表的数目。

语法

Visual Basic
Public Readonly Property **TableCount** As Integer

C#
public int **TableCount** { get;}

属性值

数据库中表的数目；如果连接未打开，则为 0。

注释

表 ID 的范围是从 1 到 TableCount（含 1 和 TableCount）。

注意：表 ID 和表计数在模式升级过程中可能发生变化。为了正确地标识表，请按名称访问它，或者在模式升级后刷新高速缓存中的 ID 和计数。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULDatabaseSchema 类”一节第 247 页
- “ULDatabaseSchema 成员”一节第 247 页

GetDatabaseProperty 方法

返回指定数据库属性的值。

语法

Visual Basic

```
Public Function GetDatabaseProperty( _
   ByVal name As String _
) As String
```

C#

```
public string  GetDatabaseProperty(
   string  name
);
```

参数

- **name**　您要获取其值的数据库属性的名称。属性名称不区分大小写。

返回值

以字符串形式返回的属性值。

注释

识别的属性有：

属性	说明
CaseSensitive	区分大小写功能的状态。如果数据库区分大小写则返回 ON。否则，将返回 OFF。 数据库是否区分大小写将影响表索引和结果集索引的排序方式。 是否区分大小写不会影响连接的 ULConnectionParms.UserID 和 ULConnectionParms.Password 的验证方式。用户 ID 始终不区分大小写，而口令始终区分大小写。
CharSet	数据库的字符集。
ChecksumLevel	为数据库启用数据库页校验和的级别。
Collation	数据库的归类序列名称。
CollationName	不建议使用此属性。请改用 "Collation"。

属性	说明
ConnCount	到数据库的连接的数量。
date_format	数据库进行字符串转换所用的日期格式。 此格式不必与 System.DateTime 使用的格式相同。
date_order	数据库用于字符串转换的日期顺序。
Encryption	数据库所应用的加密类型。返回 None、Simple、AES 或 AES_FIPS。
File	数据库的文件名称。
global_database_id	用于全局自动增量列的 global_database_id 选项的值。
isolation_level	isolation_level 选项的值用于控制一个事务中的操作对另一个并发事务中的操作可见程度。 此值基于每个连接来设置。
MaxHashSize	用于检索散列的缺省最大字节数。可基于每个索引设置此属性。
ml_remote_id	用于在同步过程中标识数据库的 ml_remote_id 选项的值。
Name	数据库的名称 (DBN)。
nearest_century	数据库用于字符串转换的最近一个世纪。
PageSize	数据库的页面大小，以字节为单位。
precision	数据库用于字符串转换的浮点精度。
scale	数据库进行字符串转换的过程中，算术结果按最大 PRECISION 值截断时小数点后的最少位数。
time_format	数据库用于字符串转换的时间格式。 此格式不必与 System.TimeSpan 使用的格式相同。
timestamp_format	数据库用于字符串转换的时间戳格式。 此格式不必与 System.DateTime 使用的格式相同。
timestamp_increment	两个唯一时间戳值之间的最小差异，以微秒（百万分之一秒）为单位。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULDatabaseSchema 类”一节第 247 页
- “ULDatabaseSchema 成员”一节第 247 页
- “SetDatabaseOption 方法”一节第 255 页
- “CollationName 属性”一节第 248 页
- “UserID 属性”一节第 186 页
- “Password 属性”一节第 185 页
- TimeSpan
- DateTime

GetPublicationName 方法

返回由指定的发布 ID 标识的发布的名称。

语法

Visual Basic

```
Public Function GetPublicationName( _
   ByVal pubID As Integer _
) As String
```

C#

```
public string  GetPublicationName(
   int pubID
);
```

参数

- **pubID**　发布的 ID。值必须在 [1,PublicationCount] 范围内。

返回值

字符串形式的发布名称。

注释

注意：发布 ID 和计数在模式升级过程中可能发生变化。为了正确地标识发布，请按名称访问它，或者在模式升级后刷新高速缓存中的 ID 和计数。

另请参见

- “ULDatabaseSchema 类”一节第 247 页
- “ULDatabaseSchema 成员”一节第 247 页
- “PublicationCount 属性”一节第 250 页
- “PublicationCount 属性”一节第 250 页

GetPublicationSchema 方法

返回与指定发布对应的发布模式.

语法

Visual Basic

```
Public Function GetPublicationSchema( _
  ByVal name As String _
) As ULPublicationSchema
```

C#

```
public ULPublicationSchema GetPublicationSchema(
  string  name
);
```

参数

- **name** 发布的名称。

返回值

表示指定发布的 ULPublicationSchema 对象。

另请参见

- “ULDatabaseSchema 类”一节第 247 页
- “ULDatabaseSchema 成员”一节第 247 页
- “GetPublicationName 方法”一节第 253 页
- “ULPublicationSchema 类”一节第 391 页

GetTableName 方法

返回由指定的表 ID 所标识的表的名称。

语法

Visual Basic

```
Public Function GetTableName( _
  ByVal tableID As Integer _
) As String
```

C#

```
public string  GetTableName(
  int tableID
);
```

参数

- **tableID** 表的 ID。值必须在 [1,TableCount] 范围内。

返回值

字符串形式的表名。

注释

表 ID 在模式升级过程中可发生变化。为了正确地标识表，请按名称访问它，或者在模式升级后刷新高速缓存中的 ID。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULDatabaseSchema 类”一节第 247 页
- “ULDatabaseSchema 成员”一节第 247 页
- “TableCount 属性”一节第 250 页

SetDatabaseOption 方法

设置指定数据库选项的值。

语法

Visual Basic

```
Public Sub SetDatabaseOption( _
  ByVal name As String, _
  ByVal value As String _
)
```

C#

```
public void SetDatabaseOption(
  string  name,
  string  value
);
```

参数

- **name** 数据库选项的名称。选项名称不区分大小写。
- **value** 选项的新值。

注释

设置数据库选项会导致执行提交。

识别的选项有：

选项	说明
global_database_id	用于全局自动增量列的值。值必须在 [0,System.UInt32.MaxValue] 范围内。缺省值为 ULConnection.INVALID_DATABASE_ID（用于指示没有为当前数据库设置数据库 ID）。

选项	说明
isolation_level	此值可用于控制一个事务中的操作对其它并发事务中的操作的可见程度。此值必须是 "read_uncommitted" 或 "read_committed" 中的一个。缺省值是 "read_committed"。 将连接的 isolation_level 设置为 "read_uncommited" 等效于使用 BeginTransaction(System.Data.IsolationLevel.ReadUncommitted) 和 Commit() 调用将该连接上的所有操作包装起来。同样，"read_committed" 等效于 System.Data.IsolationLevel.ReadCommitted。不应使用 SetDatabaseOption() 来设置当前事务的隔离级别；而应使用 BeginTransaction(IsolationLevel)。 与 ADO.NET 的 IsolationLevel 文档相比，UltraLite 的每个隔离级别定义稍有不同。有关详细信息，请参见“UltraLite 隔离级别”一节 《UltraLite - 数据库管理和参考》。 此值基于每个连接来设置。
ml_remote_id	用于在同步过程中标识数据库的值。使用空值引用（在 Visual Basic 中为 Nothing）作为可从数据库中删除 ml_remote_id 选项的值。

另请参见

- “ULDatabaseSchema 类”一节第 247 页
- “ULDatabaseSchema 成员”一节第 247 页
- “GetDatabaseProperty 方法”一节第 251 页
- UInt32.MaxValue
- “INVALID_DATABASE_ID 字段”一节第 137 页
- “BeginTransaction(IsolationLevel) 方法”一节第 145 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULDataReader 类

表示 UltraLite 数据库内的一个只读双向游标。游标是来自表的行集或来自查询的结果集的行集。

语法

Visual Basic
Public Class **ULDataReader**
  Inherits DbDataReader

C#
public class **ULDataReader**: DbDataReader

注释

没有用于 ULDataReader 的构造函数。要获取 ULDataReader 对象，请执行 ULCommand：

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "SELECT emp_id FROM employee FOR READ ONLY", conn _
  )
Dim reader As ULDataReader = cmd.ExecuteReader()

// C#
ULCommand cmd = new ULCommand(
    "SELECT emp_id FROM employee FOR READ ONLY", conn
  );
ULDataReader reader = cmd.ExecuteReader();
```

UL Ext.: ADO.NET 标准仅需要在结果集中进行只进操作，而 ULDataReader 是双向的。当在结果中移动时，ULDataReader 的 Move 方法可以提供充分的灵活性。

ULDataReader 为只读结果集。如果您需要更为灵活的对象来操作结果，请使用 ULCommand.ExecuteResultSet()、ULCommand.ExecuteTable() 或 ULDataAdapter。ULDataReader 会根据需要检索行，而 ULDataAdapter 必须先检索结果集中的所有行，然后才能对对象执行任何动作。对于较大的结果集来说，这种差别使得 ULDataReader 的响应时间大大缩短。

UL Ext.: 可以使用 GetString 检索 ULDataReader 的所有列。

Inherits: System.Data.Common.DbDataReader

Implements: System.Data.IDataReader、System.Data.IDataRecord、System.IDisposable、System.ComponentModel.IListSource

另请参见

- “ULDataReader 成员”一节第 258 页
- “ULCommand 类”一节第 86 页
- “ExecuteResultSet() 方法”一节第 115 页
- “ExecuteTable() 方法”一节第 118 页
- “ULDataAdapter 类”一节第 226 页
- “GetString 方法”一节第 286 页
- DbDataReader
- IDataReader
- IDataRecord
- IDisposable
- IListSource

ULDataReader 成员

公共属性

成员名称	说明
“Depth 属性”一节第 262 页	返回当前行的嵌套深度。最外层的表深度为 0。
“FieldCount 属性”一节第 262 页	返回游标中的列数。
“HasRows 属性”一节第 263 页	检查 ULDataReader 包含一行还是多行。
“IsBOF 属性”一节第 263 页	**UL Ext.:** 检查当前行位置是否在第一行之前。
“IsClosed 属性”一节第 264 页	检查游标当前是否处于打开状态。
“IsEOF 属性”一节第 264 页	**UL Ext.:** 检查当前行位置是否在最后一行之后。
“Item 属性”一节第 264 页	以 Object 实例的形式获取指定列的值。
“RecordsAffected 属性”一节第 266 页	返回通过执行 SQL 语句所更改、插入或删除的行数。对于 SELECT 语句或 CommandType.TableDirect 表，此值为 -1。
“RowCount 属性”一节第 267 页	**UL Ext.:** 返回游标中的行数。
“Schema 属性”一节第 267 页	**UL Ext.:** 保存此游标的模式。
VisibleFieldCount（继承自 DbDataReader）	获取 DbDataReader 中未隐藏的字段数。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

公共方法

成员名称	说明
“Close 方法”一节第 268 页	关闭游标。
Dispose（继承自 DbDataReader）	释放此 DbDataReader 所占用的资源。
“GetBoolean 方法”一节第 268 页	以 System.Boolean 形式返回指定列的值。
“GetByte 方法”一节第 269 页	以无符号 8 位值 (System.Byte) 形式返回指定列的值。
“GetBytes 方法”一节第 269 页	**UL Ext.:** 以 System.Bytes 数组的形式返回指定列的值。仅对 ULDbType.Binary、ULDbType.LongBinary 或 ULDbType.UniqueIdentifier 类型的列有效。
“GetChar 方法”一节第 272 页	UltraLite.NET 不支持此方法。
“GetChars 方法”一节第 273 页	从指定的偏移量开始，将指定的 ULDbType.LongVarchar 列的值的子集复制到目标 System.Char 数组的指定偏移量处。
GetData（继承自 DbDataReader）	返回与请求的列顺序号对应的 DbDataReader 对象。
“GetDataTypeName 方法”一节第 274 页	返回指定列的提供程序数据类型的名称。
“GetDateTime 方法”一节第 275 页	以 System.DateTime 形式返回指定列的值，其精度为毫秒。
“GetDecimal 方法”一节第 275 页	以 System.Decimal 形式返回指定列的值。
“GetDouble 方法”一节第 276 页	以 System.Double 形式返回指定列的值。
“GetEnumerator 方法”一节第 277 页	返回迭代通过 ULDataReader 的 System.Collections.IEnumerator。
“GetFieldType 方法”一节第 277 页	返回最适合于指定列的 System.Type。
“GetFloat 方法”一节第 278 页	以 System.Single 形式返回指定列的值。

成员名称	说明
“GetGuid 方法”一节 第 279 页	以 UUID (System.Guid) 形式返回指定列的值。
“GetInt16 方法”一节 第 279 页	以 System.Int16 形式返回指定列的值。
“GetInt32 方法”一节 第 280 页	以 Int32 形式返回指定列的值。
“GetInt64 方法”一节 第 281 页	以 Int64 形式返回指定列的值。
“GetName 方法”一节 第 281 页	返回指定列的名称。
“GetOrdinal 方法”一节 第 282 页	返回指定列的列 ID。
GetProviderSpecificFieldType（继承自 DbDataReader）	返回指定列特定于提供程序的字段类型。
GetProviderSpecificValue（继承自 DbDataReader）	以 Object 实例的形式获取指定列的值。
GetProviderSpecificValues（继承自 DbDataReader）	获取当前行的集合中所有特定于提供程序的属性列。
“GetSchemaTable 方法”一节 第 284 页	返回描述 ULDataReader 的列元数据的 System.Data.DataTable。
“GetString 方法”一节 第 286 页	以 System.String 形式返回指定列的值。
“GetTimeSpan 方法”一节 第 286 页	以 System.TimeSpan 形式返回指定列的值，其精度为毫秒。
“GetUInt16 方法”一节 第 287 页	以 System.UInt16 形式返回指定列的值。
“GetUInt32 方法”一节 第 288 页	以 UInt32 形式返回指定列的值。
“GetUInt64 方法”一节 第 288 页	以 System.UInt64 形式返回指定列的值。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“GetValue 方法”一节 第 289 页	返回以本地格式表示的指定列的值。
“GetValues 方法”一节 第 290 页	返回当前行的所有列值。
“IsDBNull 方法”一节 第 291 页	检查指定列的值是否为 NULL。
“MoveAfterLast 方法”一节 第 291 页	**UL Ext.:** 将游标定位到游标的最后一行之后。
“MoveBeforeFirst 方法”一节 第 292 页	**UL Ext.:** 将游标定位到游标的第一行之前。
“MoveFirst 方法”一节 第 292 页	**UL Ext.:** 将游标定位到游标的第一行。
“MoveLast 方法”一节 第 292 页	**UL Ext.:** 将游标定位到游标的最后一行。
“MoveNext 方法”一节 第 293 页	**UL Ext.:** 将游标定位到下一行；如果游标已经位于最后一行，则定位到最后一行之后。
“MovePrevious 方法”一节 第 293 页	**UL Ext.:** 将游标定位到上一行，或第一行之前。
“MoveRelative 方法”一节 第 294 页	**UL Ext.:** 相对于当前行定位游标。
“NextResult 方法”一节 第 294 页	读取批处理 SQL 语句的结果时，将 ULDataReader 推进到下一结果。
“Read 方法”一节第 295 页	将游标定位到下一行；如果游标已经位于最后一行，则定位到最后一行之后。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULCommand 类”一节第 86 页
- “ExecuteResultSet() 方法”一节第 115 页
- “ExecuteTable() 方法”一节第 118 页
- “ULDataAdapter 类”一节第 226 页
- “GetString 方法”一节第 286 页
- DbDataReader
- IDataReader
- IDataRecord
- IDisposable

Depth 属性

返回当前行的嵌套深度。最外层的表深度为 0。

语法

Visual Basic
Public Overrides Readonly Property **Depth** As Integer

C#
public override int **Depth** { get;}

属性值

所有 UltraLite.NET 结果集的深度都为 0。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页

FieldCount 属性

返回游标中的列数。

语法

Visual Basic
Public Overrides Readonly Property **FieldCount** As Integer

C#
public override int **FieldCount** { get;}

属性值

整数形式的游标中的列数。如果游标是关闭的，则返回 0。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

此方法与 ULCursorSchema.ColumnCount 方法相同。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “ColumnCount 属性”一节第 218 页

HasRows 属性

检查 ULDataReader 包含一行还是多行。

语法

Visual Basic
Public Overrides Readonly Property **HasRows** As Boolean

C#
public override bool **HasRows** { get;}

属性值

如果结果集中至少包含一行，则为 true；如果不包含任何行，则为 false。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页

IsBOF 属性

UL Ext.: 检查当前行位置是否在第一行之前。

语法

Visual Basic
Public Readonly Property **IsBOF** As Boolean

C#
public bool **IsBOF** { get;}

属性值

如果当前行位置在第一行之前，则返回 True，否则返回 false。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页

IsClosed 属性

检查游标当前是否处于打开状态。

语法

Visual Basic
Public Overrides Readonly Property **IsClosed** As Boolean

C#
public override bool **IsClosed** { get;}

属性值

如果游标当前处于打开状态，则返回 true；如果游标处于关闭状态，则返回 false。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页

IsEOF 属性

UL Ext.: 检查当前行位置是否在最后一行之后。

语法

Visual Basic
Public Readonly Property **IsEOF** As Boolean

C#
public bool **IsEOF** { get;}

属性值

如果当前行位置在最后一行之后，则返回 true，否则返回 false。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页

Item 属性

以 Object 实例的形式获取指定列的值。

Item(Int32) 属性

返回以本地格式表示的指定列的值。在 C# 中，此属性是 ULDataReader 类的索引器。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic

```
Public Overrides Default Readonly Property Item( _
  ByVal columnID As Integer _
) As Object
```

C#

```
public override object this[
  int columnID
] { get;}
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

属性值

最适合于此列的 .NET 类型的列值；如果列为 NULL，则为 DBNull。

注释

此方法在功能上与 ULDataReader.GetValue(int) 方法相同。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “Item 属性”一节第 264 页
- “GetFieldType 方法”一节第 277 页
- “Item(String) 属性”一节第 265 页
- “GetValue 方法”一节第 289 页
- “FieldCount 属性”一节第 262 页

Item(String) 属性

返回以本地格式表示的指定列的值。在 C# 中，此属性是 ULDataReader 类的索引器。

语法

Visual Basic

```
Public Overrides Default Readonly Property Item( _
  ByVal name As String _
) As Object
```

C#

```
public override object this[
  string  name
] { get;}
```

参数

- **name** 列的名称。

属性值

最适合于此列的 .NET 类型的列值；如果列为 NULL，则为 DBNull。

注释

请注意，结果集中并非所有列都有名称，并且并非所有列名都是唯一的。如果没有使用别名，非计算列的名称将以其所属表的名称为前缀。例如，MyTable.ID 是查询 "SELECT ID FROM MyTable" 返回的结果集中仅有的一列的名称。

当多次访问列时，按列 ID 访问列要比按名称访问列更有效。

此方法等效于：

```
dataReader.GetValue( dataReader.GetOrdinal( name ) )
```

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “Item 属性”一节第 264 页
- “Item(Int32) 属性”一节第 264 页
- “GetOrdinal 方法”一节第 282 页
- “GetValue 方法”一节第 289 页
- “GetFieldType 方法”一节第 277 页

RecordsAffected 属性

返回通过执行 SQL 语句所更改、插入或删除的行数。对于 SELECT 语句或 CommandType.TableDirect 表，此值为 -1。

语法

Visual Basic
Public Overrides Readonly Property **RecordsAffected** As Integer

C#
public override int **RecordsAffected** { get;}

属性值

通过执行该 SQL 语句而更改、插入或删除的行数。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- CommandType.TableDirect

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

RowCount 属性

UL Ext.: 返回游标中的行数。

语法

Visual Basic
Public Readonly Property **RowCount** As Integer

C#
public int **RowCount** { get;}

属性值

游标中的行数。

注释

RowCount 的一个用途是确定何时删除旧行以节省空间。可以使用 ULConnection.StopSynchronizationDelete 方法从 UltraLite 数据库中删除旧行，而不从统一数据库中删除旧行。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “StartSynchronizationDelete 方法”一节第 170 页
- “StopSynchronizationDelete 方法”一节第 171 页

Schema 属性

UL Ext.: 保存此游标的模式。

语法

Visual Basic
Public Readonly Property **Schema** As ULCursorSchema

C#
public ULCursorSchema **Schema** { get;}

属性值

对于结果集，为表示结果集模式的 ULResultSetSchema 对象。对于表，为表示表模式的 ULTableSchema 对象。

注释

此属性表示游标的完整模式，包括从 ULDataReader.GetSchemaTable 返回的结果中未表示的 UltraLite.NET 扩展信息。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “ULTableSchema 类”一节第 533 页
- “GetSchemaTable 方法”一节第 284 页
- “ULResultSetSchema 类”一节第 421 页

Close 方法

关闭游标。

语法

Visual Basic
Public Overrides Sub **Close()**

C#
public override void **Close();**

注释

关闭已关闭的游标不是错误。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页

GetBoolean 方法

以 System.Boolean 形式返回指定列的值。

语法

Visual Basic
Public Overrides Function **GetBoolean(_**
  ByVal *columnID* As Integer _
) As Boolean

C#
public override bool **GetBoolean(**
  int *columnID*
);

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

返回值

System.Boolean 形式的列值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- Boolean
- “FieldCount 属性”一节第 262 页

GetByte 方法

以无符号 8 位值 (System.Byte) 形式返回指定列的值。

语法

Visual Basic

```
Public Overrides Function GetByte( _
   ByVal columnID As Integer _
) As Byte
```

C#

```
public override byte GetByte(
   int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

System.Byte 形式的列值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- Byte
- “FieldCount 属性”一节第 262 页

GetBytes 方法

UL Ext.: 以 System.Bytes 数组的形式返回指定列的值。仅对 ULDbType.Binary、ULDbType.LongBinary 或 ULDbType.UniqueIdentifier 类型的列有效。

GetBytes(Int32, Int64, Byte[], Int32, Int32) 方法

从指定的偏移量开始，到目标 System.Byte 数组的指定偏移量为止，复制指定的 ULDbType.LongBinary 列的值的子集。

语法

Visual Basic

```
Public Overrides Function GetBytes( _
   ByVal columnID As Integer, _
   ByVal srcOffset As Long, _
   ByVal dst As Byte(), _
   ByVal dstOffset As Integer, _
   ByVal count As Integer _
) As Long
```

C#

```
public override long GetBytes(
   int columnID,
   long srcOffset,
   byte[] dst,
   int dstOffset,
   int count
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **srcOffset** 列值的起始位置。该值从零开始。
- **dst** 目标数组。
- **dstOffset** 目标数组的起始位置。
- **count** 要复制的字节数。

返回值

实际复制的字节数。

注释

如果传递一个为空值引用（在 Visual Basic 中是 Nothing）的 *dst* 缓冲区，GetBytes 会以字节为单位返回该字段的长度。

位于值的 *srcOffset* 到 *srcOffset*+*count*-1 之间的字节将被逐个复制到目标数组的 *dstOffset* 到 *dstOffset*+*count*-1 位置处。如果在复制完 *count* 个字节之前就到达了值尾，则目标数组的剩余部分保持不变。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

如果出现以下任一情况，则抛出代码为 ULSQLCode.SQLE_INVALID_PARAMETER 的 ULException，并且不会修改目标：

- *srcOffset* 为负值。
- *dstOffset* 为负值。
- *count* 为负值。
- *dstOffset*+*count* 大于 *dst*.Length。

对于其它错误，抛出具有相应错误代码的 ULException。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetBytes 方法”一节第 269 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- “GetBytes(Int32) 方法”一节第 271 页
- “ULDbType 枚举”一节第 297 页
- Byte
- “ULException 类”一节第 302 页
- “ULSQLCode 枚举”一节第 441 页
- “FieldCount 属性”一节第 262 页

GetBytes(Int32) 方法

UL Ext.: 以 System.Bytes 数组的形式返回指定列的值。仅对 ULDbType.Binary、ULDbType.LongBinary 或 ULDbType.UniqueIdentifier 类型的列有效。

语法

Visual Basic

```
Public Function GetBytes( _
  ByVal columnID As Integer _
) As Byte()
```

C#

```
public byte[] GetBytes(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

System.Bytes 数组形式的列值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetBytes 方法”一节第 269 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- “GetBytes(Int32, Int64, Byte[], Int32, Int32) 方法”一节第 270 页
- Byte
- “ULDbType 枚举”一节第 297 页
- “ULDbType 枚举”一节第 297 页
- “ULDbType 枚举”一节第 297 页

GetChar 方法

UltraLite.NET 不支持此方法。

语法

Visual Basic

```
Public Overrides Function GetChar( _
   ByVal columnID As Integer _
) As Char
```

C#

```
public override char GetChar(
   int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

UltraLite.NET 不支持此方法。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- “GetString 方法”一节第 286 页
- “FieldCount 属性”一节第 262 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

GetChars 方法

从指定的偏移量开始，将指定的 ULDbType.LongVarchar 列的值的子集复制到目标 System.Char 数组的指定偏移量处。

语法

Visual Basic

```
Public Overrides Function GetChars( _
  ByVal columnID As Integer, _
  ByVal srcOffset As Long, _
  ByVal dst As Char(), _
  ByVal dstOffset As Integer, _
  ByVal count As Integer _
) As Long
```

C#

```
public override long GetChars(
  int columnID,
  long srcOffset,
  char[] dst,
  int dstOffset,
  int count
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **srcOffset** 列值的起始位置。该值从零开始。
- **dst** 目标数组。
- **dstOffset** 目标数组的起始位置。
- **count** 要复制的字符数。

返回值

实际复制的字符数。

注释

如果传递一个为空值引用（在 Visual Basic 中是 Nothing）的 *dst* 缓冲区，GetChars 会以字符为单位返回该字段的长度。

位于值的 *srcOffset* 到 *srcOffset*+*count*-1 之间的字符将被逐个复制到目标数组的 *dstOffset* 到 *dstOffset*+*count*-1 位置处。如果在复制完 *count* 个字符之前就到达了值尾，则目标数组的剩余部分保持不变。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

如果出现以下任一情况，则抛出代码为 ULSQLCode.SQLE_INVALID_PARAMETER 的 ULException，并且不会修改目标：

- *srcOffset* 为负值。
- *dstOffset* 为负值。
- *count* 为负值。
- *dstOffset*+*count* 大于 *dst*.Length。

对于其它错误，抛出具有相应错误代码的 ULException。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- “ULDbType 枚举”一节第 297 页
- Char
- “ULException 类”一节第 302 页
- “ULSQLCode 枚举”一节第 441 页
- “FieldCount 属性”一节第 262 页

GetDataTypeName 方法

返回指定列的提供程序数据类型的名称。

语法

Visual Basic

```
Public Overrides Function GetDataTypeName( _
  ByVal columnID As Integer _
) As String
```

C#

```
public override string  GetDataTypeName(
  int columnID
);
```

参数

- **columnID**　列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

对应于列的 ULDbType 的字符串。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetColumnULDbType 方法”一节第 224 页
- “FieldCount 属性”一节第 262 页
- “ULDbType 枚举”一节第 297 页

GetDateTime 方法

以 System.DateTime 形式返回指定列的值，其精度为毫秒。

语法

Visual Basic

```
Public Overrides Function GetDateTime( _
   ByVal columnID As Integer _
) As Date
```

C#

```
public override DateTime GetDateTime(
   int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

System.DateTime 形式的列值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- DateTime
- “FieldCount 属性”一节第 262 页

GetDecimal 方法

以 System.Decimal 形式返回指定列的值。

语法

Visual Basic

```
Public Overrides Function GetDecimal( _
```

```
  ByVal columnID As Integer _
) As Decimal
```

C#

```
public override decimal GetDecimal(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

System.Decimal 形式的列值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- Decimal
- “FieldCount 属性”一节第 262 页

GetDouble 方法

以 System.Double 形式返回指定列的值。

语法

Visual Basic

```
Public Overrides Function GetDouble( _
  ByVal columnID As Integer _
) As Double
```

C#

```
public override double GetDouble(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

System.Double 形式的列值。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- Double
- “FieldCount 属性”一节第 262 页

GetEnumerator 方法

返回迭代通过 ULDataReader 的 System.Collections.IEnumerator。

语法

Visual Basic
Public Overrides Function **GetEnumerator()** As IEnumerator

C#
public override IEnumerator **GetEnumerator();**

返回值

ULDataReader 的 System.Collections.IEnumerator。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- IEnumerator

GetFieldType 方法

返回最适合于指定列的 System.Type。

语法

Visual Basic
```
Public Overrides Function GetFieldType( _
  ByVal columnID As Integer _
) As Type
```

C#
```
public override Type GetFieldType(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

列的 System.Type 值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetDataTypeName 方法”一节第 274 页
- “GetColumnULDbType 方法”一节第 224 页
- 类型
- “FieldCount 属性”一节第 262 页

GetFloat 方法

以 System.Single 形式返回指定列的值。

语法

Visual Basic

```
Public Overrides Function GetFloat( _
  ByVal columnID As Integer _
) As Single
```

C#

```
public override float  GetFloat(
  int columnID
);
```

参数

- **columnID**　列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

System.Single 形式的列值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- Single
- “FieldCount 属性”一节第 262 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

GetGuid 方法

以 UUID (System.Guid) 形式返回指定列的值。

语法

Visual Basic
```
Public Overrides Function GetGuid( _
  ByVal columnID As Integer _
) As Guid
```

C#
```
public override Guid GetGuid(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

Guid 形式的列值。

注释

此方法仅对 ULDbType.UniqueIdentifier 类型的列或长度为 16 的 ULDbType.Binary 类型的列有效。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- “GetColumnULDbType 方法”一节第 224 页
- “GetColumnSize 方法”一节第 223 页
- Guid
- “ULDbType 枚举”一节第 297 页
- “ULDbType 枚举”一节第 297 页
- “FieldCount 属性”一节第 262 页

GetInt16 方法

以 System.Int16 形式返回指定列的值。

语法

Visual Basic
```
Public Overrides Function GetInt16( _
  ByVal columnID As Integer _
) As Short
```

C#
```
public override short GetInt16(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

System.Int16 形式的列值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- Int16
- “FieldCount 属性”一节第 262 页

GetInt32 方法

以 Int32 形式返回指定列的值。

语法

Visual Basic
```
Public Overrides Function GetInt32( _
  ByVal columnID As Integer _
) As Integer
```

C#
```
public override int GetInt32(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

Int32 形式的列值。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- Int32
- “FieldCount 属性”一节第 262 页

GetInt64 方法

以 Int64 形式返回指定列的值。

语法

Visual Basic

```
Public Overrides Function GetInt64( _
  ByVal columnID As Integer _
) As Long
```

C#

```
public override long GetInt64(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

Int64 形式的列值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- Int64
- “FieldCount 属性”一节第 262 页

GetName 方法

返回指定列的名称。

语法

Visual Basic

```
Public Overrides Function GetName( _
```

```
  ByVal columnID As Integer _
) As String

C#
public override string  GetName(
  int columnID
);
```

参数

- **columnID**　列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

列的名称；如果该列没有名称，则为空值引用（在 Visual Basic 中是 Nothing）。如果该列在 SQL 查询中使用别名，则返回该别名。

注释

请注意，结果集中并非所有列都有名称，并且并非所有列名都是唯一的。如果没有使用别名，非计算列的名称将以其所属表的名称为前缀。例如，MyTable.ID 是查询 "SELECT ID FROM MyTable" 返回的结果集中仅有的一列的名称。

此方法与 ULCursorSchema.GetColumnName 方法相同。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “FieldCount 属性”一节第 262 页
- “GetSchemaTable 方法”一节第 284 页
- “GetColumnName 方法”一节第 220 页
- “FieldCount 属性”一节第 262 页

GetOrdinal 方法

返回指定列的列 ID。

语法

```
Visual Basic
Public Overrides Function GetOrdinal( _
  ByVal columnName As String _
) As Integer

C#
public override int GetOrdinal(
  string  columnName
);
```


Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

参数

- **columnName** 列的名称。

返回值

指定列的列 ID。

注释

列 ID 的范围是从 0 到 ULDataReader.FieldCount-1（含 0 和 ULDataReader.FieldCount-1）。

请注意，结果集中并非所有列都有名称，并且并非所有列名都是唯一的。如果没有使用别名，非计算列的名称将以其所属表的名称为前缀。例如，MyTable.ID 是查询 "SELECT ID FROM MyTable" 返回的结果集中仅有的一列的名称。

列 ID 和计数在模式升级过程中可能发生变化。为了正确地标识列，请按名称访问它，或者在模式升级后刷新高速缓存中的 ID 和计数。

此方法与 ULCursorSchema.GetColumnID 方法相同。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetSchemaTable 方法”一节第 284 页
- “FieldCount 属性”一节第 262 页
- “GetColumnID 方法”一节第 219 页

GetRowCount 方法

返回表或结果集中行数的计数。在只需要确定存在超过指定数量的可用行的情况下，为避免枚举所有的行，这时可指定阈值。例如，应用程序正在显示来自 10 个行中的信息，它可能想知道是否存在 10 个以上的可用行，以便能够为用户显示更多信息提供选择。

语法

Visual Basic

```
public int GetRowCount( _
ByVal threshold as Integer ) _
as Integer
```

C#

```
public int GetRowCount( int threshold );
```

返回值

如果阈值参数为零（无阈值），返回的结果是表或结果集中的行数；否则，表或结果集中的返回行数最多是阈值的最大值。

注释

对于较大的表或结果集，枚举其中的所有行是一项高开销操作。如果应用程序只需知道是否存在超过指定数量的行，则可使用此方法。

GetSchemaTable 方法

返回描述 ULDataReader 的列元数据的 System.Data.DataTable。

语法

Visual Basic
Public Overrides Function **GetSchemaTable()** As DataTable

C#
public override DataTable **GetSchemaTable();**

返回值

描述 ULDataReader 中各列的模式的 System.Data.DataTable。

注释

GetSchemaTable 方法以下面的顺序返回关于每个列的元数据：

DataTable 列	说明
ColumnName	列的名称；如果该列没有名称，则为空值引用（在 Visual Basic 中是 Nothing）。如果该列在 SQL 查询中使用别名，则返回该别名。请注意，结果集中并非所有列都有名称，并且并非所有列名都是唯一的。
ColumnOrdinal	列的 ID。该值在 [0,FieldCount-1] 范围内。
ColumnSize	对于指定大小的列，为列中值的最大长度。对于其它列，为以字节表示的数据类型大小。
NumericPrecision	数值列的精度（ProviderType ULDbType.Decimal 或 ULDbType.Numeric）；如果该列不是数值列，则为 DBNull。
NumericScale	数值列的小数位（ProviderType ULDbType.Decimal 或 ULDbType.Numeric）；如果该列不是数值列，则为 DBNull。
IsUnique	如果该列在提取它的表 (BaseTableName) 中是非计算唯一列，则为 true。
IsKey	如果该列是从结果集的唯一键中一起提取出来的一组列中的一列，则为 true。IsKey 设置为 true 的列集不必是作为行在结果集内唯一标识的最小集。
BaseCatalogName	包含该列的数据库中目录的名称。对于 UltraLite.NET，此值始终为 DBNull。
BaseColumnName	数据库表 BaseTableName 中该列的原始名称；如果该列为计算列或无法确定此信息，则为 DBNull。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

DataTable 列	说明
BaseSchemaName	包含该列的数据库中模式的名称。对于 UltraLite.NET，此值始终为 DBNull。
BaseTableName	数据库中包含该列的表的名称；如果该列为计算列或无法确定此信息，则为 DBNull。
DataType	最适合此类型列的 .NET 数据类型。
AllowDBNull	如果该列可以为空，则为 true；如果该列不可以为空或无法确定此信息，则为 false。
ProviderType	列的 ULDbType。
IsIdentity	如果该列是标识列，则为 true；如果该列不是标识列，则为 false。对于 UltraLite.NET，此值始终为 false。
IsAutoIncrement	如果该列是自动增量列或全局自动增量列，则为 true；否则（或无法确定此信息时）为 false。
IsRowVersion	如果该列包含无法写入的持久性行标识符，并且该标识符除了标识行以外没有其它有意义的价值，则为 true。对于 UltraLite.NET，此值始终为 false。
IsLong	如果列为 ULDbType.LongVarchar 或 ULDbType.LongBinary 列，则为 true；否则为 false。
IsReadOnly	如果该列为只读，则为 true；如果该列可修改或无法确定其访问权限，则为 false。
IsAliased	如果列名是别名，则为 true；如果不是别名，则为 false。
IsExpression	如果该列是表达式，则为 true；如果是列值，则为 false。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “Schema 属性”一节第 267 页
- “FieldCount 属性”一节第 262 页
- “ULDbType 枚举”一节第 297 页
- “ULDbType 枚举”一节第 297 页
- “ULDbType 枚举”一节第 297 页
- “ULDbType 枚举”一节第 297 页
- “ULDbType 枚举”一节第 297 页
- DataTable

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

GetString 方法

以 System.String 形式返回指定列的值。

语法

Visual Basic

```
Public Overrides Function GetString( _
  ByVal columnID As Integer _
) As String
```

C#

```
public override string  GetString(
  int columnID
);
```

参数

- **columnID**　列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

System.String 形式的列值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- String
- “FieldCount 属性”一节第 262 页

GetTimeSpan 方法

以 System.TimeSpan 形式返回指定列的值，其精度为毫秒。

语法

Visual Basic

```
Public Function GetTimeSpan( _
  ByVal columnID As Integer _
) As TimeSpan
```

C#

```
public TimeSpan GetTimeSpan(
  int columnID
);
```


Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

System.TimeSpan 形式的列值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- TimeSpan
- “FieldCount 属性”一节第 262 页

GetUInt16 方法

以 System.UInt16 形式返回指定列的值。

语法

Visual Basic

```
Public Function GetUInt16( _
  ByVal columnID As Integer _
) As UInt16
```

C#

```
public ushort GetUInt16(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

System.UInt16 形式的列值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- UInt16
- “FieldCount 属性”一节第 262 页

GetUInt32 方法

以 UInt32 形式返回指定列的值。

语法

Visual Basic
```
Public Function GetUInt32( _
  ByVal columnID As Integer _
) As UInt32
```

C#
```
public uint GetUInt32(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

UInt32 形式的列值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- UInt32
- “FieldCount 属性”一节第 262 页

GetUInt64 方法

以 System.UInt64 形式返回指定列的值。

语法

Visual Basic
```
Public Function GetUInt64( _
  ByVal columnID As Integer _
) As UInt64
```

C#
```
public ulong GetUInt64(
  int columnID
);
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

System.UInt64 形式的列值

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- UInt64
- “FieldCount 属性”一节第 262 页

GetValue 方法

返回以本地格式表示的指定列的值。

语法

Visual Basic

```
Public Overrides Function GetValue( _
  ByVal columnID As Integer _
) As Object
```

C#

```
public override object GetValue(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

最适合于此列的 .NET 类型的列值；如果列为 NULL，则为 DBNull。

注释

此方法在功能上与 ULDataReader.this[int] 相同。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- “FieldCount 属性”一节第 262 页
- “Item(Int32) 属性”一节第 264 页

GetValues 方法

返回当前行的所有列值。

语法

Visual Basic

```
Public Overrides Function GetValues( _
  ByVal values As Object() _
) As Integer
```

C#

```
public override int GetValues(
  object[] values
);
```

参数

- **values** 用于保存整个行的 System.Objects 数组。

返回值

检索的列值的数目。如果数组长度超过列数 (ULDataReader.FieldCount)，则仅检索 FieldCount 项，并且数组的剩余部分保持不变。

注释

对大多数应用程序而言，GetValues 方法提供了一种有效率的检索所有列而不是分别检索每个列的方法。

您可以传递一个包含的列数比生成的行中包含的列数少的 System.Object 数组。只有 System.Object 数组保存的数据才会复制到数组中。您还可以传递长度比生成的行中包含的列数大的 System.Object 数组。

对于值为 NULL 的数据库列，此方法返回 DBNull。对于其它列，它返回本机格式的列值。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “FieldCount 属性”一节第 262 页
- “GetFieldType 方法”一节第 277 页
- “GetValue 方法”一节第 289 页
- 对象

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

IsDBNull 方法

检查指定列的值是否为 NULL。

语法

Visual Basic
```
Public Overrides Function IsDBNull( _
  ByVal columnID As Integer _
) As Boolean
```

C#
```
public override bool IsDBNull(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

返回值

如果值为 NULL，则返回 true；如果值不为 NULL，则返回 false。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- “FieldCount 属性”一节第 262 页

MoveAfterLast 方法

UL Ext.: 将游标定位到游标的最后一行之后。

语法

Visual Basic
```
Public Sub MoveAfterLast()
```

C#
```
public void MoveAfterLast();
```

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页

MoveBeforeFirst 方法

UL Ext.: 将游标定位到游标的第一行之前。

语法

Visual Basic
Public Sub **MoveBeforeFirst()**

C#
public void **MoveBeforeFirst();**

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页

MoveFirst 方法

UL Ext.: 将游标定位到游标的第一行。

语法

Visual Basic
Public Function **MoveFirst()** As Boolean

C#
public bool **MoveFirst();**

返回值

如果成功则返回 true，否则返回 false。例如，如果没有行，此方法将失败。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页

MoveLast 方法

UL Ext.: 将游标定位到游标的最后一行。

语法

Visual Basic
Public Function **MoveLast()** As Boolean

C#
public bool **MoveLast();**

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

返回值

如果成功则返回 true，否则返回 false。例如，如果没有行，此方法将失败。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页

MoveNext 方法

UL Ext.: 将游标定位到下一行；如果游标已经位于最后一行，则定位到最后一行之后。

语法

Visual Basic
Public Function **MoveNext()** As Boolean

C#
public bool **MoveNext();**

返回值

如果成功则返回 true，否则返回 false。例如，如果没有其它行，该方法将失败。

注释

此方法与 ULDataReader.Read 方法相同。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “Read 方法”一节第 295 页

MovePrevious 方法

UL Ext.: 将游标定位到上一行，或第一行之前。

语法

Visual Basic
Public Function **MovePrevious()** As Boolean

C#
public bool **MovePrevious();**

返回值

如果成功则返回 true，否则返回 false。例如，如果没有其它行，该方法将失败。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页

MoveRelative 方法

UL Ext.: 相对于当前行定位游标。

语法

Visual Basic

```
Public Function MoveRelative( _
  ByVal offset As Integer _
) As Boolean
```

C#

```
public bool MoveRelative(
  int offset
);
```

参数

- **offset** 要移动的行数。负值对应于向后移动。

返回值

如果成功则返回 true，否则返回 false。例如，如果游标位于第一行或最后一行之外，则此方法失败。

注释

如果行不存在，则此方法返回 false，

如果 *offset* 为正值，游标位置在最后一行 (ULDataReader.IsEOF) 之后；

如果 *offset* 为负值，游标位置在第一行 (ULDataReader.IsBOF) 之前。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “IsEOF 属性”一节第 264 页
- “IsBOF 属性”一节第 263 页

NextResult 方法

读取批处理 SQL 语句的结果时，将 ULDataReader 推进到下一结果。

语法

Visual Basic

```
Public Overrides Function NextResult() As Boolean
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

C#
public override bool **NextResult();**

返回值

如果还有其它结果集则为 true；否则为 false。对于 UltraLite.NET，始终返回 false。

注释

UL Ext.: UltraLite.NET 不支持批处理 SQL 语句；ULDataReader 始终被定位到第一个（也是仅有的一个）结果集。调用 NextResult 没有任何作用。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页

Read 方法

将游标定位到下一行；如果游标已经位于最后一行，则定位到最后一行之后。

语法

Visual Basic
Public Overrides Function **Read()** As Boolean

C#
public override bool **Read();**

返回值

如果成功则返回 true，否则返回 false。例如，如果没有其它行，该方法将失败。

注释

此方法与 ULDataReader.MoveNext 方法相同。

另请参见

- “ULDataReader 类”一节第 257 页
- “ULDataReader 成员”一节第 258 页
- “MoveNext 方法”一节第 293 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULDateOrder 枚举

UL Ext.: 枚举数据库可支持的日期顺序。

语法

Visual Basic
Public Enum **ULDateOrder**

C#
public enum **ULDateOrder**

成员

成员名称	说明	值
DMY	日、月、年。	2
MDY	月、日、年。	1
YMD	年、月、日。	0

另请参见

- “DateOrder 属性”一节第 210 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULDbType 枚举

枚举 UltraLite.NET 数据库数据类型。

语法

Visual Basic
Public Enum **ULDbType**

C#
public enum **ULDbType**

注释

下表列出了与每种 ULDbType 兼容的 .NET 类型。对于整型而言，表列始终都可以使用较小的整型来进行设置；但只要实际值在该类型取值范围之内，也可以使用较大的整型来进行设置。

ULDbType	兼容的 .NET 类型	C# 内置类型	Visual Basic 内置类型
Binary, **VarBinary**	System.Byte[]，或 System.Guid（如果大小为 16）	byte[]	Byte()
Bit	System.Boolean	bool	Boolean
Char, **VarChar**	System.String	String	String
Date	System.DateTime	DateTime 无内置类型。	Date
Double	System.Double	double	Double
LongBinary	System.Byte[]	byte[]	Byte()
LongVarchar	System.String	String	String
Decimal, **Numeric**	System.String	decimal	Decimal
Float, **Real**	System.Single	float	Single
BigInt	System.Int64	long	Long
Integer	System.Int32	int	Integer
SmallInt	System.Int16	short	Short
Time	System.TimeSpan	TimeSpan 无内置类型。	TimeSpan 无内置类型。

ULDbType	兼容的 .NET 类型	C# 内置类型	Visual Basic 内置类型
DateTime, **TimeStamp**	System.DateTime	DateTime 无内置类型。	Date
TinyInt	System.Byte	byte	Byte
UnsignedBigInt	System.UInt64	ulong	UInt64 无内置类型。
UnsignedInt	System.UInt32	uint	UInt32 无内置类型。
UnsignedSmallInt	System.UInt16	ushort	UInt16 无内置类型。
UniqueIdentifier	System.Guid	Guid 无内置类型。	Guid 无内置类型。

长度为 16 的二进制列与 UniqueIdentifier 类型完全兼容。

成员

成员名称	说明	值
BigInt	有符号 64 位整数。	5
Binary	二进制数据，具有指定的最大长度。枚举值 **Binary** 和 **VarBinary** 互为对方的别名。	15
Bit	1 位标志。	8
Char	字符数据，具有指定长度。在 UltraLite.NET 中，此类型始终支持 Unicode 字符。类型 **Char** 和 **VarChar** 完全兼容。	0
Date	日期信息。	10
DateTime	时间戳信息（日期、时间）。枚举值 **DateTime** 和 **TimeStamp** 互为对方的别名。	9
Decimal	精确数字数据，具有指定的精度和小数位数。枚举值 **Decimal** 和 **Numeric** 互为对方的别名。	14
Double	双精度浮点数（8 个字节）。	12

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
Float	单精度浮点数（4 个字节）。枚举值 **Float** 和 **Real** 互为对方的别名。	13
Integer	无符号 32 位整数。	1
LongBinary	二进制数据，具有可变长度。	18
LongVarchar	字符数据，具有可变长度。在 UltraLite.NET 中，此类型始终支持 Unicode 字符。	17
Numeric	精确数字数据，具有指定的精度和小数位数。枚举值 **Decimal** 和 **Numeric** 互为对方的别名。	14
Real	单精度浮点数（4 个字节）。枚举值 **Float** 和 **Real** 互为对方的别名。	13
SmallInt	有符号 16 位整数。	3
Time	时间信息。	11
TimeStamp	时间戳信息（日期、时间）。枚举值 **DateTime** 和 **TimeStamp** 互为对方的别名。	9
TinyInt	无符号 8 位整数。	7
UniqueIdentifier	通用唯一标识符（Universally Unique Identifier，简称 UUID/GUID）。	19
UnsignedBigInt	无符号 64 位整数。	6
UnsignedInt	无符号 32 位整数。	2
UnsignedSmallInt	无符号 16 位整数。	4
VarBinary	二进制数据，具有指定的最大长度。枚举值 **Binary** 和 **VarBinary** 互为对方的别名。	15
VarChar	字符数据，具有指定的最大长度。在 UltraLite.NET 中，此类型始终支持 Unicode 字符。类型 **Char** 和 **VarChar** 完全兼容。	16

另请参见

- “GetFieldType 方法”一节第 277 页
- “GetDataTypeName 方法”一节第 274 页
- “GetColumnULDbType 方法”一节第 224 页
- Byte
- Guid
- Boolean
- String
- DateTime
- Single
- Int64
- Int32
- Int16
- TimeSpan
- UInt64
- UInt32
- UInt16

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULDBValid 枚举

枚举 UltraLite.NET 数据库的校验方法。

语法

Visual Basic
Public Enum **ULDBValid**

C#
public enum **ULDBValid**

注释

有关校验数据库的信息，请参见“ValidateDatabase 方法”一节第 245 页。

成员

成员名称	值
EXPRESS_VALIDATE	0
FULL_VALIDATE	1

另请参见

- “ValidateDatabase 方法”一节第 245 页
- “ValidateDatabase 方法”一节第 173 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULException 类

表示由 UltraLite.NET 数据库返回的 SQL 错误。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULException**
  Inherits ApplicationException

C#
public sealed class **ULException**: ApplicationException

注释

指示错误的 SQLCODE 会在 NativeError 中返回。

在 .NET Compact Framework 中，此类不可序列化。

另请参见

- “ULException 成员”一节第 302 页
- “NativeError 属性”一节第 304 页

ULException 成员

公共构造函数

成员名称	说明
“ULException 构造函数”一节第 303 页	使用给定的错误代码创建 ULException。

公共属性

成员名称	说明
Data（继承自 Exception）	获取可提供有关异常的附加用户定义信息的键/值对集合。
HelpLink（继承自 Exception）	获取或设置可转到与此异常相关联的帮助文件的链接。
InnerException（继承自 Exception）	获取导致当前异常的异常实例。
Message（继承自 Exception）	获取描述当前异常的消息。
“NativeError 属性”一节第 304 页	返回由数据库返回的 SQL 代码。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“Source 属性”一节第 304 页	返回生成该错误的提供程序的名称。
StackTrace（继承自 Exception）	获取在抛出当前异常时调用堆栈上的框架的字符串表示形式。
TargetSite（继承自 Exception）	获取抛出当前异常的方法。

公共方法

成员名称	说明
GetBaseException（继承自 Exception）	当在派生类中被替换时，返回导常，它是导致产生一个或多个后续异常的根本原因。
“GetObjectData 方法”一节第 305 页	用序列化此 ULException 所需的数据填充 SerializationInfo。
GetType（继承自 Exception）	获取当前实例的运行时类型。
ToString（继承自 Exception）	创建并返回当前异常的字符串表示形式。

另请参见

- “ULException 类”一节第 302 页
- “NativeError 属性”一节第 304 页

ULException 构造函数

使用给定的错误代码创建 ULException。

语法

Visual Basic

```
Public Sub New( _
   ByVal code As ULSQLCode, _
   ByVal s1 As String, _
   ByVal s2 As String, _
   ByVal s3 As String _
)
```

C#

```
public  ULException(
   ULSQLCode code,
   string  s1,
   string  s2,
   string  s3
);
```

参数

- **code** 异常的代码。
- **s1** 格式化消息的第一个字符串。
- **s2** 格式化消息的第二个字符串。
- **s3** 格式化消息的第三个字符串。

注释

对应于指定的 iAnywhere.Data.UltraLite.ULSQLCode 的消息字符串是从 **iAnywhere.Data.UltraLite.resources** 程序集中检索到的。按照语言的如下顺序搜索资源：CultureInfo.CurrentUICulture，然后 CultureInfo.CurrentCulture，最后是 "EN"。

另请参见

- “ULException 类”一节第 302 页
- “ULException 成员”一节第 302 页
- “ULSQLCode 枚举”一节第 441 页
- CultureInfo.CurrentUICulture
- CultureInfo.CurrentCulture

NativeError 属性

返回由数据库返回的 SQLCODE。

语法

Visual Basic
Public Readonly Property **NativeError** As ULSQLCode

C#
public ULSQLCode **NativeError** { get;}

属性值

数据库返回的 ULSQLCode 值。

另请参见

- “ULException 类”一节第 302 页
- “ULException 成员”一节第 302 页
- “ULSQLCode 枚举”一节第 441 页

Source 属性

返回生成该错误的提供程序的名称。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Overrides Readonly Property **Source** As String

C#
public override string **Source** { get;}

属性值

标识 UltraLite.NET 作为提供程序的字符串值。

另请参见

- “ULException 类”一节第 302 页
- “ULException 成员”一节第 302 页

GetObjectData 方法

用序列化此 ULException 所需的数据填充 SerializationInfo。

语法

Visual Basic
```
Public Overrides Sub GetObjectData( _
  ByVal info As SerializationInfo, _
  ByVal context As StreamingContext _
)
```

C#
```
public override void GetObjectData(
  SerializationInfo info,
  StreamingContext context
);
```

参数

- **info** 要用数据填充的 SerializationInfo。
- **context** 此序列化的目标。

注释

.NET Compact Framework 不支持此方法。

另请参见

- “ULException 类”一节第 302 页
- “ULException 成员”一节第 302 页

ULFactory 类

表示一组方法，这些方法用于创建 iAnywhere.Data.UltraLite 提供程序对数据源类实现的实例。此类属于静态类，因此无法继承，也无法将其实例化。

语法

Visual Basic
Public NotInheritable Class **ULFactory**
  Inherits DbProviderFactory

C#
public sealed class **ULFactory**: DbProviderFactory

注释

ADO.NET 2.0 添加了两个新类：System.Data.Common.DbProviderFactories 类和 System.Data.Common.DbProviderFactory 类，以简化提供程序无关代码的编写。要将它们与 UltraLite.NET 配合使用，请将 iAnywhere.Data.UltraLite 指定为传递给 GetFactory 的提供程序固定名称。例如：

```
' Visual Basic
Dim factory As DbProviderFactory =
  DbProviderFactories.GetFactory( "iAnywhere.Data.UltraLite" )
Dim conn As DbConnection =
  factory.CreateConnection()

// C#
DbProviderFactory factory =
        DbProviderFactories.GetFactory( "iAnywhere.Data.UltraLite" );
DbConnection conn = factory.CreateConnection();
```

在此示例中，以 ULConnection 对象形式创建 conn。

有关 ADO.NET 2.0 中提供程序工厂和通用编程的说明，请参见 http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnvs05/html/vsgenerics.asp。

UltraLite.NET 不支持 CreateCommandBuilder()、CreateDataSourceEnumerator() 和 CreatePermission()。

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULFactory 类。

Inherits: System.Data.Common.DbProviderFactory

另请参见

- “ULFactory 成员”一节第 307 页
- DbProviderFactories
- DbProviderFactory

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULFactory 成员

公共字段

成员名称	说明
“Instance 字段”一节 第 308 页	表示 ULFactory 类的单个实例。此字段为只读字段。

公共属性

成员名称	说明
“CanCreateDataSourceEnumerator 属性”一节第 308 页	返回 false 表示 UltraLite.NET 不支持 DbDataSourceEnumerator。

公共方法

成员名称	说明
“CreateCommand 方法”一节 第 309 页	返回强类型的 System.Data.Common.DbCommand 实例。
“CreateCommandBuilder 方法”一节第 309 页	返回强类型的 System.Data.Common.DbCommandBuilder 实例。
“CreateConnection 方法”一节 第 310 页	返回强类型的 System.Data.Common.DbConnection 实例。
“CreateConnectionStringBuilder 方法”一节第 310 页	返回强类型的 System.Data.Common.DbConnectionStringBuilder 实例。
“CreateDataAdapter 方法”一节 第 310 页	返回强类型的 System.Data.Common.DbDataAdapter 实例。
CreateDataSourceEnumerator（继承自 DbProviderFactory）	返回提供程序的类的新实例，此实例实现 DbDataSourceEnumerator。
“CreateParameter 方法”一节 第 311 页	返回强类型的 System.Data.Common.DbParameter 实例。
CreatePermission（继承自 DbProviderFactory）	返回提供程序的类的新实例，此实例实现提供程序版本的 CodeAccessPermission。

另请参见

- “ULFactory 类”一节第 306 页
- DbProviderFactories
- DbProviderFactory

Instance 字段

表示 ULFactory 类的单个实例。此字段为只读字段。

语法

Visual Basic
Public Shared Readonly **Instance** As ULFactory

C#
public const ULFactory **Instance**;

注释

ULFactory 为单个类，这意味着只能存在此类的这一实例。

通常不会直接使用此字段，而是使用 System.Data.Common.DbProviderFactories.GetFactory(String) 获取对此 ULFactory 实例的引用。有关示例内容，请参见“ULFactory 类”一节第 306 页。

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULFactory 类。

另请参见

- “ULFactory 类”一节第 306 页
- “ULFactory 成员”一节第 307 页
- DbProviderFactories.GetFactory

CanCreateDataSourceEnumerator 属性

返回 false 表示 UltraLite.NET 不支持 DbDataSourceEnumerator。

语法

Visual Basic
Public Overrides Readonly Property **CanCreateDataSourceEnumerator** As Boolean

C#
public override bool **CanCreateDataSourceEnumerator** { get;}

属性值

False，表示 ULFactory 不实现 CreateDataSourceEnumerator() 方法。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULFactory 类”一节第 306 页
- “ULFactory 成员”一节第 307 页

CreateCommand 方法

返回强类型的 System.Data.Common.DbCommand 实例。

语法

Visual Basic
Public Overrides Function **CreateCommand()** As DbCommand

C#
public override DbCommand **CreateCommand();**

返回值

一个类型为 DbCommand 的新 ULCommand 实例。

另请参见

- “ULFactory 类”一节第 306 页
- “ULFactory 成员”一节第 307 页
- DbCommand
- “ULCommand 类”一节第 86 页

CreateCommandBuilder 方法

返回强类型的 System.Data.Common.DbCommandBuilder 实例。

语法

Visual Basic
Public Overrides Function **CreateCommandBuilder()** As DbCommandBuilder

C#
public override DbCommandBuilder **CreateCommandBuilder();**

返回值

一个类型为 DbCommandBuilder 的新 ULCommandBuilder 实例。

另请参见

- “ULFactory 类”一节第 306 页
- “ULFactory 成员”一节第 307 页
- DbCommandBuilder
- “ULCommandBuilder 类”一节第 121 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

CreateConnection 方法

返回强类型的 System.Data.Common.DbConnection 实例。

语法

Visual Basic
Public Overrides Function **CreateConnection()** As DbConnection

C#
public override DbConnection **CreateConnection();**

返回值

一个类型为 DbConnection 的新 ULConnection 实例。

另请参见

- “ULFactory 类”一节第 306 页
- “ULFactory 成员”一节第 307 页
- DbConnection
- “ULConnection 类”一节第 131 页

CreateConnectionStringBuilder 方法

返回强类型的 System.Data.Common.DbConnectionStringBuilder 实例。

语法

Visual Basic
Public Overrides Function **CreateConnectionStringBuilder()** As DbConnectionStringBuilder

C#
public override DbConnectionstring Builder **CreateConnectionStringBuilder();**

返回值

一个类型为 DbConnectionStringBuilder 的新 ULConnectionStringBuilder 实例。

另请参见

- “ULFactory 类”一节第 306 页
- “ULFactory 成员”一节第 307 页
- DbConnectionStringBuilder
- “ULConnectionStringBuilder 类”一节第 189 页

CreateDataAdapter 方法

返回强类型的 System.Data.Common.DbDataAdapter 实例。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Overrides Function **CreateDataAdapter()** As DbDataAdapter

C#
public override DbDataAdapter **CreateDataAdapter();**

返回值

一个类型为 DbDataAdapter 的新 ULDataAdapter 实例。

另请参见

- “ULFactory 类”一节第 306 页
- “ULFactory 成员”一节第 307 页
- DbDataAdapter
- “ULDataAdapter 类”一节第 226 页

CreateParameter 方法

返回强类型的 System.Data.Common.DbParameter 实例。

语法

Visual Basic
Public Overrides Function **CreateParameter()** As DbParameter

C#
public override DbParameter **CreateParameter();**

返回值

一个类型为 DbParameter 的新 ULParameter 实例。

另请参见

- “ULFactory 类”一节第 306 页
- “ULFactory 成员”一节第 307 页
- DbParameter
- “ULParameter 类”一节第 354 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULFileTransfer 类

UL Ext.: 使用 MobiLink 服务器从远程数据库中传输文件。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULFileTransfer**

C#
public sealed class **ULFileTransfer**

注释

您无需数据库连接便能执行文件传输，但是，如果您的应用程序要将 UltraLite 数据库与 UltraLite 引擎运行时搭配使用，则必须在使用此 API 或任何其它 UltraLite.NET API 之前，将 ULDatabaseManager.RuntimeType 设置为适当的值。

要传输文件，必须设置 ULFileTransfer.FileName、ULFileTransfer.Stream、ULFileTransfer.UserName 和 ULFileTransfer.Version。

另请参见

- “ULFileTransfer 成员”一节第 312 页
- “RuntimeType 属性”一节第 239 页
- “FileName 属性”一节第 318 页
- “Stream 属性”一节第 320 页
- “UserName 属性”一节第 323 页
- “Version 属性”一节第 323 页

ULFileTransfer 成员

公共构造函数

成员名称	说明
“ULFileTransfer 构造函数”一节第 314 页	初始化 ULFileTransfer 对象。对数据库进行任何操作前必须打开该连接。

公共属性

成员名称	说明
“AuthStatus 属性”一节第 315 页	返回上次尝试传输文件时的授权状态码。
“AuthValue 属性”一节第 315 页	返回来自自定义用户验证同步脚本的返回值。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“AuthenticationParms 属性”一节第 315 页	为自定义用户验证脚本（MobiLink authenticate_parameters 连接事件）指定参数。
“DestinationFileName 属性”一节第 316 页	指定已下载文件的本地文件名。
“DestinationPath 属性”一节第 316 页	指定文件的下载位置。
“DownloadedFile 属性”一节第 317 页	检查在上次尝试传输文件期间是否确实下载了文件。
“FileAuthCode 属性”一节第 317 页	返回在上次尝试传输文件期间来自 authenticate_file_transfer 脚本的返回值。
“FileName 属性”一节第 318 页	指定要下载文件的名称。
“ForceDownload 属性”一节第 318 页	指定是否强制下载文件（如果该文件存在）。
“Password 属性”一节第 319 页	用 UserName 指定的用户的 MobiLink 口令。
“ResumePartialDownload 属性”一节第 320 页	指定是恢复还是放弃先前的部分下载。
“Stream 属性”一节第 320 页	指定用于文件传输的 MobiLink 同步流。
“StreamErrorCode 属性”一节第 321 页	返回在上次尝试传输文件期间由流本身报告的错误。
“StreamErrorSystem 属性”一节第 321 页	返回流错误的系统特定代码。
“StreamParms 属性”一节第 322 页	指定用于配置同步流的参数。
“UserName 属性”一节第 323 页	向 MobiLink 服务器唯一标识 MobiLink 客户端的用户名。
“Version 属性”一节第 323 页	指定要使用哪一个同步脚本。

公共方法

成员名称	说明
“DownloadFile 方法”一节 第 324 页	下载此对象的属性所指定的文件。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “RuntimeType 属性”一节第 239 页
- “FileName 属性”一节第 318 页
- “Stream 属性”一节第 320 页
- “UserName 属性”一节第 323 页
- “Version 属性”一节第 323 页

ULFileTransfer 构造函数

初始化 ULFileTransfer 对象。对数据库进行任何操作前必须打开该连接。

语法

Visual Basic
Public Sub **New()**

C#
public **ULFileTransfer();**

注释

您无需数据库连接便能执行文件传输，但是，如果您的应用程序要将 UltraLite 数据库与 UltraLite 引擎运行时搭配使用，则必须在使用此 API 或任何其它 UltraLite.NET API 之前，将 ULDatabaseManager.RuntimeType 设置为适当的值。

ULFileTransfer 对象需要先设置 ULFileTransfer.FileName、ULFileTransfer.Stream、ULFileTransfer.UserName 和 ULFileTransfer.Version，然后才能传输文件。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “DownloadFile() 方法”一节第 324 页
- “RuntimeType 属性”一节第 239 页
- “FileName 属性”一节第 318 页
- “Stream 属性”一节第 320 页
- “UserName 属性”一节第 323 页
- “Version 属性”一节第 323 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

AuthStatus 属性

返回上次尝试传输文件时的授权状态码。

语法

Visual Basic
Public Readonly Property **AuthStatus** As ULAuthStatusCode

C#
public ULAuthStatusCode **AuthStatus** { get;}

属性值

ULAuthStatusCode 值之一，用于表示上次尝试传输文件时的授权状态。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “ULAuthStatusCode 枚举”一节第 56 页

AuthValue 属性

返回来自自定义用户验证同步脚本的返回值。

语法

Visual Basic
Public Readonly Property **AuthValue** As Long

C#
public long **AuthValue** { get;}

属性值

从自定义用户验证同步脚本返回的长整数。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页

AuthenticationParms 属性

为自定义用户验证脚本（MobiLink authenticate_parameters 连接事件）指定参数。

语法

Visual Basic
Public Property **AuthenticationParms** As String

C#
public string **AuthenticationParms** { get; set; }

属性值

字符串数组，每个字符串都包含一个验证参数（空数组项将导致同步错误）。缺省值为空值引用（在 Visual Basic 中为 Nothing），表示没有验证参数。

注释

只使用前 255 个字符串，且每个字符串的长度不应超过 128 个字符（长字符串在发送到 MobiLink 服务器时会被截断）。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页

DestinationFileName 属性

指定已下载文件的本地文件名。

语法

Visual Basic
Public Property **DestinationFileName** As String

C#
public string **DestinationFileName** { get; set; }

属性值

用于指定已下载文件本地文件名的字符串。如果值为空值引用（在 Visual Basic 中为 Nothing），则使用 FileName。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “FileName 属性”一节第 318 页

DestinationPath 属性

指定文件的下载位置。

语法

Visual Basic
Public Property **DestinationPath** As String

C#
public string **DestinationPath** { get; set; }

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

属性值

用于指定文件目标目录的字符串。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

注释

缺省目标目录因设备的操作系统而异：

- 对于 Windows Mobile 设备，如果 DestinationPath 为空值引用（在 Visual Basic 中为 Nothing），则文件存储于根目录 (\)。
- 对于桌面应用程序，如果 DestinationPath 为空值引用（在 Visual Basic 中为 Nothing），则文件存储于当前目录。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “ForceDownload 属性”一节第 318 页
- “DownloadFile() 方法”一节第 324 页

DownloadedFile 属性

检查在上次尝试传输文件期间是否确实下载了文件。

语法

Visual Basic
Public Readonly Property **DownloadedFile** As Boolean

C#
public bool **DownloadedFile** { get;}

属性值

如果文件已下载，则为 true；否则为 false。

注释

如果在调用 DownloadFile() 时文件已是最新，则它将返回 true，但 DownloadedFile 将为 false。如果发生错误且 DownloadFile() 返回 false，则 DownloadedFile 将为 false。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “DownloadFile() 方法”一节第 324 页
- “DownloadFile() 方法”一节第 324 页

FileAuthCode 属性

返回在上次尝试传输文件期间来自 authenticate_file_transfer 脚本的返回值。

语法

Visual Basic
Public Readonly Property **FileAuthCode** As UInt16

C#
public ushort **FileAuthCode** { get;}

属性值

在上次尝试传输文件期间从 authenticate_file_transfer 脚本返回的无符号短整数。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页

FileName 属性

指定要下载文件的名称。

语法

Visual Basic
Public Property **FileName** As String

C#
public string **FileName** { get; set; }

属性值

字符串，用于指定由 MobiLink 服务器所识别的文件的名称。此属性没有缺省值，因此必须进行显式设置。

注释

FileName 是正在运行 MobiLink 的服务器上文件的名称。MobiLink 将首先在 UserName 子目录中搜索此文件，然后会在根目录中进行搜索（根目录是通过 MobiLink 服务器的 -ftr 选项指定的）。FileName 不得包含任何驱动器和路径信息，否则，MobiLink 服务器将无法找到该文件。例如，"myfile.txt" 有效，而 "somedir\myfile.txt"、"..\myfile.txt" 和 "c:\myfile.txt" 均无效。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “DestinationFileName 属性”一节第 316 页
- “DownloadFile() 方法”一节第 324 页

ForceDownload 属性

指定是否强制下载文件（如果该文件存在）。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Property **ForceDownload** As Boolean

C#
public bool **ForceDownload** { get; set; }

属性值

值为 true 强制下载文件，值为 false 则执行常规下载（检查文件是否存在）。缺省值为 false。

注释

如果 ForceDownload 为 true，则即使未对文件进行更改，也将始终下载该文件，且会放弃任何现有的部分下载。如果 ForceDownload 为 false，则仅当服务器的文件版本不同于客户端的文件版本时，才会下载文件。请注意，无论是在客户端还是在服务器上对文件进行了更改，将 ForceDownload 指定为 true 都会导致服务器版本覆盖客户端版本。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “ResumePartialDownload 属性”一节第 320 页
- “DownloadFile() 方法”一节第 324 页

Password 属性

用 UserName 指定的用户的 MobiLink 口令。

语法

Visual Basic
Public Property **Password** As String

C#
public string **Password** { get; set; }

属性值

用于指定 MobiLink 口令的字符串。缺省值为空值引用（在 Visual Basic 中为 Nothing），表示没有指定口令。

注释

MobiLink 用户名和口令不同于任何数据库用户 ID 和口令，它们用于向 MobiLink 服务器标识和验证应用程序。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “UserName 属性”一节第 323 页
- “DownloadFile() 方法”一节第 324 页

ResumePartialDownload 属性

指定是恢复还是放弃先前的部分下载。

语法

Visual Basic
Public Property **ResumePartialDownload** As Boolean

C#
public bool **ResumePartialDownload** { get; set; }

属性值

值为 true，则恢复先前的部分下载，值为 false，则放弃先前的部分下载。缺省值为 false。

注释

UltraLite.NET 能够通过 ULFileTransferListener 重新启动由于通信错误或用户中止而失败的下载。UltraLite.NET 在接收下载的过程中对其进行处理。如果下载被中断，则会保留部分下载文件，并会在下一次文件传输过程中将其恢复。

如果文件在服务器上已更新，则会放弃部分下载并启动新下载。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “ForceDownload 属性”一节第 318 页
- “DownloadFile() 方法”一节第 324 页

Stream 属性

指定用于文件传输的 MobiLink 同步流。

语法

Visual Basic
Public Property **Stream** As ULStreamType

C#
public ULStreamType **Stream** { get; set; }

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

属性值

ULStreamType 值之一，用于指定要使用的同步流类型。缺省值为 ULStreamType.TCPIP。

注释

大多数同步流都需要用一些参数来标识 MobiLink 服务器地址和控制其它行为。这些参数由 ULFileTransfer.StreamParms 提供。

如果设置的流类型值在平台上无效，则流类型将被设置为 ULStreamType.TCPIP。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “ULStreamType 枚举”一节第 476 页
- “StreamParms 属性”一节第 322 页
- “DownloadFile() 方法”一节第 324 页
- “ULStreamType 枚举”一节第 476 页

StreamErrorCode 属性

返回在上次尝试传输文件期间由流本身报告的错误。

语法

Visual Basic
Public Readonly Property **StreamErrorCode** As ULStreamErrorCode

C#
public ULStreamErrorCode **StreamErrorCode** { get;}

属性值

表示流本身报告的错误的 ULStreamErrorCode 值之一；如果没有发生错误，则为 ULStreamErrorCode.NONE。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “ULStreamErrorCode 枚举”一节第 458 页
- “ULStreamErrorCode 枚举”一节第 458 页

StreamErrorSystem 属性

返回流错误的系统特定代码。

语法

Visual Basic
Public Readonly Property **StreamErrorSystem** As Integer

C#
public int **StreamErrorSystem** { get;}

属性值

一个整数，用于表示流错误的系统特定代码。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页

StreamParms 属性

指定用于配置同步流的参数。

语法

Visual Basic
Public Property **StreamParms** As String

C#
public string **StreamParms** { get; set; }

属性值

用于指定流参数的字符串，以分号分隔的 "关键字-值" 对的列表形式表示。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

注释

有关配置特定流类型的信息，请参见“UltraLite 同步流的网络协议选项”一节 《UltraLite - 数据库管理和参考》。

StreamParms 是一个包含用于同步流的所有参数的字符串。参数是以分号分隔的 "名称=值" 对列表 ("param1=value1;param2=value2") 的形式指定的。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “Stream 属性”一节第 320 页
- “DownloadFile() 方法”一节第 324 页
- “ULStreamType 枚举”一节第 476 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

UserName 属性

向 MobiLink 服务器唯一标识 MobiLink 客户端的用户名。

语法

Visual Basic
Public Property **UserName** As String

C#
public string **UserName** { get; set; }

属性值

用于指定用户名的字符串。此属性没有缺省值，因此必须进行显式设置。

注释

MobiLink 服务器使用此值来查找要下载的文件。MobiLink 用户名和口令不同于任何数据库用户 ID 和口令，它们用于向 MobiLink 服务器标识和验证应用程序。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “Password 属性”一节第 319 页
- “DownloadFile() 方法”一节第 324 页

Version 属性

指定要使用哪一个同步脚本。

语法

Visual Basic
Public Property **Version** As String

C#
public string **Version** { get; set; }

属性值

用于指定要使用的同步脚本版本的字符串。此属性没有缺省值，因此必须进行显式设置。

注释

统一数据库中的每个同步脚本都带有版本字符串标记。版本字符串使 UltraLite 应用程序可以从一组同步脚本中进行选择。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “DownloadFile() 方法”一节第 324 页

DownloadFile 方法

下载此对象的属性所指定的文件。

DownloadFile() 方法

下载此对象的属性所指定的文件。

语法

Visual Basic
Public Function **DownloadFile()** As Boolean

C#
public bool **DownloadFile();**

返回值

如果成功则返回 true，否则返回 false（检查 ULFileTransfer.StreamErrorCode 以及其它状态属性以找到原因）。

注释

MobiLink 服务器使用 ULFileTransfer.Stream、ULFileTransfer.UserName、ULFileTransfer.Password 和 ULFileTransfer.Version 将 ULFileTransfer.FileName 所指定的文件下载到 ULFileTransfer.DestinationPath。影响下载的其它属性有 ULFileTransfer.DestinationFileName、ULFileTransfer.AuthenticationParms、ULFileTransfer.ForceDownload 和 ULFileTransfer.ResumePartialDownload。

为避免文件损坏，UltraLite.NET 将会下载到临时文件，且仅在下载完成后才会替换目标文件。

在此对象的 ULFileTransfer.AuthStatus、ULFileTransfer.AuthValue、ULFileTransfer.FileAuthCode、ULFileTransfer.DownloadedFile、ULFileTransfer.StreamErrorCode 和 ULFileTransfer.StreamErrorSystem 中将报告详细的结果状态。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “DownloadFile 方法”一节第 324 页
- “DownloadFile(ULFileTransferProgressListener) 方法”一节第 325 页
- “FileName 属性”一节第 318 页
- “DestinationPath 属性”一节第 316 页
- “Stream 属性”一节第 320 页
- “UserName 属性”一节第 323 页
- “Password 属性”一节第 319 页
- “Version 属性”一节第 323 页
- “DestinationFileName 属性”一节第 316 页
- “AuthenticationParms 属性”一节第 315 页
- “ForceDownload 属性”一节第 318 页
- “ResumePartialDownload 属性”一节第 320 页
- “AuthStatus 属性”一节第 315 页
- “AuthValue 属性”一节第 315 页
- “FileAuthCode 属性”一节第 317 页
- “DownloadedFile 属性”一节第 317 页
- “StreamErrorCode 属性”一节第 321 页
- “StreamErrorSystem 属性”一节第 321 页
- “StreamErrorCode 属性”一节第 321 页

DownloadFile(ULFileTransferProgressListener) 方法

下载由此对象的属性所指定的文件，并将进度事件发布到指定监听器。

语法

Visual Basic

```
Public Function DownloadFile( _
  ByVal listener As ULFileTransferProgressListener _
) As Boolean
```

C#

```
public bool DownloadFile(
  ULFileTransferProgressListener listener
);
```

参数

- **listener** 接收文件传输进度事件的对象。

返回值

如果成功则返回 true，否则返回 false（检查 ULFileTransfer.StreamErrorCode 以及其它状态属性以找到原因）。

注释

MobiLink 服务器使用 ULFileTransfer.Stream、ULFileTransfer.UserName、ULFileTransfer.Password 和 ULFileTransfer.Version 将 ULFileTransfer.FileName 所指定的文件下载到 ULFileTransfer.DestinationPath。影响下载的其它属性有 ULFileTransfer.DestinationFileName、ULFileTransfer.AuthenticationParms、ULFileTransfer.ForceDownload 和 ULFileTransfer.ResumePartialDownload。

为避免文件损坏，UltraLite.NET 将会下载到临时文件，且仅在下载完成后才会替换目标文件。

在此对象的 ULFileTransfer.AuthStatus、ULFileTransfer.AuthValue、ULFileTransfer.FileAuthCode、ULFileTransfer.DownloadedFile、ULFileTransfer.StreamErrorCode 和 ULFileTransfer.StreamErrorSystem 中将报告详细的结果状态。

如果发生错误，可能会导致不会向监听器发送任何数据。

另请参见

- “ULFileTransfer 类”一节第 312 页
- “ULFileTransfer 成员”一节第 312 页
- “DownloadFile 方法”一节第 324 页
- “FileName 属性”一节第 318 页
- “DestinationPath 属性”一节第 316 页
- “Stream 属性”一节第 320 页
- “UserName 属性”一节第 323 页
- “Password 属性”一节第 319 页
- “Version 属性”一节第 323 页
- “DestinationFileName 属性”一节第 316 页
- “AuthenticationParms 属性”一节第 315 页
- “ForceDownload 属性”一节第 318 页
- “ResumePartialDownload 属性”一节第 320 页
- “AuthStatus 属性”一节第 315 页
- “AuthValue 属性”一节第 315 页
- “FileAuthCode 属性”一节第 317 页
- “DownloadedFile 属性”一节第 317 页
- “StreamErrorCode 属性”一节第 321 页
- “StreamErrorSystem 属性”一节第 321 页
- “StreamErrorCode 属性”一节第 321 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULFileTransferProgressData 类

UL Ext.: 返回文件传输进度监控数据。

语法

Visual Basic
Public Class **ULFileTransferProgressData**

C#
public class **ULFileTransferProgressData**

另请参见

- “ULFileTransferProgressData 成员”一节第 327 页
- “ULFileTransferProgressListener 接口”一节第 330 页

ULFileTransferProgressData 成员

公共字段

成员名称	说明
“FLAG_IS_BLOCKING 字段”一节第 328 页	一个标志，表示文件传输受到阻塞，正在等待来自 MobiLink 服务器的响应。此字段为常量并是只读字段。

公共属性

成员名称	说明
“BytesReceived 属性”一节第 328 页	返回到目前为止已接收的字节数。
“FileSize 属性”一节第 328 页	返回正在传输的文件的大小。
“Flags 属性”一节第 329 页	返回用于指示当前状态更多相关信息的当前文件传输标志。
“ResumedAtSize 属性”一节第 329 页	返回文件中恢复传输的位置。

另请参见

- “ULFileTransferProgressData 类”一节第 327 页
- “ULFileTransferProgressListener 接口”一节第 330 页

FLAG_IS_BLOCKING 字段

一个标志，表示文件传输受到阻塞，正在等待来自 MobiLink 服务器的响应。此字段为常量并是只读字段。

语法

Visual Basic
Public Shared **FLAG_IS_BLOCKING** As Integer

C#
public const int **FLAG_IS_BLOCKING**;

另请参见

- “ULFileTransferProgressData 类”一节第 327 页
- “ULFileTransferProgressData 成员”一节第 327 页

BytesReceived 属性

返回到目前为止已接收的字节数。

语法

Visual Basic
Public Readonly Property **BytesReceived** As UInt64

C#
public ulong **BytesReceived** { get;}

属性值

到目前为止已接收的字节数。

另请参见

- “ULFileTransferProgressData 类”一节第 327 页
- “ULFileTransferProgressData 成员”一节第 327 页

FileSize 属性

返回正在传输的文件的大小。

语法

Visual Basic
Public Readonly Property **FileSize** As UInt64

C#
public ulong **FileSize** { get;}

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

属性值

文件的大小（以字节为单位）。

另请参见

- “ULFileTransferProgressData 类”一节第 327 页
- “ULFileTransferProgressData 成员”一节第 327 页

Flags 属性

返回用于指示当前状态更多相关信息的当前文件传输标志。

语法

Visual Basic
Public Readonly Property **Flags** As Integer

C#
public int **Flags** { get;}

属性值

一个整数，其中包含由 OR 组合在一起的标志。

另请参见

- “ULFileTransferProgressData 类”一节第 327 页
- “ULFileTransferProgressData 成员”一节第 327 页
- “FLAG_IS_BLOCKING 字段”一节第 328 页

ResumedAtSize 属性

返回文件中恢复传输的位置。

语法

Visual Basic
Public Readonly Property **ResumedAtSize** As UInt64

C#
public ulong **ResumedAtSize** { get;}

属性值

以前传输的字节数。

另请参见

- “ULFileTransferProgressData 类”一节第 327 页
- “ULFileTransferProgressData 成员”一节第 327 页

ULFileTransferProgressListener 接口

UL Ext.: 用于接收文件传输进度事件的监听器接口。

语法

Visual Basic
Public Interface **ULFileTransferProgressListener**

C#
public interface **ULFileTransferProgressListener**

另请参见

- “ULFileTransferProgressListener 成员”一节第 330 页
- “DownloadFile(ULFileTransferProgressListener) 方法”一节第 325 页

ULFileTransferProgressListener 成员

公共方法

成员名称	说明
“FileTransferProgressed 方法”一节第 330 页	在文件传输期间调用它来向用户通知进度。此方法应返回 true（取消传输）或 false（继续传输）。

另请参见

- “ULFileTransferProgressListener 接口”一节第 330 页
- “DownloadFile(ULFileTransferProgressListener) 方法”一节第 325 页

FileTransferProgressed 方法

在文件传输期间调用它来向用户通知进度。此方法应返回 true（取消传输）或 false（继续传输）。

语法

Visual Basic
```
Public Function FileTransferProgressed( _
  ByVal data As ULFileTransferProgressData _
) As Boolean
```

C#
```
public bool FileTransferProgressed(
  ULFileTransferProgressData data
);
```

参数

- **data** 包含最新文件传输进度数据的 ULFileTransferProgressData 对象。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

返回值

此方法应返回 true（取消传输）或 false（继续传输）。

注释

在调用 FileTransferProgressed 的过程中，不应该调用任何 UltraLite.NET API 方法。

另请参见

- “ULFileTransferProgressListener 接口”一节第 330 页
- “ULFileTransferProgressListener 成员”一节第 330 页
- “ULFileTransferProgressData 类”一节第 327 页

ULIndexSchema 类

UL Ext.: 表示 UltraLite 表索引的模式。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULIndexSchema**

C#
public sealed class **ULIndexSchema**

注释

没有用于此类的构造函数。索引模式是使用 ULTableSchema 的 ULTableSchema.PrimaryKey、ULTableSchema.GetIndex(string) 和 ULTableSchema.GetOptimalIndex(int) 创建的。

另请参见

- “ULIndexSchema 成员”一节第 332 页
- “PrimaryKey 属性”一节第 537 页
- “GetIndex 方法”一节第 539 页
- “GetOptimalIndex 方法”一节第 540 页
- “ULTableSchema 类”一节第 533 页

ULIndexSchema 成员

公共属性

成员名称	说明
“ColumnCount 属性”一节第 333 页	返回索引中的列数。
“IsForeignKey 属性”一节第 334 页	检查索引是否为外键。
“IsForeignKeyCheckOnCommit 属性”一节第 334 页	检查是在提交时还是在插入和更新时执行外键的参照完整性。
“IsForeignKeyNullable 属性”一节第 335 页	检查外键是否可为空。
“IsOpen 属性”一节第 335 页	确定索引模式是处于打开状态还是关闭状态。
“IsPrimaryKey 属性”一节第 336 页	检查索引是否为主键。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“IsUniqueIndex 属性”一节第 336 页	检查索引是否唯一。
“IsUniqueKey 属性”一节第 337 页	检查索引是否为唯一键。
“Name 属性”一节第 337 页	返回索引的名称。
“ReferencedIndexName 属性”一节第 337 页	索引为外键时引用的主索引的名称。
“ReferencedTableName 属性”一节第 338 页	索引为外键时引用的主表的名称。

公共方法

成员名称	说明
“GetColumnName 方法”一节第 338 页	返回此索引中的 第 *colOrdinalInIndex* 列。
“IsColumnDescending 方法”一节第 339 页	检查索引是否按降序使用指定列。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “PrimaryKey 属性”一节第 537 页
- “GetIndex 方法”一节第 539 页
- “GetOptimalIndex 方法”一节第 540 页
- “ULTableSchema 类”一节第 533 页

ColumnCount 属性

返回索引中的列数。

语法

Visual Basic
Public Readonly Property **ColumnCount** As Short

C#
public short **ColumnCount** { get;}

属性值

索引中的列数。

注释

在索引中，列序号的范围是从 1 到 ColumnCount（含 1 和 ColumnCount）。

列序号和计数在模式升级过程中可能会发生变化。索引中的列序号与表或其它索引中的列 ID 不同，即使它们引用特定表中的同一个物理列。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “ULIndexSchema 成员”一节第 332 页

IsForeignKey 属性

检查索引是否为外键。

语法

Visual Basic
Public Readonly Property **IsForeignKey** As Boolean

C#
public bool **IsForeignKey** { get;}

属性值

如果索引是外键，则返回 true；如果索引不是外键，则返回 false。

注释

外键中的列可以引用另一个表的非空唯一索引。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “ULIndexSchema 成员”一节第 332 页

IsForeignKeyCheckOnCommit 属性

检查是在提交时还是在插入和更新时执行外键的参照完整性。

语法

Visual Basic
Public Readonly Property **IsForeignKeyCheckOnCommit** As Boolean

C#
public bool **IsForeignKeyCheckOnCommit** { get;}

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

属性值

如果在提交时检查参照完整性，则返回 true；如果在插入和更新时检查参照完整性，则返回 false。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “ULIndexSchema 成员”一节第 332 页
- “IsForeignKey 属性”一节第 334 页

IsForeignKeyNullable 属性

检查外键是否可为空。

语法

Visual Basic
Public Readonly Property **IsForeignKeyNullable** As Boolean

C#
public bool **IsForeignKeyNullable** { get;}

属性值

如果外键可为空，则返回 true；如果外键不可为空，则返回 false。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “ULIndexSchema 成员”一节第 332 页
- “IsForeignKey 属性”一节第 334 页

IsOpen 属性

确定索引模式是处于打开状态还是关闭状态。

语法

Visual Basic
Public Readonly Property **IsOpen** As Boolean

C#
public bool **IsOpen** { get;}

属性值

如果索引模式处于打开状态，则返回 true；否则返回 false。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “ULIndexSchema 成员”一节第 332 页

IsPrimaryKey 属性

检查索引是否为主键。

语法

Visual Basic
Public Readonly Property **IsPrimaryKey** As Boolean

C#
public bool **IsPrimaryKey** { get;}

属性值

如果索引是主键，则返回 true；如果索引不是主键，则返回 false。

注释

主键中的列不可以为空值。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “ULIndexSchema 成员”一节第 332 页

IsUniqueIndex 属性

检查索引是否唯一。

语法

Visual Basic
Public Readonly Property **IsUniqueIndex** As Boolean

C#
public bool **IsUniqueIndex** { get;}

属性值

如果索引唯一，返回 true；否则返回 false。

注释

唯一索引中的列可以为空值。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “ULIndexSchema 成员”一节第 332 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

IsUniqueKey 属性

检查索引是否为唯一键。

语法

Visual Basic
Public Readonly Property **IsUniqueKey** As Boolean

C#
public bool **IsUniqueKey** { get;}

属性值

如果索引是唯一键，则返回 true；如果索引不是唯一键，则返回 false。

注释

唯一键中的列不可以为空值。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “ULIndexSchema 成员”一节第 332 页

Name 属性

返回索引的名称。

语法

Visual Basic
Public Readonly Property **Name** As String

C#
public string **Name** { get;}

属性值

用于指定索引名称的字符串。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “ULIndexSchema 成员”一节第 332 页

ReferencedIndexName 属性

索引为外键时引用的主索引的名称。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Readonly Property **ReferencedIndexName** As String

C#
public string **ReferencedIndexName** { get;}

属性值

用于指定所引用主索引的名称的字符串。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “ULIndexSchema 成员”一节第 332 页
- “IsForeignKey 属性”一节第 334 页

ReferencedTableName 属性

索引为外键时引用的主表的名称。

语法

Visual Basic
Public Readonly Property **ReferencedTableName** As String

C#
public string **ReferencedTableName** { get;}

属性值

用于指定所引用主表的名称的字符串。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “ULIndexSchema 成员”一节第 332 页
- “IsForeignKey 属性”一节第 334 页

GetColumnName 方法

返回此索引中的

第 *colOrdinalInIndex* 列。

语法

Visual Basic
Public Function **GetColumnName(_**
  ByVal *colOrdinalInIndex* As Short _
) As String

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

C#
```
public string  GetColumnName(
  short colOrdinalInIndex
);
```

参数

- **colOrdinalInIndex**　索引中所需列的序号。值必须在 [1,ColumnCount] 范围内。

返回值

列的名称。

注释

列序号和计数在模式升级过程中可能会发生变化。索引中的列序号与表或其它索引中的列 ID 不同，即使它们引用特定表中的同一个物理列。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “ULIndexSchema 成员”一节第 332 页
- “ColumnCount 属性”一节第 333 页
- “ColumnCount 属性”一节第 333 页

IsColumnDescending 方法

检查索引是否按降序使用指定列。

语法

Visual Basic
```
Public Function IsColumnDescending( _
  ByVal name As String _
) As Boolean
```

C#
```
public bool IsColumnDescending(
  string  name
);
```

参数

- **name**　列的名称。

返回值

如果按降序使用列，则返回 true；如果按升序使用列，则返回 false。

另请参见

- “ULIndexSchema 类”一节第 332 页
- “ULIndexSchema 成员”一节第 332 页
- “GetColumnName 方法”一节第 338 页
- “ColumnCount 属性”一节第 333 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULInfoMessageEventArgs 类

为 ULConnection.InfoMessage 事件提供数据。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULInfoMessageEventArgs**
  Inherits EventArgs

C#
public sealed class **ULInfoMessageEventArgs**: EventArgs

另请参见

- “ULInfoMessageEventArgs 成员”一节第 341 页
- “InfoMessage 事件”一节第 174 页

ULInfoMessageEventArgs 成员

公共属性

成员名称	说明
“Message 属性”一节第 341 页	数据库返回的信息性或警告消息字符串。
“NativeError 属性”一节第 342 页	与数据库返回的信息性消息或警告相对应的 SQL 代码。
“Source 属性”一节第 342 页	返回消息的 ADO.NET 数据提供程序的名称。

公共方法

成员名称	说明
“ToString 方法”一节第 343 页	返回 ULConnection.InfoMessage 事件的字符串表示形式。

另请参见

- “ULInfoMessageEventArgs 类”一节第 341 页
- “InfoMessage 事件”一节第 174 页

Message 属性

数据库返回的信息性或警告消息字符串。

语法

Visual Basic
Public Readonly Property **Message** As String

C#
public string **Message** { get;}

属性值

包含信息性或警告消息的字符串。

另请参见

- “ULInfoMessageEventArgs 类”一节第 341 页
- “ULInfoMessageEventArgs 成员”一节第 341 页

NativeError 属性

与数据库返回的信息性消息或警告相对应的 SQLCODE。

语法

Visual Basic
Public Readonly Property **NativeError** As ULSQLCode

C#
public ULSQLCode **NativeError** { get;}

属性值

信息性或警告 ULSQLCode 值。

另请参见

- “ULInfoMessageEventArgs 类”一节第 341 页
- “ULInfoMessageEventArgs 成员”一节第 341 页
- “ULSQLCode 枚举”一节第 441 页

Source 属性

返回消息的 ADO.NET 数据提供程序的名称。

语法

Visual Basic
Public Readonly Property **Source** As String

C#
public string **Source** { get;}

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

属性值

字符串 "UltraLite.NET Data Provider"。

另请参见

- “ULInfoMessageEventArgs 类”一节第 341 页
- “ULInfoMessageEventArgs 成员”一节第 341 页

ToString 方法

返回 ULConnection.InfoMessage 事件的字符串表示形式。

语法

Visual Basic
Public Overrides Function **ToString()** As String

C#
public override string **ToString();**

返回值

信息性或警告消息字符串。

另请参见

- “ULInfoMessageEventArgs 类”一节第 341 页
- “ULInfoMessageEventArgs 成员”一节第 341 页
- “InfoMessage 事件”一节第 174 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULInfoMessageEventHandler 委派

表示将处理 ULConnection.InfoMessage 事件的方法。

语法

Visual Basic

```
Public Delegate Sub ULInfoMessageEventHandler( _
  ByVal obj As Object, _
  ByVal args As ULInfoMessageEventArgs _
)
```

C#

```
public delegate void ULInfoMessageEventHandler(
  object obj,
  ULInfoMessageEventArgs args
);
```

另请参见

- “InfoMessage 事件”一节第 174 页
- “ULInfoMessageEventArgs 类”一节第 341 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULMetaDataCollectionNames 类

提供用于与 ULConnection.GetSchema(String,String[]) 配合使用以检索元数据集合的常量列表。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULMetaDataCollectionNames**

C#
public sealed class **ULMetaDataCollectionNames**

另请参见

- “ULMetaDataCollectionNames 成员”一节第 345 页
- “GetSchema(String, String[]) 方法”一节第 164 页

ULMetaDataCollectionNames 成员

公共属性

成员名称	说明
“Columns 属性”一节第 346 页	提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 Columns 集合的常量。
“DataSourceInformation 属性”一节第 347 页	提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 DataSourceInformation 集合的常量。
“DataTypes 属性”一节第 347 页	提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 DataTypes 集合的常量。
“ForeignKeys 属性”一节第 348 页	提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 ForeignKeys 集合的常量。
“IndexColumns 属性”一节第 348 页	提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 IndexColumns 集合的常量。
“Indexes 属性”一节第 349 页	提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 Indexes 集合的常量。
“MetaDataCollections 属性”一节第 350 页	提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 MetaDataCollections 集合的常量。
“Publications 属性”一节第 350 页	提供用于与 ULConnection.GetSchema(String,String[]) 配合使用以检索元数据集合的常量列表。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“ReservedWords 属性”一节第 351 页	提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 ReservedWords 集合的常量。
“Restrictions 属性”一节第 351 页	提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 Restrictions 集合的常量。
“Tables 属性”一节第 352 页	提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 Tables 集合的常量。

另请参见

- “ULMetaDataCollectionNames 类”一节第 345 页
- “GetSchema(String, String[]) 方法”一节第 164 页

Columns 属性

提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 Columns 集合的常量。

语法

Visual Basic
Public Shared Readonly Property **Columns** As String

C#
public const string **Columns** { get;}

属性值

表示 Columns 集合名称的字符串。

示例

以下代码使用 Columns 集合填充 DataTable。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.Columns )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.Columns );
```

另请参见

- “ULMetaDataCollectionNames 类”一节第 345 页
- “ULMetaDataCollectionNames 成员”一节第 345 页
- “GetSchema(String) 方法”一节第 163 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

DataSourceInformation 属性

提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 DataSourceInformation 集合的常量。

语法

Visual Basic
Public Shared Readonly Property **DataSourceInformation** As String

C#
public const string **DataSourceInformation** { get;}

属性值

表示 DataSourceInformation 集合名称的字符串。

示例

以下代码使用 DataSourceInformation 集合填充 DataTable。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.DataSourceInformation )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.DataSourceInformation );
```

另请参见

- “ULMetaDataCollectionNames 类”一节第 345 页
- “ULMetaDataCollectionNames 成员”一节第 345 页
- “GetSchema(String) 方法”一节第 163 页

DataTypes 属性

提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 DataTypes 集合的常量。

语法

Visual Basic
Public Shared Readonly Property **DataTypes** As String

C#
public const string **DataTypes** { get;}

属性值

表示 DataTypes 集合名称的字符串。

示例

以下代码使用 DataTypes 集合填充 DataTable。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.DataTypes )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.DataTypes );
```

另请参见

- “ULMetaDataCollectionNames 类”一节第 345 页
- “ULMetaDataCollectionNames 成员”一节第 345 页
- “GetSchema(String) 方法”一节第 163 页

ForeignKeys 属性

提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 ForeignKeys 集合的常量。

语法

Visual Basic
Public Shared Readonly Property **ForeignKeys** As String

C#
public const string **ForeignKeys** { get;}

属性值

表示 ForeignKeys 集合名称的字符串。

示例

以下代码使用 ForeignKeys 集合填充 DataTable。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.ForeignKeys )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.ForeignKeys );
```

另请参见

- “ULMetaDataCollectionNames 类”一节第 345 页
- “ULMetaDataCollectionNames 成员”一节第 345 页
- “GetSchema(String) 方法”一节第 163 页

IndexColumns 属性

提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 IndexColumns 集合的常量。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Shared Readonly Property **IndexColumns** As String

C#
public const string **IndexColumns** { get;}

属性值

表示 IndexColumns 集合名称的字符串。

示例

以下代码使用 IndexColumns 集合填充 DataTable。

```
' Visual Basic
Dim schema As DataTable =
  conn.GetSchema( ULMetaDataCollectionNames.IndexColumns )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.IndexColumns );
```

另请参见

- “ULMetaDataCollectionNames 类”一节第 345 页
- “ULMetaDataCollectionNames 成员”一节第 345 页
- “GetSchema(String) 方法”一节第 163 页

Indexes 属性

提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 Indexes 集合的常量。

语法

Visual Basic
Public Shared Readonly Property **Indexes** As String

C#
public const string **Indexes** { get;}

属性值

表示 Indexes 集合名称的字符串。

示例

以下代码使用 Indexes 集合填充 DataTable。

```
' Visual Basic
Dim schema As DataTable =
  conn.GetSchema( ULMetaDataCollectionNames.Indexes )

// C#
```

```
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.Indexes );
```

另请参见

- “ULMetaDataCollectionNames 类”一节第 345 页
- “ULMetaDataCollectionNames 成员”一节第 345 页
- “GetSchema(String) 方法”一节第 163 页

MetaDataCollections 属性

提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 MetaDataCollections 集合的常量。

语法

Visual Basic
Public Shared Readonly Property **MetaDataCollections** As String

C#
public const string **MetaDataCollections** { get;}

属性值

表示 MetaDataCollections 集合名称的字符串。

示例

以下代码使用 MetaDataCollections 集合填充 DataTable。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.MetaDataCollections )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.MetaDataCollections );
```

另请参见

- “ULMetaDataCollectionNames 类”一节第 345 页
- “ULMetaDataCollectionNames 成员”一节第 345 页
- “GetSchema(String) 方法”一节第 163 页

Publications 属性

提供用于与 ULConnection.GetSchema(String,String[]) 配合使用以检索元数据集合的常量列表。

语法

Visual Basic
Public Shared Readonly Property **Publications** As String

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

C#
public const string **Publications** { get;}

另请参见

- “ULMetaDataCollectionNames 类”一节第 345 页
- “ULMetaDataCollectionNames 成员”一节第 345 页
- “GetSchema(String, String[]) 方法”一节第 164 页

ReservedWords 属性

提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 ReservedWords 集合的常量。

语法

Visual Basic
Public Shared Readonly Property **ReservedWords** As String

C#
public const string **ReservedWords** { get;}

属性值

表示 ReservedWords 集合名称的字符串。

示例

以下代码使用 ReservedWords 集合填充 DataTable。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.ReservedWords )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.ReservedWords );
```

另请参见

- “ULMetaDataCollectionNames 类”一节第 345 页
- “ULMetaDataCollectionNames 成员”一节第 345 页
- “GetSchema(String) 方法”一节第 163 页

Restrictions 属性

提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 Restrictions 集合的常量。

语法

Visual Basic
Public Shared Readonly Property **Restrictions** As String

C#
public const string **Restrictions** { get;}

属性值

表示 Restrictions 集合名称的字符串。

示例

以下代码使用 Restrictions 集合填充 DataTable。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.Restrictions )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.Restrictions );
```

另请参见

- “ULMetaDataCollectionNames 类”一节第 345 页
- “ULMetaDataCollectionNames 成员”一节第 345 页
- “GetSchema(String) 方法”一节第 163 页

Tables 属性

提供用于与 ULConnection.GetSchema(String) 配合使用的、表示 Tables 集合的常量。

语法

Visual Basic
Public Shared Readonly Property **Tables** As String

C#
public const string **Tables** { get;}

属性值

表示 Tables 集合名称的字符串。

示例

以下代码使用 Tables 集合填充 DataTable。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.Tables )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.Tables );
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULMetaDataCollectionNames 类”一节第 345 页
- “ULMetaDataCollectionNames 成员”一节第 345 页
- “GetSchema(String) 方法”一节第 163 页

ULParameter 类

表示 ULCommand 的一个参数。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULParameter**
  Inherits DbParameter
  Implements ICloneable

C#
public sealed class **ULParameter**: DbParameter,
  ICloneable

注释

可以直接使用 ULParameter 对象的多个构造函数之一或使用 ULCommand.CreateParameter 方法来创建 ULParameter 对象。鉴于系统对 0 和 0.0 常量的特殊处理方法以及对重载方法的解析方式，我们强烈建议您在使用 ULParameter(string,object) 构造函数时显示地将常量值转换为类型对象。例如：

```
' Visual Basic
Dim p As ULParameter = New ULParameter( "", CType( 0, Object ) )

// C#
ULParameter p = new ULParameter( "", (object)0 );
```

参数（包括那些通过 ULCommand.CreateParameter 创建的参数）必须添加至要使用的 ULCommand.Parameters 集合中。所有参数均被视为定位参数，并且依照添加顺序由命令使用。

在 UltraLite.NET 中，参数只能被用作 IN 参数，并忽略所有映射信息。只有 ULParameter.Value 是非常重要的。

Inherits: System.Data.Common.DbParameter

Implements: System.Data.IDbDataParameter、System.Data.IDataParameter

另请参见

- “ULParameter 成员”一节第 355 页
- “ULCommand 类”一节第 86 页
- “CreateParameter 方法”一节第 104 页
- “ULParameter(String, Object) 构造函数”一节第 357 页
- “Parameters 属性”一节第 97 页
- “Value 属性”一节第 368 页
- DbParameter
- IDbDataParameter
- IDataParameter

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULParameter 成员

公共构造函数

成员名称	说明
“ULParameter 构造函数”一节第 356 页	初始化一个新的“ULParameter 类”一节第 354 页实例。

公共属性

成员名称	说明
“DbType 属性”一节第 362 页	指定参数的 System.Data.DbType
“Direction 属性”一节第 362 页	用于指示参数为仅输入、仅输出、双向还是存储过程返回值参数的值。
“IsNullable 属性”一节第 363 页	指定参数是否接受空值。
“Offset 属性”一节第 364 页	指定与 ULParameter.Value 之间的偏移量。
“ParameterName 属性”一节第 364 页	指定参数的名称。
“Precision 属性”一节第 365 页	指定用于表示 ULParameter.Value 的最多位数。
“Scale 属性”一节第 365 页	指定 ULParameter.Value 精确到的小数位数。
“Size 属性”一节第 366 页	指定列中数据的最大大小。
“SourceColumn 属性”一节第 366 页	指定映射到 DataSet 并用于装载或返回值的源列的名称。
“SourceColumnNullMapping 属性”一节第 367 页	
“SourceVersion 属性”一节第 367 页	装载 ULParameter.Value 时使用的 System.Data.DataRowVersion。
“ULDbType 属性”一节第 368 页	指定参数的 iAnywhere.Data.UltraLite.ULDbType。
“Value 属性”一节第 368 页	指定参数的值。

公共方法

成员名称	说明
“ResetDbType 方法”一节 第 369 页	UltraLite.NET 不支持此方法。
“ToString 方法”一节 第 369 页	返回此实例的字符串表示形式。

另请参见

- “ULParameter 类”一节第 354 页
- “ULCommand 类”一节第 86 页
- “CreateParameter 方法”一节第 104 页
- “ULParameter(String, Object) 构造函数”一节第 357 页
- “Parameters 属性”一节第 97 页
- “Value 属性”一节第 368 页
- DbParameter
- IDbDataParameter
- IDataParameter

ULParameter 构造函数

初始化一个新的“ULParameter 类”一节第 354 页实例。

ULParameter() 构造函数

用空值（在 Visual Basic 中为 Nothing）作为 ULParameter 对象的值对该对象初始化。

语法

Visual Basic
Public Sub **New()**

C#
public **ULParameter();**

示例

以下代码将创建一个值为 3 的 ULParameter 对象，并将其添加至名为 cmd 的 ULCommand 中。

```
' Visual Basic
Dim p As ULParameter = New ULParameter
p.Value = 3
cmd.Parameters.Add( p )

// C#
ULParameter p = new ULParameter();
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
p.Value = 3;
cmd.Parameters.Add( p );
```

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “ULParameter 构造函数”一节第 356 页
- “Value 属性”一节第 368 页
- “ULParameter(String, Object) 构造函数”一节第 357 页
- “ULCommand 类”一节第 86 页

ULParameter(String, Object) 构造函数

用指定参数名和值初始化 ULParameter 对象。

语法

Visual Basic

```
Public Sub New( _
  ByVal parameterName As String, _
  ByVal value As Object _
)
```

C#

```
public  ULParameter(
  string  parameterName,
  object value
);
```

参数

- **parameterName**　参数的名称。对于未命名参数，请使用一个空字符串 ("") 或空值引用（在 Visual Basic 中为 Nothing）作为其值。在 UltraLite.NET 中，参数名不由 ULCommand 来使用。
- **value**　要作为参数值的 System.Object。

注释

鉴于系统对 0 和 0.0 常量的特殊处理方法以及对重载方法的解析方式，我们强烈建议您在使用此构造函数时显示地将常量值转换为类型对象。

示例

以下代码将创建一个值为 0 的 ULParameter 对象，并将其添加至名为 cmd 的 ULCommand 中。

```
' Visual Basic
cmd.Parameters.Add( New ULParameter( "", CType( 0, Object ) ) )

// C#
cmd.Parameters.Add( new ULParameter( "", (object)0 ) );
```

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “ULParameter 构造函数”一节第 356 页
- “ULParameter() 构造函数”一节第 356 页
- “ULCommand 类”一节第 86 页
- Object

ULParameter(String, ULDbType) 构造函数

以指定参数名和数据类型初始化 ULParameter 对象。不建议使用此构造函数；提供它的目的是与其它数据提供程序兼容。

语法

Visual Basic

```
Public Sub New( _
  ByVal parameterName As String, _
  ByVal dbType As ULDbType _
)
```

C#

```
public  ULParameter(
  string  parameterName,
  ULDbType dbType
);
```

参数

- **parameterName**　参数的名称。对于未命名参数，请使用一个空字符串 ("") 或空值引用（在 Visual Basic 中为 Nothing）作为其值。在 UltraLite.NET 中，参数名不由 ULCommand 来使用。
- **dbType**　iAnywhere.Data.UltraLite.ULDbType 值之一。

注释

在 UltraLite.NET 中，参数只能被用作 IN 参数，并忽略所有映射信息。只有 ULParameter.Value 是非常重要的。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “ULParameter 构造函数”一节第 356 页
- “ULParameter() 构造函数”一节第 356 页
- “ULParameter(String, Object) 构造函数”一节第 357 页
- “Value 属性”一节第 368 页
- “ULCommand 类”一节第 86 页
- “ULDbType 枚举”一节第 297 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULParameter(String, ULDbType, Int32) 构造函数

以指定参数名和数据类型初始化 ULParameter 对象。不建议使用此构造函数；提供它是为了与其它数据提供程序兼容。

语法

Visual Basic

```
Public Sub New( _
  ByVal parameterName As String, _
  ByVal dbType As ULDbType, _
  ByVal size As Integer _
)
```

C#

```
public  ULParameter(
  string  parameterName,
  ULDbType dbType,
  int size
);
```

参数

- **parameterName** 参数的名称。对于未命名参数，请使用一个空字符串 ("") 或空值引用（在 Visual Basic 中为 Nothing）作为其值。在 UltraLite.NET 中，参数名不由 ULCommand 来使用。
- **dbType** iAnywhere.Data.UltraLite.ULDbType 值之一。
- **size** 参数的长度。

注释

在 UltraLite.NET 中，参数只能被用作 IN 参数，并忽略所有映射信息。只有 ULParameter.Value 是非常重要的。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “ULParameter 构造函数”一节第 356 页
- “ULParameter() 构造函数”一节第 356 页
- “ULParameter(String, Object) 构造函数”一节第 357 页
- “Value 属性”一节第 368 页
- “ULDbType 枚举”一节第 297 页

ULParameter(String, ULDbType, Int32, String) 构造函数

以指定参数名、数据类型和长度初始化 ULParameter 对象。不建议使用此构造函数；提供它是为了与其它数据提供程序兼容。

语法

Visual Basic

```
Public Sub New( _
  ByVal parameterName As String, _
  ByVal dbType As ULDbType, _
  ByVal size As Integer, _
  ByVal sourceColumn As String _
)
```

C#

```
public ULParameter(
  string parameterName,
  ULDbType dbType,
  int size,
  string sourceColumn
);
```

参数

- **parameterName** 参数的名称。对于未命名参数，请使用一个空字符串 ("") 或空值引用（在 Visual Basic 中为 Nothing）作为其值。在 UltraLite.NET 中，参数名不由 ULCommand 来使用。
- **dbType** iAnywhere.Data.UltraLite.ULDbType 值之一。
- **size** 参数的长度。
- **sourceColumn** 要映射的源列的名称。

注释

在 UltraLite.NET 中，参数只能被用作 IN 参数，并忽略所有映射信息。只有 ULParameter.Value 是非常重要的。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “ULParameter 构造函数”一节第 356 页
- “ULParameter() 构造函数”一节第 356 页
- “ULParameter(String, Object) 构造函数”一节第 357 页
- “Value 属性”一节第 368 页
- “ULDbType 枚举”一节第 297 页
- “ULCommand 类”一节第 86 页

ULParameter(String, ULDbType, Int32, ParameterDirection, Boolean, Byte, Byte, String, DataRowVersion, Object) 构造函数

以指定参数名、数据类型、长度、方向、为空性、数值精度、数值小数位数、源列、源版本和值初始化 ULParameter 对象。不建议使用此构造函数；提供它是为了与其它数据提供程序兼容。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic

```
Public Sub New( _
  ByVal parameterName As String, _
  ByVal dbType As ULDbType, _
  ByVal size As Integer, _
  ByVal direction As ParameterDirection, _
  ByVal isNullable As Boolean, _
  ByVal precision As Byte, _
  ByVal scale As Byte, _
  ByVal sourceColumn As String, _
  ByVal sourceVersion As DataRowVersion, _
  ByVal value As Object _
)
```

C#

```
public ULParameter(
  string parameterName,
  ULDbType dbType,
  int size,
  ParameterDirection direction,
  bool isNullable,
  byte precision,
  byte scale,
  string sourceColumn,
  DataRowVersion sourceVersion,
  object value
);
```

参数

- **parameterName** 参数的名称。对于未命名参数，请使用一个空字符串 ("") 或空值引用（在 Visual Basic 中为 Nothing）作为其值。在 UltraLite.NET 中，参数名不由 ULCommand 来使用。
- **dbType** iAnywhere.Data.UltraLite.ULDbType 值之一。
- **size** 参数的长度。
- **direction** System.Data.ParameterDirection 值之一。
- **isNullable** 如果此字段的值可以为空，则为 true；否则为 false。
- **precision** Value 精确到的小数点左右两侧的总位数。
- **scale** Value 精确到的总小数位数。
- **sourceColumn** 要映射的源列的名称。
- **sourceVersion** System.Data.DataRowVersion 值之一。
- **value** 要作为参数值的 System.Object。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “ULParameter 构造函数”一节第 356 页
- “ULParameter() 构造函数”一节第 356 页
- “ULParameter(String, Object) 构造函数”一节第 357 页
- “ULDbType 枚举”一节第 297 页
- ParameterDirection
- DataRowVersion
- Object
- ParameterDirection.Input
- “ULCommand 类”一节第 86 页

DbType 属性

指定参数的 System.Data.DbType

语法

Visual Basic
Public Overrides Property **DbType** As DbType

C#
public override DbType **DbType** { get; set; }

属性值

System.Data.DbType 值之一。

注释

ULParameter.ULDbType 和 DbType 属性链接在一起。因此，设置 DbType 会将 ULParameter.ULDbType 更改为支持的 iAnywhere.Data.UltraLite.ULDbType。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- DbType
- DbType
- “ULDbType 属性”一节第 368 页
- “ULDbType 枚举”一节第 297 页

Direction 属性

用于指示参数为仅输入、仅输出、双向还是存储过程返回值参数的值。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Overrides Property **Direction** As ParameterDirection

C#
public override ParameterDirection **Direction** { get; set; }

属性值

System.Data.ParameterDirection 值之一。

注释

在 UltraLite.NET 中，参数只能被用作 IN 参数，并忽略所有映射信息。只有 ULParameter.Value 是非常重要的。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- ParameterDirection
- ParameterDirection.Input
- “Value 属性”一节第 368 页

IsNullable 属性

指定参数是否接受空值。

语法

Visual Basic
Public Overrides Property **IsNullable** As Boolean

C#
public override bool **IsNullable** { get; set; }

属性值

如果接受空值，则为 true；否则为 false。缺省值为 false。使用 DBNull 类处理空值。

注释

在 UltraLite.NET 中，参数只能被用作 IN 参数，并忽略所有映射信息。只有 ULParameter.Value 是非常重要的。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “Value 属性”一节第 368 页

Offset 属性

指定与 ULParameter.Value 之间的偏移量。

语法

Visual Basic
Public Property **Offset** As Integer

C#
public int **Offset** { get; set; }

属性值

值的偏移。缺省值为 0。

注释

在 UltraLite.NET 中，参数只能被用作 IN 参数，并忽略所有映射信息。只有 ULParameter.Value 是非常重要的。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “Value 属性”一节第 368 页

ParameterName 属性

指定参数的名称。

语法

Visual Basic
Public Overrides Property **ParameterName** As String

C#
public override string **ParameterName** { get; set; }

属性值

表示参数名称的字符串；如果是未命名参数，则为空字符串 ("")。指定空值引用（在 Visual Basic 中为 Nothing）会造成使用空字符串。

注释

在 UltraLite.NET 中，参数名不由 ULCommand 来使用。所有参数均被视为定位参数，并且依照添加顺序由命令使用。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “ULCommand 类”一节第 86 页

Precision 属性

指定用于表示 ULParameter.Value 的最多位数。

语法

Visual Basic
Public Property **Precision** As Byte

C#
public byte **Precision** { get; set; }

属性值

用于表示 ULParameter.Value 的最多位数。缺省值为 0，表示数据提供程序为 ULParameter.Value 设置精度。

注释

在 UltraLite.NET 中，参数只能被用作 IN 参数，并忽略所有映射信息。只有 ULParameter.Value 是非常重要的。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “Value 属性”一节第 368 页

Scale 属性

指定 ULParameter.Value 精确到的小数位数。

语法

Visual Basic
Public Property **Scale** As Byte

C#
public byte **Scale** { get; set; }

属性值

ULParameter.Value 精确到的小数位数。缺省值为 0。

注释

在 UltraLite.NET 中，参数只能被用作 IN 参数，并忽略所有映射信息。只有 ULParameter.Value 是非常重要的。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “Value 属性”一节第 368 页

Size 属性

指定列中数据的最大大小。

语法

Visual Basic
Public Overrides Property **Size** As Integer

C#
public override int **Size** { get; set; }

属性值

列中数据的最大大小。缺省值由参数值推导而来。Size 属性用于二进制或字符串类型。

注释

在 UltraLite.NET 中，参数只能被用作 IN 参数，并忽略所有映射信息。只有 ULParameter.Value 是非常重要的。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “Value 属性”一节第 368 页

SourceColumn 属性

指定映射到 DataSet 并用于装载或返回值的源列的名称。

语法

Visual Basic
Public Overrides Property **SourceColumn** As String

C#
public override string **SourceColumn** { get; set; }

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

属性值

一个字符串，指定映射到 DataSet 并用于装载或返回值的源列的名称。

注释

在 UltraLite.NET 中，参数只能被用作 IN 参数，并忽略所有映射信息。只有 ULParameter.Value 是非常重要的。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “Value 属性”一节第 368 页

SourceColumnNullMapping 属性

语法

Visual Basic
Public Overrides Property **SourceColumnNullMapping** As Boolean

C#
public override bool **SourceColumnNullMapping** { get; set; }

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页

SourceVersion 属性

装载 ULParameter.Value 时使用的 System.Data.DataRowVersion。

语法

Visual Basic
Public Overrides Property **SourceVersion** As DataRowVersion

C#
public override DataRowVersion **SourceVersion** { get; set; }

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “Value 属性”一节第 368 页
- DataRowVersion

ULDbType 属性

指定参数的 iAnywhere.Data.UltraLite.ULDbType。

语法

Visual Basic
Public Property **ULDbType** As ULDbType

C#
public ULDbType **ULDbType** { get; set; }

属性值

iAnywhere.Data.UltraLite.ULDbType 值之一。

注释

ULDbType 和 ULParameter.DbType 链接在一起。因此，设置 ULDbType 会将 ULParameter.DbType 更改为支持的 System.Data.DbType。

在 UltraLite.NET 中，参数只能被用作 IN 参数，并忽略所有映射信息。只有 ULParameter.Value 是非常重要的。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “Value 属性”一节第 368 页
- “DbType 属性”一节第 362 页
- DbType
- “ULDbType 枚举”一节第 297 页

Value 属性

指定参数的值。

语法

Visual Basic
Public Overrides Property **Value** As Object

C#
public override object **Value** { get; set; }

属性值

用于指定参数值的 System.Object。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

该值将按原样发送到数据提供程序，没有任何的类型转换或映射。执行该命令时，该命令会尝试将值转换为所需的类型，如果无法转换该值，则发送代码为 ULSQLCode.SQLE_CONVERSION_ERROR 的 ULException 信号。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “ULException 类”一节第 302 页
- “ULSQLCode 枚举”一节第 441 页
- Object

ResetDbType 方法

UltraLite.NET 不支持此方法。

语法

Visual Basic
Public Overrides Sub **ResetDbType()**

C#
public override void **ResetDbType();**

注释

在 UltraLite.NET 中，参数只能被用作 IN 参数，并忽略所有映射信息。只有 ULParameter.Value 是非常重要的。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页
- “Value 属性”一节第 368 页

ToString 方法

返回此实例的字符串表示形式。

语法

Visual Basic
Public Overrides Function **ToString()** As String

C#
public override string **ToString();**

返回值

参数的名称。

另请参见

- “ULParameter 类”一节第 354 页
- “ULParameter 成员”一节第 355 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULParameterCollection 类

表示 ULCommand 的所有参数。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULParameterCollection**
  Inherits DbParameterCollection

C#
public sealed class **ULParameterCollection**: DbParameterCollection

注释

集合中的所有参数均被视为定位参数，并按照 ULCommand.CommandText 中问号占位符的顺序指定。例如，集合中的第一个参数对应 SQL 语句中的第一个问号，集合中的第二个参数对应 SQL 语句中的第二个问号，依此类推。ULCommand.CommandText 中问号的个数必须至少与集合中参数的个数相同。以空值替代缺少的参数。

ULParameterCollection 不具有构造函数。您通过 ULCommand.Parameters 获取 ULParameterCollection。

Inherits: System.Data.Common.DbParameterCollection

Implements: System.Data.IDataParameterCollection

另请参见

- “ULParameterCollection 成员”一节第 371 页
- “ULCommand 类”一节第 86 页
- “CommandText 属性”一节第 93 页
- “Parameters 属性”一节第 97 页
- DbParameterCollection
- IDataParameterCollection

ULParameterCollection 成员

公共属性

成员名称	说明
“Count 属性”一节第 373 页	返回集合中 ULParameter 对象的数量。
“IsFixedSize 属性”一节第 373 页	指示 ULParameterCollection 是否具有固定大小。
“IsReadOnly 属性”一节第 373 页	指示 ULParameterCollection 是否为只读。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“IsSynchronized 属性”一节第 374 页	指示 ULParameterCollection 是否已同步。
“Item 属性”一节第 374 页	获取和设置集合中的 DbParameter。
“SyncRoot 属性”一节第 376 页	返回可用于同步对 SAParameterCollection 的访问的对象。

公共方法

成员名称	说明
“Add 方法”一节第 376 页	向集合中添加一个 ULParameter。
“AddRange 方法”一节第 382 页	将值数组添加到 ULParameterCollection 的末尾。
“Clear 方法”一节第 383 页	删除集合中的所有参数。
“Contains 方法”一节第 384 页	表示集合中是否存在具有指定属性的 DbParameter。
“CopyTo 方法”一节第 385 页	将 ULParameter 对象从 ULParameterCollection 复制到指定的数组。
“GetEnumerator 方法”一节第 386 页	返回该集合的枚举器。
“IndexOf 方法”一节第 386 页	返回指定 DbParameter 对象的索引。
“Insert 方法”一节第 387 页	在集合中指定索引处插入 ULParameter。
“Remove 方法”一节第 388 页	从集合中删除一个 ULParameter。
“RemoveAt 方法”一节第 388 页	从集合中删除指定的 DbParameter 对象。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULCommand 类”一节第 86 页
- “CommandText 属性”一节第 93 页
- “Parameters 属性”一节第 97 页
- DbParameterCollection
- IDataParameterCollection

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

Count 属性

返回集合中 ULParameter 对象的数量。

语法

Visual Basic
Public Overrides Readonly Property **Count** As Integer

C#
public override int **Count** { get;}

属性值

集合中 ULParameter 对象的数量。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页

IsFixedSize 属性

指示 ULParameterCollection 是否具有固定大小。

语法

Visual Basic
Public Overrides Readonly Property **IsFixedSize** As Boolean

C#
public override bool **IsFixedSize** { get;}

属性值

如果此集合具有固定大小，则为 true，否则为 false。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页

IsReadOnly 属性

指示 ULParameterCollection 是否为只读。

语法

Visual Basic
Public Overrides Readonly Property **IsReadOnly** As Boolean

C#
public override bool **IsReadOnly** { get;}

属性值

如果此集合为只读，则返回 true；否则返回 false。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页

IsSynchronized 属性

指示 ULParameterCollection 是否已同步。

语法

Visual Basic
Public Overrides Readonly Property **IsSynchronized** As Boolean

C#
public override bool **IsSynchronized** { get;}

属性值

如果此集合已同步，则返回 true；否则返回 false。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页

Item 属性

获取和设置集合中的 DbParameter。

Item(Int32) 属性

返回指定索引处的 ULParameter。在 C# 中，此属性是 ULParameterCollection 类的索引器。

语法

Visual Basic
Public Property **Item(_**
  ByVal *index* As Integer _
) As ULParameter

C#
public ULParameter **this**[

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
  int index
] { get; set; }
```

参数

- **index** 要检索的参数的从零开始索引。值必须在 [0,ULParameterCollection.Count-1] 范围内。集合中第一个参数的索引值为 0。

属性值

指定索引处的 ULParameter。

注释

这是 DbParameterCollection.this[int] 的强类型版本。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “Item 属性”一节第 374 页
- “ULParameter 类”一节第 354 页
- DbParameterCollection.Item

Item(String) 属性

返回具有指定名称的 ULParameter。在 C# 中，此属性是 ULParameterCollection 类的索引器。

语法

Visual Basic

```
Public Property Item( _
  ByVal parameterName As String _
) As ULParameter
```

C#

```
public ULParameter this[
  string  parameterName
] { get; set; }
```

参数

- **parameterName** 要检索的参数的名称。

属性值

具有指定名称的 ULParameter。

注释

这是 DbParameterCollection.this[string] 的强类型版本。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “Item 属性”一节第 374 页
- “Item(Int32) 属性”一节第 264 页
- “GetOrdinal 方法”一节第 282 页
- “GetValue 方法”一节第 289 页
- “GetFieldType 方法”一节第 277 页
- “ULParameter 类”一节第 354 页

SyncRoot 属性

返回可用于同步对 SAParameterCollection 的访问的对象。

语法

Visual Basic
Public Overrides Readonly Property **SyncRoot** As Object

C#
public override object **SyncRoot** { get;}

属性值

用于同步对此集合的访问的对象。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页

Add 方法

向集合中添加一个 ULParameter。

Add(Object) 方法

向集合中添加一个 ULParameter。

语法

Visual Basic
Public Overrides Function **Add(_**
  ByVal *value* As Object _
) As Integer

C#
public override int **Add(**

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
  object value
);
```

参数

- **value** 要添加到集合中的 ULParameter 对象。

返回值

新 ULParameter 对象的索引。

注释

集合中的所有参数均被视为定位参数，且必须按照 ULCommand.CommandText 中对应的问号占位符的顺序添加到集合中。例如，集合中的第一个参数对应 SQL 语句中的第一个问号，集合中的第二个参数对应 SQL 语句中的第二个问号，依此类推。ULCommand.CommandText 中问号的个数必须至少与集合中参数的个数相同。以空值替代缺少的参数。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “Add 方法”一节第 376 页
- “Add(ULParameter) 方法”一节第 377 页
- “Add(String, Object) 方法”一节第 378 页
- “CommandText 属性”一节第 93 页
- “ULParameter 类”一节第 354 页

Add(ULParameter) 方法

向集合中添加一个 ULParameter。

语法

Visual Basic

```
Public Function Add( _
  ByVal value As ULParameter _
) As ULParameter
```

C#

```
public ULParameter Add(
  ULParameter value
);
```

参数

- **value** 要添加到集合中的 ULParameter 对象。

返回值

新 ULParameter 对象。

注释

集合中的所有参数均被视为定位参数，且必须按照 ULCommand.CommandText 中对应的问号占位符的顺序添加到集合中。例如，集合中的第一个参数对应 SQL 语句中的第一个问号，集合中的第二个参数对应 SQL 语句中的第二个问号，依此类推。ULCommand.CommandText 中问号的个数必须至少与集合中参数的个数相同。以空值替代缺少的参数。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “Add 方法”一节第 376 页
- “Add(String, Object) 方法”一节第 378 页
- “ULParameter 类”一节第 354 页
- “CommandText 属性”一节第 93 页

Add(String, Object) 方法

向集合中添加一个新的、用指定的参数名和值创建的 ULParameter。

语法

Visual Basic

```
Public Function Add( _
  ByVal parameterName As String, _
  ByVal value As Object _
) As ULParameter
```

C#

```
public ULParameter Add(
  string  parameterName,
  object value
);
```

参数

- **parameterName**　参数的名称。对于未命名参数，请使用一个空字符串 ("") 或空值引用（在 Visual Basic 中为 Nothing）作为其值。在 UltraLite.NET 中，参数名不由 ULCommand 来使用。
- **value**　要作为参数值的 System.Object。

返回值

新 ULParameter 对象。

注释

集合中的所有参数均被视为定位参数，且必须按照 ULCommand.CommandText 中对应的问号占位符的顺序添加到集合中。例如，集合中的第一个参数对应 SQL 语句中的第一个问号，集合中的第二个参数对应 SQL 语句中的第二个问号，依此类推。ULCommand.CommandText 中问号的个数必须至少与集合中参数的个数相同。以空值替代缺少的参数。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

鉴于系统对 0 和 0.0 常量的特殊处理方法以及对重载方法的解析方式，我们强烈建议您在使用此方法时显式地将常量值转换为类型对象。

示例

以下代码将向名为 cmd 的 ULCommand 中添加一个值为 0 的 ULParameter。

```
' Visual Basic
cmd.Parameters.Add( "", CType( 0, Object ) )

// C#
cmd.Parameters.Add( "", (object)0 );
```

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “Add 方法”一节第 376 页
- “Add(ULParameter) 方法”一节第 377 页
- “ULParameter 类”一节第 354 页
- “CommandText 属性”一节第 93 页
- “ULCommand 类”一节第 86 页
- Object

Add(String, ULDbType) 方法

向集合中添加一个新的、用指定的参数名和数据类型创建的 ULParameter。

语法

Visual Basic

```
Public Function Add( _
  ByVal parameterName As String, _
  ByVal ulDbType As ULDbType _
) As ULParameter
```

C#

```
public ULParameter Add(
  string  parameterName,
  ULDbType ulDbType
);
```

参数

- **parameterName**　参数的名称。对于未命名参数，请使用一个空字符串 ("") 或空值引用（在 Visual Basic 中为 Nothing）作为其值。在 UltraLite.NET 中，参数名不由 ULCommand 来使用。
- **ulDbType**　iAnywhere.Data.UltraLite.ULDbType 值之一。

返回值

新 ULParameter 对象。

注释

集合中的所有参数均被视为定位参数，且必须按照 ULCommand.CommandText 中对应的问号占位符的顺序添加到集合中。例如，集合中的第一个参数对应 SQL 语句中的第一个问号，集合中的第二个参数对应 SQL 语句中的第二个问号，依此类推。ULCommand.CommandText 中问号的个数必须至少与集合中参数的个数相同。以空值替代缺少的参数。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “Add 方法”一节第 376 页
- “Add(ULParameter) 方法”一节第 377 页
- “Add(String, Object) 方法”一节第 378 页
- “ULParameter 类”一节第 354 页
- “CommandText 属性”一节第 93 页
- “ULCommand 类”一节第 86 页
- “ULDbType 枚举”一节第 297 页

Add(String, ULDbType, Int32) 方法

向集合中添加一个新的、用指定的参数名、数据类型和长度创建的 ULParameter。

语法

Visual Basic

```
Public Function Add( _
   ByVal parameterName As String, _
   ByVal ulDbType As ULDbType, _
   ByVal size As Integer _
) As ULParameter
```

C#

```
public ULParameter Add(
   string  parameterName,
   ULDbType ulDbType,
   int size
);
```

参数

- **parameterName**　参数的名称。对于未命名参数，请使用一个空字符串 ("") 或空值引用（在 Visual Basic 中为 Nothing）作为其值。在 UltraLite.NET 中，参数名不由 ULCommand 来使用。
- **ulDbType**　iAnywhere.Data.UltraLite.ULDbType 值之一。
- **size**　参数的长度。

返回值

新 ULParameter 对象。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

集合中的所有参数均被视为定位参数，且必须按照 ULCommand.CommandText 中对应的问号占位符的顺序添加到集合中。例如，集合中的第一个参数对应 SQL 语句中的第一个问号，集合中的第二个参数对应 SQL 语句中的第二个问号，依此类推。ULCommand.CommandText 中问号的个数必须至少与集合中参数的个数相同。以空值替代缺少的参数。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “Add 方法”一节第 376 页
- “Add(ULParameter) 方法”一节第 377 页
- “Add(String, Object) 方法”一节第 378 页
- “ULParameter 类”一节第 354 页
- “CommandText 属性”一节第 93 页
- “ULCommand 类”一节第 86 页
- “ULDbType 枚举”一节第 297 页

Add(String, ULDbType, Int32, String) 方法

向集合中添加一个新的、用指定的参数名、数据类型、长度和源列名称创建的 ULParameter。

语法

Visual Basic

```
Public Function Add( _
  ByVal parameterName As String, _
  ByVal ulDbType As ULDbType, _
  ByVal size As Integer, _
  ByVal sourceColumn As String _
) As ULParameter
```

C#

```
public ULParameter Add(
  string  parameterName,
  ULDbType ulDbType,
  int size,
  string  sourceColumn
);
```

参数

- **parameterName**　参数的名称。对于未命名参数，请使用一个空字符串 ("") 或空值引用（在 Visual Basic 中为 Nothing）作为其值。在 UltraLite.NET 中，参数名不由 ULCommand 来使用。
- **ulDbType**　iAnywhere.Data.UltraLite.ULDbType 值之一。
- **size**　参数的长度。
- **sourceColumn**　要映射的源列的名称。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

返回值

新 ULParameter 对象。

注释

集合中的所有参数均被视为定位参数，且必须按照 ULCommand.CommandText 中对应的问号占位符的顺序添加到集合中。例如，集合中的第一个参数对应 SQL 语句中的第一个问号，集合中的第二个参数对应 SQL 语句中的第二个问号，依此类推。ULCommand.CommandText 中问号的个数必须至少与集合中参数的个数相同。以空值替代缺少的参数。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “Add 方法”一节第 376 页
- “Add(ULParameter) 方法”一节第 377 页
- “Add(String, Object) 方法”一节第 378 页
- “ULParameter 类”一节第 354 页
- “CommandText 属性”一节第 93 页
- “ULCommand 类”一节第 86 页
- “ULDbType 枚举”一节第 297 页

AddRange 方法

将值数组添加到 ULParameterCollection 的末尾。

AddRange(Array) 方法

将值数组添加到 ULParameterCollection 的末尾。

语法

Visual Basic

```
Public Overrides Sub AddRange( _
  ByVal values As Array _
)
```

C#

```
public override void AddRange(
  Array values
);
```

参数

- **values** 要添加到此集合末尾的 ULParameter 对象数组。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “AddRange 方法”一节第 382 页
- “ULParameter 类”一节第 354 页

AddRange(ULParameter[]) 方法

将值数组添加到 ULParameterCollection 的末尾。

语法

Visual Basic

```
Public Sub AddRange( _
  ByVal values As ULParameter() _
)
```

C#

```
public void AddRange(
  ULParameter[] values
);
```

参数

- **values** 要添加到此集合末尾的 ULParameter 对象数组。

注释

这是 DbParameterCollection.AddRange(Array) 的强类型版本。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “AddRange 方法”一节第 382 页
- “ULParameter 类”一节第 354 页
- DbParameterCollection.AddRange
- “ULParameterCollection 类”一节第 371 页

Clear 方法

删除集合中的所有参数。

语法

Visual Basic

```
Public Overrides Sub Clear()
```

C#

```
public override void Clear();
```

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页

Contains 方法

表示集合中是否存在具有指定属性的 DbParameter。

Contains(Object) 方法

检查集合中是否存在 ULParameter。

语法

Visual Basic

```
Public Overrides Function Contains( _
  ByVal value As Object _
) As Boolean
```

C#

```
public override bool Contains(
  object value
);
```

参数

- **value** 要检查的 ULParameter 对象。

返回值

如果集合中包含该 ULParameter，则为 true；否则为 false。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “Contains 方法”一节第 384 页
- “Contains(String) 方法”一节第 384 页
- “ULParameter 类”一节第 354 页

Contains(String) 方法

检查集合中是否存在具有指定名称的 ULParameter。

语法

Visual Basic

```
Public Overrides Function Contains( _
  ByVal value As String _
) As Boolean
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

C#
```
public override bool Contains(
  string  value
);
```

参数

- **value** 要搜索的参数的名称。

返回值

如果集合中包含该 ULParameter，则为 true；否则为 false。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “Contains 方法”一节第 384 页
- “Contains(Object) 方法”一节第 384 页
- “ULParameter 类”一节第 354 页

CopyTo 方法

将 ULParameter 对象从 ULParameterCollection 复制到指定的数组。

语法

Visual Basic
```
Public Overrides Sub CopyTo( _
  ByVal array As Array, _
  ByVal index As Integer _
)
```

C#
```
public override void CopyTo(
  Array array,
  int index
);
```

参数

- **array** 向其中复制 ULParameter 对象的数组。
- **index** 数组的起始索引。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “ULParameter 类”一节第 354 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

GetEnumerator 方法

返回该集合的枚举器。

语法

Visual Basic
Public Overrides Function **GetEnumerator()** As IEnumerator

C#
public override IEnumerator **GetEnumerator();**

返回值

用于枚举集合中参数的 ArrayList 枚举器。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页

IndexOf 方法

返回指定 DbParameter 对象的索引。

IndexOf(Object) 方法

返回 ULParameter 在集合中的位置。

语法

Visual Basic
```
Public Overrides Function IndexOf( _
  ByVal value As Object _
) As Integer
```

C#
```
public override int IndexOf(
  object value
);
```

参数

- **value** 要查找的 ULParameter 对象。

返回值

集合中 ULParameter 的从零开始索引；如果没有找到此参数，则返回 -1。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “IndexOf 方法”一节第 386 页
- “IndexOf(String) 方法”一节第 387 页
- “ULParameter 类”一节第 354 页

IndexOf(String) 方法

返回指定名称的 ULParameter 在集合中的位置。

语法

Visual Basic

```
Public Overrides Function IndexOf( _
  ByVal parameterName As String _
) As Integer
```

C#

```
public override int IndexOf(
  string  parameterName
);
```

参数

- **parameterName**　要查找的参数的名称。

返回值

集合中 ULParameter 的从零开始索引；如果没有找到此参数，则返回 -1。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “IndexOf 方法”一节第 386 页
- “IndexOf(Object) 方法”一节第 386 页
- “ULParameter 类”一节第 354 页

Insert 方法

在集合中指定索引处插入 ULParameter。

语法

Visual Basic

```
Public Overrides Sub Insert( _
  ByVal index As Integer, _
  ByVal value As Object _
)
```

C#
```
public override void Insert(
  int index,
  object value
);
```

参数

- **index** 集合内要插入参数位置处的索引（从零开始）。
- **value** 要插入的 ULParameter 对象。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “ULParameter 类”一节第 354 页

Remove 方法

从集合中删除一个 ULParameter。

语法

Visual Basic
```
Public Overrides Sub Remove( _
  ByVal value As Object _
)
```

C#
```
public override void Remove(
  object value
);
```

参数

- **value** 要删除的 ULParameter 对象。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “ULParameter 类”一节第 354 页

RemoveAt 方法

从集合中删除指定的 DbParameter 对象。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

RemoveAt(Int32) 方法

从集合中删除指定索引处的参数。

语法

Visual Basic
```
Public Overrides Sub RemoveAt( _
  ByVal index As Integer _
)
```

C#
```
public override void RemoveAt(
  int index
);
```

参数

- **index** 要删除的参数的索引（从零开始）。值必须在 [0,ULParameterCollection.Count-1] 范围内。集合中第一个参数的索引值为 0。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “RemoveAt 方法”一节第 388 页
- “RemoveAt(String) 方法”一节第 389 页
- “Count 属性”一节第 373 页

RemoveAt(String) 方法

从集合中删除具有指定名称的参数。

语法

Visual Basic
```
Public Overrides Sub RemoveAt( _
  ByVal parameterName As String _
)
```

C#
```
public override void RemoveAt(
  string  parameterName
);
```

参数

- **parameterName** 要检索的参数的名称。

另请参见

- “ULParameterCollection 类”一节第 371 页
- “ULParameterCollection 成员”一节第 371 页
- “RemoveAt 方法”一节第 388 页
- “RemoveAt(Int32) 方法”一节第 389 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULPublicationSchema 类

UL Ext.: 表示 UltraLite 发布的模式。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULPublicationSchema**

C#
public sealed class **ULPublicationSchema**

注释

没有用于此类的构造函数。发布模式是使用 ULDatabaseSchema 类的 ULDatabaseSchema.GetPublicationSchema 方法创建的。

一些需要使用发布的方法也采用以逗号分隔的发布列表。

此类还提供两个特殊的发布值。ULPublicationSchema.SYNC_ALL_DB 对应于整个数据库。ULPublicationSchema.SYNC_ALL_PUBS 对应于所有发布。

另请参见

- “ULPublicationSchema 成员”一节第 391 页
- “GetLastDownloadTime 方法”一节第 159 页
- “CountUploadRows(String, UInt32) 方法”一节第 149 页
- “Schema 属性”一节第 142 页
- “SYNC_ALL_DB 字段”一节第 392 页
- “SYNC_ALL_PUBS 字段”一节第 392 页
- “GetPublicationSchema 方法”一节第 253 页
- “ULDatabaseSchema 类”一节第 247 页

ULPublicationSchema 成员

公共字段

成员名称	说明
“SYNC_ALL_DB 字段”一节第 392 页	对应于整个数据库的空发布列表。此字段为常量并是只读字段。
“SYNC_ALL_PUBS 字段”一节第 392 页	对应于所有发布的发布名称 "*"。此字段为常量并是只读字段。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

公共属性

成员名称	说明
“IsOpen 属性”一节第 393 页	确定发布模式是处于打开状态还是关闭状态。
“Name 属性”一节第 393 页	返回此发布的名称。

另请参见

- “ULPublicationSchema 类”一节第 391 页
- “GetLastDownloadTime 方法”一节第 159 页
- “CountUploadRows(String, UInt32) 方法”一节第 149 页
- “Schema 属性”一节第 142 页
- “SYNC_ALL_DB 字段”一节第 392 页
- “SYNC_ALL_PUBS 字段”一节第 392 页
- “GetPublicationSchema 方法”一节第 253 页
- “ULDatabaseSchema 类”一节第 247 页

SYNC_ALL_DB 字段

对应于整个数据库的空发布列表。此字段为常量并是只读字段。

语法

Visual Basic
Public Shared **SYNC_ALL_DB** As String

C#
public const string **SYNC_ALL_DB**;

另请参见

- “ULPublicationSchema 类”一节第 391 页
- “ULPublicationSchema 成员”一节第 391 页

SYNC_ALL_PUBS 字段

对应于所有发布的发布名称 "*"。此字段为常量并是只读字段。

语法

Visual Basic
Public Shared **SYNC_ALL_PUBS** As String

C#
public const string **SYNC_ALL_PUBS**;

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULPublicationSchema 类”一节第 391 页
- “ULPublicationSchema 成员”一节第 391 页

IsOpen 属性

确定发布模式是处于打开状态还是关闭状态。

语法

Visual Basic
Public Readonly Property **IsOpen** As Boolean

C#
public bool **IsOpen** { get;}

属性值

如果发布模式处于打开状态，则返回 true；如果发布模式处于关闭状态，则返回 false。

另请参见

- “ULPublicationSchema 类”一节第 391 页
- “ULPublicationSchema 成员”一节第 391 页

Name 属性

返回此发布的名称。

语法

Visual Basic
Public Readonly Property **Name** As String

C#
public string **Name** { get;}

属性值

用于指定发布名称的字符串。

另请参见

- “ULPublicationSchema 类”一节第 391 页
- “ULPublicationSchema 成员”一节第 391 页

ULResultSet 类

UL Ext.: 表示 UltraLite 数据库中可编辑的结果集。

语法

Visual Basic
Public Class **ULResultSet**
  Inherits ULDataReader

C#
public class **ULResultSet**: ULDataReader

注释

没有用于此类的构造函数。ResultSets 是使用 ULCommand 类的 ULCommand.ExecuteResultSet() 方法创建的。

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "SELECT emp_id FROM employee", conn _
  )
Dim resultSet As ULResultSet = cmd.ExecuteResultSet()

// C#
ULCommand cmd = new ULCommand(
    "SELECT emp_id FROM employee", conn
  );
ULResultSet resultSet = cmd.ExecuteResultSet();
```

ULResultSet 对象表示一个可编辑的结果集，您可在其上执行定位更新和删除。要获得完全可编辑的结果集，请使用 ULCommand.ExecuteTable() 或 ULDataAdapter。

Inherits: ULDataReader

Implements: System.Data.IDataReader、System.Data.IDataRecord、System.IDisposable

另请参见

- “ULResultSet 成员”一节第 395 页
- “ExecuteResultSet() 方法”一节第 115 页
- “ULCommand 类”一节第 86 页
- “ULResultSet 类”一节第 394 页
- “ExecuteTable() 方法”一节第 118 页
- “ULDataAdapter 类”一节第 226 页
- “ULDataReader 类”一节第 257 页
- IDataReader
- IDataRecord
- IDisposable

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULResultSet 成员

公共属性

成员名称	说明
“Depth 属性”一节第 262 页（继承自 ULDataReader）	返回当前行的嵌套深度。最外层的表深度为 0。
“FieldCount 属性”一节第 262 页（继承自 ULDataReader）	返回游标中的列数。
“HasRows 属性”一节第 263 页（继承自 ULDataReader）	检查 ULDataReader 包含一行还是多行。
“IsBOF 属性”一节第 263 页（继承自 ULDataReader）	**UL Ext.:** 检查当前行位置是否在第一行之前。
“IsClosed 属性”一节第 264 页（继承自 ULDataReader）	检查游标当前是否处于打开状态。
“IsEOF 属性”一节第 264 页（继承自 ULDataReader）	**UL Ext.:** 检查当前行位置是否在最后一行之后。
“Item 属性”一节第 264 页（继承自 ULDataReader）	以 Object 实例的形式获取指定列的值。
“RecordsAffected 属性”一节第 266 页（继承自 ULDataReader）	返回通过执行 SQL 语句所更改、插入或删除的行数。对于 SELECT 语句或 CommandType.TableDirect 表，此值为 -1。
“RowCount 属性”一节第 267 页（继承自 ULDataReader）	**UL Ext.:** 返回游标中的行数。
“Schema 属性”一节第 267 页（继承自 ULDataReader）	**UL Ext.:** 保存此游标的模式。
VisibleFieldCount（继承自 DbDataReader）	获取 DbDataReader 中未隐藏的字段数。

公共方法

成员名称	说明
“AppendBytes 方法”一节第 400 页	将指定的 System.Bytes 数组的指定子集附加到指定的 ULDbType.LongBinary 列的新值中。

成员名称	说明
“AppendChars 方法”一节第 402 页	将指定的 System.Chars 数组的指定子集附加到指定的 ULDbType.LongVarchar 列的新值。
“Close 方法”一节第 268 页（继承自 ULDataReader）	关闭游标。
“Delete 方法”一节第 403 页	删除当前行。
Dispose（继承自 DbDataReader）	释放此 DbDataReader 所占用的资源。
“GetBoolean 方法”一节第 268 页（继承自 ULDataReader）	以 System.Boolean 形式返回指定列的值。
“GetByte 方法”一节第 269 页（继承自 ULDataReader）	以无符号 8 位值 (System.Byte) 形式返回指定列的值。
“GetBytes 方法”一节第 269 页（继承自 ULDataReader）	**UL Ext.:** 以 System.Bytes 数组的形式返回指定列的值。仅对 ULDbType.Binary、ULDbType.LongBinary 或 ULDbType.UniqueIdentifier 类型的列有效。
“GetChar 方法”一节第 272 页（继承自 ULDataReader）	UltraLite.NET 不支持此方法。
“GetChars 方法”一节第 273 页（继承自 ULDataReader）	从指定的偏移量开始，将指定的 ULDbType.LongVarchar 列的值的子集复制到目标 System.Char 数组的指定偏移量处。
GetData（继承自 DbDataReader）	返回与请求的列顺序号对应的 DbDataReader 对象。
“GetDataTypeName 方法”一节第 274 页（继承自 ULDataReader）	返回指定列的提供程序数据类型的名称。
“GetDateTime 方法”一节第 275 页（继承自 ULDataReader）	以 System.DateTime 形式返回指定列的值，其精度为毫秒。
“GetDecimal 方法”一节第 275 页（继承自 ULDataReader）	以 System.Decimal 形式返回指定列的值。
“GetDouble 方法”一节第 276 页（继承自 ULDataReader）	以 System.Double 形式返回指定列的值。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“GetEnumerator 方法”一节第 277 页（继承自 ULDataReader）	返回迭代通过 ULDataReader 的 System.Collections.IEnumerator。
“GetFieldType 方法”一节第 277 页（继承自 ULDataReader）	返回最适合于指定列的 System.Type。
“GetFloat 方法”一节第 278 页（继承自 ULDataReader）	以 System.Single 形式返回指定列的值。
“GetGuid 方法”一节第 279 页（继承自 ULDataReader）	以 UUID (System.Guid) 形式返回指定列的值。
“GetInt16 方法”一节第 279 页（继承自 ULDataReader）	以 System.Int16 形式返回指定列的值。
“GetInt32 方法”一节第 280 页（继承自 ULDataReader）	以 Int32 形式返回指定列的值。
“GetInt64 方法”一节第 281 页（继承自 ULDataReader）	以 Int64 形式返回指定列的值。
“GetName 方法”一节第 281 页（继承自 ULDataReader）	返回指定列的名称。
“GetOrdinal 方法”一节第 282 页（继承自 ULDataReader）	返回指定列的列 ID。
GetProviderSpecificFieldType（继承自 DbDataReader）	返回指定列特定于提供程序的字段类型。
GetProviderSpecificValue（继承自 DbDataReader）	以 Object 实例的形式获取指定列的值。
GetProviderSpecificValues（继承自 DbDataReader）	获取当前行的集合中所有特定于提供程序的属性列。
“GetSchemaTable 方法”一节第 284 页（继承自 ULDataReader）	返回描述 ULDataReader 的列元数据的 System.Data.DataTable。
“GetString 方法”一节第 286 页（继承自 ULDataReader）	以 System.String 形式返回指定列的值。

成员名称	说明
“GetTimeSpan 方法”一节 第 286 页（继承自 ULDataReader）	以 System.TimeSpan 形式返回指定列的值，其精度为毫秒。
“GetUInt16 方法”一节 第 287 页（继承自 ULDataReader）	以 System.UInt16 形式返回指定列的值。
“GetUInt32 方法”一节 第 288 页（继承自 ULDataReader）	以 UInt32 形式返回指定列的值。
“GetUInt64 方法”一节 第 288 页（继承自 ULDataReader）	以 System.UInt64 形式返回指定列的值。
“GetValue 方法”一节第 289 页（继承自 ULDataReader）	返回以本地格式表示的指定列的值。
“GetValues 方法”一节 第 290 页（继承自 ULDataReader）	返回当前行的所有列值。
“IsDBNull 方法”一节第 291 页（继承自 ULDataReader）	检查指定列的值是否为 NULL。
“MoveAfterLast 方法”一节 第 291 页（继承自 ULDataReader）	**UL Ext.:** 将游标定位到游标的最后一行之后。
“MoveBeforeFirst 方法”一节 第 292 页（继承自 ULDataReader）	**UL Ext.:** 将游标定位到游标的第一行之前。
“MoveFirst 方法”一节 第 292 页（继承自 ULDataReader）	**UL Ext.:** 将游标定位到游标的第一行。
“MoveLast 方法”一节 第 292 页（继承自 ULDataReader）	**UL Ext.:** 将游标定位到游标的最后一行。
“MoveNext 方法”一节 第 293 页（继承自 ULDataReader）	**UL Ext.:** 将游标定位到下一行；如果游标已经位于最后一行，则定位到最后一行之后。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“MovePrevious 方法”一节 第 293 页（继承自 ULDataReader）	**UL Ext.:** 将游标定位到上一行，或第一行之前。
“MoveRelative 方法”一节 第 294 页（继承自 ULDataReader）	**UL Ext.:** 相对于当前行定位游标。
“NextResult 方法”一节 第 294 页（继承自 ULDataReader）	读取批处理 SQL 语句的结果时，将 ULDataReader 推进到下一结果。
“Read 方法”一节第 295 页（继承自 ULDataReader）	将游标定位到下一行；如果游标已经位于最后一行，则定位到最后一行之后。
“SetBoolean 方法”一节 第 404 页	使用 System.Boolean 设置指定列的值。
“SetByte 方法”一节 第 404 页	使用 System.Byte（无符号 8 位整数）设置指定列的值。
“SetBytes 方法”一节 第 405 页	使用 System.Bytes 数组设置指定列的值。
“SetDBNull 方法”一节 第 406 页	将列设置为 NULL。
“SetDateTime 方法”一节 第 407 页	使用 System.DateTime 设置指定列的值。
“SetDecimal 方法”一节 第 408 页	使用 System.Decimal 设置指定列的值。
“SetDouble 方法”一节 第 409 页	使用 System.Double 设置指定列的值。
“SetFloat 方法”一节 第 410 页	使用 System.Single 设置指定列的值。
“SetGuid 方法”一节 第 411 页	使用 System.Guid 设置指定列的值。
“SetInt16 方法”一节 第 412 页	使用 System.Int16 设置指定列的值.

成员名称	说明
“SetInt32 方法”一节 第 413 页	使用 System.Int32 设置指定列的值。
“SetInt64 方法”一节 第 414 页	使用 Int64 设置指定列的值。
“SetString 方法”一节 第 415 页	使用 System.String 设置指定列的值。
“SetTimeSpan 方法”一节 第 416 页	使用 System.TimeSpan 设置指定列的值。
“SetToDefault 方法”一节 第 416 页	将指定列的值设置为其缺省值。
“SetUInt16 方法”一节 第 417 页	使用 System.UInt16 设置指定列的值。
“SetUInt32 方法”一节 第 418 页	使用 System.UInt32 设置指定列的值。
“SetUInt64 方法”一节 第 419 页	使用 System.UInt64 设置指定列的值。
“Update 方法”一节第 420 页	用当前列值（使用 set 方法指定）来更新当前行。

另请参见

- “ULResultSet 类”一节第 394 页
- “ExecuteResultSet() 方法”一节第 115 页
- “ULCommand 类”一节第 86 页
- “ULResultSet 类”一节第 394 页
- “ExecuteTable() 方法”一节第 118 页
- “ULDataAdapter 类”一节第 226 页
- “ULDataReader 类”一节第 257 页
- IDataReader
- IDataRecord
- IDisposable

AppendBytes 方法

将指定的 System.Bytes 数组的指定子集附加到指定的 ULDbType.LongBinary 列的新值中。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic

```
Public Sub AppendBytes( _
  ByVal columnID As Integer, _
  ByVal val As Byte(), _
  ByVal srcOffset As Integer, _
  ByVal count As Integer _
)
```

C#

```
public void AppendBytes(
  int columnID,
  byte[] val,
  int srcOffset,
  int count
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 要添加到列的当前新值中的值。
- **srcOffset** 源数组中的起始位置。
- **count** 要复制的字节数。

注释

val 数组中位于 *srcOffset*（从零开始）到 *srcOffset*+*count*-1 之间的字节将被附加到指定列的值中。

插入时，ULTable.InsertBegin 将新值初始化为列的缺省值。直到执行 ULTable.Insert 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

更新时，列的第一个附加操作将首先清除当前值，而后再附加新值。

如果出现以下任一情况，则抛出代码为 ULSQLCode.SQLE_INVALID_PARAMETER 的 ULException，并且不会修改目标：

- *val* 为空。
- *srcOffset* 为负值。
- *count* 为负值。
- *srcOffset*+*count* 大于 *val*.Length。

对于其它错误，抛出具有相应错误代码的 ULException。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- Byte
- “ULDbType 枚举”一节第 297 页
- “InsertBegin 方法”一节第 527 页
- “Insert 方法”一节第 526 页
- “ULException 类”一节第 302 页
- “ULSQLCode 枚举”一节第 441 页
- “ULException 类”一节第 302 页
- “FieldCount 属性”一节第 262 页

AppendChars 方法

将指定的 System.Chars 数组的指定子集附加到指定的 ULDbType.LongVarchar 列的新值。

语法

Visual Basic

```
Public Sub AppendChars( _
  ByVal columnID As Integer, _
  ByVal val As Char(), _
  ByVal srcOffset As Integer, _
  ByVal count As Integer _
)
```

C#

```
public void AppendChars(
  int columnID,
  char[] val,
  int srcOffset,
  int count
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 要附加到列的当前新值中的值。
- **srcOffset** 源数组中的起始位置。
- **count** 要复制的字节数。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

val 数组中位于 *srcOffset*（从零开始）到 *srcOffset*+*count*-1 之间的字符将被附加到指定列的值中。插入时，ULTable.InsertBegin 将新值初始化为列的缺省值。直到执行 ULTable.Insert 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

更新时，列的第一个附加操作将首先清除当前值，而后再附加新值。

如果出现以下任一情况，则抛出代码为 ULSQLCode.SQLE_INVALID_PARAMETER 的 ULException，并且不会修改目标：

- *val* 为空值。
- *srcOffset* 为负值。
- *count* 为负值。
- *srcOffset*+*count* 大于 *value*.Length。

对于其它错误，抛出具有相应错误代码的 ULException。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- Char
- “ULDbType 枚举”一节第 297 页
- “InsertBegin 方法”一节第 527 页
- “Insert 方法”一节第 526 页
- “ULException 类”一节第 302 页
- “ULSQLCode 枚举”一节第 441 页
- “FieldCount 属性”一节第 262 页

Delete 方法

删除当前行。

语法

Visual Basic
Public Sub **Delete()**

C#
public void **Delete();**

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “StartSynchronizationDelete 方法”一节第 170 页
- “StopSynchronizationDelete 方法”一节第 171 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

SetBoolean 方法

使用 System.Boolean 设置指定列的值。

语法

Visual Basic
```
Public Sub SetBoolean( _
  ByVal columnID As Integer, _
  ByVal val As Boolean _
)
```

C#
```
public void SetBoolean(
  int columnID,
  bool val
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 列的新值。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- Boolean
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetByte 方法

使用 System.Byte（无符号 8 位整数）设置指定列的值。

语法

Visual Basic
```
Public Sub SetByte( _
  ByVal columnID As Integer, _
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

```
  ByVal val As Byte _
)
```

C#

```
public void SetByte(
  int columnID,
  byte val
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 列的新值。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- Byte
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetBytes 方法

使用 System.Bytes 数组设置指定列的值。

语法

Visual Basic

```
Public Sub SetBytes( _
  ByVal columnID As Integer, _
  ByVal val As Byte() _
)
```

C#

```
public void SetBytes(
  int columnID,
  byte[] val
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 列的新值。

注释

仅适用于 ULDbType.Binary 或 ULDbType.LongBinary 类型的列，或 ULDbType.UniqueIdentifier 类型的列（val 的长度为 16 时）。直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- Byte
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页
- “ULDbType 枚举”一节第 297 页
- “ULDbType 枚举”一节第 297 页
- “ULDbType 枚举”一节第 297 页

SetDBNull 方法

将列设置为 NULL。

语法

Visual Basic

```
Public Sub SetDBNull( _
  ByVal columnID As Integer _
)
```

C#

```
public void SetDBNull(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改数据；而且只有在完成提交后该更改才会成为永久更改。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “IsColumnNullable 方法”一节第 545 页
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetDateTime 方法

使用 System.DateTime 设置指定列的值。

语法

Visual Basic

```
Public Sub SetDateTime( _
  ByVal columnID As Integer, _
  ByVal val As Date _
)
```

C#

```
public void SetDateTime(
  int columnID,
  DateTime val
);
```

参数

- **columnID**　列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val**　列的新值。

注释

设置的值精确到毫秒。直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- DateTime
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetDecimal 方法

使用 System.Decimal 设置指定列的值。

语法

Visual Basic

```
Public Sub SetDecimal( _
  ByVal columnID As Integer, _
  ByVal val As Decimal _
)
```

C#

```
public void SetDecimal(
  int columnID,
  decimal val
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 列的新值。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- Decimal
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetDouble 方法

使用 System.Double 设置指定列的值。

语法

Visual Basic

```
Public Sub SetDouble( _
  ByVal columnID As Integer, _
  ByVal val As Double _
)
```

C#

```
public void SetDouble(
  int columnID,
  double val
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 列的新值。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- Double
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetFloat 方法

使用 System.Single 设置指定列的值。

语法

Visual Basic

```
Public Sub SetFloat( _
  ByVal columnID As Integer, _
  ByVal val As Single _
)
```

C#

```
public void SetFloat(
  int columnID,
  float  val
);
```

参数

- **columnID**　列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val**　列的新值。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- Single
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetGuid 方法

使用 System.Guid 设置指定列的值。

语法

Visual Basic

```
Public Sub SetGuid( _
  ByVal columnID As Integer, _
  ByVal val As Guid _
)
```

C#

```
public void SetGuid(
  int columnID,
  Guid val
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 列的新值。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。仅对于 ULDbType.UniqueIdentifier 类型的列或长度为 16 的 ULDbType.Binary 类型的列有效。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetNewUUID 方法”一节第 160 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- “GetColumnSize 方法”一节第 223 页
- Guid
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “ULDbType 枚举”一节第 297 页
- “ULDbType 枚举”一节第 297 页
- “FieldCount 属性”一节第 262 页

SetInt16 方法

使用 System.Int16 设置指定列的值.

语法

Visual Basic

```
Public Sub SetInt16( _
  ByVal columnID As Integer, _
  ByVal val As Short _
)
```

C#

```
public void SetInt16(
  int columnID,
  short val
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 列的新值。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- Int16
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetInt32 方法

使用 System.Int32 设置指定列的值。

语法

Visual Basic

```
Public Sub SetInt32( _
  ByVal columnID As Integer, _
  ByVal val As Integer _
)
```

C#

```
public void SetInt32(
  int columnID,
  int val
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 列的新值。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- Int32
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetInt64 方法

使用 Int64 设置指定列的值。

语法

Visual Basic

```
Public Sub SetInt64( _
  ByVal columnID As Integer, _
  ByVal val As Long _
)
```

C#

```
public void SetInt64(
  int columnID,
  long val
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 列的新值。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- Int64
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetString 方法

使用 System.String 设置指定列的值。

语法

Visual Basic

```
Public Sub SetString( _
  ByVal columnID As Integer, _
  ByVal val As String _
)
```

C#

```
public void SetString(
  int columnID,
  string  val
);
```

参数

- **columnID**　列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val**　列的新值。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

SetTimeSpan 方法

使用 System.TimeSpan 设置指定列的值。

语法

Visual Basic
```
Public Sub SetTimeSpan( _
  ByVal columnID As Integer, _
  ByVal val As TimeSpan _
)
```

C#
```
public void SetTimeSpan(
  int columnID,
  TimeSpan val
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 列的新值。

注释

设置的值精确到毫秒，并且被规范化为介于 0 和 24 小时之间的非负值。直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- TimeSpan
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetToDefault 方法

将指定列的值设置为其缺省值。

语法

Visual Basic
```
Public Sub SetToDefault( _
  ByVal columnID As Integer _
)
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

C#
```
public void SetToDefault(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- “GetColumnDefaultValue 方法”一节第 538 页
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetUInt16 方法

使用 System.UInt16 设置指定列的值。

语法

Visual Basic
```
Public Sub SetUInt16( _
  ByVal columnID As Integer, _
  ByVal val As UInt16 _
)
```

C#
```
public void SetUInt16(
  int columnID,
  ushort val
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 列的新值。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “GetFieldType 方法”一节第 277 页
- UInt16
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetUInt32 方法

使用 System.UInt32 设置指定列的值。

语法

Visual Basic

```
Public Sub SetUInt32( _
  ByVal columnID As Integer, _
  ByVal val As UInt32 _
)
```

C#

```
public void SetUInt32(
  int columnID,
  uint val
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 列的新值。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- UInt32
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

SetUInt64 方法

使用 System.UInt64 设置指定列的值。

语法

Visual Basic

```
Public Sub SetUInt64( _
  ByVal columnID As Integer, _
  ByVal val As UInt64 _
)
```

C#

```
public void SetUInt64(
  int columnID,
  ulong val
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULDataReader.FieldCount-1] 范围内。游标中第一列的 ID 值为 0。
- **val** 列的新值。

注释

直到执行 ULTable.Insert 或 Update 之后，才实际更改行中的数据；而且只有在完成提交后该更改才会成为永久更改。

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页
- “GetOrdinal 方法”一节第 282 页
- “Schema 属性”一节第 267 页
- “GetFieldType 方法”一节第 277 页
- UInt64
- “Insert 方法”一节第 526 页
- “Update 方法”一节第 420 页
- “FieldCount 属性”一节第 262 页

Update 方法

用当前列值（使用 set 方法指定）来更新当前行。

语法

Visual Basic
Public Sub **Update()**

C#
public void **Update();**

另请参见

- “ULResultSet 类”一节第 394 页
- “ULResultSet 成员”一节第 395 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULResultSetSchema 类

UL Ext.: 表示 UltraLite 结果集的模式。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULResultSetSchema**
  Inherits ULCursorSchema

C#
public sealed class **ULResultSetSchema**: ULCursorSchema

注释

没有用于此类的构造函数。ULResultSetSchema 对象作为结果集的 ULTable.Schema 附加在结果集上。

当数据读取器打开时结果集模式才有效。

Inherits: ULCursorSchema

另请参见

- “ULResultSetSchema 成员”一节第 421 页
- “ULCommand 类”一节第 86 页
- “ULDataReader 类”一节第 257 页
- “ULResultSetSchema 类”一节第 421 页
- “Schema 属性”一节第 267 页
- “ULCursorSchema 类”一节第 217 页

ULResultSetSchema 成员

公共属性

成员名称	说明
“ColumnCount 属性”一节第 218 页（继承自 ULCursorSchema）	返回游标中的列数。
“IsOpen 属性”一节第 219 页（继承自 ULCursorSchema）	检查游标模式当前是否处于打开状态。
“Name 属性”一节第 423 页	返回游标的名称。

公共方法

成员名称	说明
“GetColumnID 方法”一节第 219 页（继承自 ULCursorSchema）	返回指定列的列 ID。
“GetColumnName 方法”一节第 220 页（继承自 ULCursorSchema）	返回由指定的列 ID 标识的列的名称。
“GetColumnPrecision 方法”一节第 221 页（继承自 ULCursorSchema）	如果列为数值列（SQL 类型 NUMERIC），则返回由指定的列 ID 标识的列的精度。
“GetColumnSQLName 方法”一节第 222 页（继承自 ULCursorSchema）	返回由指定的列 ID 标识的列的名称。
“GetColumnScale 方法”一节第 223 页（继承自 ULCursorSchema）	如果列为数值列（SQL 类型 NUMERIC），则返回由指定的列 ID 标识的列的小数位数。
“GetColumnSize 方法”一节第 223 页（继承自 ULCursorSchema）	如果列是指定大小的列（SQL 类型 BINARY 或 CHAR），则返回由指定的列 ID 标识的列的大小。
“GetColumnULDbType 方法”一节第 224 页（继承自 ULCursorSchema）	返回由指定的列 ID 标识的列的 UltraLite.NET 数据类型。
“GetSchemaTable 方法”一节第 224 页（继承自 ULCursorSchema）	返回描述 ULDataReader 的列模式的 System.Data.DataTable。

另请参见

- “ULResultSetSchema 类”一节第 421 页
- “ULCommand 类”一节第 86 页
- “ULDataReader 类”一节第 257 页
- “ULResultSetSchema 类”一节第 421 页
- “Schema 属性”一节第 267 页
- “ULCursorSchema 类”一节第 217 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

Name 属性

返回游标的名称。

语法

Visual Basic
Public Overrides Readonly Property **Name** As String

C#
public override string **Name** { get;}

属性值

生成 ULResultSetSchema 的 SQL 语句。

另请参见

- “ULResultSetSchema 类”一节第 421 页
- “ULResultSetSchema 成员”一节第 421 页

ULRowsCopiedEventArgs 类

表示传递给 ULRowsCopiedEventHandler 的参数集。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULRowsCopiedEventArgs**

C#
public sealed class **ULRowsCopiedEventArgs**

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULRowsCopiedEventArgs 类。

另请参见

- “ULRowsCopiedEventArgs 成员”一节第 424 页
- “ULRowsCopiedEventHandler 委派”一节第 427 页

ULRowsCopiedEventArgs 成员

公共构造函数

成员名称	说明
“ULRowsCopiedEventArgs 构造函数”一节第 424 页	创建一个新的 ULRowsCopiedEventArgs 对象实例。

公共属性

成员名称	说明
“Abort 属性”一节第 425 页	获取或设置表示是否应中止批量复制操作的值。
“RowsCopied 属性”一节第 425 页	返回在当前批量复制操作期间复制的行数。

另请参见

- “ULRowsCopiedEventArgs 类”一节第 424 页
- “ULRowsCopiedEventHandler 委派”一节第 427 页

ULRowsCopiedEventArgs 构造函数

创建一个新的 ULRowsCopiedEventArgs 对象实例。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
```
Public Sub New( _
  ByVal rowsCopied As Long _
)
```

C#
```
public  ULRowsCopiedEventArgs(
  long rowsCopied
);
```

参数

- **rowsCopied**　表示在当前批量复制操作期间复制的行数的 64 位整数值。

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULRowsCopiedEventArgs 类。

另请参见

- “ULRowsCopiedEventArgs 类”一节第 424 页
- “ULRowsCopiedEventArgs 成员”一节第 424 页

Abort 属性

获取或设置表示是否应中止批量复制操作的值。

语法

Visual Basic
```
Public Property Abort As Boolean
```

C#
```
public bool Abort { get; set; }
```

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULRowsCopiedEventArgs 类。

另请参见

- “ULRowsCopiedEventArgs 类”一节第 424 页
- “ULRowsCopiedEventArgs 成员”一节第 424 页

RowsCopied 属性

返回在当前批量复制操作期间复制的行数。

语法

Visual Basic
Public Readonly Property **RowsCopied** As Long

C#
public long **RowsCopied** { get;}

属性值

表示复制的行数的长整数。

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULRowsCopiedEventArgs 类。

另请参见

- “ULRowsCopiedEventArgs 类”一节第 424 页
- “ULRowsCopiedEventArgs 成员”一节第 424 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULRowsCopiedEventHandler 委派

表示将处理 ULBulkCopy.ULRowsCopied 事件的方法。

语法

Visual Basic

```
Public Delegate Sub ULRowsCopiedEventHandler( _
  ByVal sender As Object, _
  ByVal rowsCopiedEventArgs As ULRowsCopiedEventArgs _
)
```

C#

```
public delegate void ULRowsCopiedEventHandler(
  object sender,
  ULRowsCopiedEventArgs rowsCopiedEventArgs
);
```

注释

Restrictions: 无法在 .NET Compact Framework 2.0 中使用 ULRowsCopiedEventHandler 委派。

另请参见

- “ULRowsCopied 事件”一节第 67 页
- Object

ULRowUpdatedEventArgs 类

为 ULDataAdapter.RowUpdated 事件提供数据。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULRowUpdatedEventArgs**
  Inherits RowUpdatedEventArgs

C#
public sealed class **ULRowUpdatedEventArgs**: RowUpdatedEventArgs

注释

Inherits: System.Data.Common.RowUpdatedEventArgs

另请参见

- “ULRowUpdatedEventArgs 成员”一节第 428 页
- “RowUpdated 事件”一节第 236 页
- RowUpdatedEventArgs

ULRowUpdatedEventArgs 成员

公共构造函数

成员名称	说明
“ULRowUpdatedEventArgs 构造函数”一节第 429 页	初始化 ULRowUpdatedEventArgs 类的一个新实例。

公共属性

成员名称	说明
“Command 属性”一节第 430 页	返回调用 DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) 时所执行的 ULCommand。
Errors（继承自 RowUpdatedEventArgs）	获取执行 RowUpdatedEventArgs.Command 时由 .NET Framework 数据提供程序生成的所有错误。
“RecordsAffected 属性”一节第 431 页	返回通过执行 SQL 语句所更改、插入或删除的行数。对于 SELECT 语句，此值为 -1。
Row（继承自 RowUpdatedEventArgs）	获取通过 DbDataAdapter.Update 发送的 DataRow。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
RowCount（继承自 RowUpdatedEventArgs）	获取一批更新记录中处理的行数。
StatementType（继承自 RowUpdatedEventArgs）	获取所执行 SQL 语句的类型。
Status（继承自 RowUpdatedEventArgs）	获取 RowUpdatedEventArgs.Command 的 UpdateStatus。
TableMapping（继承自 RowUpdatedEventArgs）	获取通过 DbDataAdapter.Update 发送的 DataTableMapping。

公共方法

成员名称	说明
CopyToRows（继承自 RowUpdatedEventArgs）	允许您访问在批处理更新操作期间处理的行。

另请参见

- “ULRowUpdatedEventArgs 类”一节第 428 页
- “RowUpdated 事件”一节第 236 页
- RowUpdatedEventArgs

ULRowUpdatedEventArgs 构造函数

初始化 ULRowUpdatedEventArgs 类的一个新实例。

语法

Visual Basic

```
Public Sub New( _
  ByVal row As DataRow, _
  ByVal command As IDbCommand, _
  ByVal statementType As StatementType, _
  ByVal tableMapping As DataTableMapping _
)
```

C#

```
public  ULRowUpdatedEventArgs(
  DataRow row,
  IDbCommand command,
  StatementType statementType,
  DataTableMapping tableMapping
);
```

参数

- **row** 通过 DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) 发送的 System.Data.DataRow。
- **command** 调用 DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) 时执行的 System.Data.IDbCommand。
- **statementType** 指定所执行查询类型的 System.Data.StatementType 值之一。
- **tableMapping** 通过 DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) 发送的 System.Data.Common.DataTableMapping。

另请参见

- “ULRowUpdatedEventArgs 类”一节第 428 页
- “ULRowUpdatedEventArgs 成员”一节第 428 页
- DataRow
- IDbCommand
- StatementType
- DataTableMapping
- DbDataAdapter.Update

Command 属性

返回调用 DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) 时所执行的 ULCommand。

语法

Visual Basic

```
Public Readonly Property Command As ULCommand
```

C#

```
public ULCommand Command { get;}
```

属性值

由更新执行的 ULCommand 对象。

注释

这是 System.Data.Common.RowUpdatedEventArgs.Command 的强类型版本。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULRowUpdatedEventArgs 类”一节第 428 页
- “ULRowUpdatedEventArgs 成员”一节第 428 页
- “ULCommand 类”一节第 86 页
- DbDataAdapter.Update
- RowUpdatedEventArgs.Command

RecordsAffected 属性

返回通过执行 SQL 语句所更改、插入或删除的行数。对于 SELECT 语句，此值为 -1。

语法

Visual Basic
Public Readonly Property **RecordsAffected** As Integer

C#
public int **RecordsAffected** { get;}

属性值

更改、插入或删除的行数；如果没有任何行受到影响或语句失败，则值为 0；如果是 SELECT 语句，则值为 -1。

另请参见

- “ULRowUpdatedEventArgs 类”一节第 428 页
- “ULRowUpdatedEventArgs 成员”一节第 428 页

ULRowUpdatedEventHandler 委派

表示将处理 ULDataAdapter.RowUpdated 事件的方法。

语法

Visual Basic

```
Public Delegate Sub ULRowUpdatedEventHandler( _
  ByVal sender As Object, _
  ByVal e As ULRowUpdatedEventArgs _
)
```

C#

```
public delegate void ULRowUpdatedEventHandler(
  object sender,
  ULRowUpdatedEventArgs e
);
```

另请参见

- “RowUpdated 事件”一节第 236 页
- “ULRowUpdatedEventArgs 类”一节第 428 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULRowUpdatingEventArgs 类

为 ULDataAdapter.RowUpdating 事件提供数据。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULRowUpdatingEventArgs**
  Inherits RowUpdatingEventArgs

C#
public sealed class **ULRowUpdatingEventArgs**: RowUpdatingEventArgs

注释

Inherits: System.Data.Common.RowUpdatingEventArgs

另请参见

- “ULRowUpdatingEventArgs 成员”一节第 433 页
- “RowUpdating 事件”一节第 237 页
- RowUpdatingEventArgs

ULRowUpdatingEventArgs 成员

公共构造函数

成员名称	说明
“ULRowUpdatingEventArgs 构造函数”一节第 434 页	初始化 ULRowUpdatingEventArgs 类的一个新实例。

公共属性

成员名称	说明
“Command 属性”一节第 435 页	指定执行 DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) 时要执行的 ULCommand。
Errors（继承自 RowUpdatingEventArgs）	获取执行 RowUpdatedEventArgs.Command 时 .NET Framework 数据提供程序所产生的任何错误。
Row（继承自 RowUpdatingEventArgs）	获取将作为插入、更新或删除操作的一部分发送给服务器的 DataRow。
StatementType（继承自 RowUpdatingEventArgs）	获取要执行的 SQL 语句的类型。

成员名称	说明
Status（继承自 RowUpdatingEventArgs）	获取或设置 RowUpdatedEventArgs.Command 的 UpdateStatus。
TableMapping（继承自 RowUpdatingEventArgs）	获取要通过 DbDataAdapter.Update 发送的 DataTableMapping。

另请参见

- “ULRowUpdatingEventArgs 类”一节第 433 页
- “RowUpdating 事件”一节第 237 页
- RowUpdatingEventArgs

ULRowUpdatingEventArgs 构造函数

初始化 ULRowUpdatingEventArgs 类的一个新实例。

语法

Visual Basic

```
Public Sub New( _
  ByVal row As DataRow, _
  ByVal command As IDbCommand, _
  ByVal statementType As StatementType, _
  ByVal tableMapping As DataTableMapping _
)
```

C#

```
public  ULRowUpdatingEventArgs(
  DataRow row,
  IDbCommand command,
  StatementType statementType,
  DataTableMapping tableMapping
);
```

参数

- **row**　要更新的 System.Data.DataRow。
- **command**　更新期间执行的 System.Data.IDbCommand。
- **statementType**　指定所执行查询类型的 System.Data.StatementType 值之一。
- **tableMapping**　通过 DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) 发送的 System.Data.Common.DataTableMapping。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULRowUpdatingEventArgs 类”一节第 433 页
- “ULRowUpdatingEventArgs 成员”一节第 433 页
- DataRow
- IDbCommand
- StatementType
- DataTableMapping
- DbDataAdapter.Update

Command 属性

指定执行 DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) 时要执行的 ULCommand。

语法

Visual Basic
Public Property **Command** As ULCommand

C#
public ULCommand **Command** { get; set; }

属性值

更新时执行的 ULCommand 对象。

注释

这是 System.Data.Common.RowUpdatingEventArgs.Command 的强类型版本。

另请参见

- “ULRowUpdatingEventArgs 类”一节第 433 页
- “ULRowUpdatingEventArgs 成员”一节第 433 页
- “ULCommand 类”一节第 86 页
- DbDataAdapter.Update
- RowUpdatingEventArgs.Command

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULRowUpdatingEventHandler 委派

表示将处理 ULDataAdapter.RowUpdating 事件的方法。

语法

Visual Basic

```
Public Delegate Sub ULRowUpdatingEventHandler( _
  ByVal sender As Object, _
  ByVal e As ULRowUpdatingEventArgs _
)
```

C#

```
public delegate void ULRowUpdatingEventHandler(
  object sender,
  ULRowUpdatingEventArgs e
);
```

另请参见

- “RowUpdating 事件”一节第 237 页
- “ULRowUpdatingEventArgs 类”一节第 433 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULRuntimeType 枚举

UL Ext.: 枚举 UltraLite.NET 运行库的类型。

语法

Visual Basic
Public Enum **ULRuntimeType**

C#
public enum **ULRuntimeType**

成员

成员名称	说明	值
STANDALONE_UL	选择独立 UltraLite.NET 运行时。 独立运行库直接访问数据库。以此方式访问数据库的速度更快，但无法共享数据库。	0
UL_ENGINE_CLIENT	选择 UltraLite 引擎运行时。 UltraLite.NET 引擎客户端与 UltraLite 引擎进行通信以访问数据库。这意味着不同应用程序可以共享数据库。	1

另请参见

- “RuntimeType 属性”一节第 239 页

ULServerSyncListener 接口

UL Ext.: 用于接收服务器同步消息的监听器接口。

语法

Visual Basic
Public Interface **ULServerSyncListener**

C#
public interface **ULServerSyncListener**

另请参见

- “ULServerSyncListener 成员”一节第 438 页

ULServerSyncListener 成员

公共方法

成员名称	说明
“ServerSyncInvoked 方法”一节第 438 页	在服务器启动的同步的 MobiLink 监听器调用应用程序执行同步时被调用。

另请参见

- “ULServerSyncListener 接口”一节第 438 页

ServerSyncInvoked 方法

在服务器启动的同步的 MobiLink 监听器调用应用程序执行同步时被调用。

语法

Visual Basic
```
Public Sub ServerSyncInvoked( _
  ByVal messageName As String _
)
```

C#
```
public void ServerSyncInvoked(
  string  messageName
);
```

参数

- **messageName**　发送到应用程序的消息的名称。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

此方法由单独的线程调用。为避免多线程问题，它应向 UI 发布一个事件。如果您正在使用多线程，建议您使用一个单独的连接，并使用 lock 关键字来访问与应用程序的其余部分共享的所有对象。

示例

以下代码段介绍如何接收服务器同步请求并在 UI 线程中执行同步。

```
' Visual Basic
Imports iAnywhere.Data.UltraLite

Public Class MainWindow
  Inherits System.Windows.Forms.Form
  Implements ULServerSyncListener
  Private conn As ULConnection

  Public Sub New(ByVal args() As String)

    MyBase.New()

    'This call is required by the Windows Form Designer.
    InitializeComponent()

    'Add any initialization after the InitializeComponent() call
    ULConnection.DatabaseManager.SetServerSyncListener( _
        "myCompany.mymsg", "myCompany.myapp", Me _
      )
    'Create Connection
    ...
  End Sub

  Protected Overrides Sub OnClosing( _
        ByVal e As System.ComponentModel.CancelEventArgs _
      )
    ULConnection.DatabaseManager.SetServerSyncListener( _
        Nothing, Nothing, Nothing _
      )
    MyBase.OnClosing(e)
  End Sub

  Public Sub ServerSyncInvoked(ByVal messageName As String) _
      Implements ULServerSyncListener.ServerSyncInvoked
    Me.Invoke(New EventHandler(AddressOf Me.ServerSyncAction))
  End Sub

  Public Sub ServerSyncAction( _
        ByVal sender As Object, ByVal e As EventArgs _
      )
    ' Do Server sync
    conn.Synchronize()
  End Sub
End Class


// C#
using iAnywhere.Data.UltraLite;
public class Form1 : System.Windows.Forms.Form, ULServerSyncListener
{
  private System.Windows.Forms.MainMenu mainMenu1;
  private ULConnection conn;
```

```
    public Form1()
    {
      //
      // Required for Windows Form Designer support
      //
      InitializeComponent();

      //
      // TODO: Add any constructor code after
      // InitializeComponent call
      //
      ULConnection.DatabaseManager.SetServerSyncListener(
          "myCompnay.mymsg", "myCompany.myapp", this
        );
      // Create connection
      ...
    }

    protected override void Dispose( bool disposing )
    {
      base.Dispose( disposing );
    }

    protected override void OnClosing(
        System.ComponentModel.CancelEventArgs e
      )
    {
      ULConnection.DatabaseManager.SetServerSyncListener(
          null, null, null
        );
      base.OnClosing(e);
    }

    public void ServerSyncInvoked( string messageName )
    {
      this.Invoke( new EventHandler( ServerSyncHandler ) );
    }

    internal void ServerSyncHandler(object sender, EventArgs e)
    {
      conn.Synchronize();
    }
  }
```

另请参见

- “ULServerSyncListener 接口”一节第 438 页
- “ULServerSyncListener 成员”一节第 438 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULSQLCode 枚举

UL Ext.: 枚举 UltraLite.NET 可能会报告的 SQL 代码。

语法

Visual Basic
Public Enum **ULSQLCode**

C#
public enum **ULSQLCode**

成员

成员名称	说明	值
SQLE_AGGREGATES_NOT_ALLOWED	请参见“集合函数用法无效”一节 《错误消息》。	-150
SQLE_ALIAS_NOT_UNIQUE	请参见“别名 '%1' 不唯一”一节 《错误消息》。	-830
SQLE_ALIAS_NOT_YET_DEFINED	请参见“别名 '%1' 的定义必须显示在其第一个引用之前”一节 《错误消息》。	-831
SQLE_AMBIGUOUS_INDEX_NAME	请参见“索引名 '%1' 不明确”一节 《错误消息》。	-678
SQLE_ARGUMENT_CANNOT_BE_NULL	请参见“过程 '%2' 的参数 %1 不能为 NULL”一节 《错误消息》。	-90
SQLE_BAD_ENCRYPTION_KEY	请参见“加密密钥不正确或遗失”一节 《错误消息》。	-840
SQLE_BAD_PARAM_INDEX	请参见“输入参数索引超出范围”一节 《错误消息》。	-689
SQLE_CANNOT_ACCESS_FILE	请参见“无法访问文件 '%1' -- %2”一节 《错误消息》。	-602
SQLE_CANNOT_ACCESS_FILESYSTEM	请参见“无法访问设备上的文件系统”一节 《错误消息》。	-1108
SQLE_CANNOT_ACCESS_SCHEMA_FILE	请参见“无法访问模式文件 '%1'”一节 《错误消息》。	-951
SQLE_CANNOT_CHANGE_ML_REMOTE_ID	请参见“如果上次上载状态未知，则无法更改 MobiLink 远程 ID”一节 《错误消息》。	-1118
SQLE_CANNOT_EXECUTE_STMT	请参见“语句无法执行”一节 《错误消息》。	111

成员名称	说明	值
SQLE_CANNOT_MODIFY	请参见“无法修改表 '%2' 中的列 '%1'”一节 《错误消息》。	-191
SQLE_CANNOT_OPEN_LOG	请参见“无法打开事务日志文件 -- %1”一节 《错误消息》。	-106
SQLE_CANNOT_REGISTER_LISTENER	请参见“无法注册指定的监听器”一节 《错误消息》。	-992
SQLE_CLIENT_OUT_OF_MEMORY	请参见“客户端内存不足”一节 《错误消息》。	-876
SQLE_COLUMN_AMBIGUOUS	请参见“在多个表中找到列 '%1' -- 需要相关名”一节 《错误消息》。	-144
SQLE_COLUMN_CANNOT_BE_NULL	请参见“表 '%2' 中的列 '%1' 不能为 NULL”一节 《错误消息》。	-195
SQLE_COLUMN_IN_INDEX	请参见“不能变更索引中的列”一节 《错误消息》。	-127
SQLE_COLUMN_NOT_FOUND	请参见“未找到列 '%1'”一节 《错误消息》。	-143
SQLE_COLUMN_NOT_INDEXED	请参见“列 '%1' 不是其所在表中的任何索引的一部分”一节 《错误消息》。	-1101
SQLE_COLUMN_NOT_STREAMABLE	请参见“因为列 '%1' 的类型不支持流操作，所以该操作失败”一节 《错误消息》。	-1100
SQLE_COMMUNICATIONS_ERROR	请参见“通信错误”一节 《错误消息》。	-85
SQLE_CONNECTION_ALREADY_EXISTS	请参见“此连接已存在”一节 《错误消息》。	-955
SQLE_CONNECTION_NOT_FOUND	请参见“未找到连接”一节 《错误消息》。	-108
SQLE_CONNECTION_RESTORED	请参见“UltraLite 连接已恢复”一节 《错误消息》。	133
SQLE_CONSTRAINT_NOT_FOUND	请参见“未找到约束 '%1'”一节 《错误消息》。	-929
SQLE_CONVERSION_ERROR	请参见“无法将 %1 转换为 %2”一节 《错误消息》。	-157
SQLE_CORRUPT_PAGE_READ_RETRY	请参见“重新尝试读取损坏页（第 '%1' 页）”一节 《错误消息》。	143

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
SQLE_CORRUPT_ULTRALITE_DATABASE	请参见“数据库页校验失败，并生成代码: %1”一节 《错误消息》。	-1186
SQLE_CORRUPT_ULTRALITE_INDEX	请参见“表 %1 的索引 %2 发生索引校验失败，并生成代码: %3”一节 《错误消息》。	-1185
SQLE_COULD_NOT_FIND_FUNCTION	请参见“无法在动态库 '%2' 中找到 '%1'”一节 《错误消息》。	-621
SQLE_COULD_NOT_LOAD_LIBRARY	请参见“无法装载动态库 '%1'”一节 《错误消息》。	-620
SQLE_CURSOR_ALREADY_OPEN	请参见“游标已打开”一节 《错误消息》。	-172
SQLE_CURSOR_NOT_DECLARED	请参见“尚未声明游标”一节 《错误消息》。	-170
SQLE_CURSOR_NOT_OPEN	请参见“游标未打开”一节 《错误消息》。	-180
SQLE_CURSOR_RESTORED	请参见“UltraLite 游标（或结果集或表）已恢复”一节 《错误消息》。	134
SQLE_CURSOROP_NOT_ALLOWED	请参见“试图进行非法游标操作”一节 《错误消息》。	-187
SQLE_DATABASE_ERROR	请参见“内部数据库错误 %1 -- 事务已回退”一节 《错误消息》。	-301
SQLE_DATABASE_NAME_REQUIRED	请参见“启动服务器需要数据库名”一节 《错误消息》。	-87
SQLE_DATABASE_NOT_CREATED	请参见“数据库创建失败: %1”一节 《错误消息》。	-645
SQLE_DATABASE_STATE_RESTORED	请参见“UltraLite 数据库状态已恢复”一节 《错误消息》。	142
SQLE_DATABASE_UPGRADE_WARNING	请参见“自动数据库升级已应用”一节 《错误消息》。	149
SQLE_DATATYPE_NOT_ALLOWED	请参见“表达式有不受支持的数据类型”一节 《错误消息》。	-624
SQLE_DBSPACE_FULL	请参见“某一 dbspace 已达到其文件大小的最大值”一节 《错误消息》。	-604

成员名称	说明	值
SQLE_DESCRIBE_NONSELECT	请参见“仅能描述一条 SELECT 语句”一节 《错误消息》。	-160
SQLE_DEVICE_ERROR	请参见“I/O 错误 %1 -- 事务已回退”一节 《错误消息》。	-305
SQLE_DEVICE_IO_FAILED	请参见“'%1' 的文件 I/O 失败”一节 《错误消息》。	-974
SQLE_DIV_ZERO_ERROR	请参见“除以零”一节 《错误消息》。	-628
SQLE_DOWNLOAD_CONFLICT	请参见“由于与现有行发生冲突，下载失败”一节 《错误消息》。	-839
SQLE_DOWNLOAD_RESTART_FAILED	请参见“无法重试下载，因为上载没有完成”一节 《错误消息》。	-1102
SQLE_DROP_DATABASE_FAILED	请参见“试图删除数据库 '%1' 失败”一节 《错误消息》。	-651
SQLE_DUPLICATE_CURSOR_NAME	请参见“游标名 '%1' 已经存在”一节 《错误消息》。	-683
SQLE_DUPLICATE_FOREIGN_KEY	请参见“表 '%2' 的外键 '%1' 与现有外键重复”一节 《错误消息》。	-251
SQLE_DUPLICATE_OPTION	请参见“选项 '%1' 被指定多次”一节 《错误消息》。	139
SQLE_DUPLICATE_ROW_FOUND_IN_DOWNLOAD	请参见“为表 '%1' 下载了具有相同主键的两行”一节 《错误消息》。	145
SQLE_DYNAMIC_MEMORY_EXHAUSTED	请参见“动态内存已耗尽”一节 《错误消息》。	-78
SQLE_ENCRYPTION_INITIALIZATION_FAILED	请参见“无法初始化加密 DLL: '%1'”一节 《错误消息》。	-984
SQLE_ENCRYPTION_NOT_ENABLED	请参见“尚未启用加密”一节 《错误消息》。	-1143
SQLE_ENCRYPTION_NOT_ENABLED_WARNING	请参见“尚未启用加密”一节 《错误消息》。	140
SQLE_ENGINE_ALREADY_RUNNING	请参见“数据库服务器已在运行”一节 《错误消息》。	-96

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
SQLE_ERROR	请参见“运行时 SQL 错误 -- %1”一节 《错误消息》。	-300
SQLE_ERROR_CALLING_FUNCTION	请参见“无法分配资源来调用外部函数”一节 《错误消息》。	-622
SQLE_ERROR_IN_ASSIGNMENT	请参见“赋值出错”一节 《错误消息》。	-641
SQLE_EVENT_NOT_FOUND	请参见“未找到事件 '%1'”一节 《错误消息》。	-771
SQLE_EVENT_NOTIFICATION_QUEUE_FULL	请参见“事件通知队列 '%1' 已满，通知被放弃”一节 《错误消息》。	146
SQLE_EVENT_NOTIFICATION_QUEUE_NOT_FOUND	请参见“未找到事件通知队列 '%1'”一节 《错误消息》。	-1263
SQLE_EVENT_NOTIFICATION_QUEUE_NOT_FOUND_WARN	请参见“未找到事件通知队列 '%1' 警告”一节 《错误消息》。	148
SQLE_EVENT_NOTIFICATION_QUEUE_TIMEOUT	请参见“队列 '%1' 上超时期内没有通知”一节 《错误消息》。	-1266
SQLE_EVENT_NOTIFICATIONS_LOST	请参见“在队列 '%1' 上丢失事件通知”一节 《错误消息》。	147
SQLE_EVENT_OBJECT_ALREADY_EXISTS	请参见“名称为 '%1' 的事件对象已经存在”一节 《错误消息》。	-1265
SQLE_EVENT_PARAMETER_NOT_FOUND	请参见“未找到事件参数 '%1'”一节 《错误消息》。	-1267
SQLE_EXPRESSION_ERROR	请参见“'%1' 附近的表达式无效”一节 《错误消息》。	-156
SQLE_FEATURE_NOT_ENABLED	请参见“未对您的应用程序启用您曾尝试调用的方法”一节 《错误消息》。	-1092
SQLE_FILE_BAD_DB	请参见“无法启动指定的数据库:'%1' 不是有效的数据库文件”一节 《错误消息》。	-1006
SQLE_FILE_IN_USE	请参见“指定的数据库文件已在使用”一节 《错误消息》。	-816
SQLE_FILE_NOT_DB	请参见“无法启动指定的数据库:'%1' 不是数据库”一节 《错误消息》。	-1004

成员名称	说明	值
SQLE_FILE_VOLUME_NOT_FOUND	请参见“找不到为数据库 '%1' 指定的文件系统卷”一节 《错误消息》。	-1112
SQLE_FILE_WRONG_VERSION	请参见“无法启动指定的数据库:'%1' 由本软件的另一版本创建”一节 《错误消息》。	-1005
SQLE_FOREIGN_KEY_NAME_NOT_FOUND	请参见“未找到外键名 '%1'”一节 《错误消息》。	-145
SQLE_IDENTIFIER_TOO_LONG	请参见“标识符 '%1' 过长”一节 《错误消息》。	-250
SQLE_INCORRECT_VOLUME_ID	请参见“'%1' 的卷 ID 不正确”一节 《错误消息》。	-975
SQLE_INDEX_NAME_NOT_UNIQUE	请参见“索引名 '%1' 不唯一”一节 《错误消息》。	-111
SQLE_INDEX_NOT_FOUND	请参见“无法找到名为 '%1' 的索引”一节 《错误消息》。	-183
SQLE_INDEX_NOT_UNIQUE	请参见“表 '%2' 的索引 '%1' 将不唯一”一节 《错误消息》。	-196
SQLE_INTERRUPTED	请参见“语句被用户中断”一节 《错误消息》。	-299
SQLE_INVALID_CONSTRAINT_REF	请参见“对约束 '%1' 的引用或操作无效”一节 《错误消息》。	-937
SQLE_INVALID_DESCRIPTOR_INDEX	请参见“无效的描述符索引”一节 《错误消息》。	-640
SQLE_INVALID_DESCRIPTOR_NAME	请参见“无效的 SQL 描述符名”一节 《错误消息》。	-642
SQLE_INVALID_DISTINCT_AGGREGATE	请参见“分组查询包含多个不同的集合函数”一节 《错误消息》。	-863
SQLE_INVALID_EVENT_OBJECT_NAME	请参见“事件对象名称 '%1' 无效”一节 《错误消息》。	-1264
SQLE_INVALID_FOREIGN_KEY	请参见“表 '%2' 中的外键 '%1' 没有主键值”一节 《错误消息》。	-194
SQLE_INVALID_FOREIGN_KEY_DEF	请参见“外键的列 '%1' 与主键定义不同”一节 《错误消息》。	-113

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
SQLE_INVALID_GROUP_SELECT	请参见“对 '%1' 的函数或列引用还必须出现在 GROUP BY 中”一节 《错误消息》。	-149
SQLE_INVALID_INDEX_TYPE	请参见“'%1' 的索引类型说明无效”一节 《错误消息》。	-650
SQLE_INVALID_LOGON	请参见“无效的用户 ID 或口令”一节 《错误消息》。	-103
SQLE_INVALID_OPTION	请参见“无效的选项 '%1' -- 不存在 PUBLIC 设置”一节 《错误消息》。	-200
SQLE_INVALID_OPTION_SETTING	请参见“选项 '%1' 的设置无效”一节 《错误消息》。	-201
SQLE_INVALID_OPTION_VALUE	请参见“'%1' 是 '%2' 的无效值”一节 《错误消息》。	-1053
SQLE_INVALID_ORDER	请参见“ORDER BY 说明无效”一节 《错误消息》。	-152
SQLE_INVALID_PARAMETER	请参见“无效的参数”一节 《错误消息》。	-735
SQLE_INVALID_PARSE_PARAMETER	请参见“分析错误: %1”一节 《错误消息》。	-95
SQLE_INVALID_SQL_IDENTIFIER	请参见“无效的 SQL 标识符”一节 《错误消息》。	-760
SQLE_INVALID_STATEMENT	请参见“无效的语句”一节 《错误消息》。	-130
SQLE_INVALID_UNION	请参见“UNION、INTERSECT 或 EXCEPT 中的 SELECT 列表长度不匹配”一节 《错误消息》。	-153
SQLE_KEYLESS_ENCRYPTION	请参见“无法执行请求的操作，因为此数据库使用无密钥加密”一节 《错误消息》。	-1109
SQLE_LOCKED	请参见“用户 '%1' 锁定了 '%2' 中的行”一节 《错误消息》。	-210
SQLE_MAX_ROW_SIZE_EXCEEDED	请参见“将超出表 '%1' 的最大行大小”一节 《错误消息》。	-1132
SQLE_MEMORY_ERROR	请参见“内存错误 -- 事务已回退”一节 《错误消息》。	-309

成员名称	说明	值
SQLE_METHOD_CANNOT_BE_CALLED	请参见“此时不能调用方法 '%1'”一节 《错误消息》。	-669
SQLE_MIRROR_FILE_MISMATCH	请参见“镜像 '%1' 与数据库 '%2' 不匹配”一节 《错误消息》。	-1138
SQLE_MIRROR_FILE_REQUIRES_CHECKSUMS	请参见“镜像文件需要更高的 checksum_level”一节 《错误消息》。	144
SQLE_MOBILINK_COMMUNICATIONS_ERROR	请参见“MobiLink 通信错误；代码: %1，参数: %2，系统代码 %3”一节 《错误消息》。	-1305
SQLE_NAME_NOT_UNIQUE	请参见“项 '%1' 已经存在”一节 《错误消息》。	-110
SQLE_NO_COLUMN_NAME	请参见“派生表 '%1' 没有列 %2 的名称”一节 《错误消息》。	-163
SQLE_NO_CURRENT_ROW	请参见“没有当前的游标行”一节 《错误消息》。	-197
SQLE_NO_INDICATOR	请参见“未给 NULL 结果提供指示符变量”一节 《错误消息》。	-181
SQLE_NO_MATCHING_SELECT_ITEM	请参见“派生表 '%1' 的选择列表没有与 '%2' 匹配的表达式”一节 《错误消息》。	-812
SQLE_NO_PRIMARY_KEY	请参见“表 '%1' 无主键”一节 《错误消息》。	-118
SQLE_NOERROR	SQLE_NOERROR(0) - 此代码表示没有错误或警告。	0
SQLE_NON_UPDATEABLE_COLUMN	请参见“不能更新表达式”一节 《错误消息》。	-190
SQLE_NON_UPDATEABLE_CURSOR	请参见“为 READ ONLY 游标指定了不正确的 FOR UPDATE”一节 《错误消息》。	-813
SQLE_NOT_IMPLEMENTED	请参见“未实现功能 '%1'”一节 《错误消息》。	-134
SQLE_NOT_SUPPORTED_IN_ULTRALITE	请参见“在 UltraLite 中功能不可用”一节 《错误消息》。	-749
SQLE_NOTFOUND	请参见“未找到行”一节 《错误消息》。	100
SQLE_ONLY_ONE_TABLE	请参见“游标上的 INSERT/DELETE 只能修改一个表”一节 《错误消息》。	-199

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
SQLE_OVERFLOW_ERROR	请参见“值 %1 超出了目标的范围”一节 《错误消息》。	-158
SQLE_PAGE_SIZE_INVALID	请参见“无效的数据库页面大小”一节 《错误消息》。	-644
SQLE_PARTIAL_DOWNLOAD_NOT_FOUND	请参见“未找到部分下载”一节 《错误消息》。	-1103
SQLE_PASSTHROUGH_SQL_SCRIPT_FAILED_E	请参见“直通 SQL 脚本失败”一节 《错误消息》。	-1238
SQLE_PASSTHROUGH_SQL_SCRIPT_FAILED_W	请参见“直通 SQL 脚本失败”一节 《错误消息》。	141
SQLE_PERMISSION_DENIED	请参见“权限被拒绝: %1”一节 《错误消息》。	-121
SQLE_PRIMARY_KEY_NOT_UNIQUE	请参见“表 '%1' 的主键不唯一:主键值 ('%2')”一节 《错误消息》。	-193
SQLE_PRIMARY_KEY_TWICE	请参见“表不能有两个主键”一节 《错误消息》。	-126
SQLE_PRIMARY_KEY_VALUE_REF	请参见“表 '%1' 中行的主键被表 '%3' 中的外键 '%2' 引用”一节 《错误消息》。	-198
SQLE_PUBLICATION_NOT_FOUND	请参见“未找到发布 '%1'”一节 《错误消息》。	-280
SQLE_PUBLICATION_PREDICATE_IGNORED	请参见“发布谓语未被评估”一节 《错误消息》。	138
SQLE_RESOURCE_GOVERNOR_EXCEEDED	请参见“超出 '%1' 的资源调控器”一节 《错误消息》。	-685
SQLE_ROW_DELETED_TO_MAINTAIN_REFERENTIAL_INTEGRITY	请参见“为了保持参照完整性，已从表 %1 中删除行”一节 《错误消息》。	137
SQLE_ROW_DROPPED_DURING_SCHEMA_UPGRADE	请参见“由于未能将行转换成新的模式格式，所以该行被删除”一节 《错误消息》。	130
SQLE_ROW_EXCEEDS_PAGE_SIZE	请参见“无法存储行，因为它超过了数据库页面大小”一节 《错误消息》。	-1117
SQLE_ROW_UPDATED_SINCE_READ	请参见“上次读取后行已更改 -- 操作被取消”一节 《错误消息》。	-208

成员名称	说明	值
SQLE_SCHEMA_UPGRADE_NOT_ALLOWED	请参见“当前不允许模式升级”一节 《错误消息》。	-953
SQLE_SERVER_SYNCHRONIZATION_ERROR	请参见“由于服务器出错，同步失败: %1”一节 《错误消息》。	-857
SQLE_START_STOP_DATABASE_DENIED	请参见“启动/停止数据库的请求被拒绝”一节 《错误消息》。	-75
SQLE_STATEMENT_ERROR	请参见“SQL 语句错误”一节 《错误消息》。	-132
SQLE_STRING_RIGHT_TRUNCATION	请参见“字符串数据右截断”一节 《错误消息》。	-638
SQLE_SUBQUERY_RESULT_NOT_UNIQUE	请参见“子查询不能返回多个行”一节 《错误消息》。	-186
SQLE_SUBQUERY_SELECT_LIST	请参见“子查询只允许一个选择列表项”一节 《错误消息》。	-151
SQLE_SYNC_INFO_INVALID	请参见“同步信息不完整或无效，请检查 '%1'”一节 《错误消息》。	-956
SQLE_SYNC_INFO_REQUIRED	请参见“未提供同步信息”一节 《错误消息》。	-1111
SQLE_SYNC_NOT_REENTRANT	请参见“同步进程无法重新进入同步”一节 《错误消息》。	-1110
SQLE_SYNC_PROFILE_INVALID	请参见“同步配置文件 '%1' 具有无效的参数 '%2'”一节 《错误消息》。	-1224
SQLE_SYNC_PROFILE_NOT_FOUND	请参见“未找到同步配置文件 '%1'”一节 《错误消息》。	-1217
SQLE_SYNC_STATUS_UNKNOWN	请参见“上次同步上载的状态未知”一节 《错误消息》。	-952
SQLE_SYNTAX_ERROR	请参见“第 %2 行的 '%1' 附近有语法错误”一节 《错误消息》。	-131
SQLE_TABLE_ALREADY_INCLUDED	请参见“已包括表 '%1'”一节 《错误消息》。	-822
SQLE_TABLE_HAS_PUBLICATIONS	请参见“表 '%1' 有发布”一节 《错误消息》。	-281

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
SQLE_TABLE_IN_USE	请参见“表正在使用”一节 《错误消息》。	-214
SQLE_TABLE_NOT_FOUND	请参见“未找到表 '%1'”一节 《错误消息》。	-141
SQLE_TOO_MANY_BLOB_REFS	请参见“对 BLOB 的引用太多”一节 《错误消息》。	-1107
SQLE_TOO_MANY_CONNECTIONS	请参见“超出数据库服务器连接限制”一节 《错误消息》。	-102
SQLE_TOO_MANY_CURSORS	请参见“打开的游标过多”一节 《错误消息》。	-1230
SQLE_TOO_MANY_PUBLICATIONS	请参见“为操作指定的发布过多”一节 《错误消息》。	-1106
SQLE_TOO_MANY_TEMP_TABLES	请参见“连接中临时表过多”一节 《错误消息》。	-817
SQLE_TOO_MANY_USERS	请参见“数据库中的用户太多”一节 《错误消息》。	-1104
SQLE_TRUNCATED	请参见“值被截断”一节 《错误消息》。	101
SQLE_ULTRALITE_DATABASE_NOT_FOUND	请参见“未找到数据库 '%1'”一节 《错误消息》。	-954
SQLE_ULTRALITE_OBJ_CLOSED	请参见“在已关闭的对象上的操作无效”一节 《错误消息》。	-908
SQLE_UNABLE_TO_CONNECT	请参见“无法启动数据库 -- %1”一节 《错误消息》。	-105
SQLE_UNABLE_TO_START_DATABASE	请参见“无法启动指定的数据库: %1”一节 《错误消息》。	-82
SQLE_UNABLE_TO_START_DATABASE_VER_NEWER	请参见“无法启动指定的数据库:必须升级服务器才能启动数据库 %1”一节 《错误消息》。	-934
SQLE_UNKNOWN_FUNC	请参见“未知函数 '%1'”一节 《错误消息》。	-148
SQLE_UNKNOWN_OPTION	请参见“'%1' 是未知选项”一节 《错误消息》。	120
SQLE_UNKNOWN_USERID	请参见“用户 ID '%1' 不存在”一节 《错误消息》。	-140

成员名称	说明	值
SQLE_UNRECOGNIZED_OPTION	请参见“无法识别选项 '%1'”一节 《错误消息》。	-1002
SQLE_UPLOAD_FAILED_AT_SERVER	请参见“同步服务器无法提交上载”一节 《错误消息》。	-794
SQLE_VALUE_IS_NULL	请参见“无法将 NULL 结果作为所需数据类型返回”一节 《错误消息》。	-1050
SQLE_VARIABLE_INVALID	请参见“无效的主机变量”一节 《错误消息》。	-155
SQLE_WRONG_NUM_OF_INSERT_COLS	请参见“INSERT 的值数目错误”一节 《错误消息》。	-207
SQLE_WRONG_PARAMETER_COUNT	请参见“函数 '%1' 的参数数目错误”一节 《错误消息》。	-154

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULSqlPassthroughProgressListener 接口

UL Ext.: 用于接收 SQL 直通脚本进度事件的监听器接口。

语法

Visual Basic
Public Interface **ULSqlPassthroughProgressListener**

C#
public interface **ULSqlPassthroughProgressListener**

另请参见

- “ULSqlPassthroughProgressListener 成员”一节第 453 页

ULSqlPassthroughProgressListener 成员

公共方法

成员名称	说明
“ScriptProgressed 方法”一节第 453 页	当 SQL 直通脚本运行时调用，用以向应用程序通知进度。此方法应返回 true 来取消进一步执行脚本或返回 false 来继续执行。

另请参见

- “ULSqlPassthroughProgressListener 接口”一节第 453 页

ScriptProgressed 方法

当 SQL 直通脚本运行时调用，用以向应用程序通知进度。此方法应返回 true 来取消进一步执行脚本或返回 false 来继续执行。

语法

Visual Basic
```
Public Function ScriptProgressed( _
  ByVal data As ULSqlProgressData _
) As Boolean
```

C#
```
public bool ScriptProgressed(
  ULSqlProgressData data
);
```

参数

- **data** 包含当前脚本相关信息的 ULSqlProgressData 对象。

返回值

此方法应返回 true 来取消脚本处理或返回 false 来继续执行。

注释

在调用 ScriptProgressed 的过程中，不应该调用任何 UltraLite.NET API 方法。

另请参见

- “ULSqlPassthroughProgressListener 接口”一节第 453 页
- “ULSqlPassthroughProgressListener 成员”一节第 453 页
- “ULSqlProgressData 类”一节第 455 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULSqlProgressData 类

UL Ext.: 返回 SQL 直通脚本进度监控数据。

语法

Visual Basic
Public Class **ULSqlProgressData**

C#
public class **ULSqlProgressData**

另请参见

- “ULSqlProgressData 成员”一节第 455 页
- “ULSqlPassthroughProgressListener 接口”一节第 453 页

ULSqlProgressData 成员

公共属性

成员名称	说明
“CurrentScript 属性”一节第 455 页	到目前为止已执行的脚本的索引。
“ScriptCount 属性”一节第 456 页	返回所执行的脚本的数目。
“State 属性”一节第 456 页	返回当前的进度状态。

另请参见

- “ULSqlProgressData 类”一节第 455 页
- “ULSqlPassthroughProgressListener 接口”一节第 453 页

CurrentScript 属性

到目前为止已执行的脚本的索引。

语法

Visual Basic
Public Readonly Property **CurrentScript** As Long

C#
public long **CurrentScript** { get;}

属性值

所执行的脚本的当前索引。

另请参见

- “ULSqlProgressData 类”一节第 455 页
- “ULSqlProgressData 成员”一节第 455 页

ScriptCount 属性

返回所执行的脚本的数目。

语法

Visual Basic
Public Readonly Property **ScriptCount** As Long

C#
public long **ScriptCount** { get;}

属性值

所执行的脚本的数目。

另请参见

- “ULSqlProgressData 类”一节第 455 页
- “ULSqlProgressData 成员”一节第 455 页

State 属性

返回当前的进度状态。

语法

Visual Basic
Public Readonly Property **State** As ULSqlProgressState

C#
public ULSqlProgressState **State** { get;}

属性值

指定当前 SQL 直通回调状态的 ULSqlProgressState 值之一。

另请参见

- “ULSqlProgressData 类”一节第 455 页
- “ULSqlProgressData 成员”一节第 455 页
- “ULSyncProgressState 枚举”一节第 503 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULSqlProgressState 枚举

UL Ext.: 枚举执行 SQL 直通脚本时可出现的所有状态。

语法

Visual Basic
Public Enum **ULSqlProgressState**

C#
public enum **ULSqlProgressState**

成员

成员名称	说明	值
STATE_DONE	脚本已成功完成。	2
STATE_ERROR	脚本已完成，但出现了错误。	3
STATE_RUNNING_SCRIPT	当前正在运行 SQL 直通脚本。	1
STATE_STARTING	尚未执行任何脚本。	0

另请参见

- “ULSqlProgressData 类”一节第 455 页

ULStreamErrorCode 枚举

UL Ext.: 枚举流在同步过程中可能报告的错误代码。

语法

Visual Basic
Public Enum **ULStreamErrorCode**

C#
public enum **ULStreamErrorCode**

成员

成员名称	说明	值
ACTSYNC_NO_PORT	ActiveSync 同步只能由 ActiveSync 自身启动，方法是：将设备放入底座，或从 ActiveSync 管理器选择 [同步]。从应用程序启动同步时，请使用 TCP/IP 套接字同步流。	75
ACTSYNC_NOT_INSTALLED	尚未安装 ActiveSync 提供程序。请运行 mlasinst 进行安装（有关详细信息，请参见文档）。	76
AUTHENTICATION_FAILED	客户端无法将自身向 MobiLink 验证。	249
CONNECT_TIMEOUT	连接尝试超时。服务器不在指定的主机和端口上运行，或者需要增加超时值以允许使用更多的时间进行连接。	241
COULD_NOT_OPEN_FILE	无法打开指定文件。	264
COULD_NOT_OPEN_FILE_FOR_WRITE	无法打开指定的文件以进行写入。请确保这是正确的文件且没有其它应用程序在使用它。	230
CREATE_RANDOM_OBJECT	安全网络层无法创建随机数生成对象。请释放系统资源、重新连接，然后重试该操作。	17
DEQUEUING_CONNECTION	当试图获取排队连接（同步）请求时，MobiLink 服务器遇到错误。释放系统资源。如果问题仍出现，请重新启动 MobiLink 服务器。	19
DUN_DIAL_FAILED	自动拨号无法建立与指定拨号网络之间的连接。	204

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
DUN_NOT_SUPPORTED	拨号尝试因系统支持不足而失败。在 PocketPC 上，必须使用 cellcore.dll；而在 Windows 上，则必须使用 IE 4.0（或更高版本）的 wininet.dll。其它平台不支持拨号。	203
E2EE_DECODING_PRIVATE_KEY_FILE	已经找到文件并且读取了它的内容，但对文件进行解码时出现了错误。请与技术支持人员联系并提供错误代码。	257
E2EE_INIT_ECC	尝试初始化 ECC 时出现错误。请确保已安装了 ECC 选项。	262
E2EE_INVALID_TYPE	指定了无效的 e2ee_type。请指定一个有效值。	261
E2EE_MISMATCHED_KEYS	客户端和服务器之间无法通信，因为远程用于端对端加密的 e2ee_public_key 与服务器的 e2ee_private_key 不匹配。	253
E2EE_MISSING_PRIVATE_KEY	指定了另一个端对端加密选项，但不是 e2ee_private_key 选项。要么指定所有的端对端加密选项，要么将它们都删除。所需的端对端加密选项包括：e2ee_type、e2ee_private_key、e2ee_private_key_password。	260
E2EE_MISSING_PRIVATE_KEY_PASSWORD	如果没有 e2ee_private_key_password，将不能读取 e2ee_private_key 文件。请提供 e2ee_private_key_password。	259
E2EE_NO_PRIVATE_KEY_IN_FILE	给定的文件名不包含专用密钥。	256
E2EE_PUBLIC_KEY	尝试读取端对端加密公共密钥时出错。	263
E2EE_READING_PRIVATE_KEY	读取 e2ee_private_key 文件时出现错误。请与技术支持人员联系并提供错误代码。	255
E2EE_READING_PRIVATE_KEY_FILE	无法读取给定文件。请与技术支持人员联系并提供错误代码。	258
E2EE_UNEXPECTED_PRIVATE_KEY_TYPE	在 e2ee_private_key 文件中找到的专用密钥类型与 e2ee_type 指定的类型不匹配。	254

成员名称	说明	值
E2EE_UNEXPECTED_PUBLIC_KEY_ENC_TYPE	客户端发送的 e2ee_type 值与在服务器上指定的 e2ee_type 值不同。请确保这两者相同。	252
E2EE_UNKNOWN_PUBLIC_KEY_ENC_TYPE	客户端发送了无法被服务器识别的 e2ee_type 值。请确保服务器的版本等于或高于远程服务器的版本。	251
END_READ	无法完成从网络进行读取的操作序列。另请参见：READ	11
END_WRITE	无法完成写入到网络的操作序列。另请参见：WRITE	10
GENERATE_RANDOM	安全网络层需要一个随机数但却无法生成。请释放系统资源，重新连接后再尝试该操作。	14
HTTP_AUTHENTICATION_FAILED	提供的用户 ID 和口令被拒绝。请检查其输入是否正确。如果正确，请与您的系统管理员联系，以确保您具有适当的权限。	211
HTTP_AUTHENTICATION_REQUIRED	HTTP 服务器或网关请求进行 HTTP 验证。请使用 HTTP 同步参数 http_userid 和 http_password 提供用户 ID 和口令。	209
HTTP_BAD_STATUS_CODE	请检查状态行以确定故障的原因。	86
HTTP_BUFFER_SIZE_OUT_OF_RANGE	修正 HTTP 缓冲区大小。有效的缓冲区大小为正数，并且对于主机平台不能太大。	79
HTTP_CHUNK_LEN_BAD_CHARACTER	尝试使用固定长度的 HTTP 正文。	85
HTTP_CHUNK_LEN_ENCODED_MISSING	尝试使用固定长度的 HTTP 正文。	84
HTTP_CLIENT_ID_NOT_SET	未将客户端 ID 传递到 HTTP 客户端代码。请向技术支持人员咨询，解决该问题。	78
HTTP_CONTENT_TYPE_NOT_SPECIFIED	指定了未知的内容类型。请参见相关文档，并将内容类型更改为受支持的类型之一。	77
HTTP_CRLF_ENCODED_MISSING	您使用的代理可能与 MobiLink 不兼容。请检查您的配置。	81

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
HTTP_CRLF_MISSING	您使用的代理可能与 MobiLink 不兼容。请检查您的配置。	82
HTTP_EXPECTED_POST	您使用的代理可能与 MobiLink 不兼容。请检查您的配置。	89
HTTP_EXTRA_DATA_END_READ	在 HTTP 正文中引入了额外的数据。这可能是由代理程序添加的。请尝试删除代理。	80
HTTP_HEADER_PARSE_ERROR	尝试分析 HTTP 标头时出错。标头可能存在格式错误。	216
HTTP_INTERNAL_HEADER_STATE	解码 HTTP 标头时出现问题。这是一个绝不应该出现的内部错误。请联系技术支持部门。	236
HTTP_INTERNAL_REQUEST_TYPE	确定 HTTP 请求类型时出现问题。这是一个决不应该出现的内部错误。请联系技术支持部门。	237
HTTP_INVALID_CHARACTER	在 HTTP 标头中读取到意外字符。该 HTTP 标头可能存在格式错误，或者另一端可能根本就不能发送 HTTP。	219
HTTP_INVALID_SESSION_KEY	指定了未知的会话密钥类型。请参见相关文档，并将会话密钥类型更改为支持的类型之一。	244
HTTP_MALFORMED_SESSION_COOKIE	用来管理同步会话的 HTTP cookie 已损坏。确定 Cookie 的损坏位置。最有可能的原因是客户端错误，或者可能是 HTTP 中间错误。	235
HTTP_NO_CONTD_CONNECTION	服务器在等待来自远程站点的下一个 HTTP 请求时超时。找出该请求未能到达服务器的原因，或尝试使用持久性连接。	83
HTTP_NO_PASSWORD	提供了 HTTP 验证所需的用户 ID，但未提供口令。验证时需要提供上述两项内容。	214
HTTP_NO_USERID	提供了 HTTP 验证所需的口令，但未提供用户 ID。验证时需要提供上述两项内容。	213
HTTP_PROXY_AUTHENTICATION_FAILED	提供的用户 ID 和口令被代理服务器拒绝。请检查输入是否正确。如果正确，请与您的系统管理员联系，以确保您具有适当的权限。	212
HTTP_PROXY_AUTHENTICATION_REQUIRED	HTTP 代理请求进行 HTTP 验证。请使用 HTTP 同步参数 http_proxy_userid 和 http_proxy_password 提供用户 ID 和口令。	210

成员名称	说明	值
HTTP_SERVER_AUTH_FAILED	从服务器发送的验证信息标头包含一个不正确的值，导致验证失败。请确保您正连接到合法的 HTTP 服务器。	217
HTTP_UNABLE_TO_PARSE_COOKIE	确定设置的 Cookie 标头的损坏位置。	88
HTTP_UNKNOWN_TRANSFER_ENCODING	确定未知传输编码是如何产生的。	87
HTTP_UNSUPPORTED_AUTH_ALGORITHM	服务器请求的 HTTP 摘要式验证算法不受支持。只支持 "MD5" 和 "MD5-sess" 算法。	215
HTTP_VERSION	所请求的 HTTP 版本不受支持。请查阅文档并指定一个支持的 HTTP 版本。发布时，支持的 HTTP 版本为 1.0 和 1.1。	54
INCONSISTENT_FIPS	在 MobiLink 服务器命令行上使用 -fips 开关要求所有安全流都符合 FIPS 标准。如果安全流未配置 fips 选项，则将自动符合 FIPS 标准（如 fips=y）。将 fips 选项从安全流中删除，或通过 fips=y 将其启用。	242
INIT_RANDOM	安全网络层无法初始化随机数生成器。请释放系统资源，重新连接后再尝试该操作。	15
INTERNAL	网络层发生内部错误。请联系技术支持部门。	220
INTERNAL_API	网络层发生内部错误。请联系技术支持部门。	238
INTERNAL_PROTOCOL_NOT_LOADED	无法装载同步协议。如果您正在使用 UltraLite，请确保已调用了正确的 ULEnable 方法。	227
INTERRUPTED	当前操作被调用方中断。	218
INVALID_COMPRESSION_TYPE	无法识别指定的压缩类型。	232
INVALID_LOCAL_PATH	下载文件的本地路径无效。有关详细信息，请查阅相关文档。	243
INVALID_NETWORK_LIBRARY	可能因为无效或损坏，给定的网络接口 DLL 或共享对象无法装载。	245

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
INVALID_SYNC_PROTOCOL	指定的协议不是有效的同步协议。	226
LIBRARY_ENTRY_POINT_NOT_FOUND	无法找到指定的库入口点。	225
LOAD_LIBRARY_FAILURE	无法在路径中找到指定的库。如果您尝试使用 TLS 加密进行同步，则请确保您已获取了正确的许可。	224
LOAD_NETWORK_LIBRARY	无法找到和/或装载网络接口库。请检查下列各项： 1) 正确安装了套接字层。正确的网络接口库（或 DLL 或共享对象）必须存在并且可以访问。 2) 有足够的系统资源可用。如果运行缓慢，请释放系统资源。	73
MEMORY_ALLOCATION	网络层无法分配足够的存储字节。释放系统内存并重新尝试该操作。释放系统内存的方法取决于操作系统及其配置方式。最简单的方法就是减少活动进程的数目。有关详细信息，请查阅操作系统文档。	6
MISSING_PARAMETER	需要使用指定参数，但未提供该参数。	229
NETWORK_LIBRARY_VERSION_MISMATCH	因为版本错误，网络接口 DLL 或共享对象无法装载。	248
NO_ECC_FIPS	执行给定的压缩操作时出现了问题。请联系技术支持部门。	239
NONE	该代码指示无网络错误或发生了未知的网络错误。	0
NOT_IMPLEMENTED	请求的内部功能尚未实现。请联系技术支持部门。	12
PARAMETER	网络参数的格式为 "名称=值;[名称 2=值 2[;...]]"。该代码指示一个无效的参数值。有关对应的参数名，请查阅文档并纠正参数值。	1

成员名称	说明	值
PARAMETER_NOT_BOOLEAN	网络参数的格式为 "名称=值;[名称 2=值 2[;...]]"。参数值不是布尔值。找到不符合上述格式的参数并将参数值更改为 0（对应于 off 或 false）或者 1（对应于 on 或 true）。	4
PARAMETER_NOT_HEX	网络参数的格式为 "名称=值;[名称 2=值 2[;...]]"。参数值不是十六进制值（基数为 16）。找到不符合上述格式的参数并将参数值更改为十六进制值。	5
PARAMETER_NOT_UINT32	网络参数的格式为 "名称=值;[名称 2=值 2[;...]]"。参数值不是一个无符号的整数。找到不符合上述格式的参数并将参数值更改为无符号整数。	2
PARAMETER_NOT_UINT32_RANGE	网络参数的格式为 "名称=值;[名称 2=值 2[;...]]"。参数值不是一个无符号的整数值或范围。找到不符合上述格式的参数并将参数值更改为无符号整数或无符号范围。无符号范围的格式为：NNN-NNN。	3
PARSE	网络参数的格式为 "名称=值;[名称 2=值 2[;...]]"。参数的完整列表可以（但非必须）包括在圆括号中。给定字符串并不遵循该约定。请检查字符串，修正格式问题后重新尝试该操作。	7
PROTOCOL_ERROR	读取了意外的值或标识。	231

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
READ	无法从网络层读取给定的字节数。请注意，读取操作可能作为大型网络操作的一部分出现。例如，一些网络层具有子层，而子层可以执行多个读和写操作，这些操作又是上层基本操作的一部分。读取错误通常是由以下原因之一导致的： 1) 网络存在导致读取失败的问题。重新连接并重试该操作。 2) 连接超时。重新连接后再尝试该操作。 3) 连接的另外一方完全终止连接。请查阅客户端和/或服务器的错误日志，它们将指明连接断开的原因。请查阅输出日志错误并修正错误，然后重试该操作。 4) 位于连接另外一端的进程被异常中断。请查阅客户端和/或服务器的错误消息日志，它们将指明进程异常中断的原因。如果进程是通过非正常方法关闭，它的消息日志中就可能不会记录任何错误。重新连接后再尝试该操作。 5) 系统资源不足，无法执行读取操作。请释放系统资源，重新连接后再尝试该操作。如果随后重新尝试的努力失败，请向网络管理员进行咨询。	8
READ_TIMEOUT	无法在给定时间内从网络层读取给定的字节数。请检查网络是否工作正常、发送应用程序是否仍在运行。	201
SACI_ERROR_CLIENT	已由 SACI 实现报告的错误。	246
SACI_ERROR_SERVER	SACI 网络库正在报告错误。请与 SACI 网络库的提供商进行联系以解决此问题。	247
SACI_IMPLEMENTATION_MISMATCH	无法装载 SACI 实现，因为它有一个不兼容的实现 ID。	250
SECURE_ADD_CERTIFICATE	安全网络层无法向证书链添加证书。释放系统资源后重新尝试该操作。	39
SECURE_ADD_TRUSTED_CERTIFICATE	安全网络层无法将受信任证书添加到证书链中。最有可能的原因是系统资源不足。释放系统资源后重新尝试该操作。	48

成员名称	说明	值
SECURE_CERTIFICATE_COMMON_NAME	给定的公用名不在证书链中。请检查以下内容： 1) 正确输入了公用名称。 2) 指定的证书文件正确。 3) 公用名称在证书链中。您可以使用实用程序 viewcert 进行验证。	52
SECURE_CERTIFICATE_COMPANY_NAME	给定的组织名称不在证书链中。请检查以下内容： 1) 正确输入了组织名称。 2) 指定的证书文件正确。 3) 组织名称在证书链中。您可以使用实用程序 viewcert 进行验证。	21
SECURE_CERTIFICATE_COMPANY_UNIT	给定的组织单位不在证书链中。请检查以下内容： 1) 正确输入了公司名称。 2) 指定的证书文件正确。 3) 公司名称在证书链中。您可以使用实用程序 viewcert 进行验证。	51
SECURE_CERTIFICATE_COUNT	给定文件不包含证书。请检查以下内容： 1) 正确指定了证书文件名。 2) 证书文件包含一个或多个证书。 3) 证书文件包含正确的证书。	42
SECURE_CERTIFICATE_EXPIRED	证书链中的证书已到期。获取一个尚未过期的新证书并重新尝试该操作。	50
SECURE_CERTIFICATE_EXPIRY_DATE	无法读取证书的到期日期。请检查以下内容： 1) 正确输入了口令。 2) 证书文件包含一个或多个证书。 3) 证书文件包含正确的证书。 4) 证书文件完好无损。	37

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
SECURE_CERTIFICATE_FILE_NOT_FOUND	无法打开证书文件。请检查以下内容： 1) 正确指定了证书文件名。 2) 证书文件存在。 3) 证书文件包含一个或多个证书。 4) 证书文件包含正确的证书。 5) 试图打开证书文件的程序有读取该文件的足够权限。这仅适用于具有用户和/或文件权限的操作系统。	33
SECURE_CERTIFICATE_NOT_TRUSTED	服务器证书没有获得受信任机构的签名。请检查以下内容： 1) 正确指定了证书文件名。 2) 证书文件包含一个或多个证书。 3) 证书文件包含正确的证书。 4) 受信任的根证书的客户列表包含服务器的根证书。	24
SECURE_CERTIFICATE_ROOT	证书链中的根证书无效。发布时，该错误虽然定义但并没有使用。	20
SECURE_CREATE_CERTIFICATE	安全网络层无法为证书分配存储空间。释放系统资源后重新尝试该操作。	43
SECURE_CREATE_PRIVATE_KEY_OBJECT	安全网络层无法在装载专用密钥之前创建专用密钥对象。最可能的原因是系统资源不足。释放系统资源后重新尝试该操作。	49
SECURE_DUPLICATE_CONTEXT	安全网络层无法复制安全环境。释放系统资源后重新尝试该操作。	25
SECURE_EXPORT_CERTIFICATE	安全网络层无法复制证书。释放系统资源后重新尝试该操作。	38
SECURE_HANDSHAKE	安全握手失败。请检查以下内容： 1) 在客户端一方，指定了正确的主机和端口号。 2) 在服务器一方，指定了正确的端口号。 3) 在客户端一方指定了正确的受信任证书，同时在服务器一方指定了正确的标识文件。	53

成员名称	说明	值
SECURE_IMPORT_CERT_FROM_SYSTEM_STORE	无法从系统证书存储库导入证书。	222
SECURE_IMPORT_CERTIFICATE	安全网络层无法导入证书。请检查以下内容： 1) 正确指定了证书文件名。 2) 证书文件存在。 3) 证书文件包含一个或多个证书。 4) 证书文件包含正确的证书。	44
SECURE_NO_CERTS_IN_SYS_STORE	在系统证书存储库中未找到证书。	223
SECURE_NO_SERVER_CERTIFICATE	未提供服务器证书。安全通信需要服务器证书。提供的文件必须包含服务器的完整证书链及其专用密钥。	205
SECURE_NO_SERVER_CERTIFICATE_PASSWORD	未提供服务器证书口令。对服务器的加密专用密钥进行解密时需要此口令。	206
SECURE_NO_TRUSTED_ROOTS	未提供受信任的根证书。安全通信至少需要一个受信任的根证书。	207
SECURE_OPEN_SYSTEM_CERT_STORE	尝试打开系统证书存储库失败。	221
SECURE_READ_CERTIFICATE	证书文件无法读取。请检查以下内容： 1) 正确输入了口令。 2) 证书文件包含一个或多个证书。 3) 证书文件包含正确的证书。 4) 证书文件完好无损。	34
SECURE_READ_PRIVATE_KEY	无法从证书文件中读取专用密钥。请检查以下内容： 1) 正确输入了口令。 2) 证书文件包含一个或多个证书。 3) 证书文件包含正确的证书。 4) 证书文件完好无损。	35

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
SECURE_REDUNDANT_SERVER_CERTIFICATE_PASSWORD	在未用任何口令对服务器的专用密钥进行加密时，指定了一个口令。	208
SECURE_SET_IO	安全网络层无法与网络层相连。释放系统资源后重新尝试该操作。	26
SECURE_SET_PRIVATE_KEY	专用密钥无法使用。请检查以下内容： 1) 正确输入了口令。 2) 证书文件包含一个或多个证书。 3) 证书文件包含正确的证书。 4) 证书文件完好无损。	36
SECURE_TRUSTED_CERTIFICATE_FILE_NOT_FOUND	无法找到证书文件。请检查以下内容： 1) 正确指定了证书文件名。 2) 证书文件存在。 3) 证书文件包含一个或多个证书。 4) 证书文件包含正确的证书。 5) 试图打开证书文件的程序具有查看该文件的足够权限。这仅适用于具有用户和/或文件权限的操作系统。	40
SECURE_TRUSTED_CERTIFICATE_READ	安全网络层无法读取受信任证书文件。请检查以下内容： 1) 正确指定了证书文件名。 2) 证书文件存在。 3) 证书文件包含一个或多个证书。 4) 证书文件包含正确的证书。 5) 试图打开证书文件的程序具有查看该文件的足够权限。这仅适用于具有用户和/或文件权限的操作系统。	41
SEED_RANDOM	安全网络层无法为随机数生成器提供种子。请释放系统资源，重新连接后再尝试该操作。	16
SERVER_ERROR	服务器报告了一个错误。请联系 MobiLink 管理员以了解更多信息。	228

成员名称	说明	值
SHUTTING_DOWN	MobiLink 服务器关闭时遇到网络层的错误。可能是关闭时一些待执行的网络操作受到影响。	18
SOCKET_BIND	网络层无法将套接字绑定到给定端口。请检查以下内容： 1)（仅服务器）确认端口未被占用。如果端口被占用，请关闭监听该端口的应用程序或者指定另一个端口。 2)（仅服务器）确认所使用的端口不存在防火墙限制。 3)（仅客户端）如果使用 client_port 选项，确认给定端口未被占用。如果仅指定一个客户端端口，请考虑使用范围（例如 NNN-NNN）。如果该范围已占用，请考虑拓宽该范围或使用不同的范围。 4)（仅客户端）如果使用了 client_port 选项，请确认所使用的端口不存在防火墙限制。	60
SOCKET_CLEANUP	网络层无法清理套接字层。该错误仅在所有连接都完成后才发生，所以不会影响任何当前的连接。	61
SOCKET_CLOSE	网络层无法关闭套接字。因为未刷新待执行的写操作，网络会话可能会（也可能不会）提前终止。请检查以下内容： 1) 网络连接的另外一方发生任何错误。 2) 网络连接的另外一方正常运行。 3) 计算机仍与网络连接且网络可以作出响应。	62

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
SOCKET_CONNECT	网络层无法连接套接字。请检查以下内容： 1) 计算机与网络连接。 2) 套接字层正确初始化。 3) 指定了正确的主机和端口。 4) 主机服务器运行正常并监听正确的端口。 5) 主机监听的套接字类型正确（TCP/IP 与 UDP）。 6) 如果使用 client_port 选项，确认所使用的端口不存在防火墙限制。 7) 如果设备对开放套接字的数目有限制，确认未达到该限制。 8) 有足够的系统资源可用。如果运行缓慢，请释放系统资源。	63
SOCKET_CREATE_TCPIP	网络层无法创建 TCP/IP 套接字。请检查以下内容： 1) 计算机与网络连接。 2) 套接字层正确初始化。 5) 如果设备对开放套接字的数目有限制，确认未达到该限制。 6) 有足够的系统资源可用。如果运行缓慢，请释放系统资源。	58

成员名称	说明	值
SOCKET_CREATE_UDP	网络层无法创建 UDP 套接字。请检查以下内容： 1) 计算机与网络连接。 2) 套接字层正确初始化。 3) 如果使用 client_port 选项，确认给定端口未被占用。如果仅指定一个客户端端口，请考虑使用范围（例如 NNN-NNN）。如果该范围已占用，请考虑拓宽该范围或使用不同的范围。 4) 如果使用 client_port 选项，确认所使用的端口不存在防火墙限制。 5) 如果设备对开放套接字的数目有限制，确认未达到该限制。 6) 有足够的系统资源可用。如果运行缓慢，请释放系统资源。	59
SOCKET_GET_HOST_BY_ADDR	网络层无法使用主机的 IP 地址获取主机名。发布时，该错误虽然定义但并没有使用。	72
SOCKET_GET_NAME	网络层无法确定套接字的本地名称。在一个 TCP/IP 连接中，连接的每一端都有专门附加到端口的套接字。套接字的本地名称包含连接时网络指定的端口号。请检查以下内容： 1) 计算机仍与网络连接且网络可以作出响应。 2) 网络连接的另外一方正常运行。 3) 有足够的系统资源可用。如果运行缓慢，请释放系统资源。	64
SOCKET_GET_OPTION	网络层无法获取套接字选项。该错误可能是连接丢失时出现的第一个提示。请检查以下内容： 1) 计算机仍与网络连接且网络可以作出响应。 2) 网络连接的另外一方正常运行。 3) 有足够的系统资源可用。如果运行缓慢，请释放系统资源。	65

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
SOCKET_HOST_NAME_NOT_FOUND	无法找到给定的主机名。请检查以下内容： 1) 正确指定了主机名。 2) 主机可以访问。许多系统都包含一个 [ping] 实用程序，可使用该实用程序验证指定主机是否可以访问。 3) 域名服务器 (DNS)，或与其具有相同功能的实体是否可以使用。如果 DNS 不可用，请尝试指定主机的 IP 号（例如 NNN.NNN.NNN.NNN）而不是指定主机名。 4) 文件 HOSTS 包含将主机名映射到 IP 号的条目。	57
SOCKET_LISTEN	服务器无法监听套接字。积压请求指的是给定时间里未执行的排队连接请求的最大数。请检查以下内容： 1) 计算机仍与网络连接且网络可以作出响应。 2) 不存在可能阻止套接字监听器在当前计算机上运行的防火墙或其它限制。 3) 积压请求设置位于计算机上的限制（如果有）之内。 4) 有足够的系统资源可用。如果运行缓慢，请释放系统资源。	67
SOCKET_LIVENESS_OUT_OF_RANGE	指定的活动超时值无效。活动超时值必须是介于 0 到 65535 之间的一个整数。	200
SOCKET_LOCALHOST_NAME_NOT_FOUND	网络层无法确定 "本地主机" 的 IP 地址。请检查以下内容： 1) 域名服务器 (DNS)，或与其具有相同功能的实体是否可以使用。如果 DNS 不能使用，请尝试显式指定本地主机的 IP 地址（通常是 127.0.0.1）。 2) HOSTS 文件包含将 [本地主机 (localhost)] 名映射到 IP 地址的条目。 3) 有足够的系统资源可用。如果运行缓慢，请释放系统资源。	71
SOCKET_PORT_OUT_OF_RANGE	指定了无效端口号。端口号必须是介于 0 到 65535 之间的一个整数。	74

成员名称	说明	值
SOCKET_SELECT	网络层在试图等待套接字准备读或写时遇到错误。请检查以下内容： 1) 计算机与网络连接且网络可以作出响应。 2) 网络连接的另外一方正常运行。 3) 有足够的系统资源可用。如果运行缓慢，请释放系统资源。	69
SOCKET_SET_OPTION	网络层无法设置套接字选项。该错误可能是连接丢失时出现的第一个提示。请检查以下内容： 1) 计算机仍与网络连接且网络可以作出响应。 2) 网络连接的另外一方正常运行。 3) 有足够的系统资源可用。如果运行缓慢，请释放系统资源。	66
SOCKET_SHUTDOWN	网络层无法关闭套接字。请检查以下内容： 1) 计算机与网络连接且网络可以作出响应。 2) 网络连接的另外一方正常运行。 3) 有足够的系统资源可用。如果运行缓慢，请释放系统资源。	68
SOCKET_STARTUP	网络层无法初始化套接字层。请检查以下内容： 1) 正确安装了套接字层。必须存在正确的网络接口库并且可以访问。 2) 计算机与网络连接且网络可以作出响应。 3) 有足够的系统资源可用。如果运行缓慢，请释放系统资源。	70
UNEXPECTED_HTTP_REQUEST_TYPE	此次给定的 HTTP 请求类型是意外的请求类型。最有可能的原因是 HTTP 客户端不是 MobiLink 客户端。	234
UNRECOGNIZED_TLS_TYPE	TLS 类型无效。有关有效类型的信息，请查阅相关文档。	240
VALUE_OUT_OF_RANGE	指定的值不在该参数的可接受值的范围之内。请检查文档中的参数以了解可接受的值的范围。	233
WOULD_BLOCK	所请求的操作可能在不希望或者意想不到出现阻塞的地方导致阻塞。	13

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明	值
WRITE	无法将给定的字节数写入到网络层。请注意，写操作可能作为大型网络操作的一部分出现。例如，一些网络层具有子层，而子层可以执行多个读和写操作，这些操作又是上层基本操作的一部分。之所以出现写错误，通常有以下原因： 1) 网络存在导致写失败的问题。重新连接后再尝试该操作。 2) 连接超时。重新连接后再尝试该操作。 3) 连接的另外一方完全终止连接。请查阅客户端和/或服务器的错误日志，它们将指明连接断开的原因。请查阅输出日志错误并确定其原因，然后重新尝试该操作。 4) 位于连接另外一端的进程被异常中断。请查阅客户端和/或服务器的错误消息日志，它们将指明进程异常中断的原因。如果进程是通过非正常方法关闭，它的消息日志中就可能不会记录任何错误。重新连接后再尝试该操作。 5) 系统资源不足，无法执行写操作。请释放系统资源，重新连接后再尝试该操作。如果随后重新尝试的努力失败，请向网络管理员进行咨询。	9
WRITE_TIMEOUT	无法在给定时间内将给定的字节数写入到网络层。请检查网络是否工作正常、接收应用程序是否仍在运行。	202

另请参见

- “StreamErrorCode 属性”一节第 508 页

ULStreamType 枚举

UL Ext.: 枚举要用于同步的 MobiLink 同步流类型。

语法

Visual Basic
Public Enum **ULStreamType**

C#
public enum **ULStreamType**

注释

有关配置特定流类型的信息，请参见“UltraLite 同步流的网络协议选项”一节 《UltraLite - 数据库管理和参考》。

需要单独授予许可的组成部分
ECC 加密和 FIPS 认证的加密需要单独的许可。所有高度加密技术受出口法规约束。

请参见“单独授权的组件”一节 《SQL Anywhere 11 - 简介》。

成员

成员名称	说明	值
HTTP	通过 HTTP 进行同步。 HTTP 流使用 TCP/IP 作为其基础传输协议。UltraLite 应用程序充当 Web 浏览器，MobiLink 服务器充当 Web 服务器。UltraLite 应用程序发送 POST 请求以便将数据发送到服务器，发送 GET 请求以便从服务器读取数据。	1
HTTPS	通过 HTTPS（带有传送层安全性的 HTTP）进行同步。	2
TCPIP	通过 TCP/IP 进行同步。	0
TLS	通过具有传送层安全性的 TCP/IP 进行同步。	3

通过 HTTPS 和 TLS 流使用加密同步时，必须将相应的 DLL 文件部署到设备。下表列出了需要为各类加密协议部署的 DLL 文件。DLL 文件位于您的 SQL Anywhere 安装件的 *UltraLite\CE* 目录中。

协议	DLL
ECC	*mlcecc10.dll*
RSA	*mlcrsa10.dll*

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

协议	DLL
FIPS	*mlcrsafips10.dll*

另请参见

- “Stream 属性”一节第 487 页

ULSyncParms 类

UL Ext.: 表示同步参数，这些参数用于定义如何同步 UltraLite 数据库。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULSyncParms**

C#
public sealed class **ULSyncParms**

注释

没有用于此类的构造函数。每个连接都有自己的 ULSyncParms 实例，作为其 ULConnection.SyncParms 附属于连接。

一次最多只能指定一个同步命令（ULSyncParms.DownloadOnly、ULSyncParms.PingOnly、ULSyncParms.ResumePartialDownload 或 ULSyncParms.UploadOnly）。如果这些参数中有多个被设置为 true，则 ULConnection.Synchronize() 将抛出 ULSQLCode.SQLE_SYNC_INFO_INVALID SQLException。

ULSQLCode.SQLE_SYNC_INFO_INVALID 错误的其它原因包括未指定 ULSyncParms.Stream 值或 ULSyncParms.Version 值。

另请参见

- “ULSyncParms 成员”一节第 478 页
- “ULConnection 类”一节第 131 页
- “SyncParms 属性”一节第 143 页
- “Synchronize() 方法”一节第 171 页
- “DownloadOnly 属性”一节第 481 页
- “PingOnly 属性”一节第 484 页
- “ResumePartialDownload 属性”一节第 485 页
- “UploadOnly 属性”一节第 488 页
- “ULSQLCode 枚举”一节第 441 页
- “Synchronize() 方法”一节第 171 页
- “Stream 属性”一节第 487 页
- “Version 属性”一节第 490 页

ULSyncParms 成员

公共属性

成员名称	说明
“AdditionalParms 属性”一节第 480 页	采用以分号分隔的 "名称=值" 对的列表形式指定附加参数。这是一些较不常用的参数。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“AuthenticationParms 属性”一节第 480 页	为自定义用户验证脚本（MobiLink authenticate_parameters 连接事件）指定参数。
“DownloadOnly 属性”一节第 481 页	指定在同步时是禁用还是启用上载。
“KeepPartialDownload 属性”一节第 482 页	指定在同步时是禁用还是启用部分下载。
“NewPassword 属性”一节第 483 页	为用 UserName 指定的用户指定一个新的 MobiLink 口令。
“Password 属性”一节第 483 页	用 UserName 指定的用户的 MobiLink 口令。
“PingOnly 属性”一节第 484 页	指定客户端是否应当只强制 MobiLink 服务器回应，而不实际执行同步。
“Publications 属性”一节第 485 页	指定要同步的发布。
“ResumePartialDownload 属性”一节第 485 页	指定是恢复还是放弃先前的部分下载。
“SendColumnNames 属性”一节第 486 页	指定客户端是否应在同步过程中向 MobiLink 服务器发送列名。
“SendDownloadAck 属性”一节第 487 页	指定客户端是否应在同步过程中向 MobiLink 服务器发送下载确认消息。下载确认会在下载在远程完全应用并提交后发送（正确认），或者是在下载失败后发送（负确认）。
“Stream 属性”一节第 487 页	指定用于同步的 MobiLink 同步流。
“StreamParms 属性”一节第 488 页	指定用于配置同步流的参数。
“UploadOnly 属性”一节第 488 页	指定在同步时是禁用还是启用下载。
“UserName 属性”一节第 489 页	向 MobiLink 服务器唯一标识 MobiLink 客户端的用户名。
“Version 属性”一节第 490 页	指定要使用哪一个同步脚本。

公共方法

成员名称	说明
“CopyFrom 方法”一节 第 490 页	将指定的 ULSyncParms 对象的属性复制到此 ULSyncParms 对象。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULConnection 类”一节第 131 页
- “SyncParms 属性”一节第 143 页
- “Synchronize() 方法”一节第 171 页
- “DownloadOnly 属性”一节第 481 页
- “PingOnly 属性”一节第 484 页
- “ResumePartialDownload 属性”一节第 485 页
- “UploadOnly 属性”一节第 488 页
- “ULSQLCode 枚举”一节第 441 页
- “Synchronize() 方法”一节第 171 页
- “Stream 属性”一节第 487 页
- “Version 属性”一节第 490 页

AdditionalParms 属性

指定 "关键字=值" 对的字符串。此 "关键字=值" 对的列表通常包含不经常使用的同步参数。

语法

Visual Basic
Public Property **AdditionalParms** As String

C#
public string **AdditionalParms** { get; set; }

属性值

以分号分隔的 "关键字=值" 列表形式的附加参数。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页

AuthenticationParms 属性

为自定义用户验证脚本（MobiLink authenticate_parameters 连接事件）指定参数。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Property **AuthenticationParms** As String()

C#
public string[] **AuthenticationParms** { get; set; }

属性值

字符串数组，每个字符串都包含一个验证参数（空数组项将导致同步错误）。缺省值为空值引用（在 Visual Basic 中为 Nothing），表示没有验证参数。

注释

只使用前 255 个字符串，且每个字符串的长度不应超过 128 个字符（长字符串在发送到 MobiLink 服务器时会被截断）。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页

DownloadOnly 属性

指定在同步时是禁用还是启用上载。

语法

Visual Basic
Public Property **DownloadOnly** As Boolean

C#
public bool **DownloadOnly** { get; set; }

属性值

如果同步时禁用上载，则为 true；如果启用上载，则为 false。缺省值为 false。

注释

一次最多只能指定一个同步命令（ULSyncParms.DownloadOnly、ULSyncParms.PingOnly、ULSyncParms.ResumePartialDownload 或 ULSyncParms.UploadOnly）。如果这些参数中有多个被设置为 true，则 ULConnection.Synchronize() 将抛出 ULSQLCode.SQLE_SYNC_INFO_INVALID SQLException。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页
- “UploadOnly 属性”一节第 488 页
- “DownloadOnly 属性”一节第 481 页
- “PingOnly 属性”一节第 484 页
- “ResumePartialDownload 属性”一节第 485 页
- “UploadOnly 属性”一节第 488 页
- “ULSQLCode 枚举”一节第 441 页
- “Synchronize() 方法”一节第 171 页

KeepPartialDownload 属性

指定在同步时是禁用还是启用部分下载。

语法

Visual Basic
Public Property **KeepPartialDownload** As Boolean

C#
public bool **KeepPartialDownload** { get; set; }

属性值

如果在同步时启用部分下载，则为 true；如果禁用部分下载，则为 false。缺省值为 false。

注释

UltraLite.NET 能够通过 ULSyncProgressListener 重新启动由于通信错误或用户中止而失败的下载。UltraLite.NET 在接收下载的过程中对其进行处理。如果下载被中断，则部分下载事务仍保留在数据库中，并且可在下一次同步过程中恢复。

要指示 UltraLite.NET 应保存部分下载，请指定 connection.SyncParms.KeepPartialDownload=true，否则当发生错误时将回退下载。

如果保留了部分下载，则当 connection.Synchronize() 退出时，输出字段 connection.SyncResult.ULSyncResult.PartialDownloadRetained 将设置为 true。

如果设置了 PartialDownloadRetained，则可以继续下载。为此，请在调用 connection.Synchronize() 时将 connection.SyncParms.ULSyncParms.ResumePartialDownload 设置为 true。当出现其它通信错误时，我们也建议您保持将 KeepPartialDownload 设置为 true。如果跳过某一下载，将不执行任何上载。

您在恢复下载期间接收的下载会与下载最初开始时一样旧。如果您需要最新数据，则可以在专门的恢复下载完成后再立即进行一次下载。

在恢复下载时，许多 ULSyncParms 字段是无关的字段。例如，Publications 字段就是不使用的字段。您会收到在最初下载时请求的发布。只需要设置字段 ResumePartialDownload 和 UserName。如果您愿意，也可以设置字段 KeepPartialDownload 和 DisableConcurrency，两个字段都会正常工作。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

如果不再需要现有的部分下载，则可以调用 ULConnection.RollbackPartialDownload() 回退失败的下载事务。此外，如果您尝试再次同步但没有指定 ResumePartialDownload，则下次同步开始之前将回退部分下载。

有关详细信息，请参见“恢复失败的下载”一节 《MobiLink - 服务器管理》。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页
- “PartialDownloadRetained 属性”一节第 507 页
- “ResumePartialDownload 属性”一节第 485 页
- “RollbackPartialDownload 方法”一节第 169 页
- “ResumePartialDownload 属性”一节第 485 页
- “UserName 属性”一节第 489 页
- “AdditionalParms 属性”一节第 480 页
- “RollbackPartialDownload 方法”一节第 169 页

NewPassword 属性

为用 UserName 指定的用户指定一个新的 MobiLink 口令。

语法

Visual Basic
Public Property **NewPassword** As String

C#
public string **NewPassword** { get; set; }

属性值

指定新 MobiLink 口令的字符串。缺省值为空值引用（在 Visual Basic 中是 Nothing），表示没有更改口令。

注释

新口令将在下次同步之后生效。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页
- “UserName 属性”一节第 489 页

Password 属性

用 UserName 指定的用户的 MobiLink 口令。

语法

Visual Basic
Public Property **Password** As String

C#
public string **Password** { get; set; }

属性值

用于指定 MobiLink 口令的字符串。缺省值为空值引用（在 Visual Basic 中为 Nothing），表示没有指定口令。

注释

MobiLink 用户名和口令不同于任何数据库用户 ID 和口令，它们用于向 MobiLink 服务器标识和验证应用程序。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页
- “NewPassword 属性”一节第 483 页
- “UserName 属性”一节第 489 页

PingOnly 属性

指定客户端是否应当只强制 MobiLink 服务器回应，而不实际执行同步。

语法

Visual Basic
Public Property **PingOnly** As Boolean

C#
public bool **PingOnly** { get; set; }

属性值

如果指定客户端应当只强制 MobiLink 服务器回应，则为 ture；如果指定客户端应当实际执行同步，则为 false。缺省值为 false。

注释

一次最多只能指定一个同步命令（ULSyncParms.DownloadOnly、ULSyncParms.PingOnly、ULSyncParms.ResumePartialDownload 或 ULSyncParms.UploadOnly）。如果这些参数中有多个被设置为 true，则 ULConnection.Synchronize() 将抛出 ULSQLCode.SQLE_SYNC_INFO_INVALID SQLException。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页
- “DownloadOnly 属性”一节第 481 页
- “PingOnly 属性”一节第 484 页
- “ResumePartialDownload 属性”一节第 485 页
- “UploadOnly 属性”一节第 488 页
- “ULSQLCode 枚举”一节第 441 页
- “Synchronize() 方法”一节第 171 页

Publications 属性

指定要同步的发布。

语法

Visual Basic
Public Property **Publications** As String

C#
public string **Publications** { get; set; }

属性值

包含发布名列表的字符串，以逗号 (,) 分隔；或特殊值 ULPublicationSchema.SYNC_ALL_PUBS 或特殊值 ULPublicationSchema.SYNC_ALL_DB。缺省值为 ULPublicationSchema.SYNC_ALL_DB。有关详细信息，请参见“ULPublicationSchema 类”一节第 391 页。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页
- “SYNC_ALL_PUBS 字段”一节第 392 页
- “SYNC_ALL_DB 字段”一节第 392 页
- “ULPublicationSchema 类”一节第 391 页

ResumePartialDownload 属性

指定是恢复还是放弃先前的部分下载。

语法

Visual Basic
Public Property **ResumePartialDownload** As Boolean

C#
public bool **ResumePartialDownload** { get; set; }

属性值

值为 true，则恢复先前的部分下载，值为 false，则放弃先前的部分下载。缺省值为 false。

注释

一次最多只能指定一个同步命令（ULSyncParms.DownloadOnly、ULSyncParms.PingOnly、ULSyncParms.ResumePartialDownload 或 ULSyncParms.UploadOnly）。如果这些参数中有多个被设置为 true，则 ULConnection.Synchronize() 将抛出 ULSQLCode.SQLE_SYNC_INFO_INVALID SQLException。

有关部分下载的详细信息，请参见“KeepPartialDownload 属性”一节第 482 页。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页
- “DownloadOnly 属性”一节第 481 页
- “PingOnly 属性”一节第 484 页
- “ResumePartialDownload 属性”一节第 485 页
- “UploadOnly 属性”一节第 488 页
- “ULSQLCode 枚举”一节第 441 页
- “Synchronize() 方法”一节第 171 页
- “PartialDownloadRetained 属性”一节第 507 页

SendColumnNames 属性

指定客户端是否应在同步过程中向 MobiLink 服务器发送列名。

语法

Visual Basic
Public Property **SendColumnNames** As Boolean

C#
public bool **SendColumnNames** { get; set; }

属性值

如果指定客户端应向 MobiLink 服务器发送列名，则为 true；如果指定不发送列名，则为 false。缺省值为 false。

注释

MobiLink 服务器使用列名进行直接行处理。当 MobiLink 服务器使用行处理 API 按名称而不按索引引用列时，应设置此选项。这是此选项所发送的列名的唯一用途。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

SendDownloadAck 属性

指定客户端是否应在同步过程中向 MobiLink 服务器发送下载确认消息。下载确认会在下载在远程完全应用并提交后发送（正确认），或者是在下载失败后发送（负确认）。

语法

Visual Basic
Public Property **SendDownloadAck** As Boolean

C#
public bool **SendDownloadAck** { get; set; }

属性值

设置为 True 指定客户端应将下载确认发送到 MobiLink 服务器。设置为 False 指定不发送下载确认。缺省值为 False。

注释

如果客户端发送下载确认，则 MobiLink 服务器数据库工作线程必须等待客户端应用并提交下载。如果客户端不发送下载确认，则可更快地释放 MobiLink 服务器以进行下一次同步。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页

Stream 属性

指定用于同步的 MobiLink 同步流。

语法

Visual Basic
Public Property **Stream** As ULStreamType

C#
public ULStreamType **Stream** { get; set; }

属性值

ULStreamType 值之一，用于指定要使用的同步流类型。缺省值为 ULStreamType.TCPIP。

注释

大多数同步流都需要用一些参数来标识 MobiLink 服务器地址和控制其它行为。这些参数由 ULSyncParms.StreamParms 提供。

如果设置的流类型值在平台上无效，则流类型将被设置为 ULStreamType.TCPIP。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页
- “ULStreamType 枚举”一节第 476 页
- “StreamParms 属性”一节第 488 页
- “ULStreamType 枚举”一节第 476 页

StreamParms 属性

指定用于配置同步流的参数。

语法

Visual Basic
Public Property **StreamParms** As String

C#
public string **StreamParms** { get; set; }

属性值

用于指定流参数的字符串，以分号分隔的 "关键字-值" 对的列表形式表示。缺省值为空值引用（在 Visual Basic 中是 Nothing）。

注释

有关配置特定流类型的信息，请参见。

“UltraLite 同步流的网络协议选项”一节 《UltraLite - 数据库管理和参考》.

StreamParms 是一个包含用于同步流的所有参数的字符串。参数是以分号分隔的 "名称=值" 对列表 ("param1=value1;param2=value2") 的形式指定的。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页
- “Stream 属性”一节第 487 页
- “ULStreamType 枚举”一节第 476 页

UploadOnly 属性

指定在同步时是禁用还是启用下载。

语法

Visual Basic
Public Property **UploadOnly** As Boolean

C#
public bool **UploadOnly** { get; set; }

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

属性值

如果禁用下载，则为 true；如果启用下载，则为 false。缺省值为 false。

注释

一次最多只能指定一个同步命令（ULSyncParms.DownloadOnly、ULSyncParms.PingOnly、ULSyncParms.ResumePartialDownload 或 ULSyncParms.UploadOnly）。如果这些参数中有多个被设置为 true，则 ULConnection.Synchronize() 将抛出 ULSQLCode.SQLE_SYNC_INFO_INVALID SQLException。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页
- “DownloadOnly 属性”一节第 481 页
- “PingOnly 属性”一节第 484 页
- “ResumePartialDownload 属性”一节第 485 页
- “UploadOnly 属性”一节第 488 页
- “ULSQLCode 枚举”一节第 441 页
- “Synchronize() 方法”一节第 171 页

UserName 属性

向 MobiLink 服务器唯一标识 MobiLink 客户端的用户名。

语法

Visual Basic

```
Public Property UserName As String
```

C#

```
public string  UserName { get; set; }
```

属性值

用于指定用户名的字符串。此参数没有缺省值，因而必须显式设置。

注释

MobiLink 服务器使用此值确定下载内容、记录同步状态并在同步期间发生中断时进行恢复。此用户名和口令独立于任何数据库用户 ID 和口令，它们用于向 MobiLink 服务器标识和验证应用程序。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页
- “Password 属性”一节第 483 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

Version 属性

指定要使用哪一个同步脚本。

语法

Visual Basic
Public Property **Version** As String

C#
public string **Version** { get; set; }

属性值

用于指定要使用的同步脚本版本的字符串。此参数没有缺省值，因而必须显式设置。

注释

统一数据库中的每个同步脚本都带有版本字符串标记。例如，可能有两个不同的 download_cursor 脚本，分别用不同的版本字符串来标识。版本字符串使 UltraLite 应用程序可以从一组同步脚本中进行选择。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页

CopyFrom 方法

将指定的 ULSyncParms 对象的属性复制到此 ULSyncParms 对象。

语法

Visual Basic
```
Public Sub CopyFrom( _
  ByVal src As ULSyncParms _
)
```

C#
```
public void CopyFrom(
  ULSyncParms src
);
```

参数

- **src** 要从中复制的对象。

另请参见

- “ULSyncParms 类”一节第 478 页
- “ULSyncParms 成员”一节第 478 页
- “ULSyncParms 类”一节第 478 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULSyncProgressData 类

UL Ext.: 返回同步进度监控数据。

语法

Visual Basic
Public Class **ULSyncProgressData**

C#
public class **ULSyncProgressData**

另请参见

- “ULSyncProgressData 成员”一节第 491 页
- “ULSyncProgressListener 接口”一节第 501 页

ULSyncProgressData 成员

公共字段

成员名称	说明
“FLAG_IS_BLOCKING 字段”一节第 492 页	一个标志，指示同步受到阻塞，正在等待来自 MobiLink 服务器的响应。此字段为常量并是只读字段。

公共属性

成员名称	说明
“Flags 属性”一节第 493 页	返回指示有关当前状态的其它信息的当前同步标志。
“ReceivedBytes 属性”一节第 493 页	返回到目前为止已接收的字节数。所有状态下都将更新此信息。
“ReceivedDeletes 属性”一节第 494 页	返回到目前为止已接收的删除行的数量。
“ReceivedInserts 属性”一节第 494 页	返回到目前为止已接收的插入行的数量。
“ReceivedUpdates 属性”一节第 494 页	返回到目前为止已接收的更新行的数量。
“SentBytes 属性”一节第 495 页	返回到目前为止已发送的字节数。所有状态下都将更新此信息。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“SentDeletes 属性”一节 第 495 页	返回到目前为止已发送的删除行的数量。
“SentInserts 属性”一节 第 496 页	返回到目前为止已发送的插入行的数量。
“SentUpdates 属性”一节 第 496 页	返回到目前为止已发送的更新行的数量。
“State 属性”一节第 497 页	返回当前的同步状态。
“SyncTableCount 属性”一节 第 497 页	返回正在同步的表的数目。
“SyncTableIndex 属性”一节 第 498 页	返回当前正在同步的表的索引（表按照从 1 到 DatabaseSchema.TableCount 的顺序编号）。
“TableID 属性”一节 第 498 页	返回当前正在同步的表的数据库索引。
“TableName 属性”一节 第 499 页	返回正在上载或下载的当前表的名称。

公共方法

成员名称	说明
“SetValues 方法”一节 第 499 页	将当前同步值装入此对象

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressListener 接口”一节第 501 页

FLAG_IS_BLOCKING 字段

一个标志，指示同步受到阻塞，正在等待来自 MobiLink 服务器的响应。此字段为常量并是只读字段。

语法

Visual Basic
Public Shared **FLAG_IS_BLOCKING** As Integer

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

C#
public const int **FLAG_IS_BLOCKING**;

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页

Flags 属性

返回指示有关当前状态的其它信息的当前同步标志。

语法

Visual Basic
Public Readonly Property **Flags** As Integer

C#
public int **Flags** { get;}

属性值

一个整数，其中包含由 OR 组合在一起的标志。

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页
- “FLAG_IS_BLOCKING 字段”一节第 492 页

ReceivedBytes 属性

返回到目前为止已接收的字节数。所有状态下都将更新此信息。

语法

Visual Basic
Public Readonly Property **ReceivedBytes** As Long

C#
public long **ReceivedBytes** { get;}

属性值

到目前为止已接收的字节数。

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页
- “ULSyncProgressState 枚举”一节第 503 页
- “ULSyncProgressState 枚举”一节第 503 页

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ReceivedDeletes 属性

返回到目前为止已接收的删除行的数量。

语法

Visual Basic
Public Readonly Property **ReceivedDeletes** As Integer

C#
public int **ReceivedDeletes** { get;}

属性值

到目前为止已接收的删除行的数量。

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页
- “ULSyncProgressState 枚举”一节第 503 页
- “ULSyncProgressState 枚举”一节第 503 页

ReceivedInserts 属性

返回到目前为止已接收的插入行的数量。

语法

Visual Basic
Public Readonly Property **ReceivedInserts** As Integer

C#
public int **ReceivedInserts** { get;}

属性值

到目前为止已接收的插入行的数量。

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页
- “ULSyncProgressState 枚举”一节第 503 页
- “ULSyncProgressState 枚举”一节第 503 页

ReceivedUpdates 属性

返回到目前为止已接收的更新行的数量。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Readonly Property **ReceivedUpdates** As Integer

C#
public int **ReceivedUpdates** { get;}

属性值

到目前为止已接收的更新行的数量。

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页
- “ULSyncProgressState 枚举”一节第 503 页
- “ULSyncProgressState 枚举”一节第 503 页

SentBytes 属性

返回到目前为止已发送的字节数。所有状态下都将更新此信息。

语法

Visual Basic
Public Readonly Property **SentBytes** As Long

C#
public long **SentBytes** { get;}

属性值

到目前为止已发送的字节数。

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页
- “ULSyncProgressState 枚举”一节第 503 页
- “ULSyncProgressState 枚举”一节第 503 页

SentDeletes 属性

返回到目前为止已发送的删除行的数量。

语法

Visual Basic
Public Readonly Property **SentDeletes** As Integer

C#
public int **SentDeletes** { get;}

属性值

到目前为止已发送的删除行的数量。

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页
- “ULSyncProgressState 枚举”一节第 503 页
- “ULSyncProgressState 枚举”一节第 503 页

SentInserts 属性

返回到目前为止已发送的插入行的数量。

语法

Visual Basic
Public Readonly Property **SentInserts** As Integer

C#
public int **SentInserts** { get;}

属性值

到目前为止已发送的插入行的数量。

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页
- “ULSyncProgressState 枚举”一节第 503 页
- “ULSyncProgressState 枚举”一节第 503 页

SentUpdates 属性

返回到目前为止已发送的更新行的数量。

语法

Visual Basic
Public Readonly Property **SentUpdates** As Integer

C#
public int **SentUpdates** { get;}

属性值

到目前为止已发送的更新行的数量。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页
- “ULSyncProgressState 枚举”一节第 503 页
- “ULSyncProgressState 枚举”一节第 503 页

State 属性

返回当前的同步状态。

语法

Visual Basic
Public Readonly Property **State** As ULSyncProgressState

C#
public ULSyncProgressState **State** { get;}

属性值

指定当前同步状态的 ULSyncProgressState 值之一。

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页
- “ULSyncProgressState 枚举”一节第 503 页

SyncTableCount 属性

返回正在同步的表的数目。

语法

Visual Basic
Public Readonly Property **SyncTableCount** As Integer

C#
public int **SyncTableCount** { get;}

属性值

正在同步的表的数目。每个表都有一个发送和接收阶段，因此这个数字可能会大于正在同步的表数。

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页
- “ULSyncProgressState 枚举”一节第 503 页
- “ULSyncProgressState 枚举”一节第 503 页

SyncTableIndex 属性

返回当前正在同步的表的索引（表按照从 1 到 DatabaseSchema.TableCount 的顺序编号）。

语法

Visual Basic
Public Readonly Property **SyncTableIndex** As Integer

C#
public int **SyncTableIndex** { get;}

属性值

当前正在同步的表的索引，范围从 1 到 SyncTableCount

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页
- “ULSyncProgressState 枚举”一节第 503 页
- “ULSyncProgressState 枚举”一节第 503 页

TableID 属性

返回当前正在同步的表的数据库索引。

语法

Visual Basic
Public Readonly Property **TableID** As Integer

C#
public int **TableID** { get;}

属性值

数据库索引，范围从 1 到 ULDatabaseSchema.TableCount

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页
- “ULSyncProgressState 枚举”一节第 503 页
- “ULSyncProgressState 枚举”一节第 503 页
- “TableCount 属性”一节第 250 页

TableName 属性

返回当前正在上载或下载的表的名称。

语法

Visual Basic
Public Property **TableName** As String

C#
public string **TableName** { get; set; }

属性值

正在上载或下载的表的名称。如果不适用，则为空值。

SetValues 方法

将当前同步值装入此对象

语法

Visual Basic
```
Public Sub SetValues( _
  ByVal sync_state As ULSyncProgressState, _
  ByVal table_id As Integer, _
  ByVal table_count As Integer, _
  ByVal table_index As Integer, _
  ByVal sent_bytes As Long, _
  ByVal sent_inserts As Integer, _
  ByVal sent_updates As Integer, _
  ByVal sent_deletes As Integer, _
  ByVal recv_bytes As Long, _
  ByVal recv_inserts As Integer, _
  ByVal recv_updates As Integer, _
  ByVal recv_deletes As Integer, _
  ByVal flag_s As Integer _
)
```

C#
```
public void SetValues(
  ULSyncProgressState sync_state,
  int table_id,
  int table_count,
```

```
    int table_index,
    long sent_bytes,
    int sent_inserts,
    int sent_updates,
    int sent_deletes,
    long recv_bytes,
    int recv_inserts,
    int recv_updates,
    int recv_deletes,
    int flag_s
);
```

另请参见

- “ULSyncProgressData 类”一节第 491 页
- “ULSyncProgressData 成员”一节第 491 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULSyncProgressListener 接口

UL Ext.: 用于接收同步进度事件的监听器接口。

语法

Visual Basic
Public Interface **ULSyncProgressListener**

C#
public interface **ULSyncProgressListener**

另请参见

- “ULSyncProgressListener 成员”一节第 501 页
- “Synchronize(ULSyncProgressListener) 方法”一节第 172 页

ULSyncProgressListener 成员

公共方法

成员名称	说明
“SyncProgressed 方法”一节第 501 页	同步过程中将调用它来向用户通知进度。此方法应返回 true 取消同步或 false 继续同步。

另请参见

- “ULSyncProgressListener 接口”一节第 501 页
- “Synchronize(ULSyncProgressListener) 方法”一节第 172 页

SyncProgressed 方法

同步过程中将调用它来向用户通知进度。此方法应返回 true 取消同步或 false 继续同步。

语法

Visual Basic
```
Public Function SyncProgressed( _
  ByVal data As ULSyncProgressData _
) As Boolean
```

C#
```
public bool SyncProgressed(
  ULSyncProgressData data
);
```

参数

- **data** 包含最新同步进度数据的 ULSyncProgressData 对象。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

返回值

此方法应返回 true 取消同步或 false 继续同步。

注释

在调用 SyncProgressed 的过程中，不应该调用任何 UltraLite.NET API 方法。

另请参见

- “ULSyncProgressListener 接口”一节第 501 页
- “ULSyncProgressListener 成员”一节第 501 页
- “ULSyncProgressData 类”一节第 491 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULSyncProgressState 枚举

UL Ext.: 枚举同步过程中可能发生的所有状态。

语法

Visual Basic
Public Enum **ULSyncProgressState**

C#
public enum **ULSyncProgressState**

成员

成员名称	说明	值
STATE_canceled	已取消同步。	15
STATE_COMMITTING_DOWNLOAD	正在提交下载。所接收行数的最终统计将包含在此事件中。	9
STATE_CONNECTING	已构建同步流，但尚未打开。	1
STATE_DISCONNECTING	同步流即将关闭。	11
STATE_DONE	同步已成功完成。	12
STATE_ERROR	同步已完成但发生了错误。	13
STATE_FINISHING_UPLOAD	即将完成上载。所发送行数的最终统计将包含在此事件中。	5
STATE_RECEIVING_DATA	正在接收当前表的数据。已更新 ULSyncProgressData.ReceivedBytes、ULSyncProgressData.ReceivedInserts、ULSyncProgressData.ReceivedUpdates 和 ULSyncProgressData.ReceivedDeletes。	8
STATE_RECEIVING_TABLE	正在接收表。使用 ULSyncProgressData.SyncTableIndex 和 ULSyncProgressData.SyncTableCount 可以监控进度。	7
STATE_RECEIVING_UPLOAD_ACK	正在接收关于上载已完成的确认。	6

成员名称	说明	值
STATE_ROLLING_BACK_DOWNLOAD	同步正在回退下载，因为在下载过程中遇到错误。随后将用一个 STATE_ERROR 进度报告来报告该错误。	14
STATE_SENDING_DATA	正在发送当前表的数据。已更新 ULSyncProgressData.SentBytes、ULSyncProgressData.SentInserts、ULSyncProgressData.SentUpdates 和 ULSyncProgressData.SentDeletes。	4
STATE_SENDING_DOWNLOAD_ACK	正在发送关于下载已完成的确认。	10
STATE_SENDING_HEADER	已打开同步流，即将发送标头。	2
STATE_SENDING_TABLE	正在发送表。使用 ULSyncProgressData.SyncTableIndex 和 ULSyncProgressData.SyncTableCount 可以监控进度。	3
STATE_STARTING	尚未采取同步操作。	0

另请参见

- “ULSyncProgressData 类”一节第 491 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULSyncResult 类

UL Ext.: 表示上一次同步的状态。

语法

Visual Basic
Public Class **ULSyncResult**

C#
public class **ULSyncResult**

注释

没有用于此类的构造函数。每个连接都有其自己的 ULSyncResult 实例，该实例作为连接的 ULConnection.SyncResult 附加在该连接上。只有在连接处于打开状态时 ULSyncResult 实例才有效。

另请参见

- “ULSyncResult 成员”一节第 505 页
- “SyncResult 属性”一节第 144 页
- “Synchronize() 方法”一节第 171 页

ULSyncResult 成员

公共属性

成员名称	说明
“AuthStatus 属性”一节第 506 页	返回上次同步尝试的授权状态码。
“AuthValue 属性”一节第 506 页	返回来自自定义用户验证同步脚本的返回值。
“IgnoredRows 属性”一节第 507 页	检查在上次同步过程中是否忽略了任何上载行。
“PartialDownloadRetained 属性”一节第 507 页	检查在上次同步过程中是否保留了部分下载。
“StreamErrorCode 属性”一节第 508 页	返回流本身报告的错误。
“StreamErrorParameters 属性”一节第 508 页	返回以逗号分隔的流错误参数列表。
“StreamErrorSystem 属性”一节第 509 页	返回流错误的系统特定代码。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“Timestamp 属性”一节 第 509 页	返回上一次同步的时间戳。
“UploadOK 属性”一节 第 510 页	检查上次上载同步是否成功。

另请参见

- “ULSyncResult 类”一节第 505 页
- “SyncResult 属性”一节第 144 页
- “Synchronize() 方法”一节第 171 页

AuthStatus 属性

返回上次同步尝试的授权状态码。

语法

Visual Basic
Public Readonly Property **AuthStatus** As ULAuthStatusCode

C#
public ULAuthStatusCode **AuthStatus** { get;}

属性值

表示上次同步尝试的授权状态的 ULAuthStatusCode 值之一。

另请参见

- “ULSyncResult 类”一节第 505 页
- “ULSyncResult 成员”一节第 505 页
- “ULAuthStatusCode 枚举”一节第 56 页

AuthValue 属性

返回来自自定义用户验证同步脚本的返回值。

语法

Visual Basic
Public Readonly Property **AuthValue** As Long

C#
public long **AuthValue** { get;}

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

属性值

从自定义用户验证同步脚本返回的长整数。

另请参见

- “ULSyncResult 类”一节第 505 页
- “ULSyncResult 成员”一节第 505 页

IgnoredRows 属性

检查在上次同步过程中是否忽略了任何上载行。

语法

Visual Basic
Public Readonly Property **IgnoredRows** As Boolean

C#
public bool **IgnoredRows** { get;}

属性值

如果上次同步过程中忽略了任何已上载的行，则返回 true；如果未忽略行，则返回 false。

另请参见

- “ULSyncResult 类”一节第 505 页
- “ULSyncResult 成员”一节第 505 页
- “DownloadOnly 属性”一节第 481 页

PartialDownloadRetained 属性

检查在上次同步过程中是否保留了部分下载。

语法

Visual Basic
Public Readonly Property **PartialDownloadRetained** As Boolean

C#
public bool **PartialDownloadRetained** { get;}

属性值

如果下载中断并且保留了部分下载，则为 true；如果下载没有中断或者如果回退了部分下载，则为 false。

另请参见

- “ULSyncResult 类”一节第 505 页
- “ULSyncResult 成员”一节第 505 页
- “KeepPartialDownload 属性”一节第 482 页

StreamErrorCode 属性

返回流本身报告的错误。

语法

Visual Basic
Public Readonly Property **StreamErrorCode** As ULStreamErrorCode

C#
public ULStreamErrorCode **StreamErrorCode** { get;}

属性值

表示流本身报告的错误的 ULStreamErrorCode 值之一；如果没有发生错误，则为 ULStreamErrorCode.NONE。

另请参见

- “ULSyncResult 类”一节第 505 页
- “ULSyncResult 成员”一节第 505 页
- “ULStreamErrorCode 枚举”一节第 458 页
- “ULStreamErrorCode 枚举”一节第 458 页

StreamErrorParameters 属性

返回以逗号分隔的流错误参数列表。

语法

Visual Basic
Public Readonly Property **StreamErrorParameters** As String

C#
public string **StreamErrorParameters** { get;}

属性值

包含“StreamErrorCode 属性”一节第 508 页中报告的流错误参数代码对应的流错误参数列表（以逗号分隔）。如果错误没有参数或未设置任何错误，这将是空字符串。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULSyncResult 类”一节第 505 页
- “ULSyncResult 成员”一节第 505 页
- “ULStreamErrorCode 枚举”一节第 458 页

StreamErrorSystem 属性

返回流错误的系统特定代码。

语法

Visual Basic
Public Readonly Property **StreamErrorSystem** As Integer

C#
public int **StreamErrorSystem** { get;}

属性值

一个整数，用于表示流错误的系统特定代码。

另请参见

- “ULSyncResult 类”一节第 505 页
- “ULSyncResult 成员”一节第 505 页

Timestamp 属性

返回上一次同步的时间戳。

语法

Visual Basic
Public Readonly Property **Timestamp** As Date

C#
public DateTime **Timestamp** { get;}

属性值

指定上一次同步的时间戳的 System.DateTime。

另请参见

- “ULSyncResult 类”一节第 505 页
- “ULSyncResult 成员”一节第 505 页
- DateTime

UploadOK 属性

检查上次上载同步是否成功。

语法

Visual Basic
Public Readonly Property **UploadOK** As Boolean

C#
public bool **UploadOK** { get;}

属性值

如果上次上载同步成功，则为 true；如果上次上载同步不成功，则为 false。

另请参见

- “ULSyncResult 类”一节第 505 页
- “ULSyncResult 成员”一节第 505 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULTable 类

UL Ext.: 表示 UltraLite 数据库中的表。

语法

Visual Basic
Public Class **ULTable**
  Inherits ULResultSet

C#
public class **ULTable**: ULResultSet

注释

没有用于此类的构造函数。表是使用 ULCommand 的 ULCommand.ExecuteTable() 创建的。

Inherits: ULResultSet

Implements: System.Data.IDataReader、System.Data.IDataRecord、System.IDisposable

另请参见

- “ULTable 成员”一节第 511 页
- “ExecuteTable() 方法”一节第 118 页
- “ULCommand 类”一节第 86 页
- “ULResultSet 类”一节第 394 页
- IDataReader
- IDataRecord
- IDisposable

ULTable 成员

公共属性

成员名称	说明
“Depth 属性”一节第 262 页（继承自 ULDataReader）	返回当前行的嵌套深度。最外层的表深度为 0。
“FieldCount 属性”一节第 262 页（继承自 ULDataReader）	返回游标中的列数。
“HasRows 属性”一节第 263 页（继承自 ULDataReader）	检查 ULDataReader 包含一行还是多行。
“IsBOF 属性”一节第 263 页（继承自 ULDataReader）	**UL Ext.:** 检查当前行位置是否在第一行之前。

成员名称	说明
“IsClosed 属性”一节第 264 页（继承自 ULDataReader）	检查游标当前是否处于打开状态。
“IsEOF 属性”一节第 264 页（继承自 ULDataReader）	**UL Ext.:** 检查当前行位置是否在最后一行之后。
“Item 属性”一节第 264 页（继承自 ULDataReader）	以 Object 实例的形式获取指定列的值。
“RecordsAffected 属性”一节第 266 页（继承自 ULDataReader）	返回通过执行 SQL 语句所更改、插入或删除的行数。对于 SELECT 语句或 CommandType.TableDirect 表，此值为 -1。
“RowCount 属性”一节第 267 页（继承自 ULDataReader）	**UL Ext.:** 返回游标中的行数。
“Schema 属性”一节第 518 页	保存表模式。此属性仅在其连接打开期间才有效。
VisibleFieldCount（继承自 DbDataReader）	获取 DbDataReader 中未隐藏的字段数。

公共方法

成员名称	说明
“AppendBytes 方法”一节第 400 页（继承自 ULResultSet）	将指定的 System.Bytes 数组的指定子集附加到指定的 ULDbType.LongBinary 列的新值中。
“AppendChars 方法”一节第 402 页（继承自 ULResultSet）	将指定的 System.Chars 数组的指定子集附加到指定的 ULDbType.LongVarchar 列的新值。
“Close 方法”一节第 268 页（继承自 ULDataReader）	关闭游标。
“Delete 方法”一节第 403 页（继承自 ULResultSet）	删除当前行。
“DeleteAllRows 方法”一节第 519 页	删除表中的所有行。
Dispose（继承自 DbDataReader）	释放 DbDataReader 的当前实例使用的所有资源。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“FindBegin 方法”一节 第 519 页	准备在表上执行新的查找。
“FindFirst 方法”一节 第 520 页	在表中从开头往前移动，查找与当前索引中的一个值或整个值集完全匹配的行。
“FindLast 方法”一节 第 521 页	在表中从末尾向后移动，查找与当前索引中的一个值或整个值集完全匹配的行。
“FindNext 方法”一节 第 523 页	继续执行 ULTable.FindFirst() 搜索，从表中的当前位置向前移动，查看下一行是否与当前索引中的某个值或整个值集完全匹配。
“FindPrevious 方法”一节 第 525 页	继续执行 ULTable.FindLast() 搜索，从表中的当前位置向后移动，查看上一行是否与当前索引中的某个值或整个值集完全匹配。
“GetBoolean 方法”一节 第 268 页（继承自 ULDataReader）	以 System.Boolean 形式返回指定列的值。
“GetByte 方法”一节第 269 页（继承自 ULDataReader）	以无符号 8 位值 (System.Byte) 形式返回指定列的值。
“GetBytes 方法”一节第 269 页（继承自 ULDataReader）	**UL Ext.:** 以 System.Bytes 数组的形式返回指定列的值。仅对 ULDbType.Binary、ULDbType.LongBinary 或 ULDbType.UniqueIdentifier 类型的列有效。
“GetChar 方法”一节第 272 页（继承自 ULDataReader）	UltraLite.NET 不支持此方法。
“GetChars 方法”一节第 273 页（继承自 ULDataReader）	从指定的偏移量开始，将指定的 ULDbType.LongVarchar 列的值的子集复制到目标 System.Char 数组的指定偏移量处。
GetData（继承自 DbDataReader）	返回与请求的列顺序号对应的 DbDataReader 对象。
“GetDataTypeName 方法”一节第 274 页（继承自 ULDataReader）	返回指定列的提供程序数据类型的名称。
“GetDateTime 方法”一节 第 275 页（继承自 ULDataReader）	以 System.DateTime 形式返回指定列的值，其精度为毫秒。

成员名称	说明
“GetDecimal 方法”一节第 275 页（继承自 ULDataReader）	以 System.Decimal 形式返回指定列的值。
“GetDouble 方法”一节第 276 页（继承自 ULDataReader）	以 System.Double 形式返回指定列的值。
“GetEnumerator 方法”一节第 277 页（继承自 ULDataReader）	返回迭代通过 ULDataReader 的 System.Collections.IEnumerator。
“GetFieldType 方法”一节第 277 页（继承自 ULDataReader）	返回最适合于指定列的 System.Type。
“GetFloat 方法”一节第 278 页（继承自 ULDataReader）	以 System.Single 形式返回指定列的值。
“GetGuid 方法”一节第 279 页（继承自 ULDataReader）	以 UUID (System.Guid) 形式返回指定列的值。
“GetInt16 方法”一节第 279 页（继承自 ULDataReader）	以 System.Int16 形式返回指定列的值。
“GetInt32 方法”一节第 280 页（继承自 ULDataReader）	以 Int32 形式返回指定列的值。
“GetInt64 方法”一节第 281 页（继承自 ULDataReader）	以 Int64 形式返回指定列的值。
“GetName 方法”一节第 281 页（继承自 ULDataReader）	返回指定列的名称。
“GetOrdinal 方法”一节第 282 页（继承自 ULDataReader）	返回指定列的列 ID。
GetProviderSpecificFieldType（继承自 DbDataReader）	返回指定列特定于提供程序的字段类型。
GetProviderSpecificValue（继承自 DbDataReader）	以 Object 实例的形式获取指定列的值。
GetProviderSpecificValues（继承自 DbDataReader）	获取当前行的集合中所有特定于提供程序的属性列。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“GetSchemaTable 方法”一节第 284 页（继承自 ULDataReader）	返回描述 ULDataReader 的列元数据的 System.Data.DataTable。
“GetString 方法”一节第 286 页（继承自 ULDataReader）	以 System.String 形式返回指定列的值。
“GetTimeSpan 方法”一节第 286 页（继承自 ULDataReader）	以 System.TimeSpan 形式返回指定列的值，其精度为毫秒。
“GetUInt16 方法”一节第 287 页（继承自 ULDataReader）	以 System.UInt16 形式返回指定列的值。
“GetUInt32 方法”一节第 288 页（继承自 ULDataReader）	以 UInt32 形式返回指定列的值。
“GetUInt64 方法”一节第 288 页（继承自 ULDataReader）	以 System.UInt64 形式返回指定列的值。
“GetValue 方法”一节第 289 页（继承自 ULDataReader）	返回以本地格式表示的指定列的值。
“GetValues 方法”一节第 290 页（继承自 ULDataReader）	返回当前行的所有列值。
“Insert 方法”一节第 526 页	插入含当前列值（使用 set 方法来指定）的新行。 在每次插入之前必须先调用 ULTable.InsertBegin。
“InsertBegin 方法”一节第 527 页	通过将所有当前列值设置为其缺省值来准备在此表中插入新行。
“IsDBNull 方法”一节第 291 页（继承自 ULDataReader）	检查指定列的值是否为 NULL。
“LookupBackward 方法”一节第 527 页	在表中从末尾向后移动，查找匹配或小于当前索引中某个值或整个值集的行。
“LookupBegin 方法”一节第 529 页	准备在此表中执行新的查找。要搜索的值是通过对用于打开此表的索引中的列调用相应的 setType 方法来指定的。

成员名称	说明
“LookupForward 方法”一节 第 529 页	在表中从开头往前移动，查找匹配或大于当前索引中某个值或整个值集的行。
“MoveAfterLast 方法”一节 第 291 页（继承自 ULDataReader）	**UL Ext.:** 将游标定位到游标的最后一行之后。
“MoveBeforeFirst 方法”一节 第 292 页（继承自 ULDataReader）	**UL Ext.:** 将游标定位到游标的第一行之前。
“MoveFirst 方法”一节 第 292 页（继承自 ULDataReader）	**UL Ext.:** 将游标定位到游标的第一行。
“MoveLast 方法”一节 第 292 页（继承自 ULDataReader）	**UL Ext.:** 将游标定位到游标的最后一行。
“MoveNext 方法”一节 第 293 页（继承自 ULDataReader）	**UL Ext.:** 将游标定位到下一行；如果游标已经位于最后一行，则定位到最后一行之后。
“MovePrevious 方法”一节 第 293 页（继承自 ULDataReader）	**UL Ext.:** 将游标定位到上一行，或第一行之前。
“MoveRelative 方法”一节 第 294 页（继承自 ULDataReader）	**UL Ext.:** 相对于当前行定位游标。
“NextResult 方法”一节 第 294 页（继承自 ULDataReader）	读取批处理 SQL 语句的结果时，将 ULDataReader 推进到下一结果。
“Read 方法”一节第 295 页（继承自 ULDataReader）	将游标定位到下一行；如果游标已经位于最后一行，则定位到最后一行之后。
“SetBoolean 方法”一节 第 404 页（继承自 ULResultSet）	使用 System.Boolean 设置指定列的值。
“SetByte 方法”一节第 404 页（继承自 ULResultSet）	使用 System.Byte（无符号 8 位整数）设置指定列的值。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“SetBytes 方法”一节第 405 页（继承自 ULResultSet）	使用 System.Bytes 数组设置指定列的值。
“SetDBNull 方法”一节第 406 页（继承自 ULResultSet）	将列设置为 NULL。
“SetDateTime 方法”一节第 407 页（继承自 ULResultSet）	使用 System.DateTime 设置指定列的值。
“SetDecimal 方法”一节第 408 页（继承自 ULResultSet）	使用 System.Decimal 设置指定列的值。
“SetDouble 方法”一节第 409 页（继承自 ULResultSet）	使用 System.Double 设置指定列的值。
“SetFloat 方法”一节第 410 页（继承自 ULResultSet）	使用 System.Single 设置指定列的值。
“SetGuid 方法”一节第 411 页（继承自 ULResultSet）	使用 System.Guid 设置指定列的值。
“SetInt16 方法”一节第 412 页（继承自 ULResultSet）	使用 System.Int16 设置指定列的值.
“SetInt32 方法”一节第 413 页（继承自 ULResultSet）	使用 System.Int32 设置指定列的值。
“SetInt64 方法”一节第 414 页（继承自 ULResultSet）	使用 Int64 设置指定列的值。
“SetString 方法”一节第 415 页（继承自 ULResultSet）	使用 System.String 设置指定列的值。
“SetTimeSpan 方法”一节第 416 页（继承自 ULResultSet）	使用 System.TimeSpan 设置指定列的值。
“SetToDefault 方法”一节第 416 页（继承自 ULResultSet）	将指定列的值设置为其缺省值。

成员名称	说明
“SetUInt16 方法”一节 第 417 页（继承自 ULResultSet）	使用 System.UInt16 设置指定列的值。
“SetUInt32 方法”一节 第 418 页（继承自 ULResultSet）	使用 System.UInt32 设置指定列的值。
“SetUInt64 方法”一节 第 419 页（继承自 ULResultSet）	使用 System.UInt64 设置指定列的值。
“Truncate 方法”一节 第 531 页	在临时激活 STOP SYNCHRONIZATION DELETE 语句时，删除表中的所有行。
“Update 方法”一节第 420 页（继承自 ULResultSet）	用当前列值（使用 set 方法指定）来更新当前行。
“UpdateBegin 方法”一节 第 531 页	准备更新表中的当前行。

另请参见

- “ULTable 类”一节第 511 页
- “ExecuteTable() 方法”一节第 118 页
- “ULCommand 类”一节第 86 页
- “ULResultSet 类”一节第 394 页
- IDataReader
- IDataRecord
- IDisposable

Schema 属性

保存表模式。此属性仅在其连接打开期间才有效。

语法

Visual Basic
Public Readonly Property **Schema** As ULTableSchema

C#
public ULTableSchema **Schema** { get;}

属性值

表示表模式的 ULTableSchema 对象。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

此属性表示表的完整模式，包括从 ULDataReader.GetSchemaTable 返回的结果中未表示的 UltraLite.NET 扩展信息。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “ULTableSchema 类”一节第 533 页

DeleteAllRows 方法

删除表中的所有行。

语法

Visual Basic
Public Sub **DeleteAllRows()**

C#
public void **DeleteAllRows();**

注释

在某些应用程序中，它可用于在将一组新数据下载到表中之前删除表中的所有行。使用 ULConnection.StopSynchronizationDelete 可以将行从 UltraLite 数据库删除，但不从统一数据库删除。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “Truncate 方法”一节第 531 页
- “StopSynchronizationDelete 方法”一节第 171 页

FindBegin 方法

准备在表上执行新的查找。

语法

Visual Basic
Public Sub **FindBegin()**

C#
public void **FindBegin();**

注释

要搜索的值是通过对用于打开此表的索引中的列调用相应的 setType 方法来指定的。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “FindFirst() 方法”一节第 520 页
- “FindFirst(Int16) 方法”一节第 521 页
- “FindLast() 方法”一节第 522 页
- “FindLast(Int16) 方法”一节第 522 页

FindFirst 方法

在表中从开头往前移动，查找与当前索引中的一个值或整个值集完全匹配的行。

FindFirst() 方法

在表中从开头往前移动，查找与当前索引中的一个值或整个值集完全匹配的行。

语法

Visual Basic
Public Function **FindFirst()** As Boolean

C#
public bool **FindFirst();**

返回值

如果成功则返回 true，否则返回 false。

注释

若要指定要搜索的值，请为索引中的每一列设置列值。游标停留在第一个与索引值完全匹配的行上。失败时，游标位置在最后一行之后 (ULDataReader.IsEOF)。

每次搜索之前必须首先调用 FindBegin。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “FindFirst 方法”一节第 520 页
- “FindBegin 方法”一节第 519 页
- “FindNext() 方法”一节第 523 页
- “FindPrevious() 方法”一节第 525 页
- “FindFirst(Int16) 方法”一节第 521 页
- “IsEOF 属性”一节第 264 页
- “FindBegin 方法”一节第 519 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

FindFirst(Int16) 方法

在表中从开头往前移动，查找与当前索引中的一个值或部分值集完全匹配的行。

语法

Visual Basic

```
Public Function FindFirst( _
  ByVal numColumns As Short _
) As Boolean
```

C#

```
public bool FindFirst(
  short numColumns
);
```

参数

- **numColumns** 对于复合索引，该参数表示查找中使用的列数。例如，如果有一个三列的索引，而需要查找的值仅基于第一列进行匹配，则应为第一列设置值，然后将 numColumns 的值设置为 1。

返回值

如果成功则返回 true，否则返回 false。

注释

若要指定要搜索的值，请为索引中的每一列设置列值。游标停留在第一个与索引值完全匹配的行上。失败时，游标位置在最后一行之后 (ULDataReader.IsEOF)。

每次搜索之前必须首先调用 FindBegin。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “FindFirst 方法”一节第 520 页
- “FindBegin 方法”一节第 519 页
- “FindNext(Int16) 方法”一节第 524 页
- “FindPrevious(Int16) 方法”一节第 526 页
- “FindFirst() 方法”一节第 520 页
- “IsEOF 属性”一节第 264 页
- “FindBegin 方法”一节第 519 页

FindLast 方法

在表中从末尾向后移动，查找与当前索引中的一个值或整个值集完全匹配的行。

FindLast() 方法

在表中从末尾向后移动，查找与当前索引中的一个值或整个值集完全匹配的行。

语法

Visual Basic
Public Function **FindLast()** As Boolean

C#
public bool **FindLast();**

返回值

如果成功则返回 true，否则返回 false。

注释

若要指定要搜索的值，请为索引中的每一列设置列值。游标停留在所找到的第一个与索引值完全匹配的行上。失败时，游标位置在第一行之前 (ULDataReader.IsBOF)。

每次搜索之前必须首先调用 FindBegin。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “FindLast 方法”一节第 521 页
- “FindBegin 方法”一节第 519 页
- “FindNext() 方法”一节第 523 页
- “FindPrevious() 方法”一节第 525 页
- “FindLast(Int16) 方法”一节第 522 页
- “IsBOF 属性”一节第 263 页
- “FindBegin 方法”一节第 519 页

FindLast(Int16) 方法

在表中从末尾向后移动，查找与当前索引中的一个值或部分值集完全匹配的行。

语法

Visual Basic
```
Public Function FindLast( _
  ByVal numColumns As Short _
) As Boolean
```

C#
```
public bool FindLast(
  short numColumns
);
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

参数

- **numColumns** 对于复合索引，该参数表示查找中使用的列数。例如，如果有一个三列的索引，而需要查找的值仅基于第一列进行匹配，则应为第一列设置值，然后将 numColumns 的值设置为 1。

返回值

如果成功则返回 true；否则返回 false

注释

若要指定要搜索的值，请为索引中的每一列设置列值。游标停留在所找到的第一个与索引值完全匹配的行上。失败时，游标位置在第一行之前 (ULDataReader.IsBOF)。

每次搜索之前必须首先调用 FindBegin。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “FindLast 方法”一节第 521 页
- “FindBegin 方法”一节第 519 页
- “FindNext(Int16) 方法”一节第 524 页
- “FindPrevious(Int16) 方法”一节第 526 页
- “FindLast() 方法”一节第 522 页
- “IsBOF 属性”一节第 263 页
- “FindBegin 方法”一节第 519 页

FindNext 方法

继续执行 ULTable.FindFirst() 搜索，从表中的当前位置向前移动，查看下一行是否与当前索引中的某个值或整个值集完全匹配。

FindNext() 方法

继续执行 ULTable.FindFirst() 搜索，从表中的当前位置向前移动，查看下一行是否与当前索引中的某个值或整个值集完全匹配。

语法

Visual Basic
Public Function **FindNext()** As Boolean

C#
public bool **FindNext();**

返回值

如果成功则返回 true，否则返回 false。

注释

如果下一行与索引值完全匹配，则游标将停留在该行上。失败时，游标位置在最后一行之后 (ULDataReader.IsEOF)。

如果在行更新过程中修改了要搜索的列值，则 FindNext 行为处于不确定状态。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “FindNext 方法”一节第 523 页
- “FindFirst() 方法”一节第 520 页
- “FindNext(Int16) 方法”一节第 524 页
- “IsEOF 属性”一节第 264 页

FindNext(Int16) 方法

继续执行 ULTable.FindFirst() 搜索，从表中的当前位置向前移动，查看下一行是否与当前索引中的某个值或部分值集完全匹配。

语法

Visual Basic

```
Public Function FindNext( _
  ByVal numColumns As Short _
) As Boolean
```

C#

```
public bool FindNext(
  short numColumns
);
```

参数

- **numColumns** 对于复合索引，该参数表示查找中使用的列数。例如，如果有一个三列的索引，而需要查找的值仅基于第一列进行匹配，则应为第一列设置值，然后将 numColumns 的值设置为 1。

返回值

如果成功则返回 true，否则返回 false。

注释

如果下一行与索引值完全匹配，则游标将停留在该行上。失败时，游标位置在最后一行之后 (ULDataReader.IsEOF)。

如果在行更新过程中修改了要搜索的列值，则 FindNext 行为处于不确定状态。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “FindNext 方法”一节第 523 页
- “FindFirst(Int16) 方法”一节第 521 页
- “FindNext() 方法”一节第 523 页
- “FindFirst() 方法”一节第 520 页
- “IsEOF 属性”一节第 264 页

FindPrevious 方法

继续执行 ULTable.FindLast() 搜索，从表中的当前位置向后移动，查看上一行是否与当前索引中的某个值或整个值集完全匹配。

FindPrevious() 方法

继续执行 ULTable.FindLast() 搜索，从表中的当前位置向后移动，查看上一行是否与当前索引中的某个值或整个值集完全匹配。

语法

Visual Basic
Public Function **FindPrevious()** As Boolean

C#
public bool **FindPrevious();**

返回值

如果成功则返回 true，否则返回 false。

注释

如果上一行与索引值完全匹配，则游标将停留在该行上。失败时，游标位置在第一行之前 (ULDataReader.IsBOF)。

如果在行更新过程中修改了要搜索的列值，则 FindPrevious 行为处于不确定状态。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “FindPrevious 方法”一节第 525 页
- “FindLast() 方法”一节第 522 页
- “FindPrevious(Int16) 方法”一节第 526 页
- “IsBOF 属性”一节第 263 页

FindPrevious(Int16) 方法

继续执行 ULTable.FindLast() 搜索，从表中的当前位置向后移动，查看上一行是否与当前索引中的某个值或部分值集完全匹配。

语法

Visual Basic

```
Public Function FindPrevious( _
  ByVal numColumns As Short _
) As Boolean
```

C#

```
public bool FindPrevious(
  short numColumns
);
```

参数

- **numColumns** 对于复合索引，该参数表示查找中使用的列数。例如，如果有一个三列的索引，而需要查找的值仅基于第一列进行匹配，则应为第一列设置值，然后将 numColumns 的值设置为 1。

返回值

如果成功则返回 true，否则返回 false。

注释

如果上一行与索引值完全匹配，则游标将停留在该行上。失败时，游标位置在第一行之前 (ULDataReader.IsBOF)。

如果在行更新过程中修改了要搜索的列值，则 FindPrevious 行为处于不确定状态。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “FindPrevious 方法”一节第 525 页
- “FindLast() 方法”一节第 522 页
- “FindLast(Int16) 方法”一节第 522 页
- “FindPrevious() 方法”一节第 525 页
- “IsBOF 属性”一节第 263 页

Insert 方法

插入含当前列值（使用 set 方法来指定）的新行。

在每次插入之前必须先调用 ULTable.InsertBegin。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Sub **Insert()**

C#
public void **Insert();**

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “InsertBegin 方法”一节第 527 页

InsertBegin 方法

通过将所有当前列值设置为其缺省值来准备在此表中插入新行。

语法

Visual Basic
Public Sub **InsertBegin()**

C#
public void **InsertBegin();**

注释

调用相应的 SetType 或 AppendType 方法，以指定要插入的非缺省值。

在执行 Insert 方法之前，行并未真正插入，行中的数据也并未真正更改；此更改在提交之前不是永久更改。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “Insert 方法”一节第 526 页
- “Insert 方法”一节第 526 页

LookupBackward 方法

在表中从末尾向后移动，查找匹配或小于当前索引中某个值或整个值集的行。

LookupBackward() 方法

在表中从末尾向后移动，查找匹配或小于当前索引中某个值或整个值集的行。

语法

Visual Basic
Public Function **LookupBackward()** As Boolean

C#
public bool **LookupBackward();**

返回值

如果成功则返回 true，否则返回 false。

注释

若要指定要搜索的值，请为索引中的每一列设置列值。游标停留在第一个与索引值匹配或小于该索引值的行上。失败时（即没有任何行小于要查找的值），游标位置在第一行之前 (ULDataReader.IsBOF)。

每次搜索之前必须首先调用 LookupBegin。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “LookupBackward 方法”一节第 527 页
- “LookupBegin 方法”一节第 529 页
- “LookupBackward(Int16) 方法”一节第 528 页
- “IsBOF 属性”一节第 263 页

LookupBackward(Int16) 方法

在表中从开头向后移动，查找匹配或小于当前索引中某个值或部分值集的行。

语法

Visual Basic
```
Public Function LookupBackward( _
  ByVal numColumns As Short _
) As Boolean
```

C#
```
public bool LookupBackward(
  short numColumns
);
```

参数

- **numColumns** 对于复合索引，是指在查找中使用的列数。例如，如果有一个三列的索引，而需要查找的值仅基于第一列进行匹配，则应为第一列设置值，然后将 numColumns 的值设置为 1。

返回值

如果成功则返回 true，否则返回 false。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

若要指定要搜索的值，请为索引中的每一列设置列值。游标停留在所找到的第一个与索引值匹配或小于该索引值的行上。失败时（即没有任何行小于要查找的值），游标位置在第一行之前 (ULDataReader.IsBOF)。

每次搜索之前必须首先调用 LookupBegin。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “LookupBackward 方法”一节第 527 页
- “LookupBegin 方法”一节第 529 页
- “LookupBackward(Int16) 方法”一节第 528 页
- “IsBOF 属性”一节第 263 页

LookupBegin 方法

准备在此表中执行新的查找。要搜索的值是通过对用于打开此表的索引中的列调用相应的 setType 方法来指定的。

语法

Visual Basic
Public Sub **LookupBegin()**

C#
public void **LookupBegin();**

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “LookupForward() 方法”一节第 529 页
- “LookupForward(Int16) 方法”一节第 530 页
- “LookupBackward() 方法”一节第 527 页
- “LookupBackward(Int16) 方法”一节第 528 页

LookupForward 方法

在表中从开头往前移动，查找匹配或大于当前索引中某个值或整个值集的行。

LookupForward() 方法

在表中从开头往前移动，查找匹配或大于当前索引中某个值或整个值集的行。

语法

Visual Basic
Public Function **LookupForward()** As Boolean

C#
public bool **LookupForward();**

返回值

如果成功则返回 true，否则返回 false。

注释

若要指定要搜索的值，请为索引中的每一列设置列值。游标停留在第一个与索引值匹配或大于索引值的行上。失败时（即没有任何行大于要查找的值），游标位置在最后一行之后 (ULDataReader.IsEOF)。

每次搜索之前必须首先调用 ULTable.LookupBegin()。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “LookupForward 方法”一节第 529 页
- “LookupBegin 方法”一节第 529 页
- “LookupForward(Int16) 方法”一节第 530 页
- “IsEOF 属性”一节第 264 页

LookupForward(Int16) 方法

在表中从开头往前移动，查找匹配或大于当前索引中某个值或部分值集的行。

语法

Visual Basic
```
Public Function LookupForward( _
  ByVal numColumns As Short _
) As Boolean
```

C#
```
public bool LookupForward(
  short numColumns
);
```

参数

- **numColumns** 对于复合索引，是指在查找中使用的列数。例如，如果有一个三列的索引，而需要查找的值仅基于第一列进行匹配，则应为第一列设置值，然后将 numColumns 的值设置为 1。

返回值

如果成功则返回 true，否则返回 false。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

注释

若要指定要搜索的值，请为索引中的每一列设置列值。游标停留在第一个与索引值匹配或大于索引值的行上。失败时（即没有任何行大于要查找的值），游标位置在最后一行之后 (ULDataReader.IsEOF)。

每次搜索之前必须首先调用 LookupBegin。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “LookupForward 方法”一节第 529 页
- “LookupBegin 方法”一节第 529 页
- “LookupForward() 方法”一节第 529 页
- “IsEOF 属性”一节第 264 页
- “LookupBegin 方法”一节第 529 页

Truncate 方法

在临时激活 STOP SYNCHRONIZATION DELETE 语句时，删除表中的所有行。

语法

Visual Basic
Public Sub **Truncate()**

C#
public void **Truncate();**

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “DeleteAllRows 方法”一节第 519 页

UpdateBegin 方法

准备更新表中的当前行。

语法

Visual Basic
Public Sub **UpdateBegin()**

C#
public void **UpdateBegin();**

注释

列值通过调用相应的 setType 或 AppendType 方法进行修改。列上的第一个添加操作将在添加新值之前清除当前的列值。

在调用 ULResultSet.Update() 之前，行中的数据并未真正更改；此更改在提交之前不是永久更改。

如果修改用于打开表的索引中的列，将会对任何活动搜索产生不可预知的影响。无法更新表主键中的列。

另请参见

- “ULTable 类”一节第 511 页
- “ULTable 成员”一节第 511 页
- “Update 方法”一节第 420 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULTableSchema 类

UL Ext.: 表示 UltraLite 表的模式。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULTableSchema**
  Inherits ULCursorSchema

C#
public sealed class **ULTableSchema**: ULCursorSchema

注释

没有用于此类的构造函数。ULTableSchema 对象作为表的 ULTable.Schema 附加在表上。

Inherits: ULCursorSchema

另请参见

- “ULTableSchema 成员”一节第 533 页
- “ULTableSchema 类”一节第 533 页
- “Schema 属性”一节第 518 页
- “ULCursorSchema 类”一节第 217 页

ULTableSchema 成员

公共属性

成员名称	说明
“ColumnCount 属性”一节第 218 页（继承自 ULCursorSchema）	返回游标中的列数。
“IndexCount 属性”一节第 535 页	返回此表的索引数。
“IsNeverSynchronized 属性”一节第 536 页	检查此表是否被标记为永不同步。
“IsOpen 属性”一节第 219 页（继承自 ULCursorSchema）	检查游标模式当前是否处于打开状态。
“Name 属性”一节第 536 页	返回表的名称。
“PrimaryKey 属性”一节第 537 页	返回表的主键的索引模式。

成员名称	说明
“UploadUnchangedRows 属性”一节第 537 页	检查数据库是否上载未更改的行。

公共方法

成员名称	说明
“GetColumnDefaultValue 方法”一节第 538 页	返回指定列的缺省值。
“GetColumnID 方法”一节第 219 页（继承自 ULCursorSchema）	返回指定列的列 ID。
“GetColumnName 方法”一节第 220 页（继承自 ULCursorSchema）	返回由指定的列 ID 标识的列的名称。
“GetColumnPartitionSize 方法”一节第 538 页	返回分配给指定列的全局自动增量分区大小。
“GetColumnPrecision 方法”一节第 221 页（继承自 ULCursorSchema）	如果列为数值列（SQL 类型 NUMERIC），则返回由指定的列 ID 标识的列的精度。
“GetColumnSQLName 方法”一节第 222 页（继承自 ULCursorSchema）	返回由指定的列 ID 标识的列的名称。
“GetColumnScale 方法”一节第 223 页（继承自 ULCursorSchema）	如果列为数值列（SQL 类型 NUMERIC），则返回由指定的列 ID 标识的列的小数位数。
“GetColumnSize 方法”一节第 223 页（继承自 ULCursorSchema）	如果列是指定大小的列（SQL 类型 BINARY 或 CHAR），则返回由指定的列 ID 标识的列的大小。
“GetColumnULDbType 方法”一节第 224 页（继承自 ULCursorSchema）	返回由指定的列 ID 标识的列的 UltraLite.NET 数据类型。
“GetIndex 方法”一节第 539 页	返回指定索引的索引模式。
“GetIndexName 方法”一节第 540 页	返回由指定的索引 ID 标识的索引的名称。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

成员名称	说明
“GetOptimalIndex 方法”一节第 540 页	用于使用指定列搜索表的最佳索引。
“GetPublicationPredicate 方法”一节第 541 页	返回指定发布中此表的发布谓语。
“GetSchemaTable 方法”一节第 224 页（继承自 ULCursorSchema）	返回描述 ULDataReader 的列模式的 System.Data.DataTable。
“IsColumnAutoIncrement 方法”一节第 542 页	检查指定列的缺省值是否设置为自动增量。
“IsColumnCurrentDate 方法”一节第 542 页	检查指定列的缺省值是否设置为当前日期 (ULDbType.Date)。
“IsColumnCurrentTime 方法”一节第 543 页	检查指定列的缺省值是否设置为当前时间 (ULDbType.Time)。
“IsColumnCurrentTimestamp 方法”一节第 543 页	检查指定列的缺省值是否设置为当前时间戳 (ULDbType.TimeStamp)。
“IsColumnGlobalAutoIncrement 方法”一节第 544 页	检查指定列的缺省值是否设置为全局自动增量。
“IsColumnNewUUID 方法”一节第 545 页	检查指定列的缺省值是否设置为新的 UUID (System.Guid)。
“IsColumnNullable 方法”一节第 545 页	检查指定列是否可为空。
“IsInPublication 方法”一节第 546 页	检查此表是否包含在指定发布中。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 类”一节第 533 页
- “Schema 属性”一节第 518 页
- “ULCursorSchema 类”一节第 217 页

IndexCount 属性

返回此表的索引数。

语法

Visual Basic
Public Readonly Property **IndexCount** As Integer

C#
public int **IndexCount** { get;}

属性值

表的索引数目；如果表模式关闭，则为 0。

注释

索引 ID 的范围是从 1 到 IndexCount（含 1 和 IndexCount）。

注意：索引 ID 和计数在模式升级过程中可能发生变化。为了正确地标识索引，请按名称访问它，或者在模式升级后刷新高速缓存中的 ID 和计数。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页

IsNeverSynchronized 属性

检查此表是否被标记为永不同步。

语法

Visual Basic
Public Readonly Property **IsNeverSynchronized** As Boolean

C#
public bool **IsNeverSynchronized** { get;}

属性值

如果此表被标记为永不同步，则为 true；否则为 false。

注释

被标记为永不同步的表永远不同步，即使它们包含在发布中。这些表有时称为 "不同步" 表。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页

Name 属性

返回表的名称。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic
Public Overrides Readonly Property **Name** As String

C#
public override string **Name** { get;}

属性值

字符串形式的表名。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页

PrimaryKey 属性

返回表的主键的索引模式。

语法

Visual Basic
Public Readonly Property **PrimaryKey** As ULIndexSchema

C#
public ULIndexSchema **PrimaryKey** { get;}

属性值

表示表的主键的 ULIndexSchema 对象。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页
- “ULIndexSchema 类”一节第 332 页

UploadUnchangedRows 属性

检查数据库是否上载未更改的行。

语法

Visual Basic
Public Readonly Property **UploadUnchangedRows** As Boolean

C#
public bool **UploadUnchangedRows** { get;}

属性值

如果表被标记为在同步过程中始终上载所有行，则为 true；如果表被标记为仅上载已更改的行，则为 false。

注释

当对表进行同步时，带有此标记的表将上载未更改和已更改的行。这些表有时称为 "全部同步" 表。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页

GetColumnDefaultValue 方法

返回指定列的缺省值。

语法

Visual Basic

```
Public Function GetColumnDefaultValue( _
  ByVal columnID As Integer _
) As String
```

C#

```
public string  GetColumnDefaultValue(
  int columnID
);
```

参数

- **columnID**　列的 ID 号。值必须在 [0,ULCursorSchema.ColumnCount-1] 范围内。表中第一列的 ID 值为 0。

返回值

字符串形式的指定列的缺省值；如果缺省值为空，则为空值引用（Visual Basic 中是 Nothing）。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页
- “ColumnCount 属性”一节第 218 页

GetColumnPartitionSize 方法

返回分配给指定列的全局自动增量分区大小。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic

```
Public Function GetColumnPartitionSize( _
  ByVal columnID As Integer _
) As UInt64
```

C#

```
public ulong GetColumnPartitionSize(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULCursorSchema.ColumnCount-1] 范围内。表中第一列的 ID 值为 0。

返回值

System.UInt64 形式的列的全局自动增量分区大小。

注释

给定表中的所有全局自动增量列都共享同一全局自动增量分区。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页
- “IsColumnGlobalAutoIncrement 方法”一节第 544 页
- “ColumnCount 属性”一节第 218 页
- UInt64

GetIndex 方法

返回指定索引的索引模式。

语法

Visual Basic

```
Public Function GetIndex( _
  ByVal name As String _
) As ULIndexSchema
```

C#

```
public ULIndexSchema GetIndex(
  string  name
);
```

参数

- **name** 索引的名称。

返回值

表示指定索引的 ULIndexSchema 对象。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页
- “ULIndexSchema 类”一节第 332 页

GetIndexName 方法

返回由指定的索引 ID 标识的索引的名称。

语法

Visual Basic

```
Public Function GetIndexName( _
   ByVal indexID As Integer _
) As String
```

C#

```
public string  GetIndexName(
   int indexID
);
```

参数

- **indexID**　索引的 ID。值必须在 [1,IndexCount] 范围内。

返回值

字符串形式的索引名称。

注释

索引 ID 和索引计数在模式升级过程中可能发生变化。为了正确地标识索引，请按名称访问它，或者在模式升级后刷新高速缓存中的 ID 和计数。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页
- “IndexCount 属性”一节第 535 页

GetOptimalIndex 方法

用于使用指定列搜索表的最佳索引。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

语法

Visual Basic

```
Public Function GetOptimalIndex( _
  ByVal columnID As Integer _
) As ULIndexSchema
```

C#

```
public ULIndexSchema GetOptimalIndex(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULCursorSchema.ColumnCount>-1] 范围内。表中第一列的 ID 值为 0。

返回值

表示指定列的最佳索引的 ULIndexSchema 对象。

注释

指定列是索引中的第一列，但索引可以有多列。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页
- “ColumnCount 属性”一节第 218 页
- “ULIndexSchema 类”一节第 332 页

GetPublicationPredicate 方法

返回指定发布中此表的发布谓语。

语法

Visual Basic

```
Public Function GetPublicationPredicate( _
  ByVal pubName As String _
) As String
```

C#

```
public string  GetPublicationPredicate(
  string  pubName
);
```

参数

- **pubName** 发布的名称。

返回值

字符串形式的发布谓语。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页

IsColumnAutoIncrement 方法

检查指定列的缺省值是否设置为自动增量。

语法

Visual Basic

```
Public Function IsColumnAutoIncrement( _
  ByVal columnID As Integer _
) As Boolean
```

C#

```
public bool IsColumnAutoIncrement(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULCursorSchema.ColumnCount-1] 范围内。表中第一列的 ID 值为 0。

返回值

如果列自动递增，则返回 true；如果不自动递增，则返回 false。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页
- “ColumnCount 属性”一节第 218 页

IsColumnCurrentDate 方法

检查指定列的缺省值是否设置为当前日期 (ULDbType.Date)。

语法

Visual Basic

```
Public Function IsColumnCurrentDate( _
  ByVal columnID As Integer _
) As Boolean
```

C#

```
public bool IsColumnCurrentDate(
  int columnID
);
```

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

参数

- **columnID** 列的 ID 号。值必须在 [0,ULCursorSchema.ColumnCount-1] 范围内。表中第一列的 ID 值为 0。

返回值

如果列缺省为当前日期，则返回 true；如果列不是缺省为当前日期，则返回 false。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页
- “ColumnCount 属性”一节第 218 页
- “ULDbType 枚举”一节第 297 页

IsColumnCurrentTime 方法

检查指定列的缺省值是否设置为当前时间 (ULDbType.Time)。

语法

Visual Basic

```
Public Function IsColumnCurrentTime( _
  ByVal columnID As Integer _
) As Boolean
```

C#

```
public bool IsColumnCurrentTime(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULCursorSchema.ColumnCount-1] 范围内。表中第一列的 ID 值为 0。

返回值

如果列缺省为当前时间，则返回 true；如果列不是缺省为当前时间，则返回 false。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页
- “ColumnCount 属性”一节第 218 页
- “ULDbType 枚举”一节第 297 页

IsColumnCurrentTimestamp 方法

检查指定列的缺省值是否设置为当前时间戳 (ULDbType.TimeStamp)。

语法

Visual Basic

```
Public Function IsColumnCurrentTimestamp( _
  ByVal columnID As Integer _
) As Boolean
```

C#

```
public bool IsColumnCurrentTimestamp(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULCursorSchema.ColumnCount-1] 范围内。表中第一列的 ID 值为 0。

返回值

如果列缺省为当前时间戳，则返回 true；如果列不是缺省为当前时间戳，则返回 false。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页
- “ColumnCount 属性”一节第 218 页
- “ULDbType 枚举”一节第 297 页

IsColumnGlobalAutoIncrement 方法

检查指定列的缺省值是否设置为全局自动增量。

语法

Visual Basic

```
Public Function IsColumnGlobalAutoIncrement( _
  ByVal columnID As Integer _
) As Boolean
```

C#

```
public bool IsColumnGlobalAutoIncrement(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULCursorSchema.ColumnCount-1] 范围内。表中第一列的 ID 值为 0。

返回值

如果列全局自动递增，则返回 true；如果不全局自动递增，则返回 false。

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页
- “GetColumnPartitionSize 方法”一节第 538 页
- “DatabaseID 属性”一节第 140 页
- “ColumnCount 属性”一节第 218 页

IsColumnNewUUID 方法

检查指定列的缺省值是否设置为新的 UUID (System.Guid)。

语法

Visual Basic

```
Public Function IsColumnNewUUID( _
   ByVal columnID As Integer _
) As Boolean
```

C#

```
public bool IsColumnNewUUID(
   int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULCursorSchema.ColumnCount-1] 范围内。表中第一列的 ID 值为 0。

返回值

如果列缺省为新的 UUID，则返回 true；如果列不是缺省为新的 UUID，则返回 false。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页
- “ColumnCount 属性”一节第 218 页
- Guid

IsColumnNullable 方法

检查指定列是否可为空。

语法

Visual Basic

```
Public Function IsColumnNullable( _
   ByVal columnID As Integer _
) As Boolean
```

C#
```
public bool IsColumnNullable(
  int columnID
);
```

参数

- **columnID** 列的 ID 号。值必须在 [0,ULCursorSchema.ColumnCount-1] 范围内。表中第一列的 ID 值为 0。

返回值

如果列可为空，则返回 true；如果不可为空，则返回 false。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页
- “ColumnCount 属性”一节第 218 页

IsInPublication 方法

检查此表是否包含在指定发布中。

语法

Visual Basic
```
Public Function IsInPublication( _
  ByVal pubName As String _
) As Boolean
```

C#
```
public bool IsInPublication(
  string  pubName
);
```

参数

- **pubName** 发布的名称。

返回值

如果此表包含在发布中，则返回 True；如果此表未包含在发布中，则返回 false。

另请参见

- “ULTableSchema 类”一节第 533 页
- “ULTableSchema 成员”一节第 533 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULTransaction 类

表示 SQL 事务。此类无法继承。

语法

Visual Basic
Public NotInheritable Class **ULTransaction**
  Inherits DbTransaction

C#
public sealed class **ULTransaction**: DbTransaction

注释

ULTransaction 没有构造函数。要获取 ULTransaction 对象，请使用 ULConnection.BeginTransaction()。要将命令与事务关联，请使用 ULCommand.Transaction。

提交或回退事务后，连接恢复为在执行操作时自动提交所有操作。要将更多操作组合在一起，必须创建一个新事务。

Inherits: System.Data.Common.DbTransaction

Implements: System.Data.IDbTransaction、System.IDisposable

另请参见

- “ULTransaction 成员”一节第 547 页
- “BeginTransaction() 方法”一节第 144 页
- “Transaction 属性”一节第 98 页
- DbTransaction
- IDbTransaction
- IDisposable

ULTransaction 成员

公共属性

成员名称	说明
“Connection 属性”一节第 548 页	返回与此事务关联的连接。
“IsolationLevel 属性”一节第 549 页	返回此事务的隔离级别。

公共方法

成员名称	说明
“Commit 方法”一节 第 549 页	提交数据库事务。
Dispose（继承自 DbTransaction）	释放 DbTransaction 使用的非托管资源。
“Rollback 方法”一节 第 550 页	回退事务对数据库的未完成的更改。

另请参见

- “ULTransaction 类”一节第 547 页
- “BeginTransaction() 方法”一节第 144 页
- “Transaction 属性”一节第 98 页
- DbTransaction
- IDbTransaction
- IDisposable

Connection 属性

返回与此事务关联的连接。

语法

Visual Basic
Public Readonly Property **Connection** As ULConnection

C#
public ULConnection **Connection** { get;}

属性值

与事务关联的 ULConnection 对象；如果事务不再有效，则为空值引用（在 Visual Basic 中是 Nothing）。

注释

这是 System.Data.IDbTransaction.Connection 和 System.Data.Common.DbCommand.Connection 的强类型版本。

Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

另请参见

- “ULTransaction 类”一节第 547 页
- “ULTransaction 成员”一节第 547 页
- “BeginTransaction() 方法”一节第 144 页
- “ULConnection 类”一节第 131 页
- IDbTransaction.Connection
- DbCommand.Connection

IsolationLevel 属性

返回此事务的隔离级别。

语法

Visual Basic
Public Overrides Readonly Property **IsolationLevel** As IsolationLevel

C#
public override IsolationLevel **IsolationLevel** { get;}

属性值

System.Data.IsolationLevel 值之一。UltraLite.NET 仅支持 System.Data.IsolationLevel.ReadUncommitted。

另请参见

- “ULTransaction 类”一节第 547 页
- “ULTransaction 成员”一节第 547 页
- “BeginTransaction() 方法”一节第 144 页
- IsolationLevel
- IsolationLevel.ReadUncommitted

Commit 方法

提交数据库事务。

语法

Visual Basic
Public Overrides Sub **Commit()**

C#
public override void **Commit();**

注释

提交或回退事务后，连接恢复为在执行操作时自动提交所有操作。要将更多操作组合在一起，必须创建一个新事务。

如果由于数据库错误（例如，参照完整性错误）而导致 Commit() 失败，则事务会保持活动状态。更正错误并再次调用 Commit() 方法或调用 ULTransaction.Rollback() 以完成事务。

另请参见

- “ULTransaction 类”一节第 547 页
- “ULTransaction 成员”一节第 547 页
- “Rollback 方法”一节第 550 页

Rollback 方法

回退事务对数据库的未完成的更改。

语法

Visual Basic
Public Overrides Sub **Rollback()**

C#
public override void **Rollback();**

注释

提交或回退事务后，连接恢复为在执行操作时自动提交所有操作。要将更多操作组合在一起，必须创建一个新事务。

另请参见

- “ULTransaction 类”一节第 547 页
- “ULTransaction 成员”一节第 547 页
- “Commit 方法”一节第 549 页

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

索引

A

Abort 属性
  ULRowsCopiedEventArgs 类 [UltraLite .NET API], 425
ActiveSyncInvoked 方法
  ULActiveSyncListener 接口 [UltraLite .NET API], 53
ActiveSync 同步
  UltraLite.NET, 29
Add(Int32, Int32) 方法
  ULBulkCopyColumnMappingCollection 类 [UltraLite .NET API], 78
Add(Int32, String) 方法
  ULBulkCopyColumnMappingCollection 类 [UltraLite .NET API], 79
Add(Object) 方法
  ULParameterCollection 类 [UltraLite .NET API], 376
Add(String, Int32) 方法
  ULBulkCopyColumnMappingCollection 类 [UltraLite .NET API], 80
Add(String, Object) 方法
  ULParameterCollection 类 [UltraLite .NET API], 378
Add(String, String) 方法
  ULBulkCopyColumnMappingCollection 类 [UltraLite .NET API], 81
Add(String, ULDbType, Int32, String) 方法
  ULParameterCollection 类 [UltraLite .NET API], 381
Add(String, ULDbType, Int32) 方法
  ULParameterCollection 类 [UltraLite .NET API], 380
Add(String, ULDbType) 方法
  ULParameterCollection 类 [UltraLite .NET API], 379
Add(ULBulkCopyColumnMapping) 方法
  ULBulkCopyColumnMappingCollection 类 [UltraLite .NET API], 78
Add(ULParameter) 方法
  ULParameterCollection 类 [UltraLite .NET API], 377
AdditionalParms 属性
  ULConnectionParms 类 [UltraLite .NET API], 180
  ULSyncParms 类 [UltraLite .NET API], 480
AddRange(Array) 方法
  ULParameterCollection 类 [UltraLite .NET API], 382
AddRange(ULParameter[]) 方法
  ULParameterCollection 类 [UltraLite .NET API], 383
AppendBytes 方法
  ULResultSet 类 [UltraLite .NET API], 400
AppendChars 方法
  ULResultSet 类 [UltraLite .NET API], 402
AuthenticationParms 属性
  ULFileTransfer 类 [UltraLite .NET API], 315
  ULSyncParms 类 [UltraLite .NET API], 480
AuthStatus 属性
  ULFileTransfer 类 [UltraLite .NET API], 315
  ULSyncResult 类 [UltraLite .NET API], 506
AuthValue 属性
  ULFileTransfer 类 [UltraLite .NET API], 315
  ULSyncResult 类 [UltraLite .NET API], 506
AutoCommit 模式
  UltraLite.NET 开发, 25
安装
  SQL Anywhere Explorer for UltraLite, 7

B

BatchSize 属性
  ULBulkCopy 类 [UltraLite .NET API], 61
BeginExecuteNonQuery(AsyncCallback, Object) 方法
  ULCommand 类 [UltraLite .NET API], 99
BeginExecuteNonQuery 方法
  ULCommand 类 [UltraLite .NET API], 99
BeginExecuteReader(AsyncCallback, Object, CommandBehavior) 方法
  ULCommand 类 [UltraLite .NET API], 103
BeginExecuteReader(AsyncCallback, Object) 方法
  ULCommand 类 [UltraLite .NET API], 102
BeginExecuteReader(CommandBehavior) 方法
  ULCommand 类 [UltraLite .NET API], 101
BeginExecuteReader 方法
  ULCommand 类 [UltraLite .NET API], 100
BeginTransaction(IsolationLevel) 方法
  ULConnection 类 [UltraLite .NET API], 145
BeginTransaction 方法
  ULConnection 类 [UltraLite .NET API], 144
BulkCopyTimeout 属性

ULBulkCopy 类 [UltraLite .NET API], 62
BytesReceived 属性
ULFileTransferProgressData 类 [UltraLite .NET API], 328
帮助
技术支持, xii
表
UltraLite.NET 模式信息于, 26
部署
UltraLite.NET, 43

C

CacheSize 属性
ULConnectionParms 类 [UltraLite .NET API], 182
ULConnectionStringBuilder 类 [UltraLite .NET API], 194
CancelGetNotification 方法
ULConnection 类 [UltraLite .NET API], 146
Cancel 方法
ULCommand 类 [UltraLite .NET API], 104
CanCreateDataSourceEnumerator 属性
ULFactory 类 [UltraLite .NET API], 308
CaseSensitive 属性
ULCreateParms 类 [UltraLite .NET API], 209
ChangeDatabase 方法
ULConnection 类 [UltraLite .NET API], 147
ChangeEncryptionKey 方法
ULConnection 类 [UltraLite .NET API], 148
ChangePassword 方法
ULConnection 类 [UltraLite .NET API], 148
ChecksumLevel 属性
ULCreateParms 类 [UltraLite .NET API], 209
Clear 方法
ULParameterCollection 类 [UltraLite .NET API], 383
Close 方法
ULBulkCopy 类 [UltraLite .NET API], 64
ULConnection 类 [UltraLite .NET API], 149
ULDataReader 类 [UltraLite .NET API], 268
CollationName 属性
ULDatabaseSchema 类 [UltraLite .NET API], 248
ColumnCount 属性
ULCursorSchema 类 [UltraLite .NET API], 218
ULIndexSchema 类 [UltraLite .NET API], 333
ColumnMappings 属性
ULBulkCopy 类 [UltraLite .NET API], 62
Columns 属性
ULMetaDataCollectionNames 类 [UltraLite .NET API], 346
CommandText 属性
ULCommand 类 [UltraLite .NET API], 93
CommandTimeout 属性
ULCommand 类 [UltraLite .NET API], 94
CommandType 属性
ULCommand 类 [UltraLite .NET API], 94
Command 属性
ULRowUpdatedEventArgs 类 [UltraLite .NET API], 430
ULRowUpdatingEventArgs 类 [UltraLite .NET API], 435
commit 方法
UltraLite.NET 事务, 25
Commit 方法
ULTransaction 类 [UltraLite .NET API], 549
ConnectionName 属性
ULConnectionParms 类 [UltraLite .NET API], 183
ULConnectionStringBuilder 类 [UltraLite .NET API], 195
ConnectionString 属性
ULConnection 类 [UltraLite .NET API], 137
ConnectionTimeout 属性
ULConnection 类 [UltraLite .NET API], 138
Connection 类
UltraLite.NET, 13
Connection 属性
ULCommand 类 [UltraLite .NET API], 95
ULTransaction 类 [UltraLite .NET API], 548
Contains(Object) 方法
ULParameterCollection 类 [UltraLite .NET API], 384
Contains(String) 方法
ULParameterCollection 类 [UltraLite .NET API], 384
ContainsKey 方法
ULConnectionStringBuilder 类 [UltraLite .NET API], 202
Contains 方法
ULBulkCopyColumnMappingCollection 类 [UltraLite .NET API], 81
CopyFrom 方法
ULSyncParms 类 [UltraLite .NET API], 490
CopyTo 方法
ULBulkCopyColumnMappingCollection 类 [UltraLite .NET API], 82

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULParameterCollection 类 [UltraLite .NET API], 385
CountUploadRows(String, Int64) 方法
ULConnection 类 [UltraLite .NET API], 150
CountUploadRows(String, UInt32) 方法
ULConnection 类 [UltraLite .NET API], 149
Count 属性
ULParameterCollection 类 [UltraLite .NET API], 373
CreateCommandBuilder 方法
ULFactory 类 [UltraLite .NET API], 309
CreateCommand 方法
ULConnection 类 [UltraLite .NET API], 151
ULFactory 类 [UltraLite .NET API], 309
CreateConnectionStringBuilder 方法
ULFactory 类 [UltraLite .NET API], 310
CreateConnection 方法
ULFactory 类 [UltraLite .NET API], 310
CreateDataAdapter 方法
ULFactory 类 [UltraLite .NET API], 310
CreateDatabase 方法
ULDatabaseManager 类 [UltraLite .NET API], 240
CreateNotificationQueue 方法
ULConnection 类 [UltraLite .NET API], 152
CreateParameter 方法
ULCommand 类 [UltraLite .NET API], 104
ULFactory 类 [UltraLite .NET API], 311
CurrentScript 属性
ULSqlProgressData 类 [UltraLite .NET API], 455
插入
UltraLite.NET 表行, 23
插入模式
UltraLite.NET, 21
查寻方法
UltraLite.NET, 22
查寻模式
UltraLite.NET, 21
查找方法
UltraLite.NET, 22
查找模式
UltraLite.NET, 21
查找详细信息并请求技术协助
技术支持, xiii
从数据库表中选择数据
UltraLite.NET, 17
错误
UltraLite.NET 处理, 27
提供反馈, xii
错误处理
UltraLite.NET, 27

D

DataAdapter 属性
ULCommandBuilder 类 [UltraLite .NET API], 125
DatabaseID 属性
ULConnection 类 [UltraLite .NET API], 140
DatabaseKey 属性
ULConnectionStringBuilder 类 [UltraLite .NET API], 195
DatabaseManager 类
UltraLite.NET, 13
DatabaseManager 属性
ULConnection 类 [UltraLite .NET API], 140
DatabaseName 属性
ULConnectionStringBuilder 类 [UltraLite .NET API], 196
DatabaseOnCE 属性
ULConnectionParms 类 [UltraLite .NET API], 184
ULConnectionStringBuilder 类 [UltraLite .NET API], 196
DatabaseOnDesktop 属性
ULConnectionParms 类 [UltraLite .NET API], 184
ULConnectionStringBuilder 类 [UltraLite .NET API], 197
Database 属性
ULConnection 类 [UltraLite .NET API], 139
DataSourceInformation 属性
ULMetaDataCollectionNames 类 [UltraLite .NET API], 347
DataSource 属性
ULConnection 类 [UltraLite .NET API], 139
DataTypes 属性
ULMetaDataCollectionNames 类 [UltraLite .NET API], 347
DateFormat 属性
ULCreateParms 类 [UltraLite .NET API], 210
DateOrder 属性
ULCreateParms 类 [UltraLite .NET API], 210
DbType 属性
ULParameter 类 [UltraLite .NET API], 362
DCX
关于, viii
DeclareEvent 方法
ULConnection 类 [UltraLite .NET API], 152

DeleteAllRows 方法
ULTable 类 [UltraLite .NET API], 519
DeleteCommand 属性
ULDataAdapter 类 [UltraLite .NET API], 231
Delete 方法
ULResultSet 类 [UltraLite .NET API], 403
Depth 属性
ULDataReader 类 [UltraLite .NET API], 262
DesignTimeVisible 属性
ULCommand 类 [UltraLite .NET API], 96
DestinationColumn 属性
ULBulkCopyColumnMapping 类 [UltraLite .NET API], 73
DestinationFileName 属性
ULFileTransfer 类 [UltraLite .NET API], 316
DestinationOrdinal 属性
ULBulkCopyColumnMapping 类 [UltraLite .NET API], 73
DestinationPath 属性
ULFileTransfer 类 [UltraLite .NET API], 316
DestinationTableName 属性
ULBulkCopy 类 [UltraLite .NET API], 63
DestroyNotificationQueue 方法
ULConnection 类 [UltraLite .NET API], 153
Direction 属性
ULParameter 类 [UltraLite .NET API], 362
Dispose 方法
ULBulkCopy 类 [UltraLite .NET API], 64
DML
UltraLite.NET, 16
DocCommentXchange (DCX)
关于, viii
DownloadedFile 属性
ULFileTransfer 类 [UltraLite .NET API], 317
DownloadFile(ULFileTransferProgressListener) 方法
ULFileTransfer 类 [UltraLite .NET API], 325
DownloadFile 方法
ULFileTransfer 类 [UltraLite .NET API], 324
DownloadOnly 属性
ULSyncParms 类 [UltraLite .NET API], 481
DropDatabase 方法
ULDatabaseManager 类 [UltraLite .NET API], 241
代码列表
UltraLite.NET C# 教程, 45
UltraLite.NET Visual Basic 教程, 48
动态 SQL
UltraLite.NET 教程, 39
多线程应用程序
UltraLite.NET 概述, 13

E

EncryptionKey 属性
ULConnectionParms 类 [UltraLite .NET API], 185
EndExecuteNonQuery 方法
ULCommand 类 [UltraLite .NET API], 105
EndExecuteReader 方法
ULCommand 类 [UltraLite .NET API], 108
EquivalentTo 方法
ULConnectionStringBuilder 类 [UltraLite .NET API], 202
ExecuteNextSQLPassthroughScript 方法
ULConnection 类 [UltraLite .NET API], 154
ExecuteNonQuery 方法
ULCommand 类 [UltraLite .NET API], 111
ExecuteReader(CommandBehavior) 方法
ULCommand 类 [UltraLite .NET API], 113
ExecuteReader 方法
ULCommand 类 [UltraLite .NET API], 112
ExecuteResultSet(CommandBehavior) 方法
ULCommand 类 [UltraLite .NET API], 116
ExecuteResultSet 方法
ULCommand 类 [UltraLite .NET API], 115
ExecuteScalar 方法
ULCommand 类 [UltraLite .NET API], 117
ExecuteSQLPassthroughScripts 方法
ULConnection 类 [UltraLite .NET API], 155
ExecuteTable(CommandBehavior) 方法
ULCommand 类 [UltraLite .NET API], 119
ExecuteTable(String, String, CommandBehavior) 方法
ULConnection 类 [UltraLite .NET API], 158
ExecuteTable(String, String) 方法
ULConnection 类 [UltraLite .NET API], 156
ExecuteTable(String) 方法
ULConnection 类 [UltraLite .NET API], 155
ExecuteTable 方法
ULCommand 类 [UltraLite .NET API], 118
Explorer for UltraLite (*见* SQL Anywhere Explorer for UltraLite)

F

FieldCount 属性
ULDataReader 类 [UltraLite .NET API], 262
FileAuthCode 属性
ULFileTransfer 类 [UltraLite .NET API], 317

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

FileName 属性
ULFileTransfer 类 [UltraLite .NET API], 318
FileSize 属性
ULFileTransferProgressData 类 [UltraLite .NET API], 328
FileTransferProgressed 方法
ULFileTransferProgressListener 接口 [UltraLite .NET API], 330
FindBegin 方法
ULTable 类 [UltraLite .NET API], 519
FindFirst(Int16) 方法
ULTable 类 [UltraLite .NET API], 521
FindFirst 方法
ULTable 类 [UltraLite .NET API], 520
FindLast(Int16) 方法
ULTable 类 [UltraLite .NET API], 522
FindLast 方法
ULTable 类 [UltraLite .NET API], 522
FindNext(Int16) 方法
ULTable 类 [UltraLite .NET API], 524
FindNext 方法
ULTable 类 [UltraLite .NET API], 523
FindPrevious(Int16) 方法
ULTable 类 [UltraLite .NET API], 526
FindPrevious 方法
ULTable 类 [UltraLite .NET API], 525
FIPS 属性
ULCreateParms 类 [UltraLite .NET API], 210
FLAG_IS_BLOCKING 字段
ULFileTransferProgressData 类 [UltraLite .NET API], 328
ULSyncProgressData 类 [UltraLite .NET API], 492
Flags 属性
ULFileTransferProgressData 类 [UltraLite .NET API], 329
ULSyncProgressData 类 [UltraLite .NET API], 493
ForceDownload 属性
ULFileTransfer 类 [UltraLite .NET API], 318
ForeignKeys 属性
ULMetaDataCollectionNames 类 [UltraLite .NET API], 348
发布
UltraLite.NET 模式信息于, 26
反馈
报告错误, xii
提供, xii
文档, xii
请求更新, xii
访问模式信息
UltraLite.NET 关于, 26
访问数据
UltraLite.NET Table API, 20
复制
Visual Studio 中的 UltraLite 数据库对象, 8

G

GetBoolean 方法
ULDataReader 类 [UltraLite .NET API], 268
GetBytes(Int32, Int64, Byte[], Int32, Int32) 方法
ULDataReader 类 [UltraLite .NET API], 270
GetBytes(Int32) 方法
ULDataReader 类 [UltraLite .NET API], 271
GetByte 方法
ULDataReader 类 [UltraLite .NET API], 269
GetChars 方法
ULDataReader 类 [UltraLite .NET API], 273
GetChar 方法
ULDataReader 类 [UltraLite .NET API], 272
GetColumnDefaultValue 方法
ULTableSchema 类 [UltraLite .NET API], 538
GetColumnID 方法
ULCursorSchema 类 [UltraLite .NET API], 219
GetColumnName 方法
ULCursorSchema 类 [UltraLite .NET API], 220
ULIndexSchema 类 [UltraLite .NET API], 338
GetColumnPartitionSize 方法
ULTableSchema 类 [UltraLite .NET API], 538
GetColumnPrecision 方法
ULCursorSchema 类 [UltraLite .NET API], 221
GetColumnScale 方法
ULCursorSchema 类 [UltraLite .NET API], 223
GetColumnSize 方法
ULCursorSchema 类 [UltraLite .NET API], 223
GetColumnSQLName 方法
ULCursorSchema 类 [UltraLite .NET API], 222
GetColumnULDbType 方法
ULCursorSchema 类 [UltraLite .NET API], 224
GetDatabaseProperty 方法
ULDatabaseSchema 类 [UltraLite .NET API], 251
GetDataTypeName 方法
ULDataReader 类 [UltraLite .NET API], 274
GetDateTime 方法

ULDataReader 类 [UltraLite .NET API], 275
GetDecimal 方法
ULDataReader 类 [UltraLite .NET API], 275
GetDeleteCommand(Boolean) 方法
ULCommandBuilder 类 [UltraLite .NET API], 125
GetDeleteCommand 方法
ULCommandBuilder 类 [UltraLite .NET API], 126
GetDouble 方法
ULDataReader 类 [UltraLite .NET API], 276
GetEnumerator 方法
ULDataReader 类 [UltraLite .NET API], 277
ULParameterCollection 类 [UltraLite .NET API], 386
GetFieldType 方法
ULDataReader 类 [UltraLite .NET API], 277
GetFillParameters 方法
ULDataAdapter 类 [UltraLite .NET API], 235
GetFloat 方法
ULDataReader 类 [UltraLite .NET API], 278
GetGuid 方法
ULDataReader 类 [UltraLite .NET API], 279
GetIndexName 方法
ULTableSchema 类 [UltraLite .NET API], 540
GetIndex 方法
ULTableSchema 类 [UltraLite .NET API], 539
GetInsertCommand(Boolean) 方法
ULCommandBuilder 类 [UltraLite .NET API], 127
GetInsertCommand 方法
ULCommandBuilder 类 [UltraLite .NET API], 128
GetInt16 方法
ULDataReader 类 [UltraLite .NET API], 279
GetInt32 方法
ULDataReader 类 [UltraLite .NET API], 280
GetInt64 方法
ULDataReader 类 [UltraLite .NET API], 281
GetLastDownloadTime 方法
ULConnection 类 [UltraLite .NET API], 159
GetName 方法
ULDataReader 类 [UltraLite .NET API], 281
GetNewUUID 方法
ULConnection 类 [UltraLite .NET API], 160
GetNotificationParameter 方法
ULConnection 类 [UltraLite .NET API], 162
GetNotification 方法
ULConnection 类 [UltraLite .NET API], 161
GetObjectData 方法
ULException 类 [UltraLite .NET API], 305
GetOptimalIndex 方法
ULTableSchema 类 [UltraLite .NET API], 540
GetOrdinal 方法
ULDataReader 类 [UltraLite .NET API], 282
GetPublicationName 方法
ULDatabaseSchema 类 [UltraLite .NET API], 253
GetPublicationPredicate 方法
ULTableSchema 类 [UltraLite .NET API], 541
GetPublicationSchema 方法
ULDatabaseSchema 类 [UltraLite .NET API], 253
GetRowCount 方法
ULDataReader 类 [UltraLite .NET API], 283
GetSchema(String, String[]) 方法
ULConnection 类 [UltraLite .NET API], 164
GetSchema(String) 方法
ULConnection 类 [UltraLite .NET API], 163
GetSchemaTable 方法
ULCursorSchema 类 [UltraLite .NET API], 224
ULDataReader 类 [UltraLite .NET API], 284
GetSchema 方法
ULConnection 类 [UltraLite .NET API], 163
GetShortName 方法
ULConnectionStringBuilder 类 [UltraLite .NET API], 203
GetSQLPassthroughScriptCount 方法
ULConnection 类 [UltraLite .NET API], 162
GetString 方法
ULDataReader 类 [UltraLite .NET API], 286
GetTableName 方法
ULDatabaseSchema 类 [UltraLite .NET API], 254
GetTimeSpan 方法
ULDataReader 类 [UltraLite .NET API], 286
GetUInt16 方法
ULDataReader 类 [UltraLite .NET API], 287
GetUInt32 方法
ULDataReader 类 [UltraLite .NET API], 288
GetUInt64 方法
ULDataReader 类 [UltraLite .NET API], 288
GetUpdateCommand(Boolean) 方法
ULCommandBuilder 类 [UltraLite .NET API], 129
GetUpdateCommand 方法
ULCommandBuilder 类 [UltraLite .NET API], 129
GetValues 方法
ULDataReader 类 [UltraLite .NET API], 290
GetValue 方法
ULDataReader 类 [UltraLite .NET API], 289
GlobalAutoIncrementUsage 属性

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULConnection 类 [UltraLite .NET API], 141
grantConnectTo 方法
UltraLite.NET 开发, 28
GrantConnectTo 方法
ULConnection 类 [UltraLite .NET API], 166
更新
UltraLite.NET 表行, 23
更新模式
UltraLite.NET, 21
管理
UltraLite.NET 事务, 25
滚动
UltraLite.NET Table API, 20

H

HasRows 属性
ULDataReader 类 [UltraLite .NET API], 263
环境变量
命令 shell, xi
命令提示符, xi
回退
UltraLite.NET 事务, 25
获取帮助
技术支持, xii

I

iAnywhere.Data.UltraLite 命名空间
关于, 2
类 [UltraLite .NET API], 52
iAnywhere.Data.UltraLite 命名空间 (.NET 2.0)
iAnywhere.Data.UltraLite 命名空间, 2
iAnywhere 开发人员社区
新闻组, xiii
IgnoredRows 属性
ULSyncResult 类 [UltraLite .NET API], 507
IndexColumns 属性
ULMetaDataCollectionNames 类 [UltraLite .NET API], 348
IndexCount 属性
ULTableSchema 类 [UltraLite .NET API], 535
Indexes 属性
ULMetaDataCollectionNames 类 [UltraLite .NET API], 349
IndexName 属性
ULCommand 类 [UltraLite .NET API], 96
IndexOf(Object) 方法
ULParameterCollection 类 [UltraLite .NET API], 386
IndexOf(String) 方法
ULParameterCollection 类 [UltraLite .NET API], 387
IndexOf 方法
ULBulkCopyColumnMappingCollection 类 [UltraLite .NET API], 82
IndexSchema 类
UltraLite.NET 开发, 26
InfoMessage 事件
ULConnection 类 [UltraLite .NET API], 174
InsertBegin 方法
ULTable 类 [UltraLite .NET API], 527
InsertCommand 属性
ULDataAdapter 类 [UltraLite .NET API], 232
Insert 方法
ULParameterCollection 类 [UltraLite .NET API], 387
ULTable 类 [UltraLite .NET API], 526
install-dir
文档用法, x
Instance 字段
ULFactory 类 [UltraLite .NET API], 308
Interactive SQL
从 Visual Studio 打开, 7
INVALID_DATABASE_ID 字段
ULConnection 类 [UltraLite .NET API], 137
IsBOF 属性
ULDataReader 类 [UltraLite .NET API], 263
IsCaseSensitive 属性
ULDatabaseSchema 类 [UltraLite .NET API], 248
IsClosed 属性
ULDataReader 类 [UltraLite .NET API], 264
IsColumnAutoIncrement 方法
ULTableSchema 类 [UltraLite .NET API], 542
IsColumnCurrentDate 方法
ULTableSchema 类 [UltraLite .NET API], 542
IsColumnCurrentTimestamp 方法
ULTableSchema 类 [UltraLite .NET API], 543
IsColumnCurrentTime 方法
ULTableSchema 类 [UltraLite .NET API], 543
IsColumnDescending 方法
ULIndexSchema 类 [UltraLite .NET API], 339
IsColumnGlobalAutoIncrement 方法
ULTableSchema 类 [UltraLite .NET API], 544
IsColumnNewUUID 方法

ULTableSchema 类 [UltraLite .NET API], 545
IsColumnNullable 方法
ULTableSchema 类 [UltraLite .NET API], 545
IsDBNull 方法
ULDataReader 类 [UltraLite .NET API], 291
IsEOF 属性
ULDataReader 类 [UltraLite .NET API], 264
IsFixedSize 属性
ULParameterCollection 类 [UltraLite .NET API], 373
IsForeignKeyCheckOnCommit 属性
ULIndexSchema 类 [UltraLite .NET API], 334
IsForeignKeyNullable 属性
ULIndexSchema 类 [UltraLite .NET API], 335
IsForeignKey 属性
ULIndexSchema 类 [UltraLite .NET API], 334
IsInPublication 方法
ULTableSchema 类 [UltraLite .NET API], 546
IsNeverSynchronized 属性
ULTableSchema 类 [UltraLite .NET API], 536
IsNullable 属性
ULParameter 类 [UltraLite .NET API], 363
IsolationLevel 属性
ULTransaction 类 [UltraLite .NET API], 549
IsOpen 属性
ULCursorSchema 类 [UltraLite .NET API], 219
ULDatabaseSchema 类 [UltraLite .NET API], 249
ULIndexSchema 类 [UltraLite .NET API], 335
ULPublicationSchema 类 [UltraLite .NET API], 393
IsPrimaryKey 属性
ULIndexSchema 类 [UltraLite .NET API], 336
IsReadOnly property
ULParameterCollection 类 [UltraLite .NET API], 373
IsSynchronized 属性
ULParameterCollection 类 [UltraLite .NET API], 374
IsUniqueIndex 属性
ULIndexSchema 类 [UltraLite .NET API], 336
IsUniqueKey 属性
ULIndexSchema 类 [UltraLite .NET API], 337
Item(Int32) 属性
ULDataReader 类 [UltraLite .NET API], 264
ULParameterCollection 类 [UltraLite .NET API], 374
Item(String) 属性
ULDataReader 类 [UltraLite .NET API], 265
ULParameterCollection 类 [UltraLite .NET API], 375
Item 属性
ULBulkCopyColumnMappingCollection 类 [UltraLite .NET API], 77
ULConnectionStringBuilder 类 [UltraLite .NET API], 197

J

技术支持
新闻组, xiii
加密
UltraLite.NET 开发, 15
教程
C# 在 UltraLite.NET 中, 31
Visual Basic 在 UltraLite.NET 中, 31
结果集
UltraLite.NET 浏览, 18
结果集模式
UltraLite.NET, 18

K

KeepPartialDownload 属性
ULSyncParms 类 [UltraLite .NET API], 482
开发
UltraLite.NET, 11
开发平台
UltraLite.NET, 3
开发人员社区
新闻组, xiii
口令
UltraLite.NET 验证于, 28

L

LastIdentity 属性
ULConnection 类 [UltraLite .NET API], 141
LookupBackward(Int16) 方法
ULTable 类 [UltraLite .NET API], 528
LookupBackward 方法
ULTable 类 [UltraLite .NET API], 527
LookupBegin 方法
ULTable 类 [UltraLite .NET API], 529
LookupForward(Int16) 方法
ULTable 类 [UltraLite .NET API], 530
LookupForward 方法
ULTable 类 [UltraLite .NET API], 529

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

联机手册
PDF, viii
连接
SQL Anywhere Explorer for UltraLite, 7
UltraLite.NET 教程, 37
UltraLite.NET 数据库, 13
了解 UltraLite.Net 开发
关于, 11
列
UltraLite.NET 值修改, 21
浏览
UltraLite.NET Table API, 20
浏览 SQL 结果集
UltraLite.NET, 18

M

MaxHashSize 属性
ULCreateParms 类 [UltraLite .NET API], 211
Message 属性
ULInfoMessageEventArgs 类 [UltraLite .NET API], 341
MetaDataCollections 属性
ULMetaDataCollectionNames 类 [UltraLite .NET API], 350
MoveAfterLast 方法
ULDataReader 类 [UltraLite .NET API], 291
MoveBeforeFirst 方法
ULDataReader 类 [UltraLite .NET API], 292
moveFirst 方法
UltraLite.NET 数据检索示例, 17
MoveFirst 方法
ULDataReader 类 [UltraLite .NET API], 292
moveFirst 方法（Table 类）
UltraLite.NET 开发, 20
MoveLast 方法
ULDataReader 类 [UltraLite .NET API], 292
moveNext 方法
UltraLite.NET 数据检索示例, 17
MoveNext 方法
ULDataReader 类 [UltraLite .NET API], 293
moveNext 方法（Table 类）
UltraLite.NET 开发, 20
MovePrevious 方法
ULDataReader 类 [UltraLite .NET API], 293
MoveRelative 方法
ULDataReader 类 [UltraLite .NET API], 294
命令 shell
大括号, xi
引号, xi
括号, xi
环境变量, xi
约定, xi
命令提示符
大括号, xi
引号, xi
括号, xi
环境变量, xi
约定, xi
模糊处理
UltraLite.NET 开发, 15
模式
UltraLite.NET, 21
UltraLite.NET 访问, 26
目标平台
UltraLite.NET, 3

N

Name 属性
ULCursorSchema 类 [UltraLite .NET API], 219
ULIndexSchema 类 [UltraLite .NET API], 337
ULPublicationSchema 类 [UltraLite .NET API], 393
ULResultSetSchema 类 [UltraLite .NET API], 423
ULTableSchema 类 [UltraLite .NET API], 536
NativeError 属性
ULException 类 [UltraLite .NET API], 304
ULInfoMessageEventArgs 类 [UltraLite .NET API], 342
NearestCentury 属性
ULCreateParms 类 [UltraLite .NET API], 211
NewPassword 属性
ULSyncParms 类 [UltraLite .NET API], 483
NextResult 方法
ULDataReader 类 [UltraLite .NET API], 294
NotifyAfter 属性
ULBulkCopy 类 [UltraLite .NET API], 63

O

Obfuscate 属性
ULCreateParms 类 [UltraLite .NET API], 212
Offset 属性
ULParameter 类 [UltraLite .NET API], 364
Open 方法
ULConnection 类 [UltraLite .NET API], 166

Options 窗口
SQL Anywhere Explorer for UltraLite, 8
OrderedTableScans 属性
ULConnectionStringBuilder 类 [UltraLite .NET API], 199

P

PageSize 属性
ULCreateParms 类 [UltraLite .NET API], 212
ParameterName 属性
ULParameter 类 [UltraLite .NET API], 364
Parameters 属性
ULCommand 类 [UltraLite .NET API], 97
PartialDownloadRetained 属性
ULSyncResult 类 [UltraLite .NET API], 507
Password 属性
ULConnectionParms 类 [UltraLite .NET API], 185
ULConnectionStringBuilder 类 [UltraLite .NET API], 199
ULFileTransfer 类 [UltraLite .NET API], 319
ULSyncParms 类 [UltraLite .NET API], 483
PDF
文档, viii
PingOnly 属性
ULSyncParms 类 [UltraLite .NET API], 484
Plan 属性
ULCommand 类 [UltraLite .NET API], 97
Precision 属性
ULCreateParms 类 [UltraLite .NET API], 213
ULParameter 类 [UltraLite .NET API], 365
Prepare 方法
ULCommand 类 [UltraLite .NET API], 120
PrimaryKey 属性
ULTableSchema 类 [UltraLite .NET API], 537
PublicationCount 属性
ULDatabaseSchema 类 [UltraLite .NET API], 250
PublicationSchema 类
UltraLite.NET 开发, 26
Publications 属性
ULMetaDataCollectionNames 类 [UltraLite .NET API], 350
ULSyncParms 类 [UltraLite .NET API], 485
配置 SQL Anywhere Explorer
UltraLite 关于, 8
偏移
UltraLite.NET 相对, 20
平台
UltraLite.NET 中支持的, 3

R

Read 方法
ULDataReader 类 [UltraLite .NET API], 295
ReceivedBytes 属性
ULSyncProgressData 类 [UltraLite .NET API], 493
ReceivedDeletes 属性
ULSyncProgressData 类 [UltraLite .NET API], 494
ReceivedInserts 属性
ULSyncProgressData 类 [UltraLite .NET API], 494
ReceivedUpdates 属性
ULSyncProgressData 类 [UltraLite .NET API], 494
RecordsAffected 属性
ULDataReader 类 [UltraLite .NET API], 266
ULRowUpdatedEventArgs 类 [UltraLite .NET API], 431
ReferencedIndexName 属性
ULIndexSchema 类 [UltraLite .NET API], 337
ReferencedTableName 属性
ULIndexSchema 类 [UltraLite .NET API], 338
RegisterForEvent 方法
ULConnection 类 [UltraLite .NET API], 167
RemoveAt(Int32) 方法
ULParameterCollection 类 [UltraLite .NET API], 389
RemoveAt(String) 方法
ULParameterCollection 类 [UltraLite .NET API], 389
RemoveAt 方法
ULBulkCopyColumnMappingCollection 类 [UltraLite .NET API], 84
Remove 方法
ULBulkCopyColumnMappingCollection 类 [UltraLite .NET API], 83
ULConnectionStringBuilder 类 [UltraLite .NET API], 203
ULParameterCollection 类 [UltraLite .NET API], 388
ReservedWords 属性
ULMetaDataCollectionNames 类 [UltraLite .NET API], 351
ReserveSize 属性

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULConnectionStringBuilder 类 [UltraLite .NET API], 200
ResetDbType 方法
ULParameter 类 [UltraLite .NET API], 369
ResetLastDownloadTime 方法
ULConnection 类 [UltraLite .NET API], 168
Restrictions 属性
ULMetaDataCollectionNames 类 [UltraLite .NET API], 351
ResumedAtSize 属性
ULFileTransferProgressData 类 [UltraLite .NET API], 329
ResumePartialDownload 属性
ULFileTransfer 类 [UltraLite .NET API], 320
ULSyncParms 类 [UltraLite .NET API], 485
RevokeConnectFrom 方法
ULConnection 类 [UltraLite .NET API], 168
revokeConnectionFrom 方法
UltraLite.NET 开发, 28
RollbackPartialDownload 方法
ULConnection 类 [UltraLite .NET API], 169
rollback 方法
UltraLite.NET 事务, 25
Rollback 方法
ULTransaction 类 [UltraLite .NET API], 550
RowCount 属性
ULDataReader 类 [UltraLite .NET API], 267
RowsCopied 属性
ULRowsCopiedEventArgs 类 [UltraLite .NET API], 425
RowUpdated 事件
ULDataAdapter 类 [UltraLite .NET API], 236
RowUpdating 事件
ULDataAdapter 类 [UltraLite .NET API], 237
RuntimeType 属性
ULDatabaseManager 类 [UltraLite .NET API], 239

S

samples-dir
文档用法, x
Scale 属性
ULCreateParms 类 [UltraLite .NET API], 213
ULParameter 类 [UltraLite .NET API], 365
Schema 属性
ULConnection 类 [UltraLite .NET API], 142
ULDataReader 类 [UltraLite .NET API], 267
ULTable 类 [UltraLite .NET API], 518
ScriptCount 属性
ULSqlProgressData 类 [UltraLite .NET API], 456
ScriptProgressed 方法
ULSqlPassthroughProgressListener 接口 [UltraLite .NET API], 453
SelectCommand 属性
ULDataAdapter 类 [UltraLite .NET API], 233
SELECT 语句
UltraLite.NET 示例, 17
SendColumnNames 属性
ULSyncParms 类 [UltraLite .NET API], 486
SendDownloadAck 属性
ULSyncParms 类 [UltraLite .NET API], 487
SendNotification 方法
ULConnection 类 [UltraLite .NET API], 169
SentBytes 属性
ULSyncProgressData 类 [UltraLite .NET API], 495
SentDeletes 属性
ULSyncProgressData 类 [UltraLite .NET API], 495
SentInserts 属性
ULSyncProgressData 类 [UltraLite .NET API], 496
SentUpdates 属性
ULSyncProgressData 类 [UltraLite .NET API], 496
ServerSyncInvoked 方法
ULServerSyncListener 接口 [UltraLite .NET API], 438
ServerVersion 属性
ULConnection 类 [UltraLite .NET API], 142
SetActiveSyncListener 方法
ULDatabaseManager 类 [UltraLite .NET API], 242
SetBoolean 方法
ULResultSet 类 [UltraLite .NET API], 404
SetBytes 方法
ULResultSet 类 [UltraLite .NET API], 405
SetByte 方法
ULResultSet 类 [UltraLite .NET API], 404
SetDatabaseOption 方法
ULDatabaseSchema 类 [UltraLite .NET API], 255
SetDateTime 方法
ULResultSet 类 [UltraLite .NET API], 407
SetDBNull 方法
ULResultSet 类 [UltraLite .NET API], 406
SetDecimal 方法

ULResultSet 类 [UltraLite .NET API], 408
SetDouble 方法
ULResultSet 类 [UltraLite .NET API], 409
SetFloat 方法
ULResultSet 类 [UltraLite .NET API], 410
SetGlobalListener 方法
ULDatabaseManager 类 [UltraLite .NET API], 243
SetGuid 方法
ULResultSet 类 [UltraLite .NET API], 411
SetInt16 方法
ULResultSet 类 [UltraLite .NET API], 412
SetInt32 方法
ULResultSet 类 [UltraLite .NET API], 413
SetInt64 方法
ULResultSet 类 [UltraLite .NET API], 414
SetServerSyncListener 方法
ULDatabaseManager 类 [UltraLite .NET API], 244
SetString 方法
ULResultSet 类 [UltraLite .NET API], 415
SetTimeSpan 方法
ULResultSet 类 [UltraLite .NET API], 416
SetToDefault 方法
ULResultSet 类 [UltraLite .NET API], 416
SetUInt16 方法
ULResultSet 类 [UltraLite .NET API], 417
SetUInt32 方法
ULResultSet 类 [UltraLite .NET API], 418
SetUInt64 方法
ULResultSet 类 [UltraLite .NET API], 419
SetValues 方法
ULSyncProgressData 类 [UltraLite .NET API], 499
SignalSyncIsComplete 方法
ULDatabaseManager 类 [UltraLite .NET API], 245
Size 属性
ULParameter 类 [UltraLite .NET API], 366
SourceColumnNullMapping 属性
ULParameter 类 [UltraLite .NET API], 367
SourceColumn 属性
ULBulkCopyColumnMapping 类 [UltraLite .NET API], 74
ULParameter 类 [UltraLite .NET API], 366
SourceOrdinal 属性
ULBulkCopyColumnMapping 类 [UltraLite .NET API], 75
SourceVersion 属性
ULParameter 类 [UltraLite .NET API], 367
Source 属性
ULException 类 [UltraLite .NET API], 304
ULInfoMessageEventArgs 类 [UltraLite .NET API], 342
SQL Anywhere
文档, viii
SQL Anywhere Explorer
UltraLite 限制, 12
UltraLite.NET 关于, 12
支持 UltraLite 的编程语言, 7
SQL Anywhere Explorer for UltraLite
Visual Studio 集成, 7
使用表, 9
关于, 5
添加数据库对象, 8
连接, 7
配置, 8
StartLine 属性
ULConnectionStringBuilder 类 [UltraLite .NET API], 201
StartSynchronizationDelete 方法
ULConnection 类 [UltraLite .NET API], 170
StateChange 事件
ULConnection 类 [UltraLite .NET API], 175
State 属性
ULConnection 类 [UltraLite .NET API], 143
ULSqlProgressData 类 [UltraLite .NET API], 456
ULSyncProgressData 类 [UltraLite .NET API], 497
StopSynchronizationDelete 方法
ULConnection 类 [UltraLite .NET API], 171
StreamErrorCode 属性
ULFileTransfer 类 [UltraLite .NET API], 321
ULSyncResult 类 [UltraLite .NET API], 508
StreamErrorParameters 属性
ULSyncResult 类 [UltraLite .NET API], 508
StreamErrorSystem 属性
ULFileTransfer 类 [UltraLite .NET API], 321
ULSyncResult 类 [UltraLite .NET API], 509
StreamParms 属性
ULFileTransfer 类 [UltraLite .NET API], 322
ULSyncParms 类 [UltraLite .NET API], 488
Stream 属性
ULFileTransfer 类 [UltraLite .NET API], 320
ULSyncParms 类 [UltraLite .NET API], 487
Sybase Central
从 Visual Studio 打开, 7

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

SYNC_ALL_DB 字段
ULPublicationSchema 类 [UltraLite .NET API], 392
SYNC_ALL_PUBS 字段
ULPublicationSchema 类 [UltraLite .NET API], 392
Synchronize(ULSyncProgressListener) 方法
ULConnection 类 [UltraLite .NET API], 172
Synchronize 方法
ULConnection 类 [UltraLite .NET API], 171
SyncParms 属性
ULConnection 类 [UltraLite .NET API], 143
SyncProgressed 方法
ULSyncProgressListener 接口 [UltraLite .NET API], 501
SyncResult 属性
ULConnection 类 [UltraLite .NET API], 144
SyncRoot 属性
ULParameterCollection 类 [UltraLite .NET API], 376
SyncTableCount 属性
ULSyncProgressData 类 [UltraLite .NET API], 497
SyncTableIndex 属性
ULSyncProgressData 类 [UltraLite .NET API], 498
删除
UltraLite.NET 表行, 24
事务
UltraLite.NET 管理, 25
事务处理
UltraLite.NET 管理, 25
数据操作
UltraLite.NET Table API, 20
UltraLite.NET 使用 SQL, 16
数据库模式
UltraLite.NET 访问, 26
数据类型
UltraLite.NET API 访问于, 21
UltraLite.NET 转换, 22
搜索
UltraLite 行使用 UltraLite.NET, 22
索引
UltraLite.NET 模式信息于, 26
锁定
UltraLite.NET 关键字, 30

T

Table API
UltraLite.NET 简介, 20
TableCount 属性
ULDatabaseSchema 类 [UltraLite .NET API], 250
TableID 属性
ULSyncProgressData 类 [UltraLite .NET API], 498
TableMappings 属性
ULDataAdapter 类 [UltraLite .NET API], 234
TableName 属性
ULSyncProgressData 类 [UltraLite .NET API], 499
TableSchema 类
UltraLite.NET 开发, 26
Tables 属性
ULMetaDataCollectionNames 类 [UltraLite .NET API], 352
TimeFormat 属性
ULCreateParms 类 [UltraLite .NET API], 214
TimestampFormat 属性
ULCreateParms 类 [UltraLite .NET API], 214
TimestampIncrement 属性
ULCreateParms 类 [UltraLite .NET API], 215
Timestamp 属性
ULSyncResult 类 [UltraLite .NET API], 509
ToString 方法
ULConnectionParms 类 [UltraLite .NET API], 186
ULCreateParms 类 [UltraLite .NET API], 216
ULInfoMessageEventArgs 类 [UltraLite .NET API], 343
ULParameter 类 [UltraLite .NET API], 369
Transaction 属性
ULCommand 类 [UltraLite .NET API], 98
TriggerEvent 方法
ULConnection 类 [UltraLite .NET API], 173
Truncate 方法
ULTable 类 [UltraLite .NET API], 531
TryGetValue 方法
ULConnectionStringBuilder 类 [UltraLite .NET API], 204
提交
UltraLite.NET 事务, 25
体系结构
UltraLite.Net, 4
同步

UltraLite.NET, 29
UltraLite.NET C# 示例, 29
UltraLite.NET 中的 ActiveSync, 29
图标
此帮助文档中使用的, xi

U

ULActiveSyncListener 接口 [UltraLite .NET API]
ActiveSyncInvoked 方法, 53
说明, 53
ULAuthStatusCode 枚举 [UltraLite .NET API]
说明, 56
ULBulkCopy(String, ULBulkCopyOptions) 构造函数
ULBulkCopy 类 [UltraLite .NET API], 60
ULBulkCopy(String) 构造函数
ULBulkCopy 类 [UltraLite .NET API], 59
ULBulkCopy(ULConnection, ULBulkCopyOptions, ULTransaction) 构造函数
ULBulkCopy 类 [UltraLite .NET API], 60
ULBulkCopy(ULConnection) 构造函数
ULBulkCopy 类 [UltraLite .NET API], 58
ULBulkCopyColumnMapping(Int32, Int32) 构造函数
ULBulkCopyColumnMapping 类 [UltraLite .NET API], 70
ULBulkCopyColumnMapping(Int32, String) 构造函数
ULBulkCopyColumnMapping 类 [UltraLite .NET API], 71
ULBulkCopyColumnMapping(String, Int32) 构造函数
ULBulkCopyColumnMapping 类 [UltraLite .NET API], 71
ULBulkCopyColumnMapping(String, String) 构造函数
ULBulkCopyColumnMapping 类 [UltraLite .NET API], 72
ULBulkCopyColumnMappingCollection 类 [UltraLite .NET API]
Add(Int32, Int32) 方法, 78
Add(Int32, String) 方法, 79
Add(String, Int32) 方法, 80
Add(String, String) 方法, 81
Add(ULBulkCopyColumnMapping) 方法, 78
Contains 方法, 81
CopyTo 方法, 82
IndexOf 方法, 82
Item 属性, 77
Remove 方法, 83
RemoveAt 方法, 84
说明, 76
ULBulkCopyColumnMapping 构造函数
ULBulkCopyColumnMapping 类 [UltraLite .NET API], 70
ULBulkCopyColumnMapping 类 [UltraLite .NET API]
DestinationColumn 属性, 73
DestinationOrdinal 属性, 73
SourceColumn 属性, 74
SourceOrdinal 属性, 75
ULBulkCopyColumnMapping 构造函数, 70
ULBulkCopyColumnMapping(Int32, Int32) 构造函数, 70
ULBulkCopyColumnMapping(Int32, String) 构造函数, 71
ULBulkCopyColumnMapping(String, Int32) 构造函数, 71
ULBulkCopyColumnMapping(String, String) 构造函数, 72
说明, 69
ULBulkCopyOptions 枚举 [UltraLite .NET API]
description, 85
ULBulkCopy 类 [UltraLite .NET API]
BatchSize 属性, 61
BulkCopyTimeout 属性, 62
Close 方法, 64
ColumnMappings 属性, 62
DestinationTableName 属性, 63
Dispose 方法, 64
NotifyAfter 属性, 63
ULBulkCopy(String) 构造函数, 59
ULBulkCopy(String, ULBulkCopyOptions) 构造函数, 60
ULBulkCopy(ULConnection) 构造函数, 58
ULBulkCopy(ULConnection, ULBulkCopyOptions, ULTransaction) 构造函数, 60
ULRowsCopied 事件, 67
WriteToServer(DataRow[]) 方法, 65
WriteToServer(DataTable) 方法, 65
WriteToServer(DataTable, DataRowState) 方法, 66
WriteToServer(IDataReader) 方法, 66
说明, 57
ULCommand(String, ULConnection, ULTransaction) 构造函数
ULCommand 类 [UltraLite .NET API], 92

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ULCommand(String, ULConnection) 构造函数
ULCommand 类 [UltraLite .NET API], 91
ULCommand(String) 构造函数
ULCommand 类 [UltraLite .NET API], 90
ULCommandBuilder(ULDataAdapter) 构造函数
ULCommandBuilder 类 [UltraLite .NET API], 124
ULCommandBuilder 构造函数
ULCommandBuilder 类 [UltraLite .NET API], 124
ULCommandBuilder 类 [UltraLite .NET API]
DataAdapter 属性, 125
description, 121
GetDeleteCommand 方法, 126
GetDeleteCommand(Boolean) 方法, 125
GetInsertCommand 方法, 128
GetInsertCommand(Boolean) 方法, 127
GetUpdateCommand 方法, 129
GetUpdateCommand(Boolean) 方法, 129
ULCommandBuilder 构造函数, 124
ULCommandBuilder(ULDataAdapter) 构造函数, 124
ULCommand 构造函数
ULCommand 类 [UltraLite .NET API], 89
ULCommand 类
UltraLite.NET 数据操作示例, 16
UltraLite.NET 数据检索示例, 17
ULCommand 类 [UltraLite .NET API]
BeginExecuteNonQuery 方法, 99
BeginExecuteNonQuery(AsyncCallback, Object) 方法, 99
BeginExecuteReader 方法, 100
BeginExecuteReader(AsyncCallback, Object) 方法, 102
BeginExecuteReader(AsyncCallback, Object, CommandBehavior) 方法, 103
BeginExecuteReader(CommandBehavior) 方法, 101
Cancel 方法, 104
CommandText 属性, 93
CommandTimeout 属性, 94
CommandType 属性, 94
Connection 属性, 95
CreateParameter 方法, 104
description, 86
DesignTimeVisible 属性, 96
EndExecuteNonQuery 方法, 105
EndExecuteReader 方法, 108
ExecuteNonQuery 方法, 111
ExecuteReader 方法, 112
ExecuteReader(CommandBehavior) 方法, 113
ExecuteResultSet 方法, 115
ExecuteResultSet(CommandBehavior) 方法, 116
ExecuteScalar 方法, 117
ExecuteTable 方法, 118
ExecuteTable(CommandBehavior) 方法, 119
IndexName 属性, 96
Parameters 属性, 97
Plan 属性, 97
Prepare 方法, 120
Transaction 属性, 98
ULCommand 构造函数, 89
ULCommand(String) 构造函数, 90
ULCommand(String, ULConnection) 构造函数, 91
ULCommand(String, ULConnection, ULTransaction) 构造函数, 92
UpdatedRowSource 属性, 99
ULConnection(String) 构造函数
ULConnection 类 [UltraLite .NET API], 136
ULConnectionParms.UnusedEventHandler 委派 [UltraLite .NET API]
说明, 188
ULConnectionParms 构造函数
ULConnectionParms 类 [UltraLite .NET API], 180
ULConnectionParms 类 [UltraLite .NET API]
AdditionalParms 属性, 180
CacheSize 属性, 182
ConnectionName 属性, 183
DatabaseOnCE 属性, 184
DatabaseOnDesktop 属性, 184
EncryptionKey 属性, 185
Password 属性, 185
ToString 方法, 186
ULConnectionParms 构造函数, 180
UnusedEvent 事件, 187
UserID 属性, 186
说明, 177
ULConnectionStringBuilder(String) 构造函数
ULConnectionStringBuilder 类 [UltraLite .NET API], 193
ULConnectionStringBuilder 构造函数
ULConnectionStringBuilder 类 [UltraLite .NET API], 193
ULConnectionStringBuilder 类 [UltraLite .NET API]
CacheSize 属性, 194

ConnectionName 属性, 195
ContainsKey 方法, 202
DatabaseKey 属性, 195
DatabaseName 属性, 196
DatabaseOnCE 属性, 196
DatabaseOnDesktop 属性, 197
EquivalentTo 方法, 202
GetShortName 方法, 203
OrderedTableScans 属性, 199
Password 属性, 199
Remove 方法, 203
ReserveSize 属性, 200
StartLine 属性, 201
TryGetValue 方法, 204
ULConnectionStringBuilder 构造函数, 193
ULConnectionStringBuilder(String) 构造函数, 193
UserID 属性, 201
事件, 197
说明, 189
ULConnection 构造函数
ULConnection 类 [UltraLite .NET API], 135
ULConnection 类 [UltraLite .NET API]
BeginTransaction 方法, 144
BeginTransaction(IsolationLevel) 方法, 145
CancelGetNotification 方法, 146
ChangeDatabase 方法, 147
ChangeEncryptionKey 方法, 148
ChangePassword 方法, 148
Close 方法, 149
ConnectionString 属性, 137
ConnectionTimeout 属性, 138
CountUploadRows(String, Int64) 方法, 150
CountUploadRows(String, UInt32) 方法, 149
CreateCommand 方法, 151
CreateNotificationQueue 方法, 152
Database 属性, 139
DatabaseID 属性, 140
DatabaseManager 属性, 140
DataSource 属性, 139
DeclareEvent 方法, 152
DestroyNotificationQueue 方法, 153
ExecuteNextSQLPassthroughScript 方法, 154
ExecuteSQLPassthroughScripts 方法, 155
ExecuteTable(String) 方法, 155
ExecuteTable(String, String) 方法, 156
ExecuteTable(String, String, CommandBehavior) 方法, 158
GetLastDownloadTime 方法, 159
GetNewUUID 方法, 160
GetNotification 方法, 161
GetNotificationParameter 方法, 162
GetSchema 方法, 163
GetSchema(String) 方法, 163
GetSchema(String, String[]) 方法, 164
GetSQLPassthroughScriptCount 方法, 162
GlobalAutoIncrementUsage 属性, 141
GrantConnectTo 方法, 166
InfoMessage 事件, 174
INVALID_DATABASE_ID 字段, 137
LastIdentity 属性, 141
Open 方法, 166
RegisterForEvent 方法, 167
ResetLastDownloadTime 方法, 168
RevokeConnectFrom 方法, 168
RollbackPartialDownload 方法, 169
Schema 属性, 142
SendNotification 方法, 169
ServerVersion 属性, 142
StartSynchronizationDelete 方法, 170
State 属性, 143
StateChange 事件, 175
StopSynchronizationDelete 方法, 171
Synchronize 方法, 171
Synchronize(ULSyncProgressListener) 方法, 172
SyncParms 属性, 143
SyncResult 属性, 144
TriggerEvent 方法, 173
ULConnection 构造函数, 135
ULConnection(String) 构造函数, 136
ValidateDatabase 方法, 173
说明, 131
ULCreateParms 构造函数
ULCreateParms 类 [UltraLite .NET API], 208
ULCreateParms 类 [UltraLite .NET API]
CaseSensitive 属性, 209
ChecksumLevel 属性, 209
DateFormat 属性, 210
DateOrder 属性, 210
FIPS 属性, 210
MaxHashSize 属性, 211
NearestCentury 属性, 211
Obfuscate 属性, 212

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

PageSize 属性, 212
Precision 属性, 213
Scale 属性, 213
TimeFormat 属性, 214
TimestampFormat 属性, 214
TimestampIncrement 属性, 215
ToString 方法, 216
ULCreateParms 构造函数, 208
UTF8Encoding 属性, 215
说明, 206
ULCursorSchema 类 [UltraLite .NET API]
ColumnCount 属性, 218
GetColumnID 方法, 219
GetColumnName 方法, 220
GetColumnPrecision 方法, 221
GetColumnScale 方法, 223
GetColumnSize 方法, 223
GetColumnSQLName 方法, 222
GetColumnULDbType 方法, 224
GetSchemaTable 方法, 224
IsOpen 属性, 219
Name 属性, 219
说明, 217
ULDataAdapter(String, String) 构造函数
ULDataAdapter 类 [UltraLite .NET API], 231
ULDataAdapter(String, ULConnection) 构造函数
ULDataAdapter 类 [UltraLite .NET API], 230
ULDataAdapter(ULCommand) 构造函数
ULDataAdapter 类 [UltraLite .NET API], 229
ULDataAdapter 构造函数
ULDataAdapter 类 [UltraLite .NET API], 229
ULDataAdapter 类 [UltraLite .NET API]
DeleteCommand 属性, 231
GetFillParameters 方法, 235
InsertCommand 属性, 232
RowUpdated 事件, 236
RowUpdating 事件, 237
SelectCommand 属性, 233
TableMappings 属性, 234
ULDataAdapter 构造函数, 229
ULDataAdapter(String, String) 构造函数, 231
ULDataAdapter(String, ULConnection) 构造函数, 230
ULDataAdapter(ULCommand) 构造函数, 229
UpdateCommand 属性, 235
说明, 226
ULDatabaseManager 类 [UltraLite .NET API]
CreateDatabase 方法, 240
DropDatabase 方法, 241
RuntimeType 属性, 239
SetActiveSyncListener 方法, 242
SetGlobalListener 方法, 243
SetServerSyncListener 方法, 244
SignalSyncIsComplete 方法, 245
ValidateDatabase 方法, 245
说明, 238
ULDatabaseSchema 类
iAnywhere.Data.UltraLite 命名空间, 26
ULDatabaseSchema 类 [UltraLite .NET API]
CollationName 属性, 248
GetDatabaseProperty 方法, 251
GetPublicationName 方法, 253
GetPublicationSchema 方法, 253
GetTableName 方法, 254
IsCaseSensitive 属性, 248
IsOpen 属性, 249
PublicationCount 属性, 250
SetDatabaseOption 方法, 255
TableCount 属性, 250
说明, 247
ULDataReader 类
UltraLite.NET 数据检索示例, 17
ULDataReader 类 [UltraLite .NET API]
Close 方法, 268
Depth 属性, 262
FieldCount 属性, 262
GetBoolean 方法, 268
GetByte 方法, 269
GetBytes(Int32) 方法, 271
GetBytes(Int32, Int64, Byte[], Int32, Int32) 方法, 270
GetChar 方法, 272
GetChars 方法, 273
GetDataTypeName 方法, 274
GetDateTime 方法, 275
GetDecimal 方法, 275
GetDouble 方法, 276
GetEnumerator 方法, 277
GetFieldType 方法, 277
GetFloat 方法, 278
GetGuid 方法, 279
GetInt16 方法, 279
GetInt32 方法, 280
GetInt64 方法, 281

GetName 方法, 281
GetOrdinal 方法, 282
GetRowCount 方法, 283
GetSchemaTable 方法, 284
GetString 方法, 286
GetTimeSpan 方法, 286
GetUInt16 方法, 287
GetUInt32 方法, 288
GetUInt64 方法, 288
GetValue 方法, 289
GetValues 方法, 290
HasRows 属性, 263
IsBOF 属性, 263
IsClosed 属性, 264
IsDBNull 方法, 291
IsEOF 属性, 264
Item(Int32) 属性, 264
Item(String) 属性, 265
MoveAfterLast 方法, 291
MoveBeforeFirst 方法, 292
MoveFirst 方法, 292
MoveLast 方法, 292
MoveNext 方法, 293
MovePrevious 方法, 293
MoveRelative 方法, 294
NextResult 方法, 294
Read 方法, 295
RecordsAffected 属性, 266
RowCount 属性, 267
Schema 属性, 267
说明, 257
ULDateOrder 枚举 [UltraLite .NET API]
说明, 296
ULDbType 枚举 [UltraLite .NET API]
说明, 297
ULDbType 属性
ULParameter 类 [UltraLite .NET API], 368
ULDBValid 枚举 [UltraLite .NET API]
说明, 301
ULException 构造函数
ULException 类 [UltraLite .NET API], 303
ULException 类 [UltraLite .NET API]
GetObjectData 方法, 305
NativeError 属性, 304
Source 属性, 304
ULException 构造函数, 303
说明, 302
ULFactory 类 [UltraLite .NET API]
CanCreateDataSourceEnumerator 方法, 308
CreateCommand 方法, 309
CreateCommandBuilder 方法, 309
CreateConnection 方法, 310
CreateConnectionStringBuilder 方法, 310
CreateDataAdapter 方法, 310
CreateParameter 方法, 311
Instance 字段, 308
说明, 306
ULFileTransferProgressData 类 [UltraLite .NET API]
BytesReceived 属性, 328
FileSize 属性, 328
FLAG_IS_BLOCKING 字段, 328
Flags 属性, 329
ResumedAtSize 属性, 329
说明, 327
ULFileTransferProgressListener 接口
[UltraLite .NET API]
FileTransferProgressed 方法, 330
说明, 330
ULFileTransfer 构造函数
ULFileTransfer 类 [UltraLite .NET API], 314
ULFileTransfer 类 [UltraLite .NET API]
AuthenticationParms 属性, 315
AuthStatus 属性, 315
AuthValue 属性, 315
DestinationFileName 属性, 316
DestinationPath 属性, 316
DownloadedFile 属性, 317
DownloadFile 方法, 324
DownloadFile(ULFileTransferProgressListener) 方法, 325
FileAuthCode 属性, 317
FileName 属性, 318
ForceDownload 属性, 318
Password 属性, 319
ResumePartialDownload 属性, 320
Stream 属性, 320
StreamErrorCode 属性, 321
StreamErrorSystem 属性, 321
StreamParms 属性, 322
ULFileTransfer 构造函数, 314
UserName 属性, 323
Version 属性, 323
说明, 312
ULIndexSchema 类 [UltraLite .NET API]

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

ColumnCount 属性, 333
GetColumnName 方法, 338
IsColumnDescending 方法, 339
IsForeignKey 属性, 334
IsForeignKeyCheckOnCommit 属性, 334
IsForeignKeyNullable 属性, 335
IsOpen 属性, 335
IsPrimaryKey 属性, 336
IsUniqueIndex 属性, 336
IsUniqueKey 属性, 337
Name 属性, 337
ReferencedIndexName 属性, 337
ReferencedTableName 属性, 338
说明, 332
ULInfoMessageEventArgs 类 [UltraLite .NET API]
Message 属性, 341
NativeError 属性, 342
Source 属性, 342
ToString 方法, 343
说明, 341
ULInfoMessageEventHandler 委派 [UltraLite .NET API]
说明, 344
ULMetaDataCollectionNames 类 [UltraLite .NET API]
Columns 属性, 346
DataSourceInformation 属性, 347
DataTypes 属性, 347
ForeignKeys 属性, 348
IndexColumns 属性, 348
Indexes 属性, 349
MetaDataCollections 属性, 350
Publications 属性, 350
ReservedWords 属性, 351
Restrictions 属性, 351
Tables 属性, 352
说明, 345
ULParameter(String, Object) 构造函数
ULParameter 类 [UltraLite .NET API], 357
ULParameter(String, ULDbType, Int32, ParameterDirection, Boolean, Byte, Byte, String, DataRowVersion, Object) 构造函数
ULParameter 类 [UltraLite .NET API], 360
ULParameter(String, ULDbType, Int32, String) 构造函数
ULParameter 类 [UltraLite .NET API], 359
ULParameter(String, ULDbType, Int32) 构造函数
ULParameter 类 [UltraLite .NET API], 359
ULParameter(String, ULDbType) 构造函数
ULParameter 类 [UltraLite .NET API], 358
ULParameterCollection 类 [UltraLite .NET API]
Add(Object) 方法, 376
Add(String, Object) 方法, 378
Add(String, ULDbType) 方法, 379
Add(String, ULDbType, Int32) 方法, 380
Add(String, ULDbType, Int32, String) 方法, 381
Add(ULParameter) 方法, 377
AddRange(Array) 方法, 382
AddRange(ULParameter[]) 方法, 383
Clear 方法, 383
Contains(Object) 方法, 384
Contains(String) 方法, 384
CopyTo 方法, 385
Count 属性, 373
GetEnumerator 方法, 386
IndexOf(Object) 方法, 386
IndexOf(String) 方法, 387
Insert 方法, 387
IsFixedSize 属性, 373
IsReadOnly property, 373
IsSynchronized 属性, 374
Item(Int32) 属性, 374
Item(String) 属性, 375
Remove 方法, 388
RemoveAt(Int32) 方法, 389
RemoveAt(String) 方法, 389
SyncRoot 方法, 376
说明, 371
ULParameter 构造函数
ULParameter 类 [UltraLite .NET API], 356
ULParameter 类 [UltraLite .NET API]
DbType 属性, 362
Direction 属性, 362
IsNullable 属性, 363
Offset 属性, 364
ParameterName 属性, 364
Precision 属性, 365
ResetDbType 方法, 369
Scale 属性, 365
Size 属性, 366
SourceColumn 属性, 366
SourceColumnNullMapping 属性, 367
SourceVersion 属性, 367
ToString 方法, 369

ULDbType 属性, 368
ULParameter 构造函数, 356
ULParameter(String, Object) 构造函数, 357
ULParameter(String, ULDbType) 构造函数, 358
ULParameter(String, ULDbType, Int32) 构造函数, 359
ULParameter(String, ULDbType, Int32, ParameterDirection, Boolean, Byte, Byte, String, DataRowVersion, Object) 构造函数, 360
ULParameter(String, ULDbType, Int32, String) 构造函数, 359
Value 属性, 368
说明, 354
ULPublicationSchema 类 [UltraLite .NET API]
IsOpen 属性, 393
Name 属性, 393
SYNC_ALL_DB 字段, 392
SYNC_ALL_PUBS 字段, 392
说明, 391
ULResultSetSchema 类 [UltraLite .NET API]
Name 属性, 423
说明, 421
ULResultSet 类 [UltraLite .NET API]
AppendBytes 方法, 400
AppendChars 方法, 402
Delete 方法, 403
SetBoolean 方法, 404
SetByte 方法, 404
SetBytes 方法, 405
SetDateTime 方法, 407
SetDBNull 方法, 406
SetDecimal 方法, 408
SetDouble 方法, 409
SetFloat 方法, 410
SetGuid 方法, 411
SetInt16 方法, 412
SetInt32 方法, 413
SetInt64 方法, 414
SetString 方法, 415
SetTimeSpan 方法, 416
SetToDefault 方法, 416
SetUInt16 方法, 417
SetUInt32 方法, 418
SetUInt64 方法, 419
Update 方法, 420
说明, 394
ULRowsCopiedEventArgs 构造函数
ULRowsCopiedEventArgs 类 [UltraLite .NET API], 424
ULRowsCopiedEventArgs 类 [UltraLite .NET API]
Abort 属性, 425
RowsCopied 属性, 425
ULRowsCopiedEventArgs 构造函数, 424
说明, 424
ULRowsCopiedEventHandler 委派 [UltraLite .NET API]
说明, 427
ULRowsCopied 事件
ULBulkCopy 类 [UltraLite .NET API], 67
ULRowUpdatedEventArgs 构造函数
ULRowUpdatedEventArgs 类 [UltraLite .NET API], 429
ULRowUpdatedEventArgs 类 [UltraLite .NET API]
Command 属性, 430
RecordsAffected 属性, 431
ULRowUpdatedEventArgs 构造函数, 429
说明, 428
ULRowUpdatedEventHandler 委派 [UltraLite .NET API]
说明, 432
ULRowUpdatingEventArgs 构造函数
ULRowUpdatingEventArgs 类 [UltraLite .NET API], 434
ULRowUpdatingEventArgs 类 [UltraLite .NET API]
Command 属性, 435
ULRowUpdatingEventArgs 构造函数, 434
说明, 433
ULRowUpdatingEventHandler 委派 [UltraLite .NET API]
说明, 436
ULRuntimeType 枚举 [UltraLite .NET API]
说明, 437
ULServerSyncListener 接口 [UltraLite .NET API]
ServerSyncInvoked 方法, 438
说明, 438
ULServerSyncListener 枚举 [UltraLite .NET API]
说明, 441
ULSqlPassthroughProgressListener 接口 [UltraLite .NET API]
ScriptProgressed 方法, 453
说明, 453
ULSqlProgressData 类 [UltraLite .NET API]
CurrentScript 属性, 455
ScriptCount 属性, 456

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

State 属性, 456
说明, 455
ULSqlProgressState 枚举 [UltraLite .NET API]
说明, 457
ULStreamErrorCode 枚举 [UltraLite .NET API]
说明, 458
ULStreamType 枚举 [UltraLite .NET API]
说明, 476
ULSyncParms 类 [UltraLite .NET API]
AdditionalParms 属性, 480
AuthenticationParms 属性, 480
CopyFrom 方法, 490
DownloadOnly 属性, 481
KeepPartialDownload 属性, 482
NewPassword 属性, 483
Password 属性, 483
PingOnly 属性, 484
Publications 属性, 485
ResumePartialDownload 属性, 485
SendColumnNames 属性, 486
SendDownloadAck 属性, 487
Stream 属性, 487
StreamParms 属性, 488
UploadOnly 属性, 488
UserName 属性, 489
Version 属性, 490
说明, 478
ULSyncProgressData 类 [UltraLite .NET API]
FLAG_IS_BLOCKING 字段, 492
Flags 属性, 493
ReceivedBytes 属性, 493
ReceivedDeletes 属性, 494
ReceivedInserts 属性, 494
ReceivedUpdates 属性, 494
SentBytes 属性, 495
SentDeletes 属性, 495
SentInserts 属性, 496
SentUpdates 属性, 496
SetValues 方法, 499
State 属性, 497
SyncTableCount 属性, 497
SyncTableIndex 属性, 498
TableID 属性, 498
TableName 属性, 499
说明, 491
ULSyncProgressListener 接口 [UltraLite .NET API]
SyncProgressed 方法, 501
说明, 501
ULSyncProgressState 枚举 [UltraLite .NET API]
说明, 503
ULSyncResult 类 [UltraLite .NET API]
AuthStatus 属性, 506
AuthValue 属性, 506
IgnoredRows 属性, 507
PartialDownloadRetained 属性, 507
StreamErrorCode 属性, 508
StreamErrorParameters 属性, 508
StreamErrorSystem 属性, 509
Timestamp 属性, 509
UploadOK 属性, 510
说明, 505
ULTableSchema 类 [UltraLite .NET API]
GetColumnDefaultValue 方法, 538
GetColumnPartitionSize 方法, 538
GetIndex 方法, 539
GetIndexName 方法, 540
GetOptimalIndex 方法, 540
GetPublicationPredicate 方法, 541
IndexCount 属性, 535
IsColumnAutoIncrement 方法, 542
IsColumnCurrentDate 方法, 542
IsColumnCurrentTime 方法, 543
IsColumnCurrentTimestamp 方法, 543
IsColumnGlobalAutoIncrement 方法, 544
IsColumnNewUUID 方法, 545
IsColumnNullable 方法, 545
IsInPublication 方法, 546
IsNeverSynchronized 属性, 536
Name 属性, 536
PrimaryKey 属性, 537
UploadUnchangedRows 属性, 537
说明, 533
ULTable 类 [UltraLite .NET API]
DeleteAllRows 方法, 519
FindBegin 方法, 519
FindFirst 方法, 520
FindFirst(Int16) 方法, 521
FindLast 方法, 522
FindLast(Int16) 方法, 522
FindNext 方法, 523
FindNext(Int16) 方法, 524
FindPrevious 方法, 525
FindPrevious(Int16) 方法, 526
Insert 方法, 526

InsertBegin 方法, 527
LookupBackward 方法, 527
LookupBackward(Int16) 方法, 528
LookupBegin 方法, 529
LookupForward 方法, 529
LookupForward(Int16) 方法, 530
Schema 属性, 518
Truncate 方法, 531
UpdateBegin 方法, 531
说明, 511
UltraLite
Visual Studio 集成, 32
UltraLite.NET
activesync 同步, 29
SQL Anywhere Explorer for UltraLite, 5
Table API 简介, 20
事务处理, 25
优点, 2
体系结构, 4
使用 SQL 操作数据, 16
关于, 1
加密, 15
动态 SQL 教程, 39
同步示例, 29
应用程序中的同步, 29
开发, 11
支持的平台, 3
教程, 31
数据操作, 16
数据检索, 17
用户验证, 28
访问模式信息, 26
部署, 43
错误处理, 27
UltraLite .NET API
ULActiveSyncListener 接口, 53
ULAuthStatusCode 枚举, 56
ULBulkCopy 类, 57
ULBulkCopyColumnMapping 类, 69
ULBulkCopyColumnMappingCollection 类, 76
ULBulkCopyOptions 枚举, 85
ULCommand 类, 86
ULCommandBuilder 类, 121
ULConnection 类, 131
ULConnectionParms 类, 177
ULConnectionParms.UnusedEventHandler 委派, 188
ULConnectionStringBuilder 类, 189
ULCreateParms 类, 206
ULCursorSchema 类, 217
ULDataAdapter 类, 226
ULDatabaseManager 类, 238
ULDatabaseSchema 类, 247
ULDataReader 类, 257
ULDateOrder 枚举, 296
ULDbType 枚举, 297
ULDBValid 枚举, 301
ULException 类, 302
ULFactory 类, 306
ULFileTransfer 类, 312
ULFileTransferProgressData 类, 327
ULFileTransferProgressListener 接口, 330
ULIndexSchema 类, 332
ULInfoMessageEventArgs 类, 341
ULInfoMessageEventHandler 委派, 344
ULMetaDataCollectionNames 类, 345
ULParameter 类, 354
ULParameterCollection 类, 371
ULPublicationSchema 类, 391
ULResultSet 类, 394
ULResultSetSchema 类, 421
ULRowsCopiedEventArgs 类, 424
ULRowsCopiedEventHandler 委派, 427
ULRowUpdatedEventArgs 类, 428
ULRowUpdatedEventHandler 委派, 432
ULRowUpdatingEventArgs 类, 433
ULRowUpdatingEventHandler 委派, 436
ULRuntimeType 枚举, 437
ULServerSyncListener 接口, 438
ULSQLCode 枚举, 441
ULSqlPassthroughProgressListener 接口, 453
ULSqlProgressData 类, 455
ULSqlProgressState 枚举, 457
ULStreamErrorCode 枚举, 458
ULStreamType 枚举, 476
ULSyncParms 类, 478
ULSyncProgressData 类, 491
ULSyncProgressListener 接口, 501
ULSyncProgressState 枚举, 503
ULSyncResult 类, 505
ULTable 类, 511
ULTableSchema 类, 533
ULTransaction 类, 547
UltraLite.NET 教程

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1

C# 教程的代码列表, 45
Visual Basic 教程的代码列表, 48
UltraLite 模式
UltraLite.NET, 21
UltraLite 数据库
UltraLite.NET 信息访问, 26
UltraLite.NET 连接, 13
ULTransaction 类 [UltraLite .NET API]
Commit 方法, 549
Connection 属性, 548
IsolationLevel 属性, 549
Rollback 方法, 550
说明, 547
UnusedEvent 事件
ULConnectionParms 类 [UltraLite .NET API], 187
UpdateBegin 方法
ULTable 类 [UltraLite .NET API], 531
UpdateCommand 属性
ULDataAdapter 类 [UltraLite .NET API], 235
UpdatedRowSource 属性
ULCommand 类 [UltraLite .NET API], 99
Update 方法
ULResultSet 类 [UltraLite .NET API], 420
UploadOK 属性
ULSyncResult 类 [UltraLite .NET API], 510
UploadOnly 属性
ULSyncParms 类 [UltraLite .NET API], 488
UploadUnchangedRows 属性
ULTableSchema 类 [UltraLite .NET API], 537
UserID 属性
ULConnectionParms 类 [UltraLite .NET API], 186
ULConnectionStringBuilder 类 [UltraLite .NET API], 201
UserName 属性
ULFileTransfer 类 [UltraLite .NET API], 323
ULSyncParms 类 [UltraLite .NET API], 489
UTF8Encoding 属性
ULCreateParms 类 [UltraLite .NET API], 215

V

ValidateDatabase 方法
ULConnection 类 [UltraLite .NET API], 173
ULDatabaseManager 类 [UltraLite .NET API], 245
Value 属性
ULParameter 类 [UltraLite .NET API], 368
Version 属性
ULFileTransfer 类 [UltraLite .NET API], 323
ULSyncParms 类 [UltraLite .NET API], 490
Visual Studio
SQL Anywhere Explorer 与 UltraLite 的集成, 7
UltraLite 数据库连接, 7
与 UltraLite 的集成, 32
访问 UltraLite 数据库, 7
Visual Studio .NET
UltraLite.NET 连接到 UltraLite 数据库, 12

W

Windows Mobile
UltraLite.NET 目标平台, 3
WriteToServer(DataRow[]) 方法
ULBulkCopy 类 [UltraLite .NET API], 65
WriteToServer(DataTable, DataRowState) 方法
ULBulkCopy 类 [UltraLite .NET API], 66
WriteToServer(DataTable) 方法
ULBulkCopy 类 [UltraLite .NET API], 65
WriteToServer(IDataReader) 方法
ULBulkCopy 类 [UltraLite .NET API], 66
文档
SQL Anywhere, viii
约定, ix

X

线程
UltraLite.NET 多线程应用程序, 13
相对偏移
UltraLite.NET Table API, 20
新闻组
技术支持, xiii
行
UltraLite 删除使用 UltraLite.NET, 24
UltraLite 插入使用 UltraLite.NET, 23
UltraLite 更新使用 UltraLite.NET, 23
UltraLite.NET 当前的表访问, 21
UltraLite.NET 表浏览, 20
选择
UltraLite.NET 行, 17

Y

验证用户
UltraLite.NET 开发, 28
疑难解答
UltraLite.NET 处理错误, 27
UltraLite.NET 部署项目清单, 43
新闻组, xiii

用户验证
  UltraLite.NET 开发, 28
优点
  UltraLite.NET, 2
约定
  命令 shell, xi
  命令提示符, xi
  文档, ix
  文档中的文件名, x

Z

支持
  新闻组, xiii
支持的平台
  UltraLite.NET, 3
值
  UltraLite.NET 访问于, 21
主题
  图标, xi
转换
  UltraLite.NET 数据类型于, 22

 Copyright © 2009, iAnywhere Solutions, Inc. - SQL Anywhere 11.0.1