SYBASE iAnywhere

MobiLink
服务器启动的同步

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

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

关于本手册

主题

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。

目标读者

本手册适用于对设置服务器启动的同步感兴趣的那些 MobiLink 用户。

开始之前

有关 MobiLink 的详细信息，请参见 MobiLink - 入门。

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`。

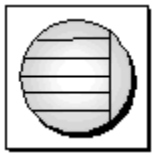
- **环境变量** 本文档介绍设置环境变量。在 Windows shell 中，环境变量使用语法 *%ENVVAR%* 来指定。在 Unix shell 中，环境变量使用语法 *$ENVVAR* 或 *${ENVVAR}* 来指定。

图标

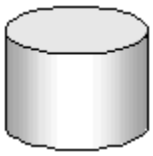
本文档中使用了下列图标。

- 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

服务器启动的同步简介

目录

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

服务器启动的同步的作用是什么

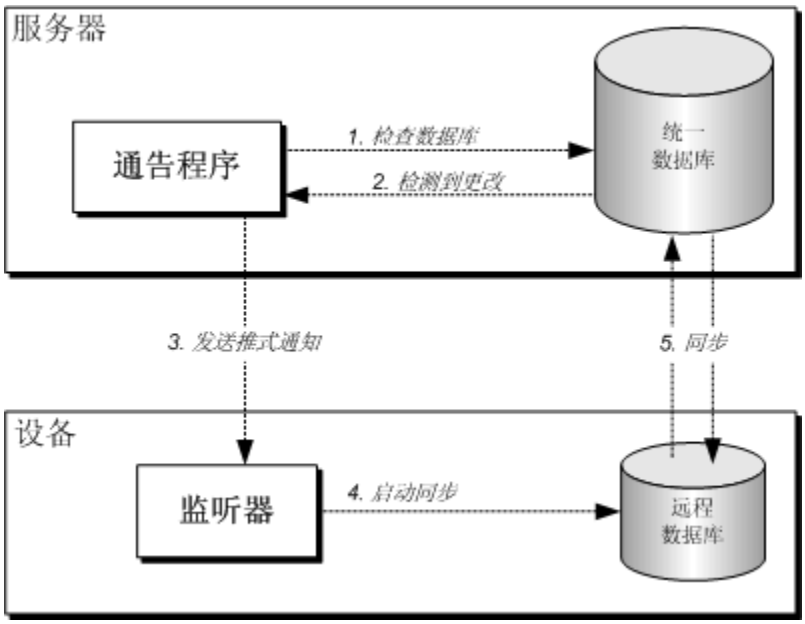
可通过 MobiLink 服务器启动的同步从统一数据库启动同步。可以向远程数据库发送推式通知，使远程数据库可以更新统一数据库。此 MobiLink 组件提供了可编程的选项，用于检测统一数据库中的更改以启动同步、选择要被发送推式通知的设备，以及确定设备如何响应这些推式通知。

示例

某货运组织向其司机分发移动设备。每个设备运行一个包含路线和交货地点的数据库。当某司机提交交通中断的通知时，该报告将发送到统一数据库。一个名为通告程序的服务器端 MobiLink 组件检测到该报告，并向路线受到交通中断影响的其他司机发送推式通知。此推式通知可使远程数据库同步，以便司机可以使用替代路线。

服务器启动的同步过程

在下图中，通告程序检查统一数据库中是否有更改。通告程序向设备发送推式通知，导致远程数据库与统一数据库同步。



在服务器启动的同步过程中，将发生以下步骤：

1. 通告程序使用基于业务逻辑的查询检查统一数据库中是否有需要与远程数据库同步的更改。
2. 检测到更改时，通告程序将准备推式通知，以发送到该设备。
3. 通告程序发送推式通知。可以通过设备跟踪器网关、UDP 网关、SMTP 网关 或 SYNC 网关发送推式通知。
4. 监听器对照消息过滤器比较主题、内容或发送者。

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

5. 如果符合过滤器条件，则启动操作。例如，在典型实现中，某操作可以运行 MobiLink 客户端或启动 UltraLite 应用程序。

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

服务器启动的同步组件

MobiLink 服务器启动的同步需要以下组成部分：

- **推式请求** 推式请求是结果集中的一行值，用于告知通告程序您想要向设备发送推式通知。推式请求会导致发生服务器启动的同步。任何数据库应用程序都可创建推式请求，包括通告程序。例如，可以使用价格变化时激活的数据库触发器创建推式请求。请参见“推式请求”一节第 8 页。
- **MobiLink 通告程序** 通告程序是集成到 MobiLink 服务器的程序。它经常检查统一数据库是否有推式请求。可以通过指定通告程序的属性控制通告程序检查推式请求的频率。必须指定用于检查推式请求和确定通知哪些设备的业务逻辑。通告程序检测到推式请求时将向设备发送推式通知。请参见“通告程序”一节第 13 页。
- **MobiLink 监听器** 监听器是在设备上运行的程序。它从通告程序接收推式通知，然后使用消息处理程序过滤消息并启动一个操作。在典型应用程序中，各操作是同步调用，但应用程序还能够执行其它操作。可以配置监听器对来自选定服务器源的推式通知或包含特定内容的推式通知做出不同反应。

  在 Windows 设备上，监听器是使用命令行选项配置的可执行程序。要接收推式通知，设备必须打开，监听器必须正在运行。请参见“用于 Windows 设备的监听器实用程序”一节第 56 页。

  要将 MobiLink 监听器用于 Palm 设备，可以通过在 Windows 桌面上运行监听器配置实用程序创建一个配置文件、将该文件复制到设备，然后运行监听器。请参见“用于 Palm 设备的实用程序”一节第 76 页。
- **轻量级轮询器** 轻量级轮询器是按指定的时间间隔轮询推式通知的设备应用程序。使用轻量级轮询器是设置网关的替代方法，也是建议的方法，因为此方法不需要与服务器保持持久性连接，并且有助于延长电池寿命。

  监听器是可以使用监听器命令行选项配置的轻量级轮询器。此外，也可以使用轻量级轮询 API 来创建自己的轻量级轮询器。

  请参见：

  - “设置轻量级轮询选项”一节第 18 页
  - “轻量级轮询 API”一节第 21 页
- **网关（轻量级轮询器的替代方法）** 网关提供一个通告程序接口，用于向设备发送推式通知。网关是轻量级轮询器的替代方法。可以使用设备跟踪网关、SYNC 网关、UDP 网关或 SMTP 网关发送消息。请参见“网关和运营公司”一节第 27 页。

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

服务器启动的同步部署注意事项

在部署服务器启动的同步应用程序之前，请考虑以下几个问题。

使用 UDP 网关时的设备限制

- 必须能够直接从 MobiLink 服务器对设备上的 IP 地址寻址。
- 如果不能直接从 MobiLink 服务器对 Windows 设备上的 IP 地址寻址，则对 UDP 通知的 IP 跟踪不起作用。

设备跟踪限制

用于 Palm 设备和 Adaptive Server Anywhere 9.0.0 的监听器或更早版本的监听器不支持设备跟踪。要将设备跟踪与这些监听器配合使用，必须手动设置设备跟踪。请参见“添加设备跟踪支持”一节 第 28 页。

轻量级轮询限制

用于 Palm 设备的监听器不支持轻量级轮询。

支持的设备平台

只有在 Windows、Windows Mobile 和 Palm OS 设备上才支持 MobiLink 监听器。

服务器启动的同步快速入门指南

在完成此过程之前，必须设置 MobiLink 执行常规同步。请参见 MobiLink - 入门。

1. 在 MobiLink 服务器上，准备统一数据库以存储推式请求。请参见“推式请求要求”一节 第 8 页。
2. 在 MobiLink 服务器上，配置通告程序事件以创建和管理推式请求。请参见“配置通告程序事件和属性”一节第 14 页。
3. 在设备上，设置轻量级轮询器。请参见“轻量级轮询器”一节第 21 页。

   如果不想使用轻量级轮询器，则在 MobiLink 服务器上设置支持的网关。使用 SMTP 网关时，还需要配置运营公司。请参见“网关和运营公司”一节第 27 页。
4. 在设备上，设置用于过滤消息和执行操作的监听器。请参见“消息处理程序”一节第 15 页。

有关如何使用 Sybase Central 设置服务器启动的同步的说明，请参见“在 [模型] 模式中设置服务器启动的同步”一节 《MobiLink - 入门》。

其它资源

- 示例应用程序安装在 *samples-dir**MobiLink*\ 目录中。（有关 *samples-dir* 的详细信息，请参见“示例目录”一节 《SQL Anywhere 服务器 - 数据库管理》。）所有与服务器启动的同步相关的应用程序都位于带有 **SIS_** 前缀的目录中。
- 请参见“查找附加信息和资源”一节 《使用本帮助》。

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

设置服务器启动的同步

目录

推式请求

推式请求是结果集中的一行值，通告程序通过检查推式请求来确定是否需要向设备发送推式通知。通告程序将推式请求放在推式通知里，然后发送推式通知。在典型的服务器启动的同步设置中，推式请求包含消息内容和目标设备信息。在发送推式通知前，需要先配置通告程序事件，使通告程序可以检测推式请求。

推式请求要求

推式请求的要求取决于 MobiLink 服务器与设备通信所使用的方法。所有推式请求都要求有主题列和内容列。如果使用轻量级轮询器轮询推式通知，则需要创建轮询键列来标识它们。如果使用网关发送推式通知，则需要创建网关列和地址列。

如果系统中已存在推式请求列，则不需要再创建。满足推式请求要求后，便可以使用推式请求。请参见“使用推式请求”一节第 9 页。

使用轻量级轮询器（建议）时的推式请求要求

使用轻量级轮询器轮询推式通知时必须创建以下列：

列	类型	说明
轮询键	VARCHAR	用于标识轻量级轮询器的键。每个轻量级轮询器都发送一个唯一键，用于在 MobiLink 服务器上标识自己。
主题	VARCHAR	消息的主题行。
内容	VARCHAR	消息的内容。

使用网关时的推式请求要求

除非另有规定，否则使用网关发送推式通知时必须创建以下列：

列	类型	说明
请求 ID	INTEGER	可选。推式请求的唯一 ID。 某些通告程序事件需要此列名称。请参见“通告程序事件”一节第 37 页。
网关	VARCHAR	作为消息发送目的地的网关的名称。
主题	VARCHAR	消息的主题行。
内容	VARCHAR	消息的内容。
地址	VARCHAR	设备的目标地址。

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

列	类型	说明
重发间隔	VARCHAR	可选。消息重发之间的时间间隔。 在不可靠的网络上使用 UDP 网关时，重发间隔很有用。通告程序假定，与推式请求关联的所有属性都不变；第一次轮询请求后将忽略后续更新。如果推式通知必须在下一次轮询时间前发送，则通告程序会自动调整下一次轮询间隔。可以使用 request_cursor 事件中的同步逻辑阻止发送推式请求。来自目标监听器的传送确认可能会阻止随后的重发。请参见“request_cursor 事件”一节第 40 页。
生存期	VARCHAR	可选。距重发到期的时间。

示例

以下示例通过在 SQL Anywhere 统一数据库表中创建必需的列来满足使用轻量级轮询的推式请求要求：

```
CREATE TABLE PushRequest (
    req_id INTEGER DEFAULT AUTOINCREMENT PRIMARY KEY,
    poll_key VARCHAR(128),
    subject VARCHAR(128),
    content VARCHAR(128)
)
```

您只需创建此表或类似项，前提是推式请求列在其它地方不存在。这些列可跨多个表存在，可存在于现有表中，也可存在于视图中。

另请参见

- “配置通告程序事件和属性”一节第 14 页
- “request_cursor 事件”一节第 40 页

使用推式请求

生成推式请求

生成推式请求的前提条件是，统一数据库中必须包含服务器启动的同步所需的推式请求列，并且必须能够通过单个数据库查询获得这些值。在 request_cursor 事件中提供用于选择推式请求列的数据库查询时自动生成推式请求。有关推式请求要求的详细信息，请参见“推式请求要求”一节第 8 页。

示例

统一数据库中包含名为 PushRequest 的表，该表包含轻量级轮询器的推式请求。远程设备在 MobiLink 服务器上标识为 unique_device_ID。服务器使用以下 SQL 脚本请求同步：

```
INSERT INTO PushRequest (poll_key, subject, content) VALUES
('unique_device_ID', 'synchronize', 'ASAP');
```

使用此脚本插入值不足以生成推式请求。必须在 request_cursor 事件中使用数据库查询选择值。为生成推式请求，服务器使用以下 request_cursor 事件脚本：

```
SELECT poll_key, subject, content FROM PushRequest;
```

unique_device_ID 设备在服务器中轮询推式通知时生成推式请求。推式通知主题是 **synchronize**，内容是 **ASAP**。

推式请求限制

下表列出了每个列的推式请求限制：

列	类型	限制
请求 ID	INTEGER	此值必须为唯一主键。
轮询键	VARCHAR	仅在使用轻量级轮询器时需要。 对轮询键没有限制。
网关	VARCHAR	仅在使用网关时需要。 此值必须设置为已启用网关的名称。可以指定自己的自定义网关名称，也可以选择以下预配置的网关名称之一： ● **Default-DeviceTracker** ● **Default-SMTP** ● **Default-SYNC** ● **Default-UDP** 请参见“将网关用作轻量级轮询器的替代方法”一节第 27 页。
主题	VARCHAR	设置此值时避免使用非字母数字字符。大括号、尖括号、双引号、圆括号、单引号和方括号专供内部使用，不能在主题列中使用。
内容	VARCHAR	对消息内容没有限制。
地址	VARCHAR	仅在使用网关时需要。 对于 UDP 网关，此值应为 IP 地址或主机名。支持以下格式的端口号后缀： ● *IP-address*:*port-number* ● *hostname*:*port-number* 对于 SMTP 网关，此值应为电子邮件地址。 对于 SYNC 网关和设备跟踪网关，此值应是使用监听器 -t+ 选项定义的接收者名称。请参见“-t 选项”一节第 66 页。

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

列	类型	限制
重发间隔	VARCHAR	缺省情况下，此值按分钟计。可以指定 **S**、**M** 和 **H**，分别代表以秒、分钟和小时为单位。还可以将各单位组合使用；例如 `1H 30M 10S` 告诉通告程序每隔 1 小时 30 分钟 10 秒重发一次消息。 如果此值为空值或未指定，则缺省情况下，仅发送一次，不重发。
生存期	VARCHAR	缺省情况下，此值按分钟计。可以指定 **S**、**M** 和 **H**，分别代表以秒、分钟和小时为单位。还可以将各单位组合使用；例如，`3H 30M 10S` 告诉通告程序在初始发送后的 3 小时 30 分钟 10 秒时停止重新发送消息。 如果此值为空值或未指定，则缺省情况下，仅发送一次，不重发。

检测推式请求和发送推式通知

通告程序通过频繁触发 request_cursor 事件检测推式请求。缺省情况下，不为此事件指定脚本；您必须提供 request_cursor 事件脚本，以使通告程序能够检测推式请求。在典型应用程序中，request_cursor 事件脚本是 SELECT 语句。请参见“request_cursor 事件”一节第 40 页。

以下示例使用 ml_add_property 系统过程为名为 Simple 的自定义通告程序创建 request_cursor 事件脚本。SELECT 语句告诉通告程序从名为 PushRequest 的表检测推式请求。

```
CALL ml_add_property('SIS', 'Notifier(Simple)', 'request_cursor',
    'SELECT poll_key, subject, content FROM PushRequest'
);
```

注意
列的选择顺序必须与推式请求中列的指定顺序相同。请参见“推式请求要求”一节第 8 页。

有关设置通告程序事件的详细信息，请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

删除推式请求

如果被通知设备的相关信息在被发送并按照业务规则被满足之后从未更新，则通告程序会重发推式通知。满足推式请求后，您需要阻止通告程序检测旧的推式请求。如果发送推式通知是为了进行同步，可以使用同步脚本删除这些推式请求。

可以使用 request_delete 事件按推式请求的请求 ID 删除推式请求，但推式请求中必须包含请求 ID 列，且必须启用传送确认。

注意
传送确认在 Palm 设备上不可用；但可以通过实施自己的传送确认机制删除推式请求。例如，使用自己的同步逻辑，可以在发生特定同步时删除推式请求。有关在 Windows 设备上禁用传送确认的详细信息，请参见“用于 Windows 的监听器关键字”一节第 68 页。

请参见：

- “推式请求要求”一节第 8 页
- “request_delete 事件”一节第 41 页
- “用于服务器启动的同步的 MobiLink 服务器设置”第 31 页

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

通告程序

通告程序是集成到 MobiLink 服务器的程序，它经常检查统一数据库是否有推式请求。一旦检测到推式请求，它便向设备发送推式通知。通告程序还执行一系列事件，允许您创建用于监控数据、管理推式请求、处理传送确认和处理错误的脚本。

第一次加载 MobiLink 服务器时启动通告程序。MobiLink 服务器的单个实例中可运行多个通告程序。有关如何使用多个通告程序的示例，请参见位于 *samples-dir\MobiLink\SIS_MultipleNotifier* 中的示例应用程序。有关 *samples-dir* 的详细信息，请参见“示例目录”一节 《SQL Anywhere 服务器 - 数据库管理》。

如果通告程序丢失数据库连接，它将尝试恢复连接，直到重新获得访问权限。恢复后，通告程序会以相同的配置设置继续运行。

MobiLink 服务器群中的通告程序

在 MobiLink 11.0 和更早版本中，MobiLink 服务器群中的服务器启动的同步可造成多余的推式通知，从而导致更多同步并增加 MobiLink 服务器群中统一数据库的负荷。现在可以在服务器群中的每个 MobiLink 服务器上运行一个通告程序；这些通告程序可共同确保不会向同一个监听器发出多余的推式通知。当其它服务器想要连接到本地 MobiLink 服务器时，可使用 mlsrv11 -lsc 服务器选项将信息传递给它们。请参见“-lsc 选项”一节 《MobiLink - 服务器管理》。

此功能使某一通告程序成为主通告程序，使所有其它通告程序成为辅助通告程序。主通告程序直接或间接地通过辅助通告程序来控制推式通知。辅助通告程序还会将监听器信息路由到主通告程序，所以主通告程序知道监听器的位置以及如何到达这些监听器。

如果运行主通告程序的 MobiLink 服务器出现故障，服务器群将选择一个新的主通告程序，通知将会继续。

监听器可以连接到群中的任何 MobiLink 服务器，而无需知道哪个服务器是主服务器。

要使用此功能，在群中的所有 MobiLink 服务器上都需要以下 mlsrv11 命令行选项：

- “-lsc 选项”一节 《MobiLink - 服务器管理》
- “-notifier 选项”一节 《MobiLink - 服务器管理》
- “-zs 选项”一节 《MobiLink - 服务器管理》

示例

在 host001 上：

```
mlsrv11 -notifier -zs ml001 -lsc tcpip(host=host001;port=2439) ...
```

在 host007 上：

```
mlsrv11 -notifier -zs ml007 -lsc tcpip(host=host007;port=2439) ...
```

配置通告程序事件和属性

通告程序事件允许您嵌入用于管理服务器启动的整个同步过程的脚本。有关通告程序事件序列以及如何触发事件的详细信息，请参见“通告程序事件”一节第 37 页。

例如，可以配置通告程序事件执行以下任务：

- 使用 request_cursor 事件确定推式请求中要发送的信息内容、发送方式和发送目的地。
- 使用 begin_poll 事件创建对统一数据库中的更改作出响应的推式请求。（高级用法）
- 使用 request_delete 事件删除推式请求。（如有必要）
- 使用 end_poll 事件跟踪通告程序轮询和清理表数据。（高级用法）

通告程序属性与事件类似。事件管理通知过程，属性管理通告程序行为。例如，通告程序属性确定通告程序轮询统一数据库的频率以及启动时是否启用通告程序。通告程序属性和事件将作为服务器端设置来配置。有关详细信息，请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

启动通告程序

MobiLink 服务器加载时将启动所有启用的通告程序。要禁用通告程序，必须将通告程序属性值 enable 设置为 false。请参见“通告程序属性”一节第 48 页。

要启动通告程序，使用以下方法之一：

- 配置通告程序，运行 mlsrv11 并指定 -notifier 选项。
- 如果您的设置存储在通告程序配置文件中，则通过在命令行运行 mlsrv11 加载数据库，并使用 -notifier 选项指定该文件。例如，如果想要使用名为 *myfirst.Notifier* 的文件，则以下命令配置 MobiLink 服务器使用该文件中指定的属性和事件：

  ```
  mlsrv11 ... -notifier c:\myfirst.Notifier
  ```

  如果使用 QAnywhere，则通过在命令行运行 mlsrv11 并指定 -m 选项加载数据库。

另请参见

- “-notifier 选项”一节 《MobiLink - 服务器管理》
- “用于服务器启动的同步的 MobiLink 服务器设置”第 31 页
- “配置通告程序事件和属性”一节第 14 页
- “-m 选项”一节 《MobiLink - 服务器管理》

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

监听器

监听器是在设备上运行的程序。它从通告程序接收推式通知，然后启动操作。使用网关进行服务器启动的同步时，监听器可以将设备跟踪信息上载到统一数据库。

另请参见

- “设备跟踪网关”一节第 27 页
- “消息处理程序”一节第 15 页
- “用于 Windows 设备的监听器实用程序”一节第 56 页
- “用于 Palm 设备的监听器配置实用程序”一节第 76 页

示例

以下命令启动 Windows 设备的 MobiLink 监听器实用程序：

```
dblsn -v2 -m -ot dblsn.log
    -l "poll_connect='host=localhost';
    poll_key=sis_user1;
    poll_every=10;
    subject=sync;
    action='start dbmlsync.exe
        -c eng=rem1;uid=DBA;pwd=sql
        -ot dbmlsyncOut.txt -qc';"
```

此命令加载监听器并将详细程度级别设置为 2，启用消息记录，并指定服务器位于 **localhost**。在将输出写入 *dblsn.log* 文件之前，将截断该文件。监听器每隔 10 秒轮询一次推式通知。如果监听器接收到主题名为 **sync** 的推式通知，则将启动 MobiLink 客户端应用程序。

有关 Windows 监听器命令行选项的详细信息，请参见“用于 Windows 的监听器选项”一节第 57 页。

消息处理程序

消息处理程序是一个监听器组件，它扫描推式通知的消息内容，以启动操作。它还可用于指定轻量级轮询选项，如服务器位置和轮询频率。

消息处理程序包含以下组成部分：

- **过滤器关键字** 预处理推式通知之后，可以使用过滤器关键字扫描消息内容。如果满足过滤器条件，则启动某操作。例如，可以指定 **subject** 关键字以过滤包含特定主题的消息，也可以指定 **sender** 关键字以过滤从特定 MobiLink 服务器接收的消息。
- **操作** 满足消息的过滤器条件后将启动某操作。在典型应用程序中，指定某操作以启动同步，但也可以执行其它操作。要协助错误处理，可以指定替代操作，用于在原始操作失败时处理实例。
- **轮询设置** 轮询设置允许您配置监听器如何轮询 MobiLink 服务器是否有推式通知。
- **选项** 各选项允许您控制远程设置，如传送和操作确认。

> **注意**
> 在 Palm 设备的 MobiLink 监听器实用程序中，某些消息处理程序选项不可用。请参见“用于 Palm 设备的监听器配置实用程序”一节第 76 页。

可以使用 dblsn -l 选项创建消息处理程序。可以指定多个消息处理程序。

另请参见

- “-l 选项”一节第 62 页
- “用于 Windows 的监听器关键字”一节第 68 页
- “用于 Windows 的监听器操作命令”一节第 70 页

使用消息处理程序

监听器收到推式通知时，它将提取消息，消息被分割，并分成几个关键字。**message** 关键字包含原始格式的整条消息。然后将消息分成 **subject**、**content** 和 **sender** 关键字。这些关键字经过消息过滤器，以确定启动哪些操作。有关使用这些关键字过滤消息的详细信息，请参见“过滤消息”一节第 16 页。

过滤消息

过滤器关键字用于将推式通知的一部分与用户定义的短语比较。如果两个短语原文相同，则启动某操作。有关预处理推式通知以进行消息过滤的详细信息，请参见“消息语法”一节第 114 页。

可以通过使用以下语法运行监听器来指定过滤器关键字：

```
dblsn ... -l "filter-keyword-name='content to filter';action='...'"
```

可以多次使用 -l 选项创建多个过滤器，但还必须为每个 -l 实例指定一个操作。只有满足所有过滤器时才启动操作。

在一个消息处理程序中，以下每个关键字只能出现一次：

- **content** 建议使用此关键字和 subject 关键字过滤消息。此关键字用于根据消息内容过滤消息。例如：

  ```
  dblsn -l "content='your content filter here';action='...'"
  ```

- **subject** 建议使用此关键字和 content 关键字过滤消息。此关键字用于根据消息主题过滤消息。例如：

  ```
  dblsn -l "subject='your subject filter here';action='...'"
  ```

- **message** 此关键字用于根据消息原始数据过滤消息。过滤器值必须与消息的精确长度相符。不建议使用此关键字，因为它有可变结构。有关预处理推式通知以进行消息过滤的详细信息，请参见“消息语法”一节第 114 页。

- **message_start** 此关键字用于根据消息原始数据的一部分过滤消息（从头开始比较）。有关预处理推式通知以进行消息过滤的详细信息，请参见“消息语法”一节第 114 页。

  指定此关键字时，监听器创建 $message_start 和 $message_end 操作变量。

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

- **sender** 此关键字用于根据消息发送者过滤消息。此关键字对于跟踪特定通告程序发送的推式通知很有用。该值取决于使用的网关。对于 UDP 网关，它是网关主机的 IP 地址。对于 SYNC 网关，它是 **MobiLink**。对于 SMTP 网关，它取决于无线运营公司。请参见“网关和运营公司”一节第 27 页。

另请参见

- “用于 Windows 的监听器关键字”一节第 68 页
- “操作变量”一节第 17 页

启动操作

消息与过滤器条件匹配时，启动某操作。有关过滤推式通知的详细信息，请参见“过滤消息”一节第 16 页。

可以通过使用以下语法运行监听器指定操作：

```
dblsn ... -l "...;action='action-command command statement'"
```

过滤某消息时，以下操作命令允许您执行不同任务：

- **START** 启动某应用程序，并允许它在后台运行。
- **RUN** 运行某应用程序，并等待它完成，然后再接收更多推式通知。
- **POST** 将窗口消息发布到已经在运行的进程上。此命令只能在 Windows 设备上使用。
- **SOCKET** 使用 TCP/IP 连接向某应用程序发送消息。
- **DBLSN FULL SHUTDOWN** 关闭监听器。

有关操作命令的详细列表和语法参考信息，请参见“用于 Windows 的监听器操作命令”一节第 70 页。

有关将操作变量合并到操作中的详细信息，请参见“操作变量”一节第 17 页。

操作变量

操作变量允许您从消息过滤器或操作引用推式通知的各部分。请参见“启动操作”一节第 17 页和“过滤消息”一节第 16 页。

如何设置操作变量

每次接收推式通知时都自动设置大部分操作变量。变量名称与在消息语法中指定的名称类似。例如，*message* 设置 $message 操作变量，*subject* 设置 $subject，*sender* 设置 $sender，*content* 设置 $content。请参见“消息语法”一节第 114 页。

使用操作变量

运行监听器时在命令行中使用操作变量。如何使用操作变量取决于消息处理程序以及想要启动的操作。以下示例演示如何使用 RUN 操作命令，此命令用于启动 MobiLink 客户端应用程序：

```
dblsn ... -l "subject=publish;action='RUN dbmlsync.exe @dbmlsync.txt -n
$content'"
```

此消息处理程序将过滤主题文字与 "publish" 相同的消息。过滤后，使用 -n 选项运行 dbmlsync，将 $content 操作变量作为参数传递。假定 *content* 引用同步发布的名称，dbmlsync 使用此发布将设备数据库与统一数据库同步。

以下示例演示如何使用操作变量过滤消息：

```
dblsn ... -l "subject=$content;action='RUN script.bat"
```

当 **subject** 文字与 **content** 相同时，此消息处理程序过滤各消息。过滤后，设备运行自定义批处理脚本。

另请参见

- “用于 Windows 的监听器操作命令”一节第 70 页
- “用于 Windows 的监听器操作变量”一节第 73 页
- “按远程 ID 过滤消息”一节第 19 页

设置轻量级轮询选项

可以使用消息处理程序处理轮询操作。轻量级轮询选项允许您指定服务器位置、通告程序名称、轮询频率和轮询键。此外，也可以使用轻量级轮询 API 指定这些属性。

可以通过使用以下语法运行监听器来指定轻量级轮询选项：

```
dblsn ... -l
    "poll_connect=protocol-options;
    poll_notifier=Notifier-name;
    poll_key=identifier-string;
    poll_every=number-of-seconds;..."
```

一个消息处理程序只能包含以下其中一个选项：

- **poll_connect** 此选项用于指定连接到服务器所需的协议选项。此外，也可以使用 dblsn -x 选项指定缺省协议选项。poll_connect 选项覆盖消息处理程序的缺省协议选项。
- **poll_notifier** 此选项用于指定 MobiLink 服务器处理推式请求所使用的通告程序。此选项为必需项，因为 MobiLink 服务器可以托管多个通告程序。
- **poll_key** 此选项用于向通告程序标识监听器。MobiLink 服务器使用此值发送计划用于该设备的推式通知。在典型应用程序中，此值应为设备的远程 ID。
- **poll_every** 此选项用于指定监听器轮询通告程序的频率。缺省情况下，监听器自动从 MobiLink 服务器检索此值。此值按秒计量。

另请参见

- “推式请求要求”一节第 8 页
- “配置通告程序事件和属性”一节第 14 页
- “轻量级轮询器”一节第 21 页
- “用于 Windows 的监听器关键字”一节第 68 页
- “轻量级轮询 API”一节第 21 页

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

高级消息处理程序功能

按远程 ID 过滤消息

可以使用 dblsn -r 选项和 $remote_id 操作变量按远程 ID 过滤消息。

第一次同步 SQL Anywhere 远程数据库时，将创建包含数据库 ID 的远程 ID 文件。此文件采用与数据库相同的名称（但具有 *.rid* 扩展名），并与数据库存储在同一目录中。对于 UltraLite 数据库，没有远程 ID 文件；远程 ID 直接从数据库提取。

启动监听器时，使用 dblsn -r 选项提供远程 ID 文件或 UltraLite 数据库的名称和位置，然后使用 dblsn -l 选项创建消息处理程序。

可以将远程 ID 直接键入消息过滤器。但在缺省情况下，远程 ID 是 GUID；除非提供一个有意义的名称，否则远程 ID 不容易记住。

注意
在 dblsn 命令行中，可以指定 -r 和 -l 选项的多个实例。在 -l 选项中使用的 $remote_id 操作变量始终在前面的 -r 选项中指定。所以，在 -l 选项之前指定 -r 选项非常重要。

以下示例演示如何使用多个远程 ID。它假定设备上有一个名为 *business.db* 的 SQL Anywhere 数据库和一个名为 *personal.udb* 的 UltraLite 数据库。在此示例中，**ulpersonal** 是 UltraLite 应用程序的窗口类名。

```
dblsn ... -r "c:\app\db\business.rid"
    -l "subject=$remote_id;action='dbmlsync.exe -k -c dsn=business';"
    -r "c:\ulapp\personal.udb"
    -l "subject=$remote_id;action=post dbas_synchronize to ulpersonal;"
```

另请参见

- “操作变量”一节第 17 页
- “远程 ID”一节 《MobiLink - 客户端管理》
- “-r 选项”一节第 65 页
- “用于 Windows 的监听器操作变量”一节第 73 页

连接启动的同步

在 Windows 设备上，连接变化时可以启动同步。

获得或丢失 IP 连接时，设备向监听器发送包含消息 **_IP_CHANGED_** 的推式通知。设备找到 MobiLink 服务器的新的最佳路径时，将向监听器发送包含消息 **_BEST_IP_CHANGED_** 的推式通知。使用消息处理程序，可以检测这些连接变化并启动操作。

标识连接的任何更改

_IP_CHANGED_ 消息指示 IP 连接发生变化。当设备在 WiFi 网络范围内、用户进行 RAS 连接或用户将设备放在底座上时，IP 连接通常会发生变化。可以通过使用以下语法运行监听器引用 **_IP_CHANGED_** 消息：

```
dblsn ... -l "message=_IP_CHANGED_;action='...'"
```

以下示例演示如何使用 **_IP_CHANGED_** 消息。消息处理程序过滤该消息，并将其发送到服务器。如果连接中断，就会生成错误。

```
dblsn -l "message=_IP_CHANGED_;
    action='
        SOCKET port=12345;
        sendText=IP changed: $adapters|$network_names;
        recvText=beeperAck;
        timeout=5';
    continue=yes;"
```

标识到 MobiLink 服务器的最佳路径的更改

_BEST_IP_CHANGED_ 消息指示 MobiLink 服务器的最佳路径发生变化。在通过使用以下语法运行监听器时可以引用此消息：

```
dblsn ... -x MobiLink-protocol-options -l
"message=_BEST_IP_CHANGED_;action='...'"
```

过滤 **_BEST_IP_CHANGED_** 消息时，$best_ip 操作变量（用代表最佳 IP 连接的本地 IP 地址替代）可帮助您启动有用的操作。如果没有 IP 连接，则 $best_ip 返回 0.0.0.0。

在以下示例中，**_BEST_IP_CHANGED_** 消息用于在最佳 IP 连接变化时启动同步。如果连接中断，就会生成错误。

```
dblsn -x http(host=mlserver.company.com)
   -v2 -m -i 3 -ot dblsn.log
   -l "message=_BEST_IP_CHANGED_;
       action='
           START dbmlsync.exe -ra -c eng=remote;uid=DBA;pwd=sql -n test_pub'"
```

注意
测试与应用程序之间的连接启动的同步时，在与 MobiLink 服务器分开的单独计算机上运行监听器。

另请参见

- “MobiLink 监听器实用程序”第 55 页
- “用于 Windows 的监听器操作命令”一节第 70 页
- “用于 Windows 的监听器操作变量”一节第 73 页

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

轻量级轮询器

轻量级轮询器是按指定的时间间隔轮询推式通知的设备应用程序。使用轻量级轮询器是设置网关的替代方法，也是建议使用的方法，因为它不像 SYNC 网关那样需要与服务器保持持久性连接，也不像 UDP 网关那样需要持续连接。

设备轮询服务器时将发送轮询键和通告程序名称。MobiLink 服务器通过检查通告程序名称确定应由哪个通告程序检查高速缓存中是否有推式请求。轮询键向通告程序标识设备，通告程序使用轮询键检测计划用于该设备的推式请求。检测到推式请求后发送推式通知。

使用监听器命令行选项配置轻量级轮询器。此外，也可使用轻量级轮询 API 将轻量级轮询器集成到设备应用程序中。

注意
Palm 设备不支持轻量级轮询。要检索推式通知，必须配置网关。请参见“网关和运营公司”一节第 27 页。

另请参见

- “设置轻量级轮询选项”一节第 18 页
- “用于 Windows 的监听器关键字”一节第 68 页

轻量级轮询 API

轻量级轮询 API 是可以集成到设备应用程序中的编程接口。它包含轮询服务器所需的方法。

所需文件

所有目录都相对于 *install-dir* 目录。以下是编译轻量级轮询 API 所需的文件列表：

文件名或位置	说明
SDK\Lib\x86\mllplib.lib	轻量级轮询 API 运行时库。
SDK\Include\mllplib.h	轻量级轮询 API 头文件。

samples-dir\MobiLink\SIS_CarDealer_LP_API 中提供了一个用 C 编写的 SIS_CarDealer_LP_API 示例应用程序。有关 *samples-dir* 的详细信息，请参见“示例目录”一节 《SQL Anywhere 服务器 - 数据库管理》。

API 方法

方法	说明
“MLLightPoller 类”一节第 22 页	表示轻量级轮询器对象。
“MLLPCreatePoller 方法”一节第 25 页	创建 MLLightPoller 的实例。

方法	说明
“MLLPDestroyPoller 方法”一节第 25 页	取消 MLLightPoller 的实例。

MLLightPoller 类

表示轻量级轮询器对象。

语法

class **MLLightPoller**

另请参见

- “auth_status 枚举”一节第 22 页
- “Poll 方法”一节第 23 页
- “return_code 枚举”一节第 23 页
- “SetConnectInfo 方法”一节第 24 页

示例

```
MLLightPoller * poller = MLLCreatePoller();
```

auth_status 枚举

指定可能的验证状态代码。

语法

```
typedef enum auth_status {
   AUTH_UNKNOWN,
   AUTH_VALID,
   AUTH_VALID_BUT_EXPIRES_SOON,
   AUTH_EXPIRED,
   AUTH_INVALID,
   AUTH_IN_USE
} auth_status;
```

成员

名称	说明
AUTH_EXPIRED	验证已到期。
AUTH_IN_USE	用户名已经过验证。
AUTH_INVALID	验证无效。
AUTH_UNKNOWN	验证未知。

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

名称	说明
AUTH_VALID	验证有效。
AUTH_VALID_BUT_EXPIRES_SOON	验证有效，但很快到期。

Poll 方法

轮询服务器，提示通告程序检查高速缓存中是否有推式请求。

语法

```
return_code Poll(
   const char * Notifier,
   const char * key,
   char * subject = 0,
   size_t * subjectSize = 0,
   char * content = 0,
   size_t * contentSize = 0
) = 0;
```

参数

- **Notifier** 通告程序的名称。
- **key** 标识监听器的轮询键的名称。
- **subject** 接收消息主题的缓冲区。（以空值终止）
- **subjectSize** IN：主题缓冲区的大小。

  OUT：所接收主题的大小，包括空终止符，其中零表示空主题。
- **content** 接收消息内容的缓冲区。（以空值终止）
- **contentSize** IN：内容缓冲区的大小。

  OUT：所接收内容的大小，包括空终止符，其中零表示空内容。

返回值

return_code 枚举中列出的代码之一。请参见“return_code 枚举”一节第 23 页。

注释

监听器连接到通告程序，然后在通告程序检查其高速缓存中是否有指定用于给定轮询键的推式通知后断开连接。

return_code 枚举

指定可能的返回代码。

语法

```
typedef enum return_code {
   OK,
   BAD_STREAM_NAME,
   BAD_STREAM_PARAM,
   CONNECT_FAILED,
   KEY_NOT_FOUND,
   SUBJECT_OVERFLOW,
   CONTENT_OVERFLOW,
   COMMUNICATION_ERROR,
   AUTH_FAILED,
   NYI
} return_code;
```

成员

名称	说明
AUTH_FAILED	无法验证与 MobiLink 服务器的连接。
BAD_STREAM_NAME	无法识别的流名称。
BAD_STREAM_PARAM	解析流参数时失败。
COMMUNICATION_ERROR	由于通信错误失败。
CONNECT_FAILED	无法连接到 MobiLink 服务器。
CONTENT_OVERFLOW	内容缓冲区过小。
KEY_NOT_FOUND	没有来自指定通告程序的适用于指定键的推式通知。
NYI	仅供内部使用。
OK	方法成功运行。
SUBJECT_OVERFLOW	主题缓冲区过小。

SetConnectInfo 方法

设置 MobiLink 客户端流类型和网络协议选项。

语法

```
return_code SetConnectInfo(
   const char * streamName,
```

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

```
    const char * streamParams
);
```

参数

- **streamName** 要使用的网络协议。可接受的值为：**tcpip**、**http**、**https** 或 **tls**。
- **streamParams** 协议选项字符串，格式为以分号分隔的一列值。有关选项的完整列表，请参见“MobiLink 客户端网络协议选项”《MobiLink - 客户端管理》。

返回值

return_code 枚举中列出的代码之一。请参见“return_code 枚举”一节第 23 页。

示例

```
poller_ret = poller->SetConnectInfo("http", "host=localhost;port=80;")
```

MLLPCreatePoller 方法

创建 MLLightPoller 的实例。

语法

```
MLLightPoller * _entry MLLPCreatePoller( );
```

返回值

新的 MLLightPoller 对象。

另请参见

- “MLLPDestroyPoller 方法”一节第 25 页

示例

```
poller = MLLPCreatePoller();
```

MLLPDestroyPoller 方法

取消 MLLightPoller 的实例。

语法

```
void _entry MLLPDestroyPoller(
    MLLightPoller * poller
);
```

参数

- **poller** 要取消的 MLLightPoller。

注释

此方法接受空值 MLLightPoller 对象。

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

另请参见

- “MLLPCreatePoller 方法”一节第 25 页

示例

```
MLLPDestroyPoller(poller);
```

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

网关和运营公司

将网关用作轻量级轮询器的替代方法

网关是用于发送推式通知的机制。网关是轻量级轮询器的替代方法，需要持续不断的网络连接。

在 MobiLink 服务器上配置网关属性。可以在一个 MobiLink 服务器上配置多个网关。请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

支持的网关

在 MobiLink 服务器上支持以下网关：

- **SYNC 网关** SYNC 网关是基于 TCP/IP 的网关；通过与 MobiLink 同步相同的协议发送推式通知。

  缺省 SYNC 网关的名称为 Default-SYNC。通常不需要更改缺省网关设置。

  请参见“SYNC 网关属性”一节第 52 页。

- **UDP 网关** UDP 网关通过 UDP 网关发送推式通知。

  缺省 UDP 网关的名称为 Default-UDP。通常不需要更改缺省网关设置。缺省情况下，监听器监听推式通知时使用 UDP。

  请参见“UDP 网关属性”一节第 52 页。

- **SMTP 网关** SMTP 网关使用 "电子邮件到 SMS" 运营公司服务发送推式通知。

  缺省 SMTP 网关的名称为 Default-SMTP。

  请参见“SMTP 网关属性”一节第 51 页。

设备跟踪网关

除了支持的网关，您还可以配置设备跟踪网关，它会自动选择最合适的网关来发送推式通知。缺省的设备跟踪网关为 Default-DeviceTracker。如果您不想使用轻量级轮询器，建议您使用此网关。有关设备跟踪的详细信息，请参见“设备跟踪网关”一节第 27 页。

设备跟踪网关

设备跟踪允许 MobiLink 服务器使用推式请求的远程 ID 信息跟踪设备。设备跟踪网关利用自动跟踪的 IP 地址、电话号码和公共无线网络提供商 ID 通过 SYNC、UDP 和 SMTP 网关传送推式通知。网关首先尝试使用 SYNC 网关连接到设备。如果传送失败，则尝试使用 UDP 网关连接，然后再尝试使用 SMTP 网关连接。如果预期设备地址会更改，此功能很有用。

设备跟踪网关最多可以有三个下级网关：一个 SYNC、一个 SMTP 和一个 UDP。将根据监听器发送的设备跟踪信息将推式通知自动路由到其中一个下级网关。通过启用这些下级网关，MobiLink 服务器可以自动管理设备地址更改。地址更改时，监听器与统一数据库同步，更新位于 ml_device_address 系统表中的跟踪信息。

多数 9.0.1 或更高版本的监听器都支持设备跟踪。如果使用不支持设备跟踪的监听器，可以通过自己提供跟踪信息使用设备跟踪网关。请参见“添加设备跟踪支持”一节第 28 页。

另请参见

- “将网关用作轻量级轮询器的替代方法”一节第 27 页
- “运营公司和运营公司配置”一节第 30 页

添加设备跟踪支持

注意
只有使用 Palm 设备的监听器时，或者使用在 Adaptive Server Anywhere 9.0.0 或更早版本上运行的监听器时，才需要支持设备跟踪。所有其它监听器都已支持设备跟踪。

有几个系统过程可用于为 9.0.0 监听器或 Palm 设备的监听器手工设置设备跟踪。这些过程将更新统一数据库上的 ml_device、ml_device_address 和 ml_listening MobiLink 系统表。

利用手工设备跟踪，可以通过 MobiLink 用户名对接收者进行寻址，无需提供网络地址信息。但是，如果信息更改，MobiLink 无法自动更新；必须手工更改。此方法对于 SMTP 网关尤为有用，因为电子邮件地址很少更改。

对于 UDP 网关，如果每次重新连接时 IP 地址都更改，则不能依赖静态条目。可以通过按照主机名而不是 IP 地址进行寻址来解决此问题。但是，此解决方案会使 DNS 服务器表的更新减慢，可能导致推式通知发错方向。您也可以通过编程方式设置系统过程来更新系统表。

♦ 为 9.0.0 监听器或 Palm 监听器手工设置设备跟踪

1. 对于每个设备，在 ml_device 系统表中都添加一个设备记录。例如：

```
CALL ml_set_device(
    'myFirstTreo180',
    'MobiLink Listeners for Treo 180 - 9.0.1 Palm Listener',
    '1',
    'not used',
    'y',
    'manually entered by administrator'
);
```

   第一个参数 **myFirstTreo180** 是唯一的用户定义的设备名。第二个参数包含有关监听器版本的可选标记。第三个参数指定监听器版本；使用 **0** 表示 SQL Anywhere 9.0.0 监听器，使用 **1** 表示 Palm 设备的 9.0.0 之后的监听器，使用 **2** 表示 Windows 的 9.0.0 之后的监听器。第四个参数指定可选的设备信息。第五个参数指定是否忽略设备跟踪。最后一个参数包含此条目的可选注释。

   请参见“ml_set_device 系统过程”一节第 108 页。

2. 对于每个设备，都在 ml_device_address 系统表中添加一个地址记录。例如：

```
CALL ml_set_device_address(
    'myFirstTreo180',
    'ROGERS AT&T',
    '55511234567',
    'y',
    'y',
```

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

```
        'manually entered by administrator'
    );
```

第一个参数 **myFirstTreo180** 是唯一的用户定义的设备名。第二个参数是网络提供商 ID，必须与 network_provider_id 运营公司属性匹配。第三个参数是 UDP 的 IP 地址。第四个参数确定是否激活此条目用于发送推式通知。第五个参数指定是否忽略设备跟踪。最后一个参数包含此条目的可选注释。

3. 对于每个远程数据库，请为添加的每个设备在 ml_listening 系统表中添加一个接收者记录。这会将设备映射到 MobiLink 用户名。例如：

```
CALL ml_set_listening(
    'myULDB',
    'myFirstTreo180',
    'y',
    'y',
    'manually entered by administrator'
);
```

第一个参数为 MobiLink 用户名。第二个参数是用户定义的唯一的设备名。第三个参数确定是否激活此条目用于设备跟踪寻址。第四个参数指定是否忽略设备跟踪。最后一个参数包含此条目的可选注释。

另请参见

- “ml_set_listening 系统过程”一节第 111 页
- “ml_set_device_address 系统过程”一节第 109 页
- “设备跟踪网关”一节第 27 页
- “运营公司属性”一节第 54 页

设备跟踪网关配置快速入门指南

♦ **设置设备跟踪**

1. 设置 SYNC、UDP 或 SMTP 网关。

   启动 MobiLink 服务器时，已使用缺省设置建立这些网关。有关配置属性或创建自己的网关的详细信息，请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

   注意
   SMTP 网关需要运营公司配置。请参见“运营公司和运营公司配置”一节第 30 页。

2. 创建新的通告程序，并使用以下条件设置 request_cursor 事件：
   - 网关名称必须设置为想要使用的设备跟踪网关的名称。缺省网关的名称为 Default-DeviceTracker。此名称由结果集的第一列表示。
   - 地址名称必须设置为设备的远程 ID。使用 dblsn -t+ 选项将远程 ID 注册到 MobiLink 服务器。此名称由结果集的第四列表示。

   有关设置 request_cursor 事件的详细信息，请参见“request_cursor 事件”一节第 40 页。

3. 将监听器名称添加到 ml_user 系统表中。

缺省监听器名称是 *device_name*-**dblsn**，其中 *device_name* 是设备的名称。

运行监听器以查看设备名称，可以在监听器消息窗口中看见此名称。此外，也可以使用 dblsn -e 选项设置设备名称，或者使用 dblsn -u 选项设置另一个监听器名称。请参见“-e 选项”一节第 61 页和“-u 选项”一节第 66 页。

有关注册 MobiLink 用户的详细信息，请参见“创建和注册 MobiLink 用户”一节 《MobiLink - 客户端管理》。

4. 使用必需的选项启动监听器。有关启动监听器的详细信息，请参见“MobiLink 监听器实用程序”第 55 页。

运营公司和运营公司配置

必须配置无线运营公司才能通过 SMTP 网关发送推式通知，因为通告程序需要构造有效的电子邮件地址。使用启用 SMTP 下级网关的设备跟踪网关时，也必须配置无线运营公司。

在 MobiLink 服务器上配置运营公司属性，如网络提供商 ID 和 SMS 电子邮件前缀。要容纳多个运营公司服务，请在 MobiLink 上配置多个运营公司。请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

Sender 语法

推式通知被监听器接收并进行预处理进行消息过滤时，会被分成几个关键字。**message** 中的 **sender** 关键字是由设备生成的电子邮件地址，视无线运营公司而异。有关预处理消息的详细信息，请参见“消息语法”一节第 114 页。

sender 语法采用以下格式：

```
sender = sms_email_user_prefix phone-number@sms_email_domain
```

注意
sms_email_user_prefix 和 *phone-number* 之间没有空格。

sms_email_user_prefix 和 *sms_email_domain* 值是运营公司属性，在 MobiLink 服务器上配置。*phone-number* 值取自 ml_device_address 系统表的地址列。

要确定 sender 语法，需要在使用运营公司服务的设备上运行监听器。使用 dblsn -m 和 -v 选项启用消息记录并将详细程度级别设置为 2。加载监听器之后检查消息日志。

另请参见

- “用于服务器启动的同步的 MobiLink 服务器设置”第 31 页
- “运营公司属性”一节第 54 页

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

用于服务器启动的同步的 MobiLink 服务器设置

目录

服务器端设置包括通告程序属性、网关属性、运营公司属性和通告程序事件。要配置这些设置，请使用以下方法之一：

- Sybase Central
- 通告程序配置文件
- ml_add_property 系统过程

Sybase Central 和 ml_add_property 系统过程方法将事件和设置添加到 ml_property 系统表。

注意
MobiLink 服务器运行时，对服务器端设置所做的更改不生效。要应用新设置，必须关闭后重新启动 MobiLink 服务器。

如果已配置 ml_property 系统表中的服务器端设置，又想使用通告程序配置文件，则始终先装载系统表设置，然后装载文件设置。通告程序配置文件覆盖现有服务器端设置，但更改不永久应用于统一数据库。

使用 ml_add_property 系统过程配置服务器端设置

要配置 SQL Anywhere 统一数据库的服务器端设置，请使用 ml_add_property 系统过程。可以使用 Interactive SQL 设置这些属性和事件。

常用属性语法

```
CALL ml_add_property('SIS', '', 'Property', Value);
```

通告程序属性和事件语法

```
CALL ml_add_property('SIS', 'Notifier(NotifierName)', 'Event-or-Property',
Value);
```

网关属性语法

```
CALL ml_add_property('SIS', 'DeviceTracker(DeviceTrackerName)', 'Property',
Value);

CALL ml_add_property('SIS', 'SMTP(SMTPName)', 'Property', Value);

CALL ml_add_property('SIS', 'UDP(UDPName)', 'Property', Value);

CALL ml_add_property('SIS', 'SYNC(SYNCName)', 'Property', Value);
```

运营公司属性语法

```
CALL ml_add_property('SIS', 'Carrier(CarrierName)', 'Property', Value);
```

有关详细信息，请参见“ml_add_property 系统过程”一节 《MobiLink - 服务器管理》。

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

使用 Sybase Central 配置服务器端设置

Sybase Central 提供图形用户界面以修改属性和事件。可以使用 Sybase Central 配置多个通告程序、网关和运营公司。通过 Sybase Central 配置服务器端设置后，使用 mlsrv11 -notifier 选项时不需要在命令行指定通告程序配置文件。

♦ 使用 Sybase Central 设置通告程序、网关和运营公司

1. 使用 MobiLink 插件从 Sybase Central 连接到统一数据库。
2. 选择 [**模式**] » [**管理**]。
3. 单击 [**通知**]。

   在右窗格中，将出现所有可用通告程序、网关和运营公司。
4. 创建新的通告程序、网关和运营公司。
   - 要创建新的通告程序，在右窗格中，单击 [**通告程序**] 选项卡，然后选择 [**文件**] » [**新建**] » [**通告程序**]。
   - 要创建新的网关，在右窗格中，单击 [**网关**] 选项卡，然后选择 [**文件**] » [**新建**] » [**网关**]。
   - 要创建新的运营公司，在右窗格中，单击 [**运营公司**] 选项卡，然后选择 [**文件**] » [**新建**] » [**运营公司**]。
5. 选择要配置的通告程序、网关或运营公司。
   - 要设置通告程序属性或事件，在右窗格中，单击 [**通告程序**] 选项卡，然后选择要配置的通告程序。
   - 要设置网关属性，在右窗格中，单击 [**网关**] 选项卡，然后选择要配置的网关。
   - 要设置运营公司属性，在右窗格中，单击 [**运营公司**] 选项卡，然后选择要配置的运营公司。

   选择 [**文件**] » [**属性**]。

   将出现一个窗口，可在其中调整适用于所选通告程序、网关或运营公司的所有设置。
6. 单击 [**确定**]。

将服务器端设置导入到通告程序配置文件

可以将服务器端设置从通告程序配置文件导入 ml_property_table。

♦ 从通告程序配置文件导入服务器端设置

1. 使用 MobiLink 插件从 Sybase Central 连接到统一数据库。
2. 选择 [**模式**] » [**管理**]。
3. 单击 [**通知**]。
4. 选择 [**文件**] » [**导入设置**]，然后按照向导中的说明操作。

将服务器端设置导出到通告程序配置文件

可以将服务器端设置从 ml_property 表导出到通告程序配置文件中。通过导出设置，可以创建服务器端设置的多个版本，然后可以使用 mlsrc11 -notifier 选项装载不同版本。

♦ 将服务器端设置导出到通告程序配置文件

1. 使用 MobiLink 插件从 Sybase Central 连接到统一数据库。
2. 选择 [**模式**] » [**管理**]。
3. 单击 [**通知**]。
4. 选择 [**文件**] » [**导入设置**]，然后按照向导中的说明操作。

另请参见

- “在 [模型] 模式中设置服务器启动的同步”一节 《MobiLink - 入门》

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

使用通告程序配置文件配置服务器端设置

可以将服务器端设置存储在通告程序配置文件中。可以使用此文件配置多个通告程序、网关和运营公司。

创建和配置通告程序配置文件

可以使用文本编辑器创建通告程序配置文件，也可以从 Sybase Central 导出的属性和事件设置生成通告程序配置文件。请参见“使用 Sybase Central 配置服务器端设置”一节第 33 页。

要查看典型通告程序配置文件的布局，打开 *samples-dir\MobiLink\template.Notifier* 模板文件，其中 *samples-dir* 是 SQL Anywhere 11 示例目录的位置。模板文件提供配置服务器端属性和事件的示例。

配置必要设置后，保存通告程序配置文件，并将服务器端属性和事件载入 MobiLink 服务器。

常用属性语法

```
Property = Value
```

通告程序事件语法

```
Notifier(NotifierName).Event = \
# Replace this text with SQL script.          \
# Be sure to put a backslash (\) at           \
# the end of every line of code               \
# if your event requires multiple             \
# lines of text.
```

通告程序属性语法

```
Notifier(NotifierName).Property = Value
```

网关属性语法

```
# For Device tracking gateways:
DeviceTracker(DeviceTrackerName).Property = Value
# For SMTP gateways:
SMTP(SMTPName).Property = Value
# For SYNC gateways:
SYNC(SYNCName).Property = Value
# For UDP gateways:
UDP(UDPName).Property = Value
```

运营公司属性语法

```
Carrier(CarrierName).Property = Value
```

装载通告程序配置文件

要将通告程序配置文件载入 MobiLink 服务器，请从命令行运行 mlsrv11，并指定 -notifier 选项。例如，要使用在 CarDealer.Notifier 配置文件中定义的服务器端设置，运行以下命令：

```
mlsrv11 ... -notifier "c:\CarDealer.Notifier"
```

如果未指定文件，缺省情况下，装载 *config.Notifier* 文件。

有关 mlsrv11 -notifier 选项的详细信息，请参见“-notifier 选项”一节 《MobiLink - 服务器管理》。

注意
如果要使用缺省 SYNC 网关，则无法将服务器端设置存储到通告程序配置文件中。必须使用替代方法将服务器端设置存储到 ml_property 系统表中。请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

使用转义序列

反斜线 (\) 是转义字符。以下列出可以在通告程序配置文件中使用的常用转义序列：

转义序列	说明
\b	退格符
\t	选项卡
\n	换行符
\r	回车符
\"	双引号 (")
\'	单引号 (')
\\	反斜线 (\)
\e	转义字符

Unicode 转义序列的形式为 \u*XXXX*，而 ASCII 转义序列的形式为 \x*XX*，其中每个 *X* 代表一个十六进制数字。

编辑需要多行文本的属性或事件时，在每行结尾添加一个单反斜线字符 (\)。

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

通告程序事件

只要通告程序轮询监听器，就会触发事件。触发事件时，将执行与事件关联的 SQL 脚本。可以将 SQL 脚本合并到本节列出的任何通告程序事件。虽然脚本编写可选，但必须编写 request_cursor 轮询事件脚本。

有三类通告程序事件：轮询事件、连接事件和异步事件。每次通告程序检查统一数据库时触发轮询事件，包括在 begin_poll 事件和 end_poll 事件之间发生的所有事件。通告程序数据库连接期间触发连接事件。同步过程期间任何时候都可以触发异步事件。

除非另行指定，否则可以使用任何建议的方法配置通告程序事件。有关配置通告程序事件的详细信息，请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

监听器轮询通告程序时，将按以下顺序触发这些事件：

```
Fire begin_connection event
For each poll (
    Fire begin_poll event
        Fire shutdown_query event
        Fire request_cursor event
        For all requests expired before required confirmation (
            Fire error_handler event
        )
        Fire request_delete event
    Fire end_poll event
)
Fire end_connection event
```

轮询事件

轮询事件是每次通告程序检查统一数据库时触发的一类通告程序事件。这些事件包括在 begin_poll 事件和 end_poll 事件之间发生的所有事件。

begin_poll 事件

此轮询事件接受 SQL 脚本，并在通告程序检查统一数据库中是否有推式请求之前触发。缺省情况下，此值为空值，所以不触发此事件。

另请参见

- “推式请求”一节第 8 页
- “通告程序事件”一节第 37 页
- “用于服务器启动的同步的 MobiLink 服务器设置”第 31 页

示例

此示例为名为 Notifier A 的通告程序创建推式请求。它使用一条 SQL 语句向名为 PushRequest 的表中插入行。此表中的每一行都代表发送到一个地址的消息。WHERE 子句决定哪些推式请求被插入到 PushRequest 表中。

要将 ml_add_property 系统过程与 SQL Anywhere 统一数据库一起使用，请运行以下命令：

```
ml_add_property(
    'SIS',
    'Notifier(Notifier A)',
    'begin_connection',
    'INSERT INTO PushRequest
        (gateway, mluser, subject, content)
        SELECT ''MyGateway'', DISTINCT mluser, ''sync'',
            stream_param
            FROM MLUserExtra, mluser_union, Dealer
            WHERE MLUserExtra.mluser = mluser_union.name
            AND (push_sync_status = ''waiting for request''
                OR datediff( hour, last_status_change, now() ) > 12 )
            AND ( mluser_union.publication_name is NULL
                OR mluser_union.publication_name =''FullSync'' )
            AND Dealer.last_modified > mluser_union.last_sync_time'
);
```

end_poll 事件

此轮询事件接受 SQL 脚本，并在通告程序检查统一数据库中是否有推式请求之后触发。缺省情况下，此值为空值，所以不触发此事件。

可以使用此事件执行表清理或记录轮询结果。

另请参见

- “通告程序事件”一节第 37 页
- “用于服务器启动的同步的 MobiLink 服务器设置”第 31 页

error_handler 事件

配置此事件在传输失败或未确认时进行指示。例如，传输失败时，可以使用此事件在审计表中插入行或发送推式通知。

下表详细说明可以使用 error_handler 事件捕获的参数：

脚本参数	类型	说明
request_option (out)	Integer	控制错误处理程序返回后通告程序对推式请求执行的操作。输出可以是以下值之一： ● **0**：执行基于错误代码的缺省操作并记录错误。 ● **1**：不执行任何操作。 ● **2**：执行 request_delete 事件。 ● **3**：尝试传送到辅助网关。
error_code (in)	Integer	针对错误代码使用以下值之一： ● **-1**：请求超时及成功确认。 ● **-8**：传送尝试过程中出错。

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

脚本参数	类型	说明
request_id (in)	Integer	标识请求。
gateway (in)	Varchar	指定与推式请求关联的网关。
address (in)	Varchar	指定与推式请求关联的地址。
subject (in)	Varchar	指定与推式请求关联的主题。
content (in)	Varchar	指定与推式请求关联的内容。

注意
此事件要求使用系统过程。不能使用 Sybase Central 直接配置此事件。请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

另请参见

- “通告程序事件”一节第 37 页
- “用于服务器启动的同步的 MobiLink 服务器设置”第 31 页

示例

在以下示例中，您将创建一个名为 CustomError 的表，然后使用名为 CustomErrorHandler 的存储过程在表中记录错误。输出参数 Notifier_opcode 始终为 0，表示使用缺省通告程序处理。

```
CREATE TABLE CustomError(
    error_code  integer,
    request_id  integer,
    gateway  varchar(255),
    address  varchar(255),
    subject  varchar(255),
    content  varchar(255),
    occurAt  timestamp not null default timestamp
);

CREATE PROCEDURE CustomErrorHandler(
    out @Notifier_opcode integer,
    in @error_code  integer,
    in @request_id  integer,
    in @gateway   varchar(255),
    in @address   varchar(255),
    in @subject   varchar(255),
    in @content   varchar(255)
)

BEGIN
    INSERT INTO CustomError(
        error_code,
        request_id,
```

```
            gateway,
            address,
            subject,
            content)
        VALUES(
            @error_code,
            @request_id,
            @gateway,
            @address,
            @subject,
            @content
        );
        SET @Notifier_opcode = 0;
    END
```

要将 ml_add_property 系统过程与 SQL Anywhere 统一数据库一起使用，请运行以下命令：

```
call ml_add_property(
    'SIS',
    'Notifier(myNotifier)',
    'error_handler',
    'call CustomErrorHandler(?, ?, ?, ?, ?, ?, ?)');
```

还可以通过将以下行添加到通告程序配置文件触发此事件：

```
Notifier(myNotifier).error_handler = call
CustomErrorHandler(?, ?, ?, ?, ?, ?, ?)
```

使用 mlsrv11 -notifier 选项运行文件。有关如何配置通告程序配置文件的详细信息，请参见“使用通告程序配置文件配置服务器端设置”一节第 35 页。

request_cursor 事件

此轮询事件接受 SQL 脚本，并会被触发以检测推式请求。必须配置此事件。

使用轻量级轮询器（建议）时读取推式请求

此事件的结果集中最多包含三列时，通告程序确认服务器和设备之间没有持久性连接，在推式通知被发送前，设备必须先轮询通告程序。在发送推式通知前，通告程序高速缓存结果集。MobiLink 服务器通过轮询键标识设备，设备每次轮询通告程序都发送轮询键。

此事件的结果集必须按指定顺序包含以下列：

- 轮询键
- 主题（可选）
- 内容（可选）

使用网关时读取推式请求

此事件的结果集中包含三列以上时，通告程序确认服务器和设备之间存在持久性连接，那么在检测到推式请求时使用网关发送推式通知。

此事件的结果集必须按指定顺序包含以下列：

- 请求 ID（可选）

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

- 网关
- 主题
- 内容
- 地址
- 重发间隔（可选）
- 生存期（可选）

另请参见

- “推式请求要求”一节第 8 页
- “通告程序事件”一节第 37 页
- “用于服务器启动的同步的 MobiLink 服务器设置”第 31 页

示例

以下示例使用 ml_add_property 系统过程为名为 Simple 的自定义通告程序创建 request_cursor 事件脚本。SELECT 语句告诉通告程序从名为 PushRequest 的表检测推式请求。

```
CALL ml_add_property('SIS', 'Notifier(Simple)', 'request_cursor',
    'SELECT poll_key,
        subject,
        content
    FROM PushRequest'
);
```

建议您在脚本中包括 WHERE 子句，以过滤掉已发送的请求。例如，可以添加推式请求列来跟踪插入请求的时间，然后在此事件中使用 WHERE 子句过滤掉在上次用户同步之前插入的请求。

request_delete 事件

此轮询事件接受 SQL 脚本，并会被触发以在检测到需要删除推式请求时执行清理操作。它接受请求 ID 作为参数，并按请求 ID 执行。request_cursor 事件必须包含请求 ID 列才能使用 request_delete 事件。可以使用指定参数或问号 (?) 引用请求 ID。如果已将清理操作指派给另一个过程或事件，如 end_poll 事件，则此事件可选。

通告程序可以使用 DELETE 语句删除以下形式的推式请求：

- **隐式删除** 这些推式请求以前出现过，但没有出现在从 request_cursor 事件获取的当前请求集中。
- **已确认** 这些是确认已发送的推式请求。
- **已到期** 这些推式请求根据其重发属性和当前时间已到期。没有重发属性的请求即使出现在下一个请求中也被认为是过期。

可以使用 request_delete 事件防止删除过期的或隐式删除的请求。例如，*samples-dir\MobiLink\SIS_CarDealer* 目录中的 CarDealer 示例使用 request_delete 事件将 PushRequest 表的状态字段设置为 'processed'。

```
UPDATE PushRequest SET status='processed' WHERE req_id = ?
```

示例中的 begin_poll 事件在删除已处理的推式请求前，使用上次同步时间来检查远程设备是否更新。

另请参见

- “通告程序事件”一节第 37 页
- “用于服务器启动的同步的 MobiLink 服务器设置”第 31 页

shutdown_query 事件

此轮询事件接受 SQL 脚本，并在 begin_poll 事件之后触发。返回值指定通告程序的关闭状态。缺省情况下，此值为空值，所以不触发此事件。

要关闭通告程序，将 SQL 脚本设置为返回 'yes'；否则，将其设置为返回 'no'。如果通告程序关闭，则不触发 end_poll 事件。

将关闭状态存储到表时，使用 end_connection 事件重置状态。

另请参见

- “end_connection 事件”一节第 43 页
- “通告程序事件”一节第 37 页
- “用于服务器启动的同步的 MobiLink 服务器设置”第 31 页

示例

以下示例使用 ml_add_property 系统过程为名为 Simple 的自定义通告程序创建 shutdown_query 事件脚本。SELECT 语句告诉通告程序如果 tooManyNotifierErrors 方法返回 true 则关闭。

```
CALL ml_add_property('SIS', 'Notifier(Simple)', 'shutdown_query',
    'SELECT
        IF tooManyNotifierErrors() THEN
            'yes'
        ELSE
            'no'
        ENDIF'
);
```

连接事件

连接事件是通告程序数据库连接期间触发的一类通告程序事件。

begin_connection 事件

此事件接受 SQL 脚本，并在通告程序连接到统一数据库之后、检查推式请求之前触发。缺省情况下，此值为空值，所以不触发此事件。

可以使用此事件创建临时表或变量。您不应使用此事件更改隔离级别。若要控制隔离级别，请使用 isolation 属性。请参见“通告程序属性”一节第 48 页。

如果通告程序失去与统一数据库的连接，在重新连接后它将立即重新运行此事件。

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

另请参见

- “通告程序事件”一节第 37 页
- “用于服务器启动的同步的 MobiLink 服务器设置”第 31 页

end_connection 事件

此事件接受 SQL 脚本，并在通告程序与统一数据库断开连接之前的一刻触发。缺省情况下，此值为空值，所以不触发此事件。

可以使用此事件清理临时存储，如 SQL 变量和临时表。

另请参见

- “通告程序事件”一节第 37 页
- “用于服务器启动的同步的 MobiLink 服务器设置”第 31 页

异步事件

异步事件是同步过程期间任何时候都可以触发的一类通告程序事件。

confirmation_handler 事件

配置此事件处理由监听器上载的传送确认信息。如果状态参数返回 0，则由 request_id 标识的推式请求被由 remote_device 参数标识的监听器成功接收。

可以使用 request_option 参数启动操作以响应传送确认。如果 request_option 为 0，则 confirmation_handler 事件启动缺省操作，执行 request_delete 事件删除原始推式请求。如果发送传送确认的设备与 request_id 标识的设备不匹配，则缺省操作是通过辅助网关发送原始推式请求。

注意
使用 dblsn -x 选项允许监听器上载传送确认信息。如果需要传送确认但不需要 IP 跟踪，可使用 dblsn -ni 选项。请参见“用于 Windows 的监听器选项”一节第 57 页。

注意
此事件要求使用系统过程。不能使用 Sybase Central 方法直接配置此事件。请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

可以使用 confirmation_handler 事件捕获以下参数：

脚本参数	类型	说明
request_option (out)	Integer	控制处理程序返回后通告程序对请求执行的操作。可以返回以下值： ● **0**：根据状态参数的值执行缺省通告程序操作。如果状态指示响应设备为目标一，则通告程序删除请求；否则通告程序尝试在辅助网关上进行传送。 ● **1**：不执行任何操作。 ● **2**：执行 Notifier.request_delete。 ● **3**：尝试传送到辅助网关。
status (in)	Integer	情况概览。可在开发过程中使用状态来确定问题，例如不正确的过滤器和处理程序属性。可以返回以下值： ● **0**：已接收并确认。 ● **-2**：正确响应但消息被拒绝。 ● **-3**：正确响应且消息被接受，但操作失败。 ● **-4**：错误响应而消息被接受。 ● **-5**：错误响应且消息被拒绝。 ● **-6**：错误响应。消息被接受且操作成功。 ● **-7**：错误响应。消息被接受但操作失败。
request_id (in)	Integer	请求 ID。request_cursor 事件必须包含请求 ID 列才能使用 confirmation_handler 事件。
remote_code (in)	Integer	远程监听器报告的汇总。可以返回以下值： ● **1**：消息被接受。 ● **2**：消息被拒绝。 ● **3**：消息被接受且操作成功。 ● **4**：消息被接受但操作失败。
remote_device (in)	Varchar	响应监听器的设备名。
remote_mluser (in)	Varchar	响应监听器的 MobiLink 用户名。
remote_action_return (in)	Varchar	远程操作的返回代码。
remote_action (in)	Varchar	为操作命令保留。
gateway (in)	Varchar	与请求关联的网关。

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

脚本参数	类型	说明
address (in)	Varchar	与请求关联的地址。
subject (in)	Varchar	与请求关联的主题。
content (in)	Varchar	与请求关联的内容。

另请参见

- “通告程序事件”一节第 37 页
- “用于服务器启动的同步的 MobiLink 服务器设置”第 31 页
- “网关属性”一节第 50 页

示例

在以下示例中，您将创建一个名为 CustomConfirmation 的表，然后使用名为 CustomConfirmationHandler 的存储过程在其中记录确认。输出参数 request_option 始终设置为 0，表示使用缺省通告程序处理。

```
CREATE TABLE CustomConfirmation(
    error_code   integer,
    request_id   integer,
    remote_code   integer,
    remote_device  varchar(128),
    remote_mluser  varchar(128),
    remote_action_return varchar(128),
    remote_action  varchar(128),
    gateway   varchar(255),
    address   varchar(255),
    subject   varchar(255),
    content   varchar(255),
    occurAt   timestamp not null default timestamp
)

CREATE PROCEDURE CustomConfirmationHandler(
    out @request_option integer,
    in @error_code  integer,
    in @request_id  integer,
    in @remote_code  integer,
    in @remote_device  varchar(128),
    in @remote_mluser  varchar(128),
    in @remote_action_return varchar(128),
    in @remote_action  varchar(128),
    in @gateway   varchar(255),
    in @address   varchar(255),
    in @subject   varchar(255),
    in @content   varchar(255)
)

BEGIN
    INSERT INTO CustomConfirmation(
        error_code,
        request_id,
        remote_code,
```

```
            remote_device,
            remote_mluser,
            remote_action_return,
            remote_action,
            gateway,
            address,
            subject,
            content)
        VALUES (
            @error_code,
            @request_id,
            @remote_code,
            @remote_device,
            @remote_mluser,
            @remote_action_return,
            @remote_action,
            @gateway,
            @address,
            @subject,
            @content
        );
        SET @request_option = 0;
    END
```

要将 ml_add_property 系统过程与 SQL Anywhere 统一数据库一起使用，请运行以下命令：

```
call ml_add_property(
    'SIS',
    'Notifier(myNotifier)',
    'confirmation_handler',
    'call CustomConfirmation(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)');
```

还可以通过将以下行添加到通告程序配置文件调用此事件：

```
Notifier(myNotifier).confirmation_handler = call
CustomConfirmation(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
```

使用 mlsrv11 -notifier 选项运行文件。有关如何配置通告程序配置文件的详细信息，请参见“使用通告程序配置文件配置服务器端设置”一节第 35 页。

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

常用属性

常用属性在通告程序、网关和运营公司之间共享。所有常用属性都是可选的。有关设置这些属性的详细信息，请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

属性	值	说明
verbosity	{ **0** | **1** | **2** | **3** }	指定通告程序、网关和运营公司的详细程度级别。可以使用以下值： ● **0**：无跟踪。 ● **1**：启动、关机和属性跟踪。 ● **2**：显示通知。 ● **3**：完全级别跟踪。 缺省值为 **0**。

通告程序属性

通告程序属性允许您更改通告程序的行为。所有通告程序属性都是可选的。有关设置这些属性的详细信息，请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

属性	值	说明
connect_string	*connection_string*	覆盖用于连接到数据库的缺省连接行为。缺省值为 **ianywhere.ml.script.ServerContext**，它使用在 mlsrv11 命令行中指定的连接字符串。 当您希望通知逻辑和数据与同步数据分离时，它对连接到其它数据库可能有用。大部分部署不设置此属性。
enable	{ **yes** | **no** }	指定是否应启用通告程序。运行 -notifier mlsrv11 选项时将启动所有已启用通告程序。
gui	{ **yes** | **no** }	指定通告程序运行时是否应显示通告程序窗口。缺省值为 **yes**。 此通告程序窗口允许用户临时更改轮询间隔或立即轮询。它还可用于关闭通告程序，而无需关闭 MobiLink 服务器。一旦停止，只有通过关闭 MobiLink 服务器，然后重新启动，才能重新启动通告程序。
isolation	{ **0** | **1** | **2** | **3** }	指定通告程序数据库连接的隔离级别。可以使用以下值： ● **0**：未提交的读操作。 ● **1**：已提交的读操作。 ● **2**：可重复读取。 ● **3**：可序列化。 缺省值为 **1**。较高的级别会增加争用程度，并可能导致性能降低。隔离级别 **0** 允许读取未提交的数据，这些数据可以被回退。

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

属性	值	说明
poll_every	*number*{ **s** | **m** | **h** }	指定确认超时之前等待的时间。以下是可接受的时间单位列表： ● **s**：表示秒。 ● **m**：表示分。 ● **h**：表示小时。 缺省值为 **1m**。可以在 *HH***h** *MM***m** *SS***s** 格式中组合时间单位。如果未指定时间单位，以秒为单位测量时间。
shared_database_connection	{ **yes** | **no** }	指定通告程序是否应共享数据库连接。缺省值为 **no**。只有在通告程序隔离级别相同时它们才能共享连接。 指定 **yes** 可在不降低性能的情况下节约资源。在某些情况下不能连接共享，如应用程序使用在通告程序之间不唯一的 SQL 变量名时。

网关属性

缺省情况下，启动 MobiLink 服务器时创建四个预配置的网关。当您运行统一数据库的 MobiLink 安装脚本时，会安装这些网关。缺省网关命名如下：

- Default-DeviceTracker 网关
- Default-SYNC 网关
- Default-UDP 网关
- Default-SMTP 网关

不应删除缺省网关或更改其名称。建议您使用不同名称创建其它网关。

不需要更改在 DefaultSYNC 和 DefaultUDP 中定义的属性，但必须向 DefaultSYNC 网关提供 SMTP 服务器信息。应使用缺省网关，但如果需要，可以使用替代配置。本节提供自定义网关属性的过程。

设备跟踪网关属性

设备跟踪网关属性允许您更改设备跟踪网关的行为。所有设备跟踪网关属性都是可选的。有关设置这些属性的详细信息，请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

属性	值	说明
confirm_action	{ **yes** | **no** }	指定是否通过此网关发送传送确认。缺省值为 **no**。
confirm_delivery	{ **yes** | **no** }	指定监听器是否应向统一数据库确认消息已接收。缺省值为 **yes**。启动监听器时必须指定 -x 监听器选项。
description	*description_text*	描述网关。
enable	{ **yes** | **no** }	指定是否应使用设备跟踪网关。
smtp_gateway	*smtp_gateway_name*	指定 SMTP 下级网关的名称。缺省值为 **DefaultSMTP**。设备跟踪网关只能使用一个 SMTP 网关。必须启用该网关。
sync_gateway	*sync_gateway_name*	指定 SYNC 下级网关的名称。缺省值为 **DefaultSYNC**。设备跟踪网关只能使用一个 SYNC 网关。必须启用该网关。
udp_gateway	*udp_gateway_name*	指定 UDP 下级网关的名称。缺省值为 **DefaultUDP**。设备跟踪网关只能使用一个 UDP 网关。必须启用该网关。

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

SMTP 网关属性

SMTP 网关属性允许您更改 SMTP 网关的行为。server 属性是必需的；所有其它 SMTP 网关属性都是可选的。有关设置这些属性的详细信息，请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

属性	值	说明
confirm_action	{ **yes** | **no** }	指定是否通过此网关发送传送确认。缺省值为 **no**。
confirm_delivery	{ **yes** | **no** }	指定此网关是否确认传送。缺省值为 **no**。
confirm_timeout	*number*{ **s** | **m** | **h** }	指定确认超时之前等待的时间。以下是可接受的时间单位列表： ● **s**：表示秒。 ● **m**：表示分。 ● **h**：表示小时。 缺省值为 **1m**。可以在 *HH***h** *MM***m** *SS***s** 格式中组合时间单位。如果未指定时间单位，以秒为单位测量时间。
description	*description_text*	描述网关。
enable	{ **yes** | **no** }	指定是否应使用 SYNC 网关。
Listeners_are_900	{ **yes** | **no** }	指定是否所有监听器都是 Adaptive Server Anywhere 9.0.0 客户端。缺省值为 **no**。对于 Adaptive Server Anywhere 9.0.1 客户端或更高版本，此值保留为 **no**。
password	*password*	指定 SMTP 服务的口令。某些服务需要此属性。
sender	*SMTP_address*	指定 SMTP 推式通知的发送者地址。缺省值为 **anonymous**。
server	*IP_address_or_hostname*	指定用于向监听器发送消息的 SMTP 服务器的 IP 地址或主机名。缺省值为 **mail**。
user	*username*	指定 SMTP 服务的用户名。某些服务需要此属性。

SYNC 网关属性

SYNC 网关属性允许您更改 SYNC 网关的行为。所有 SYNC 网关属性都是可选的。有关设置这些属性的详细信息，请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

属性	值	说明
confirm_action	{ **yes** | **no** }	指定是否通过此网关发送传送确认。缺省值为 **no**。
confirm_delivery	{ **yes** | **no** }	指定此网关是否确认传送。缺省值为 **no**。
confirm_timeout	*number*{ **s** | **m** | **h** }	指定确认超时之前等待的时间。以下是可接受的时间单位列表： ● **s**：表示秒。 ● **m**：表示分。 ● **h**：表示小时。 缺省值为 **1m**。可以在 *HH***h** *MM***m** *SS***s** 格式中组合时间单位。如果未指定时间单位，以秒为单位测量时间。
description	*description_text*	描述网关。
enable	{ **yes** | **no** }	指定是否应使用 SYNC 网关。
Listeners_are_900	{ **yes** | **no** }	指定是否所有监听器都是 Adaptive Server Anywhere 9.0.0 客户端。缺省值为 **no**。对于 Adaptive Server Anywhere 9.0.1 客户端或更高版本，此值保留为 **no**。

UDP 网关属性

UDP 网关属性允许您更改 UPD 网关的行为，如 IP 地址和端口号。所有 UDP 网关属性都是可选的。有关设置这些属性的详细信息，请参见“用于服务器启动的同步的 MobiLink 服务器设置”第 31 页。

属性	值	说明
confirm_action	{ **yes** | **no** }	指定是否通过此网关发送传送确认。缺省值为 **no**。
confirm_delivery	{ **yes** | **no** }	指定此网关是否确认传送。缺省值为 **yes**。

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

属性	值	说明
confirm_timeout	*number*{ **s** | **m** | **h** }	指定确认超时之前等待的时间。以下是可接受的时间单位列表： ● **s**：表示秒。 ● **m**：表示分。 ● **h**：表示小时。 缺省值为 **1m**。可以在 *HH***h** *MM***m** *SS***s** 格式中组合时间单位。如果未指定时间单位，以秒为单位测量时间。
description	*description_text*	描述网关。
enable	{ **yes** | **no** }	指定是否应使用 UDP 网关。
Listeners_are_900	{ **yes** | **no** }	指定是否所有监听器都是 Adaptive Server Anywhere 9.0.0 客户端。缺省值为 **no**。对于 Adaptive Server Anywhere 9.0.1 客户端或更高版本，此值保留为 **no**。
Listener_port	*port_number*	指定远程设备用于发送 UDP 包的端口。缺省值为 **5001**。
sender	*IP_address_or_hostname*	仅用于多宿主机。指定发送者的 IP 地址或主机名。缺省值为 **localhost**。
sender_port	*port_number*	指定用于发送 UDP 包的端口号。缺省情况下，由操作系统随机指派空闲端口号。

运营公司属性

运营公司属性允许您更改无线运营公司配置的行为，此配置提供有关将自动跟踪的电话号码和网络提供商映射到电子邮件地址的信息。所有运营公司属性都是可选的，只有使用 SMTP 网关时才需要。

有关设置这些属性的详细信息，请参见“用于服务器启动的同步的 MobiLink 服务器设置” 第 31 页。

属性	值	说明
enable	{ **yes** | **no** }	指定是否应使用运营公司。
description	*description_text*	描述运营公司。
network_provider_id	*id_text*	指定网络提供商 ID。要在 Windows Mobile Phone Edition 上使用 SMS，将此属性设置为 **_generic_**。
sms_email_domain	*domain_name*	指定运营公司的域名。
sms_email_user_prefix	*prefix_name*	指定电子邮件地址中使用的前缀。

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

MobiLink 监听器实用程序

目录

用于 Windows 设备的监听器实用程序

该监听器实用程序在 Windows 设备上运行，包括 Windows Mobile 设备。

语法

dblsn [*options*] **-l** *message-handler* [**-l** *message-handler*...]

message-handler :
[*polling-option*;...] [*filter*;...]*action*; [*option*;...]

polling-option :
[**;poll_connect =** *string*]
[**;poll_notifier =** *string*]
[**;poll_key =** *string*]
[**;poll_every =** *number*]

option :
[**;continue = yes**]
[**;confirm_action = yes**]
[**;confirm_delivery = no**]
[**;maydial = no**]

filter :
[**subject =** *string*]
[**content =** *string*]
[**message =** *string* | **message_start =** *string*]
[**sender =** *string*]

action :
action = *command*[**;altaction =** *command*]

command :
START *program* [*program-arguments*]
| **RUN** *program* [*program-arguments*]
| **POST** *window-message* **TO** { *window-class-name* | *window-title* }
| *tcpip-socket-action*
| **DBLSN FULL SHUTDOWN**

tcpip-socket-action :
SOCKET port=*app-port*
[**;host=***app-host*]
[**;sendText=***text1*]
[**;recvText=** *text2* [**;timeout=***num-sec*]]

window-message : *string* | *message-id*

另请参见

- “用于 Palm 设备的实用程序”一节第 76 页
- “用于 Windows 的监听器选项”一节第 57 页
- “用于 Windows 的监听器关键字”一节第 68 页
- “用于 Windows 的监听器操作命令”一节第 70 页
- “用于 Windows 的监听器操作变量”一节第 73 页

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

用于 Windows 的监听库

该监听器附带 UDP 监听库 *lsn_udp.dll*（缺省情况下加载该监听库）。

使用 SMTP 网关时，必须指定 SMTP 监听库。可以使用 dblsn -d 选项指定库，使用 dblsn -a 选项指定库选项。

UDP (lsn_udp.dll)

以下是 UDP 监听库支持的选项列表：

选项	说明
Port=*port-number*	此选项指定要监听的端口号。缺省端口为 **5001**。
Timeout=*seconds*	此选项指定 UDP 监听端口上读取操作的最大阻塞时间。该值必须小于 UDP 监听线程的轮询间隔。缺省值为 **0**。
ShowSenderPort	此选项在所有出现的 $sender 操作变量中显示发送者端口号。缺省情况下，隐藏该端口号。指定此选项时，通过 **:***port-number* 语法将端口号追加到发送者地址的结尾。
HideWSAErrorBox	禁止错误框显示套接字操作上的错误。
CodePage=*number*	根据此数字将多字节字符转换为 Unicode。此选项仅适用于 Windows Mobile 设备。

另请参见

- “-d 选项”一节第 60 页
- “-a 选项”一节第 60 页

用于 Windows 的监听器选项

以下选项可用于配置监听器：

选项	说明
@{ *variable* | *filename* }	应用来自指定环境变量或文本文件的监听器选项。请参见“@ 选项”一节第 59 页。
-a *value*	为监听库指定单个库选项。请参见“-a 选项”一节第 60 页。
-d *filename*	指定监听库。请参见“-d 选项”一节第 60 页。
-e *device-name*	指定设备名称。请参见“-e 选项”一节第 61 页。

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

选项	说明
-f *string*	指定设备的额外信息。请参见“-f 选项”一节第 61 页。
-gi *seconds*	指定 IP 跟踪器轮询间隔。请参见“-gi 选项”一节第 61 页。
-i *seconds*	指定 SMTP 连接的轮询间隔。请参见“-i 选项”一节第 61 页。
-l "*keyword=value;...***"**	定义和创建消息处理程序。请参见“-l 选项”一节第 62 页。
-m	启用消息记录。请参见“-m 选项”一节第 62 页。
-ni	禁用 IP 跟踪。请参见“-ni 选项”一节第 62 页。
-ns	禁用 SMS 监听。请参见“-ns 选项”一节第 63 页。
-nu	禁用 UDP 监听。请参见“-nu 选项”一节第 63 页。
-o *filename*	将输出记录到文件中。请参见“-o 选项”一节第 63 页。
-os *bytes*	指定日志文件的最大大小。请参见“-os 选项”一节第 64 页。
-ot *filename*	截断一个文件，然后将输出记录到该文件。请参见“-ot 选项”一节第 64 页。
-p	允许设备在空闲时自动关机。请参见“-p 选项”一节第 64 页。
-pc { + | - }	启用或禁用持久性连接。请参见“-pc 选项”一节第 65 页。
-q	以安静模式运行监听器。请参见“-q 选项”一节第 65 页。
-r *filename*	标识消息过滤器的响应操作所涉及的远程数据库。请参见“-r 选项”一节第 65 页。
-sv *script-version*	指定用于验证的脚本版本。请参见“-sv 选项”一节第 66 页。
-t {+ | - } *name*	注册或注销远程数据库的远程 ID。请参见“-t 选项”一节第 66 页。

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

选项	说明
-u *username*	指定 MobiLink 用户名。请参见“-u 选项”一节第 66 页。
-v { **0** | **1** | **2** | **3** }	指定消息日志的详细程度级别。请参见“-v 选项”一节第 67 页。
-w *password*	指定 MobiLink 口令。请参见“-w 选项”一节第 67 页。
-x { **http** | **https** | **tcpip** } [**(** *protocol-option=value;...* **)**]	指定网络协议和 MobiLink 服务器协议选项。请参见“-x 选项”一节第 68 页。
-y *newpassword*	指定新的 MobiLink 口令。请参见“-y 选项”一节第 68 页。

@ 选项

应用来自指定环境变量或文本文件的监听器选项。

语法

dblsn @{ *variable* | *filename* } ...

注释

缺省情况下，*dblsn.txt* 是监听器在未指定参数的情况下运行时的参数文件。

如果存在同名的文件和环境变量，则使用环境变量。

使用文件隐藏实用程序对口令和文本文件中的其它敏感信息进行模糊处理。请参见“文件隐藏实用程序 (dbfhide)”一节 《SQL Anywhere 服务器 - 数据库管理》。

另请参见

- “使用配置文件”一节 《SQL Anywhere 服务器 - 数据库管理》

示例

示例文本文件位于 ***samples-dir****MobiLink**SIS_SimpleListener**dblsn.txt*。

以下示例将命令行选项存储在 **dblsnoptions** 环境变量中：

```
dblsn @dblsnoptions
```

以下示例将命令行选项存储在完全限定的文本文件 **mydblsn.txt** 中：

```
dblsn @mydblsn.txt
```

-a 选项

为监听库指定单个库选项。

语法

dblsn -a *value* ...

注释

缺省情况下，如果未指定库，则监听器使用 *lsn_udp.dll*。使用 -d 选项指定替代库或附加库。

使用 **?** 值查看所有可用库选项。

多次使用 -a 选项设置附加库选项。

另请参见

- “用于 Windows 的监听库”一节第 57 页
- “-d 选项”一节第 60 页

示例

以下示例在 *lsn_udp.dll* 监听库中指定 port 选项并声明 ShowSenderPort 选项：

```
dblsn -d lsn_udp.dll -a port=1234 -a ShowSenderPort
```

以下示例为两个不同的库指定 port 选项：

```
dblsn -d lsn_udp.dll -a port=1234 -d maac750.dll -a port=2345
```

以下示例显示缺省库中的所有可用库选项：

```
dblsn -a ?
```

-d 选项

指定监听库。

语法

dblsn -d *filename* ...

注释

缺省情况下，监听器使用 *lsn_udp.dll* 监听库。

多次使用 -d 选项启用多通道监听，这将启用对多个介质的监听。

另请参见

- “用于 Windows 的监听库”一节第 57 页

示例

以下示例指定 *maac750.dll* 监听库：

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

```
dblsn -d maac750.dll
```

-e 选项

指定设备名称。

语法

dblsn -e *device-name* ...

注释

缺省情况下，由操作系统自动生成设备名称。

连接到 MobiLink 服务器时，请确保所有设备名称唯一。

可在监听器窗口中看到设备名称。

-f 选项

指定设备的额外信息。

语法

dblsn -f *string* ...

注释

缺省情况下，此信息是在设备上运行的操作系统的版本号。

-gi 选项

指定 IP 跟踪器轮询间隔。

语法

dblsn -gi *number* ...

注释

缺省情况下，IP 跟踪器每隔 60 秒轮询一次。

-i 选项

指定 SMTP 连接的轮询间隔。

语法

dblsn -i *number* ...

注释

-i 选项指定监听器检查消息的频率。

对于 SMTP 连接，缺省值为 30 秒。对于 UDP 连接，监听器立即连接。

对于使用 -d 选项指定的每个监听库，-i 选项只能使用一次。

另请参见

- “-d 选项”一节第 60 页

示例

以下示例为两个不同的库指定轮询间隔：

```
dblsn -d lsn_udp.dll -i 60 -d maac750.dll -i 45
```

-l 选项

定义和创建消息处理程序。

语法

dblsn -l "*keyword=value;...***"** ...

注释

有关可接受关键字的列表，请参见“用于 Windows 的监听器关键字”一节第 68 页。

多次使用 -l 选项为推式通知定义附加消息处理程序。消息处理程序按指定的顺序处理。

另请参见

- “消息处理程序”一节第 15 页

-m 选项

启用消息记录。

语法

dblsn -m ...

注释

缺省情况下，关闭消息记录。

-ni 选项

禁用 IP 跟踪。

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

语法

dblsn -ni ...

注释

缺省情况下，启用 IP 跟踪。

此选项不停止传送确认。

-ni 选项与 -x 选项配合使用时，将禁用 UDP 地址跟踪。如果想要设备跟踪排除 UDP 地址更新，此功能很有用。

另请参见

- “-x 选项”一节第 68 页

-ns 选项

禁用 SMS 监听。

语法

dblsn -ns ...

注释

缺省情况下，对于 Windows Mobile 2003 和更高版本启用 SMS 监听。

-nu 选项

禁用 UDP 监听。

语法

dblsn -nu ...

注释

缺省情况下，启用 UDP 监听。

-o 选项

将输出记录到文件中。

语法

dblsn -o *filename* ...

注释

缺省情况下，将输出记录到监听器窗口。

另请参见

- “-ot 选项”一节第 64 页

-os 选项

指定日志文件的最大大小。

语法

dblsn -os *bytes* ...

注释

缺省情况下无最大值限制。最小大小限制为 10000。

-ot 选项

截断一个文件，然后将输出记录到该文件。

语法

dblsn -ot *filename* ...

注释

在输出被记录之前，删除文件内容。

另请参见

- “-o 选项”一节第 63 页

-p 选项

允许设备在空闲时自动关机。

语法

dblsn -p ...

注释

缺省情况下，监听器阻止设备关机。

此选项仅适用于 Windows Mobile 设备。

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

-pc 选项

语法

启用或禁用持久性连接。

dblsn -pc { **+** | **-** } ...

注释

缺省情况下，启用持久性连接。- 标志禁用持久性连接；+ 标志启用持久性连接。

禁用持久性连接将阻止监听器接收推式通知，但允许短暂的持久性连接以进行设备跟踪和确认。

如果持久性连接断开，监听器将不断地尝试重新连接。

示例

以下示例禁用持久性连接：

```
dblsn -pc-
```

-q 选项

以安静模式运行监听器。

语法

dblsn -q ...

注释

-q 选项将最小化监听器窗口。缺省情况下，显示监听器窗口。

-r 选项

标识消息过滤器的响应操作所涉及的远程数据库。

语法

dblsn -r *filename* ...

注释

filename 必须包含 RID 文件的完整路径。此文件在第一次同步之后由 dbmlsync 自动创建。它使用与数据库文件相同的位置和名称。对于 UltraLite 数据库，*filename* 应与数据库名称相同。

应用 -r 选项时，可以在消息处理程序中使用 $remote_id 操作变量来引用 RID 文件中的远程 ID。缺省情况下，远程 ID 是 GUID。

多次使用 -r 选项标识多个数据库。

另请参见

- “使用消息处理程序”一节第 16 页
- “用于 Windows 的监听器操作命令”一节第 70 页

-sv 选项

指定用于验证的脚本版本。

语法

dblsn -sv *script-version ...*

注释

缺省情况下，监听器使用 ml_global 服务器脚本版本（如果已定义）。

-t 选项

注册或注销远程数据库的远程 ID。

语法

dblsn -t { **+** | **-** } *name ...*

注释

+ 标志注册远程 ID；- 标志注销 ID。

注册允许监听器通过引用远程 ID 对推式通知进行寻址。

成功上载设备跟踪信息后，已注册 ID 保留在服务器上的 ml_listening 系统表中。您只需注册 ID 一次。

多次使用 -t 选项注册或注销多个 ID。注册多个 ID 对于发送到多个远程数据库的推式通知的寻址非常有用。

另请参见

- “用于 Windows 的监听器操作命令”一节第 70 页

-u 选项

为监听器指定 MobiLink 用户名。

语法

dblsn -u *username ...*

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

注释

缺省情况下，用户名是 *device-name*-**dblsn**，其中 *device-name* 是设备名称。可以使用 -e 选项指定设备名称。

监听器使用该用户名连接到 MobiLink 服务器进行设备跟踪、确认和持久性连接。

该用户名必须是注册到 MobiLink 服务器的唯一 MobiLink 用户名。该名称必须存在于统一数据库上的 ml_user 系统表中。

另请参见

- “-e 选项”一节第 61 页
- “-w 选项”一节第 67 页
- “创建和注册 MobiLink 用户”一节 《MobiLink - 客户端管理》

-v 选项

指定消息日志的详细程度级别。

语法

dblsn -v { 0 | 1 | 2 | 3 } ...

注释

缺省情况下，详细程度级别设置为 0。

下表概括了可能的详细程度级别值。

详细程度级别	说明
0	关闭详细显示。
1	显示监听库消息、基本操作跟踪步骤和命令行选项。
2	显示详细程度级别 1 消息，以及详细的操作跟踪步骤。
3	显示详细程度级别 2 消息、轮询状态和监听状态。

必须使用 -m 选项将推式通知输出到消息日志。

另请参见

- “-m 选项”一节第 62 页
- “将数据库服务器消息记录到文件”一节 《SQL Anywhere 服务器 - 数据库管理》

-w 选项

指定 MobiLink 口令。

语法

dblsn -w *password* ...

注释

必须将口令注册到 MobiLink 服务器上关联的 MobiLink 用户名下。

监听器使用该口令连接到 MobiLink 服务器进行设备跟踪、确认和持久性连接。

另请参见

- “-u 选项”一节第 66 页

-x 选项

指定网络协议和 MobiLink 服务器协议选项。

语法

dblsn -x { **http** | **https** | **tcpip** } [(*protocol-option*=*value*;...)] ...

注释

监听器必须连接到 MobiLink 服务器才能向统一数据库发送设备跟踪信息和传送确认。使用 **host** 协议选项指定 MobiLink 服务器的位置。请参见“host”一节 《MobiLink - 客户端管理》。

-x 选项允许设备在服务器地址更改时更新统一数据库。

另请参见

- “MobiLink 客户端网络协议选项” 《MobiLink - 客户端管理》

-y 选项

指定新的 MobiLink 口令。

语法

dblsn -y *newpassword* ...

注释

如果验证系统不允许远程设备更改口令，则 -y 选项不适用。

用于 Windows 的监听器关键字

以下关键字可用于配置使用 dblsn -l 选项创建的消息处理程序。有关使用监听器关键字的详细信息，请参见“-l 选项”一节第 62 页。

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

过滤器关键字

以下关键字用于过滤推式通知中的消息：

关键字语法	说明
subject=*subject-string*	如果消息主题在文字上与 *subject-string* 相同，则过滤出该消息。
content=*content-string*	如果消息内容在文字上与 *content-string* 相同，则过滤出该消息。
message=*message-string*	如果整条消息在文字上与 *message-string* 相同，则过滤出该消息。
message_start=*message-string*	如果消息以 *message-string* 开头，则过滤出该消息。
sender=*sender-string*	如果消息由 *sender-string* 发送，则过滤出该消息。

操作关键字

以下关键字用于在满足过滤器条件时启动操作：

关键字语法	说明
action=*command*	指定操作命令。请参见“用于 Windows 的监听器操作命令”一节第 70 页。
altaction=*command*	指定操作命令失败时启动的替代操作命令。请参见“用于 Windows 的监听器操作命令”一节第 70 页。

轮询选项

以下选项用于配置轻量级轮询器：

关键字语法	说明
poll_connect= { **http** | **https** | **tcpip** } [(*protocol-option*=*value*;...)]	指定连接到服务器所需的轻量级网络协议选项。缺省值从 dblsn -x 选项继承。请参见“-x 选项”一节第 68 页。
poll_notifier=*Notifier-string*	指定处理推式请求的通告程序的名称。必需。
poll_key=*key-string*	指定监听器用于向通告程序标识自己的名称。此值必须唯一。必需。

关键字语法	说明
poll_every=*seconds-number*	指定监听器轮询服务器的频率。时间间隔以秒为单位测量。缺省值从 MobiLink 服务器自动检索。

选项

以下选项可用于配置消息处理程序行为：

关键字语法	说明
continue=[**yes** | **no**]	指定监听器在发现第一个匹配项后是否继续监听。缺省值为 **no**。在指定多个过滤器时 **yes** 值很有用，此时一个消息启动多项操作。
confirm_action=[**yes** | **no**]	指定过滤器是否确认操作。缺省值为 **yes**。
confirm_delivery=[**yes** | **no**]	指定过滤器是否确认符合条件的消息传送。缺省值为 **yes**，所以第一个过滤器接受消息后发送传送确认。 只有在消息需要确认且过滤器接受消息时才能确认传送。如果指定的网关将 confirm_delivery 关键字值设置为 **yes**，则消息需要确认。在多个过滤器接受同一条消息时，可使用 **no** 值，以使您更精确地控制哪个过滤器确认传送。有关在服务器上处理传送确认的信息，请参见“confirmation_handler 事件”一节第 43 页。
maydial=[**yes** | **no**]	指定操作是否拥有调制解调器的拨号权限。缺省值为 **yes**。**no** 值会使监听器在执行操作之前释放调制解调器。

另请参见

- “消息处理程序”一节第 15 页
- “用于 Windows 的监听器操作命令”一节第 70 页

用于 Windows 的监听器操作命令

配置新的消息处理程序时指定操作。如果满足过滤器条件，则启动某操作。如果操作失败，则启动替代操作。使用 **action** 关键字定义操作；使用 **altaction** 关键字定义替代操作。有关监听器用法的详细信息，请参见“用于 Windows 的监听器关键字”一节第 68 页。

以下列出操作命令：

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

命令	说明
START *program arglist*	在监听器于后台运行期间启动某程序。请参见“START 操作命令”一节第 71 页。
RUN *program arglist*	暂停监听器以运行某程序。请参见“RUN 操作命令”一节第 72 页。
POST *windowmessage* | *id* to *windowclass* | *windowtitle*	将窗口消息发布到窗口类。请参见“POST 操作命令”一节第 72 页。
SOCKET port=*windowname*[;**host=***hostname*] [;**sendText=***text*] [;**recvText=***text*[;**timeout=***seconds*]]	使用 TCP/IP 连接向应用程序发送消息。请参见“SOCKET 操作命令”一节第 73 页。
DBLSN FULL SHUTDOWN	强制监听器关闭。请参见“DBLSN FULL SHUTDOWN 操作命令”一节第 73 页。

每个 **action** 或 **altaction** 关键字只能指定一个操作。如果想要某操作执行多个任务，则创建一个包含多个命令的批处理文件，并使用 **RUN** 操作命令运行该文件。

另请参见

- “启动操作”一节第 17 页

START 操作命令

在监听器于后台运行期间启动某程序。

语法

action='START *program arglist*'

注释

启动某程序时，监听器将继续监听更多推式通知。

监听器不等待程序完成，因此，监听器只能确定操作命令是否失败，或者监听器是否无法启动指定程序。

示例

以下示例通过某些命令行选项启动 dbmlsync，其中部分选项是使用 $content 操作变量从各消息获取的：

```
dblsn -l "action='start dbmlsync.exe @dbmlsync.txt -n
  $content -wc dbmlsync_$content -e sch=INFINITE';"
```

RUN 操作命令

暂停监听器以运行某程序。

语法

action='RUN *program arglist*'

注释

监听器等待程序完成，然后恢复监听。

运行某程序时，监听器将确定程序是否失败、监听器是否无法启动程序，或者程序是否返回非零返回代码。

示例

以下示例通过某些命令行选项启动 dbmlsync，其中部分选项是使用 $content 操作变量从消息获取的：

```
dblsn -l "action='run dbmlsync.exe @dbmlsync.txt -n $content';"
```

POST 操作命令

将窗口消息发布到窗口类。

语法

action='POST *windowmessage* | *id* to *windowclass* | *windowtitle*'

注释

可使用 POST 命令通过信号通知使用窗口消息的应用程序。

可以按消息内容或窗口消息 ID（如果存在此 ID）识别窗口消息。

可以按窗口类名称或窗口标题识别窗口类。如果按名称识别，可以使用 -wc dbmlsync 选项指定窗口类名称。如果按窗口标题识别窗口类，则只能通过顶层窗口引用它。

如果窗口消息或窗口类名称中包含非字母数字字符（如空格或标点符号），则用单引号 (') 将文本括起来。转义字符也是单引号，因此，如果窗口消息或窗口类名称中包含单引号，请使用两个单引号 (') 表示该引号。

示例

为演示如何使用包含单引号的窗口消息，以下示例将 mike's_message 窗口消息发布到 mike's_class 窗口类：

```
dblsn -l "action='post mike''s_message to mike''s_class';"
```

以下示例将窗口消息 dbas_synchronize 发布到使用 dbmlsync_FullSync 类名注册的 dbmlsync 实例：

```
dblsn -l "action='post dbas_synchronize to dmblsync_FullSync';"
```

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

另请参见

- “-wc 选项”一节 《MobiLink - 客户端管理》

SOCKET 操作命令

使用 TCP/IP 连接向应用程序发送消息。

语法

action='SOCKET port=*windowname*[;**host=***hostname*][;**sendText=***text*]
[;**recvText=***text*[;**timeout=***seconds*]]'

注释

SOCKET 命令用于将动态信息传递到运行中的应用程序，以及将消息集成到 Java 应用程序和 Visual Basic 应用程序中。这两种语言都不支持自定义窗口消息传送，eMbedded Visual Basic 不支持命令行参数。

要连接到套接字，必须指定端口和主机。使用 sendText 输入消息。

确认应用程序已成功接收 sendText 时，使用 recvText 显示消息。使用 recvText 时，可以指定超时限制。如果监听器在超时限制内无法连接、发送确认或接收确认，则操作失败。

示例

以下示例将 $sender=$message 中的字符串转发到正在监听端口 12345 的本地应用程序。监听器预计应用程序在 5 秒内发送 "beeperAck" 作为确认。

```
dblsn -l "action='socket port=12345;
   sendText=$sender=$message;
   recvText=beeperAck;
   timeout=5'"
```

DBLSN FULL SHUTDOWN 操作命令

强制监听器关闭。

语法

action='DBLSN FULL SHUTDOWN'

注释

监听器关闭后，会停止处理推式通知和停止同步设备跟踪信息。要继续进行服务器启动的同步，必须重新启动监听器。

用于 Windows 的监听器操作变量

以下操作变量可用于操作或过滤器。在启动消息处理程序之前，将该值代入操作变量。

操作变量以美元符号 ($) 开头。转义字符也是美元符号，所以使用两个美元符号 ($$) 将单个美元符号指定为纯文本。

变量	说明
$subject	消息的主题。
$content	消息的内容。
$message	整条消息，包括主题、内容和发送者。
$message_start	消息的开始部分，由 message_start 过滤器关键字指定。只有指定了 message_start 过滤器关键字后，此变量才可用。请参见“用于 Windows 的监听器关键字”一节第 68 页。
$message_end	除了 message_start 关键字外的一部分消息。只有指定了 message_start 过滤器关键字后，此变量才可用。请参见“用于 Windows 的监听器关键字”一节第 68 页。
$ml_connect	MobiLink 网络协议选项，由 dblsn -x 选项指定。缺省值为空字符串。请参见“-x 选项”一节第 68 页。
$ml_user	MobiLink 用户名，由 dblsn -u 选项指定。缺省用户名为 *device-name*-**dblsn**。
$ml_password	MobiLink 口令（由 dblsn -w 选项指定）或新的 MobiLink 口令（如果使用 -y 选项）。
$priority	此变量的意义取决于运营公司库。
$request_id	在推式请求中指定的请求 ID。请参见“推式请求”一节第 8 页。
$remote_id	远程 ID。此变量仅在指定 dblsn -r 选项时使用。请参见“按远程 ID 过滤消息”一节第 19 页。
$sender	消息的发送者。
$type	此变量的意义取决于运营公司库。
$year	此变量的意义取决于运营公司库。
$month	此变量的意义取决于运营公司库。值可以从 1 到 12。
$day	此变量的意义取决于运营公司库。值可以从 1 到 31。
$hour	此变量的意义取决于运营公司库。值可以从 0 到 23。
$minute	此变量的意义取决于运营公司库。值可以从 0 到 59。

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

变量	说明
$second	此变量的意义取决于运营公司库。值可以从 0 到 59。
$best_adapter_mac	到达 dblsn -x 选项指定的 MobiLink 服务器所使用的最佳 NIC 的 MAC 地址。如果最佳路径不通过 NIC，则该值为空字符串。
$best_adapter_name	到达 dblsn -x 选项指定的 MobiLink 服务器所使用的最佳 NIC 的适配器名称。如果最佳路径不通过 NIC，则该值为空字符串。
$best_ip	到达 dblsn -x 选项指定的 MobiLink 服务器所使用的最佳 IP 接口的 IP 地址。如果服务器无法到达，则该值为 0.0.0.0。
$best_network_name	到达 dblsn -x 选项指定的 MobiLink 服务器所使用的最佳配置文件的 RAS 或拨号配置文件的名称。如果最佳路线不通过 RAS/拨号连接，则该值为空字符串。
$adapters	活动网络适配器名称列表，各个名称由竖线 (|) 分隔。
$network_names	连接的 RAS 条目名称列表，各个名称由竖线 (|) 分隔。RAS 条目名称有时指拨号网络 (DUN) 的拨号条目名称。
$poll_connect	MobiLink 网络协议选项，由 poll_connect 轮询关键字指定。缺省值为空字符串。请参见“用于 Windows 的监听器关键字”一节第 68 页。
$poll_notifier	通告程序名称，由 poll_notifier 轮询关键字指定。请参见“用于 Windows 的监听器关键字”一节第 68 页。
$poll_key	轮询键，由 poll_key 轮询关键字指定。请参见“用于 Windows 的监听器关键字”一节第 68 页。
$poll_every	轮询频率，由 poll_every 轮询关键字指定。请参见“用于 Windows 的监听器关键字”一节第 68 页。

另请参见

- “-l 选项”一节第 62 页
- “用于 Windows 的监听器关键字”一节第 68 页
- “用于 Windows 的监听器操作命令”一节第 70 页

示例

以下示例使用 $message_end 操作变量确定要同步哪个发布：

```
dblsn -l "message_start=start-of-message;action='run dbmlsync.exe -c ... -n
$message_end'"
```

用于 Palm 设备的实用程序

运行服务器启动的同步需要以下两个实用程序：

- 用于 Palm 设备的监听器配置实用程序 (dblsncfg)
- 用于 Palm 设备的监听器实用程序 (LsnT650.prc)

要准备 Palm 监听器，在 Windows 桌面上运行监听器配置实用程序以创建一个用于 Palm 的配置文件，然后使用 HotSync 将该文件传送到设备。

注意
在 Windows 桌面上运行监听器配置实用程序以生成用于 Palm 设备的配置文件时，必须指定 **RUN** 操作命令。但是，在 Palm 设备上，可以使用监听器中的处理程序编辑器删除 **RUN** 操作。此方法允许您使用消息但不启动操作。

用于 Palm 设备的监听器配置实用程序

在 Windows 桌面上运行时，监听器配置实用程序会创建一个用于 Palm 设备的配置文件。

语法

dblsncfg -n [*filename*] **-l** *message-handler* [**-l** *message-handler*...]

message-handler : [*filter*;...] *action*

filter :
[**subject =** *string*]
[**content =** *string*]
[**message =** *string* | **message_start =** *string*]
[**sender =** *string*]

action : **action=run** *application-name* [*arguments*]

另请参见

- “用于 Palm 的监听器选项”一节第 76 页
- “用于 Palm 的监听器关键字”一节第 78 页
- “用于 Palm 的监听器操作变量”一节第 79 页

用于 Palm 的监听器选项

以下选项可用于配置监听器：

选项	说明
@{ *variable* | *filename* }	应用来自指定环境变量或文本文件的选项。请参见“@ 选项”一节第 77 页。

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

选项	说明
-l "*keyword=value;...***"**	定义和创建消息处理程序。请参见“-l 选项”一节第 77 页。
-n *filename*	创建 PDB 配置文件。请参见“-n 选项”一节第 78 页。

@ 选项

应用来自指定环境变量或文本文件的选项。

语法

dblsncfg @{ *variable* | *filename* **}** ...

注释

缺省情况下，监听器在未指定参数的情况下运行，*dblsncfg.txt* 被用作参数文件。

如果存在同名的文件和环境变量，则使用环境变量。

使用文件隐藏实用程序对口令和文本文件中的其它敏感信息进行模糊处理。请参见“文件隐藏实用程序 (dbfhide)”一节 《SQL Anywhere 服务器 - 数据库管理》。

另请参见

- “使用配置文件”一节 《SQL Anywhere 服务器 - 数据库管理》

示例

示例文本文件位于 ***samples-dir**MobiLink**SIS_SimpleListener**dblsn.txt***。

以下示例将命令行选项存储在 **dblsnoptions** 环境变量中：

```
dblsn @dblsnoptions
```

以下示例将命令行选项存储在完全限定的文本文件 **mydblsn.txt** 中：

```
dblsn @mydblsn.txt
```

-l 选项

定义和创建消息处理程序。

语法

dblsncfg -l "*keyword=value;...***"** ...

注释

有关可接受关键字的列表，请参见“用于 Palm 的监听器关键字”一节第 78 页。

多次使用 -l 选项为推式通知定义附加消息处理程序。消息处理程序按指定的顺序处理。

另请参见

- “消息处理程序”一节第 15 页

-n 选项

创建 PDB 配置文件。

语法

dblsncfg -n *filename* ...

注释

缺省情况下，监听器会创建 *lsncfg.pdb* 配置文件。

用于 Palm 的监听器关键字

以下关键字可用于配置使用 dblsncfg -l 选项创建的消息处理程序。有关监听器关键字用法的详细信息，请参见“-l 选项”一节第 77 页。

过滤器关键字

以下关键字用于过滤推式通知中的消息：

关键字语法	说明
subject=*subject-string*	如果消息主题在文字上与 *subject-string* 相同，则过滤出该消息。
content=*content-string*	如果消息内容在文字上与 *content-string* 相同，则过滤出该消息。
message=*message-string*	如果整条消息在文字上与 *message-string* 相同，则过滤出该消息。
message_start=*message-string*	如果消息以 *message-string* 开头，则过滤出该消息。
sender=*sender-string*	如果消息由 *sender-string* 发送，则过滤出该消息。

操作关键字

以下关键字用于在满足过滤器条件时启动操作：

关键字语法	说明
action=*command*	指定操作命令。请参见“用于 Palm 的监听器操作变量”一节第 79 页。

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

关键字语法	说明
altaction=*command*	指定操作命令失败时启动的替代操作命令。请参见 “用于 Palm 的监听器操作变量”一节第 79 页。

操作和替代操作关键字

配置新的消息处理程序时指定操作。如果满足过滤器条件，则启动某操作。如果操作失败，则启动替代操作。使用 **action** 关键字定义操作；使用 **altaction** 关键字定义替代操作。

每个 **action** 或 **altaction** 关键字只能指定一个操作。如果想要某操作执行多个任务，则创建一个包含多个命令的批处理文件，并使用 **RUN** 操作命令运行该文件。

监听器配置实用程序支持 **RUN** *program arglist* 命令，其中 *program* 是要运行的程序的名称，*arglist* 是取决于应用程序的字符串。

以下示例说明如何使用监听器配置实用程序通过操作命令运行 yourapp.exe：

```
dblsncfg -l "action='RUN yourapp.exe';"
```

目标应用程序的 PilotMain 例程应将字符串作为命令块。

注意
在 Windows 桌面上运行监听器配置实用程序以生成用于 Palm 设备的配置文件时，必须指定 **RUN** 操作命令。但是，在 Palm 设备上，可以使用监听器中的处理程序编辑器删除 **RUN** 操作命令。此方法允许您使用消息但不启动操作。

另请参见

- “使用消息处理程序”一节第 16 页
- “用于 Palm 的监听器操作变量”一节第 79 页

用于 Palm 的监听器操作变量

以下操作变量可用于操作或过滤器。在启动消息处理程序之前，将该值代入操作变量。

操作变量以美元符号 ($) 开头。转义字符也是美元符号，所以使用两个美元符号 ($$) 将单个美元符号指定为纯文本。

操作变量	说明
$subject	消息的主题。
$content	消息的内容。
$message	整条消息，包括主题、内容和发送者。

操作变量	说明
$message_start	消息的开始部分，由 message_start 过滤器关键字指定。只有指定了 message_start 过滤器关键字后，此变量才可用。请参见“用于 Palm 的监听器关键字”一节第 78 页。
$message_end	除了 message_start 关键字外的一部分消息。只有指定了 message_start 过滤器关键字后，此变量才可用。请参见“用于 Palm 的监听器关键字”一节第 78 页。
$sender	消息的发送者。
$time	这指自 1904 年 1 月 1 日上午 12:00 至今的秒数。

另请参见

- “-l 选项”一节第 77 页
- “用于 Palm 的监听器关键字”一节第 78 页

用于 Palm 设备的监听器实用程序

监听器部署

要使 Palm 应用程序使用服务器启动的同步，必须将监听器安装在设备上。以下列出必须部署的监听器文件：

- **LsnT650.prc** Treo 650 上的监听器。
- **配置文件** 必须在 Windows 桌面上使用监听器配置实用程序创建配置文件。当前，监听器仅从 *lsncfg.pdb* 配置文件中读取数据。

Treo 设备无需打开，监听器即可运行。

监听器选项

监听器允许您设置三个选项。除非这些选项被显式更改或执行重置，否则它们将一直有效。

- **Listening** 一种禁止监听器监听消息的方式。
- **Enable Actions** 仅当启用监听时，此选项才适用。禁用操作时，不启动操作。
- **Prompt Before Actions** 仅当启用操作时该选项才可用。设置此选项时，在启动操作之前将弹出一个确认窗口。

其它 Palm 支持

提供用于 Palm 设备的 MobiLink 监听器 C API 是为了支持其它 Palm 设备。请参见“用于 Palm 设备的 MobiLink 监听器 C API”第 81 页。

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

用于 Palm 设备的 MobiLink 监听器 C API

目录

用于 Palm 设备的 MobiLink 监听器 C API 是一个精简型编程接口，其中包含消息处理方法和设备相关方法，可用于为新的 Palm 设备和无线网络适配器创建监听器。

监听器 API 文件

以下列出了运行 API 所需的文件，以及 Palm 设备示例实现。所有目录都相对于 *install-dir* 目录。

文件名或位置	说明
SDK\Listener\Palm\68k\cw\lib\PalmLsn.lib	运行时库，其中包括消息处理例程、监听器控件和处理程序编辑器。
SDK\Listener\Palm\68k\cw\rsc	包含 UI 资源。

文件名或位置	说明
SDK\Listener\Palm\src\PalmLsn.h	运行时库头文件和 API。
SDK\Listener\Palm\src\Treo650.c	Treo 650 实现。

消息处理方法

监听器 API 使用 a_palm_msg 结构表示监听器消息。以下方法用于分配和处理 a_palm_msg 实例：

方法	说明
“PalmLsnAllocate 方法”一节 第 86 页	返回一个新的 a_palm_msg 实例。
“PalmLsnDupMessage 方法”一节 第 88 页	初始化 a_palm_msg 实例的消息字段值。
“PalmLsnDupSender 方法”一节 第 89 页	初始化 a_palm_msg 实例的发送者字段。
“PalmLsnDupTime 方法”一节 第 90 页	初始化 a_palm_msg 实例的时间字段。
“PalmLsnFree 方法”一节第 91 页	释放消息内存资源。
“PalmLsnProcess 方法”一节 第 96 页	根据配置数据库中的记录处理消息。

设备相关方法

以下方法提供标识、注册和事件处理功能：

方法	说明
“PalmLsnCheckConfigDB 方法”一节第 87 页	报告监听器配置数据库中的错误。
“PalmLsnGetConfigFileName 方法”一节第 92 页	返回包含监听器配置数据库名称的字符串。
“PalmLsnNormalHandleEvent 方法”一节第 93 页	处理应用程序事件。
“PalmLsnNormalStart 方法”一节 第 94 页	当监听器应用程序启动时提供自定义操作。

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

方法	说明
“PalmLsnNormalStop 方法”一节第 95 页	当监听器应用程序退出事件循环时提供自定义操作。
“PalmLsnSpecialLaunch 方法”一节第 98 页	响应可能与设备相关的启动代码。
“PalmLsnTargetCompanyID 方法”一节第 101 页	返回设备的公司或制造商 ID。
“PalmLsnTargetDeviceID 方法”一节第 102 页	返回目标设备 ID。

LsnMain 方法

提供监听器库 *PalmLsn.lib* 的主入口点。

语法

```
UInt32 LsnMain(
   UInt16 cmd,
   MemPtr cmdPBP,
   UInt16 launchFlags
)
```

参数

- **cmd** Palm OS 应用程序启动代码。
- **cmdPBP** 指向包含启动代码参数的结构的指针。如果应用程序不具有任何启动命令特有的参数，该值为 NULL。
- **launchFlags** 提供有关启动的额外信息的标志。

返回值

Palm OS 错误代码。如果监听器库成功处理了启动代码，则方法返回 errNone。

注释

传递给 LsnMain 的值类似于传递给作为 Palm OS 应用程序主入口点的 PilotMain 的启动代码参数。

有关这些参数的详细信息，请参考《Palm OS 参考》。

另请参见

- “PalmLsnProcess 方法”一节第 96 页
- “用于 Palm 设备的 MobiLink 监听器 C API”第 81 页

示例

以下示例（用于 Treo 650 smartphone 实现）将启动代码参数传递给监听器应用程序主入口点中的 LsnMain：

```
UInt32 PilotMain(UInt16 cmd, MemPtr cmdPBP, UInt16 launchFlags ) {
    return(LsnMain(cmd, cmdPBP, launchFlags));
}
```


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

palm_lsn_ret 枚举

palm_lsn_ret 枚举指定可能的返回代码。

语法

```
typedef enum {
    PalmLsnOk = errNone,
    PalmLsnMissingConfig = appErrorClass,
    PalmLsnProblemReadingConfig,
    PalmLsnProblemParsingCmd,
    PalmLsnOutOfMemory,
    PalmLsnUnrecognizedAction,
    PalmLsnRunMissingApp
} palm_lsn_ret;
```

成员

名称	说明
PalmLsnOk	指明成功的方法调用。此值包含与 errNone（指明无错误）相同的值。
PalmLsnMissingConfig	指明缺失的监听器配置数据库。此字段包含与 appErrorClass 错误代码（指明应用程序定义的错误）相同的值。
PalmLsnProblemReadingConfig	指明读取监听器配置数据库时出现错误。
PalmLsnProblemParsingCmd	指明无法处理监听器配置数据库内存储的命令。
PalmLsnOutOfMemory	指明由于在分配用于消息处理的内存时出现错误而使方法运行未完成。
PalmLsnUnrecognizedAction	指明监听器不支持监听器配置数据库中指定的操作。
PalmLsnRunMissingApp	指明监听器无法启动 run 操作命令指定的应用程序。

另请参见

- “PalmLsnProcess 方法”一节第 96 页

PalmLsnAllocate 方法

返回一个新的 a_palm_msg 实例。

语法

struct a_palm_msg * **PalmLsnAllocate( )**

返回值

所有字段均初始化为 0 的新的 a_palm_msg 实例。

另请参见

- “PalmLsnFree 方法”一节第 91 页

示例

以下示例使用 PalmLsnAllocate 分配一个 a_palm_msg 实例：

```
a_palm_msg * ulMsg;

// Allocate a message structure
ulMsg = PalmLsnAllocate();
```

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

PalmLsnCheckConfigDB 方法

报告监听器配置数据库中的错误。

语法

```
palm_lsn_ret PalmLsnCheckConfigDB(
 Char const *  cfg,
 UInt16 * const rec
)
```

参数

- **cfg** 包含配置数据库名称的字符数组。要获得配置数据库名称，请使用 PalmLsnGetConfigFileName 方法。请参见“PalmLsnGetConfigFileName 方法”一节第 92 页。
- **rec** 标识配置数据库中有问题或格式有误的记录的索引的输出参数。

返回值

palm_lsn_ret 枚举中列出的返回代码之一。请参见“palm_lsn_ret 枚举”一节第 85 页。

注释

可以使用此方法检测打开配置数据库或读取其记录时发生的错误。

另请参见

- “PalmLsnProcess 方法”一节第 96 页

示例

以下示例使用 PalmLsnCheckConfigDB 检测配置数据库中有问题的或残缺的记录：

```
Err     ret;
UInt16  badRec;
Char    configDb[dmDBNameLength];

// Get configuration database name
PalmLsnGetConfigFileName(configDb);

// check for errors in the configuration database
ret = PalmLsnCheckConfigDB(configDb, &badRec);
if (ret!=errNone) {
    // handle error
}
```

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

PalmLsnDupMessage 方法

初始化 a_palm_msg 实例的消息字段值。

语法

```
Err PalmLsnDupMessage(
   struct a_palm_msg * const msg,
   Char const * message
)
```

参数

- **msg** 指向 a_palm_msg 实例的指针。
- **message** 包含源消息文本的输入参数。

返回值

Palm OS 错误代码。errNone 表示操作成功。

注释

PalmLsnDupMessage 方法将复制文本消息，提取主题、内容和发送者字段，然后将这些值赋给 a_palm_msg 实例。

如果消息中没有出现发送者字段，则不抽取该字段。如果您使用 PalmLsnDupSender，则它会覆盖从 PalmLsnDupMessage 中抽取的发送者字段（如果有）。

另请参见

- “PalmLsnDupSender 方法”一节第 89 页
- “PalmLsnDupTime 方法”一节第 90 页
- “消息处理方法”一节第 82 页

示例

以下示例（用于 Treo 650 smartphone 实现）检索文本消息并调用 PalmLsnDupMessage 来初始化 a_palm_msg 实例中的相应字段：

```
// Retrieve the entire message body
ret = PhnLibGetText( libRef, id, &msgBodyH );
if (ret != errNone) {
    // handle error
    goto done;
}
msgBody = (Char *)MemHandleLock( msgBodyH );
ret = PalmLsnDupMessage( ulMsg, msgBody );

// msgBodyH must be disposed of by the caller
MemHandleUnlock(msgBodyH);
MemHandleFree(msgBodyH);
if (ret != errNone) {
    // handle error
    goto done;
}
```


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

PalmLsnDupSender 方法

初始化 a_palm_msg 实例的发送者字段。

语法

```
Err PalmLsnDupSender(
   struct a_palm_msg * const msg,
   Char const * sender
)
```

参数

- **msg** 指向 a_palm_msg 实例的指针。
- **sender** 包含源发送者字段的输入参数。

返回值

Palm OS 错误代码。errNone 表示操作成功。

注释

PalmLsnDupSender 方法复制发送者输入参数，并将值赋给 a_palm_msg 实例。

另请参见

- “PalmLsnDupMessage 方法”一节第 88 页
- “PalmLsnDupTime 方法”一节第 90 页
- “消息处理方法”一节第 82 页

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

PalmLsnDupTime 方法

初始化 a_palm_msg 实例的时间字段。

语法

```
Err PalmLsnDupTime(
   struct a_palm_msg * const msg,
   UInt32 const time
)
```

参数

- **msg** 指向 a_palm_msg 实例的指针。
- **time** 包含源时间字段的输入参数。

返回值

Palm OS 错误代码。errNone 表示操作成功。

注释

PalmLsnDupTime 方法复制时间输入参数，并将值赋给 a_palm_msg 实例。

另请参见

- “PalmLsnDupMessage 方法”一节第 88 页
- “PalmLsnDupSender 方法”一节第 89 页
- “消息处理方法”一节第 82 页

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

PalmLsnFree 方法

释放消息内存资源。

语法

void **PalmLsnFree(** struct a_palm_msg * const *msg* **)**

参数

- **msg** 要释放的 a_palm_msg 实例。

示例

以下示例显示了用于分配消息结构、处理消息以及使用 PalmLsnFree 释放资源的部分列表：

```
a_palm_msg * ulMsg;
...

// Allocate the message structure
ulMsg = PalmLsnAllocate();
...

// Fill the message fields
ret = PalmLsnDupMessage(ulMsg, msgBody);
...

// Process the message
ret = PalmLsnProcess(ulMsg, configDb, NULL, handled);
...

// Free the message
PalmLsnFree(ulMsg);
```

PalmLsnGetConfigFileName 方法

返回包含监听器配置数据库名称的字符串。

语法

void **PalmLsnGetConfigFileName(** Char * *configPDBName* **)**

参数

- **configPDBName** 包含监听器配置数据库名称的输出参数。

注释

可以使用此方法获取要传递给 PalmLsnProcess 的配置数据库文件名。

要使用缺省配置数据库文件名 *lsncfg*，请将 PalmLsnDefaultConfigDB（在 *PalmLsn.h* 中定义）复制到此输出参数中。

另请参见

- “PalmLsnProcess 方法”一节第 96 页

示例

以下示例（用于 Treo 650 smartphone 实现）返回输出参数中的缺省配置数据库名称：

```
void PalmLsnGetConfigFileName(Char * configPDBName) {
    StrCopy(configPDBName, PalmLsnDefaultConfigDB);
}
```

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

PalmLsnNormalHandleEvent 方法

处理应用程序事件。

语法

Boolean **PalmLsnNormalHandleEvent(** EventPtr *eventP* **)**

参数

- **eventP** 指向应用程序事件的指针。

返回值

如果事件已被处理，则返回 true。

注释

可以使用此方法处理应用程序事件。

PalmLsnNormalStart 方法

当监听器应用程序启动时提供自定义操作。

语法

Err **PalmLsnNormalStart( )**

返回值

Palm OS 错误代码。errNone 表示操作成功。

注释

可以使用 PalmLsnNormalStart 将设备注册到 MobiLink 服务器。

另请参见

- “PalmLsnNormalStop 方法”一节第 95 页
- “PalmLsnSpecialLaunch 方法”一节第 98 页

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

PalmLsnNormalStop 方法

当监听器应用程序退出事件循环时提供自定义操作。

语法

void **PalmLsnNormalStop( )**

注释

如果要继续监听，请不要使用 PalmLsnNormalStop 注销设备。

可以使用此方法获取并设置当前应用程序的首选项。

另请参见

- “PalmLsnNormalStart 方法”一节第 94 页

PalmLsnProcess 方法

根据配置数据库中的记录处理消息。

语法

```
palm_lsn_ret PalmLsnProcess(
   struct a_palm_msg * msg,
   Char const * configPDBName,
   UInt16 * const problematicRecNum,
   Boolean * handled
)
```

参数

- **msg** 指向 a_palm_msg 实例的指针。
- **configPDBName** 包含配置数据库名称的字符数组。使用 PalmLsnGetConfigFileName 方法获取配置数据库名称。请参见“PalmLsnGetConfigFileName 方法”一节第 92 页。
- **problematicRecNum** 标识配置数据库中有问题或格式有误的记录的索引的输出参数。
- **handled** 指示 PalmLsnProcess 是否成功处理消息的输出参数。

返回值

palm_lsn_ret 枚举中定义的返回代码。请参见“palm_lsn_ret 枚举”一节第 85 页。

注释

PalmLsnProcess 确定针对进来的消息采取相应的操作。它比较消息的字段与配置数据库中存储的过滤器。

有关创建监听器配置数据库的详细信息，请参见“用于 Palm 设备的监听器配置实用程序”一节第 76 页。

配置数据库中包含的记录存储有关消息过滤器以及针对接收的消息应采取哪些操作的信息。

配置记录具有以下格式：

```
[subject=<string>;] [content=<string>;]
[message|message_start=<string>;] [sender=<string>;]
action=run <app name> [arguments]
```

arguments 是可以包含操作变量的与应用程序相关的字符串。

另请参见

- “用于 Palm 设备的监听器配置实用程序”一节第 76 页
- “消息处理程序”一节第 15 页
- “用于 Windows 的监听器操作命令”一节第 70 页
- “PalmLsnCheckConfigDB 方法”一节第 87 页
- “消息处理方法”一节第 82 页

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

示例

以下是处理消息使用的部分列表。此示例分配消息结构、初始化字段并使用 PalmLsnProcess 处理消息：

```
a_palm_msg * ulMsg;
Boolean * handled
Char configDb[dmDBNameLength];
...

// Allocate the message structure
ulMsg = PalmLsnAllocate();
...

// Fill the message fields
ret = PalmLsnDupMessage(ulMsg, msgBody);
...

// Get the configuration database name
PalmLsnGetConfigFileName(configDb);

// Process the message
ret = PalmLsnProcess(ulMsg, configDb, NULL, handled);
...

// Free the message
PalmLsnFree(ulMsg);
```

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

PalmLsnSpecialLaunch 方法

响应可能与设备相关的启动代码。

语法

Err **PalmLsnSpecialLaunch(**
  UInt16 *cmd*,
  MemPtr *cmdPBP*,
  UInt16 *launchFlags*
)

参数

- **cmd** Palm OS 应用程序启动代码。
- **cmdPBP** 指向包含启动代码参数的结构的指针。如果应用程序不具有任何启动命令特有的参数，该值为 NULL。
- **launchFlags** 指示应用程序状态信息的标志。

返回值

Palm OS 错误代码。errNone 表示操作成功。

注释

此方法响应未定义为 sysAppLaunchCmdNormalLaunch 的设备相关的或标准的启动代码。

示例

以下示例（用于 Treo 650 smartphone 实现）使用 PalmLsnSpecialLaunch 处理监听器事件：

```
Err PalmLsnSpecialLaunch( UInt16 cmd, MemPtr cmdPBP, UInt16 /*launchFlags*/ )
{
    switch(cmd) {
        case sysAppLaunchCmdSystemReset:
            // Fall through
        case sysAppLaunchCmdSyncNotify:
            // Fall through
        case phnLibLaunchCmdRegister:
            return registerListener();
        case phnLibLaunchCmdEvent: {
            if (!IsFeatureOn(PalmLsnGetFeature(), Listening)) {
                return(errNone);
            }
            PhnEventPtr phoneEventP = (PhnEventPtr) cmdPBP;
            if (phoneEventP->eventType == phnEvtMessageInd) {
              return(handleMessage(phoneEventP->data.params.id, &phoneEventP-
>acknowledge));
            }
        }
        default:
            break;
    }
    return(errNone);
}
```

如果检测到消息，则使用 handleMessage 来将消息处理为相应的操作。

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

```
static Err handleMessage( PhnDatabaseID id, Boolean * handled )
/*************************************************************/
// This routine will construct a_palm_msg and then call
// PalmLsnProcess to process it.
{
    a_palm_msg *     ulMsg;
    Err              ret;
    Boolean          newlyLoaded;
    PhnAddressList   addrList;
    PhnAddressHandle addrH;
    MemHandle        msgBodyH;
    Char *           msgSender;
    Char *           msgBody;
    UInt32           msgTime;
    Char             configDb[ dmDBNameLength ];
    UInt16           libRef = 0;

    // CDMA workaround recommended by Handspring
    DmOpenRef        openRef = 0;

    *handled = false;

    // Allocate a message structure for passing over
    // to PalmLsnProcess later
    ulMsg = PalmLsnAllocate();
    if (ulMsg == NULL) {
        return(sysErrNoFreeRAM);
    }

    // Load the phone library
    ret = findOrLoadPhoneLibrary(&libRef, &newlyLoaded);
    if (ret != errNone) {
        goto done;
    }
    openRef = PhnLibGetDBRef(libRef);

    // Retrieve sender of the message
    ret = PhnLibGetAddresses(libRef, id, &addrList);
    if (ret != errNone) {
        goto done;
    }
    ret = PhnLibGetNth(libRef, addrList, 1, &addrH);
    if (ret != errNone) {
        PhnLibDisposeAddressList(libRef, addrList);
        goto done;
    }

    msgSender = PhnLibGetField(libRef, addrH, phnAddrFldPhone);
    if (msgSender != NULL) {
        ret = PalmLsnDupSender(ulMsg, msgSender);
        MemPtrFree(msgSender);
    }
    PhnLibDisposeAddressList( libRef, addrList );
    if (ret != errNone) {
        goto done;
    }

    // Retrieve message time
    ret = PhnLibGetDate(libRef, id, &msgTime);
    if (ret != errNone) {
        goto done;
    }
    ret = PalmLsnDupTime(ulMsg, msgTime);
    if (ret != errNone) {
```

```
            goto done;
        }

        // Retrieve the entire message body
        ret = PhnLibGetText(libRef, id, &msgBodyH);
        if (ret != errNone) {
            goto done;
        }
        msgBody = (Char *) MemHandleLock(msgBodyH);
        ret = PalmLsnDupMessage(ulMsg, msgBody);

        // msgBodyH must be disposed of by the caller
        MemHandleUnlock(msgBodyH);
        MemHandleFree(msgBodyH);
        if (ret != errNone) {
            goto done;
        }

        // Get the configuration database name
        PalmLsnGetConfigFileName(configDb);

        // Call PalmLsnProcess to process the message
        ret = PalmLsnProcess(ulMsg, configDb, NULL, handled);

    done:
        if (ulMsg != NULL) {
            PalmLsnFree(ulMsg);
        }
        PhnLibReleaseDBRef(libRef, openRef);

        // Unload the phone library before any possible application switch
        if (newlyLoaded) {
            unloadPhoneLibrary(libRef);
            newlyLoaded = false;
        }
        return(ret);
    }
```

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

PalmLsnTargetCompanyID 方法

返回设备的公司或制造商 ID。

语法

UInt32 **PalmLsnTargetCompanyID( )**

返回值

包含设备的公司或制造商 ID 的值。

注释

使用 PalmLsnTargetCompanyID 和 PalmLsnTargetDeviceID 检查设备是否兼容。

另请参见

- “PalmLsnTargetDeviceID 方法”一节第 102 页

示例

以下示例（用于 Treo 650 smartphone 实现）返回 Handspring 的公司 ID 'hspr'：

```
UInt32 PalmLsnTargetCompanyID(void) {
    return('hspr');
}
```

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

PalmLsnTargetDeviceID 方法

返回目标设备 ID。

语法

UInt32 **PalmLsnTargetDeviceID( )**

返回值

包含设备 ID 的正整数。

注释

可以使用 PalmLsnTargetCompanyID 和 PalmLsnTargetDeviceID 检查设备是否兼容。

另请参见

- “PalmLsnTargetCompanyID 方法”一节第 101 页

示例

以下示例返回 Treo 650 Simulator 的设备 ID：

```
UInt32 PalmLsnTargetDeviceID(void) {
    return(kPalmOneDeviceIDTreo650);
}
```

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

服务器启动的同步系统过程

目录

服务器启动的同步系统过程在 MobiLink 系统表中添加和删除行。

注意
这些系统过程用于设备跟踪。如果使用的远程设备支持设备自动跟踪，则无需使用这些系统过程。如果使用的远程设备不支持设备自动跟踪，则可以使用这些系统过程配置设备手工跟踪。

另请参见

- “设备跟踪网关”一节第 27 页
- “添加设备跟踪支持”一节第 28 页
- “MobiLink 服务器系统表” 《MobiLink - 服务器管理》
- “MobiLink 服务器系统过程” 《MobiLink - 服务器管理》

IBM DB2 主机服务器启动的同步系统过程名的转换

DB2 主机 8.1 版本支持向后兼容模式，其中列名和其它的标识符最多可以使用 18 个字符。要支持此环境，所有 DB2 主机中的 MobiLink 系统对象的名称都不能超过 18 个字符。下表列出了 DB2 主机统一数据库的过程名与所有其它统一数据库类型的系统过程名之间的映射关系。

如果系统过程名未显示在下表中，则不需要转换。

系统过程名	DB2 主机统一数据库的系统过程名
“ml_delete_device_address 系统过程”一节第 106 页	ml_del_dev_addr
“ml_delete_listening 系统过程”一节第 107 页	ml_del_listen
“ml_set_device_address 系统过程”一节第 109 页	ml_set_dev_addr
“ml_set_sis_sync_state 系统过程”一节第 112 页	ml_set_sis_state

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

ml_delete_device 系统过程

手工设置设备跟踪时，使用此系统过程删除有关远程设备的所有信息。

参数

项	参数	说明
1	device	VARCHAR(255)。设备名称。

注释

此函数仅在手工设置设备跟踪时才有用。请参见“添加设备跟踪支持”一节第 28 页。

示例

删除一条设备记录及引用此设备记录的所有相关记录：

```
CALL ml_delete_device('myOldDevice');
```

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

ml_delete_device_address 系统过程

手工设置设备跟踪时，使用此系统过程删除设备地址。

参数

项	参数	说明
1	device	VARCHAR(255)
2	medium	VARCHAR(255)

注释

此系统过程仅在手工设置设备跟踪时才有用。请参见“添加设备跟踪支持”一节第 28 页。

对于 DB2 主机统一数据库，此过程称为 ml_del_dev_addr。请参见“IBM DB2 主机服务器启动的同步系统过程名的转换”一节第 104 页。

示例

删除一条地址记录：

```
CALL ml_delete_device_address('myFirstTreo180', 'ROGERS AT&T');
```

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

ml_delete_listening 系统过程

手工设置设备跟踪时，使用此系统过程删除 MobiLink 用户与远程设备之间的映射。

参数

项	参数	说明
1	ml_user	VARCHAR(128)

注释

此系统过程仅在手工设置设备跟踪时才有用。请参见“添加设备跟踪支持”一节第 28 页。

对于 DB2 主机统一数据库，此过程称为 ml_del_listen。请参见“IBM DB2 主机服务器启动的同步系统过程名的转换”一节第 104 页。

示例

删除一条接收者记录：

```
CALL ml_delete_listening('myULDB');
```

ml_set_device 系统过程

手工设置设备跟踪时，使用此系统过程添加或更改远程设备的相关信息。它将在 ml_device 表中添加或更新行。

参数

项	参数	说明
1	device	VARCHAR(255)。用户定义的唯一的设备名称。
2	listener_version	VARCHAR(128)。有关监听器版本的可选注释。
3	listener_protocol	INTEGER。**0** 用于版本 9.0.0，**1** 用于 9.0.0 之后版本的 Palm 监听器，**2** 用于 9.0.0 之后版本的 Windows 监听器。
4	info	VARCHAR(255)。可选设备信息。
5	ignore_tracking	CHAR(1)。设置为 **y** 以忽略跟踪并防止它覆盖手工输入的数据。
6	source	VARCHAR(255)。有关此记录的来源的可选注释。

注释

系统过程 ml_set_device、ml_set_device_address 和 ml_set_listening 用于替换设备自动跟踪，方式是更改 MobiLink 系统表 ml_device、ml_device_address 和 ml_listening 中的信息。例如，如果某些远程设备是 Palm 设备，您可能要使用设备自动跟踪，但要为 Palm 设备手工插入数据。

此系统过程仅在手工设置设备跟踪时才有用。请参见“添加设备跟踪支持”一节第 28 页。

另请参见

- “ml_set_device_address 系统过程”一节第 109 页
- “ml_set_listening 系统过程”一节第 111 页
- “ml_device”一节 《MobiLink - 服务器管理》
- “ml_device_address”一节 《MobiLink - 服务器管理》
- “ml_listening”一节 《MobiLink - 服务器管理》

示例

为每个设备添加一条设备记录：

```
CALL ml_set_device(
    'myFirstTreo180',
    'MobiLink Listeners for Treo 180 - 9.0.1',
    '1',
    'not used',
    'y',
    'manually entered by administrator'
);
```

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

ml_set_device_address 系统过程

手工设置设备跟踪时，使用此系统过程添加或更改远程设备地址的相关信息。它将在 ml_device_address 表中添加或更新行。

参数

项	参数	说明
1	device	VARCHAR(255)。现有设备名。
2	medium	VARCHAR(255)。网络提供商 ID（必须与运营公司的 network_provider_id 属性匹配）。
3	address	VARCHAR(255)。具有 SMS 功能的设备的电话号码。
4	active	CHAR(1)。设置为 **y** 以激活要用于发送推式通知的此记录。
5	ignore_tracking	CHAR(1)。设置为 **y** 以忽略跟踪并防止它覆盖手工输入的数据。
6	source	VARCHAR(255)。有关此记录的来源的可选注释。

注释

系统过程 ml_set_device、ml_set_device_address 和 ml_set_listening 用于替换设备自动跟踪，方式是更改 MobiLink 系统表 ml_device、ml_device_address 和 ml_listening 中的信息。例如，如果某些远程设备是 Palm 设备，您可能要使用设备自动跟踪，但要为 Palm 设备手工插入数据。

此系统过程仅在手工设置设备跟踪时才有用。请参见“添加设备跟踪支持”一节第 28 页。

对于 DB2 主机统一数据库，此过程称为 ml_set_dev_addr。请参见“IBM DB2 主机服务器启动的同步系统过程名的转换”一节第 104 页。

另请参见

- “ml_set_device 系统过程”一节第 108 页
- “ml_set_listening 系统过程”一节第 111 页
- “ml_device”一节 《MobiLink - 服务器管理》
- “ml_device_address”一节 《MobiLink - 服务器管理》
- “ml_listening”一节 《MobiLink - 服务器管理》

示例

为每个设备添加一个设备地址记录：

```
CALL ml_set_device_address(
    'myFirstTreo180',
    'ROGERS AT&T',
    '3211234567',
    'y',
```

```
        'y',
        'manually entered by administrator'
    );
```

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

ml_set_listening 系统过程

手工设置设备跟踪时，使用此系统过程添加或更改 MobiLink 用户与远程设备之间的映射。它将在 ml_listening 表中添加或更新行。

参数

项	参数	说明
1	ml_user	VARCHAR(128)。MobiLink 用户名。
2	device	VARCHAR(255)。现有设备名。
3	listening	CHAR(1)。设置为 **y** 以激活要用于 DeviceTracker 寻址的此记录。
4	ignore_tracking	CHAR(1)。设置为 **y** 以忽略跟踪并防止它覆盖手工输入的数据。
5	source	VARCHAR(255)。有关此记录的来源的可选注释。

注释

系统过程 ml_set_device、ml_set_device_address 和 ml_set_listening 用于替换设备自动跟踪，方式是更改 MobiLink 系统表 ml_device、ml_device_address 和 ml_listening 中的信息。例如，如果某些远程设备是 Palm 设备，您可能要使用设备自动跟踪，但要为 Palm 设备手工插入数据。

此系统过程仅在手工设置设备跟踪时才有用。请参见“添加设备跟踪支持”一节第 28 页。

另请参见

- “ml_set_device 系统过程”一节第 108 页
- “ml_set_device_address 系统过程”一节第 109 页
- “ml_device”一节 《MobiLink - 服务器管理》
- “ml_device_address”一节 《MobiLink - 服务器管理》
- “ml_listening”一节 《MobiLink - 服务器管理》

示例

对于每个远程数据库，为某设备添加一个接收者记录。这会将设备映射到 MobiLink 用户名。

```
CALL ml_set_listening(
    'myULDB',
    'myFirstTreo180',
    'y',
    'y',
    'manually entered by administrator'
);
```

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

ml_set_sis_sync_state 系统过程

使用此系统过程将 MobiLink 同步状态记录到 ml_sis_sync_state 系统表中。

参数

项	参数	说明
1	remote_id	VARCHAR(128)。
2	subscription_id	VARCHAR(255)。
3	publication_name	VARCHAR(128)。
4	user_name	VARCHAR(128)。
5	last_upload	TIMESTAMP。
6	last_download	TIMESTAMP。

注释

在 publication_nonblocking_download_ack 事件中调用 ml_set_sis_sync_state 系统过程，以允许用户创建用于引用 ml_sis_sync_state 表的 request_cursor 事件。

对于 DB2 主机统一数据库，此过程称为 ml_set_sis_state。请参见“IBM DB2 主机服务器启动的同步系统过程名的转换”一节第 104 页。

另请参见

- “ml_sis_sync_state”一节 《MobiLink - 服务器管理》
- “publication_nonblocking_download_ack 连接事件”一节 《MobiLink - 服务器管理》

示例

使用以下脚本指定 publication_nonblocking_download_ack 事件脚本，以记录同步状态：

```
CALL ml_set_sis_sync_state(
    {ml_s.remote_id},
    NULL,
    {ml_s.publication_name},
    {ml_s.username},
    NULL,
    {ml_s.last_publication_download}
);
```

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

有关服务器启动的同步的高级主题

目录

消息语法

以下消息语法适用于轻量级轮询（缺省值）、UDP 网关和 SYNC 网关：

message = [subject]content

使用 SMTP 网关发送的消息具有以下语法结构之一：

- **message** = *sender*[*subject*]*content*
- **message** = *sender*(*subject*)*content*
- **message** = *sender*{*subject*}*content*
- **message** = *sender*<*subject*>*content*
- **message** = *sender*'*subject*'*content*
- **message** = *sender*"*subject*"*content*

消息语法和 *sender* 电子邮件地址语法是否正确取决于您的无线运营公司。要确定消息语法，请在消息记录已启用且已使用 dblsn -m 和 -v 选项将详细程度级别设置为 2 的情况下运行监听器。最初运行监听器时，消息日志包含正确的语法。

使用设备跟踪网关时，消息语法取决于用于发送消息的下级网关。如果正在使用 SMTP 下级网关，则语法取决于您的公共无线运营公司。

注释

大括号、尖括号、双引号、小括号、单引号和方括号是保留符号，以供内部使用，不应在 **subject** 中使用。有关消息限制和约束的详细信息，请参见“使用推式请求”一节第 9 页。

另请参见

- “用于 Windows 的监听器关键字”一节第 68 页
- “网关和运营公司”一节第 27 页
- “-m 选项”一节第 62 页
- “-v 选项”一节第 67 页

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

使用 SA_SEND_UDP 发送推式通知

如果使用 SQL Anywhere 统一数据库，可使用 SA_SEND_UDP 系统过程经由 UDP 网关将推式通知发送到设备。此方法是使用通告程序发送推式通知的替代方法。

通过将 **1** 追加到原始消息的结尾，然后在 SA_SEND_UDP 系统过程的 **msg** 参数中使用此消息，可将原始消息发送到监听器。

♦ 使用 SA_SEND_UDP 系统过程发送推式通知

此过程假定监听器已安装在设备上，且正在监听推式通知。在设备上使用以下命令：

```
dblsn -l "message=RunBrowser;action='START iexplore.exe http://
www.ianywhere.com';"
```

出于演示的目的，假定监听器每当接收 **RunBrowser** 消息时都会装载 Internet Explorer。此过程还假定 SQL Anywhere 统一数据库正在 MobiLink 服务器上运行。

1. 运行 Interactive SQL 并连接到统一数据库。
2. 执行以下命令：

   ```
   dbisql -c "dsn=consdb_source_name"
   ```

   将 *consdb_source_name* 替换为您统一数据库的 ODBC 名称。
3. 执行以下命令发送推式通知：

   ```
   CALL SA_SEND_UDP('device_ip_address', 5001, 'RunBrowser1')
   ```

   第一个参数确保将推式通知发送到正确的设备。将 *device_ip_address* 替换为该设备的 IP 地址。如果在 MobiLink 服务器所在的计算机上运行监听器，则使用 **localhost**。

   第二个参数是端口号。缺省情况下，监听器使用端口 5001 监听 UDP。

   第三个参数是要在末尾追加 **1** 进行发送的消息。通过追加 1，即变成了服务器启动的保留同步协议，使用 UDP 网关将 **RunBrowser** 消息发送到设备。

   执行此系统调用时，将 **RunBrowser** 消息发送到设备，促使设备运行 Internet Explorer 并装载 iAnywhere 主页。

有关 SA_SEND_UDP 系统过程的详细信息，请参见“sa_send_udp 系统过程”一节 《SQL Anywhere 服务器 - SQL 参考》。

服务器启动的同步教程

目录

教程：使用轻量级轮询进行的服务器启动的同步

服务器启动的同步教程简介

本教程演示如何配置 SQL Anywhere 统一数据库和远程数据库进行服务器启动的同步。教程基于 *samples-dir\MobiLink\SIS_CarDealer_LP_DBLSN* 中的示例代码。

关于服务器启动的同步的几个示例实现位于 *samples-dir\MobiLink* 中。所有服务器启动的同步的示例目录名均以 SIS_ 前缀开头。

必需的软件

- SQL Anywhere 11

能力和经验

- 了解 MobiLink 事件脚本的基本知识。

目标

- 设置 SQL Anywhere 统一数据库进行服务器启动的同步。
- 配置服务器端属性。
- 发出提示进行服务器启动的同步的推式请求。

建议阅读的背景知识

- “服务器启动的同步的作用是什么”一节第 2 页

第 1 课：建立统一数据库

在本课中，您将使用 dbinit 实用程序创建一个名为 **SIS_CarDealer_LP_DBLSN_CONDB** 的含有同步所需脚本的统一数据库。

♦ 创建并启动新的 SQL Anywhere 统一数据库

1. 创建 SQL Anywhere 统一数据库。

   在命令提示符处，使用 dbinit 实用程序，并指定数据库的文件名和路径。例如，可运行以下命令：

   ```
   dbinit c:\MLsis\SIS_CarDealer_LP_DBLSN_CONDB.db
   ```

2. 启动统一数据库。

   ```
   dbeng11 c:\MLsis\SIS_CarDealer_LP_DBLSN_CONDB.db
   ```

使用 SQL Anywhere 11 驱动程序为数据库定义 ODBC 数据源。

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

♦ **为统一数据库定义 ODBC 数据源**

1. 选择 [**开始**] » [**程序**] » [**SQL Anywhere 11**] » [**ODBC 管理器**]。
2. 单击 [**用户 DSN**] 选项卡，然后单击 [**添加**]。
3. 在 [**名称**] 列表中，单击 [**SQL Anywhere 11**]。单击 [**完成**]。
4. 在 [**SQL Anywhere 11 的 ODBC 配置**] 窗口中，进行以下操作：
   a. 单击 [**ODBC**] 选项卡。
   b. 在 [**数据源名**] 字段中，键入 **SIS_CarDealer_LP_DBLSN_CONDB**。
   c. 单击 [**登录**] 选项卡。
   d. 在 [**用户 ID**] 字段键入 **DBA**。
   e. 在 [**口令**] 字段中键入 **sql**。
   f. 单击 [**数据库**] 选项卡。
   g. 在 [**服务器名**] 字段中键入 **SIS_CarDealer_LP_DBLSN_CONDB**。
   h. 在 [**数据库文件**] 字段中，键入 *c:\MLsis\SIS_CarDealer_LP_DBLSN_CONDB.db*。
   i. 单击 [**确定**]。
5. 单击 [**确定**]。

另请参见

- “创建 ODBC 数据源”一节 《SQL Anywhere 服务器 - 数据库管理》

第 2 课：生成数据库模式

在本课中，您将生成数据库模式，其中包括 Dealer 表、non_sync_request 表和 download_cursor 同步脚本。此数据库模式满足生成推式请求的要求。

♦ **建立数据库模式**

1. 连接到统一数据库：
   a. 在 Sybase Central 中，右击 [**SQL Anywhere 11**] 并选择 [**连接**]。
   b. 单击 [**标识**] 选项卡。
   c. 单击 [**ODBC 数据源名称**]，然后键入 **SIS_CarDealer_LP_DBLSN_CONDB**。单击 [**确定**]。
2. 启动 Interactive SQL：

   在左窗格中，右击数据库并选择 [**打开 Interactive SQL**]。
3. 执行以下语句，创建和设置 Dealer 表和 non_sync_request 表：

```
CREATE TABLE Dealer (
    name           VARCHAR(10) NOT NULL PRIMARY KEY,
    rating         VARCHAR(5),
    last_modified  TIMESTAMP DEFAULT TIMESTAMP
```

```
)

CREATE TABLE non_sync_request(
    poll_key       varchar(128)
)
```

4. 使用以下语句，将数据插入 Dealer 表：

```
INSERT INTO Dealer(name, rating) VALUES ('Audi', 'a');
INSERT INTO Dealer(name, rating) VALUES ('Buick', 'b');
INSERT INTO Dealer(name, rating) VALUES ('Chrysler', 'c');
INSERT INTO Dealer(name, rating) VALUES ('Dodge', 'd');
INSERT INTO Dealer(name, rating) VALUES ('Eagle', 'e');
INSERT INTO Dealer(name, rating) VALUES ('Ford', 'f');
INSERT INTO Dealer(name, rating) VALUES ('Geo', 'g');
INSERT INTO Dealer(name, rating) VALUES ('Honda', 'h');
INSERT INTO Dealer(name, rating) VALUES ('Isuzu', 'I');
COMMIT;
```

5. 执行以下命令，创建 MobiLink 系统表和存储过程。用 SQL Anywhere 11 安装目录的位置来替换 *c:\Program Files\SQL Anywhere 11*\。

```
read "c:\Program Files\SQL Anywhere 11\MobiLink\setup\syncsa.sql"
```

6. 执行以下语句，指定 download_cursor 同步脚本并将同步状态记录到 ml_sis_sync_state 系统表：

```
CALL ml_add_table_script(
    'CarDealer',
    'Dealer',
    'download_cursor',
    'SELECT * FROM Dealer WHERE last_modified >= ?'
);
CALL ml_add_connection_script(
    'CarDealer',
    'publication_nonblocking_download_ack',
    'CALL ml_set_sis_sync_state(
                {ml s.remote_id},
        NULL,
                {ml s.publication_name},
                {ml s.username},
                NULL,
                {ml s.last_publication_download}
    )'
);
COMMIT;
```

此脚本设置 ml_sis_sync_state 以记录仅下载同步。记录同步状态允许您从 request_cursor 事件引用 ml_sis_sync_state request_cursor 事件。您将在下一课中指定 request_cursor 事件。

7. 关闭 Interactive SQL。

另请参见

- “SQL Anywhere 数据库服务器”一节 《SQL Anywhere 服务器 - 数据库管理》
- “CREATE TABLE 语句”一节 《SQL Anywhere 服务器 - SQL 参考》
- “编写同步脚本” 《MobiLink - 服务器管理》
- “download_cursor 表事件”一节 《MobiLink - 服务器管理》
- “ml_sis_sync_state”一节 《MobiLink - 服务器管理》
- “publication_nonblocking_download_ack 连接事件”一节 《MobiLink - 服务器管理》

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

第 3 课：配置通告程序

在本课中，您将配置通告程序事件，定义通告程序如何创建推式请求以及如何向设备发送推式通知。

♦ 创建新的通告程序

1. 使用 MobiLink 同步插件连接到统一数据库：
   a. 打开 Sybase Central。
   b. 在左窗格中，单击 [**MobiLink 11**]。
   c. 单击 [**模式**] » [**管理**]。
   d. 单击 [**文件**] » [**连接**]。
   e. 单击 [**标识**] 选项卡。
   f. 在 [**ODBC 数据源名称**] 字段中，键入 **SIS_CarDealer_LP_DBLSN_CONDB**。
   g. 单击 [**确定**]。
2. 在左窗格中，单击 [**通知**] 文件夹。
3. 单击 [**文件**] » [**新建**] » [**通告程序**]。
4. 在 [通告程序] 字段中指定 **CarDealerNotifier**，然后单击 [**完成**]。
5. 输入 request_cursor 事件脚本。

   request_cursor 事件脚本将检测推式请求。每个推式请求确定发送哪些信息以及哪个设备应接收这些信息。
   a. 在右窗格中，选择 **CarDealerNotifier**。从 [**文件**] 菜单中选择 [**属性**]。
   b. 单击 [**事件**] 选项卡，并为该事件选择 [**request_cursor**]。
   c. 为 request_cursor 脚本输入以下内容：

```
SELECT ml_sis_sync_state.remote_id + '.sync' FROM ml_sis_sync_state
    WHERE (EXISTS (SELECT 1 FROM Dealer
        WHERE last_modified >= ml_sis_sync_state.last_download)
    AND
    EXISTS(select poll_key FROM non_sync_request))
```

6. 单击 [**确定**] 保存通告程序事件。

另请参见

- “request_cursor 事件”一节第 40 页
- “ml_set_sis_sync_state 系统过程”一节第 112 页

第 4 课：启动 MobiLink 服务器

在本课中，您将启动带有通告程序的 MobiLink 服务器，以便可以向设备发送推式通知。

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

♦ **运行 MobiLink 服务器 (mlsrv11)**

● 在命令提示符处，浏览到统一数据库的目录。在一行中键入以下内容：

```
mlsrv11 -notifier -c "dsn=SIS_CarDealer_LP_DBLSN_CONDB" -o ml.log -fr -v+
-zu+ -x tcpip
```

下表介绍与 mlsrv11 实用程序一起使用的每个选项。选项 -o 和 -v 提供调试和故障排除信息。适宜在开发环境中使用这些记录选项。出于性能原因，-v 通常不在生产中使用。

选项	说明
-notifier	为服务器启动的同步启动所有已启用通告程序。 请参见“-notifier 选项”一节 《MobiLink - 服务器管理》。
-c	指定连接字符串。 请参见“-c 选项”一节 《MobiLink - 服务器管理》。
-o	指定消息日志文件 *ml.log*。 请参见“-o 选项”一节 《MobiLink - 服务器管理》。
-fr	在同步不包括至少一个上载数据的脚本和一个下载数据的脚本的情况下，防止 MobiLink 服务器中止。本教程使用仅下载同步，所以需要此选项。 请参见“-fr 选项”一节 《MobiLink - 服务器管理》。
-v+	-v 选项指定记录哪些信息。使用 -v+ 设置最大详细记录。 请参见“-v 选项”一节 《MobiLink - 服务器管理》。
-zu+	自动添加新用户。 请参见“-zu 选项”一节 《MobiLink - 服务器管理》。
-x	为 MobiLink 客户端设置通信协议和协议选项。 请参见“-x 选项”一节 《MobiLink - 服务器管理》。

将出现一个窗口，指示 MobiLink 服务器正在运行。通告程序将指示已准备好从设备接收推式通知请求。

另请参见

有关本课中各主题的详细信息，请参见：

● “MobiLink 服务器” 《MobiLink - 服务器管理》
● “MobiLink 服务器选项” 《MobiLink - 服务器管理》

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

第 5 课：建立远程数据库

在本课中，您将创建 SQL Anywhere 远程数据库，创建同步发布、同步用户和同步预订。您还将创建监听器的命令文件，然后启动监听器。

♦ 创建并设置新的 SQL Anywhere 远程数据库

1. 在命令提示符处，浏览到要创建数据库的目录。
2. 键入以下命令创建数据库：

   ```
   dbinit c:\MLsis\SIS_CarDealer_LP_DBLSN_REM.db
   ```

3. 键入以下命令启动数据库：

   ```
   dbeng11 c:\MLsis\SIS_CarDealer_LP_DBLSN_REM.db
   ```

4. 生成远程数据库模式。
   a. 单击 [**开始**] » [**程序**] » [**SQL Anywhere 11**] » [**Sybase Central**]。
   b. 在左窗格中，单击 [**SQL Anywhere 11**]。
   c. 单击 [**文件**] » [**连接**]。
   d. 单击 [**标识**] 选项卡。
   e. 在 [**用户 ID**] 字段键入 **DBA**。
   f. 在 [**口令**] 字段中键入 **sql**。
   g. 单击 [**数据库**] 选项卡。
   h. 在 [**服务器名**] 字段中键入 **SIS_CarDealer_LP_DBLSN_REM**。
   i. 单击 [**确定**]。
   j. 单击 [**文件**] » [**打开 Interactive SQL**]。
   k. 执行以下命令创建 Dealer 表：

      ```
      CREATE TABLE Dealer (
          name           VARCHAR(10) NOT NULL PRIMARY KEY,
          rating         VARCHAR(5),
          last_modified  TIMESTAMP DEFAULT TIMESTAMP
      )
      COMMIT;
      ```

5. 执行以下命令创建同步发布、同步用户和同步预订：

   ```
   CREATE PUBLICATION CarDealer(TABLE DEALER WHERE 0=1)
   CREATE SYNCHRONIZATION USER test_mluser OPTION ScriptVersion='CarDealer'
   CREATE SYNCHRONIZATION SUBSCRIPTION TO CarDealer FOR test_mluser
   SET OPTION public.ml_remote_id = remote_id;
   COMMIT;
   ```

另请参见

- “MobiLink 客户端” 《MobiLink - 客户端管理》
- “CREATE TABLE 语句”一节 《SQL Anywhere 服务器 - SQL 参考》
- “发布数据”一节 《MobiLink - 客户端管理》
- “CREATE PUBLICATION 语句 [MobiLink] [SQL Remote]”一节 《SQL Anywhere 服务器 - SQL 参考》
- “CREATE SYNCHRONIZATION USER 语句 [MobiLink]”一节 《SQL Anywhere 服务器 - SQL 参考》
- “CREATE SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]”一节 《SQL Anywhere 服务器 - SQL 参考》
- “脚本版本”一节 《MobiLink - 服务器管理》

第 6 课：配置监听器

在本课中，您将配置监听器，方法是将监听器选项存储在某个文本文件中，然后通过在命令行指定该文件名来运行 dblsn。

♦ 配置监听器

1. 请先在 ODBC 数据源管理器中创建一个名为 SIS_CarDealer_LP_DBLSN_REM 的文件 dsn。运行以下命令，与 MobiLink 服务器同步并创建 *SIS_CarDealer_LP_DBLSN_REM.rid* 文件：

   ```
   dbmlsync -c filedsn=SIS_CarDealer_LP_DBLSN_REM.dsn -ot dbmlsync.log -qc -e
   sa=on
   ```

   监听器可以使用 $remote_id 操作变量定义 MobiLink 服务器用于标识设备的轮询键。可从远程 ID 文件 *SIS_CarDealer_LP_DBLSN_REM.rid* 中检索到此变量，该文件是在首次与 MobiLink 服务器同步时创建的。要使用远程 ID 文件，必须与 MobiLink 服务器同步。

2. 创建包含以下内容的文本文件 *mydblsn.txt*。

   ```
   # Verbosity level
   -v2

   # Show notification messages in console and log
   -m

   # Truncate, then write output to dblsn.log
   -ot dblsn.log

   # Remote ID file (defining the scope of $remote_id)
   -r SIS_CarDealer_LP_DBLSN_REM.rid

   # Message handlers

   # Watch for a notification without action
   -l "poll_connect=tcpip(host=localhost);
       poll_notifier=CarDealerNotifier;
       poll_key=$remote_id.no_action;"

   # Signal dbmlsync to launch, sync and then shutdown
   -l "poll_connect=tcpip(host=localhost);
       poll_notifier=CarDealerNotifier;
       poll_key=$remote_id.sync;
   ```

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

```
    action='run dbmlsync.exe -c filedsn=SIS_CarDealer_LP_DBLSN_REM.dsn -ot
dbmlsync.log -qc -e sa=on';"

# Shutdown the listener
-l "poll_connect=tcpip(host=localhost);
    poll_notifier=CarDealerNotifier;
    poll_key=$remote_id.shutdown;
    action='DBLSN FULL SHUTDOWN';"
```

3. 保存 *mydblsn.txt* 文件。
4. 启动监听器。

   在命令提示符处，浏览到监听器命令文件的目录。

   通过输入以下命令启动监听器：

   ```
   dblsn @mydblsn.txt
   ```

   将出现一个窗口，指示监听器正在运行。

另请参见

- “监听器”一节第 15 页
- “用于 Windows 设备的监听器实用程序”一节第 56 页
- “@ 选项”一节第 59 页
- “操作变量”一节第 17 页

第 7 课：发出推式请求

在本课中，您将对统一数据库中的 Dealer 表进行更改，以便在监听器轮询到推式通知时能够将信息下载到远程数据库中。然后通过将轮询键值插入统一数据库来提示进行服务器启动的同步。通告程序运行 request_cursor 事件，在 non_sync_request 表中检测到轮询键，然后向监听器发送推式通知。监听器收到推式通知后，与 MobiLink 数据库同步并更新远程数据库。

♦ 在统一数据库中进行更改

1. 通过 Interactive SQL 连接到 **SIS_CarDealer_LP_DBLSN_CONDB** 数据库。
2. 键入以下脚本：

   ```
   UPDATE Dealer
       SET RATING = 'B' WHERE name = 'Geo';
   COMMIT;
   ```

通过直接填充 non_sync_request 表发出推式请求。轮询键列确定哪个设备应接收推式通知。

♦ 提示进行服务器启动的同步

1. 通过 Interactive SQL 连接到 **SIS_CarDealer_LP_DBLSN_CONDB** 数据库。
2. 键入以下脚本：

   ```
   INSERT INTO non_sync_request(poll_key) VALUES ('%remote_id%.no_action');
   COMMIT;
   ```

3. 稍等几秒钟，等候同步发生。

   监听器应轮询统一数据库，下载推式通知，然后更新远程数据库中的 Dealer 表。

要停止与设备的服务器启动的同步，请从 non_sync_request 表中删除轮询键值。

♦ 停止服务器启动的同步

1. 通过 Interactive SQL 连接到 **SIS_CarDealer_LP_DBLSN_CONDB** 数据库。
2. 键入以下脚本：

```
DELETE FROM non_sync_request WHERE poll_key = '%remote_id%.no_action';
COMMIT;
```

另请参见

- “生成推式请求”一节第 9 页
- “INSERT 语句”一节 《SQL Anywhere 服务器 - SQL 参考》
- “UPDATE 语句”一节 《SQL Anywhere 服务器 - SQL 参考》
- “DELETE 语句”一节 《SQL Anywhere 服务器 - SQL 参考》

删除教程资料

从计算机中删除教程资料。

♦ 从计算机中删除教程资料

1. 关闭 Interactive SQL。
2. 关闭 SQL Anywhere、MobiLink 和同步客户端窗口。
3. 删除所有与教程相关的 DSN。
   a. 启动 ODBC 管理器。

      在命令提示符处，键入以下命令：

      ```
      odbcad32
      ```

   b. 删除 **SIS_CarDealer_LP_DBLSN_CONDB** 数据源。
4. 浏览到包含统一数据库和远程数据库的目录 *c:\mlsis*，并删除所有文件。

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

教程：使用网关进行的服务器启动的同步

服务器启动的同步教程简介

本教程演示如何配置 SQL Anywhere 统一数据库和远程数据库进行服务器启动的同步。教程基于 *samples-dir\MobiLink\SIS_CarDealer* 中的示例代码。

关于服务器启动的同步的几个示例实现位于 *samples-dir\MobiLink* 中。所有服务器启动的同步的示例目录名均以 SIS_ 前缀开头。

必需的软件

- SQL Anywhere 11

能力和经验

- 了解 MobiLink 事件脚本的基本知识。

目标

- 设置 SQL Anywhere 统一数据库进行服务器启动的同步。
- 配置服务器端属性。
- 发出提示进行服务器启动的同步的推式请求。

建议阅读的背景知识

- “服务器启动的同步的作用是什么”一节第 2 页

第 1 课：建立统一数据库

在本课中，您将使用 dbinit 实用程序创建一个含有同步所需脚本的统一数据库。

♦ 创建并启动新的 SQL Anywhere 统一数据库

1. 创建 SQL Anywhere 统一数据库。

   在命令提示符处，使用 dbinit 实用程序，并指定数据库的文件名和路径。例如，可运行以下命令：

   ```
   dbinit c:\MLsis\MLconsolidated.db
   ```

2. 启动统一数据库。

   ```
   dbeng11 c:\MLsis\MLconsolidated.db
   ```

使用 SQL Anywhere 11 驱动程序为数据库定义 ODBC 数据源。

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

♦ 为统一数据库定义 ODBC 数据源

1. 选择 [**开始**] » [**程序**] » [**SQL Anywhere 11**] » [**ODBC 管理器**]。
2. 单击 [**用户 DSN**] 选项卡，然后单击 [**添加**]。
3. 在 [**名称**] 列表中，单击 [**SQL Anywhere 11**]。单击 [**完成**]。
4. 在 [**SQL Anywhere 11 的 ODBC 配置**] 窗口中，进行以下操作：
   a. 单击 [**ODBC**] 选项卡。
   b. 在 [**数据源名**] 字段中，键入 **sis_cons**。
   c. 单击 [**登录**] 选项卡。
   d. 在 [**用户 ID**] 字段键入 **DBA**。
   e. 在 [**口令**] 字段中键入 **sql**。
   f. 单击 [**数据库**] 选项卡。
   g. 在 [**服务器名**] 字段中键入 **MLconsolidated**。
   h. 在 [**数据库文件**] 字段中键入 *c:\MLsis\MLconsolidated.db*。
   i. 单击 [**确定**]。
5. 单击 [**确定**]。

另请参见

- “创建 ODBC 数据源”一节 《SQL Anywhere 服务器 - 数据库管理》

第 2 课：生成数据库模式

统一数据库模式包括 Dealer 表、download_cursor 同步脚本，以及用于生成服务器启动的同步推式请求的表和存储过程。

♦ 添加 Dealer 表和 download_cursor 同步脚本

1. 连接到统一数据库：
   a. 在 Sybase Central 中，右击 [**SQL Anywhere 11**] 并选择 [**连接**]。
   b. 单击 [**标识**] 选项卡。
   c. 单击 [**ODBC 数据源名称**]，然后键入 **sis_cons**。单击 [**确定**]。
2. 启动 Interactive SQL。

   在左窗格中，右击数据库并选择 [**打开 Interactive SQL**]。
3. 执行以下语句，安装 Dealer 表和 download_cursor 同步脚本。

```
CREATE TABLE Dealer (
    name varchar(10) NOT NULL PRIMARY KEY,
    rating VARCHAR(5),
    last_modified TIMESTAMP DEFAULT TIMESTAMP
```

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

```
)
INSERT INTO Dealer(name, rating) VALUES ('Audi', 'a');
INSERT INTO Dealer(name, rating) VALUES ('Buick', 'b');
INSERT INTO Dealer(name, rating) VALUES ('Chrysler', 'c');
INSERT INTO Dealer(name, rating) VALUES ('Dodge', 'd');
INSERT INTO Dealer(name, rating) VALUES ('Eagle', 'e');
INSERT INTO Dealer(name, rating) VALUES ('Ford', 'f');
INSERT INTO Dealer(name, rating) VALUES ('Geo', 'g');
INSERT INTO Dealer(name, rating) VALUES ('Honda', 'h');
INSERT INTO Dealer(name, rating) VALUES ('Isuzu', 'I');
COMMIT;
```

4. 执行以下命令，创建 MobiLink 系统表和存储过程。用 SQL Anywhere 11 安装目录的位置来替换 *c:\Program Files\SQL Anywhere 11*\。

```
read "c:\Program Files\SQL Anywhere 11\MobiLink\setup\syncsa.sql"
```

5. 执行以下命令：

```
CALL ml_add_table_script(
    'sis_ver1',
    'Dealer',
    'download_cursor',
    'SELECT * FROM Dealer WHERE last_modified >= ?'
)
```

另请参见

- “SQL Anywhere 数据库服务器”一节 《SQL Anywhere 服务器 - 数据库管理》
- “CREATE TABLE 语句”一节 《SQL Anywhere 服务器 - SQL 参考》
- “编写同步脚本” 《MobiLink - 服务器管理》
- “download_cursor 表事件”一节 《MobiLink - 服务器管理》

第 3 课：构造推式请求

当通告程序检测到推式请求时，会向设备发送一条消息。

♦ 创建用于存储推式请求的简单表

1. 在 Interactive SQL 中执行以下命令：

```
CREATE TABLE PushRequest (
    req_id INTEGER DEFAULT AUTOINCREMENT PRIMARY KEY,
    mluser VARCHAR(128),
    subject VARCHAR(128),
    content VARCHAR(128),
    resend_interval VARCHAR(30) DEFAULT '20s',
    time_to_live VARCHAR(30) DEFAULT '1m',
    status VARCHAR(128) DEFAULT 'created'
)
COMMIT;
```

2. 关闭 Interactive SQL。

另请参见

- “推式请求”一节第 8 页
- “服务器启动的同步简介”第 1 页
- “服务器启动的同步组件”一节第 4 页

第 4 课：配置通告程序

在本课中，您将配置三个通告程序事件，分别用于影响通告程序创建推式请求、向远程监听器传输请求以及删除过期请求的方式。

♦ 配置通告程序属性和事件

1. 使用 MobiLink 同步插件连接到统一数据库：
   a. 打开 Sybase Central。
   b. 在左窗格中，单击 [**MobiLink 11**]。
   c. 单击 [**模式**] » [**管理**]。
   d. 单击 [**文件**] » [**连接**]。
   e. 单击 [**标识**] 选项卡。
   f. 在 [**ODBC 数据源名称**] 字段中，键入 **sis_cons**。
   g. 单击 [**确定**]。
2. 在左窗格中，单击 [**通知**] 文件夹。
3. 单击 [**文件**] » [**新建**] » [**通告程序**]。
4. 在 [通告程序] 字段中指定 **CarDealerNotifier**，然后单击 [**完成**]。
5. 输入 begin_poll 事件脚本。

   通告程序检测到统一数据库中的更改，并使用 begin_poll 事件创建推式请求。在这种情况下，如果 Dealer 表中发生更改，并且远程数据库不是最新的，则 begin_poll 脚本会填充 PushRequest 表。
   a. 在右窗格中，选择 CarDealerNotifier。从 [**文件**] 菜单中选择 [**属性**]。
   b. 单击 [**事件**] 选项卡，并为该事件选择 begin_poll。
   c. 为 **begin_poll** 脚本输入以下内容：

```
--
-- Insert the last consolidated database
-- modification date into @last_modified
--
DECLARE @last_modified timestamp;
SELECT MAX(last_modified) INTO @last_modified FROM Dealer;

--
-- Delete processed requests if the mluser is up-to-date
--
DELETE FROM PushRequest
    FROM PushRequest AS p, ml_user AS u, ml_subscription AS s
```

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

```
        WHERE p.status = 'processed'
            AND u.name = p.mluser
            AND u.user_id = s.user_id
            AND @last_modified <= GREATER(s.last_upload_time,
    s.last_download_time);

    --
    -- Insert new requests when a device is not up-to-date
    --
    INSERT INTO PushRequest(mluser, subject, content)
    SELECT u.name, 'sync', 'ignored'
        FROM ml_user as u, ml_subscription as s
        WHERE u.name IN (SELECT name FROM ml_listening WHERE listening =
    'y')
            AND u.user_id = s.user_id
            AND @last_modified > greater(s.last_upload_time,
    s.last_download_time)
            AND u.name NOT LIKE '%-dblsn'
            AND NOT EXISTS(SELECT * FROM PushRequest
                WHERE PushRequest.mluser = u.name
                    AND PushRequest.subject = 'sync')
```

在 begin_poll 脚本的第一个主数据段中，如果设备是最新的，将消除 PushRequest 表中已处理的请求：

```
@last_modified <= GREATER(s.last_upload_time, s.last_download_time)
```

@last_modified 是统一数据库 Dealer 表中的最大修改日期。表达式 greater(s.last_upload_time, s.last_download_time) 表示远程数据库的上次同步时间。

也可以使用 request_delete 事件直接删除推式请求。但是，在这种情况下，begin_poll 事件可确保在远程数据库同步之前不消除已过期或隐式删除的请求。

下一个代码段检查 Dealer 表的 last_modified 列中是否有变化，并为所有非最新的活动监听器（在 ml_listening 表中列出）发出推式请求：

```
@last_modified > GREATER(s.last_upload_time, s.last_download_time)
```

填充 PushRequest 表时，begin_poll 脚本将主题设置为 'sync'。

6. 输入 request_cursor 脚本。

   request_cursor 脚本将读取推式请求。每个推式请求确定在消息中发送哪些信息以及哪些远程数据库接收这些信息。

   a. 为事件选择 [**request_cursor**]。

   b. 在 request_cursor 脚本的空白处输入以下代码：

```
SELECT
    p.req_id,
    'Default-DeviceTracker',
    p.subject,
    p.content,
    p.mluser,
    p.resend_interval,
    p.time_to_live
    FROM PushRequest AS p
```

   PushRequest 表向 request_cursor 脚本提供行。

request_cursor 结果集中的顺序和值非常重要。例如，第二个参数用于定义缺省网关 Default-DeviceTracker。设备跟踪网关将跟踪如何到达用户，并自动选择 UDP 或 SMTP 连接到远程设备。

7. 输入 request_delete 脚本。

   request_delete 通告程序事件指定清理操作。使用此脚本，通告程序可自动删除已隐式删除和已过期的请求。

   a. 为事件选择 [**request_delete**]。

   b. 在 request_delete 脚本的空白处输入以下内容：

   ```
   UPDATE PushRequest SET status='processed' WHERE req_id = ?
   ```

   此 request_delete 脚本不是删除 PushRequest 表中的行，而是将行状态更新为 'processed'。

8. 单击 [**确定**] 保存通告程序属性。

另请参见

- “request_delete 事件”一节第 41 页
- “begin_poll 事件”一节第 37 页
- “ml_listening”一节 《MobiLink - 服务器管理》
- “设备跟踪网关”一节第 27 页
- “request_cursor 事件”一节第 40 页
- “request_delete 事件”一节第 41 页

第 5 课：配置网关和运营公司

网关是发送消息的机制。您可以定义 UDP 网关和 SMTP 网关。也可以使用设备跟踪网关。使用设备跟踪时，MobiLink 将跟踪如何到达用户，并自动决定是使用 UDP 网关还是使用 SMTP 网关。

在本教程中，您将使用缺省设备跟踪网关，所以不需要进行任何配置。

另请参见

- “设备跟踪网关属性”一节第 50 页
- “网关和运营公司”一节第 27 页

第 6 课：启动 MobiLink 服务器

在本课中，您将启动带有通告程序的 MobiLink 服务器，以便可以向设备发送推式通知。

♦ 运行 MobiLink 服务器 (mlsrv11)

- 在命令提示符处，浏览到统一数据库目录，并键入以下内容：

  ```
  mlsrv11 -notifier -c "dsn=sis_cons" -o ml.log -fr -v+ -zu+ -x tcpip
  ```

  下表介绍与 mlsrv11 实用程序一起使用的每个选项。选项 -o 和 -v 提供调试和故障排除信息。适宜在开发环境中使用这些记录选项。出于性能原因，-v 通常不在生产中使用。

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

选项	说明
-notifier	为服务器启动的同步启动所有已启用通告程序。 请参见“-notifier 选项”一节 《MobiLink - 服务器管理》。
-c	指定连接字符串。 请参见“-c 选项”一节 《MobiLink - 服务器管理》。
-o	指定消息日志文件 *ml.log*。 请参见“-o 选项”一节 《MobiLink - 服务器管理》。
-fr	在同步不包括至少一个上载数据的脚本和一个下载数据的脚本的情况下，防止 MobiLink 服务器中止。本教程使用仅下载同步，所以需要此选项。 请参见“-fr 选项”一节 《MobiLink - 服务器管理》。
-v+	-v 选项指定记录哪些信息。使用 -v+ 设置最大详细记录。 请参见“-v 选项”一节 《MobiLink - 服务器管理》。
-zu+	自动添加新用户。 请参见“-zu 选项”一节 《MobiLink - 服务器管理》。
-x	为 MobiLink 客户端设置通信协议和协议选项。 请参见“-x 选项”一节 《MobiLink - 服务器管理》。

将出现一个窗口，指示 MobiLink 服务器已做好了处理请求的准备。还将出现通告程序。

另请参见

有关本课中各主题的详细信息，请参见：

- “MobiLink 服务器” 《MobiLink - 服务器管理》
- “MobiLink 服务器选项” 《MobiLink - 服务器管理》

第 7 课：建立远程数据库

在本课中，您将创建 SQL Anywhere 远程数据库，创建同步发布、同步用户和同步预订。您还将创建监听器的命令文件，然后启动监听器。

♦ 创建并启动新的 SQL Anywhere 远程数据库

1. 在命令提示符处，浏览到要创建数据库的目录。
2. 键入以下命令创建数据库：

```
dbinit c:\MLsis\rem1.db
```

3. 键入以下命令启动数据库：

```
dbeng11 c:\MLsis\rem1.db
```

4. 生成远程数据库模式。
   a. 单击 [**开始**] » [**程序**] » [**SQL Anywhere 11**] » [**Sybase Central**]。
   b. 在左窗格中，单击 [**SQL Anywhere 11**]。
   c. 单击 [**文件**] » [**连接**]。
   d. 单击 [**标识**] 选项卡。
   e. 在 [**用户 ID**] 字段键入 **DBA**。
   f. 在 [**口令**] 字段中键入 **sql**。
   g. 单击 [**数据库**] 选项卡。
   h. 在 [**服务器名**] 字段中键入 **rem1**。
   i. 单击 [**确定**]。
   j. 单击 [**文件**] » [**打开 Interactive SQL**]。
   k. 执行以下命令创建 Dealer 表：

```
CREATE TABLE Dealer (
    name VARCHAR(10) NOT NULL PRIMARY KEY,
    rating VARCHAR(5),
    last_modified TIMESTAMP DEFAULT TIMESTAMP
)
COMMIT;
```

5. 执行以下命令创建同步发布、同步用户和同步预订：

```
CREATE PUBLICATION car_dealer_pub (table Dealer);
CREATE SYNCHRONIZATION USER sis_user1;
CREATE SYNCHRONIZATION SUBSCRIPTION
    TO car_dealer_pub
    FOR sis_user1
    OPTION scriptversion='sis_ver1';
```

另请参见

- “MobiLink 客户端” 《MobiLink - 客户端管理》
- “CREATE TABLE 语句”一节 《SQL Anywhere 服务器 - SQL 参考》
- “发布数据”一节 《MobiLink - 客户端管理》
- “CREATE PUBLICATION 语句 [MobiLink] [SQL Remote]”一节 《SQL Anywhere 服务器 - SQL 参考》
- “CREATE SYNCHRONIZATION USER 语句 [MobiLink]”一节 《SQL Anywhere 服务器 - SQL 参考》
- “CREATE SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]”一节 《SQL Anywhere 服务器 - SQL 参考》
- “脚本版本”一节 《MobiLink - 服务器管理》

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

第 8 课：配置监听器

监听器在远程设备上运行。它从通告程序接收消息，并将消息处理为操作。例如，指定以下 dblsn 选项时，如果监听器收到消息 **sync**，则会启动 dbmlsync：

```
-l "subject=sync;action='run dbmlsync.exe...'
```

配置监听器的一种便捷方法是将命令行选项存储在文本文件中。例如，如果将设置存储在 *mydblsn.txt* 中，则可以通过输入以下内容启动监听器：

```
dblsn @mydblsn.txt
```

另外，如果键入 dblsn 且不带任何参数，则 dblsn 将使用 *dblsn.txt* 作为缺省参数文件。

♦ 创建并启动 MobiLink 监听器

1. 创建包含以下内容的文本文件 *mydblsn.txt*。

```
#-----------------------------------
# Verbosity level
-v2

# Show notification messages in console and log
-m

# Polling interval, in seconds
-i 3

# Truncate, then write output to dblsn.log
-ot dblsn.log

# MobiLink address and connect parameter for dblsn
-x "host=localhost"

# Enable device tracking and specify the MobiLink user name.
-t+ sis_user1

# Message handlers
# Synchronize using dbmlsync
-l "subject=sync;
action='start dbmlsync.exe
 -c eng=rem1;uid=DBA;pwd=sql
 -ot dbmlsyncOut.txt -qc';"
```

2. 保存 *mydblsn.txt* 文件。
3. 启动监听器。
   a. 在命令提示符处，浏览到监听器命令文件的目录。
   b. 通过输入以下内容启动监听器：

```
dblsn @mydblsn.txt
```

   将出现一个窗口，指示监听器正在运行，并已将设备跟踪信息上载到 MobiLink 服务器。

将跟踪信息上载到统一数据库时，将在 MobiLink 服务器窗口中出现一个新条目。此信息将转发监听器与 MobiLink 服务器之间成功进行的初始通信。

另请参见

- “监听器”一节第 15 页
- “用于 Windows 设备的监听器实用程序”一节第 56 页
- “@ 选项”一节第 59 页

第 9 课：发出推式请求

对于服务器启动的同步，可以通过直接填充 PushRequest 表发出推式请求，也可以通过在 Dealer 表中进行更改发出推式请求。在后一种情况下，通告程序 begin_poll 脚本将检测 Dealer 表中的更改，并填充 PushRequest 表。

在这两种情况下，PushRequest 表都会向通告程序 request_cursor 脚本提供行，该脚本用于确定远程设备接收消息的方式。

♦ 将推式请求直接插入 PushRequest 表，提示进行服务器启动的同步

1. 从 Interactive SQL，连接到 *MLconsolidated.db* 数据库并键入以下内容：

```
INSERT INTO PushRequest(mluser, subject, content)
    VALUES ('sis_user1', 'sync', 'not used');
COMMIT;
```

2. 稍等几秒钟，等候同步发生。

填充后，PushRequest 表会向通告程序的 request_cursor 脚本提供行。request_cursor 脚本确定在消息中发送哪些信息以及哪些远程设备接收这些信息。

♦ 在统一数据库 Dealer 表中进行更改，提示进行服务器启动的同步

1. 在 Interactive SQL 中，键入以下内容：

```
UPDATE Dealer
    SET RATING = 'B' WHERE name = 'Geo';
COMMIT;
```

2. 稍等几秒钟，等候同步发生。

在这种情况下，通告程序 begin_poll 脚本检测 Dealer 表中的更改，并相应填充 PushRequest 表。如前所述，填充 PushRequest 表后，通告程序 request_cursor 脚本确定在消息中发送哪些信息以及哪些远程设备接收这些信息。

另请参见

- “生成推式请求”一节第 9 页
- “INSERT 语句”一节 《SQL Anywhere 服务器 - SQL 参考》
- “UPDATE 语句”一节 《SQL Anywhere 服务器 - SQL 参考》

删除教程资料

从计算机中删除教程资料。

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

♦ 从计算机中删除教程资料

1. 关闭 Interactive SQL。
2. 关闭 SQL Anywhere、MobiLink 和同步客户端窗口。
3. 删除所有与教程相关的 DSN。
   a. 启动 ODBC 管理器。
      在命令提示符处，键入以下命令：

      ```
      odbcad32
      ```

   b. 删除 **ml_sis** 数据源。
4. 浏览到包含统一数据库和远程数据库的目录 *c:\mlsis*，并删除所有文件。

术语表

术语表

Adaptive Server Anywhere (ASA)

SQL Anywhere Studio 的关系数据库服务器组件，专供在移动和嵌入式环境中使用，或作为中小型企业的服务器使用。在版本 10.0.0 中，Adaptive Server Anywhere 更名为 SQL Anywhere 服务器，SQL Anywhere Studio 更名为 SQL Anywhere。

另请参见： “SQL Anywhere”一节第 158 页

包

Java 中相关类的集合。

被引用对象

一种对象（如表），该对象在另一个对象（如视图）的定义中被直接引用。

另请参见： “主键”一节第 168 页

编码

也称作字符编码，编码是一种方法，通过该方法可以将字符集中的每个字符映射到一个或多个字节的信息，这些信息通常以十六进制数字表示。编码的一个例子是 UTF-8。

另请参见：

- “字符集”一节第 168 页
- “代码页”一节第 143 页
- “归类”一节第 147 页

标识符

用于引用数据库对象（如表或列）的字符串。标识符可以包含 A 到 Z、a 到 z、0 到 9、下划线 (_)、at 符号 (@)、数字符号 (#) 或美元符号 ($) 中的任何字符。

并发

同时执行两个或更多个独立并且可能存在竞争关系的进程。SQL Anywhere 会自动使用锁定来隔离事务，并确保每个并发应用程序看到的数据集均一致。

另请参见：

- “事务”一节第 155 页
- “隔离级别”一节第 146 页

参考数据库

MobiLink 中一种用于 UltraLite 客户端开发的 SQL Anywhere 数据库。在开发过程中，可以将一个 SQL Anywhere 数据库同时作为参考数据库和统一数据库使用。通过其它产品建立的数据库无法用作参考数据库。

参照完整性

遵守数据一致性控制规则（具体而言，不同表中主键值与外键值之间的关系）。若要实现参照完整性，每个外键中的值必须与被引用表中行的主键值相符。

另请参见：

- “主键”一节第 168 页
- “外键”一节第 160 页

策略

QAnywhere 中指定应在何时进行消息传输的方式。

插件模块

Sybase Central 中一种用于访问和管理产品的方法。当您安装相应的产品时，插件通常会自动安装并注册 Sybase Central。通常，插件在 Sybase Central 主窗口中作为顶级容器出现，并且使用产品本身的名称，如 SQL Anywhere。

另请参见： “Sybase Central”一节第 159 页

查询

一条或一组 SQL 语句，用于访问和/或操作数据库中的数据。

另请参见： “SQL”一节第 158 页

冲突解决

在 MobiLink 中，冲突解决是指一种逻辑，它指定当两个用户修改不同远程数据库上同一行时的处理方法。

重定向器

一种 Web 服务器插件，用于为客户端与 MobiLink 服务器之间的请求和响应选择发送路径。此插件还实现了负荷平衡和故障转移机制。

抽取

SQL Remote 复制中从统一数据库卸载相应结构和数据的行为。此信息用于初始化远程数据库。

另请参见： “复制”一节第 145 页

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

触发器

一种特殊形式的存储过程，用户运行修改数据的查询时会自动执行该存储过程。

另请参见：

- “行级触发器”一节第 147 页
- “语句级触发器”一节第 165 页
- “完整性”一节第 161 页

传输规则

QAnywhere 中用于确定何时进行消息传输、传输哪些消息以及应在何时删除消息的逻辑。

窗口

作为分析功能执行对象的行组。一个窗口可以包含一行、多行或所有行的数据，这些数据已根据窗口定义中提供的分组规格进行了分区。窗口会进行移动，以包括为输入中的当前行执行计算所需的行数或行范围。窗口结构的主要优点是，不需要执行附加查询就可以有机会对结果进行分组和分析。

创建者 ID

UltraLite Palm OS 应用程序中一种在创建应用程序时指派的 ID。

存储过程

存储过程是数据库中存储的一组 SQL 指令，用于在数据库服务器上执行一组操作或查询。

代理表

一种本地表，它所包含的元数据可以像访问本地表一样访问远程数据库服务器上的表。

另请参见： “元数据”一节第 166 页

代理 ID

另请参见： “客户端消息存储库 ID”一节第 151 页

代码页

代码页是一种将字符集的字符映射到数字表示的编码，数字表示通常是 0 到 255 之间的一个整数。例如，Windows 代码页 1252 就是一个代码页。就本文档而言，代码页和编码这两个术语可以互换。

另请参见：

- “字符集”一节第 168 页
- “编码”一节第 141 页
- “归类”一节第 147 页

DBA 权限

使用户能够在数据库中执行管理活动的权限级别。DBA 用户在缺省情况下具有 DBA 权限。

另请参见： “数据库管理员 (DBA)”一节第 157 页

dbspace

用于创建更多数据存储空间的附加数据库文件。一个数据库可以包含在最多 13 个独立的文件（一个初始文件和 12 个 dbspace）中。每个表及其索引必须包含在单个数据库文件中。SQL 命令 CREATE DBSPACE 可将新文件添加到数据库中。

另请参见： “数据库文件”一节第 158 页

动态 SQL

执行前由程序以编程方式生成的 SQL。UltraLite 动态 SQL 是一种专用于小型设备的 SQL 变体。

对象树

Sybase Central 中数据库对象的层次。对象树的顶层显示您的 Sybase Central 版本所支持的全部产品。每种产品展开后会显示其自己的对象子树。

另请参见： “Sybase Central”一节第 159 页

EBF

快速错误修正软件。快速错误修正软件是含有一个或多个错误修正软件的软件子集。错误修正软件列在更新程序的发行说明中。错误修正软件更新可能只适用于具有相同版本号的已安装软件。已对该软件执行了一些测试，但该软件尚未进行完全测试。除非您自己已验证了软件的适用性，否则不要随应用程序分发这些文件。

发布

MobiLink 或 SQL Remote 中一种用于标识将要同步的数据的数据库对象。在 MobiLink 中，发布仅存在于客户端。一个发布包括多个项目。SQL Remote 用户可以通过预订发布来接收发布。MobiLink 用户可以通过创建发布的同步预订来同步发布。

另请参见：

- “复制”一节第 145 页
- “项目”一节第 163 页
- “发布更新”一节第 144 页

发布更新

SQL Remote 复制中对一个数据库中的一个或多个发布所做更改的列表。发布更新将作为复制消息的一部分定期发送到远程数据库。

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

另请参见：

- “复制”一节第 145 页
- “发布”一节第 144 页

发布者

SQL Remote 复制中数据库内可以与其它复制数据库交换复制消息的单个用户。

另请参见： “复制”一节第 145 页。

FILE

SQL Remote 复制中一种使用共享文件来交换复制消息的消息系统。它对测试以及在没有显式消息传送系统的情况下进行的安装很有用。

另请参见“复制”一节第 145 页。

分析树

查询的代数表示。

服务

在 Windows 操作系统上，服务是在运行应用程序的用户 ID 未登录时的应用程序运行方式。

服务器管理请求

一种 QAnywhere 消息，其格式设置为 XML 并发送到 QAnywhere 系统队列，作为一种管理服务器消息存储库或监控 QAnywhere 应用程序的方法。

服务器启动的同步

一种从 MobiLink 服务器启动 MobiLink 同步的方式。

服务器消息存储库

QAnywhere 中在消息传输到客户端消息存储库或 JMS 系统之前服务器上用于临时存储消息的关系数据库。消息通过服务器消息存储库在各客户端之间进行交换。

复制

在物理上不相同的数据库之间共享数据。Sybase 有三种复制技术：MobiLink、SQL Remote 和复制服务器。

复制代理

请参见： “LTM”一节第 152 页

复制服务器

Sybase 的一种基于连接的复制技术，用于与 SQL Anywhere 和 Adaptive Server Enterprise 一起使用。它专用于在一些数据库之间进行接近实时的复制。

另请参见： “LTM”一节第 152 页

复制频率

SQL Remote 复制中一项针对每个远程用户的设置，它决定发布者的消息代理向该远程用户发送复制消息的频率应为多少。

另请参见： “复制”一节第 145 页。

复制消息

SQL Remote 或复制服务器中一种在发布数据库与预订数据库之间发送的通信。消息包含复制系统所需的数据、直通语句及信息。

另请参见：

- “复制”一节第 145 页
- “发布更新”一节第 144 页

隔离级别

一个事务中的操作对其它并发事务中的操作的可见程度。隔离级别有四级，编号依次为 0 至 3。第 3 级提供最高级别的隔离。级别 0 为缺省设置。SQL Anywhere 还支持以下三个快照隔离级别：快照、语句快照和只读语句快照。

另请参见： “快照隔离”一节第 151 页

个人服务器

与客户端应用程序在同一台计算机上运行的数据库服务器。个人数据库服务器通常由单个用户在一台计算机上使用，但它可以支持来自该用户的几个并发连接。

工作表

一种内部存储区域，用于在查询优化过程中存储中间结果。

故障切换

在活动服务器、系统或网络出现故障或意外终止时切换到冗余或备用的服务器、系统或网络。故障转移会自动进行。

关系数据库管理系统 (RDBMS)

一种以相关表的形式存储数据的数据库管理系统。

另请参见： “数据库管理系统 (DBMS)”一节第 157 页

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

规范化

对数据库模式的改进，目的在于按照基于关系数据库理论的规则消除冗余并改善组织。

归类

定义数据库中文本属性的字符集与排序顺序的组合。对于 SQL Anywhere 数据库，缺省归类取决于运行服务器时所使用的操作系统和语言；例如，英语 Windows 系统上的缺省归类为 1252LATIN1。归类（也称作归类序列）用于对字符串进行比较和排序。

另请参见：

- “字符集”一节第 168 页
- “代码页”一节第 143 页
- “编码”一节第 141 页

行级触发器

每更改一行即执行一次的触发器。

另请参见：

- “触发器”一节第 143 页
- “语句级触发器”一节第 165 页

回退日志

对在每个未提交的事务执行过程中所做更改的记录。当收到 ROLLBACK 请求或者系统出现故障时，未提交的事务会从数据库中回退，将数据库返回其原先的状态。每个事务都有一个单独的回退日志，事务完成时日志会被删除。

另请参见： “事务”一节第 155 页

iAnywhere JDBC 驱动程序

iAnywhere JDBC 驱动程序提供了一个 JDBC 驱动程序，与纯 Java jConnect JDBC 驱动程序相比，该驱动程序拥有一些性能优势和功能优点，但它不是纯 Java 解决方案。建议在大多数情况下使用 iAnywhere JDBC 驱动程序。

另请参见：

- “JDBC”一节第 148 页
- “jConnect”一节第 148 页

InfoMaker

一种报告和数据维护工具，它用于创建复杂的表格、报告、图形、交叉表和表，并创建将这些报告用作构件块的应用程序。

Interactive SQL

一种 SQL Anywhere 应用程序，用于查询和更改数据库中的数据以及修改数据库的结构。Interactive SQL 不但提供了一个用于输入 SQL 语句的窗格，还提供了一些用于返回有关查询处理过程的信息和结果集的窗格。

JAR 文件

Java 档案文件。一种压缩的文件格式，由一个或多个用于 Java 应用程序的包的集合组成。它将安装和运行 Java 程序所需的全部资源都放在一个压缩文件中。

Java 类

Java 中的主要代码结构单元。它是组合在一起的过程和变量的集合，将过程和变量组合在一起的原因是它们都与某个特定的可识别类别有关。

jConnect

JavaSoft JDBC 标准的 Java 实现。它为 Java 开发人员提供多层和异类环境中的本地数据库访问。但在大多数情况下，iAnywhere JDBC 驱动程序是首选的 JDBC 驱动程序。

另请参见：

- “JDBC”一节第 148 页
- “iAnywhere JDBC 驱动程序”一节第 147 页

JDBC

Java 数据库连接。一种 SQL 语言编程接口，它允许 Java 应用程序访问关系数据。首选的 JDBC 驱动程序是 iAnywhere JDBC 驱动程序。

另请参见：

- “jConnect”一节第 148 页
- “iAnywhere JDBC 驱动程序”一节第 147 页

基表

永久性的数据表。有时为区别于临时表和视图，会将这种表称作**基表**。

另请参见：

- “临时表”一节第 151 页
- “视图”一节第 155 页

基于会话的同步

一种同步类型，这种同步会使数据表示在统一数据库和远程数据库都一致。MobiLink 基于会话。

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

基于脚本的上载

MobiLink 中一种将上载过程自定义为使用日志文件的替代方法的方式。

基于 SQL 的同步

MobiLink 中一种使用 MobiLink 事件将表数据与支持 MobiLink 的统一数据库进行同步的方式。对于基于 SQL 的同步，可以直接使用 SQL，也可以使用面向 Java 和 .NET 平台的 MobiLink 服务器 API 返回 SQL。

基于文件的下载

在 MobiLink 中同步数据的一种方式，其中下载以文件的方式进行分发，从而支持脱机分发同步更改。

集成登录

一种登录功能，它允许将同一个用户 ID 和口令用于操作系统登录、网络登录和数据库连接。

监听器

一个程序 (dblsn)，用于 MobiLink 服务器启动的同步。监听器安装在远程设备上，它们被配置为在接收到来自通告程序的信息时启动针对设备的操作。

另请参见： “服务器启动的同步”一节第 145 页

检查点

将对数据库的所有更改都保存到数据库文件中的时间点。在其它时间，所提交的更改仅保存到事务日志中。

检查约束

对列或列集强制实施指定条件的一种限制。

另请参见：

- “约束”一节第 167 页
- “外键约束”一节第 161 页
- “主键约束”一节第 168 页
- “唯一约束”一节第 162 页

脚本

MobiLink 中为处理 MobiLink 事件而编写的代码。脚本通过编程方式控制数据交换，以满足业务需要。

另请参见： “事件模型”一节第 155 页

脚本版本

MobiLink 中为创建同步而一起应用的一组同步脚本。

校验

测试数据库、表或索引是否受到特定类型的文件损坏。

校验和

随数据库页本身一起记录的计算出的数据库页位数。校验和能够确保数据库页写入磁盘时位数相符，因此数据库管理系统可以通过它来验证数据库页的完整性。如果计数相符，即认为数据库页已成功写入。

镜像日志

另请参见： “事务日志镜像”一节第 156 页

角色

概念性数据库建模中从一个角度描述某种关系的动词或短语。您可以用两个角色来描述每种关系。例如，"包含" 和 "隶属于" 便是角色。

角色名

外键的名称。由于它命名外表和主表之间的关系，因此称作角色名。缺省情况下，角色名就是表名，除非其它外键已经使用该名称（在这种情况下，缺省的角色名是表名后接一个三位的唯一数字）。也可以自己创建角色名。

另请参见： “外键”一节第 160 页

局部临时表

一种临时表，仅在复合语句执行期间或连接结束之前存在。当您只需要将数据集装载一次时，局部临时表非常有用。缺省情况下，行会在提交时被删除。

另请参见：

- “临时表”一节第 151 页
- “全局临时表”一节第 154 页

客户端/服务器

一种软件体系结构，在这种体系结构中，一个应用程序（客户端）从另一个应用程序（服务器）获取信息并向该应用程序发送信息。这两个应用程序常位于通过网络连接的不同计算机上。

客户端消息存储库

QAnywhere 中一种用于在远程设备上存储消息的 SQL Anywhere 数据库。

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

客户端消息存储库 ID

QAnywhere 中一种对客户端消息存储库进行唯一标识的 MobiLink 远程 ID。

快照隔离

一种为发出读请求的事务返回数据的已提交版本的隔离级别。SQL Anywhere 提供了以下三种快照隔离级别：快照、语句快照和只读语句快照。使用快照隔离时，读操作不会阻塞写操作。

另请参见： “隔离级别”一节第 146 页

连接

关系系统中的一种基本操作，它通过比较指定列中的值将两个或更多个表中的行链接在一起。

连接 ID

用于标识客户端应用程序与数据库之间给定连接的唯一编号。可以使用以下 SQL 语句来确定当前连接 ID：

```
SELECT CONNECTION_PROPERTY( 'Number' );
```

连接类型

SQL Anywhere 提供了四种类型的连接：交叉连接、键连接、自然连接和使用 ON 子句的连接。

另请参见： “连接”一节第 151 页

连接配置

连接到数据库所需的一组参数，如用户名、口令和服务器名称，它们在存储后即可方便地使用。

连接启动的同步

一种 MobiLink 服务器启动的同步，在这种同步下，连接发生变化时会启动同步。

另请参见： “服务器启动的同步”一节第 145 页

连接条件

一种影响连接结果的限制。您可以通过紧跟在连接语句的后面插入 ON 子句或 WHERE 子句来指定连接条件。对于自然连接和关键连接，SQL Anywhere 会生成连接条件。

另请参见：

- “连接”一节第 151 页
- “生成的连接条件”一节第 156 页

临时表

为临时存储数据而创建的表。有两种类型：全局临时表和局部临时表。

另请参见：

- “局部临时表”一节第 150 页
- “全局临时表”一节第 154 页

LTM

日志传送管理器（Log Transfer Manager，简称 LTM）也称作复制代理。LTM 是一个与 Replication Server 一起使用的程序，它读取数据库事务日志并将提交的更改发送到 Sybase 复制服务器。

请参见： “复制服务器”一节第 146 页

轮询

在 MobiLink 服务器启动的同步中，轻量级轮询器（例如 MobiLink 监听器）从通告程序请求推式通知的方式。

另请参见： “服务器启动的同步”一节第 145 页

逻辑索引

指向物理索引的引用（指针）。磁盘上不存储逻辑索引的索引结构。

命令文件

包含 SQL 语句的文本文件。命令文件可以手工建立，也可以通过数据库实用程序自动建立。例如，dbunload 实用程序会创建一个命令文件，其中包含重新创建给定数据库所需的 SQL 语句。

MobiLink

一种基于会话的同步技术，其设计用途是将 UltraLite 和 SQL Anywhere 远程数据库与统一数据库同步。

另请参见：

- “统一数据库”一节第 159 页
- “同步”一节第 159 页
- “UltraLite”一节第 160 页

MobiLink 服务器

运行 MobiLink 同步的计算机程序，即 mlsrv11。

MobiLink 监控器

一种用于监控 MobiLink 同步的图形化工具。

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

MobiLink 客户端

有两种 MobiLink 客户端。对于 SQL Anywhere 远程数据库，MobiLink 客户端是 dbmlsync 命令行实用程序。对于 UltraLite 远程数据库，MobiLink 客户端内置于 UltraLite 运行时库中。

MobiLink 系统表

MobiLink 同步所需的系统表。它们由 MobiLink 安装程序脚本安装到 MobiLink 统一数据库中。

MobiLink 用户

MobiLink 用户用于与 MobiLink 服务器进行连接。在远程数据库上创建 MobiLink 用户，然后在统一数据库中注册该用户。MobiLink 用户名完全独立于数据库用户名。

模式

数据库的结构，其中包括表、列和索引以及它们之间的关系。

内连接

一种连接，在这种连接中，仅当两个表都满足连接条件时行才会出现在结果集中。内连接是缺省设置。

另请参见：

- “连接”一节第 151 页
- “外连接”一节第 161 页

ODBC

开放式数据库连接。一种用于与数据库管理系统连接的标准 Windows 接口。ODBC 是 SQL Anywhere 所支持的几种接口之一。

ODBC 管理器

一种随 Windows 操作系统提供的 Microsoft 程序，用于设置 ODBC 数据源。

ODBC 数据源

用户要通过 ODBC 访问的数据的规范以及获取该数据时所需的信息。

PDB

Palm 数据库文件。

PowerDesigner

一种数据库建模应用程序。PowerDesigner 为设计数据库或数据仓库提供了结构化的方法。SQL Anywhere 包括 PowerDesigner 的 Physical Data Model 组件。

PowerJ

一种 Sybase 产品，用于开发 Java 应用程序。

QAnywhere

应用程序到应用程序的消息传递（包括移动设备到移动设备和移动设备与企业之间的消息传递），它使在移动或无线设备上运行的自定义程序能够与处在中央位置的服务器应用程序进行通信。

QAnywhere 代理

QAnywhere 中一种运行在客户端设备上的进程，用于监控客户端消息存储库和确定应在何时传输消息。

嵌入式 SQL

一种 C 语言程序编程接口。SQL Anywhere 嵌入式 SQL 是 ANSI 和 IBM 标准的实现。

轻量级轮询器

在 MobiLink 服务器启动的同步中，轮询来自 MobiLink 服务器的推式通知的设备应用程序。

另请参见： “服务器启动的同步”一节第 145 页

全局临时表

一种临时表，在被显式地删除之前，其数据定义对所有用户都可见。全局临时表允许用户各自打开一个表的相同实例。缺省情况下，行在提交时被删除，并且始终是在连接结束时被删除。

另请参见：

- “临时表”一节第 151 页
- “局部临时表”一节第 150 页

日志文件

SQL Anywhere 所维护的事务日志。该日志文件用于确保在出现系统或介质故障时可以恢复数据库、提高数据库性能以及使用 SQL Remote 实现数据复制。

另请参见：

- “事务日志”一节第 155 页
- “事务日志镜像”一节第 156 页
- “完全备份”一节第 161 页

散列

散列是一种将索引条目转化为键的索引优化。索引散列旨在通过将足够的行实际数据与其行 ID 包括在一起，以避免进行先查找行、后装载行然后再将行解出才能得出索引值的高开销操作。

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

上载

同步过程的一个阶段，在此阶段数据从远程数据库传送到统一数据库。

设备跟踪

在 MobiLink 服务器启动的同步中，允许使用标识设备的 MobiLink 用户名来对消息进行寻址的功能。

另请参见： “服务器启动的同步”一节第 145 页

实例化视图

实例化视图是指已计算并已存储在磁盘上的视图。实例化视图同时具有视图的特征（使用查询说明进行定义）和表的特征（可以对其执行大多数表操作）。

另请参见：

- “基表”一节第 148 页
- “视图”一节第 155 页

世代号

MobiLink 中的一种机制，用于强制远程数据库先上载数据，然后再应用任何其它下载文件。

另请参见： “基于文件的下载”一节第 149 页

事件模型

MobiLink 中组成同步的事件（如 begin_synchronization 和 download_cursor）序列。如果为事件创建了脚本，则会调用事件。

视图

一种作为对象存储在数据库中的 SELECT 语句。它使用户能够看到一个或多个表中的行子集或列子集。每当用户使用特定表或表组合的视图时，都将利用存储在这些表中的信息重新计算视图。视图对确保安全以及定制数据库信息的外观来使数据访问简单明了有帮助。

事务

组成一个逻辑工作单元的 SQL 语句序列。事务要么全部得到处理，要么根本不做处理。SQL Anywhere 支持事务处理，并内置了锁定功能，使并发事务能够访问数据库而又不损坏数据。事务要么以 COMMIT 语句结束，该语句使对数据的更改成为永久性更改；要么以 ROLLBACK 语句结束，该语句撤消在事务执行过程中所做的全部更改。

事务日志

一种按进行更改的顺序存储对数据库所做全部更改的文件。它会提高性能并支持在数据库文件损坏时恢复数据。

事务日志镜像

同时维护的事务日志文件的完全相同副本（可选）。每当数据库更改写入事务日志文件时，也会同时写入事务日志镜像文件。

镜像文件应该与事务日志保留在不同的设备上，这样在任意设备出现故障时，日志的其它副本会确保数据可以安全地恢复。

另请参见： “事务日志”一节第 155 页

事务完整性

MobiLink 中对整个同步系统事务的有保证维护。要么同步整个事务，要么不对事务的任何部分进行同步。

生成的连接条件

一种自动生成的对连接结果的限制。有两种类型：关键和自然。指定 KEY JOIN 或指定关键字 JOIN 但不使用关键字 CROSS、NATURAL 或 ON 时，会生成关键连接。对于关键连接，所生成的连接条件取决于表之间的外键关系。指定 NATURAL JOIN 时会生成自然连接；所生成的连接条件基于两个表中的公用列名。

另请参见：

- “连接”一节第 151 页
- “连接条件”一节第 151 页

受保护的功能

数据库服务器启动时由 -sf 选项指定的功能，该数据库服务器上运行的任何数据库都无法使用该功能。

授权选项

一种权限级别，它允许用户向其他用户授予权限。

数据操作语言 (DML)

用于操作数据库中数据的 SQL 语句子集。DML 语句可以检索、插入、更新和删除数据库中的数据。

数据定义语言 (DDL)

用于定义数据库中数据结构的 SQL 语句子集。DDL 语句可以创建、修改和删除数据库对象（如表和用户）。

数据类型

数据的格式，如 CHAR 或 NUMERIC。在 ANSI SQL 标准中，数据类型也可以包括对大小、字符集和归类的限制。

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

另请参见： “域”一节第 165 页

数据立方体

一种多维结果集，每一维都以不同的方式对相同的结果进行分组和排序。数据立方体提供了有关数据的综合性信息，如果不使用数据立方体，要获得同样的信息就必须进行自连接查询和相关子查询。数据立方体是 OLAP 功能的一部分。

数据库

通过主键和外键关联的表的集合。表包含数据库中的信息。表和键一起定义数据库的结构。数据库管理系统会访问此信息。

另请参见：

- “外键”一节第 160 页
- “主键”一节第 168 页
- “数据库管理系统 (DBMS)”一节第 157 页
- “关系数据库管理系统 (RDBMS)”一节第 146 页

数据库对象

包含或接收信息的数据库组件。表、索引、视图、过程和触发器便是数据库对象。

数据库服务器

对所有针对数据库信息的访问进行管理的计算机程序。SQL Anywhere 提供了两种类型的服务器：网络服务器和个人服务器。

数据库管理系统 (DBMS)

用于创建和使用数据库的程序的集合。

另请参见： “关系数据库管理系统 (RDBMS)”一节第 146 页

数据库管理员 (DBA)

具有维护数据库所需权限的用户。DBA 通常负责对数据库模式的所有更改以及管理用户和组。数据库管理员角色自动内置于数据库中，其用户 ID 为 DBA，口令是 sql。

数据库连接

客户端应用程序与数据库之间的通信渠道。必须具有有效的用户 ID 和口令才能建立连接。为用户 ID 授予的特权决定了在连接过程中可以执行的操作。

数据库名称

服务器装载数据库时为数据库指定的名称。缺省数据库名是初始数据库文件的文件名（不含扩展名）。

另请参见： “数据库文件”一节第 158 页

数据库所有者 (dbo)

一种特殊的用户，他拥有不归 SYS 所有的系统对象。

另请参见：

- “数据库管理员 (DBA)”一节第 157 页
- “SYS”一节第 159 页

数据库文件

数据库保存在一个或多个数据库文件中。其中一个是初始文件，后面的文件称作 dbspace。每个表（包括其索引）都必须包含在单个数据库文件中。

另请参见： “dbspace”一节第 144 页

死锁

一组事务会进入的一种特殊状态，在该状态下这些事务都不能继续执行。

SQL

用于与关系数据库进行通信的语言。ANSI 定义了 SQL 的标准，其最新标准是 SQL-2003。SQL 的非官方全称是结构化查询语言。

SQL Anywhere

SQL Anywhere 的关系数据库服务器组件，专供在移动和嵌入式环境中使用，或作为中小型企业的服务器使用。SQL Anywhere 也是包含 SQL Anywhere RDBMS、UltraLite RDBMS、MobiLink 同步软件和其它组件的软件包的名称。

SQL Remote

一种基于消息的数据复制技术，用于在统一数据库与远程数据库之间进行双向复制。统一数据库和远程数据库必须是 SQL Anywhere。

SQL 语句

包含用于将指令传递给 DBMS 的 SQL 关键字的字符串。

另请参见：

- “模式”一节第 153 页
- “SQL”一节第 158 页
- “数据库管理系统 (DBMS)”一节第 157 页

锁定

一种在同时执行多个事务的过程中保护数据完整性的并发控制机制。SQL Anywhere 会自动应用锁以防止两个连接同时更改同一数据，并防止其它连接读取正接受更改的数据。

您可以通过设置隔离级别来控制锁定。

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

另请参见：

- “隔离级别”一节第 146 页
- “并发”一节第 141 页
- “完整性”一节第 161 页

索引

一组已排序的、与基表中的一个或多个列关联的键和指针。在表中一个或多个列上设置索引可以提高性能。

Sybase Central

一种数据库管理工具，通过图形用户界面提供 SQL Anywhere 数据库设置、属性和实用程序。Sybase Central 也可用于管理其它 Sybase 产品，其中包括 MobiLink。

SYS

一种拥有大多数系统对象的特殊用户。无法以 SYS 身份登录。

统一数据库

在分布式数据库环境中，是指用于存储数据主副本的数据库。出现冲突或差异时，将把统一数据库视为具有数据的主副本。

另请参见：

- “同步”一节第 159 页
- “复制”一节第 145 页

通信流

MobiLink 中 MobiLink 客户端与 MobiLink 服务器之间进行通信时所使用的网络协议。

通告程序

一种由 MobiLink 服务器启动的同步使用的程序。通告程序集成在 MobiLink 服务器中。它们会检查统一数据库是否有推式请求，并发送推式通知。

另请参见：

- “服务器启动的同步”一节第 145 页
- “监听器”一节第 149 页

同步

利用 MobiLink 技术在数据库之间复制数据的过程。

在 SQL Remote 中，同步专指以初始数据集初始化远程数据库的过程。

另请参见：

- “MobiLink”一节第 152 页
- “SQL Remote”一节第 158 页

推式请求

在 MobiLink 服务器启动的同步中，通告程序通过检查它来确定推式通知是否需要发送到设备的结果集中的一行值。

另请参见： “服务器启动的同步”一节第 145 页

推式通知

QAnywhere 中一种从服务器传送到 QAnywhere 客户端的特殊消息，用于提示客户端启动消息传输。在 MobiLink 服务器启动的同步中，从通告程序传送到包含推式请求数据和内部信息的设备的特殊消息。

另请参见：

- “QAnywhere”一节第 154 页
- “服务器启动的同步”一节第 145 页

UltraLite

一种针对小型设备、移动设备和嵌入式设备进行了优化的数据库。所面向的平台包括手机、传呼机和个人记事本。

UltraLite 运行时

一种过程中关系数据库管理系统，其中包括一个内置 MobiLink 同步客户端。每个 UltraLite 编程接口使用的库以及 UltraLite 引擎中都包括 UltraLite 运行时。

外表

包含外键的表。

另请参见： “外键”一节第 160 页

外部登录

与远程服务器通信时使用的替代登录名和口令。缺省情况下，SQL Anywhere 每次代表其客户端连接到远程服务器时都会使用这些客户端的名称和口令。但是，您可以通过创建外部登录来替换这一缺省设置。外部登录是指与远程服务器通信时使用的替代登录名和口令。

外键

一个表中复制另一个表中主键值的一个或多个列。外键建立表间的关系。

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

另请参见：

- “主键”一节第 168 页
- “外表”一节第 160 页

外键约束

对单个列或一组列的限制，指定表中的数据与某个其它表中数据的关系。对列集施加外键约束可使这些列成为外键。

另请参见：

- “约束”一节第 167 页
- “检查约束”一节第 149 页
- “主键约束”一节第 168 页
- “唯一约束”一节第 162 页

外连接

一种保留表中所有行的连接。SQL Anywhere 支持左、右和完全外连接。左外连接保留表中位于连接运算符左侧的行，当右表中的行不满足连接条件时，它将返回空值。完全外连接保留两个表中的所有行。

另请参见：

- “连接”一节第 151 页
- “内连接”一节第 153 页

完全备份

对整个数据库和事务日志（可选）的备份。完全备份包含数据库中的所有信息，因此可以在系统或介质出现故障时提供保护。

另请参见： “增量备份”一节第 167 页

完整性

遵守完整性规则的情况，完整性规则确保数据正确并准确，而且数据库的关系结构保持不变。

另请参见： “参照完整性”一节第 142 页

网关

一种 MobiLink 对象，存储在 MobiLink 系统表或通告程序属性文件中，包含有关如何发送用于服务器启动同步的消息的信息。

另请参见： “服务器启动的同步”一节第 145 页

网络服务器

从共享公共网络的计算机接受连接的数据库服务器。

另请参见： “个人服务器”一节第 146 页

网络协议

通信类型，如 TCP/IP 或 HTTP。

维护版本

维护版本是一套完整的软件，它升级已安装的具有相同主版本号的较早版本的软件（版本号格式是 *major.minor.patch.build*）。升级程序的发行说明中列出了错误修正软件和其它更改。

唯一约束

对某个列或一组列的限制，它要求所有非空值都各不相同。一个表可以有多个唯一约束。

另请参见：

- “外键约束”一节第 161 页
- “主键约束”一节第 168 页
- “约束”一节第 167 页

谓语

一种条件表达式，可以选择性地将其与逻辑运算符 AND 和 OR 组合在一起，以组成 WHERE 或 HAVING 子句中的条件集。在 SQL 中，求值结果为 UNKNOWN 的谓语将解释为 FALSE。

位数组

位数组是一种用于有效率地存储位序列的数组数据结构。位数组与字符串类似，不同的是其各个部分由 0（零）和 1（一）而不是字符组成。位数组通常用于保存一串布尔值。

Windows

Microsoft Windows 操作系统系列，如 Windows Vista、Windows XP 和 Windows 200x。

Windows CE

请参见“Windows Mobile”一节第 162 页。

Windows Mobile

Microsoft 为移动设备制造的操作系统系列。

文件定义数据库

MobiLink 中一种用于创建下载文件的 SQL Anywhere 数据库。

另请参见： “基于文件的下载”一节第 149 页

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

物理索引

索引存储在磁盘上的实际索引结构。

系统表

一种表，由 SYS 或 dbo 拥有，用于保存元数据。系统表也称作数据字典表，由数据库服务器创建并维护。

系统对象

由 SYS 或 dbo 拥有的数据库对象。

系统视图

存在于每一个数据库中的一种视图，它以易于理解的格式表示系统表中包含的信息。

下载

同步过程的一个阶段，在此阶段数据从统一数据库传送到远程数据库。

相关名

查询的 FROM 子句中使用的表或视图的名称—要么是表或视图的原始名称，要么是在 FROM 子句中定义的替代名称。

项目

在 MobiLink 或 SQL Remote 中，项目是表示整个表或表中行和列子集的数据库对象。项目在发布中组合在一起。

另请参见：

- “复制”一节第 145 页
- “发布”一节第 144 页

消息存储库

QAnywhere 中客户端和服务器设备上存储消息的数据库。

另请参见：

- “客户端消息存储库”一节第 150 页
- “服务器消息存储库”一节第 145 页

消息类型

SQL Remote 复制中指定远程用户与统一数据库发布者通信方式的数据库对象。一个统一数据库可能定义了几种消息类型，这样一来，不同的远程用户就可以使用不同的消息系统与统一数据库进行通信。

另请参见：

- “复制”一节第 145 页
- “统一数据库”一节第 159 页

消息日志

可存储来自数据库服务器或 MobiLink 服务器等应用程序的消息的日志。此类信息还可以出现在消息窗口中或记录到文件中。消息日志包括信息性消息、错误、警告以及来自 MESSAGE 语句的消息。

消息系统

SQL Remote 复制中用于在统一数据库与远程数据库之间交换消息的协议。SQL Anywhere 包括对以下消息系统的支持：FILE、FTP 和 SMTP。

另请参见：

- “复制”一节第 145 页
- “FILE”一节第 145 页

卸载

卸载数据库时会将数据库的结构和/或数据导出到文本文件（如果是结构，则导出到 SQL 命令文件中；如果是数据，则导出到 ASCII 逗号分隔文件中）。使用卸载实用程序来卸载数据库。

此外，您也可以使用 UNLOAD 语句卸载数据的选定部分。

性能统计

反映数据库系统性能的值。例如，CURRREAD 统计表示数据库服务器已发出但尚未完成的文件读取次数。

业务规则

基于实际要求的准则。通常，业务规则通过检查约束、用户定义数据类型以及事务的正确使用来实现。

另请参见：

- “约束”一节第 167 页
- “用户定义数据类型”一节第 165 页

引用对象

一种对象（如视图），其定义直接引用数据库中的另一个对象（如表）。

另请参见： “外键”一节第 160 页

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

用户定义数据类型

请参见“域”一节第 165 页。

游标

指向结果集的已命名链接，用于通过编程接口访问和更新行。在 SQL Anywhere 中，游标支持在查询结果中进行向前和向后移动。游标由两部分组成：游标结果集（通常由 SELECT 语句定义）和游标位置。

另请参见：

- “游标结果集”一节第 165 页
- “游标位置”一节第 165 页

游标结果集

与游标关联的查询所得到的行集。

另请参见：

- “游标”一节第 165 页
- “游标位置”一节第 165 页

游标位置

指向游标结果集中一个行的指针。

另请参见：

- “游标”一节第 165 页
- “游标结果集”一节第 165 页

语句级触发器

在整个触发语句完成后执行的触发器。

另请参见：

- “触发器”一节第 143 页
- “行级触发器”一节第 147 页

域

内置数据类型的别名，其中包括适用的精度值和小数位值，还可以选择是否包括 DEFAULT 值和 CHECK 条件。SQL Anywhere 中预定义了一些域，如货币数据类型。也称作用户定义数据类型。

另请参见： “数据类型”一节第 156 页

预订

MobiLink 同步中发布与 MobiLink 用户之间的客户端数据库中的一个链接，它使发布所描述的数据能够得到同步。

SQL Remote 复制中发布与远程用户之间的一种链接，它使用户能够与统一数据库交换该发布上的更新。

另请参见：

- “发布”一节第 144 页
- “MobiLink 用户”一节第 153 页

元数据

数据的数据。元数据描述其它数据的性质和内容。

另请参见： “模式”一节第 153 页

原子事务

保证成功完成或保证根本不予完成的事务。如果错误使原子事务的一部分无法完成，则将回退事务以防止数据库处于不一致的状态。

REMOTE DBA 特权

在 SQL Remote 中，消息代理 (dbremote) 所需的权限级别。MobiLink 中 SQL Anywhere 同步客户端 (dbmlsync) 所需的权限级别。当消息代理或同步客户端作为具有该权限的用户建立连接时，它将具有完全的 DBA 访问权。如果不是通过消息代理或同步客户端进行连接，则该用户 ID 将不具有附加权限。

另请参见： “DBA 权限”一节第 144 页

远程 ID

SQL Anywhere 和 UltraLite 数据库中一种由 MobiLink 使用的唯一标识符。远程 ID 初始情况下设置为 NULL，在数据库第一次同步期间将设置为 GUID。

远程数据库

MobiLink 或 SQL Remote 中一种与统一数据库交换数据的数据库。远程数据库可以共享统一数据库中的全部或部分数据。

另请参见：

- “同步”一节第 159 页
- “统一数据库”一节第 159 页

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

约束

对特定数据库对象（如表或列）中所包含值的限制。例如，列可以具有唯一性约束，该约束要求该列中的所有值互不相同。表可以具有外键约束，该约束指定该表中的信息与某个其它表中数据的关系。

另请参见：

- “检查约束”一节第 149 页
- “外键约束”一节第 161 页
- “主键约束”一节第 168 页
- “唯一约束”一节第 162 页

运营公司

一种 MobiLink 对象，存储在 MobiLink 系统表或通告程序属性文件中，包含有关供服务器启动的同步使用的公共运营公司的信息。

另请参见： “服务器启动的同步”一节第 145 页

增量备份

仅包含事务日志的备份，通常在两次完全备份之间使用。

另请参见： “事务日志”一节第 155 页

争用

为获取资源而竞争的行为。例如，就数据库而言，如果有两个或更多个用户试图编辑数据库的同一行，就会为获得编辑该行的权利而发生争用。

正则表达式

正则表达式是字符、通配符和运算符的序列，用于定义某种模式以在字符串内进行搜索。

直方图

直方图是列统计信息最重要的组成部分，是一种表示数据分布的方式。SQL Anywhere 维护直方图以为优化程序提供有关列值分布情况的统计信息。

直接行处理

MobiLink 中一种用于将表数据同步到 MobiLink 支持的统一数据库以外的数据源的方法。使用直接行处理时，上载和下载都可以实现。

另请参见：

- “统一数据库”一节第 159 页
- “基于 SQL 的同步”一节第 149 页

主表

包含外键关系中的主键的表。

主键

其值唯一标识表中各行的一个列或多个列。

另请参见： “外键”一节第 160 页

主键约束

一种对主键列的唯一性约束。一个表只能有一个主键约束。

另请参见：

- “约束”一节第 167 页
- “检查约束”一节第 149 页
- “外键约束”一节第 161 页
- “唯一约束”一节第 162 页
- “完整性”一节第 161 页

子查询

嵌套在 SELECT、INSERT、UPDATE 或 DELETE 语句或者其它子查询中的 SELECT 语句。

有两种类型的子查询：相关子查询和嵌套子查询。

字符串

字符串是以单引号围起的字符序列。

字符集

字符集是一组符号，包括字母、数字、空格和其它符号。字符集的一个例子是 ISO-8859-1，又称作 Latin1。

另请参见：

- “代码页”一节第 143 页
- “编码”一节第 141 页
- “归类”一节第 147 页

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

索引

其它

A

B

C

D

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

E

F

G

H

I

J

K

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

L

M

N

O

P

Q

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

R

S

T

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

U

W

X

Y

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

Z