KLINKMANN
www.klinkmann.com

Wonderware® FactorySuite®

Общее описание архиватора
IndustrialSQL Server™ 9.0

Редакция E

Дата пересмотра: 15.2.06 (13.07.05)

KLINKMANN
www.klinkmann.com

Санкт-Петербург	тел. +7 812 327 3752; klinkmann@klinkmann.spb.ru
Москва	тел. +7 095 461 3623; moscow@klinkmann.spb.ru
Київ	тел. +38 044 239 1250; klinkmann@klinkmann.kiev.ua
Tallinn	tel. + 372 668 4500; klinkmann.est@klinkmann.ee
Rīga	tel. +371 738 1617; klinkmann@klinkmann.lv
Vilnius	tel. +370 5 216 215 1646; post@klinkmann.lt
Helsinki	ph. +358 9 540 4940; automation@klinkmann.fi

Invensys Systems. Inc.

26561 Rancho Parkway South

Lake Forest, CA 92630 USA

(949) 727-3200

http://www.wonderware.com

Торговые марки

Содержание

Предварительные сведения

Общее описание архиватора IndustrialSQL Server™ предоставляет описание архитектуры архиватора IndustrialSQL Server и входящих в него различных подсистем и модулей, которые и составляют систему. Данное руководство может использоваться как справочник по всей основополагающей инфррмации, касающейся различных компонентов архиватора IndustrialSQL Server.

Состав документации по IndustrialSQL Server

Документация по архиватору IndustrialSQL Server содержит следующие руководства:

- *Руководство по установке архиватора IndustrialSQL Server (IndustrialSQL Server Historian Installation Guide).* Данное руководство содержит подробные сведения о порядке установки архиватора IndustrialSQL Server, а также требования к аппаратным и программным средствам и инструкции по переносу системы.
- *Общее описание архиватора IndustrialSQL Server (IndustrialSQL Server Historian Concepts Guide).* Данное руководство содержит общие сведения о всей системе IndustrialSQL Server и подробное описание каждой из его подсистем.
- *Руководство по администрированию архиватора IndustrialSQL Server (IndustrialSQL Server Historian Administration Guide).* В данном руководстве описывается проводить администрирование и поддержку установленного архиватора IndustrialSQL Server, такое как определение параметров сбора и хранения данных, параметров подсистемы защиты данных и мониторинг системы.
- *Справочник по базе данных архивирования IndustrialSQL Server (IndustrialSQL Server Historian Database Reference).* Данное руководство предоставляет описание всех объектов базы данных IndustrialSQL Server, таких как таблицы, представления и хранимые процедуры.
- *Руководство корпоративного пользователя архиватора IndustrialSQL Server (IndustrialSQL Server Historian Enterprise Edition User's Guide).* В данном руководстве описывается обеспечивающая безаварийность кластеризация системы и приводятся принципы планирования, конфигурирования, использования и контроля корпоративной архиватора IndustrialSQL Server.
- *Словарь архиватора IndustrialSQL Server (IndustrialSQL Server Historian Glossary).* Данное руководство содержит определения терминов, используемых в других документах комплекта.

PDF-файлы всех указанных документов имеются на установочном компакт-диске архиватора IndustrialSQL Server. Вы можете легко распечатать информацию из них. Кроме того, документация по архиватору IndustrialSQL Server предоставляется также в виде оперативного

справочного файла, доступ к которому осуществляется из средства управления System Management Console.

Соглашения, принятые в документе

В настоящем документе используются следующие соглашения:

Соглашение	**Для чего используется**
Заглавные Буквы	Имена файлов и пути.
Полужирный шрифт	Меню, команды, названия диалоговых окон и их параметры.
`Моноширинный шрифт`	Примеры программ и текст, отображаемый на экране.

ГЛАВА 1

Введение

Архиватор IndustrialSQL Server наводит мост между обширным производственным мониторингом реального времени и открытым и гибким административным информационным окружением. Архиватор делает следующее:

- Накапливает производственную информацию от серверов в/в[1] Wonderware, DA Server, ИЧМ[2]-приложений InTouch, Industrial Application Server и других устройств.
- Сжимает и сохраняет данные.
- Отвечает на SQL-запросы производственной информации.

Архиватор также содержит информацию о системных событиях, сводках, конфигурациях, безопасности, резервном копировании и системном мониторинге.

Архиватор IndustrialSQL Server тесно взаимодействует с Microsoft SQL Server.

Содержание

Решение Архиватор IndustrialSQL Server

Архиватор IndustrialSQL Server является реляционной базой данных реального времени, предназначенной для хранения производственной и технологической информации. Архиватор получает и записывает данные как с полной, так и с заданной разрешающей способностью и предоставляет клиентским приложениям на настольных компьютерах оперативную и архивную информацию вместе с информацией о конфигурации, событиях, сводках и другими производственными данными. Архиваторе IndustrialSQL Server сочетает в себе мощь и открытость Microsoft SQL Server с высокой скоростью сбора и упаковки информации систем реального времени.

Производственные данные

Производственные данные представляют собой информацию любого типа, имеющую значение для успешного выполнения производственного процесса. К производственной относится следующая информация.

[1] в/в – в/ва *(пер.)*
[2] ИЧМ – человеко-машинный интерфейс *(пер.)*

- Данные реального времени (real-time) – Каково текущие значения данного тэга?
- Архивные данные (historical) – Какими были значения тэгов каждую секунду прошлого понедельника?
- Сводные данные (summary) – Каким было среднее арифметическое каждого из данных пяти тэгов?
- Событийные данные (event) – Когда этот котёл отключился?
- Конфигурационные данные (configuration) – Сколько серверов в/в я имею и каких типов?

Для повышения эффективности и качества при снижении себестоимости нужно, чтобы все эти наколенные данные были доступны для анализа. Обычно производственные данные анализируются, чтобы обеспечивать:

- Характеристики, диагностику и оптимизацию производственного процесса.
- Планово-предупредительные ремонты оборудования.
- Качество продукции и процессов (SPC/SQC);
- Охрану здоровья и безопасность труда; защиту окружающей среды (EPA/FDA).
- Производственную отчётность.
- Анализ аварий.

О реляционных базах данных

Системы управления реляционными базами данных (СУРБД) хранят информацию в таблицах, имеющих отношения, или связанных, друг с дрцгом. Запись и извлечиние информации с использованием таких таблиц выполняется более эффективно, чем в случае записи данных в лдну большую таблицу. Microsoft SQL Server является реляционной базой данных.

SQL – язык, используемый для взаимодействия с реляционными базами данных, – является промышленным "суперстандартом", поддерживаемым сотнями поставщиков программных средств. Он предоставляет не имеющую себе равных в производственных условиях открытость. Сегодня реляционные базы данных полностью отработаны и являются общепринятой информационной технологией баз данных. Мощь и гибкость SQL значительно превосходит характеристики узкоспециализированных интерфейсов, пришедших в производственную среду.

Ограничения реляционных баз данных

Обычные реляционные базы данных не подходят для хранения производственных данных по из-за следующих ограничений.

- Неспособность обрабатывать объёмы производимой производственной средой информации.
- Неспособность поддерживать большую скорость записи производственных данных.
- Язык SQL не способен эффективно обрабатывать временные ряды данных.

Промышленные предприятия имеют многие тысячи тэгов, изменяющихся с различной скоростью. Несколько месяцев работы предприятия в случае обычных реляционных баз данных дадут в результате сотни гигабайт данных.

Например, завод с 10000 параметров, изменяющихся в среднем каждые две секунды, будет ежесекундно генерировать 5000 новых значений. Следовательно, ежесекундно в таблицы базы данных нужно добавлять 5000 новых строк. Такое быстродействие обычные реляционные базы, такие как Oracle или SQL Server на стандартном оборудовании, обеспечить не в состоянии.

Архиватор IndustrialSQL Server как реляционная база данных "реального времени"

Как реляционная база данных реального времени, архиватор IndustrialSQL Server представляет собой расширение Microsoft SQL Server, предоставляющее скорость накопления информации, более чем на порядок превышающую скорость Microsoft SQL Server, снижающее требования к объёмам запоминающих устройств и имеющее элегантгные расширения стандартного языка (SQL) для запросов временных рядов данных.

- **Высокая скорость сбора данных**

  Используется широкий набор серверов в/в Wonderware и серверов DAServer для подключения более 500 типов устройств контроля и сбора данных.

  Разработанный для оптимального накопления и хранения аналоговой, дискретной и строковой информации архиватор IndustrialSQL Server значительно превосходит все обычные реляционные базы данных на аналогичном оборудовании, делая вполне реальной высокоскоростное сохранение информации в реляционных базах данных. Архиватор IndustrialSQL Server в несколько раз быстрее обычных СУРБД накапливает и записывает производственные данные.

  Серверы в/в и DAServer поддерживают протокол SuiteLink™. Протокол SuiteLink позволяет серверам в/в работать с метками времени и качества, обеспечивая при этом значительное повышение скорости накопления информации.

- **Уменьшенный объём хранения**

  Архиватор IndustrialSQL Server для сохранения данных занимает лишь часть пространства, которое потребовалось обычной реляционной базе данных. Фактический размер дискового пространства, нужного для хранения производственной информации, зависит от размера и природы предприятия и от требуемой длины истории предприятия.

- **Расширение SQL во ременные параметры**

  Язык SQL не поддерживает временные ряды данных. В частности, SQL не имеет никаких средств разрешения по времени возвращаемых данных. Примером разрешения может служить равномерное распределённые данных по отрезку времени.

  Microsoft SQL Server поддерживает собственное расширение языка SQL, называемое Transact-SQL. Архиватор IndustrialSQL Server расширяет Transact-SQL, обеспечивая управление разрешением по времени, и предоставляет возможность выполнения вычислений на базе временных функций, таких как расчёты скорости изменения и обсчёт процессов на сервере.

Интеграция с Microsoft SQL Server

Значительная часть производственной информации имеет такие же характеристики, как и обычные административные данные. Например, параметры конфигурации являются относительно статическими или не изменяются в режиме реального времени. В процессе деятельности

производства могут добавляться новые и удаляться существующие тэги, могут изменяться описания и единицы измерения. Для записи информации такого типа используется база данных Microsoft SQL Server, называемая Рабочей (Runtime) базой данных.

Рабочая база данных является одним из основных компонентов всей архиватора IndustrialSQL Server. Во время поставки Рабочая база данных уже содкржит определения всех объектов, таких как таблицы, представления и хранимые процедуры, необходимых для хранения конфигурационных параметров типичного предприятия. При необходимости конфигурационные параметры могут изменяться с помощью Редактора конфигурации (Configuration Editor), входящего в состав системной консоли управления (System Management Console).

Для доступа к производственной информации реального времени, записываемой архиватором в отдельную базу данных SQL Server, используется провайдер Microsoft SQL Server OLE DB. Вы можете запрашивать у Microsoft SQL Server как конфигурационную информацию из Рабочей базы данных, так и исторические данные с диска, наблюдая при этом бесшовную интеграцию.

Так как архиватор IndustrialSQL Server тесно связан с Microsoft SQL Server и является, по существу, его расширением, он предоставляет все функции Microsoft SQL Server, включая управление доступом, тиражирование баз данных и резервное копирование.

Поддержка SQL-клиентов

Архитектура клиент-сервер архиватора IndustrialSQL Server поддерживает выполнение клиентских приложений на настольных компьютерах, обеспечевая при этом целостность и безопасность данных на сервере. Архитектура клиент-сервер предоставляет общий доступ к производственной и технологической информации: к данным реального времени и историческим данным, к соответствующим конфигурационным данным, событиям и административным данным. Чтобы улучшить производительность системы, клиентские приложения и сервер эксплуатируются в режиме оптимизации за счёт переноса интенсивных процессорных операций на сервер и минимизации передаваемых по сети данных.

Доступ к информации в архиваторе IndustrialSQL Server осуществляется только с использованием Microsoft SQL Server. Таким образом, любые клиентские приложения, которые могут связаться с Microsoft SQL Server, могут обмениваться с ним данными.

Обмениваться с архиватором информацией могут две категории клиентские приложений.

- Клиенты, специально разработанные для доступа к данным архиватора. Wonderware предоставляет целый ряд клиентских средств по представление и анализу данных. Также имеются приложения сторонних разработчиков, взаимодействующие с архиватором IndustrialSQL Server. Эти средства не требуют знания языка SQL, и поиск нужной информации и её анализ, построение графиков оперативных и архивных данных выполняется в них с помощью наглядного интерфейса.
- Любые средства построения запросов к SQL- и ODBC-источникам данных. В настоящее время на рынке присутствует множество средств генерации запросов и создания отчётов, обеспечивающих доступ к базам данных SQL. Взаимодействовать с архиватором IndustrialSQL Server могут любые клиентские средства, имеющие интерфейс к Microsoft SQL Server или ODBC-интерфейс.

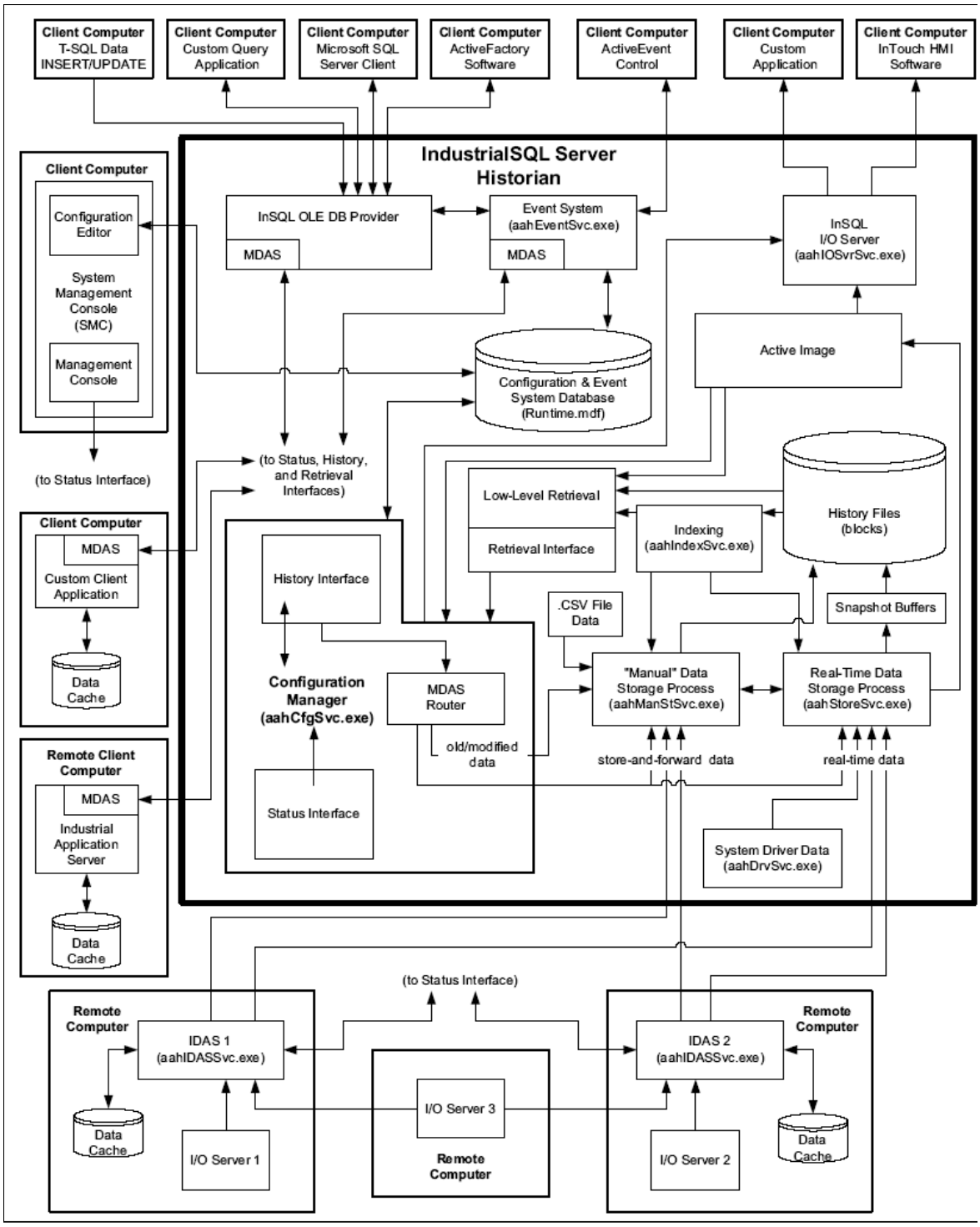
Подархиватора IndustrialSQL Server

Архиватор IndustrialSQL Server был составлен из специализированных подсистем, совместно используемых для управления потоками данных при их генерации или накоплении, сохранении и извлечении: К подсистемам относятся следующие.

- Подсистема конфигурирования.
- Подсистема накопления данных.
- Подсистема сохранения данных.
- Подсистема извлечения данных.
- Подсистема событий.

Последующие главы содержат более подробное описание каждой из подсистем.

Общая архитектура архиватора следующая:



ГЛАВА 2

Функциональность уровня системы

Некоторые концептуальные возможности, такие как обработка времени, параметры системы, безопасность и качество данных, можно использовать в архиваторе IndustrialSQL Server везде.

Содержание

О тэгах

Тэги представляет собой неделимые единицы хранения информации в архиваторе IndustrialSQL Server. Тэгом является некоторая переменная, которая характеризует некоторый атрибут производственного процесса. Тэг в пределах архиватора имеет уникальное имя. Тэг имеет набор атрибутов, таких как тип (например аналоговый), способ получения значений, способ сохранения значений (циклический или по изменению) и т.д.

Типы тэгов

Типы поддерживаемых архиватором тэгов перечислены в следующей таблице.

Тип	Описание
Аналоговый (Analog)	Аналоговая величина является переменной со значением некоторого физического параметра. Например, температура воды в водогрейном котле может измеряться как аналоговая величина.
Логический (Discrete)	Логическое значение является переменной, которая может иметь два состояния: '1' (True, 'On') или '0' ('False', 'Off').

Строковый (String)	Строковая величина является текстовым выражением, рассматриваемым как отднльный элемент данных. Строка не требует какого-либо специального формата или синтаксиса.
Событийный (Event)	Тэг события является именем определения события в системе. Например, если вы хотите регистрировать достижение в накопителе температуры 100ºC, вы можете определить тэг события и назвать его "TankAt100".
Составной (Complex)	Составной тэг содержит данные, которые не могут быть отнесены ни к одному из четырёх основных типов: аналоговому, логическому, строковому или событийному. Составные переменные могут содержать двоичные объекты, такие как массивы, .AVI-файлы, растровые изображения и т.д. В настоящее время этот тип не поддерживается.

Источники значений тэгов

Источниками значений являются следующие:

- Автоматически накапливаемые данные от серверов в/в, либо оперативные, либо запоздалые.
- Данные серверов в/в, поступающие с задержкой из кэша IDAS-источника "сохранить и направить".
- Данные, записываемые системой во внутренние тэги мониторинга состояния.
- Данные, вставляемые или обновляемые с помощью операторов Transact-SQL.
- Данные, извлекаемые из соответствующим образом отформатированных CSV-файлов при их импортировании.
- Данные, поступающие от клиентских приложений, разработанных с помощью инструментального пакета разработки приложений (IndustrialSQL Server Software Development Kit – SDK).
- Данные из приложений ArchestrA.

Правила именования тэгов

Имена тэгов могут содержать любые печатные символы ASCII, за исключением символов вертикальной черты '|', открывающей квадратной скобки '[' и закрывающей квадратной скобки ']'. Настоятельно рекомендуем вам при обозначении тэгов соблюдать правила именования, принятые в рамках Microsoft SQL Server.

Первым символом "стандартных" тэгов может быть:

- буква;
- цифра, при условии что следующим символом будет буква, например 2ETV433ET;
- символ доллара "$" или фунта "#";

- Остальными символами могут быть:
  - буква и цифра;
  - символ подчерка "_", доллара "$", фунта "#", процента "%" или обратной наклонной черты "\";

Вследствие требований подсистемы сохранения, в конце имени тэга не может стоять ни символ двойной кавычки ("), ни символ одинарной кавычки (').

Тэги, имена которых не удовлетворяют приведённым требованиям, рассматриваются как "нестандартные".

В SQL-запросах к расширенным таблицам имена "нестандартных" тэгов нужно заключать в квадратные скобки ([]), так как указанные имена используются как имена столбцов. Подробнее см. раздел "Использование нестандартных тэгов в запросах к расширенным таблицам" настоящего руководства.

Контроль доступа

В системе архиватора IndustrialSQL Server имеется два механизма контроля доступа:

- Контроль доступа операционной системы Windows.
- Контроль доступа Microsoft SQL Server.

Чтобы клиенты могли обращаться к архиватору, они должны пройти оба этих уровня идентификации пользователей. Консоль управления архиватором (в составе системной консоли управления – System Management Console) осуществляет дополнительную проверку прав выполнения тех функций, которые влияют на состояние архиватора, таких как запуск и остановка системы. Некоторые компоненты архиватора также требуют регистраци Windows и Microsoft SQL Server.

Внимание! По умолчанию всем пользователям систем Windows 2000 Professional и Windows XP предоставляется право выключения локального компьютера. Это значит, что пользователи этих систем автоматически получают права доступа к архиватору "уровня 2", поскольку пользователи архиватора именно этого уровня имеют право выключения локального компьютера. Чтобы пользователи этих операционных систем не могли запускать или останавливать приложение архиватора, нужно запретить в локальной политике безопасности Windows право остановки системы. Подробнее об определении прав пользователей в локальных политиках безопасности см. в соответствующей документации Microsoft.

Подробнее о контроле доступа см. Главу 7 "Контроль доступа" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Контроль доступа операционной системы Windows

Чтобы зарегистрироваться как клиент в архиваторе IndustrialSQL Server, первое, что должен быть в состоянии сделать пользователь, это зарегистрироваться в операционной системе своего компьютера. Принятые в Windows параметры регистрации представляют собой комбинацию регистрационного идентификатора (имени пользователя) и пароля доступа. Эта учётная запись пользователя может также использоваться для получения доступа к сетевым ресурсам домена.

Microsoft SQL Server также требует авторизации полномочий при подключении к нему. Вы можете использовать проверку учётных данных в Windows или Microsoft SQL Server. Подробнее см. в разделе "Контроль доступа SQL".

Учётные данные Windows по умолчанию для служб архиватора IndustrialSQL Server

Все модули архиватора IndustrialSQL Server, кроме консоли управления (Management Console) и редактора конфигурации (Configuration Editor), исполняются как службы Windows и поэтому требуют допустимой учётной записи пользователя Windows (учётной записи административного пользователя ArchestrA). Эта учётная запись задаётся на этапе установки.

Пользователь ArchestrA должен быть членом локальной административной группы на сервере архиватора, так же как и на всех компьютерах, содержащих удалённые IDAS-источники.

Вы можете изменить параметры учётной записи пользователя ArchestrA, используя утилиту изменения параметров регистрации в сети ArchestrA (ArchestrA Change Network Account Utility).

Внимание! Изменение этих параметров влияет на все запущенные как службы компоненты ArchestrA, а не только на службы архиватора. Чтобы изменения вступили в силу, нужно перезагрузить компьютер.

Не рекомендуется указывать запуск служб архиватора (включая удалённые IDAS-источники) с учётными параметрами конкретного пользователя. По умолчанию все службы должны конфигурироваться на регистрацию в операционной системе Windows с помощью учётных данных локальной системы. Действия служб архиватора будут представлять учётную запись пользователя ArchestrA, и эта персонификация приведёт к отказу, когда такая служба не будет запущена с локальными системными правами.

Контроль доступа SQL Server

Поскольку архиватор IndustrialSQL Server имеет встроенный Microsoft SQL Server, он использует все его функции защиты данных. Задача такой защиты заключается в том, чтобы контролировать доступ к Microsoft SQL Server и его базам данных, а также управлять выполнением тех или иных операций с данными.

Чтобы обратиться к архиватору, пользователь базы данных должен пройти две стадии проверки:

- Идентификацию, при которой проверяются права доступа к самому серверу.
- Авторизацию доступа к базе(ам) данных, во время которой определяется набор действий, которые может производить пользователь над объектами базы данных.

Идентификация и авторизация в к базе данных может осуществляться с помощью Microsoft SQL Server Enterprise Manager.

Чтобы получить доступ к информации в базах данных архиватора IndustrialSQL Server, пользователь должен иметь соответствующие права доступа к этим базам. Комплект поставки архиватора содержит набор уже определённых учётных записей и ролей, которые могут служить в качестве основы для дальнейшего развития на предприятии системы контроля доступа к данным. Управление списками ролей, параметров регистрации в Microsoft SQL Server и учётными записями также выполняется с помощью Microsoft SQL Server Enterprise Manager.

Идентификация

При проверке в Microsoft SQL Server полномочий пользователя сервер на основе учётной информации определяет личность пользователя и возможность его подключения к данному экземпляру сервера. Если процедура проверки полномочий завершается успешно, пользователю

разрешается подключитья к Microsoft SQL Server. Имеется два типа проверки полномочий:

- Идентификация Windows, при которой пользователь для подключения к Microsoft SQL Server должен указть параметры своей регистрации в операционной системе (имя пользователя и пароль, которые указываются во время регистрации в Windows).
- Идентификация SQL Server, при которой пользователь для подключения должен указать параметры своей регистрации на сервере (регистрационный идентификатор и пароль).

SQL Server может функционировать в одном из двух режимов проверки полномочий, которые определяют тип учётных данных, которые должны использоваться для доступа к серверу:

- Режим идентификации Windows. В этом режиме Microsoft SQL Server выполняет проверку только параметров регистрации Windows.
- Смешанный режим. В этом режиме используется как механизм проверки полномочий Windows, так и механизм SQL Server. Если регистрационный идентификатор совпадает с именем пользователя в сети Windows, проверка выполняется процедурами проверки полномочий Windows. Если идентификатор не совпадает, используются механизмы Microsoft SQL Server.

Средства проверки полномочий Microsoft SQL Server предназначены исключительно для обеспечения обратной совместимости. Компания Microsoft рекомендует по возможности использовать средства Windows.

Подробнее о проверке полномочий пользователя см. в документации по Microsoft SQL Server.

Ггруппы безопасности Windows по умолчанию

По умолчанию на компьютере архиватора IndustrialSQL Server создаются следующие группы пользователей Windows, для каждой из которых определены различные наборы полномочий:

- aaAdministrators;
- aaPowerUsers;
- aaUsers.

Каждая из групп автоматически сопоставляется с ролью в базе данных Microsoft SQL Server с таким же именем. Например, группа пользователей aaAdministrators является членом группы по умолчанию пользователей aaAdministrators Microsoft SQL Server с ролью администраторов базы данных. При добавлении в группу aaAdministrators новых пользователей они автоматически получать все полномочия для роли администратора Microsoft SQL Server.

Параметры регистрации по умолчанию архиватора IndustrialSQL Server

При установке архиватора IndustrialSQL Server в системе автоматически создаются учётные записи по умолчанию, данные которых могут указываться клиентскими приложениями при соединении с архиватором. Эти учётные записи по умолчанию предоставляют "нестандартную" функциональность, при которой вам не нужно создавать учётные записи, чтобы начать пользоваться системой. Предустановленные учётные записи перечислены в таблице.

Регистрационное	Пароль	Описание

имя		
aaAdmin	pwAdmin	Пользователь, который имеет доступ ко всем данным и объектам базы данных и может выполнять любые действия, за исключением удаления базы данных и усечения таблиц.
aaPower	pwPower	Пользователь, который имеет право чтения любой информации в базе данных, а также права создания новых объектов и изменения содержимого в неосновных таблицах.
aaUser	pwUser	Пользователь, который может только читать, но не может изменять данные и потреблять ресурсы базы данных.
aadbo	pwddbo	Владелец базы данных. Имеет полный доступ.

Базой данных по умолчанию для каждой из этих групп является Рабочая база архиватора. Указанная модель по умолчанию защиты данных является отправной точкой разработки защиты данных предприятия, пригодной для многих типов конфигураций.

Эти же учётные записи могут использоваться и при выборе смешанного режима проверки полномочий в Microsoft SQL Server. Если используется только режим Windows, нужно указать права доступа для каждого пользователя.

Внимание! Никогда не используйте для регистрации пустые пароли.

Следующие учётные записи предоставляются исключительно для обеспечения обратной совместимости.

Регистрационное имя	Пароль	Описание
wwUser	wwUser	То же, что и aaUser
wwPower	wwPower	То же, что и aaPower
wwAdmin	wwAdmin	То же, что и aaAdmin
wwdbo	wwdbo (только для SQL Server 2000) либо pwddbo (только для SQL Server старших версий)	То же, что и aadbo. Пароль wwdbo действителен только в системе SQL Server 2000. Пароль pwddbo используется в SQL Server старших версий.

Авторизация в базе данных

После того как пользователь успешно соединится с Microsoft SQL Server, он должен пройти проверку полномочий в базе данных сервера на выполнение требуемых действий. Проверка выполняется с использованием учётной информации, указанной в каждой базе данных. Учётные сведения представляют собой сочетание имени пользователя и пароля доступа. Каждая учётная запись пользователя должна быть сопоставлена с существующим регистрационным идентификатором.

В каждой базы данных имена пользователей содержатся в таблицах sysusers. Если пользователь обратщается к данным, Microsoft SQL Server ищет соответствующую строку в таблице sysusers и затем в таблице

sysxlogins в главной базе данных. При отсутствии для указанного имени пользователя необходимой информации пользователю в доступе к базе данных будет отказано.

Полномочия, заданные для учётной записи пользователя, определяют виды действий в базе данных, которые может выполнять пользователь. Условно эти виды можно разделить на две категории: раьота с объектами базы данных и исполнение операторов.

Действия	Описание
Работа с объектами	Действия, которые может производить пользователь над имеющимися в базе данных объектами, такими как таблицы, индексы, представления, значения по умолчанию, триггеры, правила и процедуры. Права на выполнение тех или иных действий с объектами базы данных предоставляет и аннулирует владелец объекта, то есть пользователь, который создал данный объект.
Исполнение операторов	Права на исполнение операторов Transact-SQL, таких как SELECT, INSERT или DELETE. Права на исполнение операторов предоставляются и аннулируются только системными администраторами и владельцами баз данных.

Пользователей можно группировать по "ролям", то есть в отдельные группы, которые вы можете наделить определёнными полномочиями. Наделение роли каким-либо правом, его снятие или аннулирование означает автоматическое распространение прав ан соответствующие действия на всех пользователей, назначенных на данную роль.

Пользователи и роли по умолчанию в системе архиватора IndustrialSQL Server

Архиватор IndustrialSQL Server поставляется с уже имеющимися учётными записями пользователей и ролями.

Следующая таблица содержит стандартные имена пользователей архиватора IndustrialSQL Server, регистрационные идентификаторы и роли, которым они присвоены, а также действия, которые им разрешается выполнять в Рабочей базе данных. Вы можете определить новых пользователей и роли, используя SQL Sever Enterprise Manager.

Регистрационный идентификатор	Имя пользователя в базе данных	Роль	Права
aaUser	aaUser	aaUsers	SELECT (все таблицы) INSERT, UPDATE, DELETE (таблицы PrivateNameSpace и Annotation)
aaPower	aaPower	aaPowerUsers	CREATE Table CREATE View CREATE Stored procedure CREATE Default CREATE Rule SELECT (все таблицы) INSERT, UPDATE, DELETE (в таблицах группирования)
aaAdmin	aaAdmin	aaAdministrators	CREATE Table CREATE View

			CREATE Stored procedure CREATE Default CREATE Rule DUMP Database DUMP Transaction SELECT, INSERT, UPDATE, DELETE (все таблицы)
aadbo	dbo	db_owner	Любые операции с любыми объектами базы данных и с самой базой данных

Следующие пользователи и роли включены исключительно для обратной совместимости:

Регистрационный идентификатор	Имя пользователя в базе данных	Роль	Права
wwUser	wwUser	wwUsers	Аналогичны правам группы aaUser
wwPower	wwPower	wwPowerUsers	Аналогичны правам группы aaPower
wwAdmin	wwAdmin	wwAdministrators	Аналогичны правам группы aaAdmin
wwdbo	wwdbo	db_owner	Аналогичны правам группы aadbo

Каждая роль по умолчанию имеет соответствующую учётную запись Microsoft SQL Server, также как и каждая группа пользователей Windows по умолчанию.

Регистрационные параметры по умолчанию служб архиватора IndustrialSQL Server

Некоторым компонентам архиватора IndustrialSQL Server для доступа к главной, Рабочей и Промежуточной базам данных требуется регистрация в Microsoft SQL Server. По умолчанию для регистрации на сервере Microsoft SQL Server с идентификацией Windows архиватор использует учётные записи ArchestrA.

Если учётная запись Windows является учётной записью администратора локального компьютера, Microsoft SQL Server сопоставляет с ней фиксированную роль sysadmin. (Этот пользователь будет иметь те же полномочия, что и пользователю Microsoft SQL Server с идентификатором "sa".) Так как учётные записи ArchestrA всегда являются записями локальных администраторов, в Microsoft SQL Server эти пользователи всегда обладают полномочиями администратора.

Управление полномочиями консоли

Консоль управления (Management Console) архиватора IndustrialSQL Server (которая является частью общей системной консоли управления – System Management Console) исполняется в контексте зарегистрированного пользователя Windows. Чтобы предотвратить несанкционированный доступ к архиватору IndustrialSQL Server, для консоли управления нужно определить отдельную учётную запись Windows, которая будет использоваться при установлении соединения с архиватором. Эти данные можно указать при конфигурировании параметров регистрации сервера.

Подробнее см. раздел "Регистрация сервера IndustrialSQL Server" Главы 1 *Руководства по администрированию архиватора IndustrialSQL Server™*.

Если указаны сведения, не являющиеся учётной записью пользователя группы локальных администраторов компьютера, на котором исполняется приложение архиватора, консоли управления будут предоставлены права "только чтение". В этом случае пользователь сможет только просматривать данные в окне консоли управления и не сможет выполнять никаких команд архиватора, таких как запуск или остановка системы, создание новых архивных блоков и т.д.

Внимание! Чтобы предотвратить несанкционированный доступ, пароль консоли управления НЕ должен быть пустым.

Обработка меток времени

Метки времени всех данных записываются в формате UTC (Universal Coordinated Time – Единое координатное время), известное также как гринвичское время (GMT – Greenwich Mean Time). Текущее UTC-время определяется как местным временем часовым поясом, установленным в операционной системе компьютера, на котором исполняется архиватор IndustrialSQL Server. По умолчанию при чтении данных метки времени устанавливаются по местному времени. Вы можете преобразовать метки времени, так чтобы они выводились в локальном времени с использованием специального параметра запроса.

Вы должны использовать показания времени в запросах в международном формате даты и времени суток:

ГГГГММДД чч:мм:сс.ххх,

где:

ГГГГ – год;

ММ – месяц;

ДД – день месяца;

чч – час;

мм – минута;

сс – секунда;

ххх – миллисекунда.

Формат возвращаемых запросами меток времени определяется языковыми настройками операционной системы.

Если используется несколько архиваторов и/или удалённых источников данных, очень важно синхронизировать часы компьютеров. Подробнее см. раздел "Синхронизация часов различных компонентов процесса накопления данных" в Главе 4 настоящего руководства.

Системные параметры

Системный параметр представляет собой величину, которая контролирует некоторый аспект общего поведения архиватора IndustrialSQL Server. Системные параметры по умолчанию перечислены в следующей таблице.

Параметр	Назначение
AIAutoResize	Управляет автоматическим изменением размера Активного образа: 1 = изменение разрешено, 0 = изменение запрещено. Значение по умолчанию – 1. Подробнее см. раздел "Активный образ" настоящего

	руководства.
AIResizeInterval	Интервал в минутах, через который система выполняет изменение размера Активного образа, если оно разрешено значением параметра AIAutoResize.
AllowOriginals	Разрешает вставку первоначальных значений для тэгов сервера в/в. Подробнее см. раздел "Накопление данных с помощью операторов INSERT и UPDATE" настоящего руководства.
AutoStart	Управляет автоматическим запуском архиватора при включении компьютера: 1 = запуск разрешён, 0 = запуск запрещён. Подробнее см. раздел "Автоматический запуск архиватора IndustrialSQL Server" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*. После того как значение этого параметра будет изменено, нужно выполнить фиксацию изменений в системе.
ConfigEditorVersion	(Не редактируемый.) Младшая версия редактора конфигурации (Configuration Editor), с помощью которго можно редактировать Рабочую базу данных. Используется системой для внутренних целей.
DatabaseVersion	(Не редактируемый.) Текущий номер версии базы данных.
DataImportPath	Путь доступа каталогу с CSV-файлом для импортирования или экспортирования данных. Подробнее см. раздел "Накопление данных из CSV-файлов" настоящего руководства. После изменения значения этого параметра нужно выполнить перезапуск архиватора.
EventStorageDuration	Максимальный срок в часах хранения записи о событии в таблице EventHistory.
HeadroomXXXX	Количество тэгов, для которых предварительно распределяется память. Подробнее см. раздел "Выделение памяти под будущие тэги" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.
HistorianVersion	(Не редактируемый.) Текущая версия и номер редакции архиватора IndustrialSQL Server. Значение этого параметра выводится на экран в момент запуска системы.
HistoryCacheSize	Объём системной памяти в мегабайтах, выделяемой для хранения информации о тэгах. Значение по умолчанию – 0. Подробнее см. раздел "Управление памятью" настоящего руководства. После того как значение этого параметра будет изменено, нужно зафиксировать изменения в системе и выполнить повторное сканирование архивных блоков для записи данных на диск.
HistoryDaysAlwaysCached	Период времени в днях, данные за который из архивного блока всегда должны быть загружены в память. Значение по умолчанию – 0.
HoursPerBlock	Временной интервал данных в архивном блоке. Допустимые значения 1, 2, 3, 4, 6, 8, 12, 24. Значение по умолчанию – 24. Эта величина всегда должна превышать значение максимального периода сканирования. Подробнее см. раздел "Архивные блоки" настоящего руководства.
InterpolationTypeInteger	Вид интерполяции данных целого типа: 0 = последнее

	значение, 1 = линейная интерполяция. Значение по умолчанию – 0. Подробнее см. раздел "Аргумент wwInterpolationType" настоящего руководства.
InterpolationTypeReal	Вид интерполяции данных вещественного типа: 0 = последнее значение, 1 = линейная интерполяция. Значение по умолчанию – 1.
LicenseRemoteIDASCount	(Не редактируемый.) Количество удалённых источников данных, с которыми может взаимодействовать архиватор. Это количество берётся из лицензионного файла и используется системой для внутренних целей.
LicenseTagCount	(Не редактируемый.) Максимальное количество тэгов, значения которых могут накаливаться архиватором. Это количество берётся из лицензионного файла и используется системой для внутренних целей.
ModLogTrackingStatus	Управление контролем изменений. Задаваемое значение определяет способ управления изменениями. Подробнее см. раздел "Контроль изменений" в Главе 8 *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.
QualityRule	Режим качества используемых данных: 0 = должны использоваться данные с качеством "GOOD" (достоверные) и "UNCERTAIN" (неопределённые), 1 = должны использоваться только данные с качеством "GOOD" (достоверные). Подробнее см. раздел "Аргумент wwQualityRule" настоящего руководства.
RealTimeWindow	Максимальная задержка в миллисекундах, в течение которой при расчёте порога изменения данные рассматриваются как оперативные. Задержка задаётся по отношению к текущему времени. Допустимыми являются значения: от –30 до +999 мс. Значение по умолчанию – 60. Подробнее см. раздел "Окно оперативных данных" настоящего руководства.
SuiteLinkTimeSyncInterval	Длительность интервалов времени в минутах для периодичности, с которой IDAS-источники должны синхронизировать режим меток времени с соответствующими серверами в/в. Если 0, – синхронизация не выполняется. Подробнее см. раздел "Синхронизация часов различных компонентов процесса накопления данных" настоящего руководства.
SummaryStorageDuration	Максимальный срок в часах хранения записей с результатами сводных операций в таблице SummaryHistory.
SysPerfTags	Управление использованием системных тэгов мониторинга производительности в IndustrialSQL Server: 0 = не использовать, 1 = использовать. Значение по умолчанию – 1. Подробнее см. найти в разделе "Тэги мониторинга производительности".
TimeStampRule	Устанавливает, с какой меткой времени цикла чтения должны возвращаться данные: 0 = с меткой времени начала цикла, 1 = с меткой времени конца цикла. Значение по умолчанию – 1. Подробнее см. раздел "Аргумент wwTimeStampRule".
TimeSyncIODrivers	Если разрешён, архиватор IndustrialSQL Server будет рассылать команды синхронизации времени всем удалённым IDAS-источникам. Подробнее см. раздел

	"Синхронизация часов различных компонентов процесса накопления данных" настоящего руководства.
TimeSyncMaster	Имя компьютера, часы которого будут использоваться архиватором IndustrialSQL Server в качестве эталонных для синхронизации времени.

Системные сообщения

К системным сообщениям относятся сообщения об ошибках, а также сообщения о состоянии архиватора IndustrialSQL Server в целом и о состоянии его подсистем и процессов. Системные сообщения сохраняются:

- В журнале ArchestrA Logger.
- В журнале событий Windows, который можно просматривать с помощью Windows Event Viewer. Этот журнал регистрирует не все сообщения. Обычно в него записываются только сведения о действиях пользователя и исключительных ситуациях. Сообщения записываются с указанием в качестве имени источника строки "Historian" или имени службы архиватора IndustrialSQL Server.

Примечание. В более раниих версиях архиватора, чем 9.0, сообщения об ошибках записывались в файлы с расширением .ier. После обновления архиватора IndustrialSQL Server до версии 9.0 с его помощью можно просматривать все имеющиеся журналы системных ошибок.

Системные сообщения делятся на следующие категории:

Категория	Описание
FATAL (Фатальные)	Выполнение процесса невозможно. Ошибки такого рода приводят к остановке работы системы.
CRITICAL (Критические)	Ошибки такого рода приводят к неправильному функционированию подсистем хранения или извлечения, например к потере или искажению данных.
ERROR (Некритические)	Ошибки общего вида, например ошибки проверки памяти во время запуска системы. Такиеошибки могут повлечь за собой упорядоченный останов системы, но в большинстве случаев они не препятствуют выполнению системой своих операций.
WARNING (Предупреждение)	Сообщения, уведомляющие оператора о присвоении параметрам значений или о возникновении событий. Так, невозможность вызова динамически компонуемой процедуры, выполняющей необязательную функцию, будет зарегистрирована как предупреждение.
INFO (Информационные)	Сообщения о ходе запуска системы или сохранении данных.
DEBUG (Отладочные)	Отладочные сообщения, которые, как правило, в официальных версиях системы отсутствуют.

Сообщения архиватора IndustrialSQL Server регистрируются в Log Viewer следующим образом:

- Фатальные, критические и некритические регистрируются как сообщения, начинающиеся со слова "Error". Этому слову может

предшествовать описание типа ошибки: "FATAL", "CRITICAL", "ERROR".

- Предупреждения будут начинаться со слова "Warning" без префикса.
- Информационные сообщения будут начинаться со слова "Info".
- Отладочные сообщения будут начинаться со слова "Trace".

Подробнее о мониторинге состояния системы см. Главу 9 "Мониторинг системы" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Службы архиватора IndustrialSQL Server

Следующие процессы архиватора IndustrialSQL Server запускаться как службы Windows:

Наименование службы	Имя исполняемого файла	Описание
InSQLConfiguration	aahCfgSvc.exe	Выполнение всех действий с конфигурационными параметрами, а также поддержка интерфейсов ручного ввода и извлечения данных. Подробнее см. Главу 3 "Подсистема конфигурирования" настоящего руководства.
InSQLDataAcquisition	aahIDASSvc.exe	Накопление данных от локальных и удалённых источников данных и передача их в подсистему сохранения. Подробнее см. Главу 4 "Подсистема накопления данных".
InSQLEventSystem	aahEventSvc.exe	Анализ архивных данных и определение условий возникновения событий. Подробнее см. Главу 7 "Подсистема событий" настоящего руководства.
InSQLIndexing	aahIndexSvc.exe	Индексирование архивных данных на диске. Подробнее см. Главу 5 "Подсистема хранения данных" настоящего руководства.
InSQLIOServer	aahIOSvrSvc.exe	Передача оперативных данных из архиватора сетевым клиентам. Подробнее см. Главу 6 "Подсистема извлечения данных" настоящего руководства.
InSQLManualStorage	aahManStSvc.exe	Служба ввода неоперативной информации и сохранение её на диске. Подробнее см. Главу 5 "Подсистема хранения данных" настоящего руководства.
InSQLRetrieval	aahRetSvc.exe	Извлечение данных из архива. Подробнее см. Главу 6 "Подсистема извлечения данных" настоящего руководства.
InSQLSCM	aahSCM.exe	Предоставление информации о состоянии архиватора. Относится к внутренним службам архиватора. Этот

		процесс исполняется постоянно, даже если работа архиватора остановлена.
InSQLStorage	aahStoreSvc.exe	Приём производственных данных реального времени и запись их на диск. Подробнее см. Главу 5 "Подсистема хранения данных" настоящего руководства.
InSQLSystemDriver	aahDrvSvc.exe	Генерация значений различных тэгов мониторинга системы. Подробнее см. в следующем разделе.

Подробнее о службах операционной системы Windows см. в соответствующей документации Microsoft.

Системный драйвер и системные тэги

Системный драйвер представляет собой внутренний процесс, который выполняет мониторинг основных параметров работающего архиватора IndustrialSQL Server и отображает их в виде значений заданного набора системных тэгов. Системный драйвер исполняется как служба Windows, запускаемая автоматически при запуске подсистемы хранения.

Требуемые системные тэги создаются при установке архиватора IndustrialSQL Server. Кроме того, дополнительные тэги создаются также для каждого имеющегося IDAS-источника.

Текущие значения аналоговых системных тэгов регулярно, с интервалом заданного в миллисекундах промежутка времени, отсылаются в подсистему хранения. Значения тэгов времени и даты отражают местное время архиватора.

Тэги счётчиков ошибок

Следующие аналоговые тэги сохраняются с частотой один раз в минуту (60000 мс). Все счётчики подсчитывают количество ошибок, возникших с момента запуска архиватора IndustrialSQL Server или с момента последней операции очистки счётчиков.

Имя тэга	Значение
SysCritErrCnt	Количество критических ошибок
SysErrErrCnt	Количество нефатальных ошибок
SysFatalErrCnt	Количество фатальных ошибок
SysWarnErrCnt	Количество предупреждений

Тэги даты

Следующие тэги сохраняются с периодичностью в пять минут (300000 мс).

Имя тэга	Значение
SysDateDay	День месяц
SysDateMonth	Месяц год
SysDateYear	Год (все 4 цифры)

Тэги времени

Все следующие тэги являются аналоговыми. Сохраняется изменение каждого из этих тэгов (сохранение по изменению).

Имя тэга	Значение
SysTimeHour	Час дня
SysTimeMin	Минута часа
SysTimeSec	Секунда

Тэги доступного пространства

Следующие тэги сохраняются с периодичностью в пять минут (300000 мс). Оставшееся пространство измеряется в мегабайтах.

Имя тэга	Значение
SysSpaceAlt	Объём доступного пространства в дополнительной области хранения.
SysSpaceBuffer	Объём доступного пространства в буферной области хранения.
SysSpaceMain	Объём доступного пространства в основной области хранения.
SysSpacePerm	Объём доступного пространства в постоянной области хранения.

Тэги статистики в/в

Следующие аналоговые тэги могут использоваться для контроля основных параметров в/в.

Имя тэга	Значение
SysDataAcqNBadValues*	Количество полученных данных с плохим качеством. Значение этого тэга сохраняется с периодичностью в 5 секунд. Максимальное значение – 1000000.
SysDataAcq*N*OutsideRealtime*	Количество значений в секунду, которые были игнорироваться в связи с задержкой, превышающей ширину окна оперативных данных. Значение этого тэга сохраняется с периодичностью в 5 секунд. Максимальное значение – 1000000.
SysDataAcqOverallItemsPerSec	Количество элементов данных, полученных от всех источников. Значение этого тэга сохраняется с периодичностью в 10 секунд. Максимальное значение – 100000.
SysDataAcqRxItemPerSec*N**	Количество обновлений тэгов в секунду. Значение этого тэга сохраняется с периодичностью в 10000 миллисекунд. Для данного IDAS-источника обновляется каждые 2 секунды.
SysDataAcqRxTotalItems*N**	Общее количество обновлений тэга для данного IDAS-источника, полученных с момента последнего запуска. Периодичность сохранения значений –

	10000 мс.
SysStatusRxItemsPerSec	Количество обновлений тэгов в секунду. Обновляется каждые 2 секунды для системного драйвера (aahDrvSvc.exe). Значение этого тэга сохраняется с периодичностью в 1000 миллисекунд.
SysStatusRxTotalItems	Общее количество обновлений тэга для системного драйвера, полученных с момента последнего запуска. Периодичность сохранения значений – 10000 мс.

* Такой тэг статуса имеется для каждого заданного в системе IDAS-источника. Идентификационный номер (*N*) в имени тэга является значением **IODriverKey** из таблицы **IODriver**. Число "0" обозначает MDAS и применяется только для тэгов SysDataAcqNBadValues и SysDataAcq*N*OutsideRealtime.

Тэги мониторинга системы

Если не указано иное, для последующих логических тэгов 0 = плохо (Bad), 1 = хорошо (Good).

Имя тэга	Значение
SysConfiguration	Состояние службы конфигурирования (aahCfgSvc.exe). Параметр получает значение 1, когда появляется необходимостьь динамического конфигурирования или когда оно выполняется.
SysDataAcqN*	Состояние службы IDAS (aahIDASSvc.exe).
SysEventSystem	Состояние службы подсистемы событий (aahEventSvc.exe).
SysIndexing	Состояние службы индексации (aahIndexSvc.exe).
SysInSQLIOS	Состояние сервера в/в InSQL (aahIOSrvSvc.exe).
SysManualStorage	Состояние службы ручного сохранения (aahManStSvc.exe).
SysOLEDB	Состояние службы провайдера OLEDB (загружаемого сервером SQL Server).
SysPulse	Логический тэг ("пульс"), значение которого изменяется каждую минуту.
SysRetrieval	Состояние службы извлечения данных (aahRetSvc.exe).
SysStorage	Состояние службы сохранения данных (aahStoreSvc.exe).
SysSystemDriver	Состояние системного драйвера (aahDrvSvc.exe).

* – подобный тэг статуса существует для каждого определённого в системе источника данных. Такой тэг статуса имеется для каждого заданного в системе IDAS-источника данных. Идентификационный номер (*N*), добавленный в конец тэга, является ключом **IODriverKey** из таблицы **IODriver**.

Смешанный (прочие) тэги

Имя тэга	Значение

SysConfigStatus	Количество баз данных, задействованных в динамическом конфигурировании (то есть количество строк в таблице ConfigStatusPending при выполнении команды фиксации изменений). Данная величина является кумулятивной и обнуляется только при перезапуске системы.
SysHeadroomXXX	Тэг, предназначенный для контроля числа "доступных" тэгов. Аналоговые тэги могут занимать 2, 4 или 8 байтов. Подробнее см. раздел "Выделение памяти под будущие тэги" Руководства о администрированию архиватора IndustrialSQL Server.
SysHistoryCaceFaults	Количество архивных блоков, загружаемых с диска в минуту. Максимальное значение – 1000. Периодичность сохранения значений данного аналогового тэга – 60 с. Подробнее об архивной кэш-памяти см. в разделе "Управление память" настоящего руководства.
SysHistoryCacheUsed	Количество байтов занятых сведениями архивного блока. Максимальное значение – 3000000000. Периодичность сохранения значений этого тэга – 30 с.
SysHistoryClients	Количество клиентских соединений с архиватором. Максимальное значение – 200. Периодичность сохранения значений этого тэга – 30 с.
SysMinutesRun	Количество минут, прошедших с момента запуска системы. Данная величина сохраняется каждые 60 с.
SysString	Строковый тэг, значение которого изменяется с периодичностью один раз час.
SysRateDeadbandForcedValues	Общее количество данных, которые были сохранены из-за превышения порога изменения значений. Данная величина отражает общее количество подобных операций, выполненных для всех тэгов, с момента запуска системы.
SysEventCritActionQSize	Размер очереди критических реакций. Подробнее см. раздел "Пул реакций" настоящего руководства.
SysEventDelayedActionQSize	Количество элементов в очереди задержанных реакций.
SysEventNormActionQSize	Размер очереди обычных реакций
SysEventSystem	Логический тэг, значение которого отражает состояние службы подсистемы событий (aahEventSvc.exe): 0 = неудовлетворительное, 1 = удовлетворительное.
SysStatusEvent	Событийный тэг "моментальной" копии,

	значение которого изменяется каждый час.

Тэги мониторинга производительности

Тэги мониторинга производительности системы позволяют контролировать загрузку центральных процессоров и другие параметры процессов архиватора IndustrialSQL Server. (Все эти величины отображаются на соответствующие счётчики, которые используются в приложении Microsoft Performance Logs and Alerts.)

Для мониторинга степени загрузки каждого центрального процессора (вплоть до четырёх), также как и для загрузки всех процессоров вы можете использовать следующие тэги:

- SysPerfCPU0;
- SysPerfCPU1;
- SysPerfCPU2;
- SysPerfCPU3;
- SysPerfTotal (для мониторинга общей загрузки всех процессоров).

Оставшиеся системные тэги используются для мониторинга производительности каждого процесса архиватора, исполняющегося как служба Windows, а также службы Microsoft SQL Server. Подробнее о службах см. раздел "Службы архиватора Microsoft SQL Server".

Имеется шесть системных тэгов, используемых для мониторинга каждой службы. Имена этих тэгов строго соответствуют приведенным ниже правилам, где символы "XXX" обозначают службу (Config, DataAcq, EventSys, Indexing, InSQLIOS, ManualStorage, Retrieval, SQLServer, Storage или SysDrv).

SysPerfXXXCPU,

SysPerfXXXHandleCount,

SysPerfXXXPageFaults,

SysPerfXXXPrivateBytes,

SysPerfXXXThreadCount,

SysPerfXXXVirtualBytes.

Значения этих тэгов сохраняются с периодичностью в пять секунд.

Примечание. Для каждого заданного источника данных также создаются шесть системных тэгов. Значение идентификационного номера (*N*), добавляемого в конец подстроки "DataAcq" в имени тэга, является значением столбца **IODriverKey** таблицы **IODriver**. Например 'SysPerfDataAcq1CPU'.

Следующая таблица представляет суффиксы имён тэгов.

Суффикс	Значение
CPU	Текущая загрузка процессора данной службой (в процентах).
HandleCount	Общее количество указателей, созданных каждой нитью (thread) службы. Указатель представляет собой идентификатор какого-либо ресурса системы, например ключа реестра или файла.
PageFaults	Количество возникновений в секунду ошибок обмена страницами памяти, возникающих в нитях службы. Ошибка обмена страницами возникает, когда нить обращается к

	странице виртуальной памяти, которая не входит в его набор страниц основной памяти. Это означает, что указанная страница не будет считываться с диска, если она входит в список ожидания (и, таким образом, уже находится в основной памяти) или используется другим процессом
PrivateBytes	Количество байтов памяти, занятых службой, доступ к которым со стороны других процессов запрещён.
ThreadCount	Текущее количество активных нитей службы. Нить представляют собой последовательность машинных команд, или основных единиц исполнения действий процессором.
VirtualBytes	Текущий размер в байтах виртуального адресного пространства данной службы.

Поддерживаемые протоколы

Архиватор IndustrialSQL Server поддерживает следующие протоколы:

Протокол	Описание
DDE	Протокол обмена динамическими данными (DDE) представляет собой механизм обмена информацией между приложениями, осуществляемый без участия или контроля со стороны пользователя. В Windows обмен динамическими данными реализован в виде набора сообщений различных типов, рекомендованных процедур (протоколов) их обработки и некоторых недавно введённых типов данных. Опираясь на эти протоколы, приложения, разработанные независимо друг от друга, могут обмениваться друг с другом информацией без вмешательства пользователя. Такими приложениями являются InTouch ИЧМ-интерфейс и Excel.
SuiteLink	Протокол, предоставляющий больше, чем DDE, возможностей передачи данных на прикладном уровне. В пакетах данных вместе с собственно информацией передаётся метка времени её генерации и параметры качества.
Системный протокол	Внутренний системный протокол передачи системных переменных в архиватор.

Примечание. Если архиватор работает в Windows Server 2003, DDE-протокол поддерживаться не будет.

Контроль изменений

Архиватор IndustrialSQL Server поддерживает контроль изменений (вставки, обновления, удаления) в столбцах таблиц Рабочей базы данных. Если вам нужно контролировать изменения вследствие требований со стороны регулирующих органов, вы можете настроить архиватор на использование такого контроля.

Контроль изменений распространятся на всю систему и разрешается или запрещается параметром ModLogTrackingStatus. Невозможно управлять контролем изменений на уровне отдельных таблиц. Включение контроля приводит к снижению быстродействия архиватора в случае внесения изменений в систему вследствие выполнения дополнительных операций и потребления дискового пространства, но на быстроту выполнения рабочих операций оно не сказывается.

Информация в таблицах контроля изменений хранится в виде файлов базы данных Microsoft SQL Server. При включении контроля изменений объём данных в этих файлах существенно возрастает.

Все объекты, для которых могут отслеживаться изменения состояния, перечислены в таблице HistorianSysObjects.

Архиватор позволяет контролировать изменения следующих двух типов:

- Изменения конфигурационных параметров. Нипример таких как добавление новых или изменение существующих тэгов, изменение серверов в/в, характеристик областей сохранения и т.д. Подробнее см. в следующем разделе";
- Изменения архивных данных. Нипример путём вставки или обновления с помощью операторов Transact-SQL или импортирования CSV-файлов. Подробнее см. раздел "Контроль изменений архивных данных".

Типы контролируемых изменений задаются значением системного параметра ModLogTrackingStatus. Подробнее см. раздел "Включение и отключение режима контроля изменений" в Главе 8 *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Контроль конфигурационных изменений

Когда измененяется некоторая таблица базы данных, в таблицу ModLogTable заносится соответствующая запись. Для каждого типа изменения создаётся отдельная строка, будь то вставка, обновление, удаление.

Конкретное значение изменения записывается в столбцах в таблицу ModLogColumn. Каждое изменение приводит к созданию новой строки, в которой указываются как предыдущее, так и новое значение в столбце.

Например, если мы добавляем (вставляем) в систему новый аналоговый тэг, все изменения будут зафиксированы в таблице следующим образом:

- В таблицу ModLogTable будут добавлены две строки, одна для регистрации изменений в таблице Tag, другая – в таблице AnalogTag.
- В таблицу ModLogColumn будут добавлены по одной строке для каждого из столбцов в таблицах Tag и AnalogTag.

Вот другой пример. Если вы обновляете для единичного аналогового тэга столбцы StorageType таблицы Tag и столбцы ValueDeadband и RateDeadband таблицы AnalogTag, в таблицы контроля изменений будут внесена следующая информация:

- В таблицу ModLogTable будут добавлены две строки, одна для регистрации изменений в таблице Tag, другая – в таблице AnalogTag;
- В таблице ModLogColumn будут добавлены три строки с информацией о чделанных изменениях значений в столбцах StorageType, ValueDeadband и RateDeadband.

Контроль изменений для архивных данных

Изменения архивных данных могут выполняются как с помощью операторов Transact-SQL, так и импортированием CSV-файлов. В случае использования Transact-SQL информацию об изменеиях для таблиц контроля изменений предоставляет провайдер OLEDB через вызов хранимой процедуры.Эта хранимая процедура используется подсистемой записи также для сообщения обизменениях, производимых при импортированиии CSV-файлов.

Хотя изменяемые данные физически находятся в архивных блоках на диске, подсистемой контроля изменений они рассматриваются как

находящиеся в таблице расширения с именем History_OLEDB. Подробнее о таблицах расширения см. в разделе "Таблицы расширения для хранения архивных данных" настоящего руководства.

Когда выполняются изменения архивных данных, информация о них записывается в таблицы ModLogTable. Ддля каждого типа изменения, такого как вставка или обновление, каждого тэга создаётся одна строка.

Для записи в таблицу History_OLEDB информации об изменениях значений столбцов используются данные изменений из таблицы ModLogColumn. Модифицируемым столбцом всегда будет столбец vValue. Количество последовательных изменений тэга записывается в столбец NewValue таблицы ModLogColumn.

Столбец OldValue содержит предыдущее значение столбца, если изменения выполнялись в таблице конфигурации. Если изменение данных выполнялось с помощью оператора SQL INSERT или UPDATE, в этот столбце записывается метка времени данных, которые первыми изменялись этими операторами. Если одни и те же данные изменялись несколько раз, сохранена будет метка времени только последнего изменения. При измененииии данных при импортировании CSV-файлов этот столбец не используется.

Например, ели в архив заносится 20 значений аналогового тэга ReactTemp путём импортированием CSV-файла, изменение базы данных будет зафиксировано следующим образом:

- В таблицу ModLogTable будет добавлена одна строка, которая будет использоваться для изменения содержимого таблицы History_OLEDB. Столбец UserName будет содержать имя пользователя в том виде, в каком оно указано в заголовке CSV-файла.
- В таблицу ModLogColumn будет добавлена одна запись о выполненном изменении. В столбец NewValue будет записано число 20, означающее, что в базу данных было введено 20 новых значений.

Качество данных

Качество данных представляет собой степень достоверности соответствующего значения. Качество данных может изменяться от "достоверного" ("good"), показывающего, что значение имеет то же значение, что и изначально полученное от устройства автоматики, до "недостоверного" ("bad", или "invalid"), показывающего, что значение или ошибочное, или не может быть проверено. Качество данных может ухудшаться при накоплении, сохранении, извлечении, обработке и отображении из-за влияния на систему внешних параметров и событий. Качество данных определяется следующими тремя факторами:

- Выставленным качеством данных устройствами сбора.
- Выставленным качеством подсистемой сохранения архиватора IndustrialSQL Server.
- Клиентскими установками качества.

Для описания качества сохранённых данных архиватора использует три разных параметра: Quality, QualityDetail и OPCQuality. Параметр OPCQuality тесно связан со значением параметра QualityDetail. Значение параметра Quality представляет собой производную меру достоверности значения данных, тогда как параметры QualityDetail и OPCQuality показывают либо хорошее качество, либо причину ухудшения качества данных.

Для записи информации о качестве каждого значения данных архиватор отводит 4 байта. (Это не означает, что для каждого значения в базе данных

выделяется ровно 4 байта, а лищь гарантирует, что каждому значению при извлечении его из базы данных может быть сопоставлено четырёхбайтовое значение качества). Два из этих 4 байтов хранят значения параметра QualityDetail, два других – значения параметра OPCQuality. Несмотря на то, что эти параметры взаимосвязаны, они могут иметь разные значения.

По существу параметр OPCQuality обеспечивает применение в будущем клиентских приложений, поддерживающих для оценки качества данных стандарт OPC, тогда как параметр QualityDetail (и Quality) обеспечивает функционирование приложений текущего и более ранних поколений.

Просмотр значений и характеристик качества данных

Возможные значения параметра OPCQuality находятся в Рабочей базе данных в таблице OPCQualityMap. Чтобы посмотреть коды OPCQuality и их описания, запустите анализатор запросов SQL Server Query Analyzer и выполните следующий запрос:

```
SELECT OPCQuality, Description FROM OPCQualityMap
```

Возможные значения параметра QualityDetail находятся в Рабочей базе данных в таблице QualityMa. Чтобы посмотреть коды QualityDetail и их описания, запустите анализатор запросов SQL Server Query Analyzer и выполните следующий запрос:

```
SELECT QualityDetail, QualityString FROM QualityMap
```

Для значений QualityDetail справедливы следующие замечания:

- Граничная точка для кода 448 параметра QualityDetail равна 1 секунде.

Получение и запись информации о качестве

При передаче значения архиватору IndustrialSQL Server оно обычно сопровождается показателем качества, который задаётся источником данных. Как правило, полученное значение записывается в архиве без изменений, однако есть ситуации, когда подсистема записи изменяет это значение.

Качество получаемых от серверов в/в данных

Получаемые от сервера в/в Wonderware через IDAS-источник значения всегда имеют присоединяемый к ним 2-байтный показателем качества. IDAS-источник передаёт подсистеме сохранения IndustrialSQL Server получаемое от сервера в/в значение как параметр OPCQuality, записывая в параметр QualityDetail значение 192. Исключение составляет передача сервером в/в значение 24, которое в этом случае присваивается и параметру OPCQuality, и параметру QualityDetail.

Тем не менее, в некоторых случаях IDAS-источник изменяет значение параметра QualityDetail на некоторую величину, чтобы отметить возникновение особых событий или условий. Более подиробно см. "Просмотр значений и характеристик качества данных". Когда IDAS-источник записывает в параметр QualityDetail один из зарезервированных кодов, он при этом не изменяет значение параметра OPCQuality. Таким образом, параметр QualityDetail будет передавать зарезервированную величину, а параметр OPCQuality сохранит значение, присвоенное ему источником данных.

Системный тэг SysDataAcqNBadValues накапливает количество переданных IDAS-источником данных с неудовлетворительным качеством. Если это количество окажется слишком большим, нужно проверить, нет ли проблем в IDAS-источнике или в сервере в/в.

Качество данных, поступающих не от серверов в/в

Архиватор IndustrialSQL Server может накапливать данные из различных источников, а не только от серверов в/в Wonderware. В число таких источников входят CSV-файлы специального формата, запросы SQL и сервер промышленных приложений (Industrial Application Server – IAS).

Качество получаемых от этих источников данных определяется так же, как и в случае данных, получаемых от серверов в/в, за некоторым исключением:

- Показатель качества получаемых из CSV-файлов данных всегда интерпретируется как OPCQuality, а параметру QualityDetail присваивается значение 192, если только не был получен показатель качества, равный 24, когда обоим параметрам присваивается это значение 24.
- Если за накопление данных отвечает MDAS (как в случае SQL-запросов и IAS), передаваемый источником показатель качества будет записываться в параметр OPCQuality, а параметру QualityDetail будет присвоено значение 192. Есключения составляют:
    - Если источник передаёт в MDAS, параметру OPCQuality будет присвоено значение 32, а параметру QualityDetail будет присвоено 24.
    - Если источник передаёт значение, равноебесконечности или NaN (не числовое значение), параметр OPCQuality будет содержать показатель качества, переданный источником, а параметру QualityDetail будет присвоено значение 249.
    - Если возникает особая ситуация или событие, MDAS заменит значение параметра QualityDetail одним из зарезервированных кодов. Более подробно см. "Просмотр значений и свойств показателей качества".

Качество на стороне клиентов

Клиентам передаются три показателя качества:

- 1-байтный укороченный показатель качества (Quality),
- 4-байтовый расширенный показатель уточнения качества (QualityDetail);
- 4-байтовый расширенный OPC-показатель качества (OPCQuality).

Показатель Quality содержит общую информацию о качестве, принадлежащую к категориям Good (достоверные), Bad (недостоверные), Doubtful (сомнительные) и InitialValue (исходное значение). Эта информация составляется по данным из источника данных, также как и из архиватора IndustrialSQL Server.

Показатель Quality характеризует качество соответствующего элемента данных. Следующая таблица даёт описание категорий качества:.

Категория	Шестандцате-ричное значение	Десятерич-ное значение	Описание
GOOD	0x00	0	Достоверное значение
BAD	0x01	1	Недостоверное (недействительное) значение
DOUBTFUL	0x10	16	Неопределённое значение

InitialValue	0x85	133	Исходное значение для сохранения по изменению

Параметры InitialValue и Doubtful составляются по параметру QualityDetail. Исходное значение зависит от типа исполняемого запроса. Подробнее см. в разделах "Операторы сравнения в запросах изменений", "Операторы сравнения в запросах циклических данных", "Операторы сравнения в запросах циклических данных с указанием разрешения" настоящего руководства.

Параметр QualityDetail содержит подробности, которые обычно определяются типом передавшего устройства. Значения параметра Quality могут поступать из различных источников, таких как серверы в/в или подсистемы хранения IndustrialSQL Server. Вместо параметра QualityDetail, обеспечивающего обратную совместимость, будет использоваться параметр OPCQuality.

ГЛАВА 3

Подсистема конфигурирования

Конфигурационные данные представляют собой информацию об элементах, образующих архиватор IndustrialSQL Server, таких как определения тэгов, определения серверов в/в, расположение файлов архивных данных и т.д. Конфигурационные данные являются относительно статичными, то есть их изменения непосредственно не связаны с производственной деятельностью предприятия. Хранение и обработка информации данного типа осуществляется подсистемой конфигурирования архиватора IndustrialSQL Server.

В случае типовой производственной среды создание баз данных и всех необходимых объектов, таких как таблицы, хранимые процедуры и представления, может занять не десятки часов. Применение архиватора, в составе которого уже определены все необходимые объекты, существенно облегчает задачу.

Конфигурационные параметры хранятся в таблицах Microsoft SQL Server в Рабочей (Runtime) базе данных. Если система содердит рабочие приложения InTouch, значительная часть необходимой информации может быть импортирована из словарей этих таблиц. Если используется сервер промышленных приложений IAS, часть работы по конфигурированию архиватора IndustrialSQL Server выполняется этим сервером автоматически. С помощью системной консоли управления (System Management Console) можно вручную добавить необходимые определения и настроить систему. Выполнить массовые изменения в базе данных архиватора или передать данные из одного архиватора в другой можно с помощью утилиты экспортирования и импортирования базы данных InSQL Database Export/Import Utility.

Вы можете в любой момент времени изменить конфигурацию системы, не прерывая её операции по сбору, сохранению и извлечению значений тех тэгов, на которые данноее изменение не распространяется. Конфигурационные параметры могут храниться вместе с полной предысторией их модификации.

Подробнее см. Главу 1 "Административные средства архиватора IndustrialSQL Server" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Содержание

- Компоненты подсистемы конфигурирования
- Рабочая и Промежуточная базы данных
- Менеджер конфигурирования (Configuration Manager)
- Динамическое конфигурирование

Компоненты подсистемы конфигурирования

Компонентами подсистема конфигурирования являются:

Компонент	Описание
Рабочая (Runtime) база данных	База данных сервера Microsoft SQL Server, используемая для хранения всех конфигурационных параметров системы.
Средства конфигурирования и управления	Включают клиентское приложение System Management Console, утилиты экспортирования и импортирования InSQL Database Export/Import Utility и средство конфигурирования, поставляемые вместе с сервером Microsoft SQL Server. Подробнее см. Главу 1 "Административные средства архиватора IndustrialSQL Server" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.
Менеджер конфигурирования (aahCfgSvc.exe)	Внутренний процесс, обрабатывающий конфигурационные и статусные параметры. Данный процесс исполняется как одна из служб операционной системы Windows.

Подробнее о компонентах архитектуры архиватора IndustrialSQL Server см. Главу 1 настоящего руководства.

О Рабочей и Промежуточной базах данных

Система управления реляционными базами данных (СУРБД), такая как Microsoft SQL Server, может обслуживать несколько баз данных. База данных представляет собой совокупность объектов, таких как:

- таблицы;
- хранимые процедуры;
- представления;
- пользовательские типы данных;
- пользователи и группы пользователей.

IndustrialSQL Server поставляется с двумя заранее сконфигурированными базами данных: Рабочей (Runtime) и Промежуточной (Holding).

В состав архиватора включён полнофункциональный Microsoft SQL Server, так как он поддерживает все системные таблицы сервера SQL Server. Подробнее см. в соответствующей документации Microsoft.

Примечание. Если Microsoft SQL Server является чувствительным к регистру вводимых символов, Рабочая и Промежуточная базы данных также будут чувствительными к регистру символов. При выполнении запросов необходимо учитывать правильность регистра.

Рабочая база данных

Рабочая база данных представляет собой основную базу данных, с которой взаимодействует архиватор IndustrialSQL Server. Её таблицы содержат всю конфигурационную информацию, такую как:

- Параметры системы.
- Определения тэгов.
- Информация по интеграции с системой InTouch.

- Системные имена и информация о группировании.
- Описания событий.

Эти таблицы обычно используются архиватором и клиентскими приложениями для поиска нужной информации. В них фиксируются все изменения конфигурации архиватора. Таблицы конфигурационных параметров являются обычными таблицами SQL Server, к данным в которых можно обращаться с помощью языка запросов Microsoft Transact-SQL. Подробнее об этом языке см. в соответствующей документации Microsoft.

Рабочая база данных используется также для хранения архивной информации некоторых типов:

- Данные контроля изменений.
- Данные генерируемых подсистемой событий.

Таблицы, в которых записываются информация слежения за изменениями и данные событий, также являются обычными таблицами SQL Server.

И наконец, Рабочая база представляет собой логическое хранилище значений архивируемых тэгов. И хотя физически данные записываются как архивные файлы на диске (архивные блоки), логически они включены в состав Рабочей базы данных. Подробнее об архивных блоках см. в разделе "Архивные блоки", об извлечении архивных значений – в Главе 6 "Подсистема извлечения данных" настоящего руководства.

Примечание. Вы не можете изменять имя Рабочей базы данных.

Промежуточная база данных

Промежуточная база данных используется обычно для временного хранения данных и конфигурационных параметров во время их импортирования с узла InTouch. Вначале данные отображаются на табличные структуры Промежуточной базы, а затем перемещаются в Рабочую базу.

Внимание! Не изменяйте конфигурацию объектов Промежуточной базы.

Подробнее об импортировании конфигурационной информации из приложений InTouch см. в Главе 3 "Экспортирование и импортирование конфигурационных параметров" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

О Менеджере конфигурирования (Configuration Manager)

Менеджер конфигурирования представляет собой внутренний процесс, который принимает изменения конфигурационных параметров и обновляет Рабочую базу данных. Таким образом, Менеджер конфигурирования является единственным компонентом системы, имеющим доступ к конфигурационной информации.

Менеджер конфигурирования исполняется как одна из служб операционной системы Windows и осуществляет обмен конфигурационными данными с различными частями системы через набор интерфейсов. Кроме того, он служит также шлюзом ко всей информации, описывающей состояние компонентов архиватора IndustrialSQL Server.

Динамическое конфигурирование

Архиватор IndustrialSQL Server поддерживает динамическое конфигурирование; то есть, вы можете во время работы системы изменять описания тэгов и другие объекты базы данных. Архиватор автоматически обнаруживает наличие изменений и без перезапуска соответствующим образом модифицирует своё внутреннее состояние. На взаимодействие с клиентами подобные изменения в конфигурации не влияют.

Динамическое конфигурирование предусматривает все модификации базы данных, которые могут быть выполнены во время работы системы. В некоторых случаях автоматически создаются новые архивные блоки. Подсистема конфигурирования выполняет нужные действия таким образом, что при этом потерь значений тэгов, на которые модификация не распространяется, не происходит. Тем не менее, очевидно, что потери возможны в процессе изменения базы данных, когда изменяется способ накопления значений тэгов.

В большинстве случаев система продолжает функционирование без прерывания. Перезапуск системы необходим:

- Когда измененяется местоположение основной области хранения данных, то есть параметр, который после установки архиватора обычно изменяется крайне редко.
- Когда измененяется значение системного параметра DataImportPath.

Подробнее о последствиях различных видов модификации базы данных см. в следующем разделе.

Обычно динамическое конфигурирование является процессом, выполняемым в два приёма:

- Вначале с помощью системной консоли управления, операторов языка Transact-SQL или других средств модификации баз данных добавляются новые или изменяются или удаляются существующие объекты базы данных.
  - Рабочая база данных обновляется согласно внесённым изменениям. В частности, когда с помощью редактора конфигурации созданётся новый аналоговый тэг, Рабочая база данных будет обновлена сразу же после нажатия кнопки **Завершить (Finish)**.
- После ввода всех изменений, вы должны их зафиксировать, что запускает на сервере процесс динамического переконфигурирования. Изменений выполняется в системе аналогично обработке транзакций.
  - Изменения в базе данных не будут отражены в рабочей системе до тех пор, пока они не будут зафиксированы. То есть фактически изменения фиксируются не в базе данных, а в системе.

Вы можете фиксировать изменения в системе, сериями и по отдельности, так часто, как это необходимо. Нет никаких ограничений на количество изменений, которые могут быть зафиксированы. Как правило, данный процесс при максимальной загрузке системы занимает не более 10 секунд, если только вы не создаёте новый архивный блок. Фиксация изменений с созданием архивного блока до полного их проявления занимает больше времени (от 3 до 5 минут).

Подробнее об условиях, в которых фиксация изменений не выполняется, см. в разделе "Исключительные случая фиксации изменений" настоящего руководства.

Влияние на систему изменений конфигурации

Различные виды динамического конфигурирования влияют на систему по-разному. В настоящем разделе кратко описаны виды типичных изменений конфигурации и их последствия.

Некоторые изменения базы данных могут потребовать создания нового архивного блока. Чтобы уменьшить потребности в создании новых блоков, в системе предусмотрено присутствие "запасные" тэги, для которых в архивных блоках имеется дополнительное пространство. Подробнее о таких тэгах см. в разделе "Выделение памяти под будущие тэги" в Руководстве по администрированию системы архивирования IndustrialSQL Server.

- **Изменение системных параметров**

  Последствия изменения системных параметров обычно проявляются сразу же (новый архивный блок не создаётся). Исключение представляет выделение дополнительного пространства для тэгов одного или несколько типов, что требует создания нового блока. Кроме того, когда уменьшается значение тэга HistoryCacheSize и фиксируется изменение, немедленная запись содержимого кэш-памяти на диск не производится. Для записи необходимо выполнить операцию повторного сканирования архивных блоков.

- **Изменение областей хранения**

  Изменение параметров кольцевой области хранения может потребовать останова и повторного запуска архиватора IndustrialSQL Server. Изменения параметров других областей хранения вступают в силу сразу же.

- **Добавление, удаление и изменение тэгов**

  Как правило, добавление в систему новых тэгов влечёт за собой создание нового архивного блока, если объёма "запасного" пространства недостаточно. Если превышется текущее значение доступного пространства, новый блок создаётся с учётом значения, заданного в таблице SystemParameter.

  Удаление тэгов приводит к результату немедленно.

  Для некоторых типов изменений параметров тэгов необходимо создавать новые архивные блоки. Сюда относятся изменение количества байтов, выделяемых для хранения целых значений, изменение типа исходных данных, изменение типа строковых значений с фиксированной длины на переменную длину и наоборот, изменение типа сохранения с "Not stored" (Не сохраняется) на "Stored" (Сохраняется), изменение кодировки значений строкового тэга (ASCII или Unicode), а также изменение типа накопления значений ("IOServer" или "Manual").

  В целом, к созданию нового архивного блока приводит любое изменение тэга, ведущее к изменению объёма занимаемого на диске пространства. Если изменяется только метод накопления или извлечения значений, все необходщимые действия выполняются системой без создания архивных блоков.

  Если измененить метод накопления тэга, это может привести к потере данных в течение некоторого небольшого промежутка времени. В целом, любое изменение источника данных тэга (например смена имени элемента, группы данных или сервера в/в) приводит к образованию "просвета" в его исторических значениях, что происходит вследствие отключения системы от предыдущего источника данных и подключения её к новому.

- **Добавление, удаление и модификация IDAS-источников**

Если добавлять в систему новый источник данных, это повлечёт за собой создание нового набора системных тэгов (тэги состояния и производительности для данного источника). При появлении нового источника данных новый архивный блок создавать не нужно, тем не менее, для новых тэгов это может потребоваться, если в системе недостаточно "запасного" пространства.

Если удалть определение существующего IDAS-источника, результат проявляется сразу же. Изменение параметров источника данных никогда не приводит к созданию нового архивного блока, но при этом может произойти частичная потеря значений связанных с ним тэгов (как это происходит, например, при перемещении источника данных на другой компьютер в результате операций разрыва и установления соединений).

- **Добавление, удаление и изменеие серверов в/в и групп данных**

  если вы добавляете или удаляете сервер в/в или группу данных, вам не нужно создавать новый архивный блок. Изменение характеристик сервера или группы данных может привести в частичной потере значений соответствующих тэгов (если данное изменение требует выполнения операций разрыва и установления соединения).

Ситуации, когда изменения конфигуриции не могут быть зафиксированы

Если архиватор или подсистема сохранения не запущена, изменения фиксироваться не будут, при этом таблица ConfigStatusPending будет очищаться. Это не относится к изменениям в следующих полях таблицы SystemParameter: HistoryCacheSize, HistoryDaysAlwaysCached и AutoStart.

Изменения не будут фиксироваться в рабочей системе в следующих случаях:

- Если ещё не закончено предыдущее динамическое конфигурирование.
- Ели ещё не закончено создание нового архивного блока (в результате плановой смены блоков, динамического конфигурирования или по требованию пользователя). На переключение архивных блоков после создания нового блока требуется примерно пять минут, а при плановой смене блоков – примерно десяти.

В каждом случае на экран будет выведено сообщение о невозможности выполнения фиксации изменений в системе.

ГЛАВА 4

Подсистема накопления данных

Архиватор IndustrialSQL Server разрабатывался для накопления и сохранения производственных данных с быстродействием, в несколько раз большим, чем быстродействие стандартных систем управления реляционными базами данных.

Основными источниками производсьвенных данных являются серверы промышленных приложений (Industrial Application Server – IAS), серверы доступа к даныым (DA Sever) и серверы в/в. Архиватором может получать данные более чем от 500 типов различных серверов Wonderware и сторонних разработчиков, обеспечивая самый большой в отрасли список доступа к устройствам сбора управления данными. Поддерживающие протокол SuiteLink серверы могут сопровождать передачу информации показателями качества и метками времени. В систему архиватора могут передаваться данные одновременно от нескольких серверов в/в с использованием линий связи различных типов, а также с применением режима передачи с промежуточным хранением, – всё это предотвращает потерю данных в случае сбоя сетевых соединений.

В роли источников данных реального времени также могут выступать различные клиентские приложения. Программы, разработанные с помощью службы ручного ввода IndustrialSQL Server (Manual Data Acquisition Service – MDAS), могут передавать информацию непосредственно в архиватор.

Исторические данные могут вводиться в систему в виде файлов в формате CSV (Comma Separated Values – разделённые запятыми значения), что позволяет записывать в базу данных информацию, накопленную другими архиваторами. Чтобы упростить импортирование данных из приложений InTouch, вы можете использовать утилиту InTouch History Importer.

И наконец, архиватор способен генерировать некоторые внутренние переменные, которые позволяют контролировать состояние системы.

Содержание

- Компоненты подсистемы накопления данных
- Накопление данных от серверов в/в
- Ввод данных с помощью операторов INSERT и UPDATE
- Импортирование данных из CSV-файлов
- Получение данных от приложений, поддерживающих MDAS.

Компоненты накопления данных

В приведённой далее таблице приведены компоненты подсистемы накопления данных. Многие из этих компонентов запускаются как службы Windows.

Компонент	Описание

Сервер в/в (DAServer)	Совместимая со спецификациям Wonderware программа, считывающая данные из ПЛК[3] и других устройств автоматики и передающая их в приложения Wonderware.
Служба IDAS	Процесс, который принимает оперативные данные от одного или нескольких серверов в/в и передаёт их архиватору IndustrialSQL Server.
Средства построения запросов	Любые средства, которые могут использованы для построения операторов INSERT и UPDATE языка Transact-SQL (например анализатор запросов Microsoft SQL Server Query Analyzer).
Папка импортирования данных	Указанная папка, из которой архиватор импортирует данные в историческую базу данных.
Утилита InTouch History Importer	Утилита импортирования данных из архивных словарей InTouch (.lgh). Подробнее см. раздел "Импортирование данных из архивных файлов InTouch" Главы 6 *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.
Служба MDAS	Процесс, принимающий данные от источников, не являющихся серверами в/в, и передающий их для архивирования в базе данных. Данные передаются с помощью COM-интерфейсов. Эту службу используют серверы промышленных приложений IAS, провайдер InSQL OLEDB, подсистема событий и клиентские приложения.
Служба системного драйвера	Внутренний процесс, который контролирует состояние архиватора и сообщает о его состоянии через набор системных тэгов. Системный драйвер также передаёт в подсистему сохранения показания текущей даты и времени суток, а также значения "тактовых" тэгов. Подробнее см. раздел "Системный драйвер и системные тэги" настоящего руководства.

Подробнее об общей архитектуре архиватора см. Главу 1 настоящего руководства.

Накопление данных от серверов в/в

Сервер в/в представляет собой программу, которая передаёт данные архиватору IndustrialSQL Server по протоколу DDE или SuiteLink. Данные, поступающие от программируемых контроллеров (ПЛК), удалённых устройств телемеханики и других аналогичных устройств автоматики, могут передаваться сервером в/в также в другие приложения, такие как ИЧМ-приложения InTouch.

Примечание. Протокол FastDDE не поддерживается. Если приложение архиватора исполняется в среде Windows Server 2003, протокол DDE также не поддерживается.

"Сервер" и "клиент" являются условными обозначениями. Чтобы данные могли передаваться от устройства автоматики в имеющееся в управленческой сети клиентское приложение, они должны быть "обслужены" целым рядом программ. Любое приложение, получающее данные от источника более низкого уровня, является "клиентом". Любое

[3] ПЛК – программируемый лгический контроллер.

приложение, передающее данные приложению более высокого уровня, является "сервером". Архиватор является и клиентом, и сервером: он является клиентом для серверов в/в и сервером для прикладных приложений.

Чтобы архиватор мог получать информацию от сервера в/в, нужно в базу данных внести сведения об этом сервере и сопоставить его с IDAS-источником.

Дополнительно о конфигурировании серверов в/в и IDAS-источников см. Главу 4 "Определение параметров сбора данных" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Адресация серверов в/в

Все совместимые с Wonderware серверы в/в используют адресацию DDE, которая включает следующие три составные части:

- **Имя приложения.** Это имя приложения, которое предоставляет данные. Имя приложения может включать имя компьютера, на котором это приложение исполняется;
- **Имя раздела.** Раздел представляет собой некоторое объединение элементов данных, состав которого определяется типом приложения;
- **Имя элемента.** Элемент данных представляет вместилище значения данных.

Формат адресации следующий:

\\<имя_компьютера>\<имя_приложения>|<имя_группы>|<имя_элемента>.

Следующая таблица предоставляет примеры DDE-адресации.

Компонент адреса	Сервер в/в	InTouch	Microsoft Excel
имя приложения	\\Computer1\Modbus	\\Computer1\VIEW	\\Computer1\Excel
имя раздела	ModSlave5	Имя тэга	Spreadsheet1
имя элемента	Status	ReacLevel	A1 (обозначение ячейки)

Чтобы архиватор IndustrialSQL Server мог получать данные от сервера в/в, адрес сервера должен быть включен в перечень конфигурационных параметров. Чтобы внести определение сервера в/в, можно использовать системную консоль управления (System Management Console) или выполнить импортирование из имеющихся приложений InTouch.

Подробнее о создании определений серверов в/в см. Главу 4 "Определение параметров сбора данных" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Подробнее об импортировании определений серверов в/в из приложений InTouch см. Главу 3 "Экспортирование и импортирование конфигурационных параметров" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

О службах IDAS

Служба IDAS (IndustrialSQL Server Data Acquisition Service – служба накопления данных IndustrialSQL Server) представляет собой небольшую программу, которая предназначена для приёма данных от одного или нескольких серверов в/в или серверов DAServer. IDAS исполняется как одна из служб Windows и имеет имя InSQLDataAcquisition. При необходимости служба IDAS обрабатывает получаемые данные и

пересылает их архиватору IndustrialSQL Server, где они затем сохраняются в архиве.

Примечание. Служба IDAS ранее называлась драйвером в/в. Конфигурация IDAS хранится в таблице IODriver Рабочей базы данных.

Когда вы внесите в базу данных архиватора определение нового сервера в/в, в соответствующей службе IDAS создаётся новый объект группы данных. для каждого уникального сочетания компьютера сервера в/в, приложения и группы данных имеется отдельный объект раздела. Каждый объект может находиться в одном из следующих состояний: idle (свободный), connecting (в стадии соединения), connected (соединён), disconnecting (отсоединяется), disconnected (отсоединён), overload (перегрузка), receiving (приём данных). Кроме того, для каждого объекта определён таймаут, так что вы можете установить длительность в зависимости от частоты изменений данных для данного раздела (группы).

Служба IDAS может принимать данные от нескольких серверов в/в, но передавать данные она способна только одному архиватору.

Служба IDAS может исполняться там же, где и архиватор, а также на отдельном компьютере. Прчём на одном компьютере может исполняться только одна служба IDAS. Как архиватор, так и служба IDAS используют для взаимодействия имена NetBIOS, и компьютеры должны быть доступны по этим именам. Проверку доступности сетевых узлов можно выполнить с помощью утилиты Windows Ping.

Сслужба IDAS обработывает значения параметров незаметно для архиватора, независимо от их времени. Метка времени, значение и показатель качества каждого принимаемого службой IDAS элемента данных сохраняются в архиве в соответствии с правилами, указанными для соответствующего тэга.

Подробнее о конфигурировании службы IDAS см. Главу 4 "Определение параметров сбора данных" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Конфигурация IDAS

В обычных условиях запуска архиватор IndustrialSQL Server выполняет конфигурирование службы IDAS, передавая ей информацию о тэгах (включая их источники данных), которые должны накапливаться службой. Когда подсистема сохранения будет готова к приёму, служба IDAS автоматически устанавливает соединение с источниками данных и начинает процесс сбора данных и их передачу в архиватор для записи на диск.

Главное назначение файлов конфигурации IDAS заключается в том, чтобы минимизировать сетевой трафик и предоставлять информации о службах для их автономного запуска. Дополнительно см. раздел "Автономный запуск служб IDAS" настоящего руководства.

Служба IDAS сохраняет свои параметры на локальном диске в папке "Documents and Settings\All Users\Application Data\ArchestrA\Historian\IDAS Configurations".

Файл конфигурации IDAS именуется следующим образом:

idatacfg_SERVERNAME_IDASKEY.dat

где:

- SERVERNAME является NetBIOS-именем компьютера архиватора
- IDASKEY является значением столбца IODriverKey для IDAS в Рабочей базе данных.

Если вы изменяетете параметры службы IDAS средствам системной консоли управления, это автоматически приводит к динамическому переконфигурированию архиватора. Если служба IDAS исполняется на удалённом компьютере, архиватор в первую очередь отошлёт ей изменённые конфигурационные параметры. После их получения служба IDAS изменит свою конфигурацию и обновит локальный файл конфигурационных параметров. Во время изменения конфигурации службы IDAS процесс накопления и передачи данных не прекращается. Архиватор сохраняет собственную копию конфигурационных параметров службы IDAS в папке
Documents and Settings\All Users\Application Data\ArchestrA\Historian\Configuration\IDAS Configurations.

После успешного завершения процесса изменения конфигурации файлы параметров службы IDAS на компьютерах архиватора и службы IDAS будут идентичными.

Внимание! Файлы конфигурации IDAS имеют специальный двоичный формат. Не изменяйте содержимое этих файлов.

Если на компьютере службы IDAS имеется более одного файла конфигурации (например, если вы удалите службу IDAS на узле тогда, когда с ним не было соединения, а затем добавите её снова), использоваться будет последний по времени создания файл. В журнале компьютера будет выведено соответствующее предупреждение. Подробнее об автономном запуске см. в разделе "Автономный запуск служб IDAS" настоящего руководства.

Обработка данных службами IDAS

IDAS выполняет минимальную обработку значений, получаемых от серверов в/в. IDAS преобразует данные к типам, зависящим от типов данных связанных с ними тэгов. Например, если получаемое значение должно быть записано в тэг с плавающей запятой, служба IDAS прежде чем передать в подсистему сохранения преобразует его в число с плавающей запятой.

Однако метки времени не преобразуются никак, поскольку серверы в/в и подсистема хранения IndustrialSQL Server используют один и тот же формат времени UTC (Universal Time Coordinated – универсальные координаты времени).

Служба IDAS не использует никаких правил сохранения (напрмер циклических или по изменению значений), если только она не отключена от архиватора и не функционирует в режиме передачи с промежуточным сохранением. В последнем случае запись данных в локальные архивные блоки выполняется с соблюдением всех установленных правил.

Передача данных в подсистему хранения

Получаемые службой IDAS данные записываются в буфера размером 64 Кбайт каждый и периодически пересылаются в подсистему хранения в виде пакетов.

IDAS передаёт данные из буфера либо при его заполнении, либо по истечении очередной секунды, в зависимости от того, какое из событий наступило раньше. Количество рабочих буферов можно устанавливать для каждой службы IDAS отдельно. При высоких скоростях обмена данными со службой IDAS и при появлении в журнале сообщений о переполнении буферов нужно увеличить их количество.

Подробнее см. Главу 4 "Определение параметров сбора данных" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Защита IDAS и брандмауэры

Уудалённая служба IDAS использует сетевые настройки, указанные при установке IDAS и настройки её на работу с архиватором. Архиватор использует сетевую учётную информацию, указанную при его установке, для взимодействия с удалёнными IDAS. Изменить эти настройки можно с помощью утилиты изменения параметров регистрации в сети ArchestrA (ArchestrA Change Network Account Utility).

Когда удалённая служба IDAS обмивается данными с архиватором, она использует механизмы защиты информации Windows, и она не передаёт по сети и не получает из сети никаких имён пользователей и паролей.

Когда служба IDAS передаёт данные в режиме промежуточного сохранения, используются именованные каналы и средства Windows совместного доступа к файлам. Если между службой IDAS и компьютером архиватора имеется брандмауэр (сетевой экран), он должен разрешать обмен данными через порты с номерами в диапазоне от 125 до 139 (TCP/UDP) и порт с номером 445 (TCP/UDP).

Подробнее о совместном доступе к файлам см. в разделе "Передача данных с промежуточным хранением" настоящего руководства.

Регистрация ошибок IDAS

IDAS передаёт все ошибки службе регистрации ArchestrA (ArchestrA Logger Service). Если служба IDAS установлена на удалённом компьютере, на нём же должна быть установлена и служба ArchestrA Logger Service. В обычных условиях ошибки будут регистрироваться как в локальном журнале, так и в журнале архиватора IndustrialSQL Server.

При разрыве сетевого соединения между обоими компьютерами никакая информация об ошибках в журнале архиватора не регистрируются. Поэтому нужно регулярно проверять журналы на компьютерах удалённых служб IDAS средствами системной консоли управления, чтобы не упустить случаи возникновения проблем в сети. Пропущенные сообщения после восстановления сетевого соединения повторно в журнал архиватора не посылаются.

Возможность передачи данных с промежуточным сохранением

IDAS предоставляет возможность передавать данные в режиме "записать-и-переслать" (с промежуточным сохранением), который обеспечивает защиту от потерь данных в случае разрыва соединения с архиватором (например в результате повреждения сетевой линии связи).

Примечание. Режим "записать-и-переслать" не поддерживается, если вы используете IDAS с резервированием.

Если служба IDAS не может передать данные архиватору, она записывает их во временные файлы на локальном жёстом диске. Такие файлы называются областью промежуточного сохранения ("записать-и-переслать"). Параметры области могут изменяться с помощью системной консоли управления.

Данные записываются в эту область до тех пор, пока она не будет заполнена до определённого уровня, после чего временное сохранение данных прекращается, а в журнал выводится соответствующее сообщение об ошибке. На удалённых сетевых компьютерах области промежуточного сохранения не поддерживаются.

Когда восстановлевается связь с архиватором, выполняются следующие действия:

- Архиватор проверяет версию конфигурационных параметров службы IDAS и, если они не изменялись во время отсутствия соединения, пытается восстановить передачу данных из IDAS. Служба IDAS прекращает запись данных в область промежуточного сохранения и возобновляет передачу архиватору получаемых из источников данных.
- Если архиватор обнаруживает различие между своей копией конфигурационных параметров службы IDAS и копией, хранящейся на компьютере IDAS, он динамически изменяет её параметры, чтобы устранить различия. Служба IDAS соответствующим образом обновляет локальный файл конфигурационных параметров, после чего архиватор выдаёт запрос на восстановление передачи данных службой IDAS.
- Как только служба IDAS определяет, что архиватор в доступен, она начинает передачу информации из области промежуточного сохранения, одновременно передавая оперативные данные.

Данные из области промежуточного сохранения переписываются в сетевую область InSQL8SF$ на компьютере архиватора. Данные пересылаются порциями с некоторой паузой между ними. Данные передаются с использованием сетевых учётных параметров, указанных при установке удалённой службы IDAS.

Если для заполнения архивного блока сохранённых (например при сбое питания) на удалённом узле IDAS данных недостаточно, они всё равно пересылаются в архиватор и заносятся в архив. При этом в журнал пзаписывается сообщение об обработке неполного архивного блока.

Если при обработке блока данных промежуточного хранения возникает ошибка, блок перемещается в каталог \Circular\Support. В журнал выводится соответствующее сообщение. Блоки, находящиеся в подкаталоге \Support, система автоматически не удаляет, так что для предотвращения переполнения кольцевой области хранения их нужно удалять вручную. При обновлении архиватора версий 8.0x до более высокой версии блок перемещается в существующий каталог \Log.

Если данные из области промежуточного сохранения успешно переданы, на удалённом компьютере они уничтожаются.

Включение режима передачи с промежуточным сохранением приводит к увеличению объёма системных ресурсов, потребляемых службой IDAS, поскольку для этого требуется инициализация подсистемы промежуточного сохранения и последующее её поддержание в постоянной готовности к приёму данных.

Если на компьютере архиватора имеется достаточно системных ресурсов, можно включить режим промежуточного сохранения в локальной службе IDAS, чем будет обеспечено накопление данных даже в случае остановки подсистемы хранения.

Резервирование IDAS

Для каждой службы IDAS в системе можно определить соответствующую "резервную" IDAS. Как только архиватор IndustrialSQL Server перестанет получать данные от основной службы IDAS, он автоматически переключится на резервную. Переключение на резервную IDAS выполняется достаточно быстро, тем не менее, в течение этого периода возможна потеря некоторого количества данных.

Примечание. Вы не можете определить резервную службу IDAS для работающей в режиме передачи с промежуточным сохранением основной IDAS. Эти функции являются взаимоисключающими. В приложениях, для которых требуется и резервирование, и передача с промежуточным сохранением, нужно использовать резервный сервер IAS с объектами RedundantDIObject.

Автономный запуск IDAS

Службы IDAS обычно запускаются службой конфигурирования IndustrialSQL Server (IndustrialSQL Server Configuration Service). Тем не менее возможен запуск удалённой службы IDAS с режимом передачи с промежуточным сохранением независимо от запуска архиватора. Функцию автономного запуска имеет смысл использовать в случае, когда архиватор может быть недоступен из-за отказа сети или когда приложение архиватора не запущено в момент включения удалённого компьютера со службой IDAS. Это позволит начать накопление данных, не ожидая никаких команд из основного узла.

Чтобы службу IDAS можно было запускаться автономно, для неё следует указать хотя бы один источник данных. При конфигурировании служба IDAS должна быть подключена к архиватору, чтобы на локальном компьютере был создан требуемый файл конфигурационных параметров. Так как локальныйкомпьютер содержит всю необходимую для накопления данных информацию, он должен обязательно содержать файл конфигурации. Подробнее см. раздел "Параметры службы IDAS" данной главы.

Вы можете выполнить автономный запуск службы IDAS как вручную, используя консоль служб Windows, так и автоматически во время включения компьютера.

При запуске служба IDAS пытается загрузить информацию из локального файла конфигурационных параметров. При успешной загрузке информация из файла используются для установления соединений с источниками данных и начала сбора информации. Когда внутренние буферы данных полностью заполняются, служба IDAS переходит в режим промежуточного сохранения и начинает записывать получаемые данные на жёсткий диск локального компьютера.

Если на компьютере IDAS существует несколько файлов конфигурации (например из-за удаления на узле службы IDAS в тот момент, когда с ним нет соединения, и последующего её создания), использоваться будет только последний по времени создания файл. В журнал компьютера службы IDAS будет выведено соответствующее предупреждение.

Если служба IDAS не сможет загрузить конфигурационные параметры, она переходит в режим бездействия, в котором находится до обращения к ней архиватора. Если в течение периода ожидания, длительность которого по умолчанию составляет 60 секунд, от архиватора не поступает никаких сигналов, служба IDAS завершает работу. Нужно отметить, что таймаут запуска службы IDAS отличается от таймаута ожидания при автономном запуске. Информацию о том, как изменять длительность стандартного таймаута для запуска службы IDAS, можно найти в техническом справочнике, который можно запросить в Службе технической поддержки.

IDAS возобновляет передачу данных, как только обнаруживает, что архиватор доступен.

Даже в случае, когда служба IDAS настроена на автономный запуск, в определённых условиях её может запускать служба конфигурирования IndustrialSQL Server. Таймаут автономного запуска представляет собой интервал времени в секундах, в течение которого автономная служба IDAS после запуска службой конфигурирования должна дожидаться команд конфигурирования, перед тем как она перейдёт в автономный режим. Когда служба IDAS запускается вручную с помощью консоли служб Windows или автоматически, таймаут не используется.

Если сконфигурировать IDAS как автономную службу, тип запуска службы InSQLDataAcquisition изменится на "Automatic" (автоматический), а сама служба будет запускаться всякий раз при включении или перезагрузке компьютера, на котором она установлена. Если затем перевести IDAS в

режим неавтономного запуска, тип запуска станет "Manual" (ручной). Подробнее см. технический справочник, который можно запросить в Службе технической поддержки.

Обработка службой IDAS просроченных данных

Просроченные (поздние) данные представляют собой данные, поступившие в систему архиватора с меткой времени, отличающейся от текущего времени архиватора более чем на 30 секунд. В зависимости от типа просроченных данных, получаемых от серверов в/в, они могут обрабатывать подсистемой сохранения по-разному:

- Данные просроченные, но поступают потоком. Например, показания с некоторого прибора поступают от сервера в/в с систематической задержкой в две-три минуты. Такие данные являются просроченными, тем не менее, они будут обрабатываться процессом сохранения оперативных данных, если для соответствующей группы будет разрешена передача просроченных данных;
- Данные пересылаются периодическими порциями. Например, в системе может применяться датчик, передающий блок своих показаний один раз в несколько часов. Просроченные данные такого типа будут обрабатываться процессом сохранения данных "ручного" ввода.

Чтобы обрабатывать просроченные данные, нужно при определении соответствующей группы данных указать возможность их появления. В противном случае архиватор будет игнорировать все данные, метки времени которых будут отличаться от текущего времени сервера IndustrialSQL Server более чем на 30 секунд. Количество подобных данных будет суммироваться в системном тэге SysPerfDataAcqNOutsideRealtime службы IDAS.

Подробнее см. Главу 4 "Определение параметров сбора данных" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Внимание! Поддержка просроченных данных не предназначена для устранения разницы показаний времени служб IDAS, серверов в/в и архиватора. Синхронизация часов этих компонентов является обязательной. Если одна из служб IDAS будет систематически присылать просроченные данные, не попадающие в окно оперативных данных, вероятнее всего, что отсутствует синхронизация часов. Подробнее см. раздел Синхронизация часов различных компонентов процесса накопления данных" настоящего руководства.

При обработке прострочennых данных справедливы следующиен правила:

- Если для группы просроченные данные разрешены и попадают в окно оперативных данных, они обрабатываются так же, как и обычные оперативные данные.
- Если просроченные данные не попадают в окно оперативных данных, на их обработку будет отводиться 1% ресурсов обработки оперативной информации.

Если вы разрешаете просроченные данные, нужно дополнительно определить два следующих параметра:

- **Idle Duration (Длительность ожидания)**. Длительность ожидания задаёт паузу в секундах перед обработкой данных от сервера в/в. Например, если установить в этом поле значение 60, данные от данного сервера в/в будут помещены в кэш-память и обработаны только в том случае, когда от этого сервера больше не будет поступать информация хотя бы в течение 60 секунд. 60 секунд является значением по умолчанию.

Длительность ожидания важна, когда вы предвидите периодический наплыв в службу IDAS простроченных данных. Конкретное значение зависит от параметров установленного архиватора и прикладных требований. В целом, чем больше длительность паузы, тем больше данных накапливается в клиентском приложении. Кроме того, при построении трендов не будет промежутков, указывающих на конец пакета данных (значений NULL). Вместо этого на графике будет нарисована прямая линия, продолжающаяся до начала нового блока просроченных данных.

- **Processing Interval (Интервал обработки).** Интервал обработки является одной из мер предосторожности. Если природа данных такова, что пауза никогда не выдерживается, подсистема сохранения будет обрабатывать передаваемые из группы данных значения не реже одного раза в указанный интервал обработки. По умолчанию длительность интервала обработки в два раза превышает длительность интервала ожидания. Устанавливать её менее длительности паузы ожидания нельзя.

  Интервал обработки важен, если вы определяете режим работы архиватора для тех случаев, когда просроченные данные будут поступать в службу IDAS стабильным потоком. Чем больше это значение, тем больше памяти будет потребляться архиватором, но не более 32 Мбайт. При достижении этого предела начинается обработка всех накопленных данных.

Всегда, когда это возможно, лучше использовать механизмы обработки данных в удалённой службе IDAS на компьютере сервера в/в, чем устанавливать параметры обработки просроченных данных в архиваторе.

Если установить возможность появления просроченных данных для нескольких групп данных удалённых служб IDAS, для всех групп будут действовать установки какой-либо одной группы с наименьшими значениями.

Все значения данных, получаемые от автономной службы IDAS как просроченные, сохраняются в режиме по изменению, даже если для тэга указан циклический режим сохранения. Таким образом, если предполагается, что значения тэга могут поступать с опозданием, для него следует указать режим сохранения по изменениям значений. Кроме того, если для какой-либо группы данных задана возможность появления значений с задержкой, метка времени первоначальных значений никогда не будет перезаписана подсистемой сохранения, независимо от того, насколько она ранняя. Это отличается от способа обработки оперативных данных, когда при получении первоначального значения с меткой времени, более ранней, чем время запуска система, подсистема сохранения заменяет метку времени первого значения текущим временем архиватора.

Если для автономной службы IDAS будет разрешить передачу просроченных данных, таймаут для соответствующей группы данных будет установлен в 0 и запрещён.

Внимание! Если вы разрешите просроченные значения для либо группы данных, подсистема хранения никогда не будет получать значения "отключения/подключения" от службы IDAS для входящих в эту группу тэгов. Даже при отсутствии соединения между источником данных и архиватором на тренде будет отображаться прямая линия.

Поддержка медленных и неустойчивых сетей

Вы можете настроить IDAS на работу в медленных и неустойчивых сетях. Неустойчивые сети представляют собой сети со случайными кратковременными (в несколько секунд) перерывами обслуживания, медленные сети являются сетями с низкой скоростью передачи, скажем 56

Кбит/с, или с массовыми передачами данных, занимающими практически всю полосу пропускания сети.

В случае неустойчивых сетей минимальную длительность режима промежуточного хранения придётся, возможно, увеличить по сравнению со значением по умолчанию. Это предотвратит частое переключение службы IDAS из обычного режима в режим передачи с промежуточным сохранением и обратно во время краткосрочных перебоев в работе сети. Как правило, длительность по умолчанию подходит для всех условий функционирования системы, за исключением самых неблагоприятных.

Для работы в медленных сетях вам, возможно, придётся сделать следующие настройки:

- Уменьшить размер файлового блока данных. Файловый блок данных представляет собой совокупность информации, передаваемой службой IDAS архиватору IndustrialSQL Server за один раз. Уменьшение размеров файлового блока снижает нагрузку на сеть. Признаком слишком большого размера блока является переход IDAS в режим промежуточного хранения сразу после начала текущей передачи данных вследствие того, что слишком большой объём данных приводит к перегрузке сети, и это заставляет службу IDAS переключаться в режим промежуточного хранения.
- Увеличить задержку передачи. Задержки передачи представляют собой интервалы времени, определяющие, с какой периодичностью служба IDAS посылает архиватору блоки данных из своей области промежуточного хранения. Увеличение этого значения позволяет сделать нагрузку на сеть более равномерной.
- Увеличить значение таймаута автономного запуска для тех служб IDAS, для которых эта функция определена. Признаком недостаточной длительности таймаута является переход удалённой IDAS в режим передачи с промежуточным сохранением, когда она запускается службой конфигурирования InSQL Configuration, с последующим восстановлением соединения с архиватором.
- Увеличить длительность таймаута соединения. Признаком недостаточной длительности таймаута является наличие в журнале сообщений об ошибках, относящихся к таймауту соединения со службой IDAS, несмотря на то что физически линия связи находится в исправном состоянии.

Отсутствие сетевого соединения не будет сказываться на работе удалённой IDAS до тех пор, пока ей будет хватать дискового пространства для работы в режиме накопления данных с промежуточным их сохранением на локальном диске.

Резервирование серверов в/в

Вы можете настроить сервер в/в так, чтобы включить "резервное" устройство. Этот сервер может находиться как на компьютере соответствующей службы IDAS, так и на любом другом компьютере сети. В случае разрыва соединения между основным сервером в/в и службой IDAS, она автоматически подключается к резервному серверу, если только он в это время работает. При переключении возможна потеря некоторой части данных.

Перенаправление серверов в/в на ИЧМ-приложения InTouch

Указание ИЧМ-приложений InTouch в качестве серверов в/в означает, что значения тэга будут поступать не от сервера непосредственно, а от какого-либо взаимодействующего с сервером в/в узла InTouch. Эта возможность

следует использовать, когда нагрузка на сервер в/в очень высока, или когда доступ к узлу InTouch легче, чем к соответствующему серверу.

Если таким образом переопределять источник данных в/в, вместо имени компьютера и типа сервера нужно использовать имя узла InTouch и соответствующее приложение. Предположим, например, что на компьютере с именем "I23238" использовался сервер в/в Modicon Modbus, а имя приложение в адресе сервера выглядело как **"\\I23238\modbus"**. При переназначении источника данных на узел InTouch с именем "InTouchNode1" эта часть адреса изменится на **"\\InTouchNode1\view"**.

Синхронизация часов для накопления данных

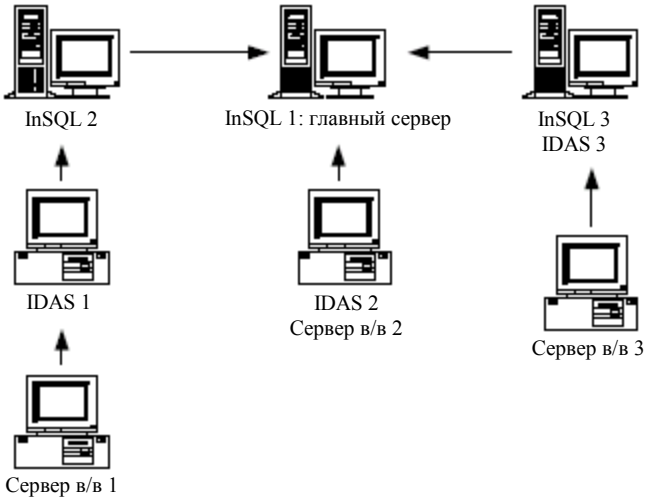
Все серверы в/в, которые поддерживают протокол SuiteLink, генерируют при получении технологических данных соответствующую метку времени. Важно понимать, как выполняется синхронизация меток време серверов в/в, показаний часов на компьютерах служб IDAS и часов архиваторов IndustrialSQL Server. Ниже дан обзор синхронизации времени.

1. Если вы имеете несколько архиваторов IndustrialSQL Server, все часы должны быть синхронизированы по часам одного из этих архиваторов, который в этом случае будет играть роль "главного" сервера времени. Часы этого источника "эталонного времени" можно синхронизировать по некоторому внешнему источнику. Выбор конкретного архиватора IndustrialSQL Server в качестве главного сервера времени выполняется путём соответствующей установки значения системного параметра *TimeSyncMaster*.
2. Архиватор периодически будет синхронизировать часы компьютеров удалённых служб IDAS по своим часам. Синхронизация служб IDAS разрешается с помощью системного параметра *TimeSyncIODrives*.
3. Каждфй час служба IDAS будет синхронизировать *механизмы генерации меток време* в связанных с ней серверах в/в по часам своего компьютера. (Настройка часов компьютера, на котором исполняется сервер в/в, не производится.) При отключении и повторном подключении некоторой группы данных при истечении таймаута или при обрыве связи генерация меток време на сервере в/в не будет изменена до тех пор, пока не истечёт очередной интервал синхронизации. Вы можете изменить периодичность синхронизации часов с помощью системного параметра *SuiteLinkTimeSyncInterval*.

Примечание. Синхронизация часов не применима для серверов в/в, использующих DDE, так как серверы не генерируют метки времени. При получении данных от серверов в/в по протоколу DDE используется текущее время компьютера службы IDAS.

Подробнее о системных параметрах см. Главу 8 "Вывод и изменение общесистемных параметров" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Следующий рисунок показывает пример того, как можно натсроить синхронизацию часов:



В случае клиентских приложений с включённым MDAS вы можете, чтобы синхронизировать часы клиентских компьютеров с часами вашего главного архиватора, использовать команду **чистое время** (для операционной системы Windows).

Явным признаком наличия проблем синхронизации является слишком большое значение системного тэга SysDataAcqNOutsideRealtime. Данный тэг показывает, сколько полученных значений от конкретной службы IDAS вышло за границы окна оперативных данных. Подробнее см. раздел "Окно оперативных данных" настоящего руководства.

Ввод данных с помощью операторов INSERT и UPDATE

Вы можете вставлять и обновлять данные в таблицы расширения IndustrialSQL Server с помощью операторов INSERT и UPDATE языка Transact-SQL.

Подробнее см. Главу 6 "Импортирование, внесение и обновление данных" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Получение данных из CSV-файлов

Данные в систему архиватора IndustrialSQL Server можно импортировать с помощью CSV-файлов, Таким способом могут импортироваться данные из других архиваторов. Чтобы импортировать данные из приложений InTouch, можно воспользоваться утилитой импортирования архивов InTouch History Importer, которая выполняет преобразование архивных файлов из формата InTouch в формат CSV. Импортированные значения будут объединены с уже имеющимися в архивных блоках данными.

Дополнительно см. Главу 6 "Импортирование, внесение и обновление данных" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Получение данных от приложений со службой MDAS

Служба MDAS представляют собой набор клиентских DLL-библиотек, которые обеспечивают получение данных (оперативных, а также вносимых

с помощью операторов вставки и обновления) от использующих эти библиотеки приложений и передачу их в систему архиватора IndustrialSQL Server. В частности, с помощью этой службы сервер промышленных приложений IAS передаёт данные архиватору IndustrialSQL Server.

Подсистема сохранения добавляет полученную от MDAS информацию к уже имеющимся данным. Вся информация хранится в таблицах расширения с именами History, WideHistory и Live.

Служба MDAS передаёт данные в архиватор, используя протокол DCOM. Поддержка DCOM должна быть разрешена (не заблокирована) на обоих компьютерах, как службы MDAS, так и архиватора IndustrialSQL Server, а порт TCP/UDP с номером 135 должен быть доступен. Данный порт становится недоступным, когда поддержка DCOM отключена хотя бы на одном из компьютеров, или когда между ними установлен брандмауэр (сетевой экран), блокирующий доступ к порту. Подробнее о передаче данных через DCOM с брандмауэрами см. в документации по Microsoft Windows.

Примечание. Получаемые от службы ручного ввода MDAS данные в таблицах ManualAnalogHistory, ManualDiscreteHistory и ManualStringHistory НЕ СОХРАНЯЮТСЯ. До появления архиватора IndustrialSQL Server версии 8.0 эти таблицы были единственным местом сохранения информации, вводимой вручную. В настоящее время они поставляются только для обратной совместимости.

ГЛАВА 5

Подсистема хранения данных

Назначение подсистемы хранения данных заключается в том, чтобы получать данные от различных источников производственных данных и записывать их на диск. Подсистема хранения записывает значения аналоговых, логических, строковых и системных тэгов в наборы дисковых файлов, называемых архивными блоками.

Вы можете извлекать из архива сохранённые данные с помощью SQL-запросов, исполняемых провайдером InSQL OLEDB, который является частью системы извлечения данных. На уровне подсистемы извлечения все данные выглядят хранящимися в таблицах Microsoft SQL Server. Подробнее см. Главу 6 "Подсистема извлечения данных" настоящего руководства.

Подсистема хранения обрабатывает только исторические данные, не выполняя никакой обработки конфигурационных параметров архиватора IndustrialSQL Server, таких как определения серверов в/в, тэгов и т.п. Подробнее о конфигурационных параметрах см. Главу 3 "Подсистема конфигурирования".

Подсистема хранения не обрабатывает также информацию, генерируемую подсистемой событий. Подробнее о событиях см. Главу 7 "Подсистема событий".

Изменения параметров подсистемы хранения выполняются с помощью системной консоли управления (System Management Console). Подробнее см. Главу 5 "Хранение данных" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Содержание

Компоненты подсистемы хранения

К подсистеме хранения относятся следующие компоненты:

Компонент	Описание
Служба сохранения оперативных данных (aahStoreSvc.exe)	Внутренний процесс, обрабатывающий оперативные данные и записывающий их на диск. Исполняется как служба операционной системы Windows.
Служба сохранения данных ручного ввода (aahManStSvc.exe)	Внутренний процесс, который принимает данные, не являющиеся оперативными, и записыват их на диск. Исполняется как служба операционной системы Windows. Данный процесс называется также "альтернативным" механизмом сохранения информации.
Активный образ	Область оперативной памяти, хранящая копии всех оперативных данных, записываемых подсистемой хранения на диск в текущий момент.
Архивные блоки	Наборы папок и файлов, в которых хранятся исторические данные.
Кэш архива	Область памяти, в которую загружается копия архивного блока для ускорения поиска и извлечения.
Интерфейс архива (IInSQLHistory)	Интерфейс передачи исторических данных от службы MDAS менеджеру конфигурирования (Configuration Manager), который затем передаёт их маршрутизатору MDAS. Данный интерфейс является частью менеджера конфигурирования.
Маршрутизатор MDAS	Часть менеджера конфигурирования (Configuration Manager), который передает информацию службе сохранения данных, введённых вручную, или службе сохранения оперативных данных.

Об общей архитектуре архиватора см. Главу 1 настоящего руководства.

Категории сохраняемых данных

В системе архиватора IndustrialSQL Server данные могут быть условно разделены на следующие категории: оперативные (реального времени), "просроченные" и "устаревшие". Каждая категория имеет свой набор характеристик и обрабатывается в системе по-своему. Характеристики категорий данных следующие:

- Последовательный поток данных. В этом случае данные, накапливаемые архиватором, могут поступать с упорядочением по времени или в произвольном порядке. В первом случае метка времени очередного элемента данных будет более поздней, чем метка времени предыдущего элемента. Как правило, данные, поступающие от серверов в/в, упорядочены по времени. Блоки импортируемых в систему данных не обязательно следуют друг за другом в хронологическом порядке, и яони вляются примером ввода информации в произвольном порядке.
- Положение относительно общесистемного "окна" оперативных данных. Окно оперативных данных представляет собой максимальную задержку поступления данных относительно текущего времени архиватора, при которой эти данные ещё рассматриваются как оперативные. Подробнее см. в следующем разделе.
- Поддержка "будущих" данных. Если в систему поступают данные, метки времени которых опережают текущее время архиватора, они обрабатываются системой в соответствии со своей категорией.

	Оперативные данные	Просроченные данные	Устаревшие данные
Поток данных	Метки времени должны быть упорядочены.	Метки времени должны быть упорядочены.	Метки времени могут быть указаны в произвольном порядке.
Окно оперативных данных	Метка времени значения должна попадать в окно.	Метка времени может и попадать в окно, и быть вне его, в зависимости от разрешения просроченных данных для группы данных.	Данные могут иметь любые метки времени.
Поддержка будущих данных	Имеется. Если значение с меткой будущего времени позднее перезаписывается другим значением, перезапись будет отражена в параметре QualityDetail.	Понятие неприменимо. Метка времени просроченных данных всегда указывает на прошлое по сравнению с сервером время. Метки времени последовательности просроченных данных могут смещаться по времени вперёд и назад, и они по-прежнему будут рассматриваться как просроченные. Если метка времени полученного значения является более ранней, чем метка времени сохранённого значения, значение с более поздней меткой будет перезаписано, и это будет отражено в параметре QualityDetail.	Нет. Устаревшие данные с меткой времени, указывающей на будущее, игнорируются.
Типичные источники значений	▪ серверы в/в ▪ системные тэги ▪ операторы INSERT, в которых аргумент wwVersion имеет значение REALTIME (wwVersion = REALTIME) ▪ данные от сервера промышленных приложений IAS	▪ серверы в/в (удалённые датчики) ▪ значения, извлекаемые из CSV-файлов в режиме "ускоренной загрузки"	▪ операторы языка Transact-SQL ▪ значения, извлекаемые из CSV-файлов в стандартном режиме импортирования
Служба сохранения	Данные обрабатываются службой сохранения оперативных данных (aahStoreSvc.exe)	Данные могут сохраняться любой из служб в зависимости от того, попадают ли их метки времени в интервал времени обрабатываемого в текущий момент	Данные обрабатываются службой сохранения данных, введённых вручную (aahManStSvc.exe)

		архивного блока (текущая "моментальная" копия). Полученные данные сперва поступают на обработку в службу сохранения оперативных данных.	

Если нужно, данных перед сохранением метки времени поступающих преобразуются в формат UTC.

Службы сохранения оперативных данных (aahStoreSvc.exe) и данных, введённых вручную (aahManStSvc.exe), записывают на диск всю полученную информацию в формате, обеспечивающем при их извлечении унифицированный поиск и представление.

Окно данных реального времени

Окно данных реального времени (оперативных) представляет собой максимальную задержку в миллисекундах относительно текущего времени архиватора, при которой данные ещё рассматриваются как оперативные. Допустимыми значениями являются значения от –30 до +999 мс.

Данных, имеющие метки времени, смещённые относительно времени архиватора, сохраняются следующим образом:

- Если получение просроченных данных для группы тэгов не разрешено и метка времени полученного значения отличается более чем на 30 секунд, такое значение игнорируется, а в журнал выводится соответствующее предупреждение. Если метка времени отличается менее чем на 30 секунд, значение сохраняется.
- Если получения просроченных данных для группы тэгов разрешено и полученное значение попадает в окно оперативных данных, оно записывается службой сохранения оперативных данных без изменений. Если полученное значение вне окна оперативных данных, оно передаётся альтернативным службам сохранения и записывается в архив без изменений.
- Если получение просроченных данных для группы тэгов разрешено, полученное значение сохраняется в режиме "по изменению", даже если для тэга задан режим циклического сохранения и принятое значение попадает в окно оперативных данных.

Подробнее см. раздел "Обработка службой IDAS просроченных данных" настоящего руководства.

Размер окна оперативных данных задаётся системным параметром RealTimeWindow. Его значение должно выбираться с учётом имеющихся ресурсов памяти системы. Если в системе определено большое количество тэгов или скорость поступления данных высока, увеличение размеров окна будет означать повышение требований к системной памяти, так как подсистема хранения должна будет обрабатывать больший объём данных, работая с ними, как с оперативными. На это потребуется больше ресурсов, чем на сохранение просроченной или устаревшией информации.

Выбор размеров окна оперативных данных важен и при использовании режима сохранения по изменению с указанием нормы изменений. Подробнее см. раздел "Норма изменений в режиме сохранения по изменениям" настоящего руководства.

Если памяти для обработки оперативных данных окажетс мало, система автоматически изменит размеры окна, а в журнал будет выведено соответствующее сообщение. Параметр RealTimeWindow при этом останется без изменений.

Внимание! Поддержка просроченных данных не предназначена для устранения разницы показаний времени служб IDAS, серверов в/в и архиватора. Синхронизация часов этих компонентов является обязательной. Если одна из служб IDAS будет систематически присылать просроченные данные, не попадающие в окно оперативных данных, вероятнее всего, что отсутствует синхронизация часов. Подробнее см. раздел Синхронизация часов различных компонентов процесса накопления данных" настоящего руководства.

Когда полученнре значение отвергается из-за того, что оно не попадает в окно оперативных данных, в журнал выводится соответствующее предупреждение. Такие предупреждения будут выводиться в журнал каждую минуту в течение всего времени, в течение которого получаемые значения игнорируются.

Модификации и версии данных

Архиватор IndustrialSQL Server способен модифицировать сохранённые данные путём ввода в архив дополнительных значений для уже зарегистрированного временного периода иили путём изменения значений существующих данных или длительности временного диапазона значений. Всем модификациям присваивается код версии, при этом всегда сохраняются предыдущие версии, что позволяет вам перед изменениями просматривать исходные данные.

Термин "исходные данные" обозначает первоначальный набор значений тэгов в архиве. Так, исходными данными является поток оперативных данных от сервера в/в. Каждая операция вставки или обновления приводит к образованию новой версии набора данных. Можно производить модификацию значениий тэгов любого типа (в/в, системных, тэгов ручного ввода). Новые значения тэгов ручного ввода и серверов в/в могут сохраняться либо как исходные данные (без версий), либо как вставленные данные (с версиями).

Архиватор позволяет просматривать значения только начальной или последней версии данных. Промежуточные версии данных в архиве сохраняются, но доступ к ним из подсистемы извлечения невозможен. Столбец QualityDetail будет хранить специальный код, который обозначает, что данные содержат модифицированные значения. Таким образом, текущим значением будет самая последняя версия данных, но все предыдущие версии при этом будут сохранены.

Режимы сохранения

Когда вы определяете тэг, нужно указывать способ сохранения его значений, то есть указать способ записи накапливаемой информации на диск. Предположим, что в систему каждую миллисекунду поступает до 100 значений тэга в/в. Нужно ли все их записывать на диск? Заданный способ сохранения будет определять и количество записываемых значений на диск, и достижимое разрешение по времени для сохранённых данных при их извлечении.

В архиваторе доступны следующие типы сохранения:

- В архив не записываются никакие значения.
- В архив записываются все значения ("принудительное" сохранение).
- В архив записываются только значения только при их изменении (сохранение по изменению).
- В архив записываются данные, разделённые по времени указанным интервалом (циклическое сохранение).

"Принудительное" сохранение

"Принудительным" называется метод сохранения, когда в архив записывается каждое значение, поступающее от устройств автоматики или клиентских приложений, причём без фильтрации (то есть не используется ни метод сохранения по изменению, ни метод циклического сохранения).

Данный метод полезен, когда в архив должны записываться данные от сервера в/в, получаемые им от устройства автоматики в некоторые особые моменты (то есть, когда неприменим метод опроса). Фильтрация подобных значений может быть выполнена сервером в/в, но не архиватором.

В случае, когда сбор данных выполняется сервером в/в с помощью опроса точек в/в без фильтрации, такой сбор будет аналогично циклическому сохранению. Но в циклическом методе значения архивируемых данных будут зависеть от указанной частоты сохранения, при этом достичь разрешения по времени, меньшего одной секунды, будет невозможно. При сохранении без фильтрации значения архивируемых данных определяются сервером в/в, а разрешение по времени может быть меньше одной секунды. Если источник данных посылает в каждом внутреннем интервале сканирования одно и то же значение, при принудительном сохранении в архив будут записываться все присылаемые данные.

Сохранение по изменению

Данный метод целиком ориентируется на запись данных в архив при изменении их значений. Запись выполняется в случае, когда поученное значение отличается от имеющегося. Этот метод сохранения называется также "сохранением по исключительным ситуациям". Обычно он используется для архивирования логических значений, строковых величин и аналоговых значений, изменяющихся сравнительно редко. Так было бы нерационально сохранять в архиве значение логического тэга, отражающего рабочее состояние какого-либо насоса, каждые десять секунд, если этот насос предназначен для безостановочной работы в течение нескольких месяцев.

Новое значение сохраняется в архиве вместе с соответствующей меткой времени, имеющей точность до 1 мс.

Данный метод допускает следующие мёртвые зоны изменений, не приводящих к регистрации в архиве:

- Мёртвая зона по времени.
- Мёртвая зона по значениям.
- Мёртвая зона (коридор колебаний) по скорости измененияй

Примечание. Вы можете добиться миллисекундного разрешения, используя оператор CONVERT в запросе OPENQUERY со специфическими режимами для архиватора IndustrialSQL Server. Подробнее об извлечении данных с миллисекундным разрешением см. в разделе "Запрос данных с миллисекундным разрешением" настоящего руководства.

Мёртвые зоны по времени и по значениям

Мёртвые зоны по времени и по значениям позволяют снижать временное разрешение сохраняемых архиватором IndustrialSQL Server данных.

- Мёртвая зона по времени (Time Deadband) представляет собой минимальную разницу в миллисекундах между метками времени для сохраняемых значений одного и того же тэга. Изменения значений тэга, происходящие в течение этого интервала, не сохраняются. Этот

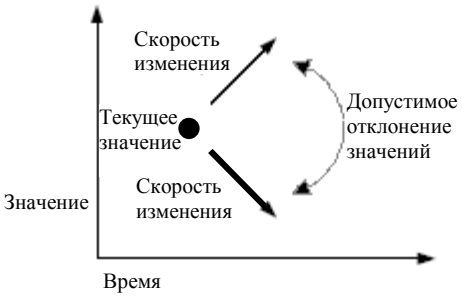
параметр действителен только для метода "сохранение по изменению" (delta storage). Величина интервала, равная 0, означает, что система должна сохранять все изменения тэга.

- Мёртвая зона по значениям (Value Deadband) выражается в процентах от величины разности между максимальным и минимальным значениями тэга, выраженными в единицах измерения. Изменения меньше этой величины не сохраняются. Этот параметр действителен только для метода "сохранение по изменению". Величина мёртвой зоны, равная 0, означает, что сохраняться должны все изменения тэга.

Мёртвая зона по скорости изменения

Мёртвая зона по скорости изменения (Swinging Door) представляет собой отклонение, выраженное в процентах от значения тэга во всем диапазоне значений аналогового тэга. Мёртвая зона по скорости изменения действительна только для метода "сохранение по изменению". Дополнительно могут задаваться мёртвые зоны по времени и по значениям. Допустимы любые положительные значения. Величина мёртвой зоны, равная 0, означает, что данный вид мёртвой зоны не используется.

Мёртвая зона по скорости изменения фактически представляет собой мёртвая зона по скорости изменения значения, рассчитываемой по последовательности входных значений. Например, указание мёртвой зоны 10% означает, что сохранение данных будет выполнено только в том случае, если последовательные значения будут отличаться больше, чем на 10%.



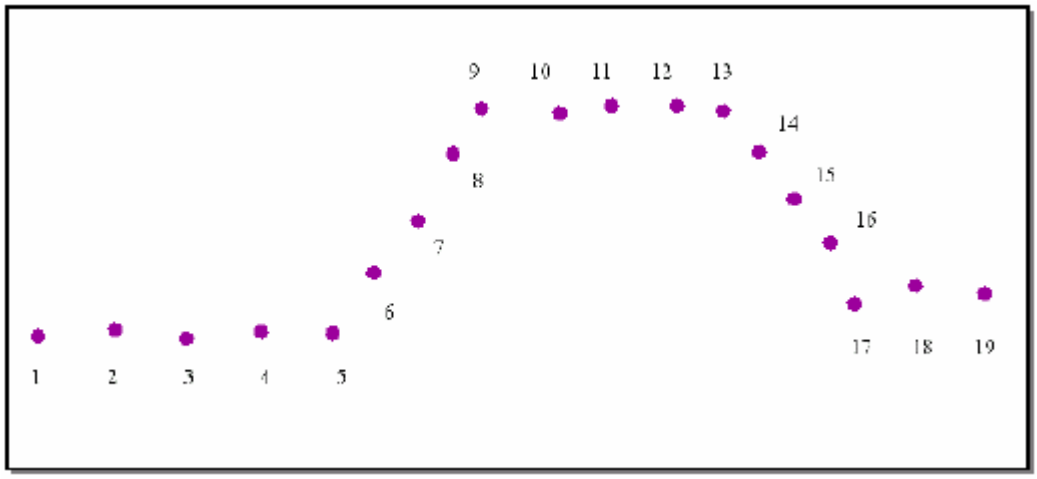
Преимущества мёртвой зоны по скорости изменения

Один из преимуществ мёртвой зоны по скорости изменения заключается в том, что при таком способе уменьшается объём дискового пространства, требуемого для сохранения значений данных. Другим преимуществом является то, что обеспечивается сохранение данных, предшествующих изменению мёртвой зоны, что невозможно при использовании мёртвой зоны по значениям. На трендах данных в этом случае пики и плоские участки становятся более выраженными, что позволяет иметь более ясную картину всего происходящего на предприятии.

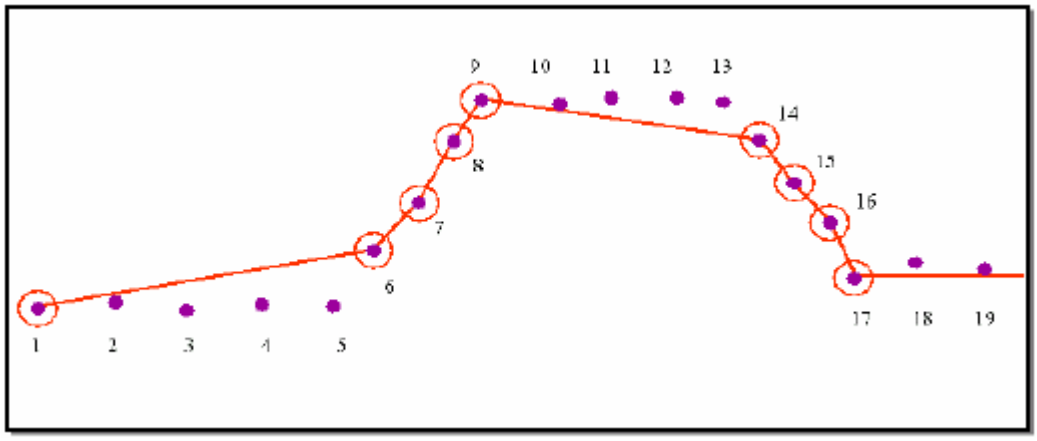
В целом, использование мёртвой зоны по скорости изменения позволяет точнее строить кривую изменения значений при той же или меньшей выборке данных, чем при использовании мёртвых зон по времени или по значениям.

На следующих рисунках показаны тренды одного и того же набора исходных данных, построенные с использованием мёртвых зон различных видов.

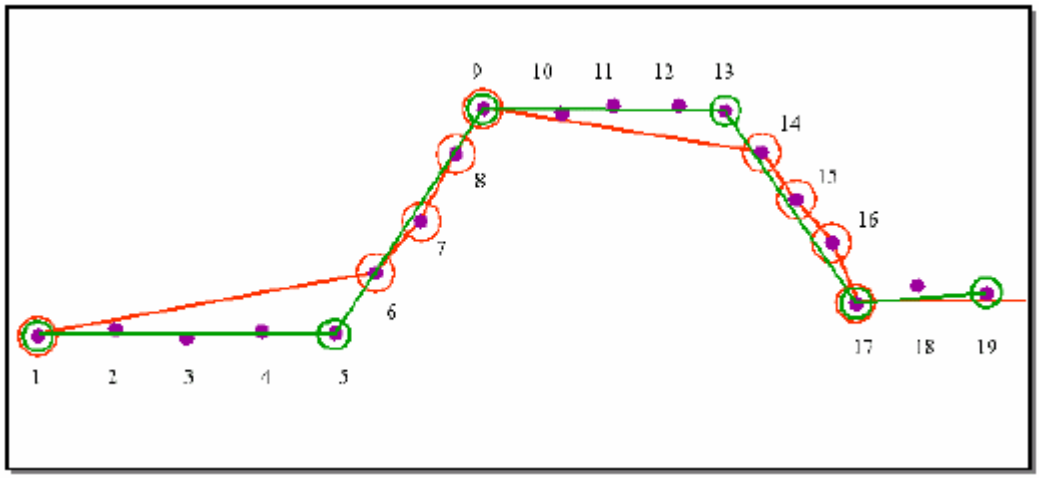
Следующий рисунок отображает исходные данные:



На следующем рисунке показан тренд данных (красная линия), построенный с применением мёртвой зоны по значениям. Нужно сказать, что в архиве будет сохранено только первое значение, отличающееся от предыдущего на величину, превышающую мёртвую зону, а все промежуточные будут пропущены.



На следующем рисунке показаны тренды данных, построенные с использованием мёртвой зоны по значениям (красная линия) и мёртвой зоны по скорости изменения (зелёная линия). Нужно сказать, что мёртвая зона по скорости изменению позволяет сохранять данные, предшествующие изменению, что обеспечивает более полную картину.



Применение мёртвой зоны по скорости изменения более целесообразно, когда значения тэгов изменяются равномерно, например повышаются или понижаются, как, скажем, температура или уровень жидкости в резервуаре.

Если же имеются частые отклонения от некоторой средней точки ("шум") в течение длительных периодов времени, следует задавать другие мёртвые зоны. Кроме того, в случае приложений с небольшим количеством тэгов, например 500, выигрыш в занимаемом пространстве может оказаться не очень значительным. В таких случаях от мёртвых зон вообще можно отказаться.

Мёртвая зона по скорости изменения обычно задаётся для сохранения следующей информации:

- Оперативные данные (реального времени), получаемые от серверов в/в или службы MDAS.
- Данные, получаемые от удалённой службы IDAS в режиме передачи с промежуточным хранением.
- Просроченных данные из группы данных сервера в/в, для которой они были разрешены.
- Данные, получаемые в результате "ускоренного" импортирования CSV-файла.
- Данные, вставляемые в режиме реального времени операторами Transact-SQL.

Эта мёртвая зона не подходит для случая ручной вставке данных путём стандартного импортирования CSV-файлов или выполнения операторов Transact-SQL.

Наилучшее представление данных, накапливаемых с применением мёртвой зоны по скорости изменения, обеспечивает приложение ActiveFactory Trend, в котором тип тренда указан как "line" (а не "step-line").

Дополнительные установки для мёртвой зоны по скорости изменения

Мёртвая зона по скорости изменения может комбинироваться с мёртвой зоной по значению, а также с мёртвой зоной интервала записи. Кроме того, сохранение данных будет задаваться величиной окна оперативных данных в архиваторе IndustrialSQL Server (системный параметр RealTimeWindow).

- Мёртвая зона по значению (Value deadband)

  При указании данного режима вместе с мёртвой зоной по скорости изменения (независимо от того, задан ли интервал записи), он всегда применяется первым (до применения других). В этом случае система вначале определяет разницу между полученным и последним сохранёнными значениями. Расчёт мёртвой зоной по скорости изменения выполняется только тогда, когда эта разница будет превышать величину мёртвой зоны по значению.

- Интервал записи

  Если время, прошедшее с момента последней операции сохранения, превышает величину мёртвой зоны, в архиве будет сохранено значение, соответствующее моменту истечения интервала записи, независимо от того, нарушалась ли мёртвая зона по значению или мёртвая зона по скорости изменения.

- Окно оперативных данных

  Размер окна оперативных данных (системный параметр RealTimeWindow) определяются интервалом времени, в течение которого данные рассматриваются как данные реального времени (оперативные). Размер окна оперативных данных имеет большое значение для метода сохранения с использованием мёртвой зоны по скорости изменения, поскольку определяют, сколько времени система

будет хранить полученную выборку в памяти без записи её в архив во время ожидания прихода очередной выборки.

Определение размеров окна оперативных данных и интервала записи позволяет сохранять в архиве данные, которые в иначе были бы отброшены вследствие того, что попадали бы под действие мёртвой зоны по скорости изменения или по значению:

- Окно оперативных данных позволяет сохранять точки, разница по времени между которыми меньше "ширины" окна, а разница значений меньше установленной мёртвой зоны по скорости изменения.
- Интервал записи позволяет сохранять те данных, разница во времени поступления которых превышает "ширину" окна оперативных данных, а разница значений меньше установленной мёртвой зоны по скорости изменения.

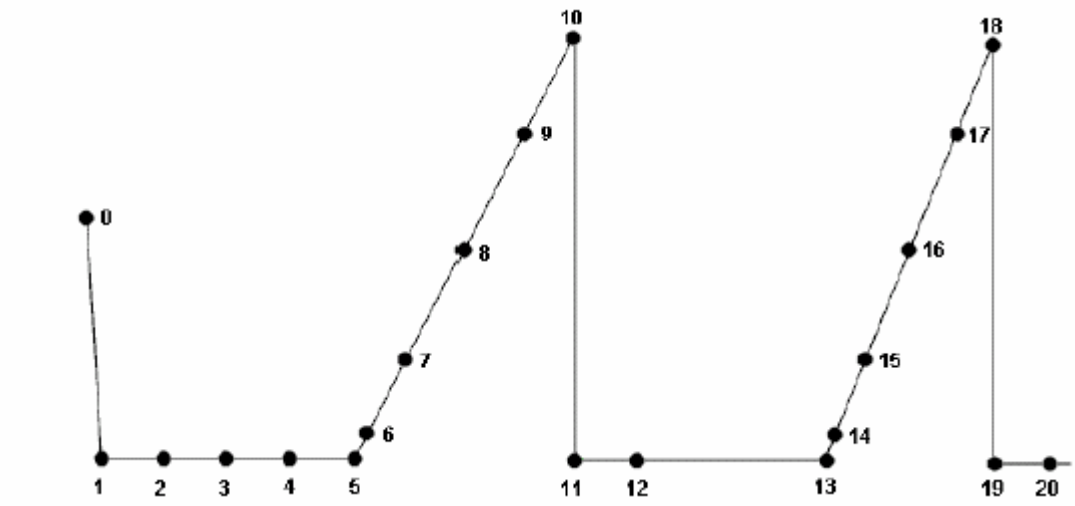
Примеры комбинирования мёртвых зон различого типа для сохранения данных приведены в следующем разделе.

Какую бы комбинацию мёртвых зон мы не задавали, в архив будут записаны только те выборки, которые фактически были получены от источников данных. То есть, никакие "расчётные" величины мёртвых зон на диск записываться не будут. Это справедливо также для использования интервала записи и окна оперативных данных.

Пример мёртвой зоны по скорости изменения

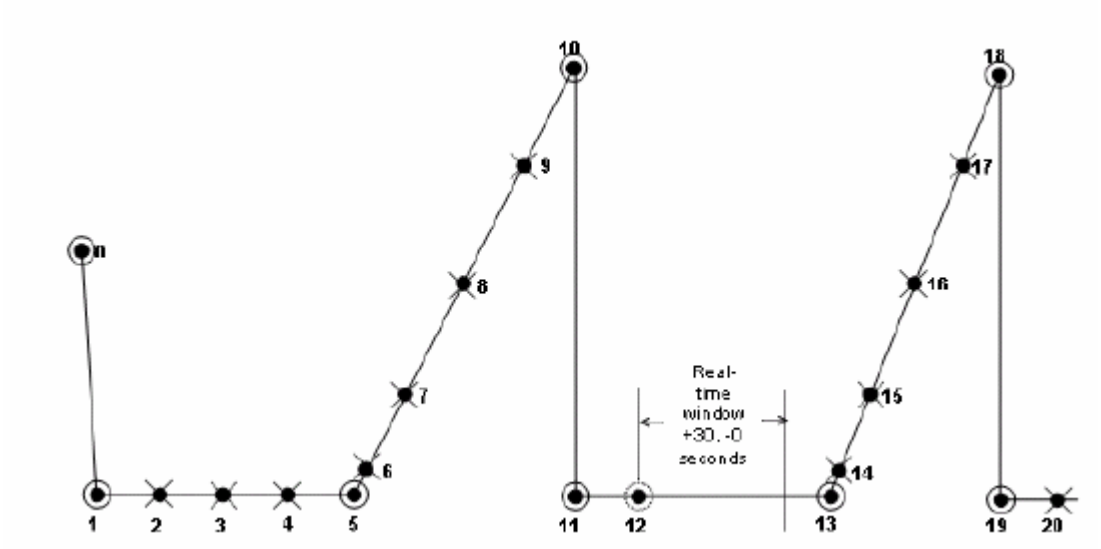
Следующие примеры показывают, как влияет на сохранение данных установка мёртвых зон.

Все примеры построены с использованием одного и того же набора исходных данных. Пронумерованные точки соответствуют фактическим значениям, полученным от источника данных.



Мёртвая зона только по скорости изменения

На следующем рисунке показана идеальная ситуация, когда шум во входном сигнале отсутствует, и в системе определена только мёртвая зона по скорости изменения (мёртвых зон по значению и по интервалу записи нет).



Предположим, что точка 0 уже сохранена в архиве. Прежде чем принять очередное решение о записи данных, система ожидает прихода двух точек. При получении точки с номером 2 подсистема сохранения вычисляет изменение угла наклона по следующим формулам. Угол0_1 – начальный угол, Угол1_2 – текущий угол.

Угол0_1 = (Значение1 – Значение0) / (Время1 – Время0)

Угол1_2 = (Значение2 – Значение1) / (Время2 – Время1)

Изменение_угла% = 100 * | (Угол1_2 – Угол0_1) / Угол0_1 |

Если

Изменение_угла% > Мёртвая зона_изменения

Другими словами, если изменение угла наклона в процентном отношении превышает указанную мёртвую зону изменений, подсистема сохранения запишет на диск точку 1. После этого система получает точку с номером 3. Начальным углом будет угол между точками 1 и 2 (если точка 1 была сохранена) или угол между точками 0 и 1 (если точка 1 не была сохранена), текущим углом – угол между точками 2 и 3.

Начальный угол при расчёте мёртвой зоны изменений для очередной точки не меняется, если предыдущая точка не была сохранена в архиве; в противном случае она будет принята равной "текущему" углу, который был определён для точки, сохранённой в архиве последней.

Предположив, что точка с номером 1 также была сохранена в архиве, получим, что, поскольку угол наклона между точками 2 и 3 практически равен углу между точками 1 и 2, мёртвая зона по скорости изменения не превышена и точка с номером 2 будет игнорироваться. При получении точки с номером 4 точка с номером 3 будет игнорироваться по той же причине и так далее вплоть до прихода точки с номером 6. В данном случае мёртвая зона по скорости изменения будет превышена (разница между углом между точками 5 и 6 и углом между точками 1 и 2 превышает величину мёртвой зоны), так что точка с номером 5 будет записана в архив.

При получении точки 7 точка 6 будет игнорироваться, несмотря на то что значение угла наклона кривой между точками 6 и 7 может быть значительным и что оно даже может превышать величину мёртвой зоны по скорости изменения. Это происходит из-за того, что его отличие от угла наклона между точками 5 и 6 недостаточно для записи точки 6 в архив. Рассуждая аналогично, мы получим, что будут попущены все данные до вплоть точки 12, кроме точек 10 и 11, которые на диск будут записаны.

Точка 13 иллюстрирует действие параметра ширины окна оперативных данных. В обычных условиях точка 12 в архив не попадёт. Если же время

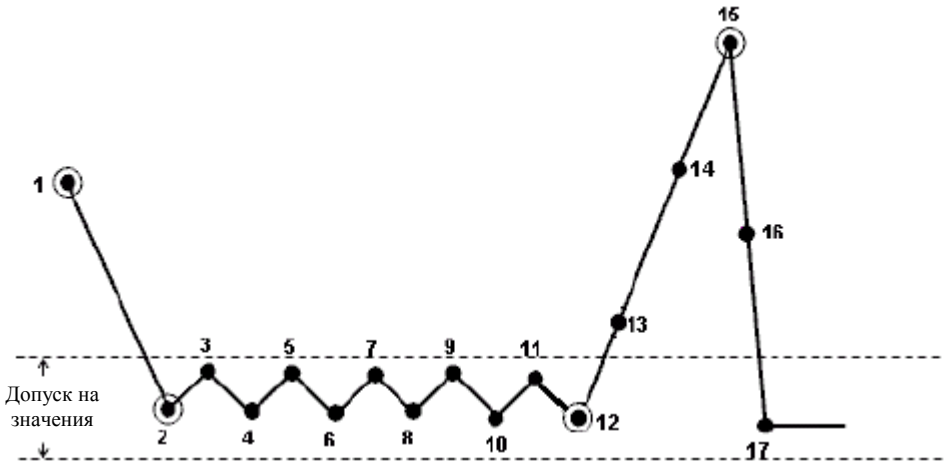
между моментами получения точек 12 и 13 превысит "ширину" окна, в котором система сохранения могла бы записать точку 12 в архив как оперативные данные, она будет сохранена в любом случае, а значение системного параметра SysRateDeadbandForcedValues будет увеличено на 1. Другими словами, если во время ожидания точки 13 метка времени точки 12 выходит за границы окна оперативных данных, точка 12 будет записана в архив независимо от того, нарушалась или нет мёртвая зона по скорости изменения.

Параметр SysRateDeadbandForcedValues накапливает количество "дополнительных" точек, которые были сохранены в архиве из-за недостаточной ширины окна оперативных данных.

При получении точки с номером 14 начальным углом для точки 13 станет угол между точками 11 и 12, а не 12 и 13, поскольку точка 12 была сохранена в архиве из-за недостаточной ширины окна оперативных данных. Сохранение точки по этой причине не приводит к новому расчёту угла наклона; так как это делается только после сохранения точек в результате превышения мёртвой зоны по скорости изменения. Затем выполняется "обычный" расчёт скорости изменения, в результате которого точка 13 сохраняется в архиве, и т.д.

Мёртвая зона по скорости изменения и по значению

На следующем рисунке показан сигнал с некоторыми "шумами". Для этого сигнала определяются мёртвые зоны по значению и по скорости изменения. Границы мёртвой зоны по значению показаны на рисунке пунктирными линиями.

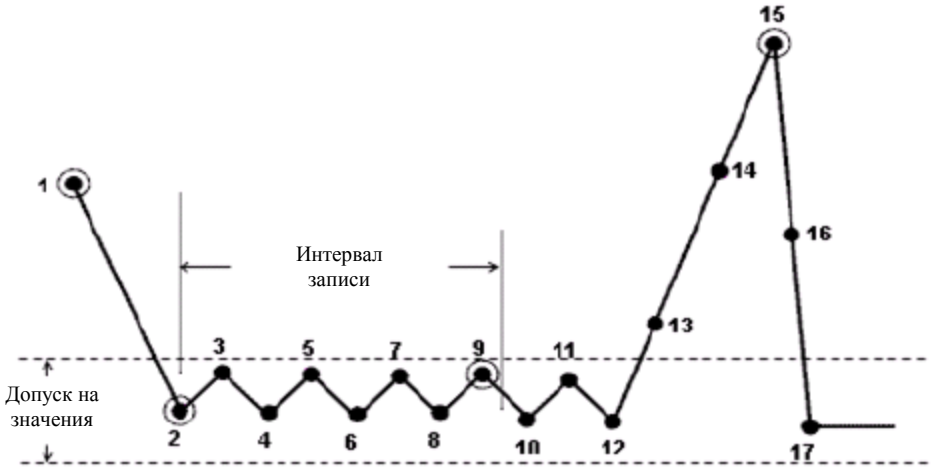


Предположим, точка с номером 1 была сохранена в архиве. Точка 3 успешно проходит проверку на мёртвую зону по значению, что позволяет оценивать превышение мёртвой зоны по скорости изменения для точек 2 и 3. Предположим, что превышение мёртвой зоны имелось. Тогда точка 2 также будет сохранена. До получения точки 13 все промежуточные точки будут игнорироваться, поскольку не будет нарушаться мёртвая зона по значению. В данном примере предполагается, что изменение углов наклона между точками 2 и 3 и точками 12 и 13 больше, чем величина мёртвой зоны по скорости изменения, поэтому точка 12 также будет сохранена в архиве. Нормальные операции начинаются по приходу точки 14.

Если бы для данного случая указывалась бы только мёртвая зона по скорости изменения, в архив были бы записаны все точки "шума" (с 3-й по 11-ю), поскольку угол наклона между последовательными значениями изменяется незначительно. "Шум" устраняется мёртвой зоной по значению, тем не менее, он и вносит некоторые искажения в выходной сигнал.

Мёртвые зоны по скорости изменения, по значению и по интервалу

На следующем рисунке показаны результаты применение мёртвой зоны по скорости изменения, комбинированной с мёртвыми зонами и по значению и по интервалу записи.



Предположим, что точка с номером 1 была сохранена в архиве. Точка 3 успешно проходит проверку на мёртвую зону по значению, что позволяет оценивать превышения мёртвой зоны по скорости изменения для точек 2 и 3. Предположим, что мёртвая зона по скорости изменения была превышена, и точка 2 также сохранена в архиве.

Применение только мёртвой зоны по значению могло бы привести к искажению сохранённых данных.

Например, пусть значение мёртвой зоны по скорости изменения таково, что точка 12 не сохраняется в архиве. То есть изменение улов наклона между точками 2 и 3 и точками 12 и 13 не превышает мёртвой зоны по скорости изменения. В этом случае представление данных (только точки с номерами 1, 2 и 15) даёт сильно искажённую картину, поскольку некоторые ключевые точки будут отброшены из-за непревышения мёртвой зоны по значению.

Для улучшения представления данных можно задать дополнительный интервал записи. Если время, прошедшее с момента последней операции сохранения, превышает эту величину мёртвой зоны, в архиве будет сохранено значение, соответствующее моменту истечения интервала записи, независимо от того, были ли нарушены мёртвые зоны по значению и по скорости изменения. В данном случае это будет точка с номером 9. Записанная в архив информация (точки 1, 2, 9 и 15) даёт лучшее представление об исходных данных.

Нужно отметить, что после сохранения точки 9 для принятия решения о сохранении данных будет использован угол наклона между точками 2 и 3, поскольку точка с номером 2 была последней, записанной в архив нормальным образом.

Можно установить любую длительность интервала записи, без какой-либо связи с "шириной" окна оперативных данных.

Определение правильности установки окна оперативных данных для всех тэгов

Чтобы определить, правильно ли установлены окна оперативных данных для всех тэгов, можно посмотреть на количество выборок, принудительно

сохранённых в архиве во время ожидания системой очередного значения для расчёта необходимых коэффициентов.

Параметр SysRateDeadbandForcedValues накапливает количество "дополнительных" выборок, которые были сохранены в архиве из-за недостаточной ширины окна оперативных данных. Определить количество значений отдельных тэгов, сохранённых таким способом, можно, составив запрос с указанием режима полного извлечения и показателя качества, равного 2240, что означает, что соответствующие величины сохранялись в результате недостаточности "ширины" окна оперативных данных.

Если количество таких выборок слишком велико, можно изменить способ сохранения значений тэга на "сохранение по изменению" или увеличить размеры окна оперативных данных.

Примечание. Первые две точки тэга, для которого определена мёртвая зона по скорости изменения, сохраняются всегда.

Требования к дисковому пространству и производительность

Одно из преимуществ метода сохранения с указанием мёртвой зоны по скорости изменения заключается в более высокой степени сжатия данных. Но поскольку подсистема хранения и без этого обеспечивает достаточное сжатие данных, экономия дискового пространства при архивировании значений медленно изменяющихся (реже двух раз в пятнадцать минут) тэгов будет незначительной. Например, объём наколенных точек тэга, изменяющегося 12 раз в час, составит 2 Кбайт. Даже если будет сохраняться каждое пятое значение, экономия составит всего 1,5 Кбайт в сутки.

При установке мёртвой зоны по скорости изменения нужно также учитывать следующие обстоятельства:

- Если в системе определено большое количество тэгов или же скорость поступления данных высокая, увеличение размеров окна будет означать повышение требований к системной памяти, поскольку подсистема хранения должна в этом случае обрабатывать больший объём данных, интерпретируемых как оперативные, и это потребует больших ресурсов, чем сохранение просроченных или устаревших данных.
- Если увеличивается размеров окна оперативных данных и определение мёртвой зоны по скорости изменения для медленно изменяющихся тэгов, объём занимаемого дискового пространства будет возрастать, поскольку в архиве будет сохраняться больше значений, чем в случае сохранения по изменению без мёртвой зоны.

Циклическое сохранение

Циклическое сохранение представляет собой запись значений аналоговых тэгов через определённые интервалы времени. Запись выполняется, только если в течение этого интервала значения тэгов изменилось. Например, вы можете записывать аналоговый тэг каждые 5 секунд.

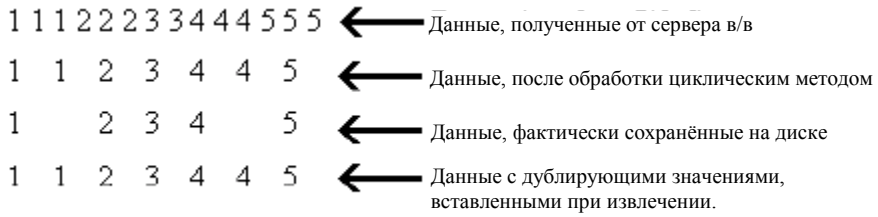
Интервал выполнения сохранение тэга называется периодом сохранения. Для каждого аналогового тэга может быть задан собственный период сохранения. Его величина должна выбираться с учётом минимального разрешения тэга по времени, с которым впоследствии будут извлекаться из архива. Сохранение данных через малые интервалы обеспечивает накопление большого количества данных в небольших промежутках времени. Указание больших интервалов времени может привести к потере значительной части нужной информации. Исключение составляют значения, получаемые или генерируемые в результате установления или разрыва соединения.

Допустимы следующие интервалы сохранения значений аналоговых тэгов:

- 1, 2, 3, 5, 6, 10, 15, 30 секунд;
- 1, 2, 3, 5, 6, 10, 15, 20, 30 минут;
- 1 час.

Все данные, полученные в течение одного интервала, будут сохраняться с меткой времени первого значения. Пусть для какого-либо аналогового тэга указан период сохранения в 10 секунд. Данные, полученные в конце этого интервала, будут сохранены с меткой сремени, сответствующей началу интервала.

Физически подсистема хранения записывает в архив циклические значения так же, как и значения, сохраняемые по изменению. То есть система сохраняет одинаковые последовательные значения как одно значение, вставляя при извлечении "пропущенные" данные.



Преобразование и зарезервированные значения

Архиватор IndustrialSQL Server сохраняет значения как 32-разрядные числа. Если источник данных присылает 64-разрядные числа, архиватор преобразует их в 32-разрядные, то есть происходит потеря точности на уровне 6 – 7 десятичных позиций. (Точное значение зависит от количества двоичных цифр, которые не всегда можно однозначно отобразить на фиксированное количество десятичных позиций.) Вследствие такой потери точности значения, отличающиеся в источнике данных, скажем, седьмой позицией, будут представлены в архиваторе одинаковым образом и при извлечении не будут восстановлены. Например, последовательные значения 1,0449991 и 1,0450001 будут округлены архиватором до 1,045000, так что второе значение в архив не попадёт.

Зарезервированными для всех режимов сохранения значениями являются:

- –2147483648, которое представляет внутреннее значение NULL для 32-разрядных тэгов.
- –32767, которое представляется внутреннее значение NULL для 16-разрядных тэгов.
- Последнее сохранённое значение тэгов всех типов.
- Последнее полученное значение аналоговых тэгов с циклическим архивированием.
- Последнее полученное значение тэгов, сохранение которых осуществляется с применением мёртвой зоны по скорости изменения.
- Недопустимые значения тэгов всех типов, за исключением символьных строк переменной длины.

Максимальная длина строкового значения с фиксированной длиной составляет 513 символов. Строки большей длины усекаются. Данное ограничение не распространяется на строки переменной длины.

Архивные блоки

Архиватор IndustrialSQL Server сохраняет данные в наборах, или блоках, файлов на диске. По существу, архивный блок представляет собой один из подкаталогов главной папки хранения данных, задаваемой при установке архиватора или при последующем динамическом переконфигурировании. Архивные блоки автономны в том смысле, что они содержат всю информацию, необходимую для извлечения данных временных периодов, представленных этими блоками. Временной период архивного блока указывается пользователем. По умолчанию он равен одним суткам, минимальная длительность периода составляет один час. Архиватор автоматически создаёт новый архивный блок при запуске системы, при плановом переключении архивных блоков, по команде пользователя, а также во время выполнения некоторых действий динамического переконфигурирования.

Примечание. Конфигурационные параметры и история событий хранятся не в архивных блоках, а в Рабочей базе данных архиватора.

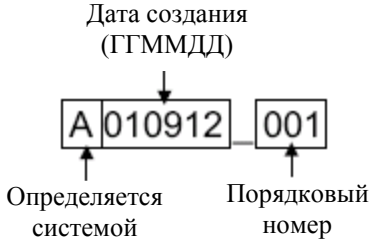
По мере поступления производственной информации размеры архивных блоков увеличиваются, ограничиваясь только объёмом жёсткого накопителя того компьютера, на котором исполняется архиватор.

При сохранении производственных данных не в таблицах SQL Server, а в архивных блоках существенно снижается потребность в свободном дисковом пространстве по сравнению с обычными реляционными базами данных, что происходит благодаря компактной упаковке информации. Представление данных для пользователей осуществляется с помощью провайдера InSQL OLEDB так, как если бы они хранились в таблицах SQL Server.

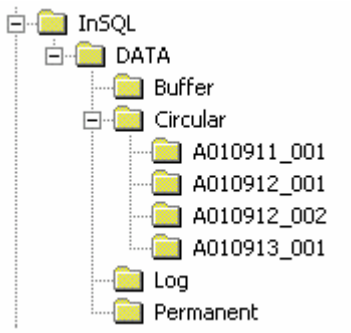
Именование архивных блоков

Каждый архивный блок хранится в отдельном подкаталоге каталога кольцевой области хранения. Имя подкаталогов отражает время и дату создания архиватором IndustrialSQL Server соответствующего архивного блока. Различие между архивными блоками, созданными в один и тот же день в результате многократных перезапусков системы, команд пользователя, или из-за того, что определяемый значением системного параметра HoursPerBlock временной диапазон блоков короче суток и пр., возникает за счёт добавления к имени архивного блока порядкового номера.

В следующем примере строка "010912" показывает дату, когда был создан архивный блок, а строка "_001" отмечает тот факт, что в этот день данный блок был создан первым.



По умолчанию временной диапазон архивных блоков равен одним суткам, то есть 24 часам. Из следующего примера следует, что система 12 сентября 2001 года запускалась два раза, что видно по наличию двух архивных блоков, приходящихся на одну дату, но различающихся порядковыми номерами. Общий временной диапазон этих блоков составляет 24 часа.



Примечание. Microsoft SQL Server кэширует извлекаемые в текущем сеансе данные для будущего использования. Если содержимое архивных блоков было изменено (например в результате копирования) и в запросе указан тот же самый временной период, клиенты должны либо сначала отсоединиться от сервера, а затем вновь соединиться с ним, чтобы в кэш-память были записаны обновлённые данные, либо указать в запросе другой интервал времени.

Создание архивных блоков

При начальной загрузке система создаёт новый архивный блок. Последующие архивные блоки в работающей системе создаются автоматически по истечении временного периода, заданного для них в базе данных. Его длительность по умолчанию равна 24 часам.

В любой момент времени вы можете создать новый архивный блок, используя однин из следующих способов:

- Выполнить команду меню системной консоли управления (System Management Console). Подробнее см. Главу 5 "Хранение данных" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.
- Запустить расширенную хранимую процедуру xp_NewHistoryBlock(). Подробнее см. Главу 4 "Хранимые процедуры" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Новый архивный блок можно также создать, если размер файлов данных текущего блока начинает превышать 1,5 Гбайт.

Примечание. Система допускает создание до 999 архивных блоков в сутки. По достижению этого предела система начинает записывать данные в самый первый блок текущих суток. Перед тем, как будет нарушен данный предел, в InSQL Console будет выдано соответствующее предупреждение. Чтобы сократить количество архивных блоков, создаваемых в течение суток, нужно увеличить значение стандартного временного диапазона (если он меньше 24 часов).

Расположение каталогов архивных блоков

Архивные блоки хранятся в следующих четырёх областях: кольцевой, дополнительной, буферной и постоянной. Каталоги кольцевой, буферной и постоянной областей хранения определяются при установке архиватора IndustrialSQL Server. Каталог дополнительной области может быть указан позже с помощью системной консоли управления.

При определении каталогов той или иной области хранения следует учитывать некоторые ограничения. Кольцевая область должна находиться на локальном диске компьютера сервера, а его имя должно указываться обычным образом (например C:\InSQL\Data\Circular). Остальные области хранения могут находиться на любом компьютере сети, при условии что пользователь службы ArchestrA имеет к ним полный доступ. В Windows

2000 сетевые каталоги можно указывать и как отображённые диски, и обычным образом в формате UNC. В Windows 2003 сетевые каталоги должны указываться только в формате UNC.

Определяя местоположение областей хранения, нужно, чтобы указанный каталог имел достаточное количество дискового пространства для хранения производственных данных в течение требуемого промежутка времени.

Расположение кольцевой области

Кольцевая область используется как основное место хранения исторических данных. При запуске подсистемы хранения она создаёт в кольцевой области архивный блок и начинает записывать в него поступающие данные. Блок производственных данных на диске сохраняется как один из подкаталогов папки кольцевой области.

Кольцевая область представляет собой единое дисковое пространство, данные в которое записываются как в "кольцевой буфер". Как только размер свободного дискового пространства уменьшается до некоторого порогового значения или "возраст" данных начинает превышать заданный срок, система начинает заменять самую старую информацию новой. Пользователь может ограничивать размер кольцевой области. Как только объём данных в каталоге кольцевой области превысит указанную величину, система начнёт удалять самую "старую" информацию. Подробнее см. раздел "Автоматическое удаление архивных блоков" настоящего руководства.

Система может не затирать данные в кольцевой области, а перемещать их в каталог дополнительной области хранения, если она была указана. Если размер свободного дискового пространства станет меньше порогового значения, самые "старые" данные будут перемещены в дополнительную область, чтобы высвободить место для новой информации.

В перечень обязанностей системного администратора необходимо включить контроль объёма свободного дискового пространства и периодическое выполнение резервного копирования архивных блоков на внешние носители, такие как ленты DAT.

Расположение дополнительные области

Когда свободного места на диске кольцевой области хранения становится меньше порогового значения, или когда размер кольцевой области начинает превышать максимальное значение, или когда архивные блоки достигают определённого "возраста", подсистема хранения начинает перемещать самые старые архивные блоки в одно или несколько дополнительных областей хранения.

Архивные блоки в дополнительной области сохраняются точно так же, как и в кольцевой. Но они не удаляются из неё по причине " возраста" до тех пор, пока не будет превышен суммарный возраст блоков в кольцевой и в дополнительной областях.

Примечание. Настоящая версия архиватора поддерживает только одну дополнительную область хранения.

Дополнительные области имеют порядковые номера. Блок данных перемещается из одной дополнительной области в последующую до тех пор, пока он не будет записан в самую последнюю область. После этого данный блок из системы удаляется.

Располагать дополнительную область рекомендуется отдельно от кольцевой области, как минимум на другом логическом диске. Хорошие результаты даёт организация этой области в отдельном разделе или на другом физическом диске, но лучше всего создать её в системе на отдельном компьютере.

Присутствие в системе дополнительной области хранения не является обязательным.

Расположение постоянных областей

В постоянных областях хранится важнейшая информация (например количество запусков реактора), делать перезапись которой нельзя. Подсистема хранения никогда не пытается удалять данные из этой области. Доступ к сведениям в постоянной области выполняется так же, как и к данным кольцевой области.

Перемещение архивных блоков эту область хранения выполняется с помощью расширенной процедуры xp_DiskCopy. Подробнее см. Главу 4 "Хранимые процедуры" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Расположение буферных областей

Буферные области используются для временного хранения данных, например при их извлечении из архива. Такие области могут располагаться на том же жёстом диске, что и кольцевая область, а также на отдельном диске. Доступ к информации в буферной области осуществляется так же, как и к данным кольцевой области. Подсистема хранения никогда не удаляет данные из этой области.

Автоматическое удаление архивных блоков

Архивные блоки в кольцевой и дополнительной областях могут удаляться автоматически для высвобождения пространства для новых блоков. Момент удаления определяется значением минимально допустимого свободного пространства, значением максимально допустимого занимаемого пространства, а также возрастом данных в указанной области хранения.

Каждый раз при обнаружении изменений, которые были внесены какой-либо из подсистем или в результате действий пользователя, менеджер конфигурирования определяет размер доступного пространства в кольцевой и дополнительной областях. Если служба оперативного сохранения функционирует, эта проверка выполняется каждые 20 секунд (период обновления файла block.inf).

Затем менеджер конфигурирования вычисляет сумму размеров всех архивных блоков (включая текущий) в кольцевой области и определяет, достаточно ли в ней места для их хранения.

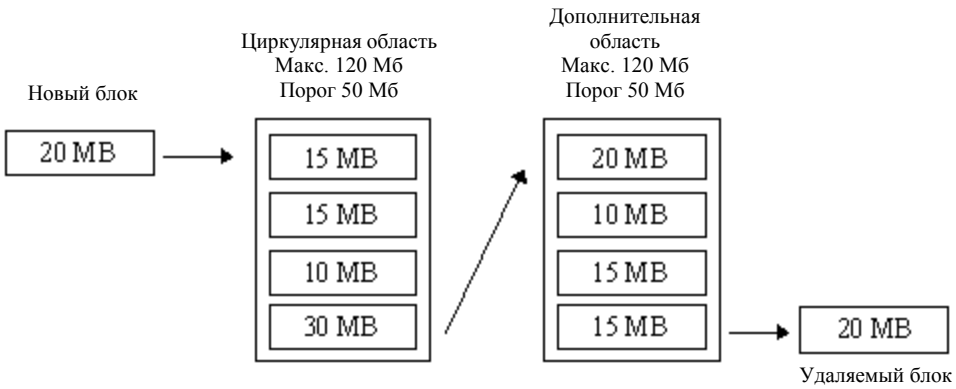
Если это значение будет меньше некоторой минимальной пороговой величины, подсистема хранения удаляет такое количество старых архивных блоков, которое достаточно для восстановления баланса, и создаёт новый архивный блок.

Если дополнительная область существует, блоки с наиболее старой информацией будут не удалены, а перемещены в эту область. Запись в эту область выполняется так же, как и в кольцевую, однако при достижении определённых пределов (минимального свободного или максимального занимаемого пространства, возраста данных) блоки уже никуда не перемещаются, а удаляются с диска.

Во избежание подобной потери данных системный администратор обязан регулярно проверять наличие достаточного свободного дискового пространства и выполнять резервное копирование архивных блоков с самыми старыми сведениями на внешние носители до их удаления.

Рассмотрим взаимодействие кольцевой и дополнительной областей. Пусть архивный блок сохраняется в кольцевой области, а её максимальный размер установлен в 120 Мбайт. Кроме неё в системе определена дополнительная область размером 120 Мбайт, минимальный объём

свободного пространства для обеих областей установлен в 50 Мбайт. По сути это означает, что фактически должно использоваться 70 Мбайт дискового пространства.



Примечание. Указанные цифры примерные, на практике они должны быть гораздо больше.

Обычно пороговое значение минимального свободного пространства выбирается хотя бы в 1,5 раза большим, чем размер самого большого архивного блока. Такая установка обеспечивает менеджеру конфигурирования достаточно времени для перемещения самых старых архивных блоков из кольцевой области в дополнительную с последующим их удалением из кольцевой.

Если взглянуть на кривую доступного дискового пространства, будет видно, что его объём колеблется между минимальной и максимальной величинами с резкими пиками, соответствующими перемещению блоков. Непосредственно перед пиками кривая будет опускаться ниже минимального порогового значения.

Если максимальное значение будет превышено до того, как "возраст" блока достигнет указанного срока, блок будет перемещён или удалён. Если по какой-либо причине "устаревший" блок нельзя будет удалить, он будет сохраняться до превышения предела на размер или пространство.

Активный образ

Активный образ представляет собой часть памяти, в которой временно хранятся копии полученных значений во время их записи на диск, так что клиенты всегда могут получать требуемую информацию. Скорость наполнения Активного образа определяется периодичностью изменения значений соответствующих тэгов. Получаемые значения будут накапливаться в Активном образе, пока не будет собрано их некоторое определённое по умолчанию количество. При установке архиватора оно равно 65 выборкам для каждого тэга.

Примечание. Скорость накопления значений в Активном образе никак НЕ СВЯЗАНА со значением, хранящемся в столбце AcquisitionRate таблицы Tag.

После того как в Активном образе будет накоплено количество данных по умолчанию, система начинает перезаписывать самые старые значения новыми. Перезапись устаревших данных высвобождает пространство для новых значений, причём в любой момент времени в Активном образе может содержаться не более 65 значений. Кроме того, на диск могут быть сохранены не все значения из Активного образа – их доля определяется периодичностью сохранения (столбец StorageRate таблицы Та). Если накопление выборок данных происходит со скоростью, превышающей

скорость сохранения, их разрешение по времени в Активном образе будет выше.

Для предотвращения перезаписи данных в Активном образе нужно увеличить стандартное количество накапливаемых значений, чтобы подсистема хранения успевала записывать их на диск.

Автоматическое изменение размеров Активного образа

Система постоянно рассчитывает оптимальное количество выборок значений каждого тэга, которые должны существовать в Активном образе, в зависимости от скорости поступления данных. Каждые пять минут, начиная с первой минуты после запуска, система проверяет, требуется ли выделять дополнительную память для Активного образа. Для каждого тэга система определяет средний интервал времени одного значения в зависимости от меток времени первой и последней выборок и их числа. При необходимости система увеличивает количество выборок. Следует учитывать, что, если длительность расчётного интервала будет превышать 65 секунд, система не будет увеличивать количество накапливаемых в Активном образе значений более 65.

В целом, если система будет получать более 65 изменений тэга в течение 65-секундного интервала, она увеличит значение поля SamplesInActiveImage таблицы Tag и увеличит объём памяти для хранения дополнительных значений быстро меняющегося тэга.

Новое расчётное значение числа выборок представляет собой количество выборок, которые необходимы для хранения в Активном образе значений, собираемых в течение 1 минуты (+15%). Эта величина обновляется только если системный параметр AIAutoResize установлен в 1 и количество требуемых выборок превышает 65. Данное число записывается в столбец, если AIAutoResize установлен в 1 и количество требуемых выборок превышает 65. Данное число записывается в столбец SamplesInActiveImage таблицы Tag во время запуска системы.

Чтобы отключить автоматическое увеличение размеров Активного образа, системному параметру AIAutoResize нужно присвоить значение 0. Кроме того, возможно изменить период расчёта количества выборок, требуемых для хранения в Активном образе значений тэгов, присвоив системному параметру AIResizeSecInterval нужное значение.

Система никогда автоматически не уменьшает количество выборок в Активном образе; это можно сделать вручную с помощью хранимой процедуры ww_SetAISamples. Однако эти изменения вступят в силу только при следующем запуске системы. Работающая система неспособна автоматически определять, что новое количество выборок меньше требуемого для хранения в Активном образе значений тэга (получаемых в течение 1 минуты). Подробнее см. Главу 4 "Хранимые процедуры" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Изменение размеров Активного образа – операция, требующая выделения значительного объёма ресурсов. Для предотвращения нежелательных помех после настройки количества выборок для хранения значений тэга в Активном образе оно в дальнейшем будет изменяться только тогда, когда прирост числа выборок составит более 10% от предшествующего количества.

Выбор размеров Активного образа позволяет оптимизировать характеристики системы, поскольку подсистема извлечения сначала ищет данные с указанной в запросе начальной датой именно в нём. Если такие данные находятся, они и возвращаются. Если данных с указанной начальной датой в Активном образе нет, система считывает их с диска и

возвращает вместе с данными, существующими в Активном образе. Если при этом происходит дублирование данных, в результатах выполнения запроса остаётся информация с диска.

Например, чтобы создать почасовые сводки о предшествующей неделе производства в зависимости от значений тэга SysTimeHour, количество выборок в Активном образе, необходимых для хранения значений этого тэга, можно увеличить до 144 (24 * 7 = 144). В результате система не будет считывать с диска требуемые значения тэга SysTimeHour, при этом время извлечения данных и нагрузка на процессор уменьшатся.

Внимание! Хотя в Активном образе можно вручную определить количество выборок, требуемых для хранения строк переменной длины, отличное от 0, NULL или 65, делать это не рекомендуется. Производительность системы может существенно снизиться, поскольку в этом случае для каждой подобной выборки будет выделяться по 1038 байтов памяти.

Влияние способа хранения данных в Активном образе на эффективность извлечения данных

Данные в Активном образе могут сохраняться в том виде, каком они поступают в систему, либо в том, какой определяется выбранным методом сохранения. Указать нужный режим можно во время определения тэга. Подробнее см. Главу 2 "Тэги" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

При выборе режима сохранения в Активном образе всех значений результаты, возвращаемые при выполнении запроса, могут различаться в зависимости от времени его выдачи. Указанное различие может также проявляться при извлечении значений тэга, которые сохранялись как циклическим способом, так и по изменению при действующих допусках на время или значение.

Например, если запросить данные, приходящиеся на период от 20011206 1:00:00.000 до 20011206 1:02:00.000, которые в этот момент хранились в Активном образе, они будут возвращены с тем разрешением по времени, с которым они накапливались.

Если же запрос будет выдан значительно позднее, когда данные из Активного образа уже будут сохранены в архивных блоках (на диске), разрешение по времени может уже не быть таким высоким. Если периодичность сохранения будет ниже, чем скорость накопления, некоторые значения будут отсутствовать. Может показаться, что произошла некоторая "потеря" данных, хотя в действительности параметры системы и не предусматривали их сохранения.

Влияние динамического конфигурирования

Во время динамического конфигурирования необходимость в создании нового архивного блока в результате внесённых изменений определяется менеджером конфигурирования (Configuration Manager).

Регистрация значений существующих тэгов в это время не прекращается, однако сохранение значений новых и модифицированных тэгов не будет осуществляться в течение пяти-десяти минут после фиксации изменений (если только в блоке не было ранее предусмотрено место под будущие тэги). Подробнее см. раздел "Выделение памяти под будущие тэги" Главы 2 "Тэги" *Руководства по администрированию архивирования сервера IndustrialSQL Server™*.

Если по каким-либо причинам подсистема хранения не может завершить конфигурирование, система генерирует критическую или фатальную ошибку и записывает в журнал соответствующее сообщение. Ошибка также возникает в случаях, когда процесс реконфигурации длится слишком долго, в результате чего буфер подсистемы хранения переполняется и происходит потеря данных.

Управление памятью

По умолчанию, архиватор IndustrialSQL Server загружает все сведения о тэгах из архивных блоков в оперативную память для более эффективного выполнения клиентских запросов. В состав этой информации входят свойства тэгов и индексные указатели, обеспечивающие быстрое перемещение по файлам с оперативными данными. Процесс, манипулирующий сведениями о тэгах в оперативной памяти, называется службой индексации IndustrialSQL Server Indexing Service (aahIndexSvc.exe).

В крупномасштабных системах общий объём информации о тэгах из всех архивных блоках может превышать 2 Гбайта – предел, который определяется операционной системой для одиночных процессов. На практике это значение может быть ещё меньше и определяется объёмом физического ОЗУ.

Общий объём информации о тэгах в архивных блоках зависит не только от общего числа тэгов, но и от числа версий тэгов, создаваемых во время модификации устаревших данных. При регулярном выполнении операций вставки и обновления данных, а также импортирования CSV-файлов рекомендуется постоянно контролировать объёмы памяти, занимаемые всеми системами, как большими, так и малыми.

Во избежание чрезмерного потребления памяти службой индексации IndustrialSQL Server нужно настроить её параметры и контролировать её функционирование с помощью следующих системных параметров и тэгов:

- Параметры HistoryCacheSize и HistoryDaysAlwaysCached.

Максимальный объём памяти в мегабайтах, в которую служба индексации может загружать информацию о тэгах, задаётся значением системного параметра HistoryCacheSize. Если его значение равно 0, архиватор будет использовать столько памяти, сколько ему нужно. Любое другое значение ограничивает доступный архиватору объём памяти для хранения информации о тэгах указанной величиной.

Заполнение памяти указанного объёма выполняется по принципу "выгрузка информации, использованной давно". Это означает, что при выдаче запроса на данные из архивного блока, которого нет в кэш-памяти, служба индексирования сначала выгружает из неё сведения о тэгах из блока, к которому система раньше всех прекратила обращаться, и затем загружает данные из запрошенного блока.

Все эти операции выполняются в фоновом режиме, вместе с тем при обращении к данным из блока, отсутствующего в кэш-памяти, возможно некоторое замедление реакции системы. Следует помнить, что чем меньше объём кэш-памяти, тем больше времени может потребоваться для выполнения запросов.

Для повышения эффективности извлечения информации из самых последних блоков, например при построении трендов данных за последнюю неделю, их можно "закрепить" в кэш-памяти. "Закрепление" выполняется путём указания в системном параметре HistoryDaysAlwaysCached числа дней хранения архивных блоков в кэш-памяти.

- Тэги SysHistoryCacheFaults и SysHistoryCacheUsed.

Необходимость в изменении объёма памяти, выделяемой службе индексации, занимаемой процессом aahIndexSvc.exe, можно определить с помощью Менеджера задач или Консоли мониторинга производительности Windows. Для этой же цели можно использовать системные тэги SysHistoryCacheFaults и SysHistoryCacheUsed. Большое количество ошибок кэш-памяти может служить признаком тог, что её объём недостаточен. Значение тэга SysHistoryCacheUsed показывает количество байтов, занятых в текущий момент сведениями о тэгах. Это позволяет оценивать потребность системы в памяти для хранения данных о тэгах, даже если управление памятью не включено.

Текущее состояние архивных блоков можно узнать и с помощью системной консоли управления System Management Console. Если сведения из него в память не загружены, пиктограмма блока отображается неотчётливо. Чтобы увидеть изменения в состоянии архивных блоков, может потребоваться выполнить команду обновления окна консоли.

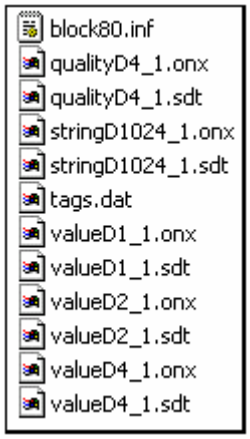
Файлы "мгновенной" копии

Изменение параметров файлов "мгновенной" копии может понадобиться в очень редких случаях.

Внимание! Не изменяйте параметры файлов "мгновенной" копии, если вы не имеете твёрдого представления об их назначении и использовании.

Параметры файлов "мгновенной" копии

Каждый архивный блок состоит из набора файлов "мгновенной" (.sdt), в которых и осуществляется хранение накопленных значений.



Формат имени файла следующий:

<тип>DX_x,

где

<тип> – строка "value" (для хранения логических и аналоговых значений) или "string" (для хранения строковых значений).

X – количество байтов, выделяемых для хранения значений. В именах файлов строковых строк фиксированной длины (128 и менее байтов) указано реальное значение в байтах, в именах файлов строковых строк фиксированной длины (более 128 байтов) – число 1024.

_x – номер "потока" данных. Если в файле "мгновенной" копии хранятся исходные оперативные значения, номер равен 1. Для хранения прочих данных, например вставленных устаревших данных или обновлённых данных, система может создавать дополнительные файлы, не добавляя их в файлы исходных значений. Наличие

нескольких файлов "мгновенной" копии обеспечивает более эффективное хранение сразу нескольких потоков данных.

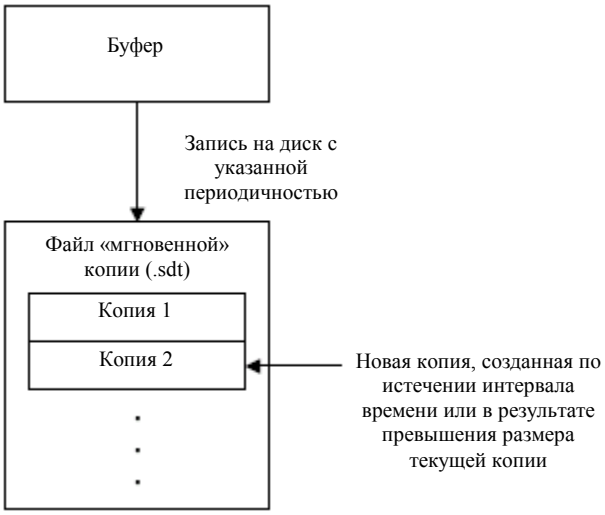
Для хранения данных одинаковой размерности (в байтах) создаются отдельные файлы "мгновенной" копии:

"Мгновенная" копия	Описание	Хранимые значения
-1	Используется для хранения величин, отличных от одно-, 16- и 32-разрядных значений.	Строковые строки фиксированной длины, строковые строки переменной длины.
0	Используется для хранения дополнительных параметров качества.	Дополнительные характеристики качества
1	Используется для хранения одноразрядных значений.	Логические значения.
2	Используется для хранения 16-разрядных значений	Аналоговые и целые величины.
4	Используется для хранения 32-разрядных значений	Вещественные (с плавающей точкой), типа long и целые величины.
8 (в будущем)	Используется для хранения 64-разрядных значений	Целые величины.

Обновление файлов "мгновенной" копии

До записи значений тэгов на диск они накапливаются в динамически распределяемых буферах в оперативной памяти системы. (Эти буферы отличаются от Активного образа.) Содержимое динамических буферов записывается в файлы "мгновенной" копии с частотой сохранения образа (по умолчанию один раз в 30 секунд).

В каждом файле хранится несколько наборов "моментальных" копий значений архивируемых тэгов, которые они имели в соответствующие моменты времени.



В момент запуска системы в файл "мгновенной" копии (.sdt) записывается первый набор данных, состоящий из первоначальных значений архивируемых тэгов. Через 30 секунд в него добавляются новые значения тэгов, взятые из динамических буферов, и т.д.

Как только размер этой копии достигает определённой величины (по умолчанию 2 Мбайта) или пройдёт час, генерируется вторая "мгновенная" копия. Когда её размер достигнет 2 Мбайт или пройдёт второй час, генерируется третья и т.д.

Каждые 30 секунд система проверяет, сколько значений каждого тэга было получено. Если более 15000, система автоматически сгенерирует очередную "мгновенную" копию.

ГЛАВА 6

Подсистема извлечения данных

Назначение подсистемы извлечения данных – принимать SQL-запросы от клиентов, находить затребованные данные, выполнять всю необходимую их обработку и возвращать результаты. Извлечение конфигурационных параметров и информации о событиях возможно с помощью обычных SQL-запросов, поскольку информация этого типа хранится в обычных таблицах базы данных SQL Server. Исторические данные должны извлекаться из архивных блоков и представляться клиентам так, словно они тоже хранятся в таблицах SQL Server. Чтобы подсистема извлечения могла считывать данные из информационных хранилищ обоих типов, в её состав включены следующие компоненты:

- Модификация поставщика данных SQL Server, который определяет, находятся затребованные сведения в обычных таблицах SQL Server или в архивных блоках.
- Служба низкоуровневого считывания, выполняющая извлечение затребованных данных из архивных блоков и передачу их провайдеру InSQL OLEDB как из "виртуальных" архивных таблиц.
- Набор расширений SQL Server, реализованных в виде дополнительных столбцов архивных таблиц. Они позволяют указывать в запросах тип возвращаемого набора строк (например количество строк, разрешение данных по вмени или режим извлечения).

Дополнительно о хранении данных см. в Главе 5 настоящего руководства.

Содержание

Компоненты подсистемы извлечения данных

В следующей таблице перечислены компоненты подсистемы извлечения данных.

Компонент	Описание
Рабочая (Runtime) база	База данных SQL Server, в которой хранятся

данных	конфигурационные параметры и сведения о событиях.
Архивные блоки	Файлы, в которых хранятся производственные данные. В контексте Microsoft SQL Server архивные блоки представляют собой не локальные источники данных.
Служба низкоуровневого считывания данных (aahRetSvc.exe)	Процесс, извлекающий информацию из архивных блоков и представляющий её в виде наборов данных. Этот процесс исполняется как одна из служб операционной системы Windows.
Интерфейс InSQLRetrieve	Интерфейс службы низкоуровневого считывания, через который провайдер InSQL OLEDB и службы MDAS получают данные из архивных блоков.
Интерфейс InSQLHistory	Интерфейс менеджера конфигурирования (Configuration Manager), при помощи которого провайдер InSQL OLEDB и службы MDAS выполняют с данными действия, не относящиеся к извлечению.
Служба ручного ввода данных MDAS	Процесс, выполняющий извлечение и вставку данных, а также действия по конфигурированию, такие как создание тэгов. К нему могут обращаться провайдер InSQL OLEDB, подсистема событий и клиентские приложения.
Провайдер InSQL OLEDB	Компонент программного обеспечения сервера SQL Server, который используется для извлечения данных из архивных блоков. Провайдер InSQL OLEDB представляет данные клиентским приложениям так, словно они хранятся в обычных таблицах SQL Server.
Расширения IndustrialSQL Server для поддержки временных параметров	Специальные расширения синтаксиса операторов Transact-SQL, позволяющие указывать в запросах интервалы времени.
Сервер в/в InSQL (aahIOSvrSvc.exe)	Внутренний процесс представления клиентам текущих значений тэгов, хранящихся в Активном образе.
Запрашивающие приложения	Приложения с командным или графическим интерфейсом, которые могут подключаться к Microsoft SQL Server и извлекать данные с помощью операторов Transact-SQL.

Подробно об общей архитектуре системы архиватора см. в Главе 1 настоящего руководства.

Характеристики подсистемы извлечения данных

Ниже перечислены некоторые основные характеристики подсистемы извлечения данных:

- В одном и том же запросе на данные из таблицы History могут быть указаны тэги любых поддерживаемых типов. Выдавать отдельные запросы на значения логических, аналоговых и символьных тэги нет никакой необходимости. В запросах могут быть указаны любые сочетания тэгов.
- Текущие значения их Активного образа передаются независимо от типа тэгов в списке, если только в запросе не была указана конструкция ORDER BY.
- Поддерживаются строки как фиксированной, так и переменной длины.

- Все внутренние операции с показаниями времени выполняются с использованием типа Win32 FILETIME. Разрешение данных (точность) типа FILETIME по времени составляет 100 наносекунд.
- Внутренний формат всех показаний времени – UTC. Преобразования в местное время и из него выполняются при извлечении данных, поэтому пользователь может указывать в запросах местное время.
- Реализована поддержка неоперативных данных (например данных, передаваемых в режиме с промежуточным хранением или импортируемых из CSV-файлов).
- Поддерживается извлечение данных разных версий.

Архивные блоки как удалённые источники данных SQL Server

Удалённые источники представляют собой хранилища информации, существующие вне файла базы данных Microsoft SQL Server (с расширением .MDF). Иногда в своей документации Microsoft называет их "нелокальными источниками данных". Для архиватора IndustrialSQL Server удалённый источник данных представляет собой набор файлов архивного блока. В архивных блоках хранятся все сведения о тэгах. Подробнее см. раздел "Архивные блоки" настоящего руководства.

Обращение к данным в удалённых хранилищах данных может быть выполнено по технологии OLEDB. В этих случаях используется программный компонент, называемый провайдером OLEDB.

Низкоуровневое извлечение данных

Служба низкоуровневого извлечения данных осуществляет поиск затребованной информации как в Активном образе в оперативной памяти, так и в архивных блоках на диске. При этом она форматирует данные так, что они могут быть переданы провайдеру InSQL OLEDB и любым другим клиентским приложениям со службой MDAS. Кроме того, в её задачи входит предоставление различныой информации об архивных блоках (например начальные и конечные даты, местоположение и т.д.)

Дополнительно эта служба обеспечивает высокоскоростную передачу данных MDAS-клиентам, для которых не нужен Transact-SQL с его гибкими и мощными возможностями. Служба низкоуровневого извлечения данных возвращает точные метки времени и поддерживает все расширения IndustrialSQL Server во временной области.

Масштабирование данных при низкоуровневом извлечении

При низкоуровневом извлечении значения аналоговых тэгов подвергаются линейному масштабированию по следующей формуле:

V_out = (V_in - MinRaw)*(MaxEU - MinEU)/(MaxRaw - MinRaw) + MinEU

где:

V_out = масштабированное выходное значение

V_in = сохранённое исходное значение

MinRaw = минимальное допустимое исходное значение тэга

MaxRaw = максимальное допустимое исходное значение тэга

MinEU = минимальное допустимое значение тэга в единицах измерения

MaxEU = минимальное допустимое значение тэга в единицах измерения

Провайдер InSQL OLEDB

OLEDB (Object Linking and Embedding for Databases – технология связывания и встраивания объектов для баз данных) представляет собой прикладной API-интерфейс, позволяющий клиентским приложениям, поддерживающим модель COM, обращаться к данным, которые физически не хранятся на сервере Microsoft SQL Server.

Достоинство этой технологии заключается в том, что она предоставляет свободный доступ к данным самых разных типов. С её помощью можно одновременно взаимодействовать с несколькими разнотипными источниками, например с базами данных Microsoft SQL Server, Oracle и Microsoft Access. Запрос на данные из нескольких несходных источников называется "гетерогенным" (в данном случае "гетерогенность" означает разнородность). Гетерогенный запрос можно также назвать распределённым, потому что возвращаемые данные могут находиться в разных источниках.

Технология OLEDB используется в сервере Microsoft SQL Server для выполнения распределённых запросов, существенно облегчая связывание данных из различных источников. Благодаря ей Microsoft SQL Server может выполнять запросы на языке Transact-SQL на получение данных, хранящихся в одной или нескольких гетерогенных базах и базах SQL Server, без какой-либо необходимости в специализированных приложениях-шлюзах.

Интерфейс доступа к нелокальным информационным хранилищам, таким как архивные блоки IndustrialSQL Server, обеспечивается "виртуальным" сервером, называемым провайдером InSQL OLEDB. Использование провайдеров OLEDB существенно облегчает связывание данных из базы Microsoft SQL Server и архивных блоков IndustrialSQL Server. Кроме того, провайдер OLEDB сервера IndustrialSQL Server обладает весьма широкими возможностями построения запросов.

Имя провайдера OLEDB сервера IndustrialSQL Server – INSQL. Провайдер INSQL OLEDB устанавливается одновременно с архиватором IndustrialSQL Server и затем связывается с сервером Microsoft SQL Server. Подробно о синтаксисе связывания провайдера INSQL OLEDB см. в разделе "Связывание провайдера INSQL OLEDB с сервером Microsoft SQL Server" настоящего руководства.

Чтобы обратиться к данным архиватора IndustrialSQL Server по технологии OLEDB, клиентское приложение должно установить соединение непосредственно с сервером Microsoft SQL Server и указать использование провайдера INSQL OLEDB в синтаксисе запроса.

При указании в синтаксисе запроса ссылки на провайдера INSQL OLEDB синтаксический анализатор сервера Microsoft SQL Server передаёт соответствующие части запроса данных провайдеру OLEDB INSQL Server. Этот провайдер обращается к службе низкоуровневого извлечения, которая находит и считывает затребованную информацию, после чего возвращает полученные результаты Microsoft SQL Server как набор строк (rowset). Microsoft SQL Server завершает требуемую обработку данных и возвращает их клиентскому приложению как набор результатов и набор выходных параметров (если это применимо).

Провайдер INSQL OLEDB должен исполняться на компьютере с функционирующим сервером Microsoft SQL Server. Перечень операций

Transact-SQL извлечения данных из архивных блоков зависит от характеристики провайдера INSQL OLEDB. Провайдер INSQL OLEDB соответствует спецификациям SQL-92.

Дополнительно о технологии OLEDB см. в документации Microsoft.

Таблицы расширения для хранения исторических данных

Некоторые из таблиц архиватора служат для доступа к накопленным производственным данным. Они не входят в число обычных таблиц SQL Server. Обычные таблицы SQL Server хранятся непосредственно в файлах базы данных (.mdf). Таблицы расширения только представляют данные так, словно они хранятся в реальных таблицах, физически при этом в устройстве базы данных не находясь. Таблицы расширения представляют собой логические таблицы, заполняемые данными из файлов других типов; так что в этом смысле они являются для сервера SQL Server "удалёнными". В качестве файлов другого типа здесь выступают архивные блоки сервера IndustrialSQL Server, создаваемые его подсистемой хранения.

Примечание. Таблицы расширения называются также удалёнными таблицами.

Обращение к архивным блокам IndustrialSQL Server осуществляется с помощью провайдера OLEDB сервера SQL Server. Клиентское приложение должно установить соединение непосредственно с сервером Microsoft SQL Server и указать использование провайдера INSQL OLEDB в синтаксисе запроса.

В число таблиц расширения входят следующие таблицы:

- History (INSQL.Runtime.dbo.History).
- HistoryBlock (INSQL.Runtime.dbo.HistoryBlock).
- Live (INSQL.Runtime.dbo.Live).
- WideHistory (INSQL.Runtime.dbo.WideHistory).

Таблицы AnalogHistory, DiscreteHistory, StringHistory, AnalogLive, DiscreteLive, StringLive, AnalogWideHistory, DiscreteWideHistory и StringWideHistory включены для обеспечения обратной совместимости. Дополнительно см. Главу 6 "Объекты для обратной совместимости" Справочника по базе данных IndustrialSQL Server™ Historian Database Reference.

Обновление данных возможно только в таблицах AnalogHistory, DiscreteHistory, StringHistory и History. Остальные являются таблицами "только для чтения".

Подробне о таблицах расширения см. также в Главе 1 "Категории таблиц" Справочника по базе данных IndustrialSQL Server™ Historian Database Reference.

Синтаксис запроса со ссылкой на провайдера INSQL OLEDB

Наиболее часто используемым запросом к архиватору IndustrialSQL Server является оператор SELECT следующего вида:

```
SELECT список_выбора
    FROM список_таблиц
    WHERE критерий_поиска
       [ GROUP BY выражение_группировки ]
```

```
[ HAVING условие_поиска ]
[ ORDER BY порядок_сортировки [ ASC | DESC ] ]
```

Конструкция WHERE обязательна, если запрос выдан к любой из таблиц расширения, за исключением таблицы HistoryBlock.

Существует четыре варианта оператора SELECT для извлечения данных с помощью провайдера INSQL OLEDB:

- в четырёхкомпонентном синтаксисе;
- с использованием представлений провайдера INSQL OLEDB;
- с использованием функции OPENQUERY;
- с использованием функции OPENROWSET.

При возможности следует всегда создавать запросы в четырёхкомпонентном синтаксисе, хотя в некоторых случаях использование функций OPENQUERY и OPENROWSET может оказаться обязательным, например при обращении к расширенным (wide) таблицам.

Об SQL-запросах см. документацию по Microsoft SQL Server.

Четырёхкомпонентный синтаксис

Имя связанного сервера представляет собой имя, под которым провайдер INSQL OLEDB известен Microsoft SQL Server. Чтобы запрос был передан провайдеру, нужно указать имя связанного сервера и имя таблицы расширения в четырёхкомпонентном названии.

В частности, следующий запрос выдан на получение данных из таблицы расширения History провайдера INSQL OLEDB:

```
SELECT * FROM INSQL.Runtime.dbo.History
    WHERE TagName = 'SysTimeSec'
          AND DateTime >= '2001-09-12 12:59:00'
          AND DateTime <= '2001-09-12 13:00:00'
```

Структура четырёхкомпонентного названия описана в следующей таблице:

Часть названия	Описание
связанный сервер	Имя связанного сервера (по умолчанию INSQL).
каталог	Каталог источника данных OLEDB с объектом, из которого нужно извлечь данные. Для баз данных типа Microsoft SQL Server – имя базы данных. При обращении к провайдеру INSQL OLEDB в качестве каталога всегда должно быть указано 'Runtime'.
схема	Схема в каталоге, содержащем объект. Для баз данных типа Microsoft SQL Server – регистрационный идентификатор. При обращении к провайдеру INSQL OLEDB в качестве схемы всегда должно быть указано 'dbo'.
имя объекта	Имя объекта данных, который может быть представлен провайдером OLEDB в виде набора строк (rowset). Для провайдера INSQL OLEDB – имя удалённой таблицы с требуемыми данными (например History).

При использовании четырёхкомпонентного синтаксиса SQL Server создаёт запросы, которые передаются провайдеру OLEDB, на основе оператора, введённого пользователем. Бывают случаи, когда оператор указан неверно, либо слишком сложен, либо не имеет конструкции WHERE, обязательной при обращении к провайдеру INSQL OLEDB.

Типичное сообщение об ошибке, выдаваемое при указании оператора с неподдерживаемым синтаксисом, выглядит следующим образом:

```
Server: Msg 7320, Level 16, State 2, Line 1

Could not execute query against OLE DB provider
'INSQL'.

[OLE/DB provider returned message: InSQL did not receive a

    WHERE clause from SQL Server. If one was specified,

    refer to the InSQL OLE DB documentation].

(Сервер: сообщение 7320, уровень 16, состояние 2,
строка 1

Невозможно выполнить запрос к провайдеру OLEDB:
'INSQL'

Провайдер OLEDB возвратил сообщение: InSQL не получил
конструкцию WHERE от сервера SQL Server. Если она была
указана, обратитесь к документации по InSQL OLEDB.)
```

В четырёхкомпонентных запросах к серверам SQL Server с языками, отличными от английского, исполняющихся в не английских версиях операционных систем, формат даты по умолчанию может отличаться от формата для английских систем. Например, для французской или немецкой версии сервера SQL Server, исполняющегося в соответствующей операционной системе, дата и время суток в четырёхкомпонентном запросе должны быть указаны в следующем формате:

гггг-дд-мм чч:мм:сс.ммм

Например:

2003-28-09 09:00:00.000

Использование представлений провайдера InSQL OLEDB

В базе данных Microsoft SQL Server уже определены представления, которые обеспечивают доступ к таблицам расширения без необходимости указания в запросе четырёхкомпонентного названия сервера. Названия этих представлений совпадают с названиями таблиц провайдера.

Примечание. Представления, созданные для обратной совместимости, именуются в соответствии с правилом: *v_имя_таблицы_провайдера*.

Например:

```
SELECT * FROM History

    WHERE TagName = 'SysTimeSec'

      AND DateTime >= '2001-09-12 12:59:00'

      AND DateTime <= '2001-09-12 13:00:00'
```

Использование функции OPENQUERY

Извлечь данные из таблицы расширения можно с помощью функции OPENQUERY, в аргументах которой указано имя связанного сервера. Вызов функции OPENQUERY выполняется при обращении к расширенным таблицам, например:

```
SELECT * FROM OPENQUERY(INSQL, 'SELECT *

    FROM History

      WHERE TagName = "SysTimeSec"
```

```
        AND DateTime >= "2001-09-12 12:59:00"
        AND DateTime <= "2001-09-12 13:00:00"
')
```

Следующий оператор извлекает данные из расширенной таблицы::

```
SELECT * FROM OPENQUERY(INSQL, 'SELECT DateTime,
SysTimeSec
    FROM WideHistory
      WHERE DateTime >= "2001-09-12 12:59:00"
        AND DateTime <= "2001-09-12 13:00:00"
')
```

Обращение к функции OPENQUERY в операторе SELECT интерпретируется сервером SQL Server как обращение к таблице, поэтому её можно указывать в операторах объединения, в представлениях и в хранимых процедурах. SQL Server пересылает указанный в кавычках вызов функции как строку символов провайдеру INSQL OLEDB. Поэтому в записи должен быть использован тот синтаксис, который поддерживается провайдером INSQL OLEDB. Кроме того, длина оператора не должна превышать 8000 символов. Рассмотрим следующий пример:

```
SELECT * FROM OpenQuery(INSQL, 'XYZ')
```

где XYZ – передаваемый оператор. При выполнении оператора SELECT нужно, чтобы длина строки XYZ не превышала 8000 символов. Обычно соблюдать выполнение этого ограничения приходится при запросе множества тэгов из нескольких расширенных таблиц.

Кроме того, показания даты и времени должны быть указаны в вызове функции OPENQUERY в следующем формате:

гггг-дд-мм чч:мм:сс.ммм

Например:

2001-01-01 09:00:00.000

Использование переменных в вызове функции OPENQUERY не допускается. Подробнее см. раздел "Использование переменных в запросах к расширенным таблицам" настоящего руководства.

Использование функции OPENROWSET

Имя связанного сервера может быть указано как один из входных аргументов функции OPENROWSET. Эта функция используется для передачи провайдеру OLEDB некоторой команды на исполнение. Возвращаемый в результате её выполнения набор строк может быть использован в операторе Transact-SQL как ссылка на таблицу или представление. Например:

```
SELECT * FROM OPENROWSET('INSQL', ' ', 'SELECT
DateTime, Quality, QualityDetail, Value
    FROM History
      WHERE TagName in ("SysTimeSec")
        AND DateTime >= "2001-09-12 12:59:00"
        AND DateTime <= "2001-09-12 13:00:00"
')
```

Допустимые конструкции в записи запроса

В следующей таблице приведены сведения о том, какие конструкции допустимы в записи четырёхкомпонентного названия (или в соответствующем представлении) и в вызове функции OPENQUERY.

Конструкция	Запрос с четырёхкомпонентным названием	Запрос с вызовом функции OPENQUERY
ORDER BY	Да.	Нет. Недействителен в вызове функции OPENQUERY. Действителен как конструкция оператора.
GROUP BY	Да.	Нет.
TagName IN (…)	Да.	Да.
TagName LIKE '…'	Да.	Да.
Функции даты и времени (например, DateAdd())	Да.	Да.
MIN, MAX, AVG, SUM, STDEV	Да.	Только MIN, MAX, AVG, SUM.
Вложенный оператор SELECT с одной обычной таблицей SQL Server и одной таблицей расширения	Да, с некоторыми ограничениями.	Нет.
Вложенный оператор SELECT с двумя таблицами расширения	Нет.	Нет.

Неподдерживаемый синтаксис и ограничения провайдера INSQL OLEDB

Провайдер InSQL OLEDB не поддерживает некоторые виды запросов. В целом, эти ограничения являются следствием соответствующих ограничений текущей реализации провайдера OLEDB сервера Microsoft SQL Server.

Подробнее сведения об SQL-запросах см. в документации к Microsoft SQL Server.

Отсутствие понятия контекста клиента

Понятие контекста клиента в провайдере OLEDB отсутствует. Провайдер OLEDB не меняет своего состояния, параметры запросов в рамках одного и того же соединения не сохраняются. Это значит, что при каждом исполнении запроса нужно заново устанавливать значение параметров временных расширений IndustrialSQL (например счётчика подынтервалов).

Кроме того, провайдер OLEDB не может возвращать данные непрерывно (аналогично "горячей" линии связи в приложениях InTouch). Спецификация OLEDB (определённая компанией Microsoft) запрещает провайдеру возвращать потребителю строки при отсутствии с его стороны соответствующего запроса.

Ограничения на использование расширенных таблиц

Расширенные таблицы не имеют фиксированной структуры, их структура меняется от запроса к запросу. Они являются кратковременными таблицами, существующими в течение выполнения только одного запроса. Именно поэтому обращение к ним должно выполняться с помощью функции OPENQUERY, которая пропускает многие проверки, связанные с фиксированными таблицами. Расширенные таблицы могут иметь до 1024 столбцов.

Подробнее о расширенных таблицах см. в разделе "Формат расширенных таблиц" Главы 1 Справочника по базе данных IndustrialSQL Server™ Historian Database Reference.

Ограничения на использование конструкции LIKE

Конструкция LIKE допустима только для столбцов TagName и Value. Синтаксис " ... Value LIKE 'строка_символов' ..." поддерживается только для таблиц символьных строк. Пример использования:

```
SELECT TagName, Value FROM History
    WHERE TagName LIKE 'Sys%'
       AND DateTime > '1999-05-24 14:30:00'
       AND DateTime < '1999-05-24 14:32:00'
```

Ограничения на использование конструкции IN

В запросах значений аналоговых, логических и символьных тэгов из таблиц AnaloTag, DiscreteTag и StringTag соответственно конструкцию LIKE внутри конструкции IN для формирования имени тэга указывать нельзя, если только не возвращается значение столбца vValue. Данное ограничение действительно также для запросов с четырёхкомпонентным названием и для представлений таблиц расширения.

Пример оператора:

```
SELECT DateTime, TagName, vValue, Quality,
QualityDetail
    FROM History
       WHERE TagName IN (SELECT TagName FROM StringTag
       WHERE TagName LIKE 'SysString')
           AND DateTime >='2001-06-21 16:00:00.000'
           AND DateTime <='2001-06-21 16:40:00.000'
           AND wwRetrievalMode = 'Delta'
```

Однако более эффективным способом для достижения таких же результатов будет использование конструкции INNER REMOTE JOIN. Подробнее см. раздел "Использование конструкции INNER REMOTE JOIN" настоящего руководства.

Ограничения на использование конструкции OR

Конструкцию OR для указания более чем одного условия в запросах с указанием времени использовать нельзя. Подробнее см. раздел "Время в запросах на данные" настоящего руководства.

Использование конструкции JOIN в вызове функции OPENQUERY

Объединения в вызовах OPENQUERY не поддерживаются. Например, следующий запрос определяет неявное объединение таблиц Ta и Live и потому не будет выполняться:

```
SELECT * FROM OPENQUERY(INSQL, 'SELECT v.DateTime,
v.TagName, v.Value, t.Description
    FROM Tag t, Live v
      WHERE t.TagName LIKE "%Date%"
          AND v.TagName = t.TagName
    ')
```

Чтобы исправить ошибку, нужно вынести объединение из вызова функции, например таким образом:

```
SELECT v.DateTime, v.TagName, v.Value, t.Description
FROM OPENQUERY(INSQL, 'SELECT DateTime, TagName, Value
    FROM Live
      WHERE TagName LIKE "%Date%"
    ') v, Tag t
    WHERE v.TagName = t.TagName
```

Явные объединения также не поддерживаются внутри вызова функции OPENQUERY. Например, следующий оператор является неверным:

```
SELECT * FROM OPENQUERY(INSQL, 'SELECT v.DateTime,
v.TagName, v.Value, e.Unit
    FROM Live v
      JOIN AnalogTag t ON v.TagName = t.TagName
      JOIN EngineeringUnit e ON t.EUKey = e.EUKey
          WHERE v.TagName LIKE "%Date%"
    ')
```

Чтобы исправить ошибку, нужно вынести объединение из вызова функции, например таким образом:

```
SELECT v.DateTime, v.TagName, v.Value, e.Unit
    FROM OPENQUERY(INSQL, 'SELECT DateTime, TagName,
    Value
      FROM Live
      WHERE TagName LIKE "%Date%"
    ') v
    JOIN AnalogTag t ON v.TagName = t.TagName
    JOIN EngineeringUnit e ON t.EUKey = e.EUKey
      ORDER BY t.TagName
```

В целом, рекомендуется при возможности записывать запросы в четырёхкомпонентном формате. Все предыдущие запросы становятся более понятными при записи в таком виде. Например, предыдущий оператор будет выглядеть следующим образом:

```
SELECT v.DateTime, v.TagName, v.Value, e.Unit
    FROM INSQL.Runtime.dbo.History v
    JOIN AnalogTag t ON v.TagName = t.TagName
    JOIN EngineeringUnit e ON t.EUKey = e.EUKey
      WHERE v.TagName LIKE '%Date%'
      ORDER BY t.TagName
```

Сложные объединения в запросах

В запросах допускаются только простые объединения таблиц SQL Server и таблиц расширения OLEDB IndustrialSQL Server. Как правило, для этого требуется использование конструкции INNER REMOTE JOIN.

Подробнее см. раздел "Использование конструкции INNER REMOTE JOIN" настоящего руководства.

Вложенные операторы SELECT в запросах к таблицам SQL Server и к таблицам расширения IndustrialSQL Server

Следует избегать использования вложенных операторов SELECT в запросах к обычным таблицам SQL Server и таблицам расширения IndustrialSQL Server (поскольку они крайне неэффективны из-за особенностей их исполнения сервером SQL Server). Пример:

```
SELECT TagName, DateTime, Value
    FROM INSQL.Runtime.dbo.History
      WHERE TagName IN (select TagName FROM
      SnapshotTag
          WHERE EventTagName = 'SysStatusEvent')
          AND DateTime = '2001-12-20 0:00'
```

Вместо этого оператора рекомендуется воспользоваться конструкцией INNER REMOTE JOIN:

```
SELECT h.TagName, DateTime, Value
    FROM SnapshotTag st INNER REMOTE JOIN
      INSQL.Runtime.dbo.History h
      ON st.TagName = h.TagName
          AND EventTagName = 'SysStatusEvent'
          AND DateTime = '2001-12-20 0:00'
```

Результаты выполнения этого оператора будут выглядеть следующим образом:

TagName	DateTime	Value
SysPerfCPUTotal	2001-12-20 00:00:00.000	15.0
SysSpaceMain	2001-12-20 00:00:00.000	1302.0

В целом, конструкция INNER REMOTE JOIN используется в запросах к архиватору IndustrialSQL Server следующим образом:

```
Имя_таблицы_SQL_Server INNER REMOTE JOIN
Имя_таблицы_расширения_архиватора
```

Дополнительно об этой конструкции см. документацию Microsoft.

Ошибки обработки конструкции WHERE

В некоторых (очень редких) случаях процессор запросов SQL Server усекает конструкцию WHERE, стремясь оптимизировать выполнение запроса. Если при выполнении запроса с этой конструкцией будет выдано сообщение о том, что в запросе её нет, просто добавьте в запрос ещё одно условие.

Например, при выполнении следующего запроса процессор SQL Server отбросит конструкцию WHERE, потому что посчитает её избыточной:

```
SELECT DateTime, Value, QualityDetail
    FROM History
          WHERE TagName LIKE '%'
```

Чтобы исправить ошибку, добавьте в запрос ещё одно условие, например:

```
SELECT DateTime, Value, QualityDetail
    FROM History
      WHERE TagName LIKE '%'
      AND wwRetrievalMode = 'delta'
```

Ограничения на использование функции CONVERT

Функция CONVERT с аргументом vValue не поддерживается в вызове функции OPENQUERY. Чтобы извлечь данные из таблицы History с помощью вызова функции OPENQUERY, нужно произвести фильтрацию значений столбца vValue вне запроса.

В следующем примере значение столбца vValue преобразуется в число с плавающей запятой. Следует отметить, что в запросе не указан ни один символьный тэг.

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime,
Quality, OPCQuality, QualityDetail, Value, vValue  ,
TagName
    FROM History
      WHERE TagName IN ("SysTimeMin", "SysPulse")
          AND DateTime >= "2001-12-30 04:00:00.000"
          AND DateTime <= "2001-12-30 09:00:00.000"
          AND wwRetrievalMode = "Delta"
    ')
    WHERE convert(float, vValue) = 20.0
```

Допускается использование следующего синтаксиса:

```
WHERE convert(float, vValue) = 0
WHERE convert(float, vValue) = 0.0
WHERE convert(float, vValue) = 1.0
WHERE convert(float, vValue) = 1
WHERE convert(float, vValue) = 20
```

```
WHERE convert(float, vValue) = 2.0000e01
```

В следующем запросе указан символьный тэг, также в нём показано, как можно осуществить преобразование значения столбца vValue в величину типа char или varchar. Все возвращаемые величины могут быть преобразованы в символьный тип.

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime,
Quality, OPCQuality, QualityDetail, Value, vValue,
TagName

    FROM History

        WHERE TagName IN ("SysString", "SysTimeMin",
        "SysPulse")

           AND DateTime >= "2001-12-30 04:00:00.000"

           AND DateTime <= "2001-12-30 09:00:00.000"

           AND wwRetrievalMode = "Cyclic"

           AND wwCycleCount = 300

    ')

    WHERE convert(varchar(30), vValue) = '2001-12-30
    14:00:00'
```

Допускается указывать следующий синтаксис:

```
WHERE convert(varchar(30), vValue) = '20'

WHERE convert(varchar(30), vValue) = '1'

WHERE convert(varchar(30), vValue) = '0'
```

OPENQUERY и Microsoft Query

Microsoft Query не способен обрабатывать оператор OPENQUERY.

Привязка провайдера INSQL OLEDB к Microsoft SQL Server

Поскольку провайдер INSQL OLEDB извлекает данные их архивных блоков IndustrialSQL Server и представляет их Microsoft SQL Server в виде таблицы, он сам является своего рода сервером. Прежде чем пользоваться им для обработки запросов, нужно добавить его к Microsoft SQL Server как "связанный" сервер.

Данная привязка выполняется автоматически во время установки архиватора IndustrialSQL Server. Если впоследствии потребуется повторно привязать провайдера INSQL OLEDB к северу Microsoft SQL Server, выполните следующую последовательность команд:

```
sp_addlinkedserver

@server = 'INSQL',

@srvproduct = '',

@provider = 'INSQL'

go

sp_serveroption 'INSQL', 'collation compatible', true

go

sp_addlinkedsrvlogin 'INSQL', 'TRUE', NULL, NULL, NULL

go
```

Здесь 'INSQL' – имя связываемого с сервером Microsoft SQL Server провайдера INSQL OLEDB. Именно его следует указывать в запросах.

Чтобы впоследствии можно было выполнять операцию объединения с унаследованными архивными таблицами аналоговых и логических значений, программа установки создаёт альтернативное имя того же самого провайдера INSQL OLEDB:

```
sp_addlinkedserver
@server = 'INSQLD',
@srvproduct = '',
@provider = 'INSQL'
go
sp_serveroption 'INSQLD', 'collation compatible', true
go
sp_addlinkedsrvlogin 'INSQLD', 'TRUE', NULL, NULL,
NULL
go
```

Для выполнения запроса, в котором указано объединение таблиц аналоговых и логических значений, для первой таблицы (аналоговых значений) следует указать обычное имя провайдера, а для второй (логических значений) – имя с буквой "D".

Время в запросах на данные

Обработка данных в таблицах расширения может быть выполнена с помощью как обычных операторов языка Transact-SQL, так и специальных расширений архиватора IndustrialSQL Server, предназначенных для использования параметров времени в запросах на данные из архивных таблиц. Эти возможности не поддерживаются языком Transact-SQL.

В число указанных расширений входят следующие элементы:

- wwCycleCount
- wwResolution
- wwRetrievalMode
- wwTimeDeadband
- wwValueDeadband
- wwEdgeDetection
- wwTimeZone
- wwVersion
- wwInterpolationType
- wwTimeStampRule
- wwQualityRule

Примечание. Параметр wwParameters зарезервирован для использования в будущем. Параметр wwRowCount поддерживается, но постепенно заменяется параметром wwCycleCount.

Указанные параметры расширения реализованы как "виртуальные" столбцы таблиц расширения. Присваивание им тех или иных значений в запросе позволяет уточнять критерии выборки данных из архивных блоков

по временным характеристикам. Указывать значения этих столбцов нужно каждый раз при выполнении очередного запроса.

Например, использование столбца wwResolution позволяет задавать разрешение данных по времени:

```
SELECT DateTime, Value
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND DateTime >= '2001-12-02 10:00:00'
          AND DateTime <= '2001-12-02 10:02:00'
          AND Value >= 50
          AND wwResolution = 10
          AND wwRetrievalMode = 'cyclic'
```

Поскольку таблицы расширения обеспечивают дополнительны возможности, не поддерживаемые стандартными серверами SQL Server, при написании операторов Transact-SQL нужно соблюдать определённые условия. Подробнее см. раздел Неподдерживаемый синтаксис и ограничения провайдера INSQL OLEDB" настоящего руководства.

Несмотря на то что Microsoft SQL Server может быть чувствителен к регистру вводимых символов, значения для виртуальных столбцов таблиц расширения могут указываться в любом регистре.

Примечание. Конструкции IN и OR для указания более чем одного условия, связанного с временными характеристиками данных, не поддерживаются. Например, записи типа "wwVersion IN ('original', 'latest')" и "wwRetrievalMode = 'Delta' OR wwVersion = 'latest'" недопустимы.

Подробно об SQL-запросах см. в документации Microsoft.

Параметр wwCycleCount

Счётчик подынтервалов представляет собой количество строк, которые должны извлекаться из архивных таблиц IndustrialSQL Server. Счётчик подынтервалов архиватора IndustrialSQL Server отличается от аналогичного элемента сервера SQL Server (при циклическом извлечении). Способ его применения зависит от метода извлечения – циклического или по изменению. Подробнее см. раздел "Использование параметров wwResolution, wwCycleCount и wwRetrievalMode в одних и тех же запросах" настоящего руководства.

Применение счётчика подынтервалов зависит также от того, выполняется ли запрос к расширенной (wide) таблице или нет. В запросе данных из таблицы History (укороченная таблица) в циклическом режиме (то есть когда параметр wwRetrievalMode имеет значение 'cyclic') счётчик подынтервалов задаёт количество строк, возвращаемых для каждого тэга. Например, запрос значений двух тэгов, счётчик подынтервалов в котором установлен в 20, возвратит 40 строк (по 20 значений каждого тэга). В запросе к расширенной таблице WideHistory счётчик подынтервалов определяет общее количество возвращаемых строк независимо от числа указанных тэгов. Например, запрос значений двух тэгов, счётчик подынтервалов в котором установлен в 20, возвратит 20 строк данных.

Следующий пример иллюстрирует использование счётчиков подынтервалов в запросах значений одного или нескольких тэгов.

Запрос значений одного тэга:

```
SELECT DateTime, TagName, Value
    FROM History
```

```
    WHERE TagName = 'SysTimeSec'
        AND DateTime >= '2001-12-09 11:35'
        AND DateTime < '2001-12-09 11:36'
        AND wwRetrievalMode = 'Cyclic'
        AND wwCycleCount = 300
```

Возвращаемые результаты:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0.0
2001-12-09 11:35:00.200	SysTimeSec	0.0
2001-12-09 11:35:00.400	SysTimeSec	0.0
2001-12-09 11:35:00.600	SysTimeSec	0.0
…		
2001-12-09 11:35:59.200	SysTimeSec	59.0
2001-12-09 11:35:59.400	SysTimeSec	59.0
2001-12-09 11:35:59.600	SysTimeSec	59.0
2001-12-09 11:35:59.800	SysTimeSec	59.0

Запрос значений нескольких тэгов:

```
SELECT DateTime, TagName, Value
    FROM History
      WHERE TagName IN ('SysTimeMin', 'SysTimeSec')
          AND DateTime >= '2001-12-09 11:35'
          AND DateTime < '2001-12-09 11:36'
          AND wwRetrievalMode = 'Cyclic'
          AND wwCycleCount = 300
```

Возвращаемые результаты:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeMin	35.0
2001-12-09 11:35:00.000	SysTimeSec	0.0
2001-12-09 11:35:00.200	SysTimeMin	35.0
2001-12-09 11:35:00.200	SysTimeSec	0.0
2001-12-09 11:35:00.400	SysTimeMin	35.0
2001-12-09 11:35:00.400	SysTimeSec	0.0
2001-12-09 11:35:00.600	SysTimeMin	35.0
2001-12-09 11:35:00.600	SysTimeSec	0.0
…		
2001-12-09 11:35:59.000	SysTimeMin	35.0
2001-12-09 11:35:59.000	SysTimeSec	59.0
2001-12-09 11:35:59.200	SysTimeMin	35.0
2001-12-09 11:35:59.200	SysTimeSec	59.0

2001-12-09 11:35:59.400	SysTimeMin	35.0
2001-12-09 11:35:59.400	SysTimeSec	59.0
2001-12-09 11:35:59.600	SysTimeMin	35.0
2001-12-09 11:35:59.600	SysTimeSec	59.0

Следует отметить, что значения всех тэгов будут выводиться в одном и том же столбце таблицы результатов.

Параметр wwResolution

Разрешение данных представляет собой интервал выборки данных в миллисекундах из архивных таблиц IndustrialSQL Server. Эта возможность поддерживается только архиватором IndustrialSQL Server и недоступна в стандартных серверах SQL Server.

Для извлечения данных с указанным разрешением по времени в запросе нужно указать общий временной диапазон. Система возвратит только те значения, которые попадают в этот диапазон и разнесены по времени на указанный интервал. Например, если разрешение по времени составляет 5000 мс, система извлечёт из архива все данные, попадающие в указанный период времени, и возвратит только те, которые отделены друг от друга интервалом в 5000 мс. Количество возвращаемых строк таблицы результатов определяется длительностью временного диапазона и разрешающей способностью. То есть количество строк = общий период времени / разрешение данных.

Разрешение действительно только для режима циклического извлечения, однако его аналог можно определить и для режима извлечения по изменению, указав соответствующую мёртвую зону по времени.

Дополнительно об использовании этого параметра с другими см. в разделе "Использование параметров wwResolution, wwCycleCount и wwRetrievalMode в одних и тех же запросах" настоящего руководства.

Примечание. В наборе строк будет содержаться по одной строке для каждого тэга в нормализованном запросе в каждом интервале разрешения независимо от того, существует ли в архиве реальное значение, соответствующее этому моменту времени. Значение, приведённое в строке результатов, будет представлять собой последнее известное значение тэга на данный момент.

Следующий запрос возвращает данные, разнесённые по времени на 2 секунды (2000 мс) в указанном периоде времени. Режим извлечения данных – циклический.

```
SELECT DateTime, TagName, Value
    FROM History
       WHERE TagName IN ('SysTimeMin', 'SysTimeSec')
           AND DateTime >= '2001-12-09 11:35'
           AND DateTime <= '2001-12-09 11:36'
           AND wwRetrievalMode = 'Cyclic'
           AND wwResolution = 2000
```

Результаты:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeMin	35.0
2001-12-09 11:35:00.000	SysTimeSec	0.0

2001-12-09 11:35:02.000	SysTimeMin	35.0
2001-12-09 11:35:02.000	SysTimeSec	2.0
2001-12-09 11:35:04.000	SysTimeMin	35.0
2001-12-09 11:35:04.000	SysTimeSec	4.0
2001-12-09 11:35:06.000	SysTimeMin	35.0
…		
2001-12-09 11:35:54.000	SysTimeMin	35.0
2001-12-09 11:35:54.000	SysTimeSec	54.0
2001-12-09 11:35:56.000	SysTimeMin	35.0
2001-12-09 11:35:56.000	SysTimeSec	56.0
2001-12-09 11:35:58.000	SysTimeMin	35.0
2001-12-09 11:35:58.000	SysTimeSec	58.0
2001-12-09 11:36:00.000	SysTimeMin	36.0
2001-12-09 11:36:00.000	SysTimeSec	0.0

Параметр wwRetrievalMode

Режим извлечения определяет набор возвращаемых данных.

Режим циклического извлечения

Циклическое извлечение представляет собой извлечение из архива данных, приходящихся на указанный период времени и разнесённых на указанный интервал временного разрешения, независимо от того, менялось ли значение тэгов или нет. При выполнении запроса с циклическим режимом создаётся виртуальный набор строк, который может содержать или не содержать реальные строки. Метки времени данных, записанных в архив в режиме сохранения по изменению и извлекаемых в циклическом режиме, будут определяться так, как если бы значения тэгов сохранялись циклически с интервалом, длительность которого указана в запросе. По умолчанию, данные из таблиц аналоговых значений извлекаются в циклическом режиме.

Циклический режим задаётся следующим образом:

```
wwRetrievalMode = 'Cyclic'
```

Количество возвращаемых строк можно также контролировать с помощью параметров wwCycleCount и wwResolution. Использование счётчика подынтервалов и разрешения в режиме циклического извлечения даёт различные результаты. Подробнее см. раздел "Использование параметров wwResolution, wwCycleCount и wwRetrievalMode в одних и тех же запросах" настоящего руководства.

Следующий запрос возвращает значения аналогового тэга ReactLevel. Без указания параметров wwCycleCount и wwResolution будет возвращено 100 строк (количество возвращаемых строк по умолчанию).

```
SELECT DateTime, Sec = DATEPART(ss, DateTime),
TagName, Value
    FROM History
      WHERE TagName = 'ReactLevel'
      AND DateTime >= '2001-03-13 1:15:00pm'
      AND DateTime <= '2001-03-13 2:15:00pm'
```

```
    AND wwRetrievalMode = 'Cyclic'
```

Результаты:

DateTime	Sec	TagName	Value
2001-03-13 13:15:00.000	0	ReactLevel	1775.0
2001-03-13 13:15:00.000	36	ReactLevel	1260.0
2001-03-13 13:16:00.000	12	ReactLevel	1650.0
2001-03-13 13:16:00.000	49	ReactLevel	1280.0
2001-03-13 13:17:00.000	25	ReactLevel	1525.0
2001-03-13 13:18:00.000	1	ReactLevel	585.0
2001-03-13 13:18:00.000	38	ReactLevel	1400.0
2001-03-13 13:19:00.000	14	ReactLevel	650.0
2001-03-13 13:19:00.000	50	ReactLevel	2025.0
2001-03-13 13:20:00.000	27	ReactLevel	765.0
2001-03-13 13:21:00.000	3	ReactLevel	2000.0
2001-03-13 13:21:00.000	39	ReactLevel	830.0
2001-03-13 13:22:00.000	16	ReactLevel	1925.0

…

(100 строк показано)

Режим извлечения по изменению

Извлечение по изменению (то есть извлечение по исключительным ситуациям) представляет собой режим считывания из архива только изменившихся значений тэгов. Таким образом, возврат дублирующих значений исключён.

В режим извлечения по изменению возвращаемый набор строк всегда состоит из строк с реальными данными (то есть возвращаются строки, действительно хранящиеся в архиве). Данный режим является режимом по умолчанию при извлечении значений логических и символьных тэгов, а также данных из таблицы History.

Режим извлечения по изменению задаётся следующим образом:

```
wwRetrievalMode = 'Delta'
```

Количество возвращаемых строк можно также контролировать с помощью параметров wwTimeDeadband, wwValueDeadband и wwCycleCount. При использовании счётчика подынтервалов возвращаются первые строки в заданном количестве из указанного в запросе периода времени. Подробнее см. раздел "Использование параметров wwResolution, wwCycleCount и wwRetrievalMode в одних и тех же запросах" настоящего руководства.

Кроме того, указание мёртвых зон по времени и/или мёртвых зон по значению в режиме извлечения по изменению даёт различные результаты. Дополнительн см. разделы, посвящённые описанию параметров wwTimeDeadband и wwValueDeadband.

Пример 1

```
SELECT DateTime, TagName, Value
    FROM History
      WHERE TagName IN ('SysTimeSec', 'SysTimeMin')
          AND DateTime >= '2001-12-09 11:35'
```

```
        AND DateTime <= '2001-12-09 11:36'
        AND wwRetrievalMode = 'Delta'
```

Результаты:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0.0
2001-12-09 11:35:00.000	SysTimeMin	35.0
2001-12-09 11:35:01.000	SysTimeSec	1.0
2001-12-09 11:35:02.000	SysTimeSec	2.0
2001-12-09 11:35:03.000	SysTimeSec	3.0
2001-12-09 11:35:04.000	SysTimeSec	4.0
…		
2001-12-09 11:35:58.000	SysTimeSec	58.0
2001-12-09 11:35:59.000	SysTimeSec	59.0
2001-12-09 11:36:00.000	SysTimeSec	0.0
2001-12-09 11:36:00.000	SysTimeMin	36.0

Пример 2

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime,
Value, Quality, QualityDetail
    FROM AnalogHistory
      WHERE TagName = "SysTimeSec"
          AND wwRetrievalMode = "Delta"
          AND Value = 10
          AND DateTime >= "2001-07-27 03:00:00.000"
          AND DateTime <= "2001-07-27 03:05:00.000"
    ')
```

Результаты:

DateTime	Value	Quality	QualityDetail
2001-07-27 03:00:10.000	10.0	0	192
2001-07-27 03:01:10.000	10.0	0	192
2001-07-27 03:02:10.000	10.0	0	192
2001-07-27 03:03:10.000	10.0	0	192
2001-07-27 03:04:10.000	10.0	0	192

Пример 3

Если в запросе с режимом извлечения по изменению в условии сравнения указаны оба параметра wwCycleCount и Value, возвращены будут первые строки (если они существуют) в заданном количестве с указанным значением.

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime,
Value, Quality, QualityDetail
    FROM AnalogHistory
      WHERE TagName = "SysTimeSec"
```

```
            AND wwRetrievalMode = "Delta"
            AND Value = 20
            AND wwCycleCount = 10
            AND DateTime >= "2001-07-27 03:00:00.000"
            AND DateTime <= "2001-07-27 03:20:00.000"
    ')
```

Результаты:

DateTime	Value	Quality	QualityDetail
2001-07-27 03:00:20.000	20.0	0	192
2001-07-27 03:01:20.000	20.0	0	192
2001-07-27 03:02:20.000	20.0	0	192
2001-07-27 03:03:20.000	20.0	0	192
2001-07-27 03:04:20.000	20.0	0	192
2001-07-27 03:05:20.000	20.0	0	192
2001-07-27 03:06:20.000	20.0	0	192
2001-07-27 03:07:20.000	20.0	0	192
2001-07-27 03:08:20.000	20.0	0	192
2001-07-27 03:09:20.000	20.0	0	192

Режим полного извлечения

Этот режим, допустимый для тэгов любого типа, наиболее точно может быть охарактеризован как режим извлечения по изменению без проверок нарушения допусков. В режиме полного извлечения возвращаются все сохранённые данные независимо от того, изменялись ли их значения или качество. Кроме того, он обеспечивает возвращение одинаковых пар значений и качества вместе с соответствующими метками времени (таким же образом выводятся последовательные значения NULL).

В сочетании с режимом сохранения без фильтрации, то есть не в режиме циклического сохранения или сохранения по изменению, режим полного извлечения обеспечивает выдачу всех значений и качества данных, полученных от источников данных в производственной системе и в других приложениях.

Пример запроса:

```
SELECT DateTime, TagName, Value
    FROM History
      WHERE TagName IN ('SysTimeSec', 'SysTimeMin')
          AND DateTime >= '2001-12-09 11:35'
          AND DateTime <= '2001-12-09 11:36'
          AND wwRetrievalMode = 'Full'
```

Режим извлечения с интерполяцией

В режиме извлечения с интерполяцией значения данных на границах интервала определяются методом линейного приближения.

Обычно в этом режиме выполняется извлечение значений тэгов, которые медленно изменяются во времени. При построении трендов в этом режиме

будут получены более сглаженные кривые. Кроме того, он позволяет одновременно извлекать значения тэгов, изменяющихся во времени как медленно, так и быстро. И, наконец, возвращаемые в этом режиме значения будут равномернее распределены по времени, чем значения, возвращаемые запросом без интерполяции.

Режим извлечения с интерполяцией задаётся следующим образом:

```
wwRetrievalMode = 'Interpolated'
```

Данный режим, по сути, является циклическим. В этом режиме возвращается по одному значению тэгов в каждом подынтервале.

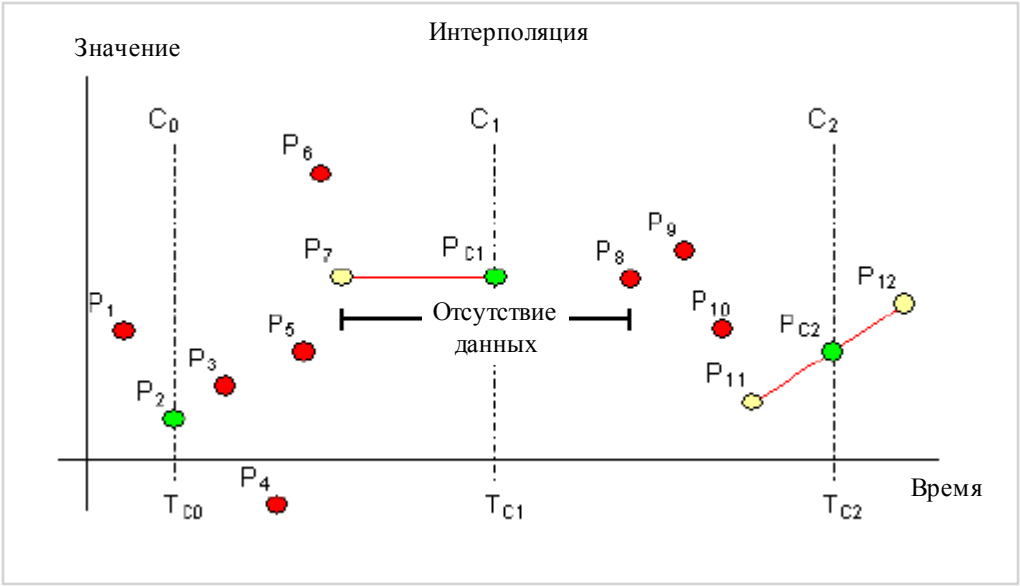
Количество подынтервалов определяется по указанному значению счётчика подынтервалов либо разрешения.

Возможны следующие уточнения режима полного извлечения:

- Отмена указанного типа интерполяции для расчёта значений на границах подынтервала.
- Указание правила определения меток времени.

Интерполяция применяется только к значениям аналоговых тэгов. Если в запросе будут указаны тэги других типов, расчётное значение этих тэгов будет принято равным последнему сохранённому значению.

На следующем рисунке показано, как рассчитываются приближённые значения аналоговых тэгов при выполнении запросов с интерполяцией.



Пусть в запросе с интерполяцией указаны начальный и конечный моменты T_{C0} и T_{C2}, причём возвращены должны быть значения на трёх границах подынтервалов: T_{C0}, T_{C1}, T_{C2}.

Всего в указанный период попадают 12 значений тэга (на рисунке показаны точками P_1 – P_{12}). Из них 11 представляют реальные значение, а точка P_7 – значение NULL (из-за отключения сервера в/в), что в результате создаёт интервал отсутствия данных между точками P_7 и P_8.

Зелёным цветом отмечены точки, которые будут возвращены при выполнении запроса. Красным цветом отмечены точки, которые учитываются, но не используются в расчётах возвращаемых значений. Точки, отмеченные жёлтым цветом, используются для расчёта приближённых значений в каждом подынтервале.

Поскольку точка P_2 точно попадает на границу подынтервала, она будет возвращена как граничное значение без какой-либо интерполяции. Для следующей границы будет возвращено значение P_{C1}, которое будет равно

NULL (вследствие сдвига значения P_7 до границы подынтервала). И, наконец, для последней границы будет возвращено значение P_{C2}, которое будет рассчитано методом интерполяции по значениям выборок P_{11} и P_{12}.

Пример 1

Из таблицы History запросом с интерполяцией извлекаются значения двух аналоговых тэгов и одного логического. Начальный и конечный моменты периода времени изменены, чтобы продемонстрировать интерполяцию значений тэга SysTimeMin.

```
SELECT DateTime, TagName, Value, wwInterpolationType
    FROM History
        WHERE TagName IN ('SysTimeMin', 'ReactTemp',
        'SysPulse')
          AND DateTime >= '2005-04-11 12:02:30'
          AND DateTime <= '2005-04-11 12:06:30'
          AND wwRetrievalMode = 'Interpolated'
          AND wwInterpolationType = 'Linear'
          AND wwResolution = 60000
```

Результаты:

DateTime	TagName	Value	wwInterpolationType
2005-04-11 12:02:30.000	SysTimeMin	2.5	LINEAR
2005-04-11 12:02:30.000	ReactTemp	23.2	LINEAR
2005-04-11 12:02:30.000	SysPulse	1.0	STAIRSTEP
2005-04-11 12:03:30.000	SysTimeMin	3.5	LINEAR
2005-04-11 12:03:30.000	ReactTemp	139.96753	LINEAR
2005-04-11 12:03:30.000	SysPulse	0.0	STAIRSTEP
2005-04-11 12:04:30.000	SysTimeMin	4.5	LINEAR
2005-04-11 12:04:30.000	ReactTemp	111.49636	LINEAR
2005-04-11 12:04:30.000	SysPulse	1.0	STAIRSTEP
2005-04-11 12:05:30.000	SysTimeMin	5.5	LINEAR
2005-04-11 12:05:30.000	ReactTemp	17.00238	LINEAR
2005-04-11 12:05:30.000	SysPulse	0.0	STAIRSTEP
2005-04-11 12:06:30.000	SysTimeMin	6.5	LINEAR
2005-04-11 12:06:30.000	ReactTemp	168.99531	LINEAR
2005-04-11 12:06:30.000	SysPulse	1.0	STAIRSTEP

Пример 2

Если в запросе тип интерполяции не будет указан, архиватор определит его по значению столбца IntepolationType таблицы AnalogTag с учётом

значений системных параметров InterpolationTypeInteger и InterpolationTypeReal.

В следующем запросе обоим аналоговым тэгам присвоены системные значения по умолчанию с помощью таблицы AnalogTag, при этом системным параметрам InterpolationTypeInteger и InterpolationTypeReal присвоены значения 0 и 1 соответственно. Поскольку тэг SysTimeMin является двухбайтной величиной целого типа, а тэг ReactTemp определён как вещественный, видно, что интерполируются только значения тэга ReactTemp.

```
SELECT DateTime, TagName, Value, wwInterpolationType
    FROM History
        WHERE TagName IN ('SysTimeMin', 'ReactTemp',
        'SysPulse')
           AND DateTime >= '2005-04-11 12:02:30'
           AND DateTime <= '2005-04-11 12:06:30'
           AND wwRetrievalMode = 'Interpolated'
           AND wwResolution = 60000
```

Результаты:

DateTime	TagName	Value	wwInterpolationType
2005-04-11 12:02:30.000	SysTimeMin	2.0	STAIRSTEP
2005-04-11 12:02:30.000	ReactTemp	23.2	LINEAR
2005-04-11 12:02:30.000	SysPulse	1.0	STAIRSTEP
2005-04-11 12:03:30.000	SysTimeMin	3.0	STAIRSTEP
2005-04-11 12:03:30.000	ReactTemp	139.96753	LINEAR
2005-04-11 12:03:30.000	SysPulse	0.0	STAIRSTEP
2005-04-11 12:04:30.000	SysTimeMin	4.0	STAIRSTEP
2005-04-11 12:04:30.000	ReactTemp	111.49636	LINEAR
2005-04-11 12:04:30.000	SysPulse	1.0	STAIRSTEP
2005-04-11 12:05:30.000	SysTimeMin	5.0	STAIRSTEP
2005-04-11 12:05:30.000	ReactTemp	17.00238	LINEAR
2005-04-11 12:05:30.000	SysPulse	0.0	STAIRSTEP
2005-04-11 12:06:30.000	SysTimeMin	6.0	STAIRSTEP
2005-04-11 12:06:30.000	ReactTemp	168.99531	LINEAR
2005-04-11 12:06:30.000	SysPulse	1.0	STAIRSTEP

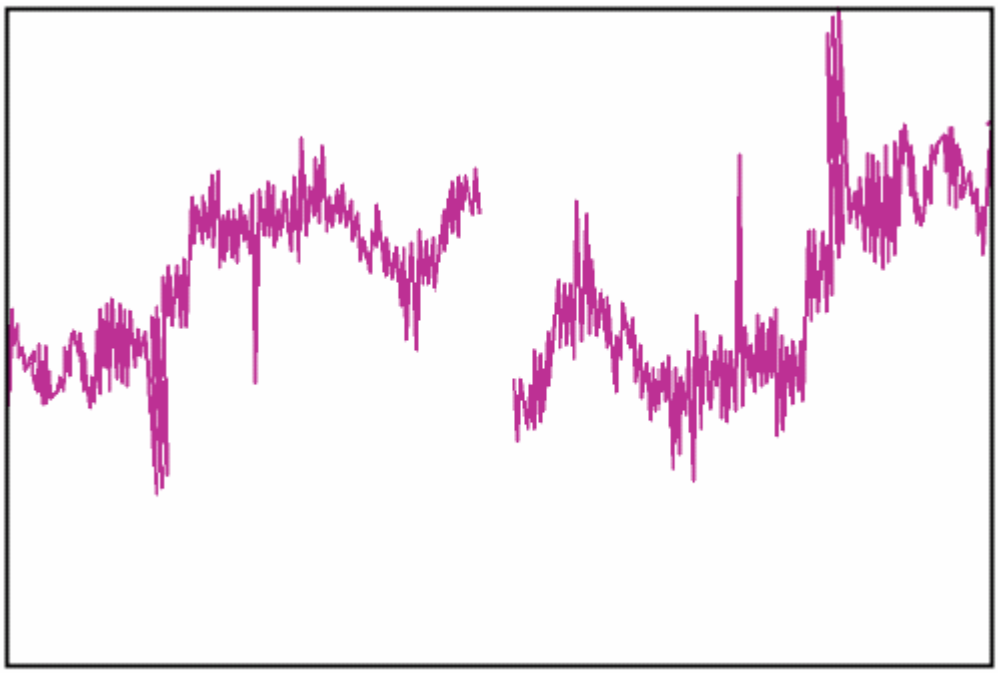
Режим извлечения по наилучшему соответствию

В режиме извлечения данных по наилучшему соответствию ("best fit") весь временной диапазон, указанный в запросе, делится на равные

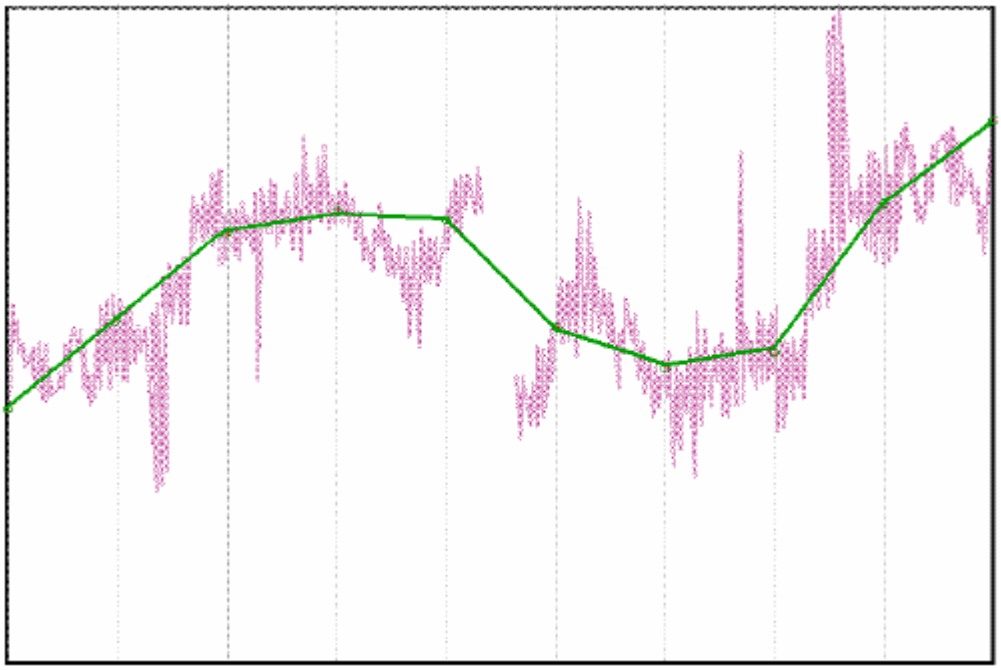
подынтервалы, для каждого из которых вычисляются следующие величины:

- Первое значение.
- Последнее значение
- Минимальное значение с соответствующей меткой времени.
- Максимальное значение с соответствующей меткой времени.
- Первое отклонение (данные с качеством, отличным от "GOOD").

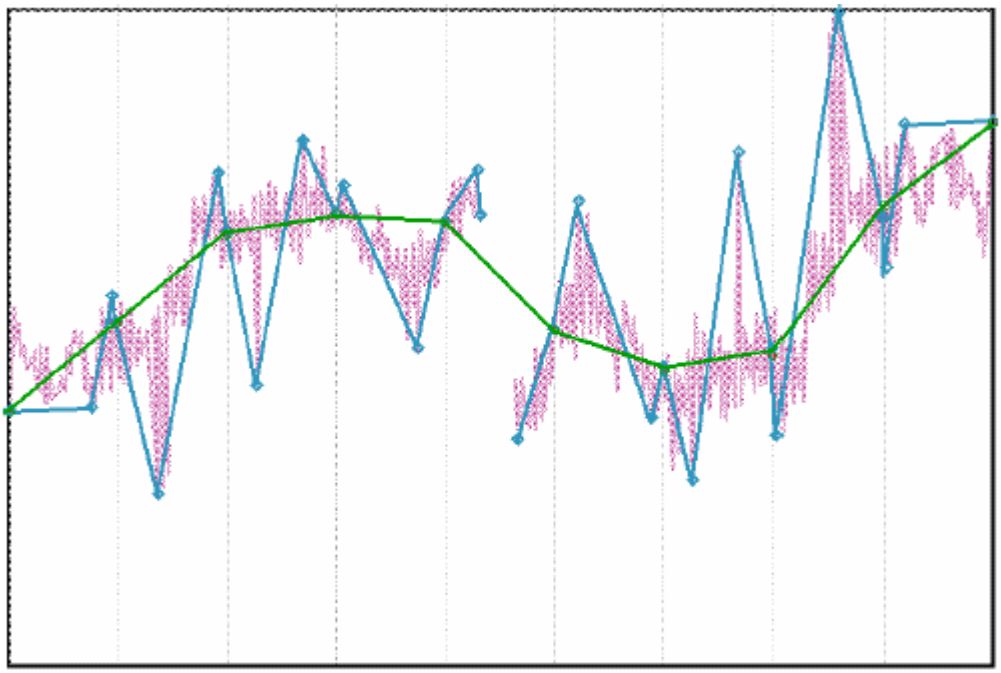
Режим извлечения данных по наилучшему соответствию представляет собой нечто среднее между извлечением по изменению и циклическим извлечением. В частности, извлечение данных по изменениям позволяет достаточно точно отражать характеристики процесса в течение долгого периода времени, как показано на следующем рисунке. Однако, чтобы получить подобную точность, требуется извлекать довольно большой объём информации.



Режим циклического извлечения гораздо эффективнее, потому что в нём из архива извлекается меньше значений. Однако представление процесса в этом случае будет неточным, как видно из следующего рисунка.



В режиме извлечения по наилучшему соответствию скорость извлечения данных сравнима с циклическим режимом, а качество представления данных – с режимом извлечения по изменению. Это показано на следующем рисунке.



В частности, в режиме извлечения по изменению запрос недельных данных, сохранение которых выполнялось каждые пять секунд, возвратит 120960 значений, в то время как в режиме извлечения по наилучшему соответствию будет возвращено только 300 значений.

Данный режим указывается следующим образом:

```
wwRetrievalMode = 'BestFit'
```

Хотя извлечение данных выполняется циклически, на самом деле это не циклический режим. Кроме первоначального значения возвращаются только фактические изменения.

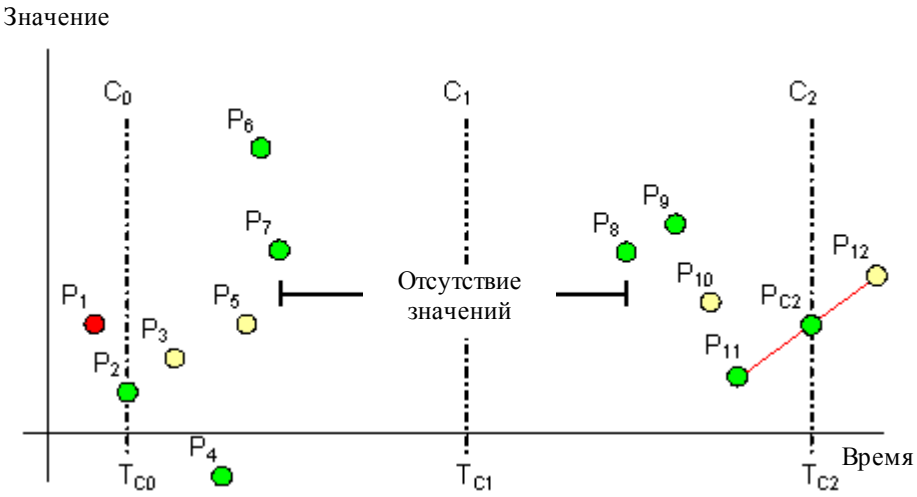
Количество подынтервалов определятся по указанному разрешению по времени и по счётчику подынтервалов. Количество возвращаемых строк в большинстве случаев будет превышать одну на подынтервал.

Единственный изменяемый параметр данного режима извлечения представляет собой подавление типа интерполяции для вычисления первоначального и конечного граничного значений.

Метод извлечения по наилучшему соответствию применим только к аналоговым тэгам. Значения остальных тэгов извлекаются методом "извлечение по изменению". Если какая-то точка оказывается одновременно и первой, и минимальной, она возвращаться будет только один раз. Если в каком-либо подынтервале нет выборок, никакие значения не возвращаются.

Все данные возвращаются в хронологическом порядке. Если какому-либо моменту соответствует несколько точек, их порядок в выходных результатах будет определяться порядком указания соответствующих запросов в запросе.

Следующий рисунок иллюстрирует точку значений аналогового тэга методом извлечения по наилучшему соответствию.



Пусть временной диапазон запроса начинается в момент T_{C0} и заканчивается в момент T_{C2}, а разрешение по времени таково, что данные должны будут возвращены для двух полных подынтервалов (начинающихся в моменты T_{C0} и T_{C1}) и одного неполного (начинающегося в момент T_{C2}).

Предположим, что в архиве всего находится 12 точек. Соответствующие точки на рисунке помечены символами P_1 – P_{12}. Из них 11 представляют нормальные значения тэга, одна (P_7) – значение NULL, полученное вследствие разрыва соединения с сервером в/в. В результате в архиве образовался промежуток в данных (между точками P_7 и P_8).

Поскольку точка P_2 попадает точно на начало подынтервала, интерполировать начальное значение в начале временного диапазона запроса не нужно. Таким образом, точка P_1 не учитывается. Остальные точки учитываются (но возвращены будут только те, которые на рисунке отмечены зелёным цветом).

Из первого подынтервала будут возвращены четыре точки:

- P_2 как первое значение результатов выполнения запроса, а также как начальное значение подынтервала.
- P_4 как минимальное значение подынтервала.
- P_6 как максимальное и одновременно последнее значение подынтервала.

- P_7 как первое (и единственное) отклонение в подынтервале.

Из второго подынтервала будут возвращены три точки:

- P_8 как начальное значение подынтервала.
- P_9 как максимальное значение подынтервала.
- P_{11} как минимальное и одновременно последнее значение подынтервала.

Значительно отклоняющихся точек в этом подынтервале нет.

Так как никакие значения тэга не попадают на конец временного диапазона запроса, граничное значение PC_2 будет определено методом интерполяции точек P_{11} и P_{12} (предполагается линейной).

Пример 1

Нужно извлечь значения аналогового тэга, соответствующие пятиминутному интервалу, методом наилучшего соответствия. Параметру wwResolution присвоено значение 60000 (то есть задано пять одноминутных подынтервалов). Для каждого подынтервала система извлечения определяет начальное, минимальное, максимальное и последнее значения. В указанном временном периоде значительно отклоняющихся точек нет. Нужно отметить, что первое и последнее значения, возвращаемые указанным запросом, будут рассчитываться методом интерполяции, остальные будут соответствовать реальным данным.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 1),
    Value) AS Value, wwInterpolationType AS IT FROM
    History
       WHERE TagName = 'ReactTemp'
           AND DateTime >= '2005-04-11 12:15:00'
           AND DateTime <= '2005-04-11 12:20:00'
           AND wwRetrievalMode = 'BestFit'
           AND wwResolution = 60000
```

Результаты:

	DateTime	TagName	Value	Interpolation Type
(начальное, первое, мин.)	2005-04-11 12:15:00.000	ReactTemp	40.7	LINEAR
(макс. в подынтервале 1)	2005-04-11 12:15:38.793	ReactTemp	196.0	STAIRSTEP
(последнее в подынтервале 1)	2005-04-11 12:15:58.810	ReactTemp	159.2	STAIRSTEP
(первое, макс. в подынтервале 2)	2005-04-11 12:16:00.013	ReactTemp	156.9	STAIRSTEP
(последнее, мин. в подынтервале 2)	2005-04-11 12:16:58.857	ReactTemp	16.3	STAIRSTEP
(первое, мин. в подынтервале 3	2005-04-11 12:17:00.060	ReactTemp	14.0	STAIRSTEP
(последнее, макс. в подынтервале 3)	2005-04-11 12:17:58.793	ReactTemp	151.0	STAIRSTEP
(первое в	2005-04-11 12:18:00.107	ReactTemp	156.0	STAIRSTEP

подынтервале 4)				
(макс. в подынтервале 4)	2005-04-11 12:18:10.057	ReactTemp	196.0	STAIRSTEP
(последнее, мин. в подынтервале 4)	2005-04-11 12:18:58.837	ReactTemp	106.3	STAIRSTEP
(первое, макс. в подынтервале 5)	2005-04-11 12:19:00.040	ReactTemp	104.0	STAIRSTEP
(мин. в подынтервале 5)	2005-04-11 12:19:31.320	ReactTemp	14.0	STAIRSTEP
(последнее в подынтервале 5)	2005-04-11 12:19:58.773	ReactTemp	26.0	STAIRSTEP
(конечное граничное значение в результатах)	2005-04-11 12:20:00.000	ReactTemp	30.7	LINEAR

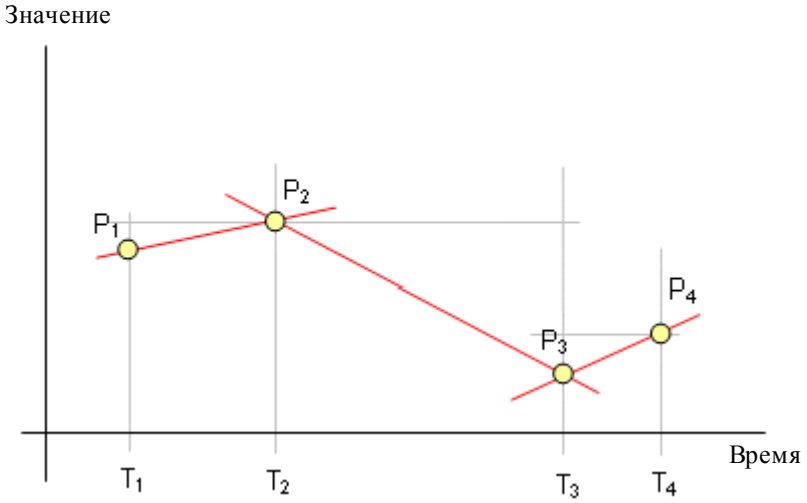
Режим извлечения среднего с временным взвешиванием

Возвращаемые в этом режиме извлечения данные рассчитываются методом усреднения с взвешиванием по времени (time-weighted average) для каждого подынтервала.

Обычное среднее арифметическое рассчитывается как результат деления суммы нескольких величин на их количество. Например, среднее арифметическое четырёх значений определяется по следующей формуле:

$(P_1 + P_2 + P_3 + P_4) / 4 = \text{Среднее}$

Среднее арифметическое

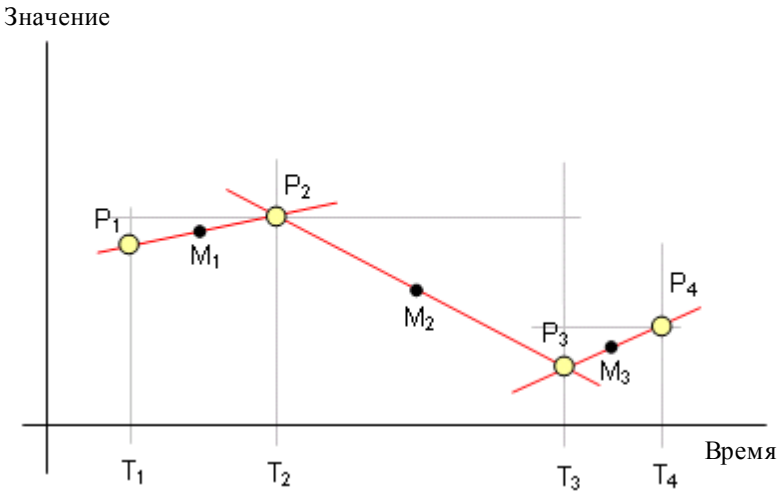


Средневзвешенное по времени определяется следующим образом. Сначала для каждой пары последовательных выборок определяется результат произведения их среднего на разницу соответствующих отметок времени. Последняя величина и определяет временной вес этой пары: чем больше интервал времени между точками, тем большее влияние они оказывают на конечную величину. Полученные таким образом значения складываются, и сумма делится на длительность общего периода времени.

Для четырёх точек средневзвешенное по времени будет определяться по следующей формуле:

$(((P_2+P_1)/2)*(T_2-T_1)) + (((P_3+P_2)/2)*(T_3-T_2)) + (((P_4+P_3)/2)*(T_4-T_3)) / (T_4-T_1)$ = Среднее

Средневзвешенное по времени



Функция AVG сервера SQL Server вычисляет среднее арифметическое. Расчёт средневзвешенного по времени с указанием длительности подынтервалов в 1 секунду даёт такой же результат, как и эта функция, однако результаты выдаются значительно быстрее. Кроме того, подсистема событий также может выдавать средние значения, но это будут средние арифметические величины.

Режим усреднения с взвешиванием по времени устанавливается следующим образом:

```
wwRetrievalMode = 'Average'
```

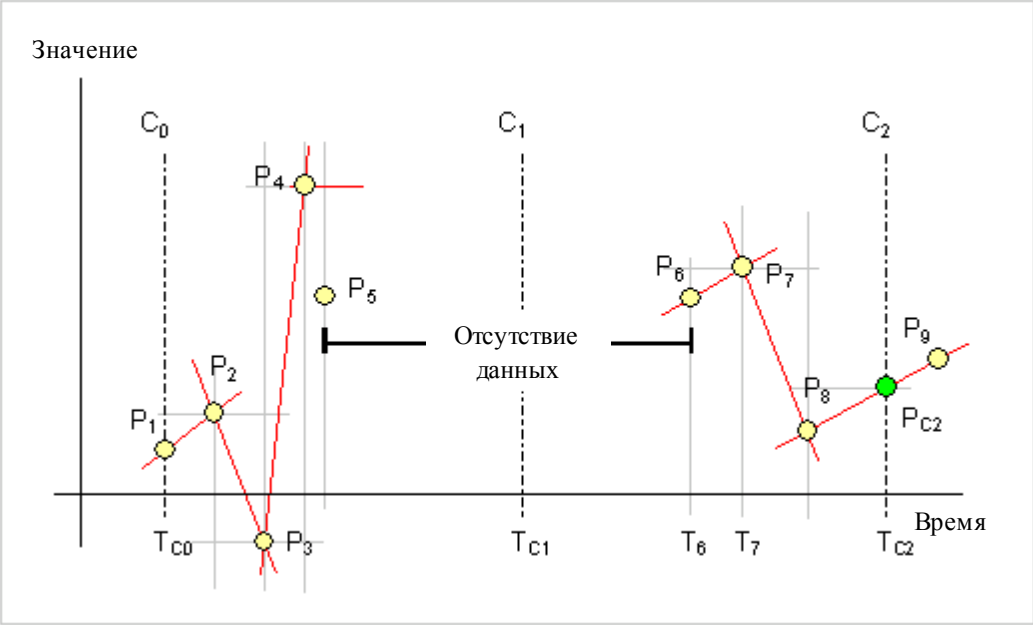
Данный режим является полностью циклическим и возвращает по одной строке результатов на подынтервал для каждого тэга в запросе.

Дополнительно возможны следующие настройки:

- Типа интерполяции.
- Указание правила расчёта метки времени.
- Указание правила определения качества.

Усреднение с взвешиванием по времени действительно для значений только аналоговых тэгов. Если в запросе указаны тэги других типов, значения для них возвращаются в обычном циклическом режиме (как будто для них выполняется обычный циклический запрос).

Следующий рисунок иллюстрирует результаты применения метода усреднения с взвешиванием по времени.



Пусть в запросе указан временной диапазон с началом в момент T_{C0} и концом в момент T_{C2}, а разрешение по времени установлено так, что возвращаться должны значения из двух полных подынтервалов (с началом в моменты T_{C0} и T_{C1}) и одного неполного (с началом в момент T_{C2}).

Пусть для указанного диапазона было найдено 9 точек (представленных на рисунке точками P_1 – P_9), из них 8 соответствуют нормальным значениям, одна (P_5) – значению NULL (полученному в момент отсутствия соединения с сервером в/в), в результате чего в архиве образовался промежуток в данных между точками P_5 и P_6.

Предположив, что метки времениопределяются по концу подынтервалов, получим:

- "Начальное значение", соответствующее началу временного диапазона запроса (T_{C0} на рисунке) будет представлять собой не интерполированное значение, а средневзвешенное по времени значений в подынтервале, предшествующему периоду запроса.
- Значение в момент T_{C1} будет представлять собой средневзвешенное по времени всех точек в подынтервале, начинающемся в начальный момент временного диапазона запроса.
- Значение в конечный момент T_{C2} будет представлять собой средневзвешенное по времени всех выборок в подынтервале, начинающемся в момент T_{C1}.

Чтобы понять работу алгоритма усреднения с взвешиванием по времени, рассмотрим последний подынтервал. Сначала нужно рассчитать площадь под кривой значений. Эта кривая, которая отмечена красным цветом, проходит через точки P_6, P_7, P_8 и P_{C2}, где PC2 представляет собой точку на границе интервала TC2, определённую по точкам P_8 и P_9 методом интерполяции. Площадь под кривой в области отсутствия данных, вызванного отсоединением сервера в/в, равна 0. При указании конкретного критерия качества ("GOOD") площадь под кривой, проходящей через точки с неудовлетворительным качеством данных, также не будет учитываться в общей сумме.

Расчёт площади рассмотрим на примере точек P_6 и P_7. Искомая величина равна произведению их среднего арифметического на разницу их меток време:

$$((P_6 + P_7) / 2) * (T_7 – T_6)$$

После подсчёта общей площади в подынтервале значение средневзвешенного по времени определяется путём деления площади на длительность этого интервала за вычетом периодов, площадь кривой над которыми была принята равной 0. Результат возвращается в конце подынтервала.

Из рисунка видно, что линия красного цвета, проходящая через точку P_4, параллельна оси X. Причина в том, что точка P_5 представляет значение NULL, которое не может использоваться для вычисления арифметического среднего. Вместо неё в расчёте взята точка P_4.

Знак полученной величины учитывается. Если среднее арифметическое двух точек отрицательно, и площадь под этим участком кривой также будет отрицательна.

Пример 1

Средневзвешенное по времени вычисляется для каждого одноминутного подынтервала.

Следует заметить, что значение параметра wwTimeStampRule в запросе равно "Start".Это означает, что расчётное значение с меткой времени 11:18:00.000 представляет собой среднее арифметическое для подынтервала от 11:18 до 11:19, значение с меткой времени 11:19:00.000 – среднее арифметическое для подынтервала от 11:19 до 11:20 и т.д. Если в запросе параметр wwTimeStampRule не указан, будет использовано значение по умолчанию (системный параметр TimeStampRule).

В первом подынтервале нет выборок, поэтому в результатах будет возвращено значение NULL. Для второго подынтервала, в котором точки имеются, значение PercentGood равно 77,72. Это означает, что возвращаемое значение было рассчитано по 77,72% всей длительности этого подынтервала. Поскольку не все точки в этом подынтервале имеют одинаковое значение OPCQuality, выходное значение OPCQuality установлено в "Doubtful". Качество данных в остальных трёх подынтервалах хорошее ("GOOD"), поэтому выходное значение качества равно 192.

Поскольку в запросе не было указано никакого правила определения качества (параметр wwQualityRule), при выполнении запроса используется значение по умолчанию, определяемое системным параметром QualityRule. Если для параметра wwQualityRule указать значение "Extended", в расчётах среднего арифметического и "полезной" длительности подынтервала PercentGood будут учитываться точки с недостоверным качеством данных.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 2),
Value) AS Value, OPCQuality, PercentGood
    FROM History
      WHERE TagName = 'ReactTemp'
          AND DateTime >= '2005-04-11 11:18:00'
          AND DateTime < '2005-04-11 11:23:00'
          AND wwRetrievalMode = 'Average'
          AND wwCycleCount = 5
          AND wwTimeStampRule = 'Start'
```

Результаты:

	DateTime	TagName	Value	OPCQuality	PercentGood
подынтер-вал 1	2005-04-11 11:18:00.000	ReactTemp	NULL	0	0.0
подынтер-	2005-04-11 11:19:00.000	ReactTemp	70.00	64	77.72

вал 2					
подынтервал 3	2005-04-11 11:20:00.000	ReactTemp	153.99	192	100.0
подынтервал 4	2005-04-11 11:21:00.000	ReactTemp	34.31	192	100.0
подынтервал 5	2005-04-11 11:22:00.000	ReactTemp	134.75	192	100.0

Пример 2

Следующий пример иллюстрирует применение метода в расширенном запросе. Для аналоговых тэгов ReactTemp и ReactLevel возвращаются средневзвешенные по времени значения, для логического тэга WaterValve – обычные циклические значения.

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime,
ReactTemp, ReactLevel, WaterValve
    FROM WideHistory
      WHERE DateTime >= "20040607 08:00"
          AND DateTime < "20040607 08:05"
          AND wwRetrievalMode = "Average"
          AND wwCycleCount = 5
')
```

Результаты:

DateTime	ReactTemp	ReactLevel	WaterValve
2004-06-07 08:00:00.000	47.71621	1676.69716	1
2004-06-07 08:01:00.000	157.28076	1370.88097	0
2004-06-07 08:02:00.000	41.33734	797.67296	1
2004-06-07 08:03:00.000	122.99525	1921.66771	0
2004-06-07 08:04:00.000	105.28866	606.40488	1

Режим извлечения минимального значения

В этом режиме возвращается минимальное значение, найденное в каждом подынтервале общего временного диапазона запроса, вместе с соответствующей меткой времени. Никакие данные не возвращаются, если в подынтервале значений тэга нет. Если в подынтервале есть одно или несколько значений NULL, возвращается NULL.

Запрос со значением счётчика подынтервалов, равным 1, возвращает такое же значение, как и обычная функция MIN, однако значительно быстрее.

Режим извлечения минимального значения задаётся следующим образом:

```
wwRetrievalMode = 'Minimum'
```

Данный режим не является полностью циклическим. Кроме начального значения, все остальные точки возвращаются в режиме "по изменению".

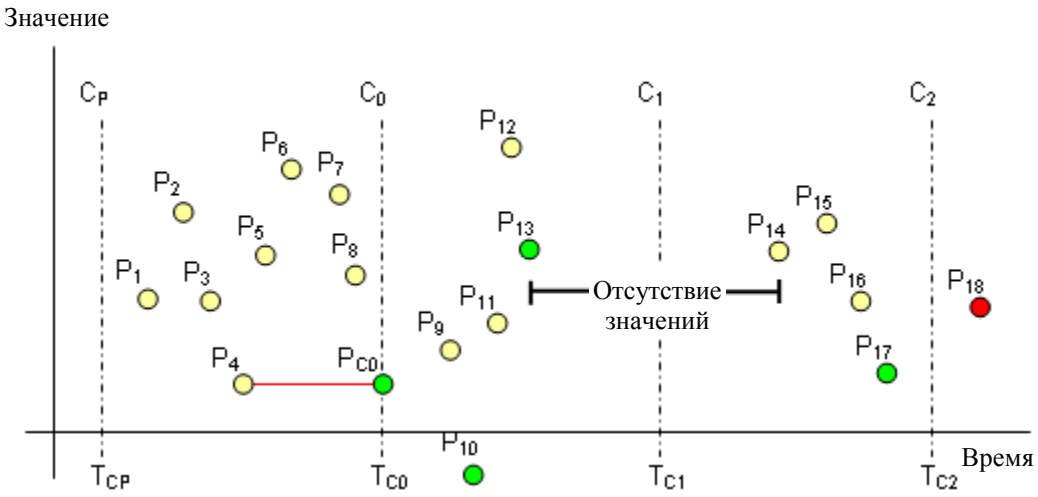
Количество подынтервалов определяется значением счётчика подынтервалов, при этом количество строк может отличаться от одной на подынтервал.

Для этого режима дополнительные параметры не устанавливаются.

Режим извлечения минимального значения действителен только для аналоговых тэгов, значения остальных тэгов, указанных в запросе этого типа, возвращаются методом извлечения "по изменению".

Возвращаемые значения отсортированы в хронологическом порядке. Если одной и той же метке времени соответствует несколько выборок данных, их порядок определяется порядком указания тэгов в запросе.

Следующий рисунок иллюстрирует извлечение значений аналоговых тэгов в данном режиме.



Пусть в запросе указан временной диапазон с началом в момент T_{C0} и концом в момент T_{C2}, а разрешение по времени установлено так, что возвращаться должны значения из двух полных подынтервалов (с началом в моменты T_{C0} и T_{C1}), одного "постороннего" (с началом в момент T_{CP}) и одного неполного (с началом в момент T_{C2}).

Пусть для указанного диапазона было найдено 18 точек (представленных на рисунке точками P_1 – P_{18}), из них 17 соответствуют нормальным значениям, одна (P_{13}) – значению NULL (полученному в момент отсутствия соединения с сервером в/в), в результате чего в архиве образовался промежуток в данных между точками P_{13} и P_{14}.

Точка P_4 соответствует минимальной точке данных в "постороннем" подынтервале. Длительность этого подынтервала равна длительности первого подынтервала запроса, простирается он "в прошлое" от начального момента запроса. Точка P_{18} вообще не будет учитываться, потому что она находится вне временного диапазона запроса. В результате выполнения запроса извлечены из архива будут все показанные точки, но возвращены только те, которые отмечены зелёным цветом.

Всего в результатах выполнения запроса будет присутствовать четыре точки:

- P_4 как минимальное значение "постороннего" подынтервала и начальное значение.
- P_{10} как минимальное значение первого подынтервала.
- P_{13} как первое и единственное значительно отклоняющееся значение в первом подынтервале.
- P_{17} как минимальное значение второго подынтервала.

Для третьего, неполного, подынтервала не будет возвращено ни одного значения, поскольку на его начало (и конец временного диапазона запроса) точно не попадает ни одна точка.

Если бы минимальное значение первого подынтервала приходилось на начало временного диапазона запроса, возвращены были бы оно и минимальное значение "постороннего" подынтервала.

Пример 1

Ниже приведён пример запроса на значения аналогового тэга в течение пятиминутного интервала. Поскольку параметр wwResolution установлен в 60000, каждый подынтервал длится ровно одну минуту. Система возвращает минимальные значения для каждого из пяти подынтервалов.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 2),
Value) AS Value
    FROM History
      WHERE TagName = 'ReactTemp'
          AND DateTime >= '2005-04-11 11:21:00'
          AND DateTime <= '2005-04-11 11:26:00'
          AND wwRetrievalMode = 'Minimum'
          AND wwResolution = 60000
```

Начальное значение, метка времени которого совпадает с началом временного диапазона запроса, одновременно является и минимальным значением подынтервала, предшествующего временному диапазону запроса.

Результаты:

	DateTime	TagName	Value
"Посторонний" подынтервал	2005-04-11 11:21:00.000	ReactTemp	104.00
подынтервал 1	2005-04-11 11:21:30.837	ReactTemp	14.00
подынтервал 2	2005-04-11 11:22:00.897	ReactTemp	36.00
подынтервал 3	2005-04-11 11:23:59.567	ReactTemp	18.60
подынтервал 4	2005-04-11 11:24:02.083	ReactTemp	14.00
подынтервал 5	2005-04-11 11:25:59.550	ReactTemp	108.60

Пример 2

Данный запрос возвращает результаты аналогично обычному запросу к Microsoft SQL Server с использованием функции MIN. Следует отметить, что подынтервалом, в котором будет найден результат, является пятиминутный "посторонний" подынтервал, предшествующий временному диапазону запроса.

```
SELECT TOP 1 DateTime, TagName, CONVERT(DECIMAL(10,
2), Value) AS Value
    FROM History
      WHERE TagName = 'ReactTemp'
          AND DateTime >= '2005-04-11 11:31:00'
          AND DateTime <= '2005-04-11 11:31:00'
          AND wwRetrievalMode = 'Minimum'
          AND wwResolution = 300000
```

Результаты:

DateTime	TagName	Value

"Посторонний" подынтервал 2005-04-11 11:31:00.000 ReactTemp 14.00

Пример 3

В примере показано, как параметр QualityDetail используется для индикации того, что минимальное значение обнаружено в неполном подынтервале. В данном случае неполным циклом является тот, для которого в архиве нет никаких данных, или тот, внутрь которого попадает конец временного диапазона запроса. Все значения, возвращаемые в столбце QualityDetail, хранятся в таблице QualityMap Рабочей базы данных.

```
SELECT DateTime, TagName, Value, QualityDetail
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND DateTime >= '2005-04-11 11:18:50'
          AND DateTime <= '2005-04-11 11:20:50'
          AND wwRetrievalMode = 'Minimum'
          AND wwResolution = 60000
```

Результаты:

	DateTime	TagName	Value	QualityDetail
"Посторонний" подынтервал	2005-04-11 11:18:50.000	SysTimeSec	NULL	65536
подынтервал 1	2005-04-11 11:19:13.000	SysTimeSec	13.0	4140
подынтервал 2	2005-04-11 11:20:00.000	SysTimeSec	0.0	192
подынтервал 3	2005-04-11 11:20:50.000	SysTimeSec	50.0	4288

Режим извлечения максимального значения

В этом режиме возвращается максимальное значение, найденное в каждом подынтервале общего временного диапазона запроса, вместе с соответствующей меткой времени. Никакие данные не возвращаются, если в подынтервале значений тэга нет. Если в подынтервале есть одно или несколько значений NULL возвращается NULL.

Запрос со значением счётчика подынтервалов, равным 1, возвращает такое же значение, как и обычная функция MAX, однако значительно быстрее.

Режим извлечения максимального значения задаётся следующим образом:

```
wwRetrievalMode = 'Maximum'
```

Данный режим не является полностью циклическим. Кроме начального значения, все остальные выборки возвращаются в режиме "по изменению".

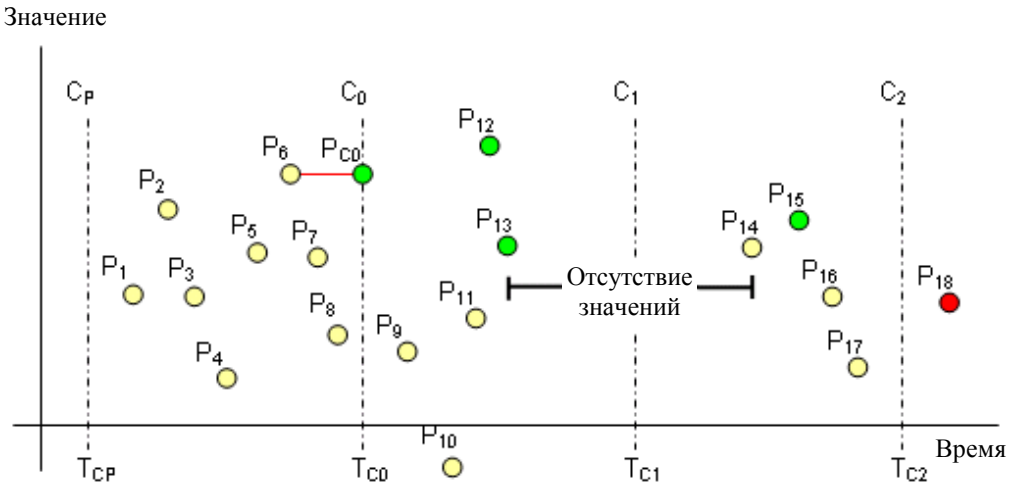
Количество подынтервалов определяется значением счётчика подынтервалов, при этом количество строк может отличаться от одной на подынтервал.

Для этого режима дополнительные параметры не устанавливаются.

Режим извлечения максимального значения действителен только для аналоговых тэгов, значения остальных тэгов, указанных в запросе этого типа, возвращаются методом извлечения "по изменению".

Возвращаемые значения отсортированы в хронологическом порядке. Если одной и той же метке времени соответствует несколько выборок данных, их порядок определяется порядком указания тэгов в запросе.

Следующий рисунок иллюстрирует извлечение значений аналоговых тэгов в данном режиме.



Пусть в запросе указан временной диапазон с началом в момент T_{C0} и концом в момент T_{C2}, а разрешение по времени установлено так, что возвращаться должны значения из двух полных подынтервалов (с началом в моменты T_{C0} и T_{C1}), одного "постороннего" (с началом в момент T_{CP}) и одного неполного (с началом в момент T_{C2}).

Пусть для указанного диапазона было найдено 18 выборок (представленных на рисунке точками P_1 – P_{18}), из них 17 соответствуют нормальным значениям, одна (P_{13}) – значению NULL (полученному в момент отсутствия соединения с сервером в/в), в результате чего в архиве образовался промежуток в данных между точками P_{13} и P_{14}.

Точка P_6 соответствует максимальной выборке данных в "постороннем" подынтервале. Длительность этого подынтервала равна длительности первого подынтервала запроса, простирается он "в прошлое" от начального момента запроса. Точка P_{18} вообще не будет учитываться, потому что она находится вне временного диапазона запроса. В результате выполнения запроса извлечены из архива будут все показанные точки, но возвращены только те, которые отмечены зелёным цветом.

Всего в результатах выполнения запроса будет присутствовать четыре выборки:

- P_6 как максимальное значение "постороннего" подынтервала и начальное значение.
- P_{12} как максимальное значение первого подынтервала.
- P_{13} как первое и единственное значительно отклоняющееся значение в первом подынтервале.
- P_{15} как максимальное значение второго подынтервала.

Для третьего, неполного, подынтервала не будет возвращено ни одного значения, поскольку на его начало (и конец временного диапазона запроса) точно не попадает ни одна точка.

Если бы максимальное значение первого подынтервала приходилось на начало временного диапазона запроса, возвращены бы были оно и максимальное значение "постороннего" подынтервала.

Пример 1

Ниже приведён пример запроса на значения аналогового тэга в течение пятиминутного интервала. Поскольку параметр wwResolution установлен в 60000, каждый подынтервал длится ровно одну минуту. Система возвращает максимальные значения для каждого из пяти подынтервалов.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 2),
Value) AS Value
    FROM History
      WHERE TagName = 'ReactTemp'
          AND DateTime >= '2005-04-11 11:21:00'
          AND DateTime <= '2005-04-11 11:26:00'
          AND wwRetrievalMode = 'Maximum'
          AND wwResolution = 60000
```

Начальное значение, метка времени которого совпадает с началом временного диапазона запроса, одновременно является и максимальным значением подынтервала, предшествующего временному диапазону запроса.

Результаты:

	DateTime	TagName	Value
"Посторонний" подынтервал	2005-04-11 11:21:00.000	ReactTemp	196.00
подынтервал 1	2005-04-11 11:21:00.853	ReactTemp	101.70
подынтервал 2	2005-04-11 11:22:40.837	ReactTemp	196.00
подынтервал 3	2005-04-11 11:23:00.833	ReactTemp	159.20
подынтервал 4	2005-04-11 11:24:59.613	ReactTemp	146.00
подынтервал 5	2005-04-11 11:25:12.083	ReactTemp	196.00

Пример 2

Данный запрос возвращает результаты аналогично обычному запросу к Microsoft SQL Server с использованием функции MAX. Следует отметить, что подынтервалом, в котором будет найден результат, является пятиминутный "посторонний" подынтервал, предшествующий временному диапазону запроса.

```
SELECT TOP 1 DateTime, TagName, CONVERT(DECIMAL(10,
2), Value) AS Value
    FROM History
      WHERE TagName = 'ReactTemp'
          AND DateTime >= '2005-04-11 11:31:00'
          AND DateTime <= '2005-04-11 11:31:00'
          AND wwRetrievalMode = 'Maximum'
          AND wwResolution = 300000
```

Результаты:

	DateTime	TagName	Value
"Посторонний" подынтервал	2005-04-11 11:31:00.000	ReactTemp	196.00

Пример 3

В примере показано, как параметр QualityDetail используется для индикации того, что максимальное значение обнаружено в неполном подынтервале. В данном случае неполным циклом является тот, для которого в архиве нет никаких данных, или тот, внутрь которого попадает

конец временного диапазона запроса. Все значения, возвращаемые в столбце QualityDetail, хранятся в таблице QualityMap Рабочей базы данных.

```
SELECT DateTime, TagName, Value, QualityDetail
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND DateTime >= '2005-04-11 11:19:10'
          AND DateTime <= '2005-04-11 11:21:10'
          AND wwRetrievalMode = 'Maximum'
          AND wwResolution = 60000
```

Результаты:

	DateTime	TagName	Value	QualityDetail
"Посторонний" подынтервал	2005-04-11 11:19:10.000	SysTimeSec	NULL	65536
подынтервал 1	2005-04-11 11:19:59.000	SysTimeSec	59.0	4288
подынтервал 2	2005-04-11 11:20:59.000	SysTimeSec	59.0	192
подынтервал 3	2005-04-11 11:21:10.000	SysTimeSec	10.0	4288

Интегральный режим извлечения

Значения, возвращаемые в этом режиме, рассчитываются для границ подынтервала извлечения путём интегрирования графика сохранённых значений тэга. Этот режим отличается от режима извлечения с взвешиванием по времени тем, что результат умножается на коэффициент масштабирования.

Интегральный режим извлечения обычно используется, когда нужно вычислить значения какого-либо тэга "нарастающим итогом".

Режим извлечения максимального значения задаётся следующим образом:

```
wwRetrievalMode = 'Integral'
```

Данный режим является полностью циклическим и возвращает по одной строке результатов для каждого тэга в каждом подынтервале извлечения.

Количество подынтервалов определяется значением счётчика подынтервалов. Для каждого тэга, указанного в запросе, возвращается ровно одна строка в подынтервале.

Дополнительно возможны следующие настройки:

- Подавление типа интерполяции.
- Указание правила расчёта метки времени.
- Указание правила определения качества.

Режим интегрального извлечения действителен для значений только аналоговых тэгов. Если в запросе указаны тэги других типов, значения для них возвращаются в обычном циклическом режиме (как будто для них выполняется обычный циклический запрос).

На первом шаге выполнения запроса определяется площадь под кривой значений тэга. Алгоритм нахождения площади такой же, как и в режиме извлечения среднего с взвешиванием по времени.

Найденное значение умножается на значение столбца IntegralDivisor таблицы EngineeringUnit, который используется для преобразования исходной величины в одну из величин в секунду.

Пример 1

В следующем примере интегральная величина вычисляется для каждого из пяти одноминутных подынтервалов. Установка параметра wwQualityRule означает, что учитываться будут только выборки с качеством, равным "GOOD", а остальные будут игнорироваться. Правила определения качества возвращаемых результатов такие же, как и в методе извлечения среднего с взвешиванием по времени.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 2),
Value) AS Flow, OPCQuality, PercentGood
    FROM History
      WHERE TagName = 'FlowRate'
          AND DateTime >= '2004-06-07 08:00'
          AND DateTime < '2004-06-07 08:05'
          AND wwRetrievalMode = 'Integral'
          AND wwCycleCount = 5
          AND wwQualityRule = 'Good'
```

Результаты:

	DateTime	TagName	Flow	OPCQuality	PercentGood
подынтер-вал 1	2004-06-07 08:00:00.000	FlowRate	2862.97	192	100.0
подынтер-вал 2	2004-06-07 08:01:00.000	FlowRate	9436.85	192	100.0
подынтер-вал 3	2004-06-07 08:02:00.000	FlowRate	2480.24	192	100.0
подынтер-вал 4	2004-06-07 08:03:00.000	FlowRate	7379.71	192	100.0
подынтер-вал 5	2004-06-07 08:04:00.000	FlowRate	6317.32	192	100.0

Режим извлечения с определением скорости изменения

В этом режиме выполняется расчёт наклона прямой, проходящей через текущую и предшествующую точки (то есть определяется скорость изменения значений).

В таком режиме значения обычно извлекаются для определения, изменяется ли значение того или иного тэга с заданной скоростью. Например, если при измерении температуры жидкости, которая должна равномерно повышаться, будут обнаружены резкие изменения, они могут означать появление определённых проблем.

Режим извлечения с определением скорости изменения задаётся следующим образом:

```
wwRetrievalMode = 'Slope'
```

Данный режим можно рассматривать как вариант извлечения по изменению. Кроме начального значения и значения в конце временного диапазона запроса, все выборки рассчитываются как изменившиеся и возвращаются вместе с фактическими метками времени.

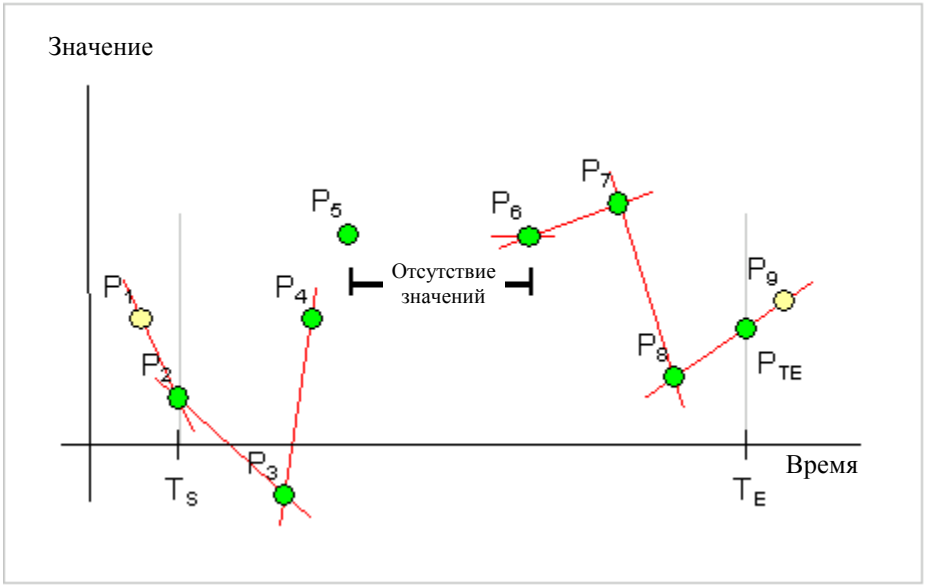
Временной диапазон запроса на подынтервалы не разбивается. Таким образом, указание разрешения по времени не оказывает никакого влияния на возвращаемые результаты.

Для исключения из расчётов данных с неудовлетворительным качеством можно дополнительно задавать правило определения качества.

Режим извлечения с определением скорости изменения действителен только для аналоговых тэгов. Для тэгов всех остальных типов возвращаются обычные изменённые значения.

Все данные возвращаются в хронологическом порядке. Если какому-либо моменту соответствует несколько выборок, их порядок в выходных результатах будет определяться порядком указания соответствующих подзапросов в запросе.

Следующий рисунок иллюстрирует выборку значений аналогового тэга в режиме определения скорости изменений.



Пусть временной диапазон запроса начинается в момент T_S и заканчивается в момент T_E.

Запрос возвращает 9 точек данных, которым на рисунке соответствуют точки P_1 – P_9. Из них 8 представляют нормальные значение, одна (P_5) – значение NULL (отсутствие соединения с сервером в/в), в результате чего в данных образовался промежуток между точками P_5 и P_6.

В этом режиме рассчитывается угол наклона прямой, проходящей через текущую и предыдущую точки. Расчёт выполняется для односекундного интервала. В данном примере точка P_2 попадает на начало временного диапазона запроса, и так как предыдущая точка (P_1) находится вне этого диапазона, угол наклона прямой, проходящей через эти точки, возвращается как соответствующий моменту времени T_S. Расчёт выполняется по следующей формуле:

$(P_2 – P_1) / (T_2 – T_1)$

Аналогичным образом рассчитываются углы наклона для моментов T_3, T_4, T_7 и T_8. Кроме того, угол наклона вычисляется и для точки P_9, но возвращается он как угол наклона в точке, соответствующей концу временного диапазона запроса P_{TE}.

Для точки P_6 нет предшествующей, которую можно было бы использовать в расчётах, поэтому угол наклона для неё принят равным 0.

Для момента T_5 возвращается значение NULL.

Пример 1

Следующий запрос позволяет определять скорость изменения тэга ReactTemp, измеряемую в градусах в секунду (ºC/c). Первое значение в

столбце Quality с меткой времени, равной началу временного диапазона запроса, показывает, что точно приходящейся на этот момент воемени точки данных нет, поэтому угол наклона будет принят равным углу наклона для следующей точки. На момент 08:01:17.947 приходится два значения тэга, поэтому угол наклона рассчитывается для первого из них, а для второго возвращается NULL с записью специального кода 17 в столбец QualityDetail (код, означающий невозможность расчёта угла наклона, потому что он равен либо плюс бесконечности, либо минут бесконечности).

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 4),
Value) AS Slope, Quality, QualityDetail
    FROM History
      WHERE TagName = 'ReactTemp'
          AND DateTime >= '2005-04-17 08:00'
          AND DateTime <= '2005-04-17 08:05'
          AND wwRetrievalMode = 'Slope'
```

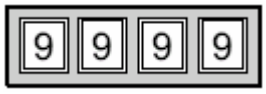
Результаты:

DateTime	TagName	Slope	Quality	QualityDetail
2005-04-17 08:00:00.000	ReactTemp	3.8110	133	192
2005-04-17 08:00:00.510	ReactTemp	3.8110	0	192
2005-04-17 08:00:01.713	ReactTemp	4.1563	0	192
2005-04-17 08:00:02.917	ReactTemp	4.1563	0	192
2005-04-17 08:00:04.230	ReactTemp	3.8081	0	192
2005-04-17 08:00:05.433	ReactTemp	4.1563	0	192
… … …	…	…		
2005-04-17 08:01:16.743	ReactTemp	-1.7517	0	192
2005-04-17 08:01:17.947	ReactTemp	-27.0158	0	192
2005-04-17 08:01:17.947	ReactTemp	NULL	1	17
2005-04-17 08:01:19.260	ReactTemp	-1.7530	0	192
2005-04-17 08:01:20.463	ReactTemp	-1.9119	0	192
2005-04-17 08:01:21.667	ReactTemp	-1.9119	0	192
2005-04-17 08:01:22.977	ReactTemp	-1.7517	0	192
… … …	…	…		

Счётный режим извлечения

В этом режиме определяется общая величина изменения тэга в последовательных подынтервалах извлечения с использованием его предельного значения.

Данный режим обычно используется для подсчёта единиц продукции, произведённой в некоторый период времени. Пусть, например, в системе для этого определён счётчик, который отображается на экране в следующем виде:



Предельным значением счётчика называется число, следующее за максимально допустимой для счётчика величиной. В данном случае после 9999 следует 10000. Когда счётчик достигнет 9999, он будет сброшен в 0.

Счётный режим извлечения задаётся следующим образом:

```
wwRetrievalMode = 'Counter'
```

Данный режим является полностью циклическим и возвращает по одной строке результатов для каждого тэга в каждом подынтервале запроса.

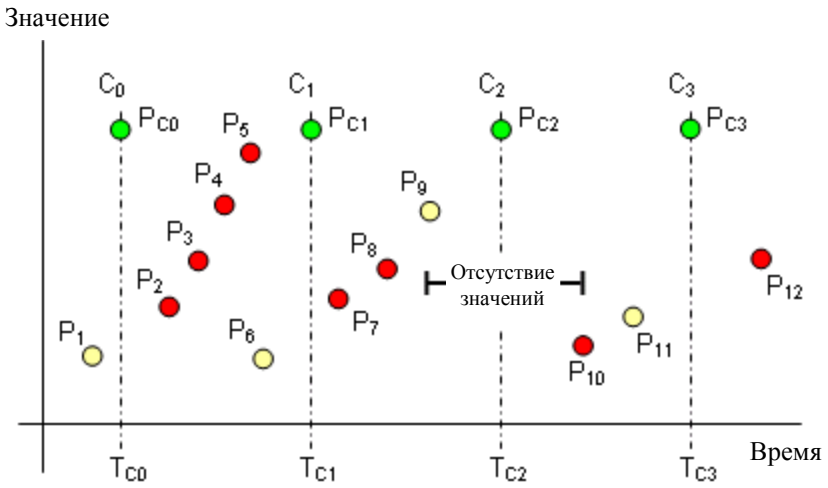
Количество подынтервалов определяется значением счётчика подынтервалов.

Дополнительно могут быть заданы следующие параметры:

- Правило образования отметок времени.
- Правило определения качества.

Счётный режим извлечения действителен только для невещественных аналоговых и логических тэгов. Если в запросе указаны тэги других типов, значения для них не возвращаются. Для логических тэгов предельным значением всегда является число 2.

Следующий рисунок иллюстрирует счётный режим извлечения.



Пусть временной диапазон запроса начинается в момент T_{C0} и заканчивается в момент T_{C3}, а разрешение по времени таково, что данные должны будут возвращены для трёх полных подынтервалов (начинающихся в моменты T_{C0}, T_{C1} и T_{21}) и одного неполного (начинающегося в момент T_{C3}).

Предположим, что в архиве всего находится 12 точек. Соответствующие точки на рисунке помечены символами P_1 – P_{12}. Из них 11 представляют нормальные значения тэга, одна (P_7) – значение NULL, полученное вследствие разрыва соединения с сервером в/в. В результате в архиве образовался промежуток в данных (между точками P_9 и P_{10}). Точка P_{12} не учитывается, потому что она находится вне временного диапазона запроса.

При выполнении запроса извлекаются все точки, однако набор возвращаемых клиенту значений будет определяться только теми, что отмечены на рисунке жёлтым цветом. Возвращаемые значения – P_{C0}, P_{C1}, P_{C2} и P_{C3} – отмечены зелёным цветом и нарисованы в верхней части рисунка, чтобы показать, что между ними и остальными реальными выборками явной взаимосвязи нет.

Все выборки в подынтервале рассматриваются как изменённые по сравнению с предыдущим подынтервалом с учётом того, сколько раз достигалось предельное значение. Первое значение, соответствующее

началу временного диапазона запроса (P_{C1}), рассчитывается аналогичным образом, но только на основании данных подынтервала, непосредственно предшествующего временному периоду запроса. В алгоритме подсчёта предполагается сброс счётчика, если текущее значение меньше предыдущего.

Формула расчёта значения P_{C1} выглядит следующим образом:

$$P_{C1} = n * V_R + P_6 – P_1,$$

где

n = количество сбросов значения тэга, выполненных в подынтервале,

V_R = установленное предельное значение тэга.

Если n или V_R равны нулю, значение P_{C1} просто равно разности значений P_6 и P_1.

Поскольку в конце подынтервала C_2 нет значений, возвращается значение NULL, представляемое точкой P_9. В подынтервале C_3 также возвращается NULL, поскольку для границы предыдущего цикла не было определено значение счётчика, которое можно было бы использовать в расчётах. Для расчёта значения счётчика нужно, чтобы подынтервал был полностью заполнен выборками.

Правила определения качества такие же, как и в режиме извлечения среднего с взвешиванием по времени.

Если промежуток в значениях входит в подынтервал полностью и вне обеих границ этого промежутка существуют какие-либо точки, значения счётчика определяются (хотя иногда они могут оказаться ошибочными). Предполагается наличие не более одного сброса счётчика, даже если на практике их было больше.

При использовании счётчика, который сбрасывается вручную до достижения предельного значения, нужно предельное значение для этого тэга установить в 0, чтобы общий итог показывал, насколько данный тэг изменился с момента последнего сброса.

Например, предположим, что для границ пяти последовательных подынтервалов были получены следующие значения, и что число 0 является первой точкой в последнем подынтервале:

100, 110, 117, 123, 3

Если предельное значение будет установлено в 0, алгоритм счётного извлечения будет предполагать, что значение 0, следующее за значением 123, означает ручной сброс счётчика, и возвратит значение 3 как первое значение счётчика после сброса. В этом случае число 0 не приводит к увеличению значения счётчика.

Если же предельное значение будет установлено в 200, алгоритм счётного извлечения предположит, что число 0 означает обычный сброс, и вернёт значение итога, равное 80 (80 = 200 – 123 + 3). В данном случае значение 0 является причиной увеличения счётчика, представляя собой изменение от 199 до 200.

Режим извлечения с определением времени нахождения в различных состояниях

В этом режиме возвращается время, в течение которого указанный в запросе тэг находился в указанном состоянии в каждом подынтервале.

Запросы этого типа позволяют определять, например, сколько времени станок работал или простаивал, сколько времени вентиль находился в открытом или закрытом состоянии и т.д.

Режим извлечения с определением времени нахождения в различных состояниях задаётся следующим образом:

```
wwRetrievalMode = 'ValueState'
```

Хотя в этом режиме временной диапазон запроса и разбивается на подынтервалы, он не является полностью циклическим. При выполнении запроса будут возвращаться по одной строке результатов для каждого значения каждого тэга в каждом подынтервале.

Количество подынтервалов определяется заданным разрешением по времени либо значением счётчика подынтервалов.

Дополнительно возможно указание следующих параметров:

- Правило определения меток време
- Правило определения качества.

Если в запросе будет указано качество типа "GOOD", все точки с неудовлетворительным качеством данных будут игнорироваться.

Режим извлечения с определением времени нахождения в различных состояниях действителен только для целых, логических и символьных тэгов. Для тэгов остальных типов никакие данные не возвращаются. Значения NULL в этом режиме специальным образом не обрабатываются, а трактуются как одно из состояний, время нахождения в котором также подсчитывается и возвращается в первой строке результатов для каждого тэга в каждом подынтервале.

Виды подсчётов

В запросе можно указать, какого рода информацию требуется извлекать о том или ином состоянии:

- **Minimum (минимально).** Самый короткий период времени нахождения тэга в каждом состоянии.
- **Maximum (максимально).** Самый долгий период времени нахождения тэга в каждом состоянии.
- **Average (в среднем).** Средняя продолжительность интервала нахождения тэга в каждом состоянии.
- **Total (в общем).** Общая продолжительность интервалов нахождения тэга в каждом состоянии.
- **Percent (доля).** Время нахождения тэга в каждом состоянии, выраженное в процентах от общей длительности подынтервала.

Указание того или иного типа подсчёта выполняется путём присваивания параметру wwStateCalc соответствующего значения, например:

```
wwStateCalc = 'Total'
```

Расчёты общей продолжительности и доли нахождения в том или ином состоянии всегда выполняются точно, расчёты минимальной, максимальной и средней величин связаны с "произвольно" устанавливаемыми границами подынтервалов (не всегда совпадающими с моментами изменения значений) и могут приводить к неожиданным результатам. Особенно часто это проявляется при извлечении данных о медленно изменяющихся тэгах за большие периоды времени.

Например, пусть какой-либо символьный тэг переходит из одного (из двух возможных) состояний в другое каждые 10 минут, а длительность подынтервала запроса равна двум часам. При наступлении очередного подынтервала фиксируется значение (состояние), которое этот тэг имел в данный момент. Если это состояние изменяется при входе в подынтервал на короткий промежуток времени, то, скорее всего, состояние, которое тэг имел в момент начала подынтервала, будет рассматриваться как

минимальное значение. Аналогично, состояние тэга на конец подынтервала будет "отсечено" в момент конца подынтервала, и эти два "отсечённых" состояния исказят расчёт средних значений.

Возвращаемые результаты

Значения, указанные для начала подынтервала, не являются обычными "начальными", или первыми, значениями. Скорее, они представляют собой результат расчётов для подынтервала, непосредственно предшествующего временному диапазону запроса. Подразумевается, что значения тэга на момент начала подынтервала будут иметь эту метку времени. При выборе режима отметки концом подынтервала данные каждого подынтервала будут иметь метку времени, совпадающую с моментом его завершения.

В запросах можно указывать столбец StateTime, однако это имеет смысл только в запросах к таблице History. Выдача запроса этого типа к другим таблицам приведёт к ошибке.

Результаты в столбце StateTime представляют собой время нахождения тэга в определённом состоянии (в миллисекундах) и выражаются как число с плавающей запятой (64-разрядная двоичная величина) за исключением состояния "Percent". Для запросов этого типа время нахождения в каждом из состояний, выраженное в процентах от общей длительности интервала, возвращается как число от 0.0 до 100.0.

Столбец vValue может использоваться для возврата символьных представления соответствующих состояний, обозначений тех состояний, для которых нет символьного представления, а также строки NULL для промежутков в значениях либо "плохих" значений.

В результатах запросов к расширенной таблице WideHistory каждому тэгу будет соответствовать отдельный столбец, в каждом из которых будет находиться тот же код, что и в столбце vValue результатов аналогичного запроса к "суженной" таблице. Соответствующее состояние тэга будет отражено в столбце vValue таблицы WideHistory. Если какое-либо состояние недействительно для данного тэга, в соответствующем столбце будет указана строка "NULL".

Пример 1

Следующий запрос возвращает минимальную длительность нахождения логического тэга SteamValve в каждом из его состояний. Следует отметить, что минимальные значения возвращаются как для предшествующего пятиминутного "фантомного" подынтервала, так и для единственного подынтервала извлечения от 10:00 до 10:05.

```
SELECT DateTime, TagName, vValue, StateTime,
wwStateCalc
    FROM History
      WHERE TagName IN ('SteamValve')
          AND DateTime >= '2005-04-17 10:00:00'
          AND DateTime <= '2005-04-17 10:05:00'
          AND wwCycleCount = 1
          AND wwRetrievalMode = 'ValueState'
          AND wwStateCalc = 'Min'
```

Результаты:

DateTime	TagName	vValue	StateTime	wwStateCalc
2005-04-17 10:00:00.000	SteamValve	0	35359.0	MINIMUM
2005-04-17 10:00:00.000	SteamValve	1	43749.0	MINIMUM

2005-04-17 10:05:00.000	SteamValve	0	37887.0	MINIMUM
2005-04-17 10:05:00.000	SteamValve	1	43749.0	MINIMUM

Пример 2

Следующий запрос возвращает максимальное время нахождения логического тэга SteamValve в каждом из состояний в таком же интервале времени, что и в запросе примера 1. Следует отметить, что минимальные и максимальные длительности единичного состояния приблизительно совпадают, сильно различаясь для нулевого состояния. Причина кроется в описанном эффекте "отсечки".

```
SELECT DateTime, TagName, vValue, StateTime,
wwStateCalc
    FROM History
      WHERE TagName IN ('SteamValve')
          AND DateTime >= '2005-04-17 10:00:00'
          AND DateTime <= '2005-04-17 10:05:00'
          AND wwCycleCount = 1
          AND wwRetrievalMode = 'ValueState'
          AND wwStateCalc = 'Max'
```

Результаты:

DateTime	TagName	vValue	StateTime	wwStateCalc
2005-04-17 10:00:00.000	SteamValve	0	107514.0	MAXIMUM
2005-04-17 10:00:00.000	SteamValve	1	43750.0	MAXIMUM
2005-04-17 10:05:00.000	SteamValve	0	107514.0	MAXIMUM
2005-04-17 10:05:00.000	SteamValve	1	43750.0	MAXIMUM

Пример 3

Следующий запрос возвращает общее время нахождения логического тэга в каждом из состояний. Системный параметр TimeStampRule в этом примере имеет значение "End" (значение по умолчанию), поэтому возвращаемые величины имеют метку времени конца подынтервала. Исследуемый период начинается в 00:00:00.000 13 апреля 2005 года и заканчивается в 00:00:00.000 14 апреля 2005 года.

```
SELECT DateTime, vValue, StateTime, wwStateCalc
    FROM History
      WHERE DateTime > '2005-04-13 00:00:00.000'
          AND DateTime <= '2005-04-14 00:00:00.000'
          AND TagName IN ('PumpStatus')
          AND wwRetrievalMode = 'ValueState'
          AND wwStateCalc = 'Total'
          AND wwCycleCount = 1
```

Результаты:

DateTime	vValue	StateTime	wwStateCalc
2005-04-14 00:00:00	NULL	1041674.0	TOTAL
2005-04-14 00:00:00	On	56337454.0	TOTAL

2005-04-14 00:00:00	Off	29020872.0	TOTAL

Пример 4

Следующий запрос возвращает время нахождения логического тэга в каждом из состояний, выраженное в процентах от общей длительности, в нескольких подынтервалах. Системный параметр TimeStampRule в этом примере имеет значение "End" (значение по умолчанию), поэтому возвращаемые величины имеют метку времени конца подынтервала. Следует отметить, что в первой строке содержатся результаты для периода с 22:00:00.000 3 июля 2003 года до 00:00:00.000 4 июля 2003 года.

Режим "Percent" является единственным, в котором в столбце результатов StateTime возвращается не время, а доля нахождения тэгов в том или ином состоянии, выраженная в процентах (от 0 до 100).

```
SELECT DateTime, vValue, StateTime, wwStateCalc
    FROM History
      WHERE DateTime >= '2003-07-04 00:00:00.000'
          AND DateTime <= '2003-07-05 00:00:00.000'
          AND TagName IN ('PumpStatus')
          AND Value = 1
          AND wwRetrievalMode = 'ValueState'
          AND wwStateCalc = 'Percent'
          AND wwCycleCount = 13
```

Результаты:

DateTime	vValue	StateTime	wwStateCalc
2003-07-04 00:00:00	1	50.885	PERCENT
2003-07-04 02:00:00	1	82.656	PERCENT
2003-07-04 04:00:00	1	7.082	PERCENT
2003-07-04 06:00:00	1	7.157	PERCENT
2003-07-04 08:00:00	1	55.580	PERCENT
2003-07-04 10:00:00	1	28.047	PERCENT
2003-07-04 12:00:00	1	47.562	PERCENT
2003-07-04 14:00:00	1	74.477	PERCENT
2003-07-04 16:00:00	1	40.450	PERCENT
2003-07-04 18:00:00	1	78.313	PERCENT
2003-07-04 20:00:00	1	54.886	PERCENT
2003-07-04 22:00:00	1	39.569	PERCENT
2003-07-05 00:00:00	1	50.072	PERCENT

Параметр wwTimeDeadband

Мёртвая зона по времени определяет временное разрешение данных, возвращаемых в режиме извлечения по изменению. Изменения в рамках мёртвой зоны по времени (в мс) не возвращаются.

Мёртвая зона по времени учитывается после указания в запросе параметра wwTimeDeadband.

Если режим извлечения циклический, все мёртвые зоны игнорируются. Мёртвая зона по времени действительна для аналоговых, логических и символьных тэгов.

"Базовое" значение мёртвой зоны обновляется каждый раз при возврате очередного значения, поэтому в качестве основы расчёта полосы нечувствительности используется последняя возвращённая величина. Решение о включении выборки в результаты запроса принимается после сравнения этой выборки с параметрами мёртвой зоны.

Следующие запросы возвращают значения аналогового тэга SysTimeSec.

Пример 1

В данном запросе указан возврат значений, изменявшихся в течение 5-секундного интервала.

```
SELECT DateTime, TagName, Value FROM History
    WHERE TagName = 'SysTimeSec'
      AND DateTime >= '2001-12-09 11:35'
      AND DateTime <= '2001-12-09 11:37'
      AND wwRetrievalMode = 'Delta'
      AND wwTimeDeadband = 5000
```

Результаты:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0.0
2001-12-09 11:35:06.000	SysTimeSec	6.0
2001-12-09 11:35:12.000	SysTimeSec	12.0
2001-12-09 11:35:18.000	SysTimeSec	18.0
2001-12-09 11:35:24.000	SysTimeSec	24.0
2001-12-09 11:35:30.000	SysTimeSec	30.0
2001-12-09 11:35:36.000	SysTimeSec	36.0
2001-12-09 11:35:42.000	SysTimeSec	42.0
2001-12-09 11:35:48.000	SysTimeSec	48.0
2001-12-09 11:35:54.000	SysTimeSec	54.0
2001-12-09 11:36:00.000	SysTimeSec	0.0
2001-12-09 11:36:06.000	SysTimeSec	6.0
2001-12-09 11:36:12.000	SysTimeSec	12.0
2001-12-09 11:36:18.000	SysTimeSec	18.0
2001-12-09 11:36:24.000	SysTimeSec	24.0
2001-12-09 11:36:30.000	SysTimeSec	30.0
2001-12-09 11:36:36.000	SysTimeSec	36.0
2001-12-09 11:36:42.000	SysTimeSec	42.0
2001-12-09 11:36:48.000	SysTimeSec	48.0
2001-12-09 11:36:54.000	SysTimeSec	54.0
2001-12-09 11:37:00.000	SysTimeSec	0.0

Пример 2

Следующий запрос возвращает только те данные, которые изменились в течение интервала длительностью 4900 мс.

```
SELECT DateTime, TagName, Value FROM History
    WHERE TagName = 'SysTimeSec'
      AND DateTime >= '2001-12-09 11:35'
      AND DateTime <= '2001-12-09 11:37'
      AND wwRetrievalMode = 'Delta'
      AND wwTimeDeadband = 4900
```

Результаты:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0.0
2001-12-09 11:35:05.000	SysTimeSec	5.0
2001-12-09 11:35:10.000	SysTimeSec	10.0
2001-12-09 11:35:15.000	SysTimeSec	15.0
2001-12-09 11:35:20.000	SysTimeSec	20.0
2001-12-09 11:35:25.000	SysTimeSec	25.0
2001-12-09 11:35:30.000	SysTimeSec	30.0
2001-12-09 11:35:35.000	SysTimeSec	35.0
2001-12-09 11:35:40.000	SysTimeSec	40.0
2001-12-09 11:35:45.000	SysTimeSec	45.0
2001-12-09 11:35:50.000	SysTimeSec	50.0
2001-12-09 11:35:55.000	SysTimeSec	55.0
2001-12-09 11:36:00.000	SysTimeSec	0.0
2001-12-09 11:36:05.000	SysTimeSec	5.0
2001-12-09 11:36:10.000	SysTimeSec	10.0
2001-12-09 11:36:15.000	SysTimeSec	15.0
2001-12-09 11:36:20.000	SysTimeSec	20.0
2001-12-09 11:36:25.000	SysTimeSec	25.0
2001-12-09 11:36:30.000	SysTimeSec	30.0
2001-12-09 11:36:35.000	SysTimeSec	35.0
2001-12-09 11:36:40.000	SysTimeSec	40.0
2001-12-09 11:36:45.000	SysTimeSec	45.0
2001-12-09 11:36:50.000	SysTimeSec	50.0
2001-12-09 11:36:55.000	SysTimeSec	55.0
2001-12-09 11:37:00.000	SysTimeSec	0.0

Пример 3

Следующий запрос возвращает только те данные, которые изменились в течение интервала длительностью 2000 мс.

```
SELECT DateTime, TagName, Value
    FROM History
      WHERE TagName IN ('SysTimeSec', 'SysTimeMin')
```

```
AND DateTime >= '2001-12-09 11:35'
AND DateTime <= '2001-12-09 11:36'
AND wwRetrievalMode = 'Delta'
AND wwTimeDeadband = 2000
```

Результаты:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0.0
2001-12-09 11:35:00.000	SysTimeMin	35.0
2001-12-09 11:35:03.000	SysTimeSec	3.0
2001-12-09 11:35:06.000	SysTimeSec	6.0
2001-12-09 11:35:09.000	SysTimeSec	9.0
2001-12-09 11:35:12.000	SysTimeSec	12.0
2001-12-09 11:35:15.000	SysTimeSec	15.0
2001-12-09 11:35:18.000	SysTimeSec	18.0
2001-12-09 11:35:21.000	SysTimeSec	21.0
2001-12-09 11:35:24.000	SysTimeSec	24.0
2001-12-09 11:35:27.000	SysTimeSec	27.0
2001-12-09 11:35:30.000	SysTimeSec	30.0
2001-12-09 11:35:33.000	SysTimeSec	33.0
2001-12-09 11:35:36.000	SysTimeSec	36.0
2001-12-09 11:35:39.000	SysTimeSec	39.0
2001-12-09 11:35:42.000	SysTimeSec	42.0
2001-12-09 11:35:45.000	SysTimeSec	45.0
2001-12-09 11:35:48.000	SysTimeSec	48.0
2001-12-09 11:35:51.000	SysTimeSec	51.0
2001-12-09 11:35:54.000	SysTimeSec	54.0
2001-12-09 11:35:57.000	SysTimeSec	57.0
2001-12-09 11:36:00.000	SysTimeSec	0
2001-12-09 11:36:00.000	SysTimeMin	36

Параметр wwValueDeadband

Мёртвая зона по значению определяет разрешающую способность данных, возвращаемых в режиме извлечения по изменению. Значения, отличающиеся от предыдущих на величину, меньшую мёртвой зоны, не возвращаются. Мёртвые зоны могут указываться в процентах от шкалы значений, в единицах измерения. Мёртвые зоны становятся действительными при указании в запросе соответствующего параметра. В циклическом режиме извлечения мёртвые зоны игнорируются.

"Базовое" значение мёртвой зоны обновляется каждый раз при возврате очередного значения, поэтому в качестве основы расчёта полосы нечувствительности используется последняя возвращённая величина. Решение о включении точки в результаты запроса принимается после сравнения этой точки с параметрами мёртвой зоны.

Следующие запросы возвращают значения аналогового тэга SysTimeSec. Минимальное значение этого тэга в единицах измерения равно 0, максимальное – 59.

Пример 1

Следующий запрос возвращает только те точки данных, которые отличаются друг от друга более чем на 10% от всей шкалы значений тэга. Мёртвая зона в 10% соответствует абсолютной величине отклонения в 5,9 единиц измерения значений тэга SysTimeSec.

```
SELECT DateTime, Value
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND DateTime >= '2001-12-09 11:35'
          AND DateTime <= '2001-12-09 11:37'
          AND wwRetrievalMode = 'Delta'
          AND wwValueDeadband = 10
```

Результаты:

DateTime	Value
2001-12-09 11:35:00.000	0.0
2001-12-09 11:35:06.000	6.0
2001-12-09 11:35:12.000	12.0
2001-12-09 11:35:18.000	18.0
2001-12-09 11:35:24.000	24.0
2001-12-09 11:35:30.000	30.0
2001-12-09 11:35:36.000	36.0
2001-12-09 11:35:42.000	42.0
2001-12-09 11:35:48.000	48.0
2001-12-09 11:35:54.000	54.0
2001-12-09 11:36:00.000	0.0
2001-12-09 11:36:06.000	6.0
2001-12-09 11:36:12.000	12.0
2001-12-09 11:36:18.000	18.0
2001-12-09 11:36:24.000	24.0
2001-12-09 11:36:30.000	30.0
2001-12-09 11:36:36.000	36.0
2001-12-09 11:36:42.000	42.0
2001-12-09 11:36:48.000	48.0
2001-12-09 11:36:54.000	54.0
2001-12-09 11:37:00.000	0.0

Пример 2

Следующий запрос возвращает данные, отличающиеся не менее чем на 5% от полной шкалы значений тэга, выраженных в единицах измерения. Мёртвая зона в 5% соответствует абсолютному отклонению в 2,95 единиц измерения значений тэга SysTimeSec.

```
SELECT DateTime, Value
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND DateTime >= '2001-12-09 11:35'
          AND DateTime <= '2001-12-09 11:37'
          AND wwRetrievalMode = 'Delta'
          AND wwValueDeadband = 5
```

Результаты:

DateTime	Value
2001-12-09 11:35:00.000	0.0
2001-12-09 11:35:03.000	3.0
2001-12-09 11:35:06.000	6.0
2001-12-09 11:35:09.000	9.0
2001-12-09 11:35:12.000	12.0
2001-12-09 11:35:15.000	15.0
2001-12-09 11:35:18.000	18.0
2001-12-09 11:35:21.000	21.0
2001-12-09 11:35:24.000	24.0
2001-12-09 11:35:27.000	27.0
2001-12-09 11:35:30.000	30.0
2001-12-09 11:35:33.000	33.0
2001-12-09 11:35:36.000	36.0
2001-12-09 11:35:39.000	39.0
2001-12-09 11:35:42.000	42.0
2001-12-09 11:35:45.000	45.0
2001-12-09 11:35:48.000	48.0
2001-12-09 11:35:51.000	51.0
2001-12-09 11:35:54.000	54.0
2001-12-09 11:35:57.000	57.0
2001-12-09 11:36:00.000	0.0
2001-12-09 11:36:03.000	3.0
2001-12-09 11:36:06.000	6.0
2001-12-09 11:36:09.000	9.0
2001-12-09 11:36:12.000	12.0
2001-12-09 11:36:15.000	15.0
2001-12-09 11:36:18.000	18.0
2001-12-09 11:36:21.000	21.0
2001-12-09 11:36:24.000	24.0
2001-12-09 11:36:27.000	27.0
2001-12-09 11:36:30.000	30.0

2001-12-09 11:36:33.000	33.0
2001-12-09 11:36:36.000	36.0
2001-12-09 11:36:39.000	39.0
2001-12-09 11:36:42.000	42.0
2001-12-09 11:36:45.000	45.0
2001-12-09 11:36:48.000	48.0
2001-12-09 11:36:51.000	51.0
2001-12-09 11:36:54.000	54.0
2001-12-09 11:36:57.000	57.0
2001-12-09 11:37:00.000	0.0

Параметр wwTimeZone

Начиная с восьмой версии архиватора IndustrialSQL Server, все показания времени сохраняются в архиве в формате UTC (Universal Time Coordinated – Единое скоординированное время). Параметр wwTimeZone позволяет указывать, какой часовой пояс следует учитывать при извлечении данных. Подсистема извлечения будет преобразовывать все метки времени в местное время в данном часовом поясе.

Параметру wwTimeZone может быть присвоено любое из значений, хранящихся в столбце TimeZone таблицы TimeZone Рабочей базы данных. Указать часовой пояс можно не только путём присвоения параметру wwTimeZone соответствующего кода, но и путём определения значения параметра TimeZoneID (как строки символов). Например, записи "wwTimeZone = 'Mountain Standard Time'" and "wwTimeZone = '64'" в американской системе дадут один и тот же результат.

Таблица TimeZone заполняется значениями из Реестра операционной системы Windows при каждом запуске системы и, таким образом, содержит сведения обо всех часовых поясах, определённых в этой операционной системе (включая сведения о часовых поясах, определённых в пакетах обновления или в устанавливаемом программном обеспечении).

Подсистема извлечения учитывает момент перехода на летнее и зимнее время в указанном часовом поясе. При этом используются параметры базовой операционной системы. Подсистема извлечения не предоставляет никаких возможностей использования клиентских установок.

Если значение параметра wwTimeZone не определено, при извлечении данных учитывается часовой пояс того компьютера, в котором исполняется приложение архиватора IndustrialSQL Server.

Пример:

```
SELECT TagName, DateTime, Value, wwTimeZone
    FROM History
      WHERE TagName IN ('SysTimeHour', 'SysTimeMin')
          AND DateTime >= '2001-12-20 0:00'
          AND DateTime <= '2001-12-20 0:05'
          AND wwRetrievalMode = 'Delta'
          AND wwTimeZone = 'W. Europe Standard Time'
```

Результаты:

TagName	DateTime	Value	wwTimeZone

SysTimeMin	2001-12-20 00:00:00.000	0.0	W. Europe Standard Time
SysTimeHour	2001-12-20 00:00:00.000	15.0	W. Europe Standard Time
SysTimeMin	2001-12-20 00:01:00.000	1.0	W. Europe Standard Time
SysTimeMin	2001-12-20 00:02:00.000	2.0	W. Europe Standard Time
SysTimeMin	2001-12-20 00:03:00.000	3.0	W. Europe Standard Time
SysTimeMin	2001-12-20 00:04:00.000	4.0	W. Europe Standard Time
SysTimeMin	2001-12-20 00:05:00.000	5.0	W. Europe Standard Time

Одновременное использование в запросе и функций даты и времени и параметра wwTimeZone обеспечивается функцией faaTZgetdata(). Подробнее см. раздел "Использование функций даты и времени" настоящего руководства.

Параметр wwVersion

Параметр wwVersion позволяет указывать в запросе версию требуемых данных. Версии данных представляют собой новые значения в базе данных, создаваемые подсистемой хранения в результате выполнения операций вставки или обновления. Каждое вставляемое или обновлённое значение сопровождается датой создания, на основе которой и определяется версия данных.

При выполнении нескольких операций вставки или обновления в базе данных будет несколько версий данных с разными метками времени. Однако подсистема извлечения может представлять клиентам данные только двух версий: "ORIGINAL" (первоначальные, или исходные значения) и "LATEST" (последние значения).

Значение "REALTIME" параметра wwVersion недействительно при извлечении данных.

Примечание. Обозначения первоначальной (ORIGINAL) и последней (LATES) версий данных могут записываться в любом регистре.

Если в запросе версия не будет указана, возвращены будут последние модифицированные значения.

Пример запроса:

```
SELECT TagName, DateTime, Value, wwVersion
    FROM History
      WHERE TagName IN ('SysTimeHour', 'SysTimeMin')
          AND DateTime >= '2001-12-20 0:00'
          AND DateTime <= '2001-12-20 0:05'
          AND wwRetrievalMode = 'Delta'
          AND wwVersion = 'Original'
```

Результаты:

TagName	DateTime	Value	wwVersion
SysTimeMin	2001-12-20 00:00:00.000	0.0	ORIGINAL
SysTimeHour	2001-12-20 00:00:00.000	0.0	ORIGINAL
SysTimeMin	2001-12-20 00:01:00.000	1.0	ORIGINAL
SysTimeMin	2001-12-20 00:02:00.000	2.0	ORIGINAL
SysTimeMin	2001-12-20 00:03:00.000	3.0	ORIGINAL

SysTimeMin	2001-12-20 00:04:00.000	4.0	ORIGINAL
SysTimeMin	2001-12-20 00:05:00.000	5.0	ORIGINAL

Параметр wwInterpolationType

Там, где осуществляется интерполяция, она обозначает способ определения возвращаемого при выполнении запроса аналогового значения, которое соответствует границе подынтервала. Допустимые значения параметра: "STAIRSTEP" (последнее значение) и "LINEAR" (линейное приближение). Если в запросе указан тип "STAIRSTEP", никакой интерполяции не выполняется, просто возвращается последнее известное значение. Если известных значений нет, возвращается NULL. Если в запросе указан тип "LINEAR", для данного момента времени система вычисляет соответствующее значение методом линейного приближения последней известной и первой, следующей вслед за текущим подынтервалом, точек данных.

Как правило, тип интерполяции определяется типом извлекаемых данных. Например, при извлечении информации о температуре (которая менялась линейно) целесообразнее пользоваться методом линейного приближения, а при извлечении логических установок (например состояний вентиля) – методом "последнего значения". В целом, методом линейного приближения рекомендуется пользоваться для получения информации об обычных условиях, а методом "последнего значения" – для получения данных об исключительных ситуациях.

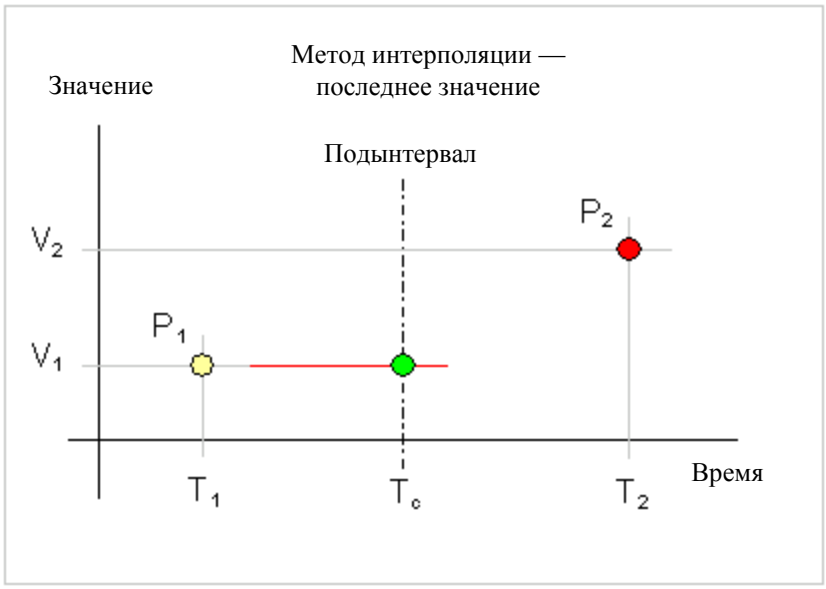
Установка параметра в определённое значение недействительна для неаналоговых тэгов в запросах с циклическим режимом, поскольку в циклическом режиме извлечения всегда возвращаются последние значения.

Линейное приближение может быть задано на трёх уровнях системы:

- Как общесистемный параметр архиватора IndustrialSQL Server: или "последнее значение", или "линейное приближение".
- Как установка для отдельного тэга. Метод интерполяции значений отдельного тэга определяется с помощью менеджера конфигурирования (Configuration Manager). Индивидуальная установка отменяет общесистемную.
- Как значение параметра wwInterpolationType в запросе. Явное присваивание этому параметру какого-либо значения отменяет как индивидуальную, так и общесистемную установку.

При возникновении всех следующих условий подсистема извлечения просто возвращает значение V_1 как соответствующее моменту T_C:

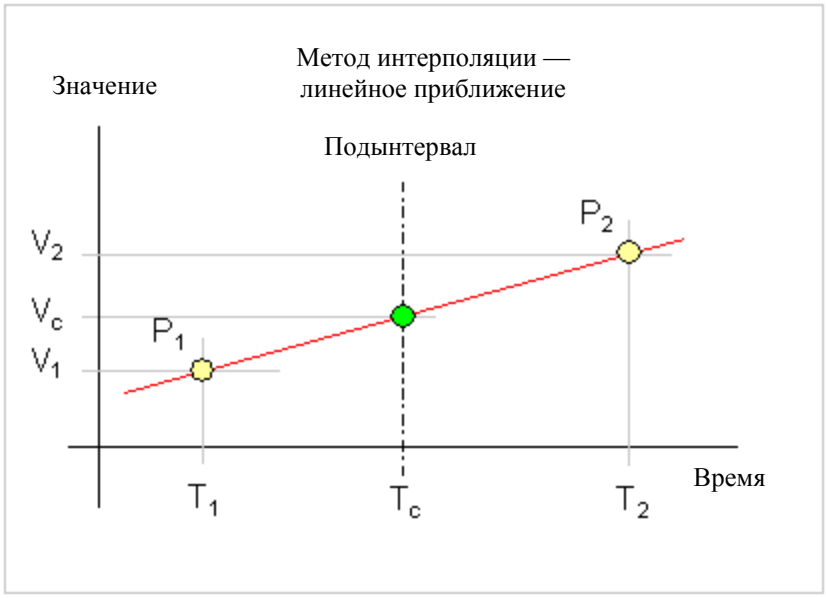
- Тип интерполяции – STAIRSTEP.
- Подсистема извлечения должна вернуть значение, соответствующее границе подынтервала с моментом T_C.
- Предшествующее сохранённое значение тэга – P_1.



Аналогично, для метода линейного приближения, если какая-либо из точек представляет значение NULL, для момента T_C будет возвращено значение P_1. Если ни одна из двух точек не представляет значение NULL, требуемое значение рассчитывается по следующей формуле линейного приближения:

$$V_C = V_1 + ((V_2 – V_1) * (T_C – T_1) / (T_2 – T_1))),$$

в которой разница $(T_2 – T_1)$ должна быть отлична от 0.



Параметр wwInterpolationType используется как входной (для отмены установок для отдельных тэгов), так и выходной (для отображения в каждой строке результатов метода расчёта соответствующего значения тэга).

Чтобы при выполнении запроса всегда использовался метод линейного приближения, в конструкции WHERE нужно указать следующую строку:

```
AND wwInterpolationType = 'Linear'
```

Чтобы при выполнении запроса всегда использовался метод "последнего значения", в конструкции WHERE нужно указать следующую строку:

```
AND wwInterpolationType = 'StairStep'
```

Параметр wwTimeStampRule

С помощью этого параметра определяется режим генерации меток времи возвращаемых циклических значений: моментом начала либо моментом конца подынтервала извлечения. Допустимые значения: "START" и "END". Если правило определения меток времи в запросе не указано, учитывается значение системного параметра TimeStampRule.

Чтобы возвращаемые значения помечались моментом начала подынтервала, в запросе нужно указать следующее:

```
AND wwTimeStampRule = 'Start'
```

Чтобы возвращаемые значения помечались моментом конца подынтервала, в запросе нужно указать следующее:

```
AND wwTimeStampRule = 'End'
```

При указании параметра wwTimeStampRule как одного из извлекаемых столбцов оператора SELECT в результатах будет показано, какое правило генерации меток времи применялось для каждой строки.

Параметр wwQualityRule

Этот параметр позволяет указывать, данные какого качества следует возвращать в результатах выполнения запроса. Его установка отменяет действие системного параметра QualityRule. Допустимые значения: 'GOOD' и 'EXTENDED'. Первое означает, что данные с неудовлетворительным качеством OPC не будут использованы для вычисления характеристик возвращаемого набора данных. 'EXTENDE' означает, что в расчётах будут учитываться данные как с хорошим, так и с неудовлетворительным качеством.

Чтобы исключить из набора возвращаемых результатов данные неудовлетворительного качества, в запросе укажите следующую строку:

```
AND wwQualityRule = 'Good'
```

Учёт данных любого качества задаётся следующей строкой:

```
AND wwQualityRule = 'Extended'
```

При указании параметра wwQualityRule как одного из извлекаемых столбцов оператора SELECT в результатах будет показано, какое правило определения качества применялось для каждой строки.

Параметр wwEdgeDetection

Событием называется момент выполнения некоторых условий, определённых для того или иного тэга системы архиватора IndustrialSQL Server. В общем, событием можно назвать всё, что можно определить на основе анализа сохранённых данных.

При обнаружении событий иногда бывает желательным выделять те строки набора возвращаемых данных, в которых значение условия, заданного конструкцией WHERE, меняется с истинного на ложное или наоборот, например, в какой момент уровень жидкости в резервуаре стал превышать 1 м. В соответствующем запросе в конструкции WHERE может быть записано условие типа 'TANKLEVEL > 1'. Если уровень жидкости не превышает 1 м, значение этого условия будет 'FALSE'. Событие возникнет, когда данное условие перейдёт из ложного состояния в истинное. Воображаемая "граница" между этими двумя состояниями называется фронтом.

С течением времени данное условие может неоднократно пересекать этот "фронт", переходя из одного состояния в другое. Система сможет определять значения событийного тэга по обеим сторонам "фронта", если в

определении тэга указать эту возможность. Существует четыре варианта обнаружения фронта: none (отсутствие), leading (передний фронт), trailing (задний фронт), both (оба фронта). В зависимости от выбора того или иного варианта будут получены различные наборы результатов:

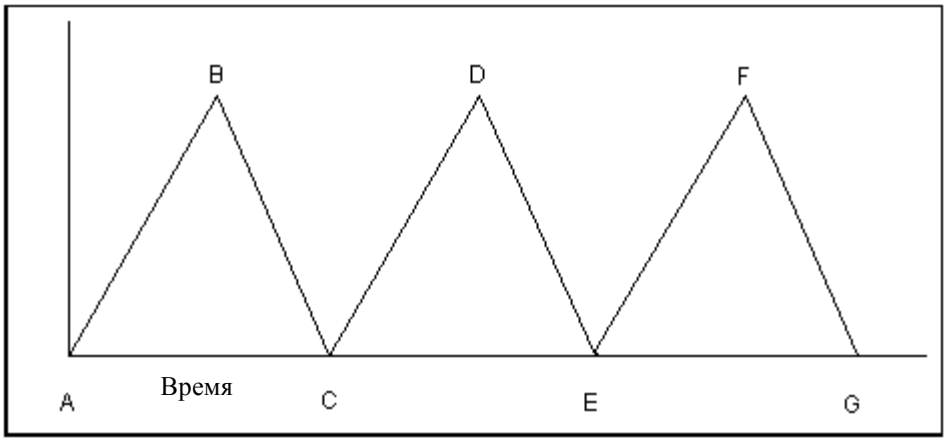
Вариант	Набор результатов
None	Все строки, в которых указанное условие выполняется, обнаружение фронта не выполняется.
Leading	Строки, в которых данное условие начинает выполняться, непосредственно следующие за строками, в которых условие не выполнялось (то есть переход "false -> true").
Trailing	Строки, в которых данное условие перестаёт выполняться, непосредственно следующие за строками, в которых это условие выполнялось (то есть переход " true -> false ").
Both	Все строки, в которых выполняются оба типа указанных переходов.

Определение фронта действительно только для аналоговых и логических детекторов. Кроме того, определение фронтов для тэгов разного типа слегка различается.

Дополнительно о событиях см. в Главе 7 "Подсистема событий".

Определение фронтов для аналоговых тэгов

Предположим, что набор результатов выполнения запросов выглядит следующим образом:



Прямые A-B, C-D и E-F будут передними фронтами, а прямые B-C, D-E и F-G – задними. Вид этих прямых определяется выполнением условия конструкции WHERE, в котором установка значения параметра wwEdgeDetection объединена с оператором сравнения с заданной величиной.

В следующей таблице содержится описание наборов строк, которые могут быть получены в зависимости от различных значений параметра wwEdgeDetection.

Оператор	=	<	>	<=	>=
Передний фронт	Передний и задний фронты; первое значение, удовлетворяющее условию	Только задний фронт; первое значение, удовлетворяющее условию*	Только передний фронт; первое значение, удовлетворяющее условию	Только задний фронт; первое значение, удовлетворяющее условию*	Только передний фронт; первое значение, удовлетворяющее условию
Задний	Передний и	Только передний	Только задний	Только передний	Только задний

фронт	задний фронты; первое значение, которое не удовлетворяет условию, после значения, которое ему удовлетворяет.	фронт; первое значение, которое не удовлетворяет условию.	фронт; первое значение, которое не удовлетворяет условию*.	фронт; первое значение, которое не удовлетворяет условию.	фронт; первое значение, которое не удовлетворяет условию*.

* если задний фронт представляет собой вертикальную линию, в результатах будет возвращено минимальное значение на этой линии.

При выполнении следующего запроса возвращаются все значения тэга SysTimeSec из таблицы History, которые превышают число 50 и были получены между 10:99 и 10:02 второго декабря 2001 года. Определение фронтов не указано.

```
SELECT DateTime, Value
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND DateTime >= '2001-12-02 10:00:00'
          AND DateTime <= '2001-12-02 10:02:00'
          AND wwRetrievalMode = 'Cyclic'
          AND wwResolution = 2000
          AND Value >= 50
          AND wwEdgeDetection = 'None'
```

Результаты:

DateTime	Value
2001-12-02 10:00:50.000	50
2001-12-02 10:00:52.000	52
2001-12-02 10:00:54.000	54
2001-12-02 10:00:56.000	56
2001-12-02 10:00:58.000	58
2001-12-02 10:01:50.000	50
2001-12-02 10:01:52.000	52
2001-12-02 10:01:54.000	54
2001-12-02 10:01:56.000	56
2001-12-02 10:01:58.000	58

Определение переднего фронта для аналоговых тэгов

Если в параметрах запроса указано определение переднего фронта, результирующий набор будет состоять только из тех строк, в которых заданное конструкцией WHERE условие стало выполняться (то есть имеет значение 'TRUE') после того, как оно не выполнялось (то есть имело значение 'FALSE').

Следующий запрос возвращает первые из всех значений тэга SysTimeSec, которые удовлетворяют условию, налагаемому на значение Value, и которые попадают в период от 10:00 до 10:02 второго декабря 2001 года.

```
SELECT DateTime, Value
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND DateTime >= '2001-12-02 10:00:00'
          AND DateTime <= '2001-12-02 10:02:00'
          AND wwRetrievalMode = 'Cyclic'
          AND wwResolution = 2000
          AND Value >= 50
          AND wwEdgeDetection = 'Leading'
```

Результаты:

DateTime	Value
2001-12-02 10:00:50.000	50
2001-12-02 10:01:50.000	50

Следует отметить различие между этими результатами и результатами выполнения запроса из предыдущего раздела, в котором определение фронтов не было указано. Если в первом случае было возвращено 10 строк, то в этом – только две, в которых содержатся первые значения, удовлетворяющие условию конструкции WHERE.

Определение задних фронтов для аналоговых тэгов

Если в параметрах запроса указано определение заднего фронта, результирующий набор будет состоять только из тех строк, в которых заданное конструкцией WHERE условие перестало выполняться (то есть имеет значение 'FALSE') после того, как оно выполнялось (то есть имело значение 'TRUE').

Следующий запрос возвращает первые из всех значений тэга SysTimeSec, которые не удовлетворяют условию, налагаемому на значение Value, и которые попадают в период от 10:00 до 10:02 второго декабря 2001 года.

```
SELECT DateTime, Value
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND DateTime >= '2001-12-02 10:00:00'
          AND DateTime <= '2001-12-02 10:02:00'
          AND wwRetrievalMode = 'Cyclic'
          AND wwResolution = 2000
          AND Value >= 50
          AND wwEdgeDetection = 'Trailing'
```

Запрос возвращает только две строки для указанного временного диапазона, в которых заданное условие перестало выполняться:

DateTime	Value
2001-12-02 10:01:00.000	00
2001-12-02 10:02:00.000	00

Следует отметить различие между этими результатами и результатами выполнения запроса из раздела "Определение фронтов для аналоговых

тэгов", в котором определение фронтов не было указано. Если в первом случае было возвращено 10 строк, то в этом – только две, в которых содержатся первые значения, переставшие удовлетворять условию конструкции WHERE.

Определение обоих фронтов для аналоговых тэгов

Если в запросе указано определение фронтов обоих типов, в результатах будут возвращены все строки, удовлетворяющих либо одному, либо другому условию образования фронта.

Следующий запрос возвращает все строки, в которых был обнаружен переход от выполнения условия конструкции WHERE к невыполнению или наоборот:

```
SELECT DateTime, Value
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND DateTime >= '2001-12-02 10:00:00'
          AND DateTime <= '2001-12-02 10:02:00'
          AND wwRetrievalMode = 'Cyclic'
          AND wwResolution = 2000
          AND Value >= 50
          AND wwEdgeDetection = 'Both'
```

Результаты:

DateTime	Value
2001-12-02 10:00:50.000	50
2001-12-02 10:01:00.000	00
2001-12-02 10:01:50.000	50
2001-12-02 10:02:00.000	00

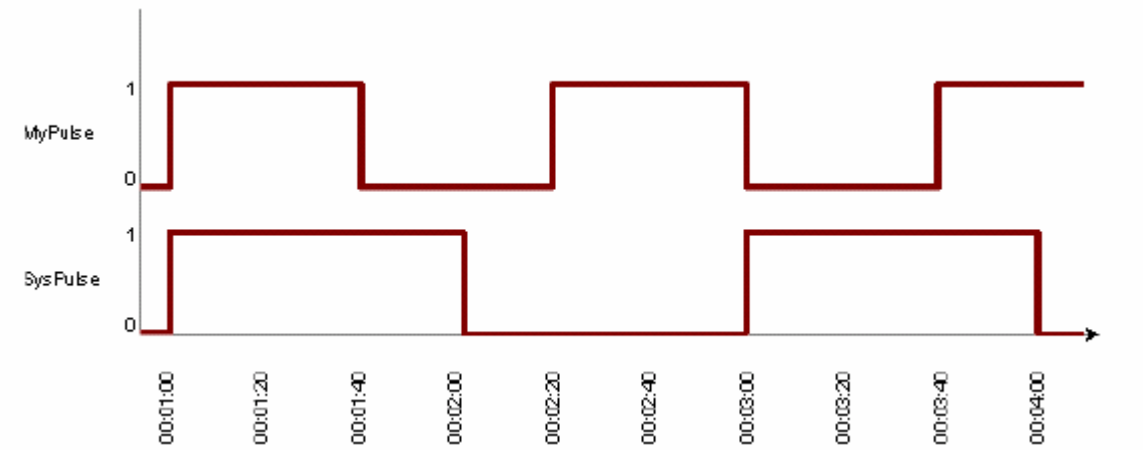
В этих результатах присутствует только одна строка из таблицы результатов выполнения запроса из раздела "Определение фронтов для аналоговых тэгов", в котором определение фронтов не было указано.

Определение фронтов для логических тэгов

Определение фронтов для логических тэгов выполняется несколько по-другому, нежели для аналоговых. Например, предположим, что в архиве хранятся значения следующих логических тэгов:

Тэг	Описание
SysPulse	Переходы от 1 к 0 и обратно каждую минуту.
MyPulse	Переходы от 1 к 0 и обратно каждые 40 с.

Изменения значений этих тэгов во времени показаны на следующем рисунке:



Следующие запросы извлекают из таблиц History и WideHistory единичные значения тэгов SysPulse и MyPulse, которые они имели в период с 00:59 до 01:04 восьмого декабря 2001 года. Определение фронтов не указано.

Пример 1

Запрос к таблице *History*

```
SELECT DateTime, TagName, Value
    FROM History
      WHERE TagName IN ('SysPulse', 'MyPulse')
          AND DateTime > '2001-12-08 00:59:00'
          AND DateTime <= '2001-12-08 01:04:00'
          AND wwRetrievalMode = 'Delta'
          AND Value = 1
          AND wwEdgeDetection = 'None'
```

Результаты:

DateTime	TagName	Value
2001-12-08 00:01:00.000	SysPulse	1
2001-12-08 00:01:00.000	MyPulse	1
2001-12-08 00:02:20.000	MyPulse	1
2001-12-08 00:03:00.000	SysPulse	1
2001-12-08 00:03:40.000	MyPulse	1

Пример 2

Запрос к таблице *WideHistory*

```
SELECT * FROM OPENQUERY(INSQL, 'SELECT DateTime,
SysPulse, MyPulse
    FROM WideHistory
      WHERE DateTime > "2001-12-08 00:59:00"
          AND DateTime < "2001-12-08 01:05:00"
          AND SysPulse = 1
          AND MyPulse = 1
          AND wwRetrievalMode = "Delta"
          AND wwEdgeDetection = "None"
```

```
')
```

Результаты:

DateTime	SysPulse	MyPulse
2001-12-08 00:01:00.000	1	1

Определение переднего фронта для логических тэгов

Если в параметрах запроса указано определение переднего фронта, результирующий набор будет состоять только из тех строк, в которых заданное конструкцией WHERE условие стало выполняться (то есть имеет значение 'TRUE') после того, как оно не выполнялось (то есть имело значение 'FALSE').

Следующие запросы извлекают из таблиц History и WideHistory единичные значения тэгов SysPulse и MyPulse, которые они имели в период с 00:59 до 01:04 восьмого декабря 2001 года.

Пример 1

Запрос к таблице *History*

Если в конструкции WHERE запроса к таблице History содержится условие извлечения только единичных логических значений, указание переднего фронта не приводит к изменению результирующего набора данных:

```
SELECT DateTime
    FROM History
      WHERE TagName IN ('SysPulse', 'MyPulse')
          AND DateTime > '2001-12-08 00:59:00'
          AND DateTime <= '2001-12-08 01:04:00'
          AND Value = 1
          AND wwEdgeDetection = 'Leading'
```

Результаты:

DateTime
2001-12-08 00:01:00.000
2001-12-08 00:01:00.000
2001-12-08 00:02:20.000
2001-12-08 00:03:00.000
2001-12-08 00:03:40.000

Пример 2

Запрос к таблице *WideHistory*

Указание в запросе к таблице WideHistory условия определения переднего фронта означает, что условие только тогда примет значение 'TRUE', когда оба тэга будут равны 1.

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime
    FROM WideHistory
      WHERE DateTime > "2001-12-08 00:59:00"
          AND DateTime <= "2001-12-08 01:04:00"
          AND SysPulse = 1
```

```
            AND MyPulse = 1
            AND wwEdgeDetection = "Leading"
    ')
```

Результаты:

DateTime

2001-12-08 00:01:00.000

2001-12-08 00:03:40.000

Сравним полученные результаты с результатами выполнения запроса, в котором не было указано условие определения фронтов. Можно ожидать, что в результатах должна быть строка с отметкой 00:03:00; однако, поскольку именно в этот момент оба тэга меняют своё состояние, их значения не возвращаются. На практике бывает маловероятно, чтобы два каких-либо параметра меняли своё значение в одно и то же время.

Определение заднего фронта для логических тэгов

Если в параметрах запроса указано определение заднего фронта, результирующий набор будет состоять только из тех строк, в которых заданное конструкцией WHERE условие перестало выполняться (то есть имеет значение 'FALSE') после того, как оно выполнялось (то есть имело значение 'TRUE').

Пример 1

Запрос к таблице *History*

Если в конструкции WHERE запроса к таблице History содержится условие извлечения только единичных логических значений, указание условия определения заднего фронта аналогично инверсии условия конструкции WHERE (то есть если бы было указано "AND Value = 0"):

```
SELECT DateTime
    FROM History
      WHERE TagName IN ('SysPulse', 'MyPulse')
          AND DateTime > '2001-12-08 00:59:00'
          AND DateTime <= '2001-12-08 01:04:00'
          AND Value = 1
          AND wwEdgeDetection = 'Trailing'
```

Результаты:

DateTime

2001-12-08 00:01:40.000

2001-12-08 00:02:00.000

2001-12-08 00:03:00.000

2001-12-08 00:04:00.000

Пример 2

Запрос к таблице *WideHistory*

Указание в запросе к таблице WideHistory условия определения заднего фронта приводит к возврату границ подынтервалов, на которых условие переставало выполняться (то есть одно из значений было равно 0).

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime
    FROM WideHistory
      WHERE DateTime > "2001-12-08 00:59:00"
          AND DateTime <= "2001-12-08 01:04:00"
          AND SysPulse = 1
          AND MyPulse = 1
          AND wwEdgeDetection = "Trailing"
    ')
```

Результаты:

DateTime

2001-12-08 00:01:40.000

2001-12-08 00:04:00.000

Сравним полученные результаты с результатами выполнения запроса, в котором не было указано условие определения фронтов. Можно было бы ожидать, что в результатах должна быть строка с отметкой 00:03:00; однако, поскольку именно в этот момент оба тэга меняют своё состояние, их значения не возвращаются. На практике бывает маловероятно, чтобы два каких-либо параметра меняли своё значение в одно и то же время.

Определение фронтов обоих типов для логических тэгов

Если в запросе указано определение фронтов обоих типов, в результатах будут возвращены все строки, удовлетворяющих либо одному, либо другому условию образования фронта.

Следующие запросы извлекают из таблиц History и WideHistory единичные значения тэгов SysPulse и MyPulse, которые они имели в период с 00:59 до 01:04 восьмого декабря 2001 года с условием определения фронтов обоих типов.

Пример 1

Запрос к таблице *History*

```
SELECT DateTime
    FROM History
      WHERE TagName IN ('SysPulse', 'MyPulse')
          AND DateTime > '2001-12-08 00:59:00'
          AND DateTime <= '2001-12-08 01:04:00'
          AND Value = 1
          AND wwEdgeDetection = 'Both'
```

Результаты:

DateTime

2001-12-08 00:01:00.000

2001-12-08 00:01:40.000

2001-12-08 00:02:00.000

2001-12-08 00:02:20.000

2001-12-08 00:03:00.000

2001-12-08 00:03:40.000

2001-12-08 00:04:00.000

Пример 2

Запрос к таблице *WideHistory*

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime
    FROM WideHistory
      WHERE DateTime > "2001-12-08 00:59:00"
          AND DateTime <= "2001-12-08 01:04:00"
          AND SysPulse = 1
          AND MyPulse = 1
          AND wwEdgeDetection = "Both"
    ')
```

Результаты:

DateTime

2001-12-08 00:01:00.000

2001-12-08 00:01:40.000

2001-12-08 00:03:40.000

2001-12-08 00:04:00.000

Примеры построения запросов

В этом разделе описаны запросы более сложной структуры, чем рассмотренные в предыдущих разделах.

Приведённые примеры не являются исчерпывающими вариантами запросов к базе данных, скорее, они демонстрируют возможности собственной разработки.

Подробнее о создании SQL-запросов см. в документации к Microsoft SQL Server.

Примечание. Если Microsoft SQL Server чувствителен к регистру вводимых символов, и в запросах также следует соблюдать использование правильного регистра.

Запросы к таблице History

Таблица History используется для хранения производственных данных в архивном формате. Дополнительнсм. Главу 1 "Категории таблиц" Справочника по базе данных архиватора IndustrialSQL Server.

Следующий запрос возвращает метки времени и значения тэга с названием ReactLevel. В запросе используется представление удалённой таблицы ("History" вместо "INSQL.Runtime.dbo.History"):

```
SELECT DateTime, TagName, Value
    FROM History
      WHERE TagName = 'ReactLevel'
          AND DateTime >= '2001-03-02 6:08'
          AND DateTime <= '2001-03-02 6:28'
```

```
        AND wwRetrievalMode = 'Cyclic'
```

Результаты:

DateTime	TagName	Value
2001-03-02 06:08:00.000	ReactLevel	2000.0
2001-03-02 06:08:00.000	ReactLevel	65.0
2001-03-02 06:08:00.000	ReactLevel	2025.0
2001-03-02 06:10:00.000	ReactLevel	0.0
2001-03-02 06:10:00.000	ReactLevel	2000.0
2001-03-02 06:11:00.000	ReactLevel	-25.0
2001-03-02 06:11:00.000	ReactLevel	2025.0
2001-03-02 06:12:00.000	ReactLevel	0.0
2001-03-02 06:13:00.000	ReactLevel	000.0

…

100 row(s) affected
(Найдено 100 строк)

Запрос к таблице Live

В таблице Live хранятся последние значения каждого тэга. Дополнительн см. Главу 1 "Категории таблиц" Справочника по базе данных архиватора IndustrialSQL Server.

Следующий запрос возвращает текущее значение указанного тэга. В запросе используется представление удалённой таблицы ("Live" вместо "INSQL.Runtime.dbo.Live"):

```
SELECT TagName, Value
    FROM Live
      WHERE TagName = 'ReactLevel'
```

Результаты:

TagName	Value
ReactLevel	1145.0

1 row(s) affected (Найдена 1 строка)

Запрос к таблице WideHistory

Расширенная таблица WideHistory представляет собой модификацию таблицы History. Запросы к этой таблице обычно выдаются для получения значений одного или нескольких тэгов, удовлетворяющих каким-либо условиям, в определённые периоды времени.

Дополнительн см. Главу 1 "Категории таблиц" Справочника по базе данных архиватора IndustrialSQL Server.

Следующий запрос возвращает из таблицы WideHistory значения двух тэгов. Обратиться к этой таблице можно только с использованием функции OPENQUERY. Квалификатор "Runtime.dbo" является необязательным.

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime,
ReactLevel, ReactTemp
    FROM Runtime.dbo.WideHistory
      WHERE Reactlevel > 1500
```

```
        AND ReactTemp > 150
    ')
```

Результаты:

DateTime	ReactLevel	ReactTemp
2001-03-02 06:20:00.000	1865.0	191.3
2001-03-02 06:21:00.000	2025.0	195.9
2001-03-02 06:22:00.000	2000.0	195.9
2001-03-02 06:23:00.000	2025.0	180.9
2001-03-02 06:27:00.000	1505.0	177.5

5 row(s) affected (найдено 5 строк)

Тип столбца таблицы WideHistory определяется по типу тэга.

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime,
SysTimeMin, SysPulse, SysString
    FROM WideHistory
      WHERE DateTime >= "2001-12-20 0:00"
          AND DateTime <= "2001-12-20 0:05"
          AND wwRetrievalMode = "delta"
    ')
```

Результаты:

DateTime	SysTimeMin	SysPulse	SysString
2001-12-20 00:00:00.000	0.0	0	2001/12/20 08:00:00
2001-12-20 00:01:00.000	1.0	1	2001/12/20 08:00:00
2001-12-20 00:02:00.000	2.0	0	2001/12/20 08:00:00
2001-12-20 00:03:00.000	3.0	1	2001/12/20 08:00:00
2001-12-20 00:04:00.000	4.0	0	2001/12/20 08:00:00
2001-12-20 00:05:00.000	5.0	1	2001/12/20 08:00:00

Запросы к расширенным таблицам в режиме извлечения по изменению

Извлечение данных из расширенных таблиц осуществляется обычным образом, если в запросе указан только один тэг. Если же в запросе указано несколько тэгов, клиенту будут возвращаться целые строки для каждого случая, когда значение одного или нескольких тэгов меняется. В строке будут содержаться новые значения изменившихся тэгов и прежние значения остальных тэгов в наборе (аналогично режиму циклического извлечения).

Примечание. Значения могут быть "недействительными", а также быть другого качества.

Следующий запрос возвращает из таблицы WideHistory значений трёх тэгов, причём тэг MyTagName периодически становится "недействительным".

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime,
SysTimeSec, SysTimeMin, MyTagName
    FROM WideHistory
```

```
        WHERE DateTime >= "2001-05-12 13:00:00"
            AND DateTime <= "2001-05-12 13:02:00"
            AND wwRetrievalMode = "Delta"
    ')
```

Результаты:

DateTime	SysTimeSec	SysTimeMin	MyTagName
:	:	:	:
:	:	:	:
2001-05-12 13:00:55.000	55	00	1
2001-05-12 13:00:56.000	56	00	1
2001-05-12 13:00:57.000	57	00	1
2001-05-12 13:00:57.500	57	00	null
2001-05-12 13:00:58.000	58	00	null
2001-05-12 13:00:59.000	59	00	null
2001-05-12 13:01:00.000	00	01	null
2001-05-12 13:01:00.500	00	01	2
2001-05-12 13:01:01.000	01	01	2
2001-05-12 13:01:02.000	02	01	2
2001-05-12 13:01:03.000	03	01	2
:	:	:	:
:	:	:	:
:	:	:	:

Следует отметить, что строка со значением 57 появляется в результатах дважды, поскольку тэг MyTagName меняет своё значение с 1 на null приблизительно между 57-й и 58-й секундами. То же самое происходит при изменении его значения с null на 2.

Таким же образом извлекаются значения логических тэгов.

Указание тэгов с нестандартными именами в запросах к расширенным таблицам

Нестандартные имена тэгов в SQL-запросах к расширенным таблицам должны быть заключены в квадратные скобки, поскольку они используются как имена столбцов. Например, имена с символом минуса (-) или символом наклонной черты (/) обязательно следует заключать в скобки, так как в противном случае блок грамматического разбора будет трактовать их как знаки арифметических операций. Ошибки в необязательном употреблении скобок нет. Дополнительные сведения о нестандартных именах можно найти в разделе "Правила именования тэгов" настоящего руководства.

Ниже приведён пример выделения названий тэгов в запросе к расширенной таблице. В данном случае ими являются "ReactTemp-2" и "ReactTemp+2". Если не указывать ограничители имени, части "-2" и "+2" будут рассматриваться как арифметические операции вычитания и сложения.

Вместе с тем настоятельно не рекомендуется без особой необходимости именовать тэги нестандартным образом, так как это приводит к ухудшению понятности запросов и, соответственно, повышению вероятности ошибок.

```
SELECT * FROM OpenQuery(INSQL, 'SELECT ReactTemp,
[ReactTemp-2]-2, [ReactTemp+2]+2 FROM WideHistory
WHERE ... ')
```

Использование конструкции INNER REMOTE JOIN

Использование конструкции INNER REMOTE JOIN более эффективно, чем указание в запросах строк вида " … WHERE TagName IN (SELECT TagName …) ".

В целом, запросы к архиватору с этой конструкцией рекомендуется строить следующим образом:

<Имя таблицы SQL Server> INNER REMOTE JOIN <Имя таблицы расширения архиватора>

Пример 1

Следующий запрос возвращает из архивной таблицы данные, определяемые названием символьного тэга, которое извлекается из таблицы StringTag:

```
SELECT DateTime, T.TagName, vValue, Quality,
QualityDetail
    FROM StringTag T inner remote join History H ON
    T.TagName = H.TagName
      WHERE T.MaxLength = 64
          AND DateTime >= '2002-03-10 12:00:00.000'
          AND DateTime <= '2002-03-10 16:40:00.000'
          AND wwRetrievalMode = 'Delta'
```

Пример 2

Следующий запрос возвращает из архивной таблицы данные, определяемые названием логического тэга, которое извлекается из таблицы Tag:

```
SELECT DateTime, T.TagName, vValue, Quality,
QualityDetail
    FROM Tag T inner remote join History H ON
    T.TagName = H.TagName
      WHERE T.TagType = 2
          AND T.Description like 'Discrete%'
          AND DateTime >= '2002-03-10 12:00:00.000'
          AND DateTime <= '2002-03-10 16:40:00.000'
          AND wwRetrievalMode = 'Delta'
```

Указание в запросах допусков на время и на значения

Если в запросе указана мёртвая зона по времени или по значениям, система проверяет каждую точку, используя значение предыдущей точки в качестве базовой величины для определения ширины полосы нечувствительности.

Если текущая точка проходит обе проверки, она включается в набор результатов и используется как базовое значение проверки следующей точки.

Пример:

```
SELECT DateTime, TagName, Value
```

```
FROM History
  WHERE TagName = 'ReactTemp'
      AND DateTime >= '2002-03-13 10:08'
      AND DateTime <= '2002-03-13 10:28'
      AND wwRetrievalMode = 'Delta'
      AND wwTimeDeadband = 5000
      AND wwValueDeadband = 5
```

Шкала значений тэга ReactTemp в единицах измерения начинается от 0 (MinEU) и заканчивается 220 (MaxEU). Таким образом, мёртвая зона по значению в 5% равна 11 единицам измерения, то есть 5% от (220 – 0). Значение ReactTemp колеблется между этими пределами довольно быстро, оставаясь при этом постоянным в течение коротких промежутков времени около верхнего и нижнего значений. Таким образом, при быстрых изменениях температуры мёртвая зона по значению нарушается первой, после чего нарушается мёртвая зона по времени. В этой области поведение определяется допуском на время, поэтому возвращаемые строки разнесены по времени на 5 с. Если температура более или мене постоянна (в частности, около нижнего температурного предела), сначала нарушается мёртвая зона по времени, а затем мёртвая зона по значению. Обе мёртвые зоны нарушаются, если температура отклоняется от предыдущего значения более чем на 11 градусов. Таким образом, влияние мёртвой зоны по времени больше проявляется около нижнего и верхнего предельных значений тэга.

Результаты выполнения запроса:

DateTime	TagName	Value
2002-03-13 10:08:00.000	ReactTemp	121.0
2002-03-13 10:08:10.000	ReactTemp	189.10000610351562
2002-03-13 10:08:20.000	ReactTemp	147.69999694824219
2002-03-13 10:08:30.000	ReactTemp	106.30000305175781
2002-03-13 10:08:40.000	ReactTemp	30.100000381469727
2002-03-13 10:08:50.000	ReactTemp	16.399999618530273
2002-03-13 10:09:00.000	ReactTemp	61.0
2002-03-13 10:09:10.000	ReactTemp	151.0
2002-03-13 10:09:20.000	ReactTemp	173.0
2002-03-13 10:09:30.000	ReactTemp	131.60000610351562
2002-03-13 10:09:40.000	ReactTemp	57.700000762939453
2002-03-13 10:09:50.000	ReactTemp	16.299999237060547
2002-03-13 10:10:10.000	ReactTemp	96.0
2002-03-13 10:10:20.000	ReactTemp	186.0
2002-03-13 10:10:30.000	ReactTemp	156.89999389648437
2002-03-13 10:10:40.000	ReactTemp	115.5
2002-03-13 10:10:50.000	ReactTemp	41.599998474121094
2002-03-13 10:11:00.000	ReactTemp	21.0
2002-03-13 10:11:10.000	ReactTemp	41.0
2002-03-13 10:11:20.000	ReactTemp	131.0

2002-03-13 10:11:30.000	ReactTemp	184.5
2002-03-13 10:11:40.000	ReactTemp	140.80000305175781
2002-03-13 10:11:50.000	ReactTemp	99.400001525878906
2002-03-13 10:12:00.000	ReactTemp	25.5
2002-03-13 10:12:20.000	ReactTemp	76.0
2002-03-13 10:12:30.000	ReactTemp	166.0
2002-03-13 10:12:50.000	ReactTemp	124.69999694824219
2002-03-13 10:13:00.000	ReactTemp	50.799999237060547
2002-03-13 10:13:10.000	ReactTemp	16.399999618530273
2002-03-13 10:13:30.000	ReactTemp	111.0
2002-03-13 10:13:40.000	ReactTemp	193.69999694824219
2002-03-13 10:13:50.000	ReactTemp	152.30000305175781
2002-03-13 10:14:00.000	ReactTemp	108.59999847412109
2002-03-13 10:14:10.000	ReactTemp	34.700000762939453
2002-03-13 10:14:20.000	ReactTemp	21.0
2002-03-13 10:14:30.000	ReactTemp	51.0
2002-03-13 10:14:40.000	ReactTemp	146.0
2002-03-13 10:14:50.000	ReactTemp	177.60000610351562
2002-03-13 10:15:00.000	ReactTemp	136.19999694824219
2002-03-13 10:15:10.000	ReactTemp	92.5
2002-03-13 10:15:20.000	ReactTemp	18.600000381469727
2002-03-13 10:15:40.000	ReactTemp	86.0
2002-03-13 10:15:50.000	ReactTemp	181.0
2002-03-13 10:16:00.000	ReactTemp	161.5
2002-03-13 10:16:10.000	ReactTemp	120.09999847412109
2002-03-13 10:16:20.000	ReactTemp	76.400001525878906
2002-03-13 10:16:30.000	ReactTemp	20.899999618530273
2002-03-13 10:16:50.000	ReactTemp	81.0
2002-03-13 10:17:00.000	ReactTemp	176.0
2002-03-13 10:17:10.000	ReactTemp	163.80000305175781
2002-03-13 10:17:20.000	ReactTemp	122.40000152587891
2002-03-13 10:17:30.000	ReactTemp	46.200000762939453
2002-03-13 10:17:40.000	ReactTemp	18.700000762939453
2002-03-13 10:18:00.000	ReactTemp	116.0
2002-03-13 10:18:10.000	ReactTemp	189.10000610351562
2002-03-13 10:18:20.000	ReactTemp	147.69999694824219

…

Указание в запросе параметров wwResolution, wwCycleCount и wwRetrievalMode

Результаты выполнения запроса к базе данных зависят от того, указаны в нём или нет параметры wwResolution (разрешение по времени), wwCycleCount (счётчик подынтервалов) и wwRetrievalMode (режим извлечения). В следующей таблице содержится описание получаемых результатов в зависимости от наличия того или иного параметра и его значения. N – числовое значение.

Режим извлечения	Разрешение	Счётчик подынтервалов	Результаты
CYCLIC	N	0 (или значение не присвоено)	Из базы данных извлекаются все сведения для указанного временного диапазона, после чего применяется разрешение в N мс.
CYCLIC	0 (или значение не присвоено)	0	Сервер возвращает по 100000 строк для каждого указанного тэга.
CYCLIC	0 (или значение не присвоено)	N	Из базы данных извлекаются все сведения для указанного временного диапазона, после чего возвращается N равноотстоящих друг от друга по времени строк.
CYCLIC	N	N	Ничего не возвращается.
CYCLIC	значение отсутствует	значение отсутствует	Сервер возвращает по 100 строк для каждого тэга.
DELTA	любое указанное значение игнорируется	0	Возвращаются все данные, изменившие своё значение в указанном временном интервале (всего до 100000 строк).
DELTA	любое указанное значение игнорируется	N	Из базы данных извлекаются все сведения для указанного временного диапазона, после чего используется счётчик подынтервалов (первые N строк), счётчик подынтервалов ограничивает максимальное количество возвращаемых строк, независимо от количества указанных тэгов. Например, запрос значений четырёх тэгов, в котором значение счётчика равно 20, возвратит не более 20 строк данных. В результатах будет присутствовать 4 строки с начальными значениями тэгов, а содержание остальных 16 будет определяться последующими изменениями значений этих тэгов.
DELTA	любое указанное значение игнорируется	значение отсутствует	Возвращаются все данные, изменившие своё значение в указанном временном интервале (ограничения на количество возвращаемых строк нет).

В целом, если в виртуальных столбцах встретится какая-либо ошибка или возникнет неразрешимое противоречие, никакие данные выдаваться не будут.

Указание в запросе нескольких тэгов разных типов

Тип данных столбца vValue таблиц History и Live – sql_variant, что позволяет запрашивать из него значения любых типов. Другими словами, в одном и том же запросе для него можно указывать тэги любых типов без использования конструкции JOIN.

Пример запроса:

```
SELECT TagName, DateTime, vValue
    FROM History
        WHERE TagName IN ('SysTimeMin', 'SysPulse',
        'SysString')
          AND DateTime >= '2001-12-20 0:00'
          AND DateTime <= '2001-12-20 0:05'
          AND wwRetrievalMode = 'delta'
```

Результаты:

TagName	DateTime	vValue
SysTimeMin	2001-12-20 00:00:00.000	0
SysPulse	2001-12-20 00:00:00.000	0
SysString	2001-12-20 00:00:00.000	2001/12/20 08:00:00
SysTimeMin	2001-12-20 00:01:00.000	1
SysPulse	2001-12-20 00:01:00.000	1
SysTimeMin	2001-12-20 00:02:00.000	2
SysPulse	2001-12-20 00:02:00.000	0
SysTimeMin	2001-12-20 00:03:00.000	3
SysPulse	2001-12-20 00:03:00.000	1
SysTimeMin	2001-12-20 00:04:00.000	4
SysPulse	2001-12-20 00:04:00.000	0
SysTimeMin	2001-12-20 00:05:00.000	5
SysPulse	2001-12-20 00:05:00.000	1

Указание условий для столбцов с данными вариантного типа

Провайдер InSQL OLEDB пересылает данные вариантного типа серверу SQL Server как строки символов. Если в запросе присутствует условие выборки значений столбца вариантного типа, требуемую фильтрацию выполняет SQL Server. Ниже приведен пример подобного условия:

```
WERE … vValue = 2
```

Для выполнения требуемого действия SQL Server должен определить тип константы (в данном случае "2") и преобразовать данные вариантного типа (строку символов) в данные этого типа назначения. SQL Server предполагает, что константа без десятичной точки представляет собой целую величину, пытается преобразовать строку символов в целое число. Преобразование завершится неудачно, если строка символов будет представлять собой обозначение вещественного числа (например, 2,00123).

Нужно явно указывать конечный тип данных, пользуясь функцией CONVERT(). Это единственно надёжный способ фильтрации по значениям столбца vValue, содержащего вариантные данные.

Пример запроса:

```
SELECT DateTime, Quality, OPCQuality, QualityDetail,
Value, vValue, TagName
    FROM History
        WHERE TagName IN ('ADxxxF36', 'SysTimeMin',
        'SysPulse')
          AND DateTime >= '12-04-2001 04:00:00.000'
          AND DateTime <= '12-04-2001 04:03:00.000'
          AND wwRetrievalMode = 'Delta'
          AND convert(float, vValue) = 2
```

Другой запрос:

```
SELECT DateTime, Quality, OPCQuality, QualityDetail,
Value, vValue, TagName
    FROM History
        WHERE TagName IN ('VectorX', 'SysTimeMin',
        'SysPulse')
          AND DateTime >= '20020313 04:00:07.000'
          AND DateTime <= '20020313 04:01:00.000'
          AND wwRetrievalMode = 'Delta'
          AND convert(float, vValue) > 1
          AND convert(float, vValue) < 2
```

Использование функций даты и времени

Функции даты и времени выполняют определённые действия с указанными датой и временем суток и возвращают строку символов, числовое значение или другое значение даты и времени.

Следующий запрос возвращает метки времении значения тэга SysTimeSec за предшествующие 10 минут.

```
SELECT DateTime, TagName, Value, Quality
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND DateTime >= dateadd(Minute, -10,
          GetDate())
          AND DateTime <= GetDate()
          AND wwRetrievalMode = 'Cyclic'
```

Результаты:

DateTime	TagName	Value	Quality
2001-12-15 13:00:00.000	SysTimeSec	0.0	0
2001-12-15 13:00:06.060	SysTimeSec	6.0	0
2001-12-15 13:00:12.120	SysTimeSec	12.0	0
2001-12-15 13:00:18.180	SysTimeSec	18.0	0
2001-12-15 13:00:24.240	SysTimeSec	24.0	0
2001-12-15 13:00:30.300	SysTimeSec	30.0	0
2001-12-15 13:00:36.360	SysTimeSec	36.0	0

2001-12-15 13:00:42.420 SysTimeSec 42.0 0

…

Одновременное использование в запросе и функций даты и времени, и параметра wwTimeZone обеспечивается функцией faaTZgetdata(). Её применение вызвано различиями между сервером SQL Server и провайдером InSQL OLEDB в определении конечной даты запроса.

Выполняя запрос, SQL Server осуществляет все временные расчёты по своему локальному времени, определяет конкретные моменты и пересылает запрос провайдеру InSQL OLEDB. Провайдер InSQL OLEDB изменяет полученный набор результатов с учётом часового пояса, указываемого параметром wwTimeZone.

Далее приведён пример запроса на данные за последние 30 минут, время в котором выражено по Восточному летнему времени (EDT – Eastern Daylight Time). Сервер расположен в зоне Тихоокеанского летнего времени (PDT – Pacific Daylight Time).

```
SELECT DateTime, TagName, Value
    FROM History
        WHERE TagName IN ('SysTimeHour', 'SysTimeMin',
        'SysTimeSec')
          AND DateTime > DateAdd(mi, -30, GetDate())
          AND wwTimeZone = 'eastern daylight time'
```

Если в часовом поясе PDT текущее время 14:00:00, в зоне EDT – 17:00:00. Можно ожидать, что запрос вернёт данные для периода с 16:30:00 по 17:00:00 EDT, или за последние 30 минут по восточному времени.

Однако сервер вернёт данные для периода с 13:30:00 по 17:00:00 EDT. Причина в том, что SQL Server вычисляет значение выражения "DateAdd(mi, -30, GetDate())" по локальному времени сервера (то есть по тихоокеанскому времени), а затем пересылает провайдеру InSQL OLEDB преобразованный запрос следующего вида:

```
SELECT DateTime, TagName, Value
    FROM History
        WHERE TagName IN ('SysTimeHour', 'SysTimeMin',
        'SysTimeSec')
          AND DateTime > 'YYYY-MM-DD 13:30:00.000'
          AND wwTimeZone = 'eastern daylight time'
```

Поскольку провайдеру OLEDB конечная дата не была передана, он предполагает, что она равна текущему времени в указанном часовом поясе, то есть 17:00:00 EDT.

Решить эту проблему можно с помощью вызова функции faaTZgetdate() с промежуточными аргументами, например:

```
DECLARE @starttime datetime
SET @starttime = dbo.faaTZgetdate('eastern daylight
time')
SELECT DateTime, TagName, Value
    FROM History
        WHERE TagName IN ('SysTimeHour', 'SysTimeMin',
        'SysTimeSec')
          AND DateTime > DateAdd(mi, -30, @starttime)
```

```
        AND DateTime < DateAdd(mi, -5, @starttime)
        AND wwTimeZone = 'eastern daylight time'
```

В следующем примере выполнено обращение к расширенной таблице:

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime,
SysTimeHour, SysTimeMin, SysTimeSec
    FROM WideHistory
      WHERE DateTime > DateAdd(mi, -30,
      faaTZgetdate("eastern daylight time"))
        AND DateTime < DateAdd(mi, -5,
        faaTZgetdate("eastern daylight time"))
        AND wwTimeZone = "eastern daylight time"
    ')
```

Использование конструкции GROUP BY

Конструкция GROUP BY может использоваться только в запросах в четырёхкомпонентном формате либо в запросах к соответствующим представлениям.

Следующий запрос возвратит максимальное среди всех значений тэгов в указанной группе за указанный период времени:

```
SELECT TagName, Max(Value)
    FROM INSQL.Runtime.dbo.History
      WHERE TagName IN ('ReactTemp', 'ReactLevel',
      'SysTimeSec')
        AND DateTime > '2001-12-20 0:00'
        AND DateTime < '2001-12-20 0:05'
          GROUP BY TagName
```

Результат может выглядеть следующим образом:

SysTimeSec 59.0

Использование функции COUNT()

Функция COUNT() может использоваться в запросах в четырёхкомпонентном формате и не поддерживается внутри функции OPENQUERY().

Пример запроса:

```
SELECT count(*)
    FROM History
      WHERE TagName = 'SysTimeSec'
        AND DateTime >= '2001-12-20 0:00'
        AND DateTime <= '2001-12-20 0:05'
        AND wwRetrievalMode = 'delta'
        AND Value >= 30
```

Результаты:

150

При использовании функции OPENQUERY() выполнять арифметические вычисления со значением столбца COUNT(*) нельзя, хотя это допускается вне вызова функции OPENQUERY():

```
SELECT count(*), count(*)/2 FROM OPENQUERY(INSQL,
'SELECT DateTime, vValue, Quality, QualityDetail
    FROM History
      WHERE TagName IN ("SysTimeSec")
        AND DateTime >= "2002-04-16 03:00:00.000"
        AND DateTime <= "2002-04-16 06:00:00.000"
        AND wwRetrievalMode = "Delta"
    ')
```

Результаты:

```
10801           5400
```

1 row(s) affected (найдена 1 строка)

Использование арифметических функций

В следующем запросе вычисляется сумма значений двух тэгов, извлекаемых из таблицы WideHistory:

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime,
ReactLevel, ProdLevel, "Sum" = ReactLevel+Prodlevel
    FROM WideHistory
      WHERE DateTime > "2001-02-28 18:56"
          AND DateTime < "2001-02-28 19:00"
          AND wwRetrievalMode = "Cyclic"
    ')
```

Результаты:

DateTime	ReactLevel	Prodlevel	Sum
2001-02-28 18:56:00.000	1525.0	2343.0	3868.0
2001-02-28 18:56:00.000	1525.0	2343.0	3868.0
2001-02-28 18:56:00.000	1525.0	2343.0	3868.0
2001-02-28 18:56:00.000	1525.0	2343.0	3868.0
2001-02-28 18:56:00.000	2025.0	2343.0	4368.0
2001-02-28 18:56:00.000	2025.0	2343.0	4368.0

…

100 row(s) affected (найдено 100 строк)

Перед знаком арифметической операции и после него (сложения '+', вычитания '-', умножения '*' или деления '/') нужно вставлять по одному пробелу, например записывать 'value – 2', а не 'value-2'.

Использование групповых функций

В следующем запросе вычисляются минимальное, максимальное, среднее и суммарное значения для тэга ReactLevel по данным из таблицы WideHistory:

```
SELECT * FROM OpenQuery(INSQL, 'SELECT "Minimum" =
min(ReactLevel), "Maximum" = max(ReactLevel),
"Average" = avg(ReactLevel), "Sum" = sum(ReactLevel)
    FROM WideHistory
      WHERE DateTime > "2001-02-28 18:55:00"
          AND DateTime < "2001-02-28 19:00:00"
          AND wwRetrievalMode = "Cyclic"
    ')
```

Результаты:

Minimum	Maximum	Average	Sum
-25.0	2025.0	1181.2	118120.0

1 row(s) affected (найдена 1 строка)

При использовании функций SUM или AVG в запросах к таблице Wide в режиме "по изменению" указанное действие будет выполняться только тогда, когда значение изменится. Функция группирования со всеми значениями, возвращаемыми для каждого столбца, не выполняется.

Рассмотрим запрос без группирования данных:

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime,
SysTimeHour, SysTimeMin, SysTimeSec, SysDateDay
    FROM AnalogWideHistory
      WHERE DateTime >= "2001-08-15 13:20:57.345"
          AND DateTime < "2001-08-15 13:21:03.345"
          AND wwRetrievalMode = "Delta"
    ')
GO
```

Результаты:

DateTime	SysTimeHour	SysTimeMin	SysTimeSec	SysDateDay
2001-08-15 13:20:57.343	13.0	20.0	57.0	15.0
2001-08-15 13:20:58.000	13.0	20.0	58.0	15.0
2001-08-15 13:20:59.000	13.0	20.0	59.0	15.0
2001-08-15 13:21:00.000	13.0	21.0	0.0	15.0
2001-08-15 13:21:01.000	13.0	21.0	1.0	15.0
2001-08-15 13:21:02.000	13.0	21.0	2.0	15.0
2001-08-15 13:21:03.000	13.0	21.0	3.0	15.0

В следующем запросе выполняется суммирование всех возвращённых значений:

```
SELECT * FROM OpenQuery(INSQL, 'SELECT
Sum(SysTimeHour), Sum(SysTimeMin), Sum(SysTimeSec),
Sum(SysDateDay)
    FROM WideHistory
      WHERE DateTime >= "2001-08-15 13:20:57.345"
          AND DateTime < "2001-08-15 13:21:03.345"
          AND wwRetrievalMode = "Delta"
```

```
    ')
GO
```

Результаты будут выглядеть следующим образом:

SysTimeHour	SysTimeMin	SysTimeSec	SysDateDay
13.0	41.0	180.0	15.0

Таким образом, в режиме извлечения по изменению в операциях суммирования и усреднения будут использоваться только те величины, которые изменились по сравнению с предыдущей строкой.

Среднее арифметическое в режиме извлечения по изменению будет рассчитываться по такой формуле:

(Сумма изменившихся значений) / (количество изменившихся значений)

В следующем запросе вычисляется среднее арифметическое всех возвращённых значений:

```
SELECT * FROM OpenQuery(INSQL, 'SELECT
Avg(SysTimeHour), Avg(SysTimeMin), Avg(SysTimeSec),
Avg(SysDateDay)
    FROM WideHistory
      WHERE DateTime >= "2001-08-15 13:20:57.345"
          AND DateTime < "2001-08-15 13:21:03.345"
          AND wwRetrievalMode = "Delta"
    ')
GO
```

Результаты:

SysTimeHour	SysTimeMin	SysTimeSec	SysDateDay
13.0	20.5	25.714285714285715	15.0

Создание и извлечение аннотаций

Следующий запрос создаёт аннотацию для указанного тэга. Аннотация заносится в базу данных в момент отключения насоса. Затем выполняется извлечение всех аннотаций.

```
DECLARE @@UserKey INT


SELECT @@UserKey = UserKey
    FROM UserDetail
      WHERE UserName = 'wwAdmin'


INSERT INTO Annotation (TagName, UserKey, DateTime,
Content)
    VALUES ('ReactLevel', @@UserKey, GetDate(), 'The
    Pump is off')


SELECT DateTime, TagName, Content
    FROM Annotation
      WHERE Annotation.TagName = 'ReactLevel'
```

```
        AND DateTime > '27 Feb 01'
        AND DateTime <= GetDate()
```

Результаты:

DateTime	TagName	Content
2001-02-28 19:18:00.000	ReactLevel	The Pump is off (Насос выключен)

1 row(s) affected (найдена 1 строка)

Операторы сравнения в запросах в режиме "по изменению"

Запросы, в которых начальные и конечные даты указаны в условиях сравнения с операторами ">=", ">", "<", "<=", выполняются по-разному. Эти операторы сравнения могут использоваться в запросах к таблицам History и WideHistory. Сравнение выполняется всегда независимо от способа записи запроса (в четырёхкомпонентном формате, в виде представления и т.д.)

Запросы с операторами сравнения возвращают все допустимые значения для указанного набора тэгов, изменение которых произошло в указанный период времени. Уточнить набор возвращаемых результатов можно с помощью мёртвых зон и других фильтров.

Указание начальной даты с оператором ">="

Если начальная дата определена с использованием оператора ">=" (больше чем или равно), в результатах всегда будет присутствовать строка с указанной датой. Если изменение данных произошло точно в указанную дату и время суток, возвращаемое значение качества – 0, в противном случае будет возвращено значение 133 (поскольку длительность времени, в течение которого тэг находился в состоянии X, неизвестна).

Пример 1

В следующем примере момент начала временного диапазона запроса не совпадает с моментом изменения значений тэга:

```
SELECT DateTime, Value, Quality
    FROM History
      WHERE TagName = 'SysTimeMin'
          AND wwRetrievalMode = 'Delta'
          AND DateTime >= '2001-01-13 12:00:30'
          AND DateTime < '2001-01-13 12:10:00'
```

Поскольку данные в момент начала интересуемого периода (12:00:30) не изменялись, они будут иметь качество с кодом 133:

DateTime	Value	Quality
2001-01-13 12:00:30.000	0.0	133
2001-01-13 12:01:00.000	1.0	0
2001-01-13 12:02:00.000	2.0	0
2001-01-13 12:03:00.000	3.0	0
2001-01-13 12:04:00.000	4.0	0
2001-01-13 12:05:00.000	5.0	0

2001-01-13 12:06:00.000	6.0	0
2001-01-13 12:07:00.000	7.0	0
2001-01-13 12:08:00.000	8.0	0
2001-01-13 12:09:00.000	9.0	0

10 row(s) affected (найдено 10 строк)

Пример 2

В данном примере момент начала временного диапазона совпадает с моментом изменения данных.

```
SELECT DateTime, Value, Quality
    FROM History
      WHERE TagName = 'SysTimeMin'
          AND wwRetrievalMode = 'Delta'
          AND DateTime >= '2001-01-13 12:01:00'
          AND DateTime < '2001-01-13 12:10:00'
```

Поскольку момент изменения данных и начало временного диапазона запроса совпадают, качество возвращается с кодом 0:

DateTime	Value	Quality
2001-01-13 12:01:00.000	1.0	0
2001-01-13 12:02:00.000	2.0	0
2001-01-13 12:03:00.000	3.0	0
2001-01-13 12:04:00.000	4.0	0
2001-01-13 12:05:00.000	5.0	0
2001-01-13 12:06:00.000	6.0	0
2001-01-13 12:07:00.000	7.0	0
2001-01-13 12:08:00.000	8.0	0
2001-01-13 12:09:00.000	9.0	0

9 row(s) affected (найдено 9 строк)

Пример 3

В данном случае будет возвращено не менее одной строки (для начальной даты), даже если в указанный период не происходило никаких изменений в данных.

```
SELECT DateTime, Value, Quality
    FROM History
      WHERE TagName = 'SysTimeMin'
          AND wwRetrievalMode = 'Delta'
          AND DateTime >= '2001-01-13 12:00:30'
          AND DateTime < '2001-01-13 12:01:00'
```

Поскольку в указанный период времени данные не изменялись, возвращаемый код качества данных, соответствующих моменту начала временного диапазона, равен 133.

DateTime	Value	Quality
2001-01-13 12:00:30.000	0.0	133

1 row(s) affected (найдена 1 строка)

Указание начальной даты с оператором ">"

Если в условии сравнения указан оператор ">" (больше чем), в первой строке результатов будут выведены данные, первый раз изменившие своё значение только после указанного момента. Строка с начальным значением не выводится. Запросы с оператором ">" могут не возвращать никаких строк.

Пример 1

При выполнении следующего запроса в первой строке результатов будут выведены данные, изменившиеся после начала временного диапазона запроса (то есть после 12:00:30).

```
SELECT DateTime, Value, Quality
    FROM History
      WHERE TagName = 'SysTimeMin'
          AND wwRetrievalMode = 'Delta'
          AND DateTime > '2001-01-13 12:00:30'
          AND DateTime < '2001-01-13 12:10:00'
```

В первой строке будут оказаны данные, изменение которых произошло после 12:00:30.

DateTime	Value	Quality
2001-01-13 12:01:00.000	1.0	0
2001-01-13 12:02:00.000	2.0	0
2001-01-13 12:03:00.000	3.0	0
2001-01-13 12:04:00.000	4.0	0
2001-01-13 12:05:00.000	5.0	0
2001-01-13 12:06:00.000	6.0	0
2001-01-13 12:07:00.000	7.0	0
2001-01-13 12:08:00.000	8.0	0
2001-01-13 12:09:00.000	9.0	0

9 row(s) affected (найдено 9 строк)

Пример 2

В этом примере момент начала временного диапазона (12:01:00) совпадает с моментом изменения значений тэга, однако эти сведения не войдут в набор результатов, поскольку в запросе указан оператор ">", а не ">=".

```
SELECT DateTime, Value, Quality
    FROM History
      WHERE TagName = 'SysTimeMin'
          AND wwRetrievalMode = 'Delta'
          AND DateTime > '2001-01-13 12:01:00'
          AND DateTime < '2001-01-13 12:10:00:
```

DateTime	Value	Quality
2001-01-13 12:02:00.000	2.0	0

2001-01-13 12:03:00.000	3.0	0
2001-01-13 12:04:00.000	4.0	0
2001-01-13 12:05:00.000	5.0	0
2001-01-13 12:06:00.000	6.0	0
2001-01-13 12:07:00.000	7.0	0
2001-01-13 12:08:00.000	8.0	0
2001-01-13 12:09:00.000	9.0	0

8 row(s) affected (найдено 8 строк)

Пример 3

Следующий запрос не возвращает никаких результатов, поскольку в указанный период времени никаких изменений в данных не происходило.

```
SELECT DateTime, Value, Quality
    FROM History
      WHERE TagName = 'SysTimeMin'
          AND wwRetrievalMode = 'Delta'
          AND DateTime > '2001-01-13 12:00:30'
          AND DateTime < '2001-01-13 12:01:00'
```

DateTime	Value	Quality

1 row(s) affected (найдена 1 строка)

Указание конечной даты с оператором "<="

Если в условии будет указан оператор сравнения "<=" (меньше чем или равно), в последней строке результатов будут выведены данные, изменение которых произошло или ранее указанной даты, или точно в этот момент.

В следующем запросе выполняется обращение к представлению удалённой таблицы.

```
SELECT DateTime, Value, Quality
    FROM History
      WHERE TagName = 'SysTimeMin'
          AND wwRetrievalMode = 'Delta'
          AND DateTime > '2001-01-13 12:00:30'
          AND DateTime <= '2001-01-13 12:10:00'
```

Следует отметить, что последнее изменение значения тэга произошло точно в момент конечной даты запроса.

DateTime	Value	Quality
2001-01-13 12:01:00.000	1.0	0
2001-01-13 12:02:00.000	2.0	0
2001-01-13 12:03:00.000	3.0	0
2001-01-13 12:04:00.000	4.0	0
2001-01-13 12:05:00.000	5.0	0
2001-01-13 12:06:00.000	6.0	0
2001-01-13 12:07:00.000	7.0	0

2001-01-13 12:08:00.000	8.0	0
2001-01-13 12:09:00.000	9.0	0
2001-01-13 12:10:00.000	10.0	0

10 row(s) affected (найдено 10 строк)

Указание конечной даты с оператором "<"

Если в условии будет указан оператор сравнения "<" (меньше чем), в последней строке результатов будут выведены данные, изменение которых произошло только ранее указанной даты, но не в этот момент. Если изменения произошли именно в этот момент, они в набор возвращаемых результатов не войдут.

В следующем запросе выполняется обращение к представлению удалённой таблицы.

```
SELECT DateTime, Value, Quality
    FROM History
      WHERE TagName = 'SysTimeMin'
          AND wwRetrievalMode = 'Delta'
          AND DateTime > '2001-01-13 12:00:30'
          AND DateTime < '2001-01-13 12:10:00'
```

Следует отметить, что, хотя последнее изменение значения тэга и произошло точно в момент конечной даты запроса, оно в наборе возвращаемых результатов отсутствует.

DateTime	Value	Quality
2001-01-13 12:01:00.000	1.0	0
2001-01-13 12:02:00.000	2.0	0
2001-01-13 12:03:00.000	3.0	0
2001-01-13 12:04:00.000	4.0	0
2001-01-13 12:05:00.000	5.0	0
2001-01-13 12:06:00.000	6.0	0
2001-01-13 12:07:00.000	7.0	0
2001-01-13 12:08:00.000	8.0	0
2001-01-13 12:09:00.000	9.0	0

9 row(s) affected (найдено 9 строк)

Операторы сравнения в циклических запросах данных со счётчиками подынтервалов

Запросы в режиме циклического извлечения, в которых используется параметр wwCycleCount, возвращают наборы строк, метки времени в которых равномерно распределены по указанному временному диапазону. Количество возвращаемых строк всегда равно значению счётчика для каждого тэга в запросе. Разрешение возвращаемых данных по времени равно результату деления длительности временного интервала на значение счётчика.

Два оператора равенства

Если временной диапазон задаётся с использованием операторов ">=" и "<=", в первой строке всегда будут выводиться данные на указанную начальную дату, а в последней – на указанную конечную дату. В этом случае разрешение по времени вычисляется по формуле:

(конечная_дата – начальная_дата) / (счётчик_подынтервалов – 1)

Значение счётчика в следующем запросе равно 60, что соответствует односекундному разрешению возвращаемых данных по времени. Обращение выполняется к представлению удалённой таблицы.

```
SELECT DateTime, Value
    FROM History
       WHERE TagName = 'SysTimeSec'
           AND DateTime >= '2001-01-13 12:00:00'
           AND DateTime <= '2001-01-13 12:00:59'
           AND wwCycleCount = 60
           AND wwRetrievalMode = 'Cyclic'
```

Результаты:

DateTime	Value
2001-01-13 12:00:00.000	0.0
2001-01-13 12:00:01.000	1.0
2001-01-13 12:00:02.000	2.0
2001-01-13 12:00:03.000	3.0
2001-01-13 12:00:04.000	4.0
…	
2001-01-13 12:00:56.000	56.0
2001-01-13 12:00:57.000	57.0
2001-01-13 12:00:58.000	58.0
2001-01-13 12:00:59.000	59.0

60 row(w) affected (найдено 60 строк)

Один оператор равенства

Если какой-либо из концов временного диапазона исключается (путём использования оператора ">" либо "<" вместо ">=" либо "<=" соответственно), в начале или в конце набора результатов получается "пропуск" подынтервала.

Разрешение по времени рассчитывается по формуле:

(конечная_дата – начальная_дата) \ (счётчик_подынтервалов)

Строка с меткой времени, определяемой оператором ">" или "<", в набор возвращаемых результатов не входит.

В следующих запросах выполняется обращение к представлению удалённой таблицы.

Пример 1

Значение счётчика в следующем запросе равно 60, что соответствует односекундному разрешению возвращаемых данных по времени. Из временного диапазона исключён его конец.

```
SELECT DateTime, Value
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND wwCycleCount = 60
          AND DateTime >= '2001-01-13 12:00:00'
          AND DateTime < '2001-01-13 12:01:00'
```

Результаты:

DateTime	Value
2001-01-13 12:00:00.000	0.0
2001-01-13 12:00:01.000	1.0
2001-01-13 12:00:02.000	2.0
2001-01-13 12:00:03.000	3.0
2001-01-13 12:00:04.000	4.0
…	
2001-01-13 12:00:56.000	56.0
2001-01-13 12:00:57.000	57.0
2001-01-13 12:00:58.000	58.0
2001-01-13 12:00:59.000	59.0

60 row(w) affected (найдено 60 строк)

Пример 2

Значение счётчика в следующем запросе равно 60, что соответствует односекундному разрешению возвращаемых данных по времени. Из временного диапазона исключено его начало.

```
SELECT DateTime, Value
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND wwCycleCount = 60
          AND DateTime > '2001-01-13 12:00:00'
          AND DateTime <= '2001-01-13 12:01:00'
```

Результаты:

DateTime	Value
2001-01-13 12:00:01.000	1.0
2001-01-13 12:00:02.000	2.0
2001-01-13 12:00:03.000	3.0
2001-01-13 12:00:04.000	4.0
…	
2001-01-13 12:00:56.000	56.0
2001-01-13 12:00:57.000	57.0

2001-01-13 12:00:58.000	58.0
2001-01-13 12:00:59.000	59.0
2001-01-13 12:01:00.000	0.0

60 row(w) affected (найдено 60 строк)

Отсутствие операторов равенства

Если оба конца временного диапазона исключены (путём указания операторов ">" и "<"), "пропуски" образуются как в начале, так и в конце набора возвращаемых результатов.

Разрешение по времени вычисляется по формуле:

(конечная_дата – начальная_дата) / (счётчик_подынтервалов + 1).

Строки с метками времени, совпадающими с началом и концом временного диапазона запроса, в наборе возвращаемых результатов отсутствуют.

В следующем запросе осуществляется обращение к представлению удалённой таблицы.

```
SELECT DateTime, Value
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND wwCycleCount = 60
          AND DateTime > '2001-01-13 12:00:00'
          AND DateTime < '2001-01-13 12:01:01'
```

Результаты:

DateTime	Value
2001-01-13 12:00:01.000	1.0
2001-01-13 12:00:02.000	2.0
2001-01-13 12:00:03.000	3.0
2001-01-13 12:00:04.000	4.0
…	
2001-01-13 12:00:56.000	56.0
2001-01-13 12:00:57.000	57.0
2001-01-13 12:00:58.000	58.0
2001-01-13 12:00:59.000	59.0
2001-01-13 12:01:00.000	0.0

60 row(w) affected (найдено 60 строк)

Операторы сравнения в циклических запросах с указанием разрешения по времени

Запросы в режиме циклического извлечения, в которых указано разрешение по времени, возвращают наборы строк, метки времени в которых равномерно распределены по указанному временному диапазону.

Два оператора равенства

Если временной диапазон указан с использованием операторов ">=" и "<=", в первой строке будет выведена метка начала диапазона. Метка конца диапазона будет выведена в последней строке, если длительность временного диапазона делится на значение разрешения нацело. Если же она делится не нацело, метка времени последней строки будет иметь наиболее позднюю дату, вычисляемую по формуле (начальный_момент + N * разрешение_по_времени), которая ещё предшествует указанному концу временного диапазона запроса.

В целом:

- "<= конечный_момент" МОЖЕТ возвращать в последней строке данные с меткой времени, равной моменту конца временного диапазона запроса (однако это не гарантируется).
- "< конечный_момент" гарантированно НЕ возвращает в последней строке данные с меткой времени, равной моменту конца временного диапазона запроса.

В следующем запросе разрешение по времени равно 1 секунде.

```
SELECT DateTime, Value
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND wwResolution = 1000
          AND DateTime >= '2001-01-13 12:00:00'
          AND DateTime <= '2001-01-13 12:01:00'
```

Результаты:

DateTime	Value
2001-01-13 12:00:00.000	0.0
2001-01-13 12:00:01.000	1.0
2001-01-13 12:00:02.000	2.0
2001-01-13 12:00:03.000	3.0
2001-01-13 12:00:04.000	4.0
…	
2001-01-13 12:00:56.000	56.0
2001-01-13 12:00:57.000	57.0
2001-01-13 12:00:58.000	58.0
2001-01-13 12:00:59.000	59.0
2001-01-13 12:01:00.000	0.0

61 row(s) affected (найдена 61 строка)

Один оператор равенства

Если начало временного диапазона запроса исключается из рассмотрения (использованием оператора ">" вместо ">="), в начале набора результатов появляется "пропуск". В данном случае в первой строке будут выведены данные с меткой времени, равной (начальная_дата + разрешение_по_времени). При исключении конца временного диапазона в последней строке будут выведены данные с меткой времени, вычисленной

по формуле (начальный_момент + N * разрешение_по_времени), которая ещё предшествует указанному концу временного диапазона запроса.

Данные с меткой времени, указанной в строке условия с оператором ">" (или "<"), не выводятся.

Пример 1

В запросе указано разрешение по времени в 1000 мс (1 секунду). Конечный момент исключён оператором "<".

```
SELECT DateTime, Value
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND wwResolution = 1000
          AND DateTime >= '2001-01-13 12:00:00'
          AND DateTime < '2001-01-13 12:01:00'
```

Результаты:

DateTime	Value
2001-01-13 12:00:00.000	0.0
2001-01-13 12:00:01.000	1.0
2001-01-13 12:00:02.000	2.0
2001-01-13 12:00:03.000	3.0
2001-01-13 12:00:04.000	4.0
…	
2001-01-13 12:00:56.000	56.0
2001-01-13 12:00:57.000	57.0
2001-01-13 12:00:58.000	58.0
2001-01-13 12:00:59.000	59.0

60 row(s) affected (найдено 60 строк)

Пример 2

В этом запросе также указано разрешение по времени в 1000 мс (1 с). Оператором ">" исключён начальный момент.

```
SELECT DateTime, Value
    FROM History
      WHERE TagName = 'SysTimeSec'
          AND wwResolution = 1000
          AND DateTime > '2001-01-13 12:00:00'
          AND DateTime <= '2001-01-13 12:01:00'
```

Результаты:

DateTime	Value
2001-01-13 12:00:01.000	1.0
2001-01-13 12:00:02.000	2.0
2001-01-13 12:00:03.000	3.0
2001-01-13 12:00:04.000	4.0

…

2001-01-13 12:00:56.000	56.0
2001-01-13 12:00:57.000	57.0
2001-01-13 12:00:58.000	58.0
2001-01-13 12:00:59.000	59.0
2001-01-13 12:01:00.000	0.0

60 row(s) affected (найдено 60 строк)

Отсутствие операторов равенства

Если оба конца временного диапазона исключены (путём указания операторов ">" и "<"), "пропуски" образуются как в начале, так и в конце набора возвращаемых результатов.

Строки с метками времени, совпадающими с началом и концом временного диапазона запроса, в наборе возвращаемых результатов отсутствуют.

В запросе указано разрешение по времени в 1000 мс (1 с).

```
SELECT DateTime, Value
    FROM v_AnalogHistory
      WHERE TagName = 'SysTimeSec'
          AND wwResolution = 1000
          AND DateTime > '2001-01-13 12:00:00'
          AND DateTime < '2001-01-13 12:01:01'
```

Результаты:

DateTime	Value
2001-01-13 12:00:01.000	1.0
2001-01-13 12:00:02.000	2.0
2001-01-13 12:00:03.000	3.0
2001-01-13 12:00:04.000	4.0

…

2001-01-13 12:00:56.000	56.0
2001-01-13 12:00:57.000	57.0
2001-01-13 12:00:58.000	58.0
2001-01-13 12:00:59.000	59.0
2001-01-13 12:01:00.000	0.0

60 row(s) affected (найдено 60 строк)

Запросы на данные из архивных таблиц с помощью оператора SELECT INTO

Первый запрос вставляет указанные данные из таблицы WideHistory в таблицу MyTable, второй извлекает из неё все данные. В первом запросе используется функция OPENQUERY().

```
DROP TABLE MyTable
```

```
SELECT DateTime,
      "Sec" = datepart(ss, DateTime),
      "mS" = datepart(ms, DateTime),
      ReactTemp, ReactLevel
  INTO MyTable
      FROM OpenQuery(INSQL, 'SELECT DateTime,
          ReactTemp, ReactLevel FROM WideHistory
          WHERE wwResolution = 5000
          AND DateTime >= "2001-03-13 1:58pm"
          AND DateTime <= "2001-03-13 2:00pm" ')

SELECT * FROM MyTable
```

Результаты:

DateTime	Sec	mS	ReactTemp	ReactLevel
2001-03-13 13:58:00.000	0	0	190.9	2025.0
2001-03-13 13:58:00.000	5	0	190.9	2025.0
2001-03-13 13:58:00.000	10	0	168.3	1215.0
2001-03-13 13:58:00.000	15	0	168.3	1215.0
2001-03-13 13:58:00.000	20	0	133.8	315.0
2001-03-13 13:58:00.000	25	0	133.8	315.0
2001-03-13 13:58:00.000	30	0	101.6	0.0
2001-03-13 13:58:00.000	35	0	101.6	0.0
2001-03-13 13:58:00.000	40	0	32.4	750.0
2001-03-13 13:58:00.000	45	0	32.4	750.0
2001-03-13 13:58:00.000	50	0	20.9	1700.0
2001-03-13 13:58:00.000	55	0	20.9	1700.0
2001-03-13 13:59:00.000	0	0	85.9	2000.0
2001-03-13 13:59:00.000	5	0	85.9	2000.0
2001-03-13 13:59:00.000	10	0	185.9	2000.0
2001-03-13 13:59:00.000	15	0	185.9	2000.0
2001-03-13 13:59:00.000	20	0	168.3	1235.0
2001-03-13 13:59:00.000	25	0	168.3	1235.0
2001-03-13 13:59:00.000	30	0	136.1	335.0
2001-03-13 13:59:00.000	35	0	136.1	335.0
2001-03-13 13:59:00.000	40	0	103.9	-25.0
2001-03-13 13:59:00.000	45	0	103.9	-25.0
2001-03-13 13:59:00.000	50	0	34.7	625.0
2001-03-13 13:59:00.000	55	0	34.7	625.0
2001-03-13 14:00:00.000	0	0	20.9	1575.0

25 row(s) affected (найдено 25 строк)

Перемещение данных из таблицы SQL Server в таблицу расширения

Следующие запросы показывает, как можно записывать данные в обычные таблицы SQL Server и перемещать их в таблицу расширения History.

Сначала нужно внести данные в таблицу SQL Server. Следующий запрос записывает накопленные значения тэга SysTimeSec для двухминутного интервала в таблицу ManualAnalogHistory.

```
INSERT INTO ManualAnalogHistory (DateTime, TagName,
    Value, Quality, QualityDetail, wwTagKey)
    SELECT DateTime, TagName, Value, Quality,
      QualityDetail, wwTagKey
      FROM History WHERE TagName = 'SysTimeSec'
          AND DateTime >= '20050329 12:00:00'
          AND DateTime <= '20050329 12:02:00'
```

Затем с помощью системной консоли управления System Management Console нужно создать тэг, метод накопления значений которого должен быть "MDAS/Manual Acquisition" (вручную), и зафиксировать изменения в системе. В данном примере был создан тэг с названием MDAS1.

И, наконец, следующий запрос выполняет перенос значений из таблицы ManualAnalogHistory в таблицу History.

```
INSERT INTO History (TagName, DateTime, Value,
    QualityDetail)
    SELECT 'MDAS1', DateTime, Value, QualityDetail
      FROM ManualAnalogHistory
          WHERE TagName = 'SysTimeSec'
            AND DateTime >= '20050329 12:00:00'
            AND DateTime <= '20050329 12:02:00'
```

Использование курсоров сервера

Курсоры представляют собой одну из очень мощных возможностей сервера SQL Server, допускающих контролируемое перемещение данных посредством наборов записей, которые создаются на основе результатов выполнения запроса.

Более подробные сведения о курсорах можно найти в документации по Microsoft SQL Server.

Провайдер InSQL OLEDB позволяет использовать курсоры сервера, например для объединений, которые другим способом выполнить невозможно (даты и времена из каких-либо источников с датами и временами из архивных таблиц).

Следующий запрос представляет собой пример использования курсоров сервера, выполняя такие операции, как:

- Извлечение информации обо всех событиях из таблицы EventHistory.
- Отображение "мгновенных" значений трёх тэгов, которые они имели в момент возникновения каждого события.
- Вывод информации о событийном тэге и соответствующее ключевое значение.

Данный запрос без труда может быть использован внутри хранимой процедуры.

```
SET QUOTED_IDENTIFIER OFF
DECLARE @DateValue DateTime
DECLARE @EventTag nvarchar(256)
DECLARE @EventKey int
DECLARE @Qry1 nvarchar(500)
DECLARE @Qry2 nvarchar(500)
DECLARE @Qry3 nvarchar(500)

SELECT @Qry1 = N'SELECT EventTag = @EventTag,
    EventKey = @EventKey, DateTime, TagName, Value,
    Quality FROM History
        WHERE TagName IN (N''SysTimeSec'',
        N''SysTimeMin'', N''SysTimeHour'')
           AND DateTime = '''

SELECT @Qry2 = N''''
SELECT @Qry3 = N''

DECLARE Hist_Cursor CURSOR FOR

SELECT DateTime, TagName, EventLogKey
    FROM Runtime.dbo.EventHistory
OPEN Hist_Cursor

FETCH NEXT FROM Hist_Cursor INTO @DateValue,
    @EventTag, @EventKey

WHILE @@FETCH_STATUS = 0
BEGIN
    SELECT @Qry3 = @Qry1 + convert(nvarchar,
    @DateValue, 121) + @Qry2
    --PRINT @Qry3
    EXEC sp_executesql @Qry3, N'@EventTag
    nvarchar(256), @EventKey int', @EventTag,
    @EventKey
    FETCH NEXT FROM Hist_Cursor INTO @DateValue,
      @EventTag, @EventKey
END

CLOSE Hist_Cursor
```

```
DEALLOCATE Hist_Cursor
```

Результаты выполнения:

EventTag	EventKey	DateTime	TagName	Value	Quality
SysStatusEvent	3	2001-01-12 13:00:27.000	SysTimeSec	27.0	0
SysStatusEvent	3	2001-01-12 13:00:27.000	SysTimeMin	0.0	0
SysStatusEvent	3	2001-01-12 13:00:27.000	SysTimeHour	13.0	0

3 row(s) affected (найдено 3 строки)

EventTag	EventKey	DateTime	TagName	Value	Quality
SysStatusEvent	4	2001-01-12 14:00:28.000	SysTimeSec	28.0	0
SysStatusEvent	4	2001-01-12 14:00:28.000	SysTimeMin	0.0	0
SysStatusEvent	4	2001-01-12 14:00:28.000	SysTimeHour	14.0	0

3 row(s) affected (найдено 3 строки)

Использование хранимых процедур в запросах OLE DB

Таблицы провайдера InSQL OLEDB могут использоваться в любой обычной хранимой процедуре сервера Microsoft SQL Server. Доступ к архивным данных IndustrialSQL Server в процедурах может осуществляться любыми операторами, записанными в допустимом синтаксисе Transact-SQL.

Другими словами, в хранимых процедурах допускается применение названия в четырёхкомпонентном формате, функции OPENQUERY() и OPENROWSET(), курсоры, параметризованные запросы и представления. Хранимые процедуры могут использоваться для реализации сложных объединений и прочих операций, неоднократно выполняемых различными приложениями и конечными пользователями.

Запросы на данные с миллисекундными разрешениями по времени

Внутреннее представление даты и времени в архиваторе IndustrialSQL Server выполняется в формате Win32 FILETIME. Данные в этом формате представляют собой 64-разрядные целые числа, благодаря чему разрешение данных по времени равно 100 наносекунд. Ограничивающим фактором при извлечении на самом деле является разрешающая способность типа datetime, обеспечиваемая сервером Microsoft SQL Server. Показания времени в SQL Server имеют точность в 1/300 секунды, что соответствует разрешающей способности по времени в 3,33 мс. SQL Server округляет показания времени с точностью до .000, .003 и .007 с.

Для совместимости с сервером SQL Server очень важно, чтобы все временные функции OLE DB представляли данные аналогичным образом. Для достижения требуемых результатов алгоритм округления уже встроен в эти функции. Об этом следует помнить, создавая запросы с миллисекундами в вызове функции OPENQUERY.

Например, запрос вида

```
SELECT * FROM OpenQuery(INSQL,
    'SELECT DATEPART(millisecond, getdate()),
      Value FROM Live
```

```
        WHERE TagName = "ReactTemp" ')
```

вернёт результаты в том же формате, что и запрос вида

```
SELECT DATEPART(millisecond, getdate())
```

Архиватор IndustrialSQL Server позволяет обходить это ограничение сервера Microsoft SQL Server, возвращая показания времени в виде строки символов. Подобное преобразование реализуется функцией CONVERT при указании особых спецификатором стиля: 909, 913, 914, 921 и 926, при этом строка символов возвращается в том же формате, что и при указании соответствующих спецификаторов сотой серии, но с миллисекундной точностью. Например, формат строки со спецификатором 909 совпадает с форматом строки со стандартным спецификатором 109, формат для спецификатора 913 совпадает с форматом для спецификатора 113 и т.д.

Подробно о форматах, определяемых стандартными спецификаторами стиля, см. в описании функции CONVERT(тип_данных, выражение, стиль) в документации по Microsoft SQL Server.

Правильность передачи вызова функции CONVERT() блоку грамматического разбора архиватора IndustrialSQL Server обеспечивается при использовании функции OPENQUERY, поскольку указанные спецификаторы являются специфическим расширением функции CONVERT, реализуемым архиватором.

Например, запрос вида

```
SELECT CONVERT(varchar(30), DateTime, 113), Value
    FROM Live
      WHERE TagName = 'ReactTemp'
```

вернёт значение типа "2001-08-10 14:30:45:337" (точность 3,33 мс), в то время как запрос

```
SELECT * FROM OpenQuery(INSQL,
    'SELECT CONVERT(varchar(30), DateTime, 913), Value
      FROM Live
        WHERE TagName = "ReactTemp" ')
```

вернёт аналогичное значение в виде "2001-08-10 14:30:45:335" (точность 1 мс).

Всего архиватор IndustrialSQL Server поддерживает десять спецификаторов стиля: 20, 120, 21, 121, 126, 909, 913, 914, 921 и 926. Первые пять обеспечивают совместимость с форматами сервера SQL Server 2000, спецификаторы 900-й серии дают миллисекундное разрешение по времени.

Использование переменных в запросах к расширенным таблицам

Явное указание переменных в аргументах функции OPENQUERY не допускается. Чтобы использовать переменные в запросах к расширенным таблицам, нужно сначала сформировать оператор OPENQUERY как строку символов, а затем передать эту строку на исполнение. Например:

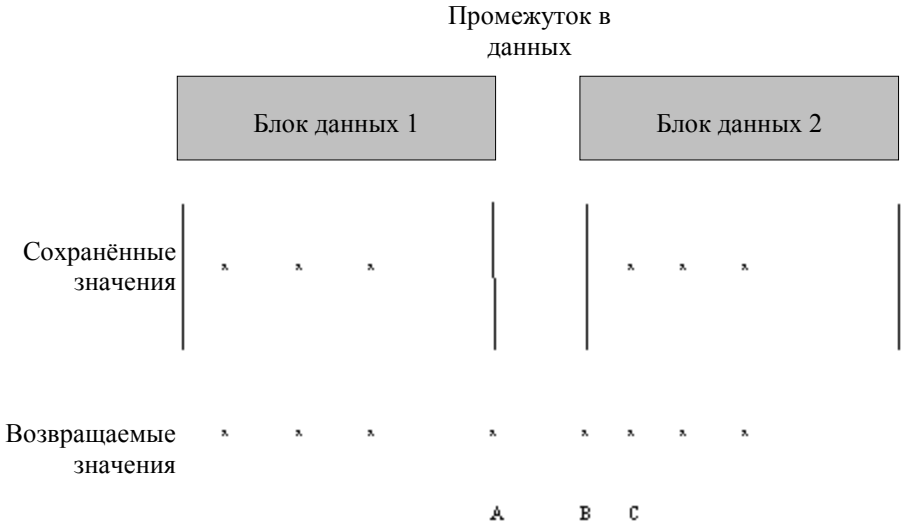
```
DECLARE @sql nvarchar(1000)
DECLARE @DateStart datetime
DECLARE @DateEnd datetime

SET @DateStart = '2001-8-29 11:00:00'
SET @DateEnd = '2001-8-29 11:11:00'
```

```
SET @sql = N'SELECT * FROM OPENQUERY(INSQL, "'SELECT
DateTime, ReactLevel, ReactTemp, ProdLevel,
BatchNumber, ConcPump, Mixer, TransferValve,
TransferPump, WaterValve, ConcValve, OutputValve,
SteamValve

FROM WideHistory

    WHERE DateTime >= " ' + CONVERT(varchar(26),

      @DateStart, 113) + ' "

      and DateTime <= " ' + CONVERT(varchar(26),

          @DateEnd, 113) + ' "

      AND wwResolution = 1000

      AND wwRetrievalMode = "cyclic" ' ') '


EXEC sp_executesql @sql
```

Извлечение данных при наличии промежутков между блоками

Если требуемые данные находятся в нескольких архивных блоках, причём конечный момент одного блока совпадает с начальным моментом следующего (с точностью до одного тика), их извлечение ничем не отличается от извлечения данных из одного блока.

Однако если система останавливалась и запускалась повторно, между временными интервалами архивных блоков будут промежутки, как показано на следующем рисунке:



При извлечении в возвращаемые данные будут добавлены точки (отмеченные на рисунке буквами A и B), которые будут соответствовать концу первого блока и началу следующего. Точка C представляет собой сохранённое значение, сгенерированное подсистемой хранения. (При перезапуске первое значение от каждого IDAS-источника будет смещено по времени от начального момента на 2 секунды и будет иметь показатель качества, равный 252.)

В следующих разделах эта ситуация рассматривается более подробно.

В режиме извлечения по изменению данные первого блока возвращаются в том виде, в каком они были сохранены. При достижении конца блока в

выходной поток вставляется точка A, отмечающая конец данных. Эта точка будет иметь следующие характеристики:

Атрибут выборки A	Значение (16-ричное)	Значение (десятичное)
Value	0	0
Quality	100	256
Quality Detail	0	0

Если в начале следующего архивного блока данных нет, подсистема извлечения вставит в поток данных точку B, соответствующую начальным данным и имеющую следующие характеристики:

Атрибут выборки B	Значение (16-ричное)	Значение (десятичное)
Value	Моментальное начальное значение	Моментальное начальное значение
Quality	0	0
Quality Detail	96	150

При циклическом извлечении каждому моменту времени должно соответствовать определённое значение. Если этот момент попадает в промежуток между блоками, система возвращает для него значение NULL. Вставленные точки будут иметь следующие характеристики:

Атрибут выборки со значением NULL	Значение (16-ричное)	Значение (десятичное)
Value	0	0
Quality	100	256
Quality Detail	0	0

Если для режима извлечения по изменению будут определены дополнительные мёртвые зоны по времени или по значению, промежутки между блоками будут обрабатываться следующим образом:

- При указании мёртвой зоны по значению будут возвращены все значения NULL, а также все следующие за ними выборки данных. То есть мёртвая зона по значению недействительна для точек, разделённых значениями NULL;
- При указании мёртвой зоны по времени значения NULL обрабатываются таким же образом, как любые другие точки. Значения NULL не оказывают никакого влияния на действие мёртвой зоны по времени.

Возврат значений для недопустимых начальных моментов

Один из примеров недопустимого начального момента запроса – начало временного диапазона, предшествующее начальному моменту первого архивного блока. В режиме извлечения по изменению в первой строке результатов будут содержаться значения NULL, а метка времени будет равна начальному моменту, указанному в запросе. Метка времени в следующей строке будет совпадать с начальной отметкой архивного блока. Данные в этой строке будут иметь следующие характеристики:

Атрибут выборки	Значение (16-ричное)	Значение (десятичное)
Value	Мгновенное значение	Мгновенное значение
Quality	0	0
Quality Detail	96	150

При циклическом извлечении моментам, предшествующим начальному моменту архивного блока, будут соответствовать значения NULL.

Другой пример недопустимого начального момента – начальный момент, более поздний по сравнению с текущим временем архиватора IndustrialSQL Server (то есть будущее время). В режиме извлечения по изменению будет возвращено единственное значение NULL. В циклическом режиме значения NULL будут возвращены для каждого запрошенного тэга.

Извлечение данных из архивных блоков и Активного образа

В процессе инициализации подсистема извлечения определяет самые ранние метки времени значений каждого тэга в Активном образе. Если начальный момент в запросе является более поздним по сравнению с этими отметками, это означает, что из Активного образа могут быть извлечены все требуемые данные. Если же начальный момент более ранний, чем самая ранняя отметка, данные будут извлекаться из архивных блоков.

Если временной диапазон запроса захватывает самую раннюю метку, система извлечения заново просматривает Активный образ с целью повторного определения самых ранних меток (поскольку устаревшие данные в Активном образе перезаписываются новыми значениями). Если требуемые данные находятся в Активном образе, они и возвращаются системой, в противном случае данные считываются из архивных блоков. Этот процесс продолжается до тех пор, пока система не сможет больше пользоваться Активным образом либо пока не будет достигнут конечный момент запроса. Важно отметить, что подобная операция выполняется для каждого тэга в отдельности.

Подробнее см. раздел "Влияние способа хранения данных в Активном образе на эффективность извлечения данных" настоящего руководства.

Сервер в/в InSQL

Сервер в/в InSQL I/O Server (aaIOSvrSVC.exe) представляет собой интерфейс, посредством которого клиенты получают данные из Активного образа архиватора IndustrialSQL Server по протоколу DDE или SuiteLink. Сервер InSQL I/O Server способен записывать текущие значения в указанные группы данных, тем самым обеспечивая "режим реального времени". Значения тэгов сервер получает из Активного образа. Подробнее см. раздел "Активный образ" настоящего руководства.

Первоначально в сервере InSQL I/O Server определена единственная группа данных с названием Tagname. В процессе работы сервер "слушает", кто из клиентов, таких как WWClient или WindowViewer™, пытается установить соединение со сконфигурированной группой. После того как клиент установит соединение с сервером InSQL I/O Server, между ними создаётся "горячая" линия связи. Дополнительно об установлении соединений с сервером в/в см. в разделе "Адресация серверов в/в" настоящего руководства.

В частности, приложение InTouch WindowViewer при помощи сервера InSQL I/O Server может получать значения системных тэгов архиватора IndustrialSQL Server для контроля работоспособности системы и генерировать алармы всякий раз, как оно обнаружит нештатную ситуацию.

Примечание. После обновления архиватора IndustrialSQL Server до версии 9.0 нужно изменить имя канала доступа в клиентском приложении с "InSQLIOS" на "aahIOSvrSvc".

По умолчанию, сервер InSQL I/O Server исполняется как одна из служб операционной системы Windows и может запускаться и останавливаться с

помощью системной консоли управления (System Management Console). Системная консоль управления может также использоваться для контроля состояния сервера. Дополнительно о консоли управления см. в Главе 1 "Административные средства" Руководства по администрированию системы архивирования IndustrialSQL Server.

Сервер InSQL I/O Server может только передавать информацию клиентам; с его помощью изменять данные в Активном образе нельзя.

Установить соединение с сервером InSQL I/O Server из удалённого компьютера с помощью NetDDE, указывая как имя виртуального сервера IndustrialSQL Server, так и имя виртуального сервера SQL Server, невозможно. Для этого следует воспользоваться либо протоколом SuiteLink, либо протоколом DDE с указанием настоящего имени активного узла вместо имени виртуального сервера.

ГЛАВА 7

Подсистема событий

Производственные события представляют собой любые изменения производственной ситуации: запуски и остановки оборудования, завершение производственного цикла или текущей смены, изменение типа обработки продукции и действия оператора и т.д.

Подсистема событий, входящая в состав архиватора IndustrialSQL Server, позволяет определять условия возникновения событий и действия, которые должны быть выполнены при их возникновении. В общем, событием можно назвать любую ситуацию, обнаруженную в результате анализа сохранённых данных. Подсистема событий может регулярно проверять условия возникновения событий. Это называется обнаружением событий. При обнаружении того или иного события система может выполнять некоторые заранее определённые действия, называемые реакцией на событие. Следует иметь в виду, что гарантии немедленного выполнения этих действий нет, в некоторых случаях они могут быть приостановлены более приоритетными процессами.

В системе архиватора сохранение информации о событии означает большее, чем просто регистрацию факта его возникновения. Событие представляет собой определённый набор информации, характеризующей момент нарушения каких-либо ограничений на значения тэгов. В состав этой информации входят как момент возникновения собственно события, так и момент его обнаружения по архивным данным. Записи о событиях могут сохраняться в базе данных независимо от того, выполнялись или нет какие-либо ответные действия (т.к. иногда бывает желательно регистрировать только сам факт возникновения события и ничего более).

Подсистема событий выполняет следующие основные функции:

- Определяет возникновение событий путём сравнения архивных данных с некоторыми эталонными значениями.
- Регистрирует записи о возникших событиях в специальной таблице сервера SQL Server (с названием EventHistory).
- Запускает предварительно определённые действия каждый раз при выполнении критериев обнаружения событий.

Дополнительно о конфигурировании событий см. Главу 10 "Определение событий" Руководства по администрированию системы архивирования IndustrialSQL Server.

Содержание

- Детекторы событий
- Реакции на события
- Управление ресурсами подсистемы событий
- Переменные подсистемы событий.

Компоненты подсистемы событий

Компоненты подсистемы событий перечислены в следующей таблице.

Компонент	Описание
Редактор конфигурации Configuration Editor	Одна из частей системной консоли управления System Management Console, используемая для определения событий и возможных реакций.
Рабочая база данных	База данных SQL Server, в которой хранятся определения событий и вся генерируемая подсистемой событий информация (записи детекторов событий, сводки данных и "моментальные" копии значений).
Служба подсистемы событий (aahEventSvc.exe)	Внутренний процесс, координирующий действия по обнаружению событий и выполнению ответных действий. Данный процесс исполняется как одна из служб операционной системы Windows. С помощью системной консоли управления можно настроить параметры автоматического запуска и остановки службы одновременно с системой архиватора IndustrialSQL Server. Служба событий выполняет следующие задачи: ▪ считывает определения событий из Рабочей базы данных; ▪ создаёт детекторы событий и реакции на события, организуя при этом все необходимые нити обработки и устанавливая соединения с базой данных; ▪ выполняет циклы обнаружения событий.
SQL-переменные	Внутренние переменные, которые могут использоваться в запросах на данные о событиях.

Настройка всех параметров подсистемы событий выполняется с помощью системной консоли управления System Management Console.

Об общей архитектуре системы архиватора см. Главу 1 настоящего руководства.

Назначение подсистемы событий

Как правило, подсистема событий архиватора IndustrialSQL Server используется для мониторинга некритичных системных состояний, возникающих только изредка. Например, в число подобных состояний могут входить:

- Все случая в архиве, когда значение некоторого логического тэга было равно 0.
- Наступление определённых даты и времени суток.

- Определение состояния информации в базе данных с помощью SQL-оператора.

При обнаружении события система может выполнять следующие действия:

- Рассылать по электронной почте уведомления о необходимости выполнения мероприятий по техобслуживанию.
- Рассчитывать статистические данные за определённый период времени.
- Сохранять "мгновенные" копии значений технологических параметров.
- Изменять условия сохранения данных (например изменять мёртвые зоны по времени и по значению).
- Выполнять общие действия с базой данных.

Система не предназначена для выполнения задач типа постоянной передачи данных и не должна использоваться подобным образом. Единственное исключение – расчёты сводной информации; система в состоянии непрерывно выполнять подобные вычисления и предоставлять их пользователю для включения в отчёты.

Подсистема событий не должна использоваться в качестве системы выдачи предупреждений. Для уведомления операторов о возникновении особых условий следует пользоваться системами алармов, входящими, например, в состав ИЧМ-приложений на основе приложений InTouch. Эти системы специально предназначены для своевременной выдачи сигналов о возникновении особых производственных и системных ситуаций и обеспечивают отображение, регистрацию и вывод на печать информации о производственных алармах и системных событиях. (Алармы представляют собой сообщения, предупреждающие об отклонениях от нормальных производственных условий, в то время как события представляют собой обычные сообщения о переходе системы в некоторое состояние.) Дополнительно о системе алармов InTouch см. соответствующую документацию.

В отличие от этих систем подсистема событий архиватора предназначена для выполнения указанных действий при обнаружении определённых событий в архивных данных. Система алармов предполагает немедленную выдачу уведомлений о возникновении тех или иных производственных условий. В этом смысле подсистема событий не является системой алармов. Сведения об обнаруженных событиях упорядочиваются в виде очередей и обрабатываются в соответствии с заданными приоритетами.

Характеристики и достоинства подсистемы событий

Подсистема событий обладает следующими основными характеристиками и достоинствами:

- В отличие от систем оперативной выдачи алармов, подсистема событий определяет возникновение событий на основе анализа архивной информации и не функционирует в режиме реального времени. Никакие события не будут пропущены, если только компьютер не будет очень перегружен в течение длительного периода времени.
- Подсистема событий опирается на возможности языка SQL и, таким образом, представляет собой одно из системных средств взаимодействия с базой данных. Пользователь может применять в качестве детекторов событий собственные SQL-запросы и разрабатывать свои реакции на события на основе SQL-операторов.
- В системе имеется множество уже определённых детекторов и реакций на события.

- Обнаружение событий может выполняться внешними средствами. (Запуск детекторов в подсистеме событий обеспечивается COM-механизмом.)
- Временные детекторы (запускаемые при наступлении определённого системного времени) обеспечивают плановое выполнение ряда задач, таких как расчёт сводной информации.
- Система предусматривает возможность функционирования в условиях перегрузки. Если в текущий момент она занята каким-либо видом обработки данных, подсистема событий может "приостановиться" и включиться позднее в моменты меньшей нагрузки. Если система архиватора постоянно работает в условиях перегрузки, уменьшение функциональных возможностей подсистемы событий будет происходить постепенно.
- Пользователь может определять приоритеты реакций на события и указывать, выполнение каких действий никогда не должно задерживаться даже в условиях значительной перегрузки.
- Для мониторинга состояния подсистемы событий имеется набор специальных системных тэгов.

Факторы, определяющие эффективность подсистемы событий

На общую производительность подсистемы событий архиватора IndustrialSQL Server влияет целый ряд факторов, связанных с характеристиками хранения информации и временем обработки запросов на данные. Очень часто в спецификациях эксплуатируемых систем указывается средняя, или "типовая", нагрузка. Вместе с тем они должны предусматривать и пиковые нагрузки, которые могут возникнуть в течение срока службы системы. Производительность системы может зависеть от следующих факторов:

- Достаточности оборудования. Выбор аппаратных средств имеет большое значение для гарантии пиковых рабочих характеристик в различных условиях эксплуатации. Например, нужно предусматривать достаточное дисковое пространство для хранения информации об обнаруженных событиях и результатов выполнения соответствующих действий (различных сводок, "мгновенных" значений и т.д.)
- Доступности процессоров. Подсистема событий зависит от характеристик процессоров так же, как и любая другая программная система на той же платформе. В каждый момент времени за процессорное время конкурирует сразу несколько программных процессов.
- Природы запросов к базе данных, выполняемых подсистемой событий. В частности, поскольку подсистема событий, как правило, оперирует с обычными таблицами SQL Server, её характеристики зависят от возможностей Microsoft SQL Server. Кроме того, обработка запросов обычно требует выполнения множества вычислительных операций и потому очень чувствительна к наличию параллельно исполняющихся задач на том же сервере.
- Временных интервалов SQL-детекторов. Подробнее см. раздел "Временные интервалы SQL-детекторов" настоящей главы.

Производительность подсистемы может колебаться в зависимости от типа события, конфигурации компьютера, действий пользователя и других непредсказуемых факторов в производственной системе с общей базой данных и серверными ресурсами. Нередко бывает очень трудно

определить, какое сочетание аппаратных и программных средств будет оптимально для данных условий эксплуатации. Таким образом, нужно тщательно тестировать параметры подсистемы событий, прежде чем пользоваться ею в производственной системе, особенно в условиях перегрузки.

Событийные тэги

Событийный тэг представляет собой обозначение некоторого определения события в системе. Например, если событие – это достижение температуры 100 ºC в накопителе, описание этого события может быть сохранено в системе как тэг с названием "TankAt100". Событийные тэги архиватора IndustrialSQL Server отличаются от тэгов других типов (аналоговых, логических и символьных). Аналоговые, логические и символьные типы представляют собой определения сохраняемых значений. В отличие от них событийный тэг представляет собой обозначение характеристик того события, которое требуется обнаруживать, включая описание тех действий, которые должны быть выполнены при его обнаружении. Цель определения событийного тэга – обеспечить средство доступа ко всей этой информации.

Событийные тэги создаются и модифицируются средствами системной консоли управления (System Management Console). При определении событийных тэгов нужно указывать следующую информацию:

- Имя, описание и прочие общие конфигурационные параметры.
- Критерии наступления события (условия, которые должны возникнуть), а также периодичность проверки наступления события.
- Необходимость регистрации факта обнаружения события.
- Разрешение или запрет обнаружения данного события.
- Возможные действия, которые должны выполняться при обнаружении события.

Детекторы событий

Для каждого событийного тэга должен существовать соответствующий детектор события. Детектор события представляет собой механизм определения моментов, в которые выполняются критерии наступления события. При определении детектора сначала указывается его тип, а затем соответствующие параметры детектора этого типа. Возможны следующие типы детекторов:

- SQL-детекторы;
- временные (schedule);
- внешние (external).

Общие SQL-детекторы, детекторы аналоговых значений, детекторы логических значений представляют собой все разновидности SQL-детекторов. Временной детектор запускается показаниями часов. Внешние детекторы используются, когда событие генерируется с помощью ActiveX-элемента ActiveEvent.

Для всех типов детекторов в момент запуска подсистемы событий она выдаёт запросы к базе данных. В последующем эти запросы формируются на основе успешного выполнения предыдущего запроса, то есть момент последнего успешного обнаружения события становится начальным моментом планирования его следующего обнаружения.

SQL-детекторы

Детекторы аналоговых и логических значений, а также общие SQL-детекторы обрабатывают информацию, хранящуюся в базе данных. Критерий обнаружения события устанавливается SQL-оператором, который должен выполнять архиватор IndustrialSQL Server. Общие SQL-детекторы могут выдавать запросы как архиватору IndustrialSQL Server, так и Microsoft SQL Server.

Общие SQL-детекторы

Общий SQL-детектор определяет событие на основании критериев, указанных в SQL-операторе. Пользователь может разрабатывать общие SQL-детекторы на базе имеющихся в системе шаблонов либо полностью с самого начала.

В первом случае шаблон выбирается из соответствующего списка во время определения событийного тэга.

Во втором случае новый скрипт нужно внести в таблицу SQLTemplates Рабочей базы данных, чтобы он появился в списке готовых шаблонов. Перед использованием нового скрипта в качестве основы общего SQL-детектора, нужно протестировать его в анализаторе запросов SQL Server Query Analyzer.

Детекторы значений

Имеется два типа детекторов значений:

- Детекторы аналоговых значений (analog specific value detector).
- Детекторы логических значений (discrete specific value detector).

Указанные детекторы позволяют определять, когда значение архивируемого тэга начинает удовлетворять заданному условию. Критерием является сравнение с некоторой эталонной величиной. Если критерий сравнения выполняется, система вносит запись об этом событии в таблицу EventHistory и начинает выполнять указанные ответные действия. Например, критерием для детектора аналогового значения может быть определено условие превышения каким-либо аналоговым тэгов значения 1500, а критерием для детектора логического значения – переход логического тэга в нулевое состояние.

Для детекторов любого типа может быть указан как вид обнаружения по фронту, так и разрешение (точность) возвращаемых результатов по времени. Разрешение по времени используется, если вид обнаружения по фронту установлен в "NONE" (в этом случае режим извлечения является циклическим). Подробнее см. раздел "Параметр wwEdgeDetection" настоящего руководства.

Временные интервалы для SQL-детекторов

Для любого из SQL-детекторов нужно указать интервал времени, через которые система должна его периодически запускать для проверки возникновения события. Выбор длительности этого интервала влияет как на быстроту выполнения ответного действия, так и на общую эффективность системы.

Обнаружение события может произойти значительно позднее его возникновения, будучи зависимым от интервала запуска детектора. Разница во времени между моментами возникновения и обнаружения события называется задержкой обнаружения.

Например, пусть интервал обнаружения для какого-либо детектора установлен равным в 10000 мс (10 с). Это значит, что система будет

запускать детектор один раз в 10 секунд. Если событие возникнет через 2000 мс после очередной проверки, оно останется незамеченным в течение ещё 8000 мс (то есть до тех пор, пока текущий интервал не завершится). Таким образом, чтобы система быстрее обнаруживала событие, интервал запуска детекторов следует делать короче.

Длительность этого интервала влияет также и на момент выполнения ответных действий, поскольку в системе могут быть операции, запланированные на время, равное или превышающее этот интервал.

Устанавливая длительность интервалов запуска, рекомендуется соблюдать следующие условия:

- При определении нескольких детекторов событий для них следует задавать разные интервалы, а не одинаковые.

  Все детекторы сначала объединяются в группы в зависимости от длительности соответствующего интервала запуска и затем упорядочиваются внутри каждой группы. Чем больше детекторов внутри группы, тем позже будет обслужен последний из них. Хотя на обнаружение событий это и не оказывает большого влияние, большое количество детекторов в группе может привести к увеличению задержки обнаружения.

- Не следует без необходимости устанавливать слишком короткие интервалы обнаружения.

  Интервалы обнаружения не следует путать с периодичностью определения системой реального времени изменений в значениях тэгов. Для подсистемы событий больший интервал обнаружения означает, что при каждом сканировании архивных данных возвращается больший объём результатов; никакие события не будут пропущены, если только не будет превышен размер окна обнаружения (см. раздел "Перегрузки детекторов"). Например, пусть в системе определён событийный тэг с интервалом обнаружения в 1 минуту, а периодичность возникновения события равна 5 секундам. Это значит, что в каждом интервале обнаружения система будет находить сведения о 12 событиях. В большинстве случаев это вполне приемлемая быстрота обнаружения. Кроме того, укорочение интервалов обнаружения будет приводить к повышению вычислительной нагрузки на процессор и снижению общей эффективности системы.

Более подробно о работе детекторов см. в разделе "Управление ресурсами подсистемы событий" настоящего руководства.

Таблица EventHistory позволяет определять, следует ли изменять количество детекторов, для которых указаны одинаковые интервалы времени. Если задержка между моментом возникновения события (столбец DateTime) и моментом определения (столбец DetectDateTime) постоянно увеличивается или в одном из интервалов обнаруживается множество событий, возможно, для некоторых детекторов следует либо уменьшить, либо увеличить длительность интервала обнаружения.

Временные детекторы

Временные детекторы определяют момент наступления указанных даты и времени суток по системным часам. Например, подобным событием можно назвать время 14:00 каждый понедельник.

Детекторы времени отличаются от других тем, что они являются детекторами реального времени. Показания системных часов проверяются каждую секунду. Временные детекторы исполняются очень быстро и мало влияют на общую производительность. Это единственный системный механизм обработки событий в масштабе реального времени. Вместе с тем, гарантии точного времени выполнения ответной реакции нет.

Все временные детекторы обслуживаются одной и той же выделенной нитью. Это позволяет отделять операции по детектированию времени от всех остальных действий по обнаружению событий. Все указанные в детекторах моменты времени собраны в виде очереди. При определении нового детектора времени нить вносит соответствующий момент в этот список и сортирует его от самого раннего до самого позднего.

Затем показания системных часов сравниваются с самым первым элементом этой очереди. Если системное время равно значению этого элемента или превышает его, тренд запускает соответствующий механизм детектирования.

Подсистема событий не учитывает переходы на зимнее или летнее время. Если какой-либо детектор времени должен периодически запускаться, начиная с указанной даты, её следует вручную изменять соответствующим образом. Другое решение – выполнять расчёты усреднённых величин не средствами подсистемы событий, а в режиме извлечения среднего с взвешиванием по времени, поскольку в этом режиме переходы на летнее и зимнее время учитываются. С другой стороны, если период усреднения равен одному часу, лучше выполнять эту операцию с помощью подсистемы событий, так как скорость вычислений не будет существенно зависеть от объёма требуемых данных.

Внешние детекторы

При использовании внешних детекторов обнаружение событий осуществляется внешними источниками с помощью ActiveX-объекта ActiveEvent, поставляемого как часть системы архиватора IndustrialSQL Server. В частности, запустить необходимые методы объекта ActiveEvent для генерации события могут любые скрипты InTouch или Visual Basic. Это объект должен быть предварительно установлен в том компьютере, из которого планируется генерировать внешние события.

Дополнительн см. Главу 10 "Определение событий" Руководства по администрированию системы архивирования IndustrialSQL Server.

Реакции на события

Для любого события можно при необходимости определить ответную реакцию системы. Реакция на событие представляет собой действие, которое должно быть выполнено, когда соответствующий детектор обнаружит наступление этого события. Подсистема событий не рассчитана на запуск внешних процессов. Она обладает очень ограниченными возможностями запуска внешних программ и вызова методов посредством COM-интерфейсов внутри каких-либо систем или сети.

Определение реакций на события не обязательно, бывают ситуации, когда система должна просто регистрировать факт наступления событий. В этом случае при определении событийного тэга в списке типов ответного действия надо выбрать пункт "None" (отсутствует).

Общие SQL-реакции

Общая SQL-реакция представляет собой действие, которое выражено в виде SQL-оператора (например запрет сохранения значений тэга в базе данных или копирование данных в отдельную таблицу или базу).

Общие SQL-реакции могут создаваться как на основе имеющихся шаблонов, так и с самого начала. В общей SQL-реакции нельзя указывать несколько запросов к архиватору IndustrialSQL Server и пользоваться операторами GO. Кроме того, при обращении к архивным данным нужно соблюдать синтаксис, поддерживаемый провайдером InSQL OLEDB. Перед

использованием SQL-оператора нужно протестировать его с помощью анализатора запросов SQL Server Query Analyzer.

Создание "мгновенных" копий

Реакция в виде создания "мгновенной" копии представляет собой запись в специальные таблицы SQL Server значений указанных аналоговых, логических и символьных тэгов, метки времени которых совпадают с временем обнаруженного события. Сведения о качестве также регистрируются. Значения тэгов сохраняются в таблицах соответственно их типам: AnalogSnapshot, DiscreteSnapshot и StringSnapsot.

Выполнение реакции этого типа требует SQL-объединения таблиц расширения и таблиц "мгновенных" копий. Формирование объединений и запись найденных значений в таблицы "мгновенных" копий может занимать долгое время. Причина заключается в том, что большинство из используемых для "мгновенных" копий таблиц являются обычными таблицами SQL Server, и потому им свойственны все ограничения Microsoft SQL Server. Таким образом, чем больше "мгновенных" копий создаёт подсистема событий, тем выше будет нагрузка на сервер по выполнению транзакций.

Внимание! Подсистема событий не является системой накопления данных. НЕ пользуйтесь реакциями этого типа для передачи данных из таблиц расширения в таблицы SQL Server. Подобное решение приведёт к чрезвычайно низкой производительности системы и существенному замедлению сохранения данных.

Чтобы определить, сколько "мгновенных" копий может делать система, соответствующие запросы к серверу следует оформлять в виде пакетного файла и не пользоваться подсистемой событий. Периодическое исполнение этого файла с максимально допустимой для компьютера сервера быстротой позволяет точнее оценить соответствующую производительность системы в течение заданного периода времени с учётом коэффициента надёжности. Следует иметь в виду, что "мгновенные" копии логических тэгов создаются гораздо дольше "мгновенных" копий аналоговых тэгов.

Рассылка сообщений по электронной почте

Реакции этого типа представляют собой отсылку готового сообщения средствами Microsoft Exchange. Хотя они и обеспечивают распространение сообщений о возникновении событий, реакции этого типа не должны заменять собой подсистему алармов. Рассылку уведомлений о нештатных ситуациях по электронной почте следует выполнять средствами специальных подсистем алармов типа SCADAlarm и других.

Изменение мёртвых зон

Реакции этого типа представляют собой изменение величины мёртвой зоны по времени или значения для одного или нескольких тэгов, для которых указан режим сохранения по изменению. (Мёртвые зоны по значению действительны только для аналоговых тэгов.) Изменение мёртвых зон выполняется, в частности, для повышения частоты сохранения значений тэгов в архиве после возникновения определённого события. Например, для определения причин отключения какого-либо оборудования можно повысить частоту сохранения некоторых его рабочих параметров.

Расчёт сводок

Реакции этого типа представляют собой вычисление сводных величин по значениям определённого набора тэгов в указанном временном диапазоне с указанной точностью. При определении реакции типа расчёта сводки нужно указать тип действия (называемого сводной операцией) и перечень

соответствующих тэгов, значения которых должны обрабатываться. Подсистема событий может находить среднее арифметическое, минимальное, максимальное и суммарное значения. Расчёт сводных величин обычно выполняется в следующих ситуациях:

- Когда данные должны храниться в течение очень долгого периода времени. Поскольку сводные величины занимают значительно меньше места, чем исходные, то даже в системах скромного размера подобная информация может храниться годами.
- При создании производственных отчётов. Во многих случаях сводные величины дают больше информации, чем исходные данные. Например, знание общего объёма выпуска за день может иметь больше значения, чем знание темпов производства.
- При необходимости интеграции с управленческими системами. Обращение к исходным архивным и оперативным данным архиватора IndustrialSQL Server будет эффективно при использовании средств, которые могут указывать в своих запросах временные характеристики данных. Однако не все клиентские программы поддерживают эти расширения языка SQL. Сводные таблицы позволяют уменьшать количество данных до разумных объёмов, с которыми могут работать обычные клиентские SQL-приложения.

Действия по расчёту сводных величин, как правило, запускаются детекторами времени. Однако их также можно выполнять и детекторами любых других типов.

Перед выполнением групповых функций данные с неудовлетворительным качеством не отфильтровываются. Для использования данных только с достоверным качеством, нужно указать общую SQL-реакцию, выдающую запрос на выполнение сводных действий к таблице History, в котором указано, что в столбце Quality должно быть записано значение 0.

Результаты выполнения групповых функций сохраняются в таблице SummaryData Рабочей базы данных.

Внимание! Реакциями типа расчёта сводки следует пользоваться с осторожностью. Указание слишком большой временной точности может привести к существенному снижению производительности системы.

Средняя арифметическая, минимальная и максимальная величины могут также определяться в режимах извлечения среднего с взвешиванием по времени, минимального и максимального значений соответственно. Подробнее см. раздел "Параметр wwRetrievalMode" настоящего руководства. При выборе типа расчёта сводок следует иметь в виду следующие обстоятельства:

- В режиме извлечения среднего с взвешиванием по времени величина среднего зависит от разницы между выборками данных по времени, в то время как при расчёте среднего подсистемой событий она является обычной статистической величиной (средним арифметическим). Подробнее см. раздел "Режим извлечение среднего с взвешиванием по времени" настоящего руководства;
- Расчёт средних величин во время извлечения устраняет проблемы, связанные с переходами на летнее или зимнее время. Подробнее см. раздел "Детекторы времени" настоящего руководства.

Приоритизация реакций на события

В подсистеме событий определены три следующие очереди реакций на события:

- Очередь "критичных" (critical) реакций, в которую входят все реакции на события с важнейшим приоритетом. События данного уровня важности

обрабатываются в первую очередь. Назначать приоритеты следует с осторожностью, присваивать высокий приоритет нужно действительно важным событиям с малыми временами обслуживания. Никогда не следует назначать высокий приоритет реакциям типа создания "мгновенной" копии или расчёта сводки. Защиты очереди критичных событий от перегрузки системы нет; в условиях перегрузки реакции этого типа могут выполняться с задержкой или не выполняться вообще.

- Очередь "обычных" (normal) реакций, в которую входят все реакции на события с обычным приоритетом. События с данным уровнем важности обрабатываются только после всех важных событий.
- Очередь "задержанных" (delayed) реакций, в которую входят все реакции на события, определённые с задержкой выполнения. Задержка выполнения представляет собой минимальный интервал времени, который должен пройти после обнаружения события, прежде чем система начнёт выполнение соответствующих ответных действий.

Управление ресурсами подсистемы событий

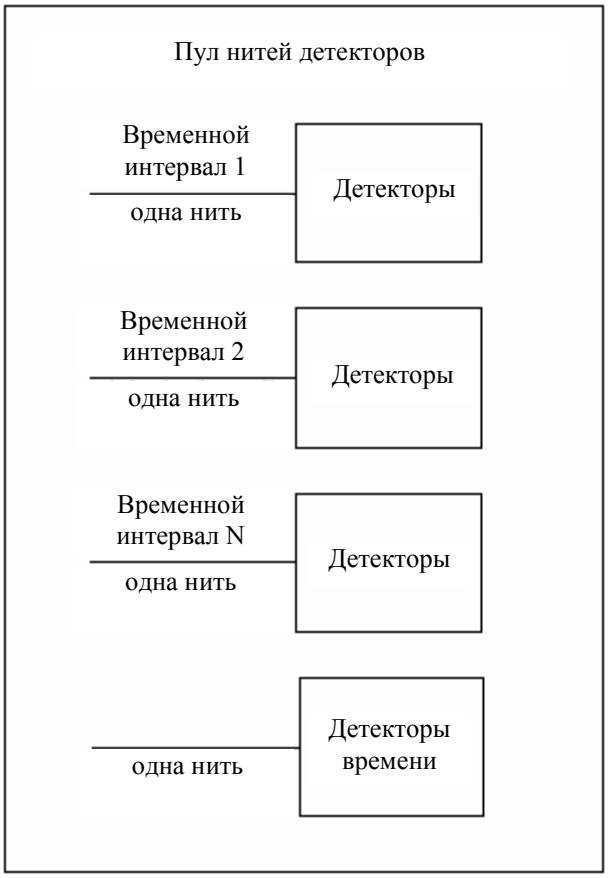
Всеми ресурсами системы, необходимыми для обнаружения событий и выполнения ответных действия, управляет Служба подсистемы событий (aahEventSvc.exe). Системные ресурсы выделяются под детекторы и реакции на события в виде нитей. Нить представляет собой один из компонентов операционной системы, независимо от других выполняющий конкретную функцию в рамках объемлющего процесса. В рамках общего процесса подсистемы событий детекторы событий и реакции на события распределяются по разным нитям, которые могут исполняться независимо друг от друга и, таким образом, являются более эффективными.

В подсистеме событий имеется две группы, или "пулы", нитей. Один пул нитей предназначен для детекторов, другой – для реакций. Служба событий автоматически создаёт оба пула нитей, если в системе определён хотя бы один событийный тэг.

В задачи управления ресурсами входит также контроль соединений с базой данных, требуемых компонентами подсистемы событий, а также разрешение ситуаций перегрузки событиями и отказов выполнения запросов.

Пул нитей детекторов событий

Пул нитей детекторов событий состоит из набора нитей, выделяемых SQL-детекторам, и единственной нити, обслуживающей все детекторы времени. Каждая нить имеет собственное соединение с базой данных. Структура пула показана на следующем рисунке:



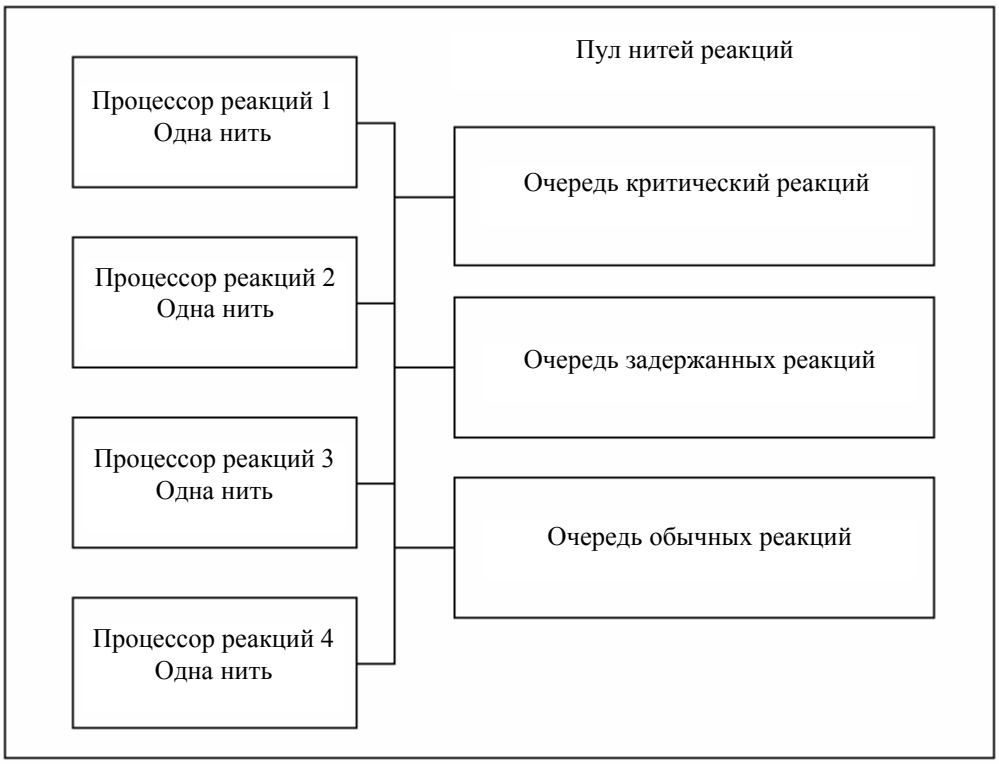
SQL-нить связывается с тем или иным тредом в зависимости от интервала обнаружения, указанного при определении событийного тэга. Каждому интервалу соответствует отдельная нить. Например, пусть в системе определено три детектора событий с интервалами в 10, 15 и 20 секунд. Каждый из них будет обслуживаться отдельной нитью, которых в системе будет три.

Если в системе будет определено три детектора, для двух из которых будут указаны одинаковые интервалы длительностью в 10 с, а для третьего – интервал в 15 с, то первые будут обслуживаться одной и той же нитью, а третий детектор – другой нитью.

При обслуживании нескольких детекторов с одинаковой длительностью интервалов обнаружения соответствующие SQL-операторы будут исполняться в последовательном порядке. Таким образом, предыдущий SQL-оператор должен вернуть результаты прежде, чем будет запущен последующий. После обнаружения события (после возвращения результатов) сведения о нём записываются в таблицу EventHistory и соответствующая реакций ставится в очередь в пуле нитей реакций.

Пул нитей реакций на события

Пул нитей реакций по сути состоит из четырёх нитей, выполняющих действия из трёх очередей. Каждая нить имеет собственное соединение с базой данных.



Нити обслуживают следующие очереди:

- Очередь критичных реакций.
- Очередь обычных реакций.
- Очередь задержанных реакций.

После того как нить завершит выполнение текущего ответного действия, она извлекает новый элемент из одной из очередей. Если очередь критичных реакций не пуста, они обрабатываются первыми. Ответные действия в этой очереди выполняются в том порядке, в каком они были в неё поставлены, то есть действие, которое было поставлено в очередь первым, первым и будет выполнено.

После выполнения всех действий из очереди критичных реакций нити начинают обслуживание очереди задержанных реакций. Элементы очереди упорядочены по длительности задержки. Первыми исполняются действия с самой короткой задержкой.

Обслуживание очереди реакций с обычным приоритетом начинается только осле того, как опустеют очереди критичных и задержанных реакций. Как и критичные, обычные реакции обслуживаются в порядке их постановки в очередь.

Соединения подсистемы событий с базой данных

В следующей таблице приведены сведения о количестве соединений с базой данных SQL Server, необходимых для различных компонентов подсистемы событий.

Компонент	Количество требуемых соединений с базой данных
Служба событий	1
SQL-детекторы	по 1 для каждого интервала обнаружения
Детекторы	1

времени	
Нити реакций на события	4

Перегрузки событиями и отказы выполнения запросов

Подсистема событий способна обрабатывать случаи отказа выполнения запросов SQL-детекторов и реакций на события, а также постепенно прекращать выполнение функций в случае перегрузки детекторами и реакциями на события.

- Отказы выполнения запросов.

  При ошибке выполнения запроса для SQL-детектора он автоматически будет запущен на выполнение снова. Начальный момент окна обнаружения останется таким же, пока обнаружение не будет выполнено.

  При ошибке выполнения запроса для SQL-реакции он дополнительно будет запускаться три раза. При каждом исполнении запроса система будет создавать новое соединение с базой данных. Если тип реакции – создание "мгновенной" копии, при выполнении повторного запроса таблицы "мгновенных" копий будут сначала "очищаться".

- Перегрузки детекторами.

  Перегрузка детекторами возникает, когда система становится не в состоянии своевременно обслужить все детекторы. Эта ситуация разрешается с помощью окна обнаружения. Окно обнаружения определяется как разница по времени между текущим системным моментом и моментом последнего обнаружения. Если "ширина" окна начинает превышать один час, некоторые события будут пропущены, а в журнал будет записано соответствующее сообщение.

- Перегрузка реакциями на события.

  Перегрузка реакциями на события возникает, когда система становится не в состоянии своевременно выполнить все ответные действия. Защита от перегрузки реализована только для очереди реакций с обычным приоритетом. Реакция на событие не будет ставиться в эту очередь, если самая ранняя реакция находится в ней более часа. (Предполагается, что система настолько перегружена, что у неё нет ресурсов выполнить одну реакцию в течение часа.) Таким способом предотвращается накопление реакций в очереди в ситуациях, когда система неспособна их обслуживать. После восстановления работоспособности системы реакции не ставятся в очередь до тех пор, пока разница по времени между самым ранним и самым поздним элементами не составит 45 минут (75% предельного времени). В общем, если система становится слишком перегруженной, реакции в очередь не ставятся. Данная ситуация отражается в журнале ошибок, однако информация о каждой пропущенной реакции в нём не регистрируются (сначала о первой пропущенной, затем только о каждой сотой).

  Защиты от перегрузки для критичных реакций нет, поскольку предполагается, что событий такой степени важности в системе определено очень немного. Кроме того, защита от перегрузок не реализована и для задержанных реакций.

Переменные подсистемы событий

В подсистеме событий определён набор переменных, применение которых облегчает обнаружение событий и выполнение ответных действий. Их

назначение – облегчить разработку запросов пользователем (или редактором конфигурации). Непосредственно перед выполнением такого запроса вместо переменных подсистема событий подставляет соответствующие требуемые значения. Запрос, который получает архиватор IndustrialSQL Server, никогда не содержит ссылок на какие-либо переменные.

Сведения о переменных подсистемы событий приведены в следующей таблице:

Переменная	Описание и допустимые значения
@EventTime	Дата и время обнаружения события текущим детектором.
@EventTagName	Имя соответствующего событийного тэга.
@StartTime	Начальный момент, указываемый в запросе детектора.
@EndTime	Конечный момент, указываемый в запросе детектора.

Переменные @StartTime и @EndTime могут использоваться только в запросах детекторов событий. Переменные @EventTime и @EventTagName могут использоваться только в запросах реакций на события.

Имена переменных чувствительны к регистру символов.

Запрос детектора может выглядеть следующим образом:

```
SELECT DateTime
    FROM History
      WHERE Tagname = 'BoilerPressure' AND Value > 75
          AND DateTime > '@StartTime'
          AND DateTime < '@EndTime'
```

Переменные @StartTime и @EndTime представляют собой просто фиктивные параметры, которые заменяются реальными при обнаружении событий с течением времени.

Ниже показан пример использования переменных в запросах реакций на события.

```
SELECT * INTO TEMPTABLE
    FROM History
      WHERE DateTime = '@EventTime'
          AND TagName IN
          (SELECT TagName FROM SnapshotTag
          WHERE EventTagName = '@EventTagName'
          AND TagType = 1)
```

Примечание. Указанные переменные действительны только в контексте подсистемы событий и не могут использоваться в запросах других клиентских средств типа анализатора запросов SQL Server Query Analyzer.