

Wonderware® FactorySuite™ SQL Access Manager

Руководство пользователя

Редакция А
Декабрь 1997

Wonderware Corporation

Все права сохранены. Никакая часть настоящей документации не может воспроизводиться, храниться в информационно системе или передаваться любым способом, электронным или механическим, или путем фотокопирования, записи или как-то иначе без предварительного письменного согласия корпорации Wonderware Corporation. Использование содержащейся здесь информации не влечет за собой никакой ответственности, связанной с авторскими или патентными правами. Хотя при подготовке этой документации авторами и издателями были приложены все усилия, они не несут ответственность за возможные ошибки или неточности, равно как за возможный ущерб, причиненный в результате использования содержащейся здесь информации.

Содержащаяся здесь информация может быть изменена без предупреждения и не носит характера обязательств со стороны корпорации Wonderware. Описанное в документации программное обеспечение предоставляется по соглашениям о лицензии и неразглашении. Это программное обеспечение может использоваться только на условиях данных соглашений.

Данное руководство переведено с английской версии А от 12/97.

© 1997 Wonderware Corporation. Все права сохранены.

100 Technology Drive
Irvine, CA 92618
U.S.A.
(949) 727-3200
<http://www.wonderware.com>

Торговые марки

Все упоминаемые в этой книге названия, известные как торговые или сервисные марки, помечены должным образом. Корпорация Wonderware не может гарантировать точность этой информации. Использование любых названий в этой книге не следует считать нарушением каких-либо торговых или сервисных марок.

Wonderware – зарегистрированная торговая марка Wonderware Corporation.

Wonderware FactorySuite, InTouch, WindowMaker, WindowViewer, SQL Access Manager, Recipe Manager, SPC Pro, DBDump, DBLoad, HDMerge, HistData, Wonderware Logger, InControl, InTrack, InBatch, IndustrialSQL, FactoryOffice, Scout, SuiteLink и NetDDE – торговые марки Wonderware Corporation.

Оглавление

Глава 1 - SQL Access Manager	1-1
Введение	1-2
Об этом руководстве	1-3
Техническая поддержка	1-4
Просмотр лицензии FactorySuite	1-4
Совместимость с ODBC	1-5
 Глава 2 -	
Настройка и соединение баз данных	2-1
Работа с Oracle 6	2-2
Настройка клиента	2-2
Работа с Oracle 7.2	2-4
Настройка клиента	2-4
Запись даты и времени в поле даты Oracle	2-7
Работа с Sybase или Microsoft SQL Server	2-8
Настройка клиента	2-8
Поддерживаемые типы данных	2-9
Работа с dBASE	2-10
Работа с Microsoft Access	2-12
Работа с Paradox	2-13
Значения типов данных для поддерживаемых СУБД	2-14
 Глава 3 - Настройка SQL Access Manager	3-1
Обзор программы SQL Access Manager	3-2
Настройка списка привязок	3-2
Использование специальных разделителей	3-6
Настройка шаблонов таблиц	3-7
Файл SQL.DEF	3-10
 Глава 4 - Использование функций SQL	4-1
Функции SQL	4-2
Параметры SQL	4-5
Использование функций SQL в сценариях InTouch	4-8
Определение сложных запросов	4-8
Динамическое создание запросов	4-9
Чтение инструкций SQL из файла	4-9
Изменение расширенных инструкций SQL	4-10
Выполнение расширенных инструкций SQL	4-11
Поддержка хранимых в памяти процедур	4-12

Глава 5 - Отладка	5-1
Функции отладки	5-2
Сообщения об ошибках с кодом результата	5-2
Сообщения об ошибках базы данных	5-3
 Приложение А - Резервные ключевые слова	
SQL Access	
и ODBC	A-1
 Предметный указатель	I-1

Г Л А В А 1

SQL Access Manager

Входящая в пакет Wonderware® FactorySuite™ программа SQL Access Manager позволяет обращаться к базе данных, изменять, создавать или удалять в ней таблицы. Информация базы данных хранится в таблицах, имеющих общие атрибуты или поля. Structured Query Language (SQL) — это язык, используемый для получения этой информации.

Содержание

- Введение
- Об этом руководстве
- Техническая поддержка
- Просмотр лицензии FactorySuite
- Совместимость с ODBC

Введение

Программа InTouch SQL Access Manager разработана для передачи данных, таких как пакетные рецепты, из базы данных SQL в приложения InTouch. Эта программа используется также для ускоренной передачи данных, статуса алармов или архивных данных в среде выполнения из приложения InTouch в базу данных SQL. Например, после завершения машинного цикла может потребоваться сохранение части данных для различных приложений. База данных SQL легко обеспечивает обмен информацией между различными приложениями третьих фирм. SQL Access Manager позволяет получать доступ к этим данным и просматривать их из любого приложения InTouch.

Продукт InTouch SQL Access состоит из программы SQL Access Manager и SQL функций. Программа SQL Access Manager используется для создания колонок базы данных и их привязки к тэгам в имеющемся словаре тэгов InTouch. Процесс привязки колонок базы данных к тэгам называется "связыванием". Связывание позволяет программе SQL Access Manager манипулировать данными непосредственно в базе данных. SQL Access Manager сохраняет имена полей базы данных и их связи в файле формата переменных, разделенных запятыми (CSV). Этот файл называется "SQL.DEF" и постоянно хранится в каталоге приложения InTouch, где его можно просмотреть или изменить с помощью программы SQL Access Manager или любого текстового редактора, такого как Notepad (Блокнот). Кроме того, программа SQL Access Manager создает шаблоны таблиц, определяющие структуру и формат базы данных.

☞ Для получения дополнительной информации по спискам связей и шаблонам таблиц смотри главу 3, "Настройка SQL Access Manager".

Функции SQL могут использоваться в любом действующем сценарии InTouch и автоматически выполняться при вводе оператором, при изменении значения тэга или при существовании ряда условий. Например, если возникает ситуация аларма, то приложение будет выполнять команду SQLInsert() или SQLUpdate(), чтобы сохранить все используемые точки данных и состояние аларма. С помощью функций SQL можно создавать новые таблицы, заносить новые записи в таблицу, редактировать уже существующие записи, очищать или удалять таблицы, выбирать записи, перемещаться по ним и т. п.

Продукт InTouch SQL Access использует библиотеку Intersolv/Q+E Software Database Library QELIB, которая работает со множеством различных СУБД и включает все коды, необходимые для взаимодействия с драйверами ODBC. Эти драйверы не включены в программный продукт InTouch SQL Access и должны приобретаться отдельно у третьих фирм (например, Intersolv/Q+E или Microsoft). Драйверы ODBC необходимы для непосредственного «соединения» с базой данных. Например, для внедрения функций SQL в базу данных Microsoft® Access необходимо приобрести и должным образом установить набор драйверов Microsoft® ODBC Desktop. Следующий раздел этой главы содержит некоторые общие примеры по конфигурации драйверов.

Примечание. СУБД, о которых не идет речь в этом руководстве, не поддерживаются.

Об этом руководстве

Это руководство логически разделено на несколько глав, которые описывают различные аспекты использования программы SQL Access Manager. Они написаны в форме «процедур», которые по шагам рассказывают, как выполнять те или иные функции и задачи.

☞ Если при просмотре электронной версии данного руководства вам встретилась подобная ссылка, то это — «переход» к другому разделу. Нажав на него, вы окажетесь в указанном разделе или главе документации. После перехода в другой раздел у вас всегда имеется возможность перейти «назад» к исходному разделу.

📖 Такой тип перекрестных ссылок указывает, что дополнительная информация содержится в другой книге FactorySuite.

🔑 Такие "подсказки" указывают более легкий или быстрый путь выполнения функции или задачи.

«Руководство администратора FactorySuite» содержит исчерпывающую информацию об общих компонентах FactorySuite и требованиях к системе, советы по работе в сети, интеграции продуктов, сведения о технической поддержке и многое другое.

«Руководство пользователя InTouch» поможет ознакомиться со средой разработки и инструментальными средствами WindowMaker (глава 1, "Программные компоненты WindowMaker"). О том, как работать с окнами, графическими объектами, мастерами, элементами ActiveX и т. п., рассказывается в главе 2, "Работа с WindowMaker".

Подробное описание среды выполнения InTouch (WindowViewer) содержится в книге «Руководство пользователя InTouch Runtime».

«Справочное руководство по InTouch» содержит детальные сведения о языке сценариев InTouch, о системных тэгах и полях тэгов.

🔑 В пакет программ FactorySuite входит электронная документация по всем компонентам, включенным в пакет. Например, FactorySuite System Administrator's Guide, SPC Pro, SQLAccess Manager, Recipe Manager, IndustrialSQL Server, InControl и все 32-битные серверы ввода/вывода Wonderware. Если вы приобрели пакет FactorySuite Plus, то в нем также имеется электронная документация для компонентов InTrack и InBatch.

Требования к пользователю

Это руководство рассчитано на пользователя, который:

- Знаком с операционными системами Windows 95 и/или Windows NT.
- Умеет работать с мышью, меню Windows, выбирать параметры и пользоваться справочной системой.
- Имеет опыт программирования или знаком с одним из макроязыков. Необходимо также иметь представление об основных понятиях программирования, таких как переменные, выражения, функции и методы.

Техническая поддержка

Служба технической поддержки корпорации Wonderware оказывает различные виды помощи, связанные с любыми аспектами использования продуктов Wonderware.

Прежде чем обращаться в службу технической поддержки, попытайтесь найти в соответствующих главах *"Руководства пользователя SQL Access Manager"* возможные решения тех проблем, которые возникли у вас при работе с системой. Если же обращение в службу технической поддержки окажется необходимым, будьте готовы сообщить следующую информацию:

1. Серийный номер программного обеспечения.
2. Номер версии InTouch.
3. Тип и номер версии используемой операционной системы. Например, Microsoft Windows NT workstation версии 4.0.
4. Точный текст полученного сообщения об ошибке.
5. Любые, важные на ваш взгляд, листинги Wonderware Logger, утилиты Microsoft Diagnostic (MSD) или другой утилиты диагностики.
6. Описание предпринятых вами попыток по устранению проблемы и полученных результатов.
7. Инструкции по воспроизведению проблемы.
8. Номер, присвоенный вашему запросу службой технической поддержки Wonderware (если вы обращаетесь с этой проблемой не первый раз).

 Подробные сведения о службе технической поддержки даны в электронном *«Руководстве администратора системы FactorySuite»*.

Просмотр лицензии FactorySuite

Информация о лицензии на систему FactorySuite может быть получена при помощи утилиты просмотра лицензии, которая запускается из диалогового окна InTouch Справка/**О программе**.

 Для обращения к диалоговому окну **О программе** выберите команду **О программе** в меню **Справка**.

 Дополнительная информация об этой утилите содержится в *«Руководстве администратора системы FactorySuite»*.

Совместимость с ODBC

Программа SQL Access Manager является приложением, совместимым с ODBC, и поэтому обеспечивает связь с любой СУБД, для которой имеется драйвер ODBC. Перед тем, как использовать драйвер ODBC, его необходимо сконфигурировать программой Microsoft ODBC Administrator для установки связей между приложением ODBC и базой данных.

➤ Как сконфигурировать драйвер ODBC:

1. Запустите программу Microsoft ODBC Administrator.
2. Выберите драйвер или источник данных и нажмите на **Add New Name**, **Set Default** или **Configure**. Появится диалоговое окно **ODBC Driver Setup**.

Параметр	Описание
Data Source Name	Имя, определяемое пользователем и идентифицирующее источник данных.
Description	Определяемое пользователем описание источника данных.
Database Directory	Каталог, содержащий файлы базы данных. Если каталог не определен, берется текущий.  Введите любую другую информацию, которая требуется для настройки выбранного драйвера.

3. Нажмите **ОК**.
 Драйвер записывает значение каждого поля в файл ODBC.INI. Эти значения используются по умолчанию для соединения с источником данных. Значения, действующие по умолчанию, могут быть изменены путем редактирования полей источника данных. Для любого атрибута, который не поддерживается в диалоговом окне ODBC Driver Setup, можно ввести соответствующие записи вручную в раздел источника данных файла ODBC.INI.

Требования к соединению

Строка соединения, используемая функцией **SQLConnect()** в версии 4.x программы InTouch SQL Access Manager, вводится как условие "DRV=database_dll".

Поскольку версия 5.6 (и более поздние) является совместимой с ODBC, поле database_dll в условии "DRV=database_dll" должно иметь точное имя источника данных, которое вы установили при помощи программы ODBC Administrator Program (см. выше). Вместо "DRV" можно использовать "DSN".

Г Л А В А 2

Настройка и соединение баз данных

Базы данных, описываемые в этой главе, прошли испытания и в настоящее время поддерживаются. Требования к каждой базе данных уникальны и специфичны. Эта глава содержит отдельные разделы по каждой базе данных, где описывается, как сконфигурировать конкретную базу данных для соединения с программой SQL Access Manager.

Содержание

- Работа с Oracle 6
- Работа с Oracle 7.2
- Работа с Sybase или Microsoft SQL Server
- Работа с dBASE
- Работа с Microsoft Access
- Работа с Paradox
- Значения типов данных для поддерживаемых СУБД

Работа с Oracle 6

➤ Как обеспечить соединение с Oracle 6:

1. Настройте модуль клиента базы данных в Windows.
2. Запустите SQL*Net TSR и программу NETINIT.EXE.
3. Установите соединение с Oracle® путем выполнения функции **SQLConnect()** в действующем сценарии InTouch.

☞ За дополнительной информацией о функции **SQLConnect()** обращайтесь к главе 4, "Использование функций SQL".

Настройка клиента

Для доступа к базе данных Oracle необходимо установить на локальном компьютере продукт Oracle's SQL*Net, состоящий из SQL*Net TSR и программы NETINIT.EXE. Чтобы обеспечить соединение с сервером базы данных Oracle, обе программы — SQL*Net TSR и NETINIT.EXE — должны выполняться.

Запуск SQL*Net TSR и NETINIT.EXE

Программа SQL*Net TSR, предназначенная для использования в сети, должны загружаться из DOS перед запуском Windows. Выполнение программы NETINIT.EXE необходимо для инициализации соединения с сервером Oracle. Для автоматического запуска NETINIT.EXE при загрузке Windows переместите ее значок в группу программ Windows "Автозагрузка".

Формат функции SQLConnect()

Функция **SQLConnect()** необходима для соединения с базой данных Oracle. Формат строки соединения функции **SQLConnect** следующий:

```
SQLConnect(ConnectionId, "<attribute>=<value>;<attribute>=<value>;...");
```

Далее описываются атрибуты, используемые Oracle. Они должны быть определены в следующем порядке:

Атрибут	Значение
DSN	Имя источника данных, такое же как в Microsoft ODBC Administrator или
DRV	Это значение используется для совместимости с SQL Access Manager, если имя источника данных (DSN) не представлено в строке соединения. QELIB изменяет его для имени источника данных.
UID	Имя пользователя.
PWD	Пароль.
SRVR	Строка соединения SQL*Net, задающая компьютер сервера и базу данных, к которым требуется доступ.

Пример:

```
SQLConnect(ConnectionString, "DSN=Oracle_Data;UID=SCOTT;PWD=TIGER;  
SRVR=B:MKTG_SRV");
```

Где: **B:** определяет префикс уровня соединения NetBIOS (в данном примере **SRVR=B:MKTG_SRV**) или

T: определяет префикс TCP/IP или

P: определяет префикс Named Pipes

MKTG_SRV — имя сервера базы данных Oracle.

Примечание. Некоторую сложность для определения представляет значение атрибута SRVR. Требуемая информация зависит от драйвера SQL*Net. Ее можно найти в документации по SQL*Net (или в файле CONFIG.ORA, в строке "remote=").

Поддерживаемые типы данных

Программа SQL Access Manager поддерживает два типа данных Oracle 6.

Символьный тип данных использует переменную длину символьных строк.

Тэги InTouch типа Message требуют данные символьного типа. Если длина не определена, то по умолчанию она равна одному (1) символу. Oracle

поддерживает символьные поля с максимальной длиной в 255 символа.

Однако тэги InTouch типа Message имеют ограничение в 131 символ. Если текстовая переменная включает в себя больше символов, чем допускается для поля базы данных, то строка будет усечена перед сохранением в базе.

Числовые типы данных представлены тэгами InTouch типа Integer и Real.

Если длина не определена, то значение будет задано с плавающей точкой с

точностью до 38 знаков. Если длина определена, то используется формат

Width.Decimal. Значение Width определяет максимальное количество знаков

столбца. Значение Decimal определяет количество знаков справа от точки.

Определение длины поля для этого типа данных не требуется.

Работа с Oracle 7.2

➤ Как обеспечить соединение с Oracle 7.2:

1. Установите модель Oracle Standard Client на узле InTouch.
2. Запустите программу SQL_Net Easy Configuration, чтобы определить Database Alias (псевдоним базы данных) для строки соединения SQL.
3. Задайте имя источника данных.
4. Выполните соединение с Oracle® с помощью функции **SQLConnect()** в действующем сценарии InTouch.

☞ За дополнительной информацией о функции **SQLConnect()** обращайтесь к главе 4, "Использование функций SQL".

Настройка клиента

➤ Как запустить программу установки Oracle 7:

1. В окне Oracle7 Workgroup Server выберите тип установки **Standard Client** и нажмите **OK**.
2. Выберите **Application User Setup Client** и нажмите кнопку **OK**.
3. В окне **Database Connection Setup** введите базовое имя узла, где установлен Oracle7 Workgroup Server. Например: WWServer.
4. Нажмите кнопку **OK**.

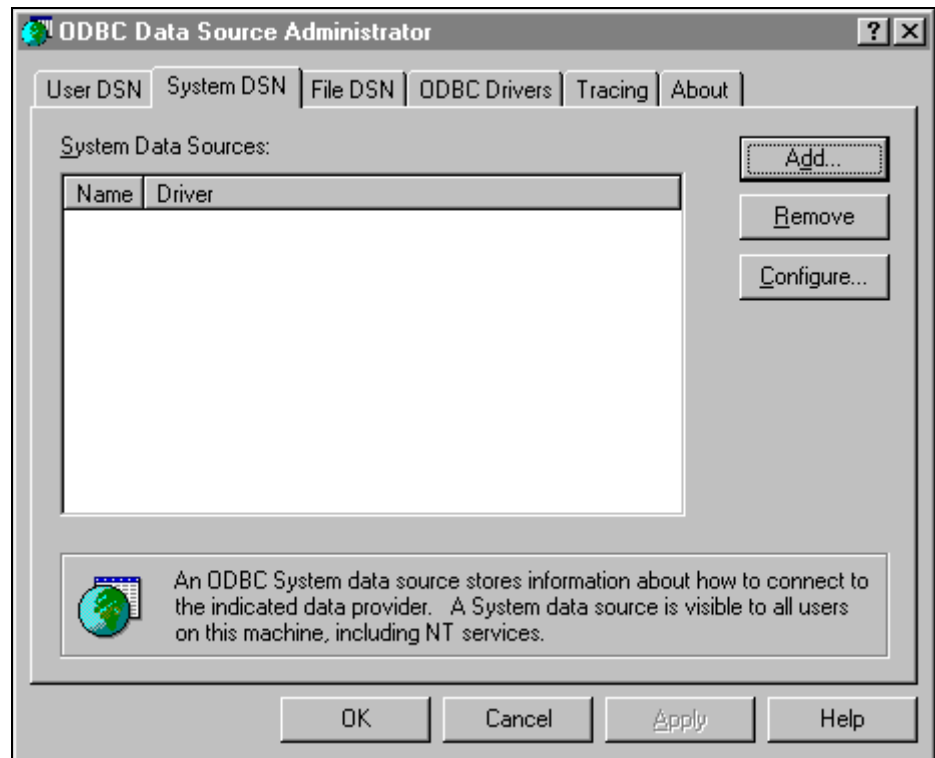
Настройка SQL_Net

1. На панели задач Windows нажмите кнопку **Пуск**. Выберите **Программы, Oracle**, затем нажмите на пункт **SQL_Net Easy Configuration**.
2. По умолчанию псевдоним сервера будет начинаться с **wgs_ServerName_orcl**. Но его можно изменить.
 - ☞ Псевдоним базы данных используется функцией **SQLConnect()** в приложении InTouch.
3. Для изменения псевдонима базы данных выберите имя сервера и нажмите **OK**.
4. Нажмите **Modify Database Alias Select Network protocol**. Именем компьютера, на котором работает сервер Oracle, является Named Pipe Server. Работа программы SQL_Net на этом завершена.
 - ☞ Запишите псевдоним базы данных — это пригодится в дальнейшем.

Настройка имени источника данных

➤ Как настроить драйверы ODBC на узле клиента:

1. На панели задач нажмите кнопку **Пуск**. Выберите **Настройка**, затем **Панель управления**. Откроется панель управления Windows.
2. Дважды нажмите значок **ODBC**. Появится диалоговое окно **ODBC Data Source Administrator**.
3. Откройте вкладку **System DSN**:



4. Нажмите кнопку **Add**. Появится диалоговое окно **Create New Data Source**.
5. Выберите драйвер Oracle7 ODBC и нажмите кнопку **Finish**. Появится диалоговое окно **ODBC Oracle Driver Setup**.
6. В окне **Data Source Name** введите имя сервера Oracle.
7. Нажмите кнопку **Advanced**. Используйте настройку по умолчанию в диалоговом окне **ODBC Oracle Advanced Driver Setup**. Нажмите кнопку **Close**. Снова появится диалоговое окно **ODBC Data Source Administrator**.
8. Нажмите **OK**.

Формат функции SQLConnect()

Функция **SQLConnect()** используется для соединения с базами данных Oracle.

Формат строки соединения **SQLConnect()** следующий:

```
SQLConnect(ConnectionId, "<attribute>=<value>;<attribute>=<value>;...");
```

Далее описываются атрибуты, используемые Oracle. Они должны быть определены в следующем порядке:

Атрибут	Значение
DSN	Имя источника данных, заданное в программе Microsoft ODBC Administrator.
UID	Имя пользователя.
PWD	Пароль.
SRVR	Псевдоним базы данных SQL_NET.

Пример:

```
SQLConnect(ConnectionId, "DSN=Oracle;UID=SCOTT;PWD=TIGER;  
SRVR=wsg_wwServer_orcl");
```

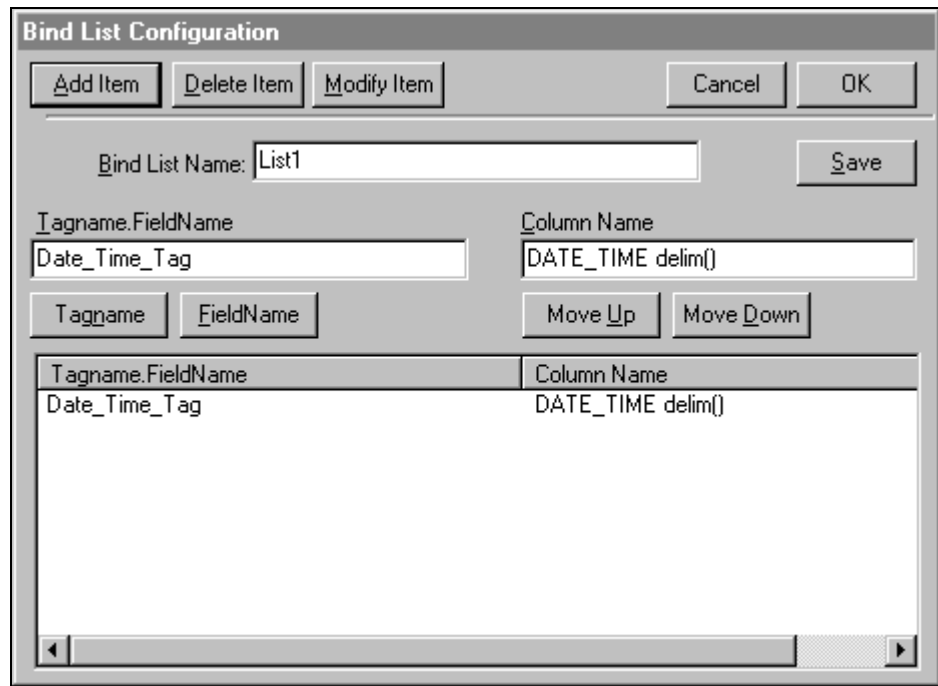
Примечание. Если псевдоним базы данных был изменен, т. е. он отличается от формата по умолчанию (wsg_ИмяСервера_orcl), используйте новый псевдоним, заданный в SQL_NET Easy Configuration, SRVR=WWServer.

Запись даты и времени в поле даты Oracle

Чтобы записать дату и время в поле даты Oracle date field, нужно сконфигурировать список связей, используя функцию `delim`.

➤ **Как записать дату и время в поле даты Oracle:**

1. В программе Application Explorer под веткой **SQL Access Manager** нажмите дважды на **Bind List**. Появится диалоговое окно **Bind List Configuration**:



2. В окне **Tagname.FieldName** напишите тэг, который вы хотите использовать.
3. В окне **Column Name** напишите функцию **DATE_TIME delim()**.

Примечание. В окне ввода **Column Name** пространство должно быть использовано функцией **delim()**. Например, `DATE_TIME`. Должен быть введен пробел между `DATE` и `TIME`.

4. В приложении InTouch создайте QuickScript для подготовки данных ввода из настоящих дат и времени. Например:

```
DATE_TIME_TAG = "TO_DATE('" + $DateString + "' +  
StringMid($TimeString,1,8) + "','mm/dd/yy hh24:mi:ss')";
```

☞ Имя `Date_Time_Tag` будет отображено так, как в следующем примере:

```
TO_DATE('08/22/97 23:32:18' , 'mm/dd/yy hh24:mi:ss')
```

Работа с Sybase или Microsoft SQL Server

➤ Как обеспечить соединение с Sybase или Microsoft SQL Server:

1. Настройте модуль клиента базы данных в Windows.
2. Используя функцию **SQLConnect()** в сценарии InTouch, установите соединение с Sybase® или Microsoft® SQL Server.

☞ За дополнительной информацией о функции **SQLConnect()** обращайтесь к главе 4, "Использование функций SQL".

Настройка клиента

➤ Как установить соединение с базой данных:

Для соединения с базой данных в каталоге WINDOWS/SYSTEM должны присутствовать следующие DLL-модули:

- DBNMP3.DLL
- W3DBLIB.DLL

Примечание. Если для соединения с базой данных приложение использует Winsock (TCP/IP), то DBNMP3.DLL не требуется.

Формат функции SQLConnect()

Функция **SQLConnect()** используется для соединения с Sybase или Microsoft SQL Server. Выполнение этой функции регистрирует вас на сервере базы данных и открывает соединение для выполнения других функций SQL.

Формат строки соединения **SQLConnect()** должен быть следующий:

```
SQLConnect(ConnectionId, "<attribute>=<value>;<attribute>=<value>;...");
```

Далее описываются атрибуты, используемые Sybase или Microsoft SQL Server. Они должны быть определены в следующем порядке:

Атрибут	Значение
DSN	Имя источника данных, такое же как в Microsoft ODBC Administrator или
DRV	Это значение используется для совместимости с SQL Access Manager, если имя источника данных (DSN) не представлено в строке соединения. QELIB изменяет его в соответствии с именем источника данных.
UID	Вход в систему ID, с учетом регистра.
PWD	Пароль, с учетом регистра.
SRVR	Имя сервера компьютера с таблицами базы данных, к которым требуется доступ.
DB	Имя базы данных, к которой требуется доступ.

Пример:

```
SQLConnect(ConnectionId, "DSN=SQL_Data;UID=OPERATOR;PWD=XYZZ");
```

Поддерживаемые типы данных

Программа SQL Access Manager поддерживает три типа данных для Sybase и Microsoft SQL Server. Символьный тип данных (char) содержит фиксированную длину символьной строки. Тэги InTouch типа Message требуют символьного типа данных. Длина поля должна быть определена. Базы данных Sybase и Microsoft SQL Server поддерживают символьные поля максимальной длиной 255 символов. Однако тэги типа Message имеют ограничение в 131 символ. Если текстовая переменная включает в себя большее количество символов чем позволено для поля базы данных, строка будет усечена перед записью в базу данных.

Целочисленный тип данных (int) используется целочисленными (integer) тэгами InTouch. Если длина поля не определена, то она устанавливается такой, какая принята базой данных по умолчанию. Если же длина определена, она будет в форме Width. Width определяет максимальное количество знаков ширины столбца.

Тип данных с плавающей точкой (float) используется действительными (real) тэгами InTouch. Длина поля фиксируется базой данных. Определять длину поля для этого типа данных не требуется.

Примечание. Типы данных (char, int, float) вводятся с учетом регистра и при использовании сервера Sybase должны вводиться в нижнем регистре.

Sybase и Microsoft SQL Server поддерживают только одну активную инструкцию на каждый ConnectionId. При выполнении функции **SQLSelect()** или **SQLNext()** данные могут просматриваться, но не могут быть вставлены, удалены или изменены. Чтобы это можно было сделать, выполните **SQLSelect()** в ConnectionId отделенном от функций **SQLInsert()**, **SQLUpdate()** и **SQLDelete()**. Имена столбцов и таблиц также вводятся с учетом регистра. Имена таблиц должны быть уточнены у владельца таблиц. Например, dbo.MyTable)

Работа с dBASE

Для соединения с dBASE® необходимо выполнить функцию **SQLConnect()** в сценарии InTouch.

Формат функции SQLConnect()

Функция **SQLConnect()** используется для соединения с базой данных dBASE. Выполнение этой функции регистрирует вас на сервере базы данных и открывает соединение для выполнения других функций SQL. Формат строки соединения **SQLConnect()** должен быть следующий:

```
SQLConnect(ConnectionId, "<attribute>=<value>;<attribute>=<value>;...");
```

Далее описываются атрибуты, используемые dBASE. Они должны быть определены в следующем порядке:

Атрибут	Значение
DSN	Имя источника данных, такое же как в Microsoft ODBC Administrator или
DRV	Используется для совместимости с SQL Access Manager, если имя источника данных (DSN) не представлено в строке соединения. QELIB изменяет его в соответствии с именем источника данных.
CS	Атрибут, сообщающий драйверу, в каком наборе символов хранятся данные: IBM PC или ANSI. По умолчанию используется набор символов IBMPC.
DB	Указывает каталог, где хранятся файлы dBASE. Если атрибут не задан, используется текущий рабочий каталог.
FOC	Максимальное число кэшируемых открытых файлов. По умолчанию имеет значение 0. Драйвер dBASE использует кэширование открытых файлов для повышения производительности. Если FOC больше 0, самые последние открывавшиеся файлы остаются открытыми. Когда другое приложение попытается открыть эти файлы в монопольном режиме, будет выдано сообщение о том, что файлы используются.
LCK	Определяет уровень блокировки записей для файла базы данных. Допускаются значения FILE, RECORD (по умолчанию) или NONE (например, LCK=FILE).
CSZ	Число блоков по 64К, используемых драйвером для кэширования записей базы данных. Чем больше блоков, тем выше производительность. По умолчанию используется 4 блока. Максимально возможное количество блоков зависит от объема системной памяти. При просмотре записей в обратном порядке обновления, сделанные другими пользователями, не будут видны, пока не будет повторно выполнена функция SQLSelect() .

Атрибут	Значение
USF	Определяет, когда драйвер должен обновлять элементы каталога DOS. Если этот атрибут равен 1, драйвер обновляет элементы каталога после каждого COMMIT. Это приводит к снижению производительности. По умолчанию действует значение 0. То есть драйвер обновляет каталог при закрытии файла. В этом случае зависание системы до закрытия файла приводит к потере последних вставленных записей.
MS	Этот атрибут используется для обратной совместимости со старыми программами Q+E. Значение MS=0 позволит драйверу понимать выражения SQL, встречающиеся в ранних версиях драйверов Q+E. По умолчанию имеет значение 1.
LCOMP	Определяет метод блокировки: совместимый с dBASE или с Q+E. Значение LCOMP=DBASE заставляет драйвер использовать dBASE-совместимую блокировку; а значение LCOMP=Q+E предписывает использование Q+E-совместимой блокировки. По умолчанию — dBASE. Q+E-совместимая блокировка нужна для совместимости с ранними версиями Q+E.
COMP	Обеспечивает обратную совместимость с ранними версиями программ Q+E. Значение COMP=DBASE используется для обратной совместимости; COMP=ANSI — для переносимости. По умолчанию используется ANSI.

Пример:

```
SQLConnect(ConnectionString, "DSN=DBASE_FILES;CS=ANSI;DB=C:\ODBC\EMP;
LCK=NONE");
```

Поддерживаемые типы данных

Программа SQL Access Manager поддерживает три типа данных для dBASE. Символьный тип данных (char) содержит фиксированную длину символьной строки и используется тэгами InTouch типа Message. Длина поля должна быть определена. dBASE поддерживает символьные поля максимальной длиной 254 символов. Однако тэги типа Message имеют ограничение в 131 символ. Если текстовая переменная включает в себя большее количество символов чем позволено для поля базы данных, строка будет усечена перед записью в базу данных.

Числовые и с плавающей точкой типы данных представляют теги InTouch типа Integer или Real. Длина должны быть определена. Формат должен быть Width.Decimal. Значение Width определяет максимальное количество знаков в столбце. Значение Decimal задает количество знаков справа от десятичной точки. dBASE может хранить аналоговые значения с точностью до 19 знаков.

Работа с Microsoft Access

Для соединения с базой Microsoft Access необходимо выполнить функцию **SQLConnect()** в сценарии InTouch.

Формат функции SQLConnect()

Функция **SQLConnect()** используется для соединения с базой данных Microsoft Access. Выполнение этой функции регистрирует вас на сервере базы данных и открывает соединение для выполнения других функций SQL.

Формат строки соединения **SQLConnect()** должен быть следующий:

```
SQLConnect(ConnectionId, "<attribute>=<value>;<attribute>=<value>;...");
```

Ниже описаны атрибуты, используемые в Microsoft Access. Они приведены в том порядке, в котором должны задаваться:

Атрибут	Значение
DSN	Имя источника данных, такое же как в Microsoft ODBC Administrator или
DRV	Это значение используется для совместимости с SQL Access Manager, если имя источника данных (DSN) не представлено в строке соединения. QELIB изменяет его в соответствии с именем источника данных.

Пример:

```
SQLConnect(ConnectionId, "DSN=MSACC");
```

Поддерживаемые типы данных

SQL Access Manager поддерживает пять типов данных Microsoft Access. Допустимые типы данных зависят от версии используемого драйвера ODBC. Текстовый тип данных (text) содержит фиксированную длину символьной строки и используется тэгами InTouch типа Message. Длина поля должна быть определена. Microsoft Access поддерживает текстовые поля длиной до 255 символов. Однако тэги типа Message имеют ограничение в 131 символ. Если текстовая переменная включает в себя большее количество символов чем позволено для поля базы данных, строка будет усечена перед записью в базу данных. Драйвер Microsoft Access ODBC поддерживает до 17 символов в заголовке столбца. Максимальное количество столбцов при использовании **SQLSetStatement(Select Col1, Col2, ...)** равно 40.

Для программ, использующих драйвер "SIMBA.DLL" для Microsoft Access версии 1.1 (включая Excel 5.0, Microsoft Word 6.0 и т. д.), допустимыми числовыми типами данных являются "short" или "long" для тэгов типа Integer и "singlefloat" или "doublefloat" для тэгов типа Real.

Для программ, использующих драйвер "ODBCJT16.DLL" для Microsoft Access версии 2.0, допустимыми числовыми типами данных являются "short" или "long" для тэгов типа Integer и "number with a modifier of Double or Single" для тэгов типа Real. Длина вычисляется автоматически в зависимости от типа данных и не должна указываться в определении шаблона таблицы (Table Template).

Работа с Paradox

Для соединения с Paradox® необходимо выполнить функцию **SQLConnect()** в сценарии InTouch.

Формат функции SQLConnect()

Функция **SQLConnect()** используется для соединения с базой данных Paradox. Выполнение этой функции регистрирует вас на сервере базы данных и открывает соединение для выполнения других функций SQL. Формат строки соединения **SQLConnect()** должен быть следующий:

```
SQLConnect(ConnectionId, "<атрибут>=<значение>;<атрибут>=<значение>;..." );
```

Далее описываются атрибуты, используемые Paradox. Они должны быть определены в следующем порядке:

Атрибут	Значение
DSN	Имя источника данных, такое же как в Microsoft ODBC Administrator или
или	
DRV	Это значение используется для совместимости с SQL Access Manager, если имя источника данных (DSN) не представлено в строке соединения. QELIB изменяет его в соответствии с именем источника данных.

Пример:

```
SQLConnect(ConnectionId, "DSN=PARADOX_FILE");
```

Поддерживаемые типы данных

InTouch SQL поддерживает три типа данных Paradox. Буквенноцифровой тип данных содержит фиксированную длину символьной строки и используется тэгами InTouch типа Message. Длина поля должна быть определена. Paradox поддерживает буквенноцифровые поля длиной до 255 символов. Однако тэги типа Message имеют ограничение в 131 символ. Если текстовая переменная включает в себя большее количество символов чем позволено для поля базы данных, строка будет усечена перед записью в базу данных.

Допустимыми числовыми типами данных являются "number" для тэгов типа Integer, Real или Discrete и "short" (целые значения между -32,767 и 32,767) для тэгов типа Integer или Discrete. Длина поля не должны указываться.

Значения типов данных для поддерживаемых СУБД

Oracle

Тип данных	Длина	По умолчанию	Диапазон	Тип тэга
Char	255 символов	1 символ		Текстовый
number	38 знаков	38 знаков		Целый

Sybase или Microsoft SQL Server

Тип данных	Длина	По умолч.	Диапазон	Тип тэга
Char	255 символов			Текстовый
int			от -2,147,483,647 до 2,147,483,647	Целый
float	15 знаков		от $1.7E^{-294}$ до $1.7E^{+308}$	Действительный

dBASE

Тип данных	Длина	По умолч.	Диапазон	Тип тэга
char	254 символов			Текстовый
numeric or	19 знаков			Целый или действительный
float				

Microsoft Access 1.1 (Simba.DLL)

Тип данных	Длина	По умолч.	Диапазон	Тип тэга
text	254 символа			Текстовый
int				Целый

Microsoft Access 2.0 (ODBCJT16.DLL), 7.0 (ODBCJT32.DLL)

Тип данных	Длина	По умолч.	Диапазон	Тип тэга
text	254 символа			Текстовый
number				Целый
number				Действительный

Paradox

Тип данных	Длина	По умолч.	Диапазон	Тип тэга
alphanumeric number	255 символов			Текстовый, целый, действитель- ный или дискретный
short			от -32,767 до 32,767	Целый или дискретный

Г Л А В А 3

Настройка SQL Access Manager

Утилита SQL Access Manager создает списки привязок (Bind Lists) и шаблоны таблиц (Table Templates). Список привязок ассоциирует колонки базы данных с тэгами из словаря данных InTouch. Шаблон таблиц определяет структуру и формат новой таблицы в базе данных.

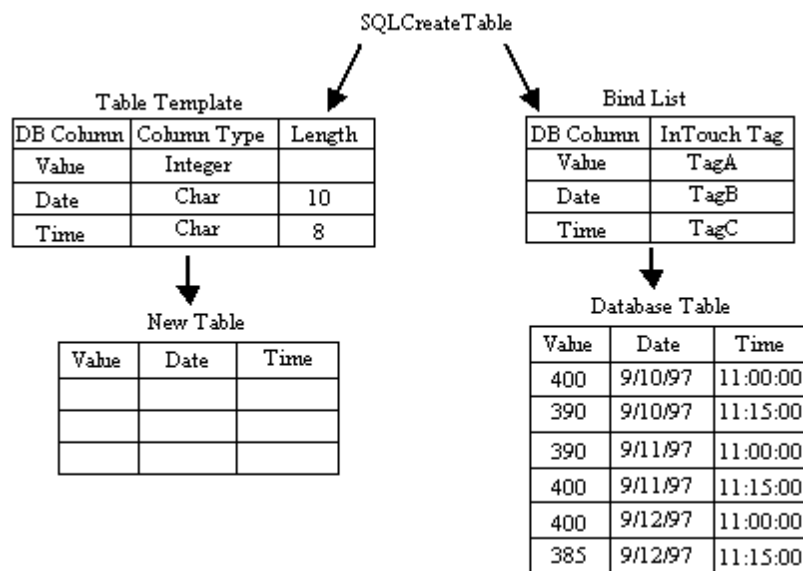
Содержание

- Обзор программы SQL Access Manager
- Использование специальных разделителей
- Настройка шаблонов таблиц
- Файл SQL.DEF

Обзор программы SQL Access Manager

Когда приложение InTouch выполняет команду **SQLCreateTable()**, аргумент шаблона таблиц определяет структуру нового файла базы данных.

При выполнении команды **SQLInsert()**, **SQLSelect()** или **SQLUpdate()** аргумент списка привязок определяет, какие тэги InTouch будут использоваться и какие колонки базы данных будут с ними связаны.



Настройка списка привязок

Список привязок определяет ассоциации столбцов базы данных с тэгами из словаря данных InTouch.

➤ **Как создать новый список привязок:**

1. В меню **Сервис** выберите **Диспетчер доступа SQL** и нажмите на **Список привязок**. Или в Менеджере приложения (Application Explorer) под веткой **Диспетчер доступа SQL** дважды нажмите на **Список привязок**.

☞ При выборе этой команды в первый раз появится диалоговое окно **Выберите список привязок**.

Выберите список привязок	
Необходимо создать список привязок	
Отмена	Создать

2. Нажмите кнопку **Создать**.

3. Появится диалоговое окно **Настройка списка привязок**:

- ☞ Если нажать правой кнопкой мыши на любое из текстовых полей ввода данных, откроется меню с командами, которые применимы к выделенному тексту.

4. В поле **Имя списка привязок** введите имя списка.

- ☞ Имя списка привязок может быть до 32 символов в длину. Новый список привязок связывает колонки базы данных с тэгами InTouch. Например, если создается список служащих, нужно ввести имя списка привязок, которое связывает данные о служащих.

Примечание. Функции **SQLInsert()**, **SQLSelect()** и **SQLUpdate()** используют параметр списка привязок. Кроме того, при использовании функции **SQLExecute()** имеет значение порядок тэгов, так как эта функция записывает тэги в определенной последовательности.

5. В поле **Тэг или поле тэга** введите имя **.поля** тэга InTouch.

- ☞ Словарь тэгов ассоциирует **tagname.field** с именем колонки в базе данных. Если этот тэг в данный момент не определен в словаре тэгов, нажмите на него дважды для открытия диалогового окна **Словарь тэгов**, чтобы определить новый тэг.

6. Нажмите кнопку **Имя тэга** для выбора существующего тэга. Появится окно браузера тэгов.

- ☞ Браузер тэгов будет отображать тэги из выбранного в данный момент источника тэгов. Чтобы выбрать тэг, нажмите на него дважды и затем нажмите на **ОК**. Для выбора **.поля** тэга нажмите на стрелку **поле** и выберите **.поле**, которое хотите использовать. Затем нажмите **ОК**.

📖 Дополнительную информацию о браузере тэгов можно найти в «Руководстве пользователя InTouch».

7. Нажмите на кнопку **Поле** для добавления **.поля** к тэгу. Появится диалоговое окно **Выберите имя поля**.

8. Нажмите на нужное **.поле**. Диалоговое окно закроется, а выбранное **.поле** автоматически добавится к тэгу в поле **Тэг или поле тэга**.
 Подробная информация о **.полях** тэгов содержится в главе 4 «Руководства пользователя InTouch».
9. В поле **Имя колонки** введите имя колонки базы данных.
 Имя колонки может иметь длину до 30 символов. Это имя напрямую связано с именем колонки базы данных. Имя колонки нужно заключать в квадратные скобки в списке привязок, а также при использовании в сценарии, если имя содержит пробел. Например:

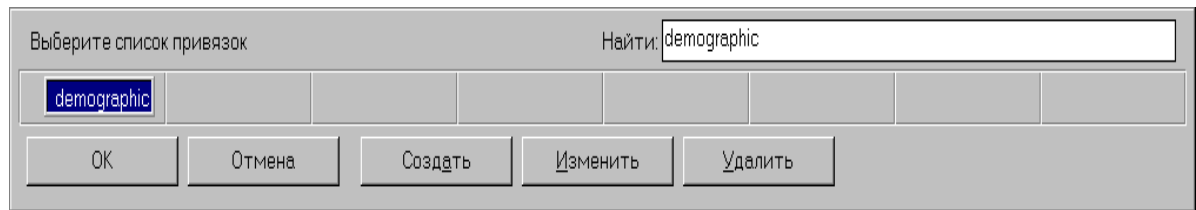
```
WHERE Expr= "[Pipe Flow] = " text (tagname, "#");
```

Также могут использоваться специальные разделители для ассоциации имени колонки с базой данных.

 For Дополнительную информацию о специальных разделителях можно найти в разделе «Использование специальных разделителей».
10. Нажмите **Сдвиг вверх** для перемещения выбранного тэга в списке на один уровень вверх.
11. Нажмите **Сдвиг вниз** для перемещения выбранного тэга в списке на один уровень вниз.
12. Нажмите **Добавить элемент** для добавления новых **тэга или поля тэга и имени колонки** в список привязок.
13. Нажмите **Удалить элемент** для удаления выбранных **тэга или поля тэга и имени колонки** из списка привязок.
14. Нажмите **Изменить элемент** для изменения выбранных **тэга или поля тэга и имени колонки** в списке привязок.
15. Нажмите **ОК** для сохранения новой конфигурации списка привязок и для закрытия диалогового окна.
 Можно нажать на кнопку **Сохранить** для сохранения новых данных без закрытия диалогового окна.

➤ **Как изменить список привязок:**

1. В меню **Сервис** нажмите на **Диспетчер доступа SQL**, затем на **Список привязок**. Или в Менеджере приложения (Application Explorer) под веткой **Диспетчер доступа SQL** дважды нажмите на **Список привязок**.
2. Появится диалоговое окно **Выберите список привязок**:

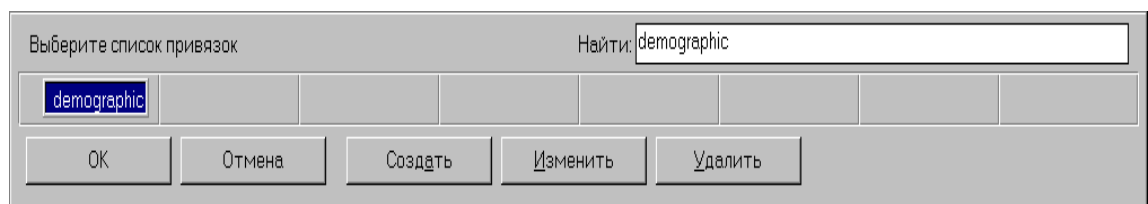


3. Выберите имя списка привязок, которое нужно изменить, и нажмите на кнопку **Изменить**. Появится диалоговое окно **Настройка списка привязок**.
4. Измените нужные пункты.
5. Нажмите **ОК** для сохранения изменений и закрытия диалогового окна.

☞ Дополнительную информацию о настройке списка привязок смотрите выше в подразделе «Как создать новый список привязок».

➤ **Как удалить список привязок:**

1. В меню **Сервис** нажмите на **Диспетчер доступа SQL**, затем на **Список привязок**. Или в Менеджере приложения (Application Explorer) под веткой **Диспетчер доступа SQL** дважды нажмите на **Список привязок**.
2. Появится диалоговое окно **Выберите список привязок**:



3. Выберите имя списка привязок, которое хотите удалить.
4. Нажмите кнопку **Удалить**. Появится окно сообщения, в котором для подтверждения удаления выбранного списка привязок нажмите на кнопку **Да** или на кнопку **Нет** — для отмены команды. Вновь откроется диалоговое окно **Настройка списка привязок**.
5. Нажмите **ОК** для закрытия диалогового окна.

Использование специальных разделителей

Функции `SQLInsert()` и `SQLUpdate()` используют по умолчанию формат, включающий в себя текстовые строки с одинарными кавычками. Некоторые базы данных SQL ожидают получения текстовых строк с другим типом разделителя. Например, Oracle ожидает строки данных, заключенные в скобки. При этом функция `Delim()` должна использоваться следующим образом:

В диалоговом окне **Настройка списка привязок**, в поле **Имя колонки**, после имени колонки введите ключевое слово "delim" (без учета регистра). За словом "delim" должны вводиться:

- левая круглая скобка
- левый разделитель
- запятая
- правый разделитель
- правая круглая скобка

Пример: `datestring delim ('')`

Чтобы использовать одинаковый правый и левый разделитель, обозначьте его в круглых скобках без запятой.

Пример: `datestring delim ( ' ' )`

В следующем примере используются разные правый и левый разделители. Обратите внимание где `date delim ('')` вводится в поле **Имя колонки**:

Тэг.поле	Имя поля
\$DateTime	date_time delim (TO_DATE (','))

☞ Подробная информация о записи даты и времени в поле даты Oracle содержится в главе 2 в разделе ["Работа с Oracle"](#).

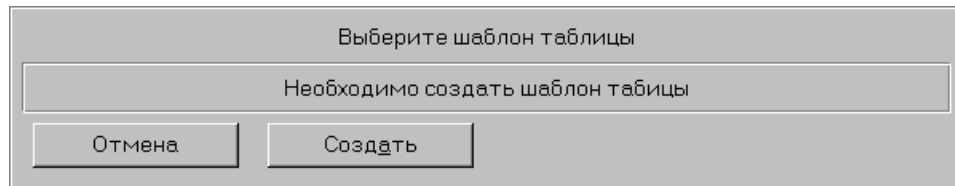
Настройка шаблонов таблиц

Эта команда создает шаблон таблиц, определяющий структуру и формат новой таблицы в базе данных.

➤ Как создать новый шаблон таблицы:

1. В меню **Сервис** нажмите на **Диспетчер доступа SQL**, затем на **Шаблон таблицы**. Или в Менеджере приложения (Application Explorer) под веткой **Диспетчер доступа SQL** дважды нажмите на **Шаблон таблицы**.

☞ При выборе этой команды впервые появится диалоговое окно **Выберите шаблон таблицы**:



2. Нажмите кнопку **Создать**.
3. Появится диалоговое окно **Настройка шаблона таблицы**:

Диалоговое окно с заголовком 'Настройка шаблона таблицы'. В верхней панели находятся кнопки: 'Добавить элемент', 'Удалить элемент', 'Изменить элемент', 'Отмена' и 'OK'. Ниже находится поле 'Имя шаблона таблицы:' со значением 'Template1'. В центре расположены поля для настройки колонки: 'Имя колонки' (EmployeeID), 'Тип колонки' (Decimal) и 'Длина' (7.2). Ниже этих полей находится группа 'Тип индекса' с тремя радиокнопками: 'Уникальное', 'Не-уникальное' и 'Нет' (выбрано). Также есть флажок 'Допускать нулевые вхождения', который отмечен. В нижней части окна находится таблица с данными:

Имя поля	Тип поля	Длина	Допуска...	Тип индекса
EmployeeID	Decimal	7.2	Null	None

☞ Если нажать правой кнопкой мыши на любое из текстовых полей ввода данных, откроется меню с командами, которые применимы к выделенному тексту.

4. В поле **Имя шаблона таблицы** введите имя шаблона таблицы.

☞ Имя шаблона таблицы может иметь длину до 32 символов. Это имя используется для идентификации структуры базы данных в функции **SQLCreateTable()**.

5. В поле **Имя колонки** введите имя колонки для шаблона таблицы.

- ☞ Имя колонки может быть до 30 символов в длину.
- 6. В поле **Тип колонки** введите тип данных для колонки.
 - ☞ Выбор типа данных зависит от используемой базы данных.
 - ☞ Дополнительная информация о типах данных для различных СУБД содержится в главе 2 «Поддерживаемые типы данных».
- 7. В поле **Тип индекса** выберите один из типов:
 - Уникальное** Каждое значение колонки должно быть уникальным.
 - Неуникальное** Каждое значение колонки может быть неуникальным.
 - Нет** Без индекса.
 - ☞ При выполнении функции **SQLCreateTable()** автоматически создается индексный файл.
- 8. Выберите поле **Допускать нулевые вхождения** для ввода в эту колонку нулевых данных.
 - ☞ InTouch не поддерживает нулевые данные. При вводе данных, если значение для тэга не указано, то оно будет следующим:
 - Дискретное 0
 - Целое 0
 - Текстовое Строки без символов
 - При выборке данных нулевые значения будут преобразованы в соответствии с типом данных, как показано выше.
- 9. Нажмите **Добавить элемент** для добавления нового имени колонки, типа колонки, длины и индексного типа в шаблон таблицы.
- 10. Нажмите **Удалить элемент** для удаления выбранных имени колонки, типа колонки, длины и индексного типа из списка шаблонов таблицы.
- 11. Нажмите **Изменить элемент** для изменения выбранных имени колонки, типа колонки, длины и индексного типа в списке шаблонов таблицы.
- 12. Нажмите **ОК**, чтобы сохранить новую конфигурацию шаблона таблиц и закрыть диалоговое окно.
 - ☞ Можно нажать на кнопку **Сохранить**, чтобы сохранить установленные данные без закрытия диалогового окна.

➤ **Как изменить шаблон таблиц:**

1. В меню **Сервис** нажмите на **Диспетчер доступа SQL**, затем на **Шаблон таблицы**. Или в Менеджере приложения (Application Explorer) под веткой **Диспетчер доступа SQL** дважды нажмите на **Шаблон таблицы**.
2. Появится диалоговое окно **Выберите шаблон таблицы**:

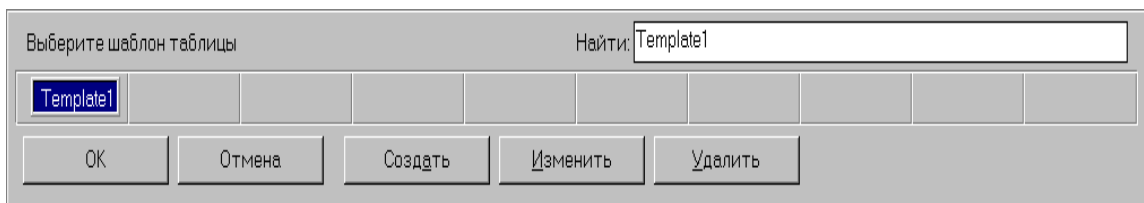


3. Выберите имя шаблона таблиц и нажмите кнопку **Изменить**. Появится диалоговое окно **Настройка шаблона таблицы**.
4. Измените нужные пункты.
5. Нажмите **ОК**, чтобы сохранить сделанные изменения и закрыть диалоговое окно.

☞ Дополнительную информацию о настройке шаблонов таблиц можно найти выше в подразделе «Как создать новый шаблон таблиц».

➤ **Как удалить шаблон таблиц:**

1. В меню **Сервис** нажмите на **Диспетчер доступа SQL**, затем на **Шаблон таблицы**. Или в Менеджере приложения (Application Explorer) под веткой **Диспетчер доступа SQL** дважды нажмите на **Шаблон таблицы**.
2. Появится диалоговое окно **Выберите шаблон таблицы**:



3. Выберите имя шаблона таблиц, которое нужно удалить.
4. Нажмите на кнопку **Удалить**. Появится окно сообщения, в котором для подтверждения удаления выбранного списка привязок нажмите на кнопку **Да** или на кнопку **Нет** — для отмены команды. Вновь откроется диалоговое окно **Настройка шаблона таблицы**.
5. Нажмите **ОК** для закрытия диалогового окна.

Файл SQL.DEF

Программа SQL Access Manager сохраняет информацию для списков привязок и шаблонов таблиц в файле "SQL.DEF" в формате разделенных запятыми переменных (.CSV). Этот файл можно просмотреть или изменить с помощью SQL Access Manager или любого текстового редактора, такого, как Notepad (Блокнот). Данные запишутся в файл в следующем виде:

:BindListName,*BindListName*

Tagname1.FieldName,ColumnName1

Tagname2.FieldName,ColumnName2

Tagname3.FieldName,ColumnName3

:TableTemplateName,*TableTemplateName*

ColumnName1,ColumnType,[ColumnLength],Null,Index

ColumnName2,ColumnType,[ColumnLength],Null,Index

ColumnName3,ColumnType,[ColumnLength],Null,Index

Г Л А В А 4

Использование функций SQL

Приложение InTouch использует функции SQL для работы с базами данных. Эти функции являются расширениями стандартных функций сценариев InTouch и могут использоваться в любом сценарии. Они позволяют выбирать, изменять, вставлять или удалять записи в используемых таблицах.

Содержание

- [Функции SQL](#)
- [Параметры SQL](#)
- [Использование функций SQL в сценариях InTouch](#)
- [Определение сложных запросов](#)

Функции SQL

Этот раздел содержит описание каждой функции SQL. Следует помнить о том, что все операции SQL выполняются синхронно. Управление не возвращается приложению InTouch, пока не завершена работа SQL (выполнение опроса, трендов и т. д. приостанавливается).

Все функции SQL, за исключением **SQLNumRows()**, возвращают *ResultCode* (код результата). Если *ResultCode* не равен нулю, это значит, что функция не отработала успешно, и следует перейти к другим операциям. *ResultCode* может использоваться функцией **SQLErrorMsg()**.

Общий формат функций SQL:

SQLFunction(Parameter1, Parameter2, ...)

 Подробная информация о функциях SQL и их использовании содержится в "Справочном руководстве InTouch".

Функция	Описание
SQLAppendStatement	Продолжает выполнение инструкции SQL, используя содержимое <i>SQLStatement</i>
SQLClearParam	Очищает значение конкретного параметра.
SQLClearStatement	Освобождает ресурсы, которые ассоциируются с инструкцией, определенной в <i>SQLHandle</i> .
SQLClearTable	Удаляет все записи в таблице базы данных, не удаляя самой таблицы.
SQLCommit	Определяет конец группы команд транзакции.
SQLConnect	Связывает приложение InTouch с базой данных, определенной в <i>ConnectString</i> .
SQLCreateTable	Создает таблицу в базе данных, используя параметры шаблона таблиц.
SQLDelete	Удаляет запись или записи.
SQLDisconnect	Отключает пользователя от базы данных.
SQLDropTable	Уничтожает таблицу.
SQLEnd	Используется после SQLSelect() для освобождения ресурсов, которые были заняты для хранения таблицы результатов.
SQLErrorMsg	Отыскивает сообщение о текстовой ошибке, которое ассоциируется с конкретным <i>ResultCode</i> . <i>ErrorMsg</i> является системным тэгом InTouch типа Message (длиной до 131 символа), который ассоциируется с <i>ResultCode</i> .
SQLExecute	Выполняет инструкцию SQL. Если инструкция является инструкцией выбора, то параметр <i>BindList</i> обозначает имя <i>СпискаСвязей</i> , используемое для связывания столбцов базы данных с тэгами. Если значение <i>BindList</i> равно нулю, связи тэгов не формируются.
SQLFirst	Выбирает первую запись в таблице результатов, созданную последней функцией SQLSelect() .
SQLGetRecord	Отыскивает запись по <i>RecordNumber</i> в выбранном в настоящий момент буфере.

Функция	Описание
SQLInsert	Вставляет новую запись в таблицу, используя значения тэгов в имеющемся <i>BindList</i> . Параметр <i>BindList</i> определяет, какой из тэгов InTouch используется и с каким столбцом базы данных он связан.
SQLInsertEnd	Освобождает инструкцию.
SQLInsertExecute	Выполняет уже готовую инструкцию.
SQLInsertPrepare	Создает и подготавливает инструкцию Insert для выполнения.
SQLLast	Выбирает последнюю запись в таблице результатов, которая создана последней функцией SQLSelect .
SQLLoadStatement	Читает инструкцию, которая находится в <i>FileName</i> . С этой точки зрения, данная инструкция подобна той, которая создана функцией SQLSetStatement() , и может быть добавлена при помощи SQLAppendStatement() или выполнена функцией SQLExecute() . В файле может быть только одна такая инструкция. Однако функция SQLAppendStatement() может использоваться для добавления к инструкции, если не было обращения к функциям SQLPrepareStatement() и SQLExecute() .
SQLManageDSN	Выполняет программу Microsoft ODBC Manager. Она может быть использована для добавления, удаления или изменения всех имен источников данных.
SQLNext	Выбирает следующую запись из таблицы результатов, которая создана последней функцией SQLSelect() .
SQLNumRows	Указывает, сколько строк обнаружил критерий, определенный в последней функции SQLSelect() .
SQLPrepareStatement	Подготавливает существующую инструкцию SQL для использования ее функцией SQLSetParam() . Инструкция может создаваться функцией SQLSetStatement() или SQLLoadStatement() .
SQLPrev	Выбирает предыдущие записи из таблицы результатов, которая создана последней функцией SQLSelect() .

Функция	Описание
SQLRollback	Команда SQLRollback() производит отмену, или "откат" последнего недавно выполненного набора транзакций. Набор транзакций – это набор команд, выполненных в промежутке между командами SQLTransact() и SQLCommit() или SQLRollback() . Набор транзакций обрабатывается как единая транзакция. После выполнения команды SQLTransact() все последующие операции не выполняются в базе данных, пока не будет выполнена команда SQLCommit() .
SQLSelect	Дает указания базе данных искать данные в таблице. После выполнения SQLSelect() в памяти создается временная таблица результатов, содержащая записи, которые могут быть просмотрены с помощью команд SQLFirst() , SQLLast() , SQLNext() , SQLPrev() .
SQLSetParamChar	Устанавливает значение конкретного параметра в конкретной строке. Эта функция может вызываться много раз перед выполнением, получением результата значение параметра, которое было установлено для связи со всеми переданными значениями.
SQLSetParamDate	Устанавливает значение конкретного параметра даты в конкретной строке.
SQLSetParamDateTime	Устанавливает значение конкретного параметра даты-времени в конкретной строке.
SQLSetParamDecimal	Устанавливает значение конкретного десятичного параметра в конкретной строке.
SQLSetParamFloat	Устанавливает значение конкретного параметра в конкретном <i>ParameterValue</i> .
SQLSetParamInt	Устанавливает значение конкретного параметра в конкретном <i>ParameterValue</i> .
SQLSetParamLong	Устанавливает значение конкретного параметра в конкретном <i>ParameterValue</i> .
SQLSetParamNull	Устанавливает значение конкретного параметра в нуль.
SQLSetParamTime	Устанавливает значение конкретного параметра времени в конкретной строке.
SQLSetStatement	Запускает буфер инструкций SQL, используя содержимое <i>SQLStatement</i> на установленном соединении, <i>ConnectionID</i> . В <i>ConnectionID</i> может находиться только один буфер инструкций SQL.

Функция	Описание
SQLTransact	Определяет начало группы команд транзакции. Эта группа производит действия в интервале между выполнением команд SQLTransact() и SQLCommit() и называется набором транзакций. Набор транзакций обрабатывается как единая транзакция. После выполнения команды SQLTransact() все последующие операции не фиксируются в базе данных до тех пор, пока не будет выполнена команда SQLCommit().
SQLUpdate	Обновляет запись в соответствие со значением текущего тэга.
SQLUpdateCurrent	Берет выбранную в данный момент запись и обновляет ее любым новым значением InTouch. .

Параметры SQL

Этот раздел описывает параметры, необходимые для каждой функции SQL. Когда параметр вносится в сценарий заключенным в кавычки, например, «Параметр1», будет использована заданная строка. Если же кавычек нет, то предполагается, что Параметр1 является тэгом и система обратится к словарю тэгов InTouch за его значением.

Пример:

"c:\main\file" против Location

где : location — это тэг InTouch текстового типа
"c:\main\file" — текстовая строка

Один или несколько следующих параметров будут входить в большинство функций SQL:

Параметр	Описание
BindList	Соответствует одному из имен списка связей в файле SQL.DEF.
ConnectionID	Внутренний целочисленный тэг, созданный пользователем для хранения номера (ID), присвоенного функцией соединения SQL для каждой связи в базе данных.
ConnectionString	Строка, которая идентифицирует базу данных и дополнительную информацию, используемую функцией SQLConnect() для входа в систему.
ErrorMsg	Текстовая переменная, содержащая описание сообщения об ошибке. Подробное описание всех сообщений об ошибках содержится в главе 5, «Отладка приложений».
FileName	Имя файла, в котором содержатся данные.
MaxLen	Максимальный размер столбца, с которым ассоциируется данный параметр. Эта установка определяет какого типа параметр, переменных символов или длинных переменных символов. Если <i>MaxLen</i> меньше или равен самой длинной символьной строке, дозволенной в базе данных, то это будет параметр типа переменных символов, а если больше, то он будет относиться к типу длинных переменных символов.

Параметр	Описание
OrderByExpression	Определяет столбцы и порядок сортировки. Сортировка может происходить только по именам столбцов. Выражению должен быть задан формат: ColumnName [ASC DESC] Сортировать выбранную таблицу по имени столбца (например, менеджер) в возрастающем порядке: "manager ASC" Сортировать по нескольким колонкам, формат выражения следующий: ColumnName [ASC DESC], ColumnName [ASC DESC] Сортировать выбранную таблицу по одному имени столбца (например, температура) в возрастающем порядке, а по другому имени (время) — в убывающем: "temperature ASC, time DESC"
ParameterNumber	Фактический номер параметра в инструкции.
ParameterType	Тип данных конкретного параметра. Правильные значения приведены в таблице:

Тип	Значение	Описание
Char	1	Фиксированная длина набранных пробелов
Var Char	2	Строка переменной длины
Decimal	3	Номер BCD
Integer	4	4-х байтное целое число со знаком
Small integer	5	2-х байтное целое число со знаком
Var Char	2	Строка переменной длины
Decimal	3	Номер BCD
Integer	4	4-х байтное целое число со знаком
Small integer	5	2-х байтное целое число со знаком
Float	6	4-х байтное число с плавающей точкой
Double Precision Float	7	8-ми байтное число с плавающей точкой
DateTime	8	26-ти байтное значение даты-времени
Date	111	26-ти байтное значение даты-времени

	Time	112	26-ти байтное значение даты-времени
	No Type	0	Нет типа данных
ParameterValue	Установка фактического значения.		
Precision	Значение с точностью до десятых, максимальный размер символа или длина значения даты-времени в байтах.		
RecordNumber	Фактический номер записи для поиска.		
ResultCode	Целая переменная возвращаемая из большинства функций SQL. ResultCode возвращается нулем (0), если функция достигла успеха и отрицательной величиной, если не достигла. Подробная информация содержится в главе 5, «Отладка приложений».		
Scale	Десятичный масштаб значения. Это значение требуется только в том случае, если оно применимо к параметру при установке на нуль.		
SQLHandle	Когда используются инструкции с улучшенными функциональными возможностями, SQL возвращает <i>SQLHandle</i> и использует его для внутренних нужд.		
SQLStatement	Фактическая инструкция, пример: <code>ResultCode=SQLSetStatement(Connecti onID,"Select LotNo, LotName from LotInfo");</code>		
TableName	Имя таблицы базы данных, к которой нужно обратиться.		
TemplateName	Имя определения шаблона, которое нужно использовать.		
WhereExpression	Определяет состояние, которое может быть истинным или ложным для любой строки в таблице. Команда извлекает из таблицы лишь те строки, для которых условие является истинным. Выражение должно иметь следующий формат:		

ColumnName comparison_operator expression

Примечание. Если столбец символьного типа данных, выражение должно заключаться в одинарную кавычку.

Следующий пример будет выбирать все строки с именами столбцов, содержащих значение

EmployeeID:

name='EmployeeID'

Следующий пример будет выбирать все строки, содержащие часть чисел от 100 до 199:

partno>=100 and partno<200

Следующий пример будет выбирать все строки, где значение температуры превышает 350:

temperature>350

Использование функций SQL в сценариях InTouch

Функции SQL могут быть автоматически вставлены в сценарии InTouch за текущей позицией курсора путем нажатия на кнопку **Add-ons** в диалоге редактора сценариев QuickScript.

📖 Подробную информацию о сценариях InTouch можно найти в главе 6 "Руководства пользователя InTouch".

Определение сложных запросов

Программа SQL Access Manager позволяет определять сложные запросы и инструкции SQL. Эти запросы могут формироваться динамическим путем либо находиться во внешнем файле. К тому же, эти запросы могут содержать параметры, которые должны быть «переданы» в запрос за время выполнения. Запросы должны быть выполнены и, по возможности, иметь возвращаемые результаты. SQL Access Manager API позволяет выполнять любую инструкцию SQL, с которой может оперировать база данных и отыскивать результат этого запроса. Хранимые в памяти процедуры доступны для выполнения. (Эти процедуры поддерживаются не полностью.)

🌀 Подробную информацию о процедурах, хранимых в памяти, можно найти в разделе «Поддержка хранимых в памяти процедур».

Для создания сложных запросов и строковых выражений используется функция `SQLSetStatement`. В случае, если строковое выражение превышает 131 символ, используйте функцию `SQLAppendStatement`. Например:

```
SQLSetStatement( Connect_Id, "Select Speed, Ser_No from  
tablename where Ser_No =' " + Serial_input + "'");
```

В следующем примере `SQLHandle` устанавливается в нуль, поэтому инструкции перед выполнением не нужно обращаться к `SQLPepare(Connect_Id,SQLhandle)`. Поскольку `SQLHandle` не создается функцией `SQLPepare()`, то для правильного завершения инструкции выбора необходимо использовать функцию `SQLEnd()` вместо `SQLClearStatement()`.

```
SQLExceute(Connect_Id, 0);
```

```
SQLSetStatement( Connect_Id, "Select Speed, Ser_No from  
tablename where Ser_No =' " + Serial_input + "'");
```

```
SQLPrepareStatement(Connect_Id, SQLhandle);
```

В следующем примере `SQLHandle` создается функцией `SQLPrepareStatement()` и используется в `SQLExecute()`. Для завершения инструкции выбора используйте `SQLClearStatment()`, чтобы ресурсы и `SQLHandle` были полностью освобождены.

```
SQLExceute(Connect_Id, SQLhandle);
```

Динамическое создание запросов

Для динамического создания запросов требуются две дополнительные функции: **SQLSetStatement()** и **SQLAppendStatement()**. Функция **SQLSetStatement()** запускает новую инструкцию SQL. Это может быть любая правильная инструкция SQL, включающая имя хранимой в памяти процедуры. Поскольку InTouch поддерживает только символьные строки, не превышающие 131 символ, то **SQLAppendStatement()** позволяет соединять дополнительные строки в инструкцию.

Примечание. Жирным шрифтом выделены текстовые обращения к командам языка SQL.

Пример

```
ResultCode = SQLSetStatement ( ConnectionID, "Select LotNo,
LotName, LotDescription, LotQuantity from LotInfo,
ProductionInfo" );
```

```
ResultCode = SQLAppendStatement (ConnectionID, " where
LotInfo.LotNo = ProductionInfo.LotNo" );
```

```
ResultCode = SQLAppendStatement (ConnectionID, " order by
LotNo, LotName, LotQuantity" );
```

Эта инструкция готова для выполнения.

Примечание. Во многих столбцах и именах таблиц базы данных необходимо учитывать регистр. Чтобы сценарий нормально функционировал, имена должны быть такими же, как они созданы в таблицах базы данных.

Чтение инструкций SQL из файла

Запросы можно создавать в любых СУБД (таких как Microsoft Access или СУБД третьих фирм), а затем выполнять их с помощью SQL Access для InTouch. Поскольку различные СУБД будут создавать свои инструкции SQL, их можно легко сохранить в файле, используя функцию **SQLLoadStatement()** для добавления к SQL Access API.

Пример

```
ResultCode = SQLLoadStatement ( ConnectionID,
"c:\myappdir\lotquery.sql" );
```

Эта инструкция готова для выполнения.

Изменение расширенных инструкций SQL

Чтобы создать полноценный набор функций SQL, программа SQL Access Manager позволяет определять оператор "где", который будет содержать значение тэга InTouch. Для определения параметров SQL в среде выполнения предусмотрены следующие функции:

- **SQLPrepareStatement()**
- **SQLSetParamType()**
- **SQLClearStatement()**
- **SQLClearParam()**

Для выполнения замены параметра в инструкции SQL вставьте "?" в то место инструкции, где позднее должен быть определен параметр. Инструкция «готовится», параметры «устанавливаются» в инструкции, а потом инструкция выполняется.

SQLPrepareStatement() подготавливает инструкцию для выполнения, но не выполняет инструкцию, а лишь делает ее активной, чтобы можно было установить значение параметра. **SQLSetParamType()** — это набор функций, которые позволяют устанавливать значения параметров в инструкции SQL.

Пример

```
ResultCode = SQLSetStatement ( ConnectionID, "Select LotNo,
LotName, LotDescription, LotQuantity from LotInfo,
ProductionInfo" );

ResultCode = SQLAppendStatement (ConnectionID, " where
LotInfo.LotNo = ?");

ResultCode = SQLAppendStatement (ConnectionID, " order by
LotNo, LotName, LotQuantity" );

ResultCode = SQLPrepareStatement (ConnectionID, SQLHandle
);

{return the statement handle into tag 'SQLHandle'}

ResultCode = SQLSetParamInt ( SQLHandle, 1, tagLotNumber );
{put the value of tagLotNumber into param}
```

Поскольку инструкция имеет лишь один параметр, она готова для выполнения.

Одновременно с выполнением и завершением подготовки инструкции может быть вызвана функция **SQLClearStatement()** для освобождения ресурсов, ассоциируемых с этой инструкцией.

Примечание. **SQLEnd()** вызывается для освобождения инструкций SQL «без имени» (созданных существующими функциями SQL Access) и инструкций, созданных функциями **SQLSetStatement()** и **SQLLoadStatement()**, но еще не готовых.

Выполнение расширенных инструкций SQL

После того, как инструкция либо создана динамическим путем, либо считана из файла, затем произвольно подготовлена и изменена, она готова к выполнению. Для выполнения инструкции SQL Access Manager API использует функцию **SQLExecute()**. **SQLExecute()** выполнит либо активную на данный момент инструкцию (т. е. созданную с помощью **SQLSetStatement()** или **SQLLoadStatement()**), либо инструкцию, которая предварительно подготовлена и определяется основным параметром инструкции.

Пример 1

```

ResultCode = SQLLoadStatement ( ConnectionID,
    "c:\myappdir\lotquery.sql" );

ResultCode = SQLExecute (ConnectionID, "BindList", 0);
{put the results of the select into the tags specified in
BindList. prepared statement handle is zero}

ResultCode = SQLNext ( ConnectionID );
{Get results of Select}

```

Пример 2

```

ResultCode = SQLSetStatement ( ConnectionID, "Select LotNo,
LotName, LotDescription, LotQuantity from LotInfo,
ProductionInfo" );

ResultCode = SQLAppendStatement (ConnectionID, " where
LotInfo.LotNo = ?");
{question mark means I'll get back to you}

ResultCode = SQLAppendStatement (ConnectionID, " order by
LotNo, LotName, LotQuantity" );

ResultCode = SQLPrepareStatement (ConnectionID, SQLHandle
);
{return the statement handle into tag 'SQLHandle'}

ResultCode = SQLSetParamInt ( SQLHandle, 1, tagLotNumber );
{put the value of tagLotNumber into param}

ResultCode = SQLExecute (ConnectionID, "BindList",
SQLHandle); {put the results of the Select into the tags
specified in
BindList prepared statement handle is in SQLHandle}

ResultCode = SQLNext ( ConnectionID );
{Get results of Select }

```


Пример 3

SQLSetStatement – эта инструкция должна использоваться, когда сложные запросы и строчные выражения превышают 131 символ. Если строчное выражение больше чем 131 символ, используйте SQLAppend. Например:

```
SQLSetStatement( Connect_Id, "Select Speed, Ser_No from
tablename where Ser_No =' " + Serial_input + "'");
```

```
SQLExceute(Connect_Id, 0);
```

В этом примере SQLhandle устанавливается в нуль, поэтому инструкции перед выполнением не нужно вызывать SQLPepare(Connect_Id,SQLhandle).

Поскольку *SQLHandle* не создается функцией **SQLPepare()**, то для правильного завершения инструкции выбора используйте функцию **SQLEnd()** вместо **SQLClearStatement()**.

```
SQLSetStatement( Connect_Id, "Select Speed, Ser_No from
tablename where Ser_No =' " + Serial_input + "'");
```

```
SQLPrepareStatement(Connect_Id,SQLhandle);
```

```
SQLExceute(Connect_Id,SQLhandle);
```

В этом примере SQLhandle создается с помощью SqlPrepareStatement и использует функцию SQLExceute. Для завершения инструкции выбора используйте **SQLClearStatment()**, чтобы ресурсы и *SQLHandle* были полностью освобождены.

Поддержка хранимых в памяти процедур

Функция **SQLExecute()** поддерживает выполнение некоторых хранимых в памяти процедур. Например, предположим, что вы создаете хранимые в памяти процедуры на сервере базы данных "LotInfoProc," который содержит следующие инструкции выбора: "Select LotNo, LotName from LotInfo." Для выполнения процедуры и получения результатов необходимо написать следующий сценарий:

Для Microsoft SQL Server

```
ResultCode = SQLSetStatement ( ConnectionID, "LotInfoProc" );
```

```
ResultCode = SQLExecute(ConnectionID, "BindList", 0);
```

```
ResultCode = SQLNext ( ConnectionID );
```

```
{ Get results of Select}
```

Для Oracle или Microsoft Access

```
ResultCode = SQLSetStatement ( ConnectionID, "{CALL
LotInfoProc}" );
```

```
ResultCode = SQLExecute(ConnectionID, "BindList", 0);
```

```
ResultCode = SQLNext ( ConnectionID );
```

```
{ Get results of Select}
```

Г Л А В А 5

Отладка

Эта глава рассказывает о том, как производить отладку приложений SQL, используя коды результатов, возвращаемые функциями SQL. Первый раздел главы описывает функцию **SQLErrorMsg()**, содержит таблицу кодов результатов SQL и соответствующие сообщения об ошибках. Второй раздел содержит таблицы с сообщениями об ошибках базы данных.

Содержание

- [Функции отладки](#)
- [Сообщения об ошибках базы данных](#)

Функции отладки

Все функции SQL возвращают *ResultCode*, который может быть использован для отладки. Функция **SQLErrorMsg()** возвращает сообщение об ошибке, соответствующее *ResultCode*.

Пример:

```
ErrorMsg=SQLErrorMsg(ResultCode);
```

где : **ErrorMsg** — внутренний текстовый тэг.
ResultCode — целочисленная величина, полученная от предыдущей функции SQL.

Сообщения об ошибках с кодом результата

За дополнительной информацией о кодах результата, которая не изложена в настоящем руководстве, обращайтесь к документации по используемой базе данных и просмотрите журнал Wonderware Logger.

Функция **SQLErrorMsg()** устанавливает значение текстового тэга *ErrorMsg* приложения InTouch. Ниже приведены некоторые возможные коды результата SQL с описанием соответствующих сообщений об ошибках:

Код результата	Сообщение об ошибке	Описание
0	Не было ошибок	Успешный ввод команды.
-5	Нет больше рядов для выборки	Достигнута последняя запись в таблице.
-1001	Недостаточно памяти	Недостаточно памяти для выполнения данной функции.
-1002	Неверная связь	Неправильное соединение, переданное функции.
-1003	Не было найдено списка привязок	Заданное имя списка связей не существует.
-1004	Не были найдены шаблоны	Заданное имя шаблона таблицы не существует.
-1005	Внутренняя ошибка	Произошла внутренняя ошибка. Обратитесь в службу технической поддержки.
-1006	Предупреждение: строка - null	Предупреждение: прочитанная в базе данных строка – нулевая.
-1007	Предупреждение: Строка была усечена	Предупреждение: строка, прочитанная в базе данных, превышает 131 символ и произвольно усечена.
-1008	Нет оператора where	Нет оператора Where в команде Delete.
-1009	Ошибка при соединении	Просмотрите в журнале Wonderware Logger детальное описание сбоя соединения.
-1010	База данных, указанная в блоке DB= строки соединения, не существует	Заданная база данных не существует.

Код результата	Сообщение об ошибке	Описание
-1011	Не было выбрано рядов	Попытка выполнить команды SQLNumRows() , SQLFirst() , SQLNext() , SQLPrev() без выполнения SQLSelect() .
-4149	Неверный оператор соединения или указатель запроса	Неправильно определен тип столбца. Например, если тип столбца в шаблоне таблицы определен как character вместо char для файла dBASE , то будет возвращена данная ошибка.

Сообщения об ошибках базы данных

Oracle

Сообщение об ошибке	Решение
ORA-03112 - Host String Syntax error	Если файл NETINIT.EXE не запущен, поместите его в папку "Автозагрузка" Windows. Если вы запустили файл NETINIT.EXE и хотите установить более одного соединения, необходимо выделить дополнительную память. Для этого нужно установить параметр WIN_REMOTE_SESSIONS в файле CONFIG.ORA равный числу требуемых соединений. Следующий пример выделит память для 4-х соединений: WIN_REMOTE_SESSIONS=4
ORA-3121 - No interface driver connected	Запустите соответствующий SQL*NET TSR для сетевой системы <u>до</u> входа в Windows и использования InTouch SQL Access.
ORA-6435 - NetBIOS: Unable to add local name to name table	Необходимо выполнение сетевого ПО (Novell, LAN Manager, и т. п.).
ORA-09301 - Local kernel only supported in standard mode	Укажите имя сервера (" SRVR= ") в строке соединения.
ORA-06430 - Unable to make connection	Необходимо, чтобы атрибуты в строке соединения были точными и расположенными в правильном порядке.

Sybase или Microsoft SQL Server

Сообщение об ошибке**Решение**

You cannot have more than one statement active at a time

Попытайтесь выполнить команду SQL после выполнения **SQLSelect()**. Выполните **SQLEnd()** для освобождения системных ресурсов **SQLSelect()** или используйте отдельный **ConnectionId** для второй инструкции.

There is not enough memory available to process the command

Попытайтесь повторить загрузку системы рабочей станции клиента.

Invalid object name table name

Имя таблицы в используемой базе данных не существует. Попробуйте задать имя DB=database.

dBASE

Сообщение об ошибке**Решение**

File or DLL not found

В Windows файл QEDBF.DLL должен находиться либо в текущем каталоге, либо в каталоге \windows\system, указанном в пути DOS .

Invalid connection

Убедитесь в наличии файлов DLL, указанных в пути. Для поддержки DBF необходим файл QLDBF.DLL.

The connection or statement handle you provided is not valid

Дополнительно просмотрите журнал Wonderware Logger. В инструкции могут быть синтаксические ошибки.

П Р И Л О Ж Е Н И Е А**Резервные ключевые слова SQL
Access и ODBC**

В этом приложении перечислены ключевые слова, запрещенные для использования в списках связей и шаблонах таблиц SQL Access, и в интерфейсе Open Database Connectivity (ODBC).

Если резервное ключевое слово будет использовано в качестве заголовка столбца в списке связей или в шаблоне таблицы, будет сгенерировано сообщение об ошибке в Wonderware Logger. Тип сообщения об ошибке зависит от используемого драйвера ODBC и от местонахождения ключевого слова. Например, наиболее частой ошибкой является использование слов DATE и TIME в качестве заголовка столбца в списке связей или в шаблоне таблицы. Во избежание этой ошибки используйте немного отличные заголовки, напр., "aDATE" и "aTIME".

Резервные ключевые слова определяются языком SQL, используемым программой InTouch SQL Access. Они также распознаются используемым драйвером ODBC. SQL Access передает команду SQL, содержащую хотя бы одно резервное ключевое слово, в файл ODBC.DLL. Если команда SQL не может быть интерпретирована корректно, SQL Access генерирует сообщение об ошибке в журнале Wonderware Logger.

Резервные ключевые слова приводятся ниже в алфавитном порядке:

ABSOLUTE	COLLATE	DISCONNECT
ADA	COLLATION	DISPLACEMENT
ADD	COLUMN	DISTINCT
ALL	COMMIT	DOMAIN
ALLOCATE	CONNECT	DOUBLE
ALTER	CONNECTION	DROP
AND	CONSTRAINT	ELSE
ANY	CONSTRAINTS	ENDE
ARE	CONTINUE	ESC
AS	CONVERT	EXCEPT
ASC	CORRESPONDING	EXCEPTION
ASSERTION	COUNT	EXEC
AT	CREATE	EXECUTE
AUTHORIZATION	CURRENT	EXISTS
AVG	CURRENT_DATE	EXTERNAL
BEGIN	CURRENT_TIME	EXTRACT
BETWEEN	CURRENT_TIMESTAMP	FALSE
BIT	CURSOR	FETCH
BIT_LENGTH	DATE	FIRST
BY	DAY	FLOAT
CASCADE	DEALLOCATE	FOR FOREIGN
CASCADDED	DEC	FORTRAN
CASE	DECIMAL	FOUND
CAST	DECLARE	FROM FULL
CATALOG	DEFERRABLE	GET
CHAR	DEFERRED	GLOBAL
CHAR_LENGTH	ENTF	GO
CHARACTER	DESC	GOTO
CHARACTER_LENGTH	DESCRIBE	GRANT
CHECK	DESCRIPTOR	GROUP
CLOSE COALESCE	DIAGNOSTICS	HAVING
COBOL	DICTIONARY	HOURL

IDENTITY
IGNORE
IMMEDIATE

IN	NUMERIC	SQL
INCLUDE	OCTET_LENGTH	SQLCA
INDEX	OF	SQLCODE
INDICATOR	OFF	SQLERROR
INITIALLY	ON	SQLSTATE
INNER	ONLY	SQLWARNING
INPUT	OPEN	SUBSTRING
INSENSITIVE	OPTION	SUM
EINFÜGEN	OR	SYSTEM
INTEGER	ORDER	TABLE
INTERSECT	OUTER	TEMPORARY
INTERVALL	OUTPUT	THEN
INTO	OVERLAPS	TIME
IS	PARTIAL	TIMESTAMP
ISOLATION	PASCAL	TIMEZONE_HOUR
JOIN	PLI	TIMEZONE_MINU
KEY	POSITION	TO
LANGUAGE	PRECISION	TRANSACTION
LAST	PREPARE	TRANSLATE
NACH-LINKS	PRESERVE	TRANSLATION
LEVEL	PRIMARY	TRUE
LIKE	PRIOR	UNION
LOCAL	PRIVILEGES	UNIQUE
LOWER	PROCEDURE	UNKNOWN
MATCH	PUBLIC	UPDATE
MAX	RESTRICT	UPPER
MIN	REVOKE	USAGE
MINUTE	NACH RECHTS	OPRN
MODULE	ROLLBACK	USING
MONTH	ROWS	WERT
MUMPS	SCHEMA	VALUES
NAMES	SCROLL	VARCHAR
NATIONAL	SECOND	VARING
NCHAR	SECTION	VIEW
NEXT	SELECT	WHEN
NONE	SEQUENCE	WHENEVER
NOT	SET	WHERE
NULL	SIZE	WITH
NULLIF	SMALLINT	WORK
	SOME	YEAR

Предметный указатель

“B”

BindListName, 4-5

“C”

Column Name, 3-4

ConnectionID, 4-5

ConnectionString, 4-5

CSV, 1-2“,3-10

“D”

dBASE

Поддерживаемые типы данных, 2-11“,2-14

Требования к соединению, 2-10

DBNMP3.DLL, 2-8

Delim, функция, 3-6

“E”

ErrorMsg, 4-5

“F”

FileName, 4-5

“M”

MaxLen, 4-5

Microsoft Access

Поддерживаемые типы данных, 2-12“,2-14

Требования к соединению, 2-12

Microsoft SQL Server

настройка клиента, 2-8

Поддерживаемые типы данных, 2-9

Требования к соединению, 2-8

“O”

ODBC.INI, 1-5

Oracle

Поддерживаемые типы данных, 2-14

Oracle 6, 2-2

запуск SQL*Net TSR и NETINIT.EXE, 2-2

настройка клиента для Windows, 2-2

Требования к соединению, 2-2

Oracle 7, 2-4

Настройка имени источника данных, 2-5

OrderByExpression, 4-6

“P”

Paradox

Поддерживаемые типы данных, 2-13“,2-16

Требования к соединению, 2-13

ParameterNumber, 4-6

Parameters, 4-5

ParameterType, 4-6

ParameterValue, 4-7

Precision, 4-7

“Q”

QuickScripts, 4-8

“R”

RecordNumber, 4-7

ResultCode, 4-7“,5-2

“S”

Scale, 4-7

SQL Access Manager

Введение, 1-2

Обзор, 3-2

SQL Function format, 4-2

SQL.DEF, 1-2“,3-10

SQLAppendStatement, 4-2

SQLClearParam, 4-2

SQLClearStatement, 4-2

SQLClearTable, 4-2

SQLCommit, 4-2

SQLConnect, 2-2“,2-8“,4-2

SQLCreateTable, 4-2

SQLDelete, 4-2

SQLDisconnect, 4-2

SQLDropTable, 4-2

SQLEnd, 4-2

SQLErrorMsg, 4-2“,5-2

SQLExecute, 4-2

SQLFirst, 4-2

SQLGetRecord, 4-2

SQLHandle, 4-7

SQLInsert, 4-3

SQLInsert, 3-6

SQLInsertEnd, 4-3

SQLInsertExecute, 4-3

SQLInsertPrepare, 4-3

SQLLast, 4-3

SQLLoadStatement, 4-3

SQLManageDSN, 4-3
 SQLNext, 4-3
 SQLNumRows, 4-3
 SQLPrepareStatement, 4-3
 SQLPrev, 4-3
 SQLRollback, 4-3
 SQLSelect, 4-4
 SQLSetParamChar, 4-4
 SQLSetParamDate, 4-4
 SQLSetParamDateTime, 4-4
 SQLSetParamDecimal, 4-4
 SQLSetParamFloat, 4-4
 SQLSetParamInt, 4-4
 SQLSetParamLong, 4-4
 SQLSetParamNull, 4-4
 SQLSetParamTime, 4-4
 SQLSetStatement, 4-4
 SQLStatement, 4-7
 SQLTransact, 4-4
 SQLUpdate, 3-6“, 4-4
 SQLUpdateCurrent, 4-4
 Structured Query Language, 1-1
 Sybase
 Настройка клиента, 2-8
 Поддерживаемые типы данных, 2-9“, 2-14
 Требования к соединению, 2-8
 Sybase или Microsoft SQL Server
 Поддерживаемые типы данных, 2-14

“Т”

Table Template
 команда, 3-8
 TableName, 4-7
 Tagname.FieldName, 3-4
 TemplateName, 4-7

“W”

WhereExpression, 4-7

“Б”

База данных, 1-1
 Броузер тэгов, 3-3

“В”

Выполнение расширенных инструкций SQL,
 4-11

“Д”

Динамическое создание запросов, 4-9

“З”

Запись даты и времени в поле даты Oracle, 2-7

Запросы

 динамическое создание, 4-9
 сложные, 4-9

“И”

Изменение расширенных инструкций SQL, 4-10

Изменение списка привязок, 3-5

Изменение шаблона таблицы, 3-9

Имя списка привязок, 3-10

Имя шаблона таблицы, 3-10

Использование функций SQL в InTouch, 4-8

“К”

Команды

 Table Template, 3-8

“Л”

Лицензия FactorySuite, 1-4

“Н”

Настройка SQL Access Manager, 3-1

Настройка и соединение баз данных, 2-1

Настройка списка привязок, 3-2

Настройка шаблонов таблицы, 3-7

“О”

Об этом руководстве, 1-3

Определение сложных запросов, 4-8

Отладка, 5-1

Отладка функций SQL, 5-2

“П”

Параметр

 BindListName, 4-5

 ConnectionID, 4-5

 ConnectionString, 4-5

 ErrorMsg, 4-5

 FileName, 4-5

 MaxLen, 4-5

 OrderByExpression, 4-6

- ParameterNumber, 4-6
- ParameterType, 4-6
- ParameterValue, 4-7
- Precision, 4-7
- RecordNumber, 4-7
- ResultCode, 4-7
- Scale, 4-7
- SQLHandle, 4-7
- SQLStatement, 4-7
- TableName, 4-7
- TemplateName, 4-7
- WhereExpression, 4-7
- Параметры SQL, 4-5
- Поддерживаемые базы данных
 - dBASE, 2-10
 - Microsoft Access, 2-12
 - Microsoft SQL Server, 2-8
 - Oracle 6, 2-2“,2-4
 - Paradox, 2-13
 - Sybase, 2-8
- Поддерживаемые типы данных, 2-14
- Поддерживаемые типы данных Oracle 6, 2-3
- Поддержка хранимых в памяти процедур, 4-12
- программа NETINIT.EXE, 2-2
- Программа ODBC Administrator, 1-5
- “P”
 - Работа с dBASE, 2-10
 - Работа с Microsoft Access, 2-12
 - Работа с Oracle 7.2, 2-4
 - Работа с Paradox, 2-13
 - Работа с Sybase или Microsoft SQL Server, 2-8
 - Разделители, 3-6
- “C”
 - Совместимость с ODBC, 1-5
 - Соединение с Oracle 6, 2-2“,2-4
 - Создание списка привязок, 3-2
 - Сообщения об ошибках базы данных
 - dBASE, 5-4
 - Oracle, 5-3
 - Sybase или Microsoft SQL Server, 5-4
 - Сообщения об ошибке с кодом результата, 5-2
 - Специальные разделители, 3-6
 - Список привязок, 3-2
 - Броузер тэгов, 3-3
 - изменение, 3-5
 - создание, 3-2
 - удаление, 3-5
- “T”
 - Техническая поддержка Wonderware, 1-4
 - Требования к соединению, 1-5
- “Y”
 - Удаление списка привязок, 3-5
 - Удаление шаблона таблицы, 3-9
- “Ф”
 - Функции
 - SQLAppendStatement, 4-2
 - SQLClearParam, 4-2
 - SQLClearStatement, 4-2
 - SQLClearTable, 4-2
 - SQLCommit, 4-2
 - SQLConnect, 2-2“,2-8“,4-2
 - SQLCreateTable, 4-2
 - SQLDelete, 4-2
 - SQLDisconnect, 4-2
 - SQLDropTable, 4-2
 - SQLEnd, 4-2
 - SQLErrorMsg, 4-2
 - SQLExecute, 4-2
 - SQLFirst, 4-2
 - SQLGetRecord, 4-2
 - SQLInsert, 4-3
 - SQLInsertEnd, 4-3
 - SQLInsertExecute, 4-3
 - SQLInsertPrepare, 4-3
 - SQLLast, 4-3
 - SQLLoadStatement, 4-3
 - SQLManageDSN, 4-3
 - SQLNext, 4-3
 - SQLNumRows, 4-3
 - SQLPrepareStatement, 4-3
 - SQLPrev, 4-3
 - SQLRollback, 4-3
 - SQLSelect, 4-4
 - SQLSetParamChar, 4-4
 - SQLSetParamDate, 4-4
 - SQLSetParamDateTime, 4-4
 - SQLSetParamDecimal, 4-4
 - SQLSetParamFloat, 4-4
 - SQLSetParamInt, 4-4
 - SQLSetParamLong, 4-4
 - SQLSetParamNull, 4-4

SQLSetParamTime, 4-4

SQLSetStatement, 4-4

SQLTransact, 4-4

SQLUpdate, 4-4

SQLUpdateCurrent, 4-4

“Ч”

Чтение инструкций SQL из файла, 4-9

“Ш”

Шаблон таблицы, 3-7

изменение, 3-9

создание, 3-7“,3-9

удаление, 3-9

“Э”

Электронная документация, 1-3