АВАДС® Сервер архивирования

Руководство пользователя

Москва 2024

Товарные знаки

АВАДС®, AVADS®, АВАДС Сервер архивирования, АВАДС СА являются товарными знаками, принадлежащими ООО «АВАДС СОФТ» (далее по тексту – компания АВАДС СОФТ). Прочие названия и обозначения продуктов в документе являются товарными знаками их производителей, продавцов или разработчиков.

Интеллектуальная собственность

О документе

Материал, содержащийся в данном документе, прошел доскональную проверку, но компания АВАДС СОФТ не гарантирует, что документ не содержит ошибок и пропусков, поэтому оставляет за собой право в любое время вносить в документ исправления и изменения, пересматривать и обновлять содержащуюся в нем информацию.

Оглавление

Введение

Назначение продукта

Сервер архивирования (АВАДС СА) предназначен для создания высокоскоростных и высоконадежных систем хранения данных реального времени. Он относится к классу программ TSDB (база данных временных рядов).

АВАДС Сервер архивирования является российским продуктом. Он внесен в реестр российского программного обеспечения Минкомсвязи под номером №17156.

АВАДС СА позволяет вести архивы изменения технологических и коммерческих показателей в системах АСУТП, диспетчеризации промышленных объектов и объектов ЖКХ, системах коммерческого и технического учета.

Функция архивирования является одной из главных в подобных системах. Поэтому к ней предъявляются очень жесткие требования:

- По надежности хранения данных,
- Скорости сохранения,
- Скорости доступа,
- Обеспечению заданной глубины хранения данных.

Сегодня задача создания систем архивирования с возможностью сохранять в секунду миллионы параметров, обеспечивая при этом быстрый доступ к сохраненным данным и высокую надежность, очень актуальна. Причина в стремительно растущих запросах на цифровизацию городов и промышленности. Сервер архивирования позволяет решить такую задачу.

Инновационный продукт

Сервер архивирования разработан на базе инновационной технологии SSDS (Solid Segment Data Storage). Эта технология обеспечивает высочайшую скорость записи и извлечения данных.

Технология SSDS, на которой базируется Сервер архивирования, включает в себя три основных решения:

- оригинальная организация данных;
- уникальная и очень компактная система индексации;
- мощная система кеширования.

Эти решения позволяют не только получать мгновенный доступ к нужным данным, но и быстро восстановить целостность базы при частичном повреждении носителя или индекса.

Технология SSDS разработана компанией АВАДС СОФТ и защищена **патентом РФ** № **2793082**.

Поддерживаемые платформы

Сервера архивирования является кроссплатформенным продуктом. В его коде использовались только те решения, которые не зависят от операционной системы и аппаратной платформы. Поэтому СА может работать на самых разных платформах и под управлением большого набора операционных систем. Такими платформами и ОС являются:

- **Windows** на платформах X86, AMD64, ARM, ARM64;
- **Linux** на платформах X86, AMD64, ARM, ARM64, PPC64, PPC64le, MIPS, MIPCle, MIPS64, RISCV64, S390X;
- **Darwin** компьютеров Apple на платформах AMD64 и ARM64;
- **IOS** мобильных устройств Apple на платформе ARM64;
- **Эльбрус** на платформе VLIW (2C+, 4C, 8C, 8CB, 2C3, 16C).
- **Android** на платформах X86, AMD64, ARM, ARM64;
- **NetBSD** на платформах X86, AMD64, ARM;
- **FreeBSD** на платформах X86, AMD64, ARM, ARM64;
- **OpenBSD** на платформах X86, AMD64, ARM, ARM64;
- **Plan9** на платформах X86, AMD64, ARM;
- **AIX** на платформе PPC64;
- **Dragonfly** на платформе AMD64;
- **Illumos** на платформе AMD64;
- **Solaris** на платформе AMD64;

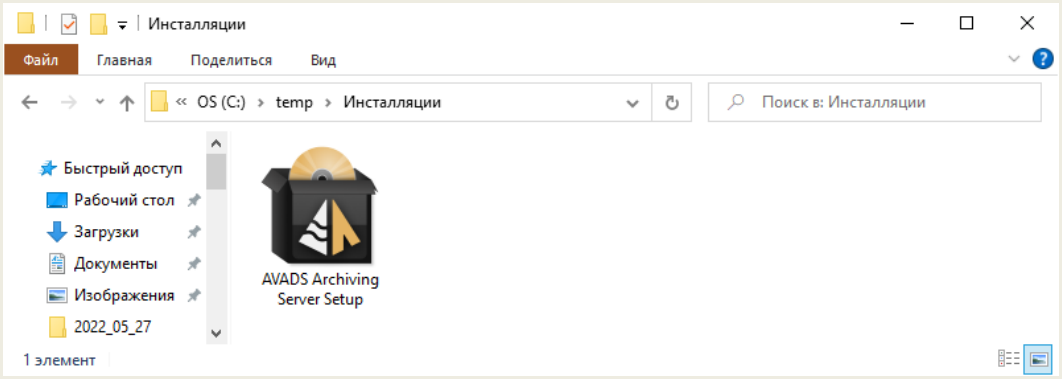
Инсталляция Сервера архивирования

Процедура инсталляции зависит от операционной системы, на которой будет работать.

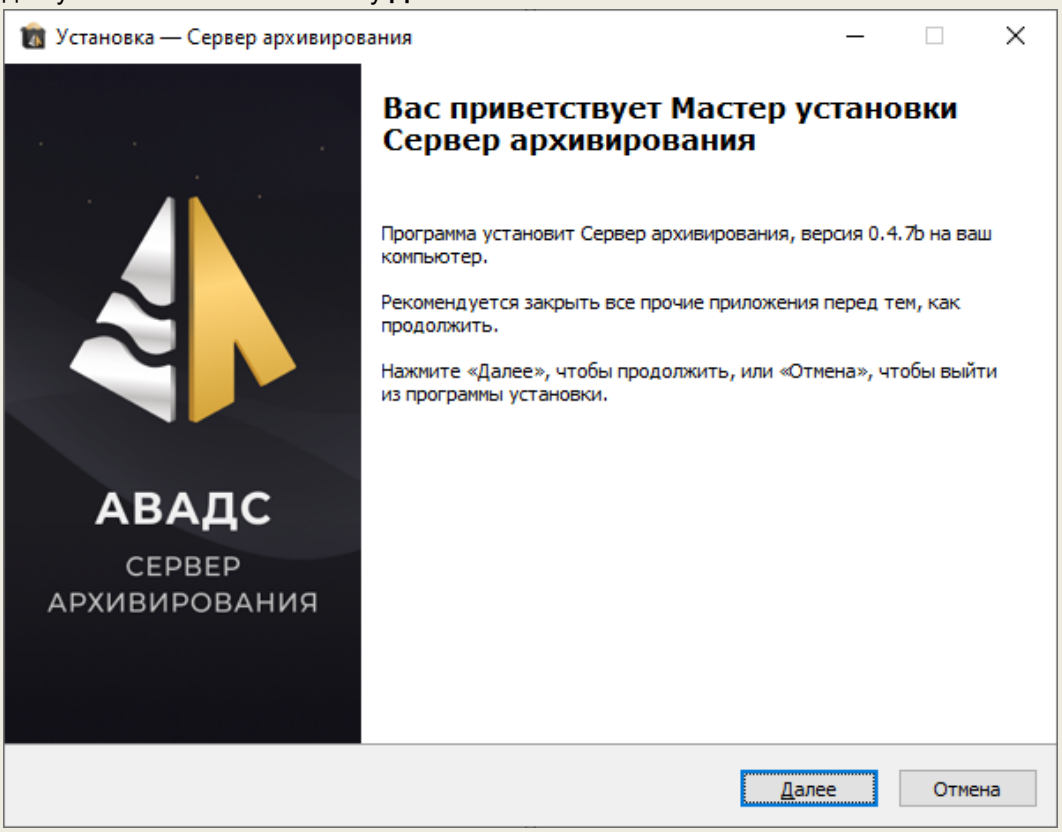
Инсталляция под Windows

Для инсталляции Сервера Архивирования под ОС Windows:

- скачайте соответствующий дистрибутив с сайта **avads.ru**;
- запустите исполняемый файл **AVADS Archiving Server Setup.exe**;

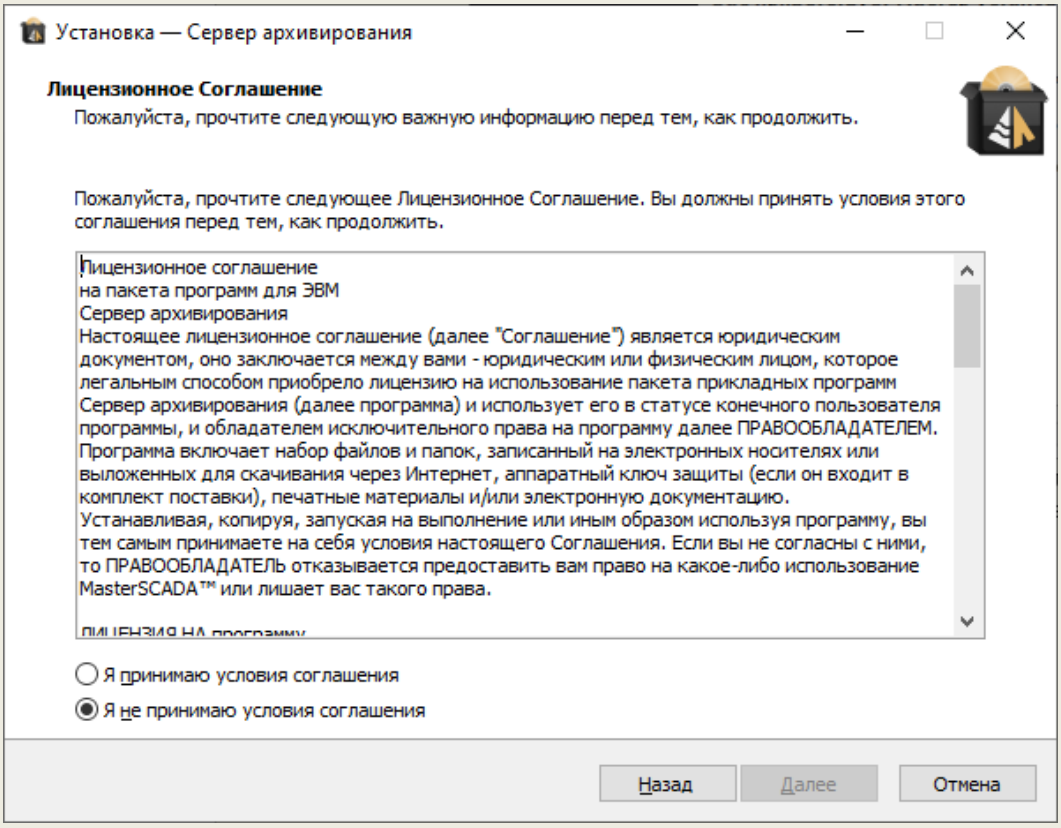


- На экране появится диалог с предложением инсталлировать Сервер архивирования для установки нажмите кнопку **Далее**

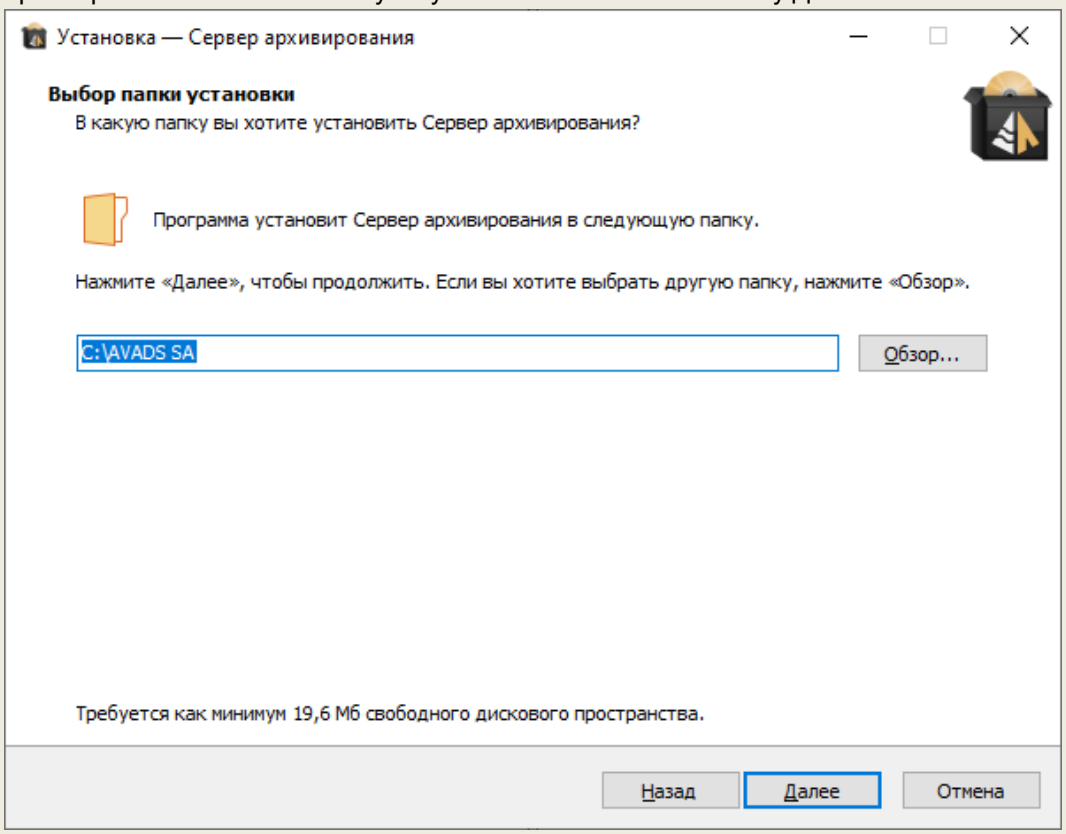


- на экране появится диалог с предложением принять лицензионное соглашение. Если Вы с ним согласны и принимаете, то выберите соответствующий пункт в меню под

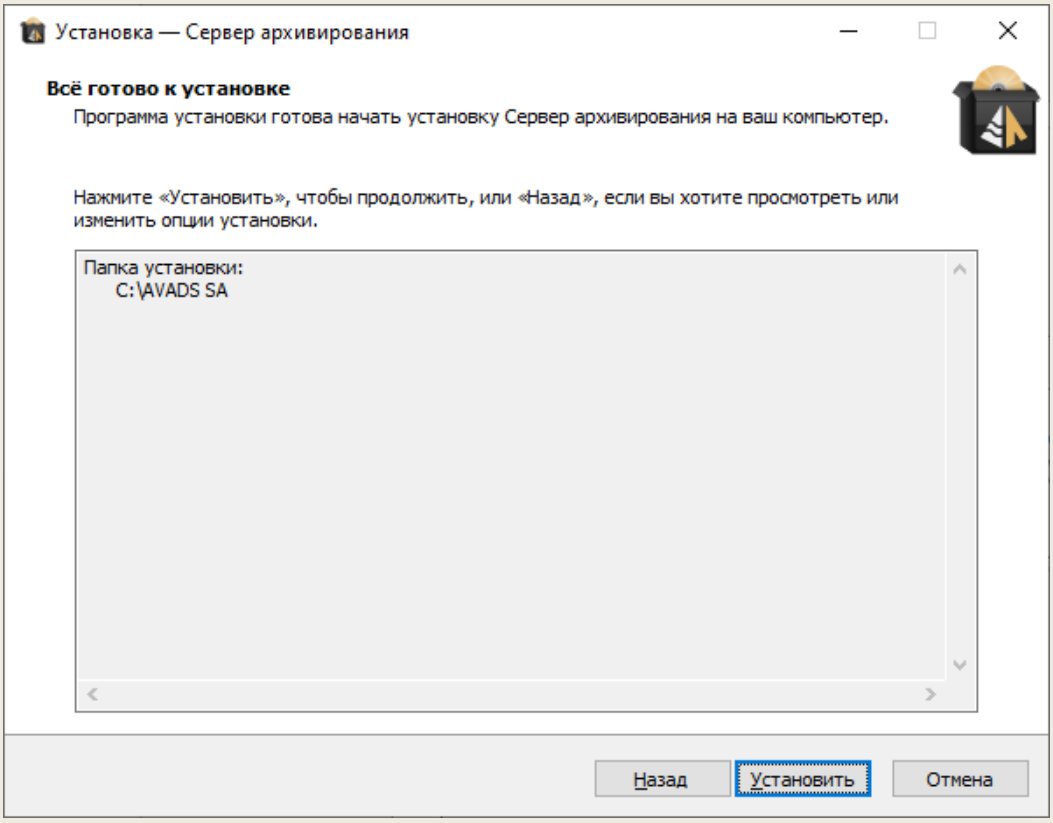
текстом лицензионного соглашения и нажмите кнопку **Далее**.



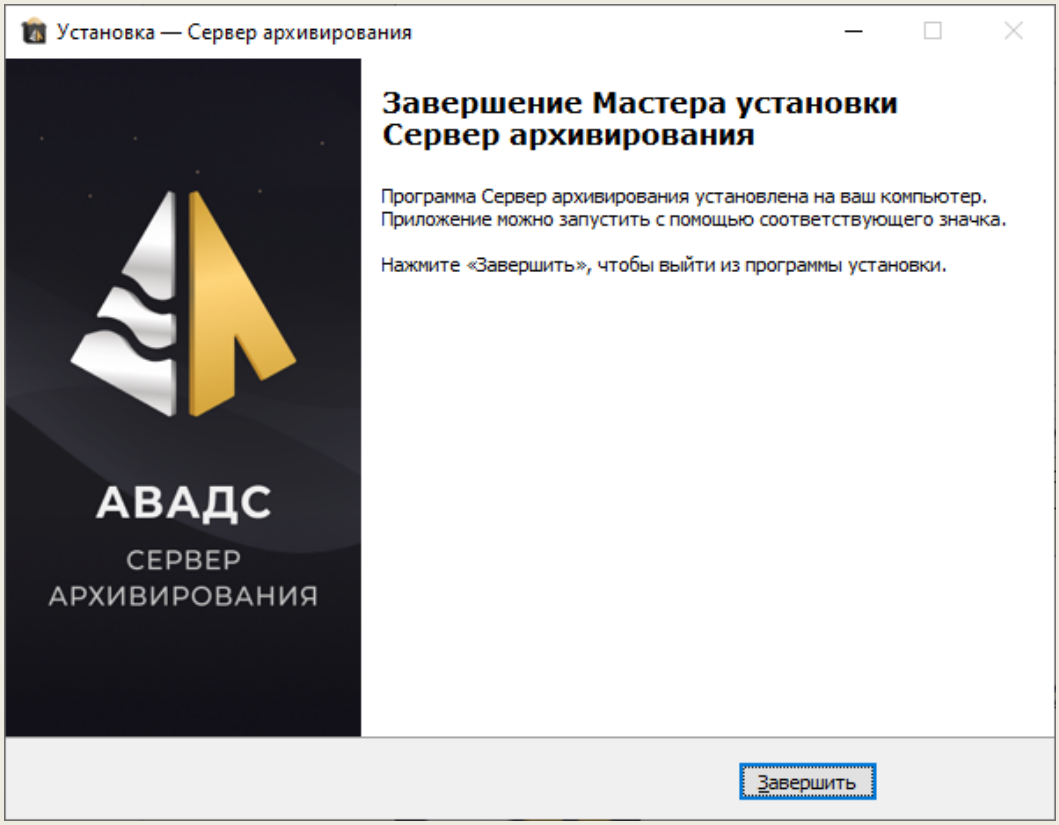
- На экране появится диалог для выбора папки, в которую будет установлен Сервер архивирования. Укажите папку с путем к ней и нажмите кнопку Далее.



- На экране появится диалог с установленными настройками инсталляции. Для их изменения нажмите кнопку **Назад**. Если все задано верно, то для переходя непосредственно к установке Сервера архивирования нажмите кнопку **Установить**.



- Если процесс установки Сервера архивирования завершился успешно, то на экране появится диалог завершения установки. Чтобы закончить инсталляцию нажмите на этом диалоге кнопку **Завершить**.



После завершения инсталляции в меню **Пуск** Windows появится папка **Сервер архивирования**. В ней будут ярлыки:

- **Деинсталлировать Сервер архивирования** – запуск программы деинсталляции;
- **Руководство пользователя** – открыть Руководство пользователя Сервера архивирования. Документ поставляется в формате pdf.
- **Сайт АВАДС СОФТ** – открыть сайт производителя;
- **Сайт Клиент администрирования** – открыть клиент администрирования на локальной машине.

Сервер архивирования устанавливается под Windows как сервис. Поэтому он стартует автоматически при загрузке операционной системы и не требует ярлыка для запуска.

При установке коммерческой версии для дальнейшей работы ее надо активизировать. Чтобы сделать это запустите **Клиент администрирования** кликнув по соответствующему ярлыку. Нажав правой кнопкой мыши на адресе Сервера архивирования, в появившемся меню выберите команду **Активировать** и в появившемся диалоге введите код активации (подробнее см. раздел **Лицензирование** Руководства пользователя).

Инсталляция под Linux

Для инсталляции Сервера архивирования под операционную систему Linux выполните следующие действия:

- Скачайте с сайта avads.ru дистрибутив, соответствующий используемой аппаратной платформе. Дистрибутив представляет из себя архивный файл;
- Создайте на компьютере директорию, в которую собираетесь установить Сервер архивирования;
- Распакуйте содержимое архива в эту директорию;
- Прочитайте файл readme.txt для получения информации об особенностях запуска Сервера архивирования на используемой аппаратной платформе и версии Linux;

При установке коммерческой версии ее надо активировать (если устанавливается ДЕМО-версия, то этот пункт можно пропустить). Для этого запустите Сервер архивирования, откройте клиент администрирования, нажмите правую кнопку мыши на названии сервера в панели управления, выберите пункт меню **Активировать** и в появившемся диалоге введите код (подробнее см. раздел **Лицензирование** Руководства пользователя).

Инсталляция под другие операционные системы

Сервер архивирования может быть скомпилирован под очень большое количество платформ и операционных систем. При необходимости использовать Сервер архивирования под какой-нибудь платформой, не приведенной в этом описании, обращайтесь в техническую поддержку.

Архитектура и запуск

Сервер архивирования компании АВАДС СОФТ включает в себя **Сервер БД** (сервер баз данных) и **Клиент администрирования**.

- **Сервер БД** обслуживает запросы клиентов, сохраняет полученные данные и предоставляет их по запросу, выполняет математические вычисления, сохранение и восстановление бекапов, а также ряд других функций;
- **Клиент администрирования** предназначен для настройки Сервера БД и его баз данных, контроля за его работой и для просмотра сохраненных данных в табличном виде и в виде трендов, а также для ряда других сервисных функций.

Сервер Архивирования не имеет ограничений на число баз и число клиентов кроме лицензионных и накладываемых возможностями вычислительных средств, на которых он устанавливается.

> ***Внимание**! Из-за особенностей передачи данных по протоколу JSON и обработки их браузером **Клиент администрирования** не может просматривать базы данных с числом тегов более 16 миллионов. При этом администрирование подобных баз доступно.*

Запуск Сервера БД

Под Windows

Под Windows Сервер базы данных может быть запущен следующими способами:

- Автоматический запуск как сервиса при старте операционной системы;
- Запуск с командной строки.

При запуске из командной строки надо перейти в директорию установки Сервера архивирования и запустить на исполнение файл:

db_core.exe [-debug]

При запуске с параметром **debug** Сервер архивирования будет вести расширенные логи своей работы.

Под Linux

Для запуска Сервера базы данных под Linux надо из папки установки Сервера архивирования выполнить команду:

db_core [-debug]

Параметр **debug** запускает сохранение расширенных логов работы Сервера архивирования.

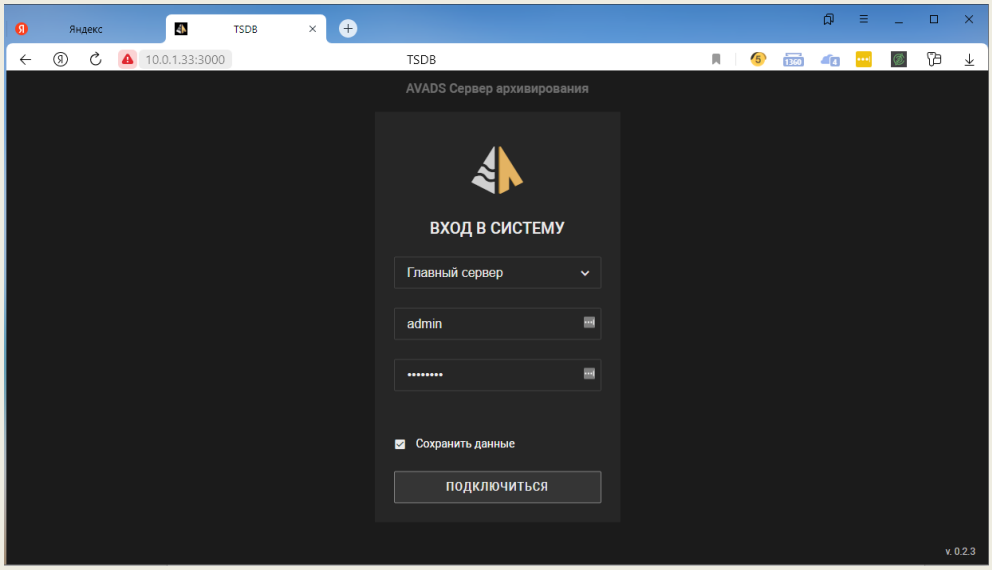
> ***Внимание**! Перед запуском Сервера базы данных надо объявить файл **db_core** исполняемым. Это делается следующей командой из папки установки Сервера архивирования:*
>
> ***chmod +x db_core***

Запуск клиента администрирования

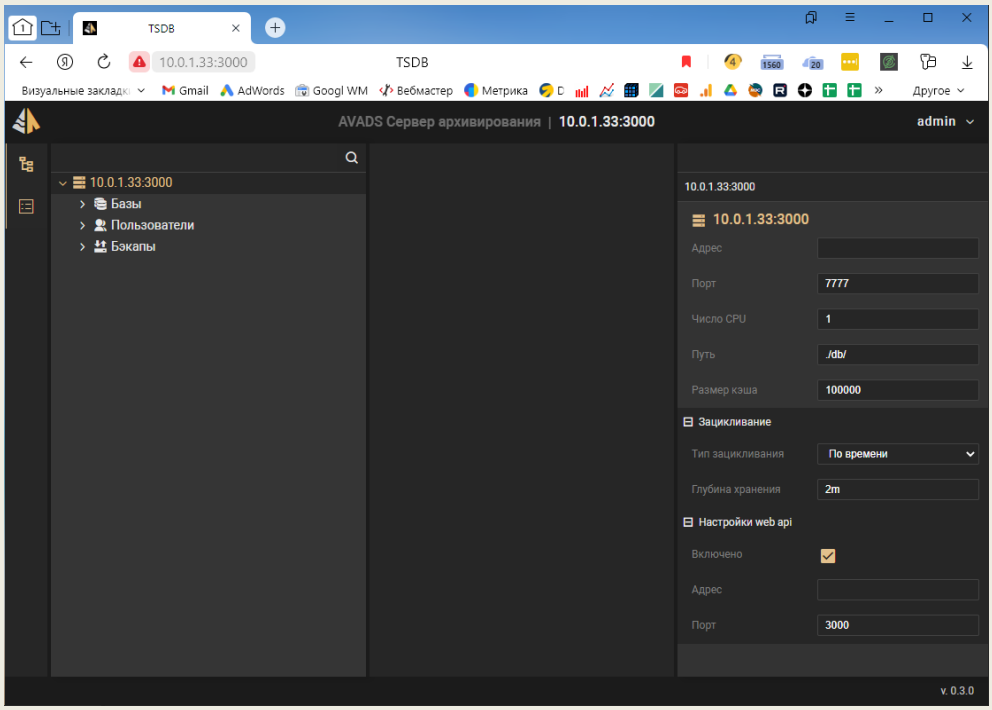
Клиент администрирования загружается Сервером БД по WEB-интерфейсу в интернет браузер. Для этого в адресной строке браузера надо указать IP-адреса сервера БД и порт доступа. Как правило используется порт 7779.

Если подключение выполняется к Серверу БД, работающему на том же компьютере, то IP-адрес будет 127.0.0.1. При этом если используется операционная система Windows, то для запуска Клиента администрирования можно воспользоваться ярлыком **Сайт Клиент администрирования** из папки **Сервер архивирования** меню **Пуск**.

Если адрес набран верно, то появится следующий диалог:



После ввода логина и пароля, позволяющих подключиться к серверу БД появится интерфейс клиента администрирования, представленный на следующем рисунке.



Интерфейс клиента администрирования

На рисунке представлен интерфейс клиента администрирования Сервера архивирования.

На нем можно выделить три основные области:

- Панель управления
- Область просмотра данных
- Панель свойств

Кроме того, в интерфейсе присутствует тулбар управления видимостью панелей и строка статуса. Каждый клик мыши на иконках тулбара управления видимостью панелей убирает или открывает соответствующую панель.

В строку статуса выводится информация о наличии системных сообщений Сервера архивирования и версия продукта. Клик мыши на индикаторах системных сообщений в строке статуса открывает окно просмотра журнала системных сообщений.

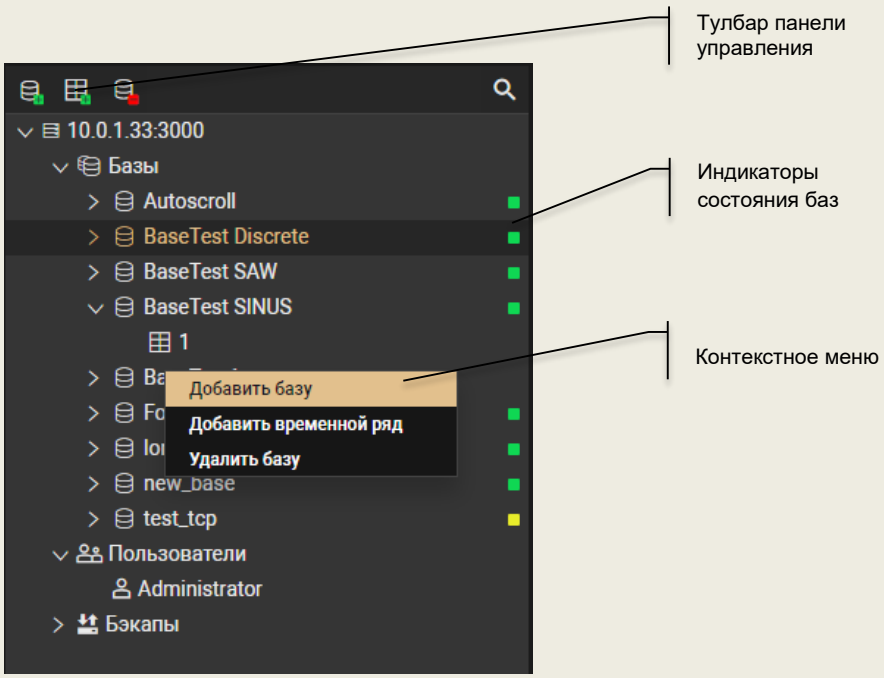
Панель управления

Панель управления – это область в которой представлена структура элементов, обслуживаемых сервером архивирования. Такими элементами являются:

- базы данных;
- временные ряды;
- группы временных рядов;
- пользователи;

- средства управления бэкапами;
- средства управления синхронизацией баз данных;
- средства управления резервированием;
- средства визуализации данных.

При нажатии правой кнопкой мыши (ПК) на любом элементе в панели управления откроется контекстное меню с допустимым набором действий относительно данного элемента. Пример такого меню показан на следующем рисунке.



В **тулбаре** панели управления выводятся иконки управления. Набор иконок меняется в зависимости от выделенного в дереве элемента. Эти иконки соответствуют командам контекстного меню для данного элемента.

В разделе **Базы** дерева панели управления напротив каждой базы выводится индикатор состояния этой базы данных. Цвет этих иконок соответствует следующим состояниям:

- Зеленый – данный клиент администрирования подключен к этой базе;
- Желтый – с этой базой данных подключен еще один или несколько клиентов;
- Красный – база повреждена и работать с ней невозможно;
- Бурый – база данных занята системным процессом, работа с ней пока невозможна;
- Нет индикатора – с базой никто не работает и ее статус неизвестен.

В зависимости от состояния базы накладываются ограничения на действия с ней. Например, невозможно удалить базы данных со статусами зеленый, желтый и бурый.

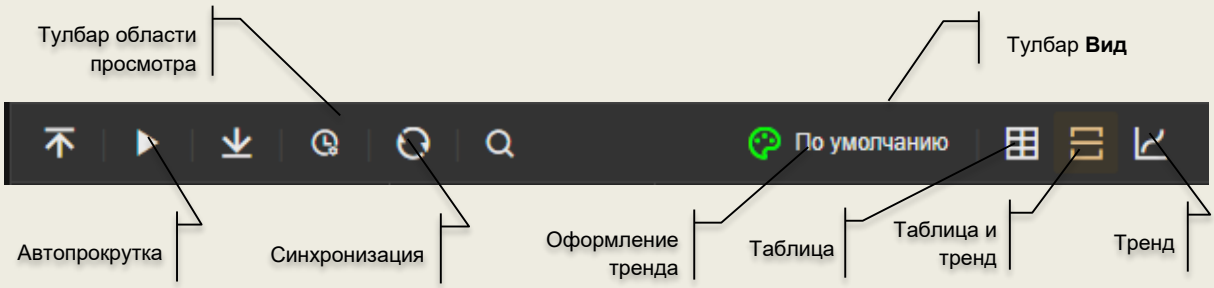
Область просмотра данных

Область просмотра данных – в эту область выводятся для просмотра сохраненных значений из баз данных. Значения временных рядов могут быть представлены в табличном виде и в виде трендов.

Для просмотра данных временного ряда надо дважды кликнуть на его имени в дереве панели управления. Клиент администрирования позволяет открыть для просмотра сразу несколько временных рядов. Для каждого из них в области просмотра создается отдельная вкладка. Чтобы перейти к просмотру любого из открытых рядов надо кликнуть на вкладке с его именем.

На представленном ниже рисунке открыта вкладка с данными временного ряда с именем **SIN**.

Архивные данные временных рядов могут выводиться в табличной форме, в виде тренда или одновременно и в том и в другом виде. Для управления представлением данных используется тулбар **Вид**.



При совместном просмотре данных в виде таблицы и тренда можно их синхронизовать в обоих представлениях. Для этого надо кликнуть на команде **Синхронизация** тулбара области просмотра. Повторный клик на этой команде отключает синхронизацию.

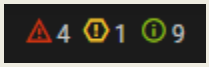
Если включить режим Автопрокрутка, то данные будут автоматически подчитываться из базы данных по мере их поступления. Для включения этого режима надо кликнуть на команде **Автопрокрутка** в тулбаре области просмотра.

Панель свойств

Панель свойств – в эту панель выводятся свойства элемента выделенного в **панели управления**. Часть свойств доступны для редактирования, а часть имеет чисто информационный характер. Объем свойств доступных для редактирования зависит от прав, заданных текущему пользователю.

Журнал сообщений

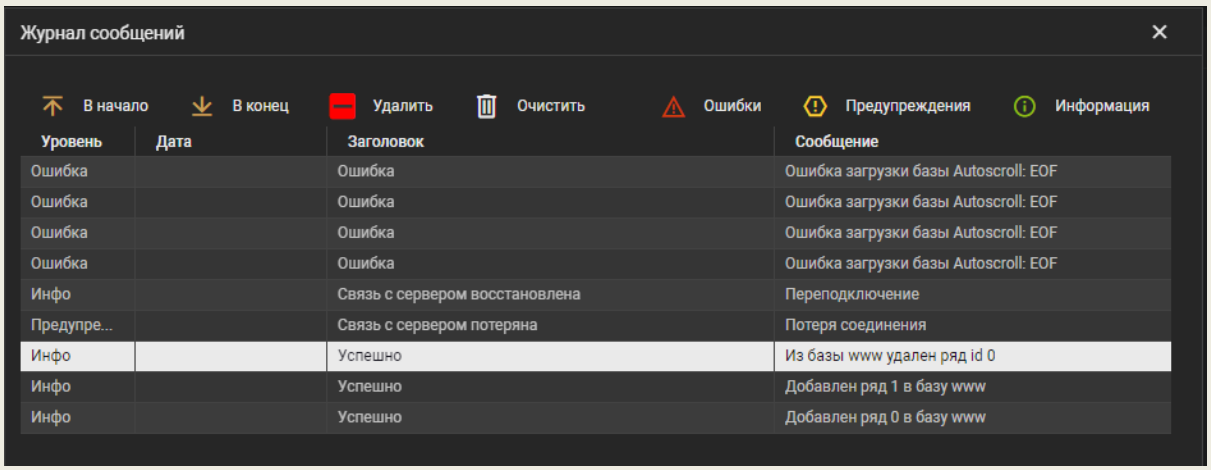
В строке статуса клиента администрирования выводится индикатор сообщений Сервера архивирования. Он выглядит следующим образом:



Этот индикатор содержит значки типов сообщений и число сообщений соответствующего типа. Всего Сервер архивирования формирует сообщения трех типов:

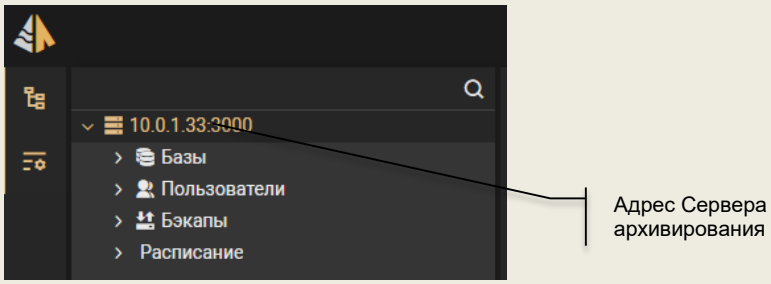
- Информация об ошибках;
- Предупреждения;
- Информационные сообщения.

При клике мышкой по индикатору сообщений открывается окно журнала сообщений, показанное на следующем рисунке.

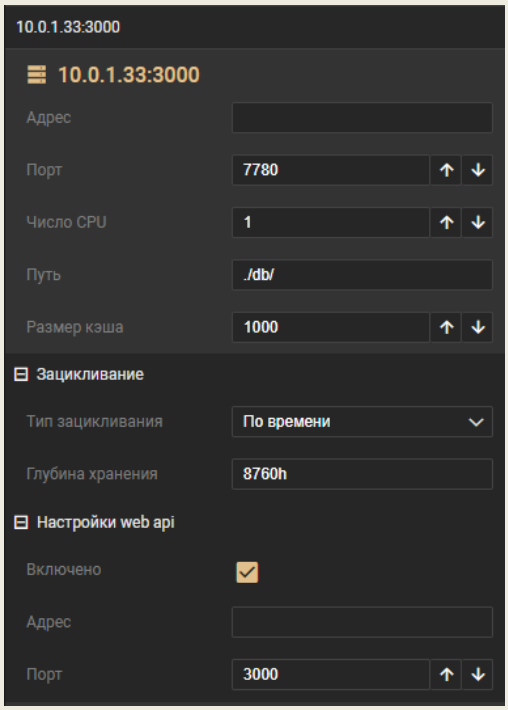


Настройка Сервера архивирования

Чтобы настроить Сервер архивирования надо кликом мыши выделить его адрес в панели управления как показано на рисунке.



При этом панель свойств примет следующий вид:



Доступ по интерфейсам

Настройки **Адрес** и **Порт** в верхней части панели свойств используются для задания IP-адреса и порта для обмена данными с сервером архивирования по протоколу AVADS TCP.

Для настройки обмена по протоколу AVADS WEB предусмотрен специальный раздел **Настройки web api**. В этом разделе можно сняв флаг **Включено** можно запретить подключаться к Серверу архивирования через Интернет.

Настройки **Адрес** и **Порт** раздела **Настройки web api** используются для указания IP-адреса и порта обмена по протоколу AVADS WEB.

Многоядерность

Сервер архивирования может решать очень крупные задачи по архивированию и представлению данных, требующие значительных вычислительных ресурсов. При этом чаще

всего ему придется делить ресурсы процессора с другими приложениями. Поэтому по умолчанию Сервер архивирования запускается **на 1 ядре** процессора.

Число используемых ядер процессора можно изменить с помощью паромера **Число ядер CPU** в панели свойств. Он задает максимальное число ядер, которые могут использоваться Сервером архивирования.

Использование многоядерного режима эффективно только при ведении нескольких баз и взаимодействии с несколькими клиентами.

> ***Внимание**! При работе только с одной базой и взаимодействии только с одним клиентом использование нескольких ядер процессора снижает производительность сервера*

Зацикливание баз данных

Механизмы зацикливания

Зацикливание баз данных высвобождает место на носителе для сохранение новых поступающих данных за счет удаления наиболее старых. Такая возможность является необходимой для систем, ведущих архивирование длительное время.

В Сервере архивирования зацикливание архивов может выполняться в двух режимах:

- По заданной глубине хранения данных. Глубина хранения может задаваться как для всей базы данных, так и индивидуально для каждого тега;
- По достижению заданного размера файла базы данных.

Возможен еще один вариант зацикливания. В нем не указывается предельный размер файла базы данных. Этот размер устанавливается автоматически при исчерпании свободного места на носителе.

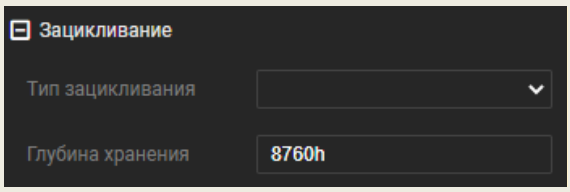
Сервер архивирования позволяет настроить зацикливание на трех уровнях:

- Сервер;
- База данных;
- Тег (временной ряд).

Настройки на уровне сервера по умолчанию применяются к базам данных, которые ведет данный сервер. Если для базы данных задать собственные настройки, то настройки сервера будут игнорироваться. Тоже самое с настройками для тегов. По умолчанию к ним применяются настройки базы данных. Но если для тега задать собственные настройки, то настройки базы данных будут игнорироваться.

Настройка зацикливания

Для настройки зацикливания для сервера, базы данных и тега в **панели свойств** предусмотрен специальный раздел. Он показан на следующем рисунке.



В поле **Тип зацикливания** указывается зацикливать ли данные, если зацикливать, то каким способом и наследовать ли настройки.

В поле Глубина хранения задается время, в течении которого хранятся данные. Время задается в виде текстовой строки. На рисунке задано, что данные хранятся 8760 часов. Это соответствует одному году. В строке задания глубины хранения можно использовать следующие символы: **d** – дни, **h** – часы, **m** – минуты. Вот несколько примеров настройки глубины хранения:

- Два года – «730d»;
- Полтора часа – «1h30m»;
- Полтора дня – «1d12h».

Настроить зацикливание можно также используя протоколы AVADS TCP и AVADS WEB. Описание этих протоколов приведено ниже.

Размер кэша

Среди настроек Сервера архивирования присутствует такая настройка как **Размер кэша**. Эта настройка позволяет задать число блоков записей, которые будут храниться в памяти при чтении данных. Наличие этих данных в оперативной памяти позволяет гораздо быстрее передать их клиенту при повторном запросе.

Таким образом увеличение размера кэша позволяет более быстро отдавать часто запрашиваемые данные. Однако при задании большого размера кэша мы можем занять всю оперативную память и Сервер архивирования не сможет работать.

По умолчанию размер кэша установлен 1000 блоков записей. Это требует оперативной памяти приблизительно 4 МБ. Но поскольку Сервер архивирования постоянно работает с кэшем перезаписывает его и обновляет, то реально используемый размер памяти может быть больше. Задавать размер кэша следует с учетом размера оперативной памяти в компьютере и тех задач, которые будет выполнять Сервер архивирования.

> ***Внимание****! Чрезмерное увеличение размера кэша может привести к неработоспособности Сервера архивирования.*

Классы хранимых данных

Сервер архивирования может сохранять следующие классы данных:

- **Атомарные данные** - данные любых типов, размер которых не превышает 8 байт. К таким относятся, например: bool, int, long, dlong, float, dfloat, ...;
- **Данные типа blob** - данные произвольного размера. Это могут быть, например, структуры, массивы, тексты, изображения и пр.

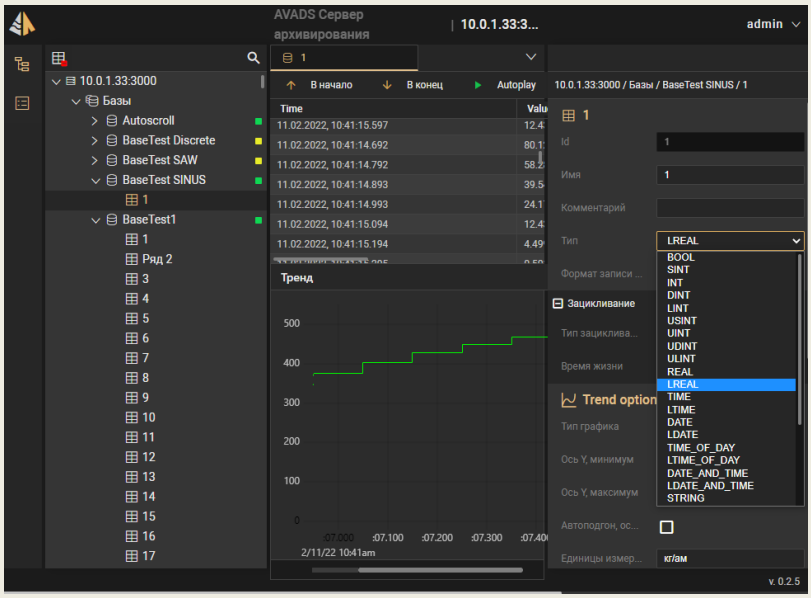
Базовые механизмы хранения и доступа к данным обоих классов одинаковые. Отличие состоит только в логике их обработки. Такой унифицированный подход позволяет обеспечить одинаково высокую скорость сохранения и доступа к данным независимо от их типа.

Атомарные данные

Для хранения любых атомарных данных используются записи одинаковой структуры и размера. Каждая запись хранит время в Unix-формате с точностью до 100 ns (8 байт) и содержит 8 байт для хранения значения. Кроме того, 4 байта в записи используются для хранения признака достоверности (качества) данных.

Серверу архивирования не важно какой тип данных он сохраняет. По умолчанию для всех временных рядов устанавливается тип **SANY**. Это внутренний тип, который означает, что сервер никак не учитывает реальный тип значений записей. Клиент, который передает данные серверу на сохранение и затем запрашивает их знает реальный тип данных и соответственно их обрабатывает.

Однако, Сервер архивирования может проводить обработку данных: выполнять прореживание и вычислять статистические параметры. Чтобы сервер мог это сделать для временного ряда надо указать тип его значений. Кроме того, если для временного ряда задан тип значений в клиенте администрирования можно просматривать его значения в табличном виде и на графике. Задать тип значений временного ряда можно с помощью клиента администрирования в панели свойств выделенного временного ряда



Кроме того, задать тип временного ряда можно при его создании или редактировании используя команды 8 и 10 протокола AVADS TCP (См. ниже).

Данные класса blob

Назначение и структуру записанного в **blob** массива байтов определяет приложение, которое его записывает. Сервер Архивирования принимает от клиента массив байтов и отдает его по запросу. Никакой обработки этих данных в сервере не предусмотрено. Поэтому для временных рядов, сохраняющих данные класса **blob**, устанавливается тип значений **BANY**. Однако клиент администрирования может визуализировать некоторые типы данных, которые сохраняются в записях со значениями данного класса. Например, это могут быть данные типа **string**.

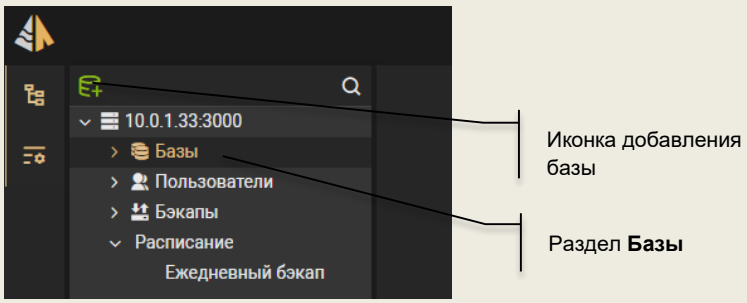
Установить тип данных для временного ряда можно так же, как и для атомарных данных, из клиента администрирования или с помощью команд создания или редактирования 8 и 10 протокола AVADS TCP (См. ниже).

Записи с данными класса **blob** также размещаются в базе данных поблочно. Размеры блоков одинаковы для всей базы данных как для атомарных данных, так и для данных класса **blob**. Отличием в данном случае является отсутствие требования одинакового количества записей в одном блоке. Поскольку **blob** может быть весьма значительного размера (например, фотография, осциллограмма или видео), то возможны ситуации, когда одна запись размещается в нескольких блоках.

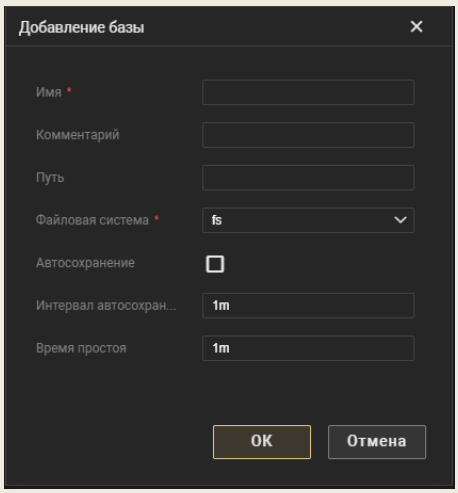
Работа с базами данных

Создание и настройка баз данных

Для создания новой базы данных следует выделить в клиенте администрирования раздел **Базы** и нажать иконку **Добавить базу** в тулбаре панели управления.



После этого появится диалог **Добавление базы**, показанный на следующем рисунке.



В этом диалоге надо задать имя базы данных, комментарий к базе данных, путь к файлам базы данных, тип хранения базы данных и параметры автосохранение (автосохранение и тип хранения базы описаны ниже).

Если не задать путь к базе данных, то она будет создан в директории, указанной в настройках Сервера архивирования.

Чтобы редактировать свойства базы данных следует выделить ее имя в разделе **Базы** панели управления. При этом в панели свойств отобразятся настройки выбранной базы данных и ее текущий статус.

Статус базы данных формируется Сервером архивирования. Каждый бит статуса передает информации о активации соответствующего состояния базы данных:

- 1-й бит = 1 – имеются активные подключения к базе;
- 2-й бит = 1 – база повреждена и не пригодна к работе;
- 3-й бит = 1 – база занята системным процессом, никакие операции невозможны.

Для удаления базы данных следует нажать правую кнопку мыши на ее имени в панели управления и в появившемся меню выбрать команду **Удалить базу**. При этом база удаляется из списка Сервера архивирования, а также удаляются ее файлы с диска.

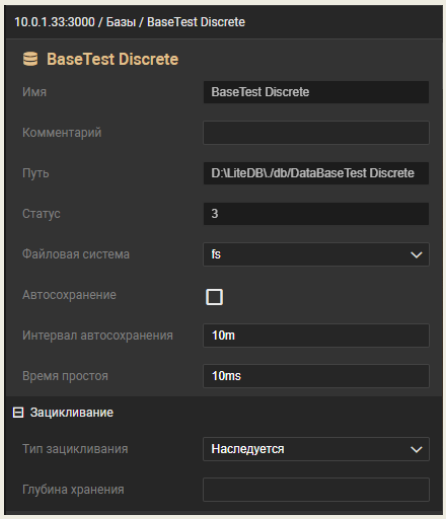
Автосохранение баз данных

Высокая производительность Сервера архивирования по сохранению и выборке данных обеспечивается в частности тем, что эти операции выполняются поблочно.

То есть перед сохранением на диск в памяти компьютера формируются блоки записей. Причем для каждого тега формируется свой блок. После того как число записей в памяти достигнет заданного значения блок сохраняется на диск.

Такой подход позволяет добиться очень высокой производительности, но имеет один большой недостаток. Он заключается в том, что если тег редко меняется, то может пройти значительное время до заполнения блока записей в памяти компьютера и записи его на диск. Если при этом произойдет какая-либо аварийная ситуация (например, пропадание питания), то по данному тегу может быть потеряна история за продолжительный срок.

Чтобы избежать такой ситуации в Сервере архивирования предусмотрена процедура **автосохранения**. Для ее включения и настройки надо выделить имя базы данных в панели управления. После этого в панели свойств надо установить флаг **Автосохранение** и задать параметры **Интервал автосохранения** и **Время простоя**.



Эти параметры задаются в виде числа и временной размерности. Вот несколько примеров:

- 10m – 10 минут;
- 20s – 20 секунд;
- 1h – 1 час;
- 100ms – 100 миллисекунд.

Интервал автосохранения задает период времени через который выполняется процедура автосохранения. При выполнении этой процедуры Сервер архивирования проверяет по каждому тегу было ли добавление новых записей с последней проверки. Если новые записи были, то проверяется сколько прошло времени после последней записи. Если это время больше указанного **Времени простоя**, то выполняется сохранение блока на диск.

Таким образом с помощью автосохранения можно настроить гарантированный минимум потери данных.

> ***Важно!*** *Уменьшая параметр Интервал автосохранения мы уменьшаем возможные потери при аварийной ситуации, но при этом мы сокращаем производительность Сервера архивирования и увеличиваем число актов записи блоков на диск. Последнее может быть важно при использовании SSD-дисков.*

Параметр **Время простоя** позволяет не выполнять автосохранение блоков для тегов, значения которых меняются часто. У таких тегов блоки записей в памяти заполняются быстро. Поэтому нет необходимости выполнять их принудительное сохранение – они итак часто сохраняются. Если задать значение времени простоя заведомо маленьким (на рисунке задано 10 миллисекунд), то сохраняться будут блоки всех тегов, по которым добавлялись новые записи.

Тип хранения базы данных

Существует два типа хранения базы данных:

- В виде одного файла;
- В виде группы файлов одинакового размера.

В первом случае все блоки записей пишутся в один файл. Этот файл растет в размере по мере добавления новых записей. После зацикливания базы данных рост размера файла прекращается и новые данные пишутся поверх старых.

Во втором случае сначала процесс выглядит так же, как и в первом. До тех пор, пока размер базы не превысит ограничение (**1 ГБ**). После этого все новые данные начинают записываться в новый файл. Далее при достижении новым файлом ограничения в 1 ГБ вновь создается еще один файл. Так продолжается пока база не зациклится.

В обоих случаях в рамках базы создается единое пространство блоков записей. Только в первом случае все блоки хранятся в одном файле, а во втором – в нескольких.

Преимуществом второго способа является скорость открытия базы данных при ее значительных размерах. Дело в том, что процедура открытия большого файла может занимать значительное время.

Другим достоинством хранения базы данных в нескольких файлах является более высокая надежность. Дело в том, что при повреждении одного из файлов база в целом остается работоспособной – пропадают данные только из поврежденного файла.

Задать тип хранения базы данных можно только при ее создании. Это делается в диалоге **Добавление базы** с помощью настройки **Тип хранения**. В дальнейшем поменять тип хранения базы невозможно. Если все же требуется изменить тип хранения, то надо создать новую базу данных с нужным типом хранения и перенести в нее данные из старой базы с помощью механизма бэкапов (см. ниже).

Создание и восстановление бекапов

Сервер архивирования позволяет выбрать произвольный набор параметров и сделать по ним **бэкап** - сохранить их значения за заданный интервал времени в файл. Затем данные из файла **бэкапа** могут быть восстановлены в ту же или другую базу данных.

Для бэкапа могут быть выбраны все сохраняемые в базу параметры. Например, можно раз в месяц сохранять бэкап архива на съемный носитель. Это позволяет обеспечить защиту от потери данных.

Ввозможность сохранить бэкап позволяет вести аналитику событий на отдельном компьютере не нагружая дополнительными запросами сервер, обеспечивающий сохранение данных.

Сервер архивирования может сохранять или восстанавливать бэкап одновременно с записью новых данных в базу. Причем процесс восстановления данных из бэкапа слабо влияет на производительность сервера.

Особенности работы с бэкапами

Чтобы работать с бэкапами в Сервере архивирования предусмотрены два типа задач:

- Задача создания бэкапа;
- Задача восстановления бэкапа.

Эти задачи имеют почти одинаковый набор параметров. Подробное описание задач и их настроек приведено ниже.

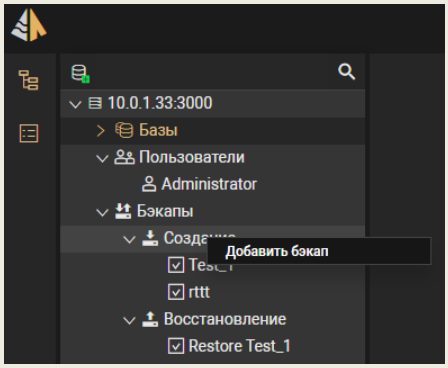
В настройках задач существуют такие параметры как **имя файла бэкапа** и **путь к файлу бэкапа**. В этих настройках можно использовать следующие лексемы:

- "%y" - год
- "%m" - месяц
- "%d" - день
- "%t" - время
- "%base" - название базы

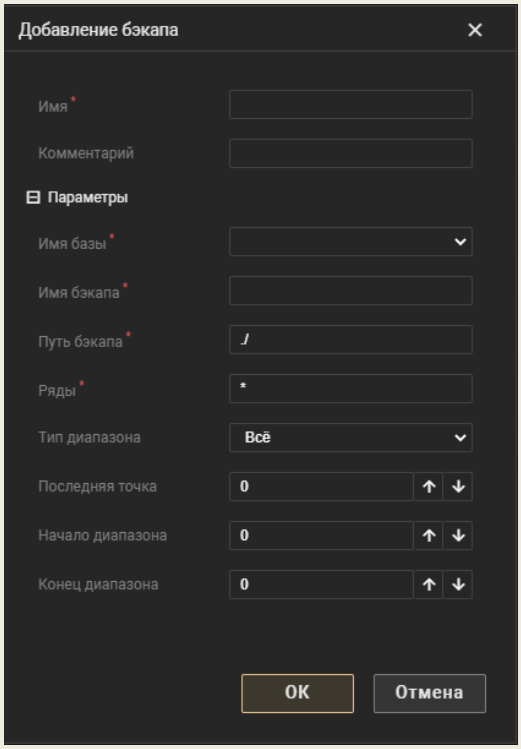
Например, если путь сохранения файла бэкапа для базы данных Base_1 года задать следующим образом **‘”%base”/”%y”/”%m”’**, то путь к файлу бэкапа в сентябре 2022 будет **/Base_1/2022/09.**

Создание бэкапов

Чтобы выполнить бэкап надо сначала создать его описание, а затем дать команду его выполнить. Чтобы создать описание бэкапа надо в **Панели управления сервером** открыть раздел **Бэкапы**. Затем следует нажать правую кнопку мыши на подразделе **Создание** и в контекстном меню выбрать команду **Добавить бэкап,** как показано на рисунке.



После этого на экране появится диалог настройки бэкапа.



В нем следует задать:

- имя бэкапа;
- комментарий к бэкапу;
- базу данных из которой делается бэкап;
- имя файла бэкапа,
- путь к файлу бэкапа,
- список временных рядов, записи которых будут сохраняться в бэкап.

Для настройки списка временных рядов, сохраняемых в бэкап можно использовать следующие способы:

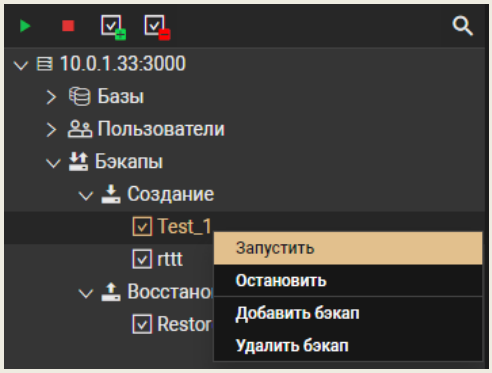
1. «*» - сохранение всех временных рядов из базы данных;
2. a,b,c,… - перечисление через запятую числовых идентификаторов. Например, если записать **3, 7, 15** , то в бэкап будут сохранены временные ряды с числовыми идентификаторами 3, 7 и 15;
3. m-n – указание диапазона числовых идентификаторов. Например, чтобы указать все временные ряды с 5 по 27 надо написать **5-27**;
4. Комбинация второго и третьего способа. Например, запись **1,5,7-15, 28** означает, что в бэкап будут сохранены времянные ряды с идентификаторами 1, 5, с 7 по 15 и 28.

Кроме того, следует настроить временные рамки для бэкапа. В бэкап могут сохраняться:

- Все записи выбранных временных рядов;
- Записи временных рядов из заданного диапазона времени;
- Записи временных рядов от времени последнего сохранения бэкапа до текущего времени.

Тип временного диапазона задается в поле **Тип диапазона** диалога **Добавление бэкапа** выбором из списка. Если выбрать **Фиксированный диапазон** , то затем в том же диалоге надо время начала и конца диапазона.

Команда выполнения бэкапа может быть подана вручную из клиента администрирования.

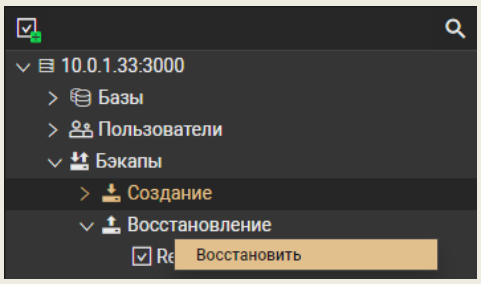


Для этого надо кликнуть правой кнопкой мыши на названии выбранного бэкапа в дереве панели управления и в появившемся меню выбрать команду **Запустить**.

Кроме того, можно настроить выполнение команды создание бэкапа по расписанию.

Восстановление бэкапов

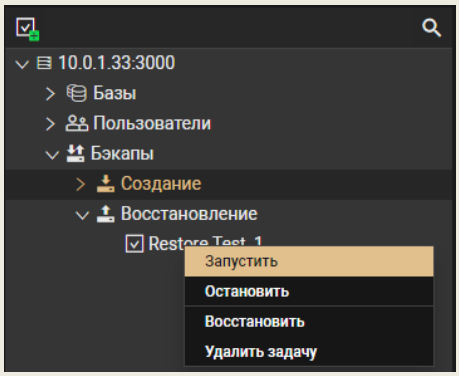
Для восстановления бэкапа в базу следует выполнить следующий набор действий. Сначала надо создать описание задачи восстановления. Для этого надо кликнуть правой кнопкой мыши на разделе **Восстановление** папки **Бэкап** панели управления.



В появившемся меню надо выбрать команду **Восстановить**. После этого на экране появится диалог в котором надо указать:

- имя задачи;
- имя базы данных;
- имя файла бэкапа;
- путь к файлу бэкапа.

После создания задачи ее можно запустить на выполнение. Для этого надо кликнуть правой кнопкой мыши на названии задачи в панели управления и в появившемся меню выбрать команду **Запустить**.

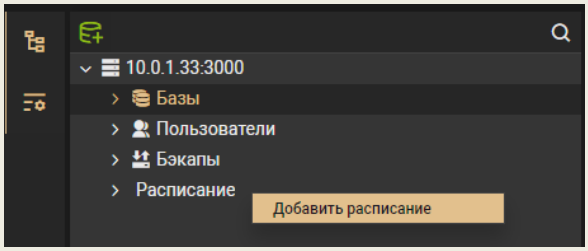


Создавать описание и выполнять бэкапы можно с помощью специальных команд протоколов AVADS TCP и AVADS WEB. Описание этих протоколов приведено ниже.

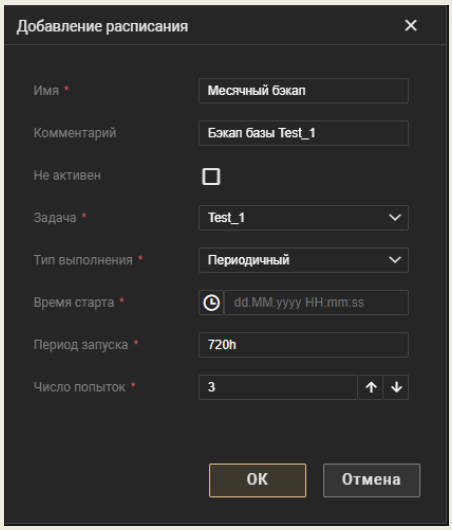
Кроме того, можно настроить выполнение команды создание бэкапа по расписанию.

Запуск задач бекапов по расписанию

Чтобы запускать задачу бэкапа по расписанию надо создать такое расписание. Для этого надо нажать правой кнопкой мыши на раздел **Расписания** панели управления и в появившемся меню выбрать команду **Добавить расписание**.



В результате откроется диалог **Добавление расписания**.



В нем следует задать имя расписания и комментарий к нему, а также ряд настроек. Такими настройками являются:

- **Задача** – надо выбрать из меню задачу бэкапа, которая будет выполняться по этому расписанию;
- **Тип выполнения** – может принимать значения:
  - **Одноразовый** – задача выполняется один раз в указанное время;
  - **Периодический** – задача выполняется многократно с заданным периодом времени после первого выполнения в указанное время;
- **Время старта –** время запуска задачи;
- **Период запуска –** период времени, через который будет повторяться запуск задачи при многоразовом типе выполнения расписания;
- **Число попыток –** число попыток выполнить задачу при неудачном завершении.

Кроме того, у расписания есть флаг **Не активен**. Если он установлен, то расписание игнорируется, и задача не запускается на выполнение.

Для дальнейшего изменения расписания следует выделить его в панели управления. При этом в панели свойств будут доступны для редактирования все перечисленные настройки.

Кроме настроек в панели свойств для расписания выводятся диагностические параметры:

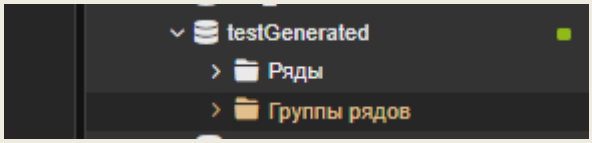
- Состояние задачи;
- Время последнего успешного запуска;
- Длительность выполнения задачи при последнем успешном запуске.

Группы тегов

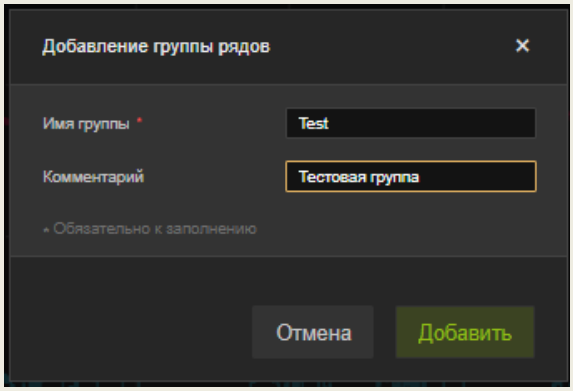
Для структурирования базы тегов в Сервере архивирования предусмотрен механизм группировки. Над собранными в группу тегами можно проводить групповые операции, а также просматривать их архивные значения на групповом тренде.

Создание групп

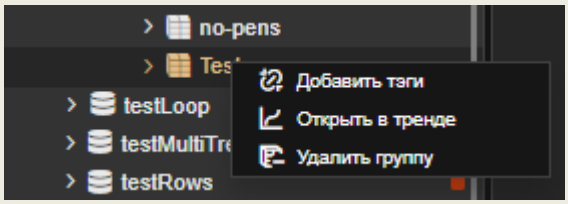
Если открыть базу данных в дереве проекта, то там будут две папки: **Ряды** и **Группы рядов**. В первой из них будут присутствовать все временные ряды (теги) данной базы, а во вторую можно помещать дополнительные папки (**группы**). В группы можно добавлять теги из папки **Ряды**.



Чтобы создать группу следует нажать правую кнопку мыши на папке **Группы рядов** и в появившемся меню выбрать команду **Создать группу**. На экране появится диалог, в котором следует задать имя группы и комментарий к ней.



Чтобы поместить теги из папки Ряды в нужную группу следует поместить сначала ссылки на них в буфер. Для этого надо выделить нужные теги в папке ряды. Доступно выделение группы тегов. Для этого следует при выделении зажать клавишу Ctrl или Shift. После выделения следует нажать правую кнопку мыши на любом выделенном теге и выполнить команду **Копировать ссылку**. После этого следует нажать правую кнопку мыши на имени групп и выполнить команду **Добавить теги**.

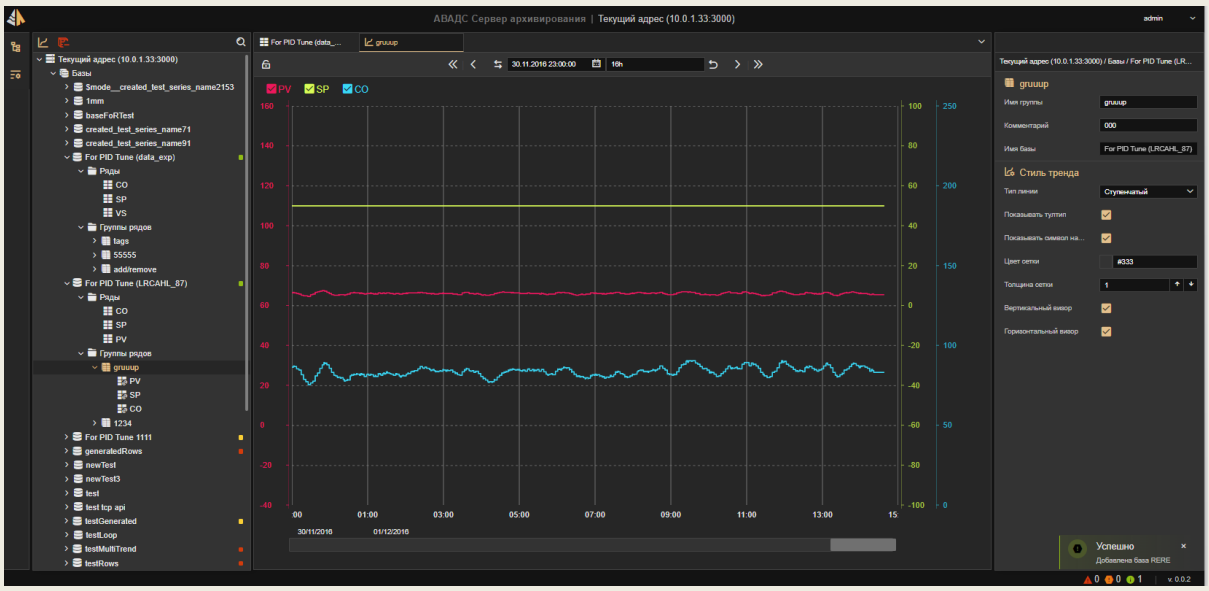


После этого выделенные ранее теги появятся в группе.

Обратите внимание. Каждый тег может присутствовать в неограниченном количестве групп.

Групповые тренды

Теги, помещенные в группу могут просматриваться на едином групповом тренде.



Чтобы перейти в просмотр тегов на тренде следует нажать правую кнопку мыши на названии группы в дереве и выполнить команду **Открыть в тренде**.

Следует иметь в виду, что в группе может быть сколько угодно тегов, но на тренд будут выведены только первые 16 из них.

Взаимодействие с внешними приложениями

Для обмена данными с клиентами Сервер архивирования обеспечивает два протокола:

- **AVADS TCP** – API, обеспечивающее высокоскоростные методы передачи и доступа к данным, а также методы управления сервером, реализованные в рамках TCP/IP стека;
- **AVADS WEB** – WEB-API, реализованное через WEB-socket на базе протокола JSON.

Сервер архивирования может взаимодействовать с клиентами как в рамках одного компьютера, так и по сети. Число клиентов, которые могут одновременно подключиться к Серверу архивирования ограничивается лицензией.

Кроме того, для взаимодействия Сервером архивирования можно использовать протокол OPC UA.

AVADS TCP

Использование протокола AVADS TCP – это высокоскоростной способ взаимодействия клиентов с Сервером Архивирования. Ниже в отдельном разделе приведено подробное описание этого протокола.

Для создания драйвера взаимодействия с Сервером архивирования по протоколу AVADS TCP разработчику понадобятся знания в области работы с TCP/IP стеком.

С сайта компании АВАДС СОФТ можно скачать примеры драйверов, написанные на языках Go, C#, JS, Python.

AVADS WEB

Протокол AVADS WEB используется для взаимодействия **Сервера БД** с **Клиентом администрирования**. Он проще в реализации, чем AVADS TCP, но медленнее в работе. Поэтому его не следует использовать если требуется максимальное быстродействие.

OPC-DB шлюз

Для взаимодействия Сервера архивирования с клиентами по протоколу OPC UA разработана специальная программа – OPC-DB шлюз. Она взаимодействует по протоколу OPC UA с клиентами и транслирует запросы этого протокола в протокол AVADS TCP.

OPC-DB шлюз может быть установлен на том же компьютере, что и Сервер архивирования, а может – на удаленном компьютере. Один OPC-DB шлюз подключенный к Серверу архивирования считается одним клиентом. Но при этом сам OPC-шлюз лицензируется по числу подключённых к нему OPC-клиентов.

Встроенная математическая обработка

Сервер Архивирования может по запросу выполнять математическую обработку сохраненных данных. Для этого в API предусмотрены специальные команды. Запросы на обработку позволяют выполнить следующие действия:

- **Прореживание**. Сокращение передаваемых данных с сохранением качественного представления запрошенной информации.
- **Вычисление параметров**. По архивным данным выбранного тега за заданный интервал вычисляется набор параметров.

Прореживание

Прореживание выполняется для сокращения объема передаваемых данных и уменьшения нагрузки на средства визуализации данных. Примером является отображение на тренде данных за большой интервал времени, когда записей в архиве намного больше чем пикселей в рабочем поле тренда. В этом случае используя прореживание можно получить такую же картинку при резком сокращении времени на запрос и отрисовку данных.

Используется следующий алгоритм прореживания. Весь диапазон времени разбивается на указанное число интервалов. В выборку помещаются первая и последняя запись из каждого интервала, а также записи с максимальным и минимальным значением для каждого интервала.

Для получения данных тега за указанный период времени используются команды 24 и 25 протокола AVADS TCP, а также команды getRange и getFromCP протокола AVADS WEB. Последний параметр команд 24 и 25 протокола AVADS TCP, а также параметр dpi команд протокола AVADS WEB задают число интервалов. Если указать 0, то прореживание не выполняется. Описание протоколов AVADS TCP и AVADS WEB приведено ниже.

Вычисление параметров

Сервер архивирования может вычислять ряд параметров тега по его архивным данным за указанный интервал времени. Такими параметрами являются:

- Максимальное значение;
- Минимальное значение;
- Среднее значение;
- Интеграл;
- Наработка/простой в часах и процентах.

Первые четыре параметра вычисляются для тегов, имеющих один из следующих типов: int, long, dlong, float, dfloat.

Максимальное и минимальное значение получают просто перебором архивных данных. Среднее значение вычисляют как интеграл деленный на интервал времени. Значение интеграла рассчитывают, как площадь фигуры под ломанной, образованной значениями тега, соединенными между собой отрезками.

Наработка вычисляется для тегов, имеющих тип bool. Она определяется как сумма времен интервалов, когда значение тега было true. Для получения наработки в процентах полученную величину делят на величину периода, за который велось вычисление, и умножают на 100.

Для получения расчётных параметров используется команда 31 протокола AVADS TCP или команда getMath протокола AVADS WEB. Описание этих протоколов приведено ниже.

Резервирование

Сервер Архивирования позволяет создавать высоконадежные резервированные системы непрерывного хранения данных. Эта особенность очень важна при автоматизации ответственных и опасных технологических объектов, а также для систем хранения коммерческой информации.

Чтобы обеспечить высокую надежности системы Сервер Архивирования реализует горячее резервирование как серверов, так и носителей данных (дисков).

При резервировании серверов два и более Серверов Архивирования, запущенных на разных компьютерах, работают в горячем резерве. Один имеет статус Мастер, остальные - Резерв. С клиентами взаимодействует Мастер. Он пересылает все получаемые им данные серверам со статусом Резерв. Серверы со статусом Резерв контролируют идентичность своих баз с Мастером и запрашивают у него недостающие данные. При отказе Мастера один из резервных серверов получает статус Мастер.

Для обеспечения надежности хранения данных в рамках одного компьютера Сервер Архивирования поддерживает резервирование дисков. Если эта функция включена, то Сервер Архивирования ведет две одинаковые базы на двух разных дисках. Он контролирует идентичность баз и синхронизирует их. Это позволяет при обеспечении горячей замены дисков не останавливая работы сервера заменить вышедший из строя диск. При аппаратном зеркалировании дисков такая замена недоступна.

…..

Резервирование доступно начиная с версии 1.3 Сервера архивирования. Для более старых версий следует провести обновление продукта. Обратитесь для этого в службу технической поддержки.

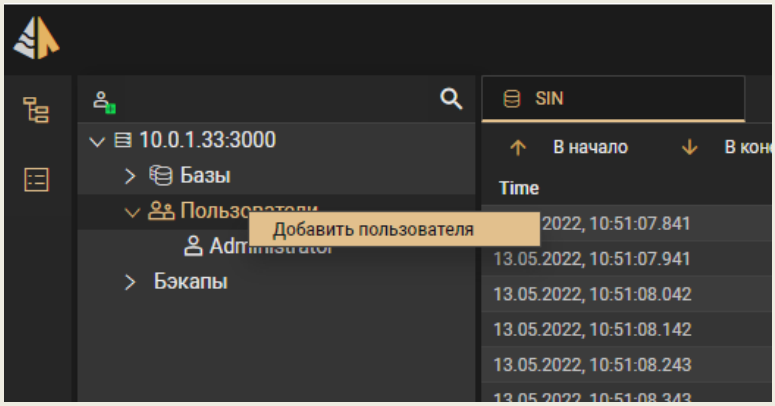
Разграничение прав пользователей

Сервер архивирования имеет систему разграничения прав. Данные о пользователях хранятся в зашифрованном виде.

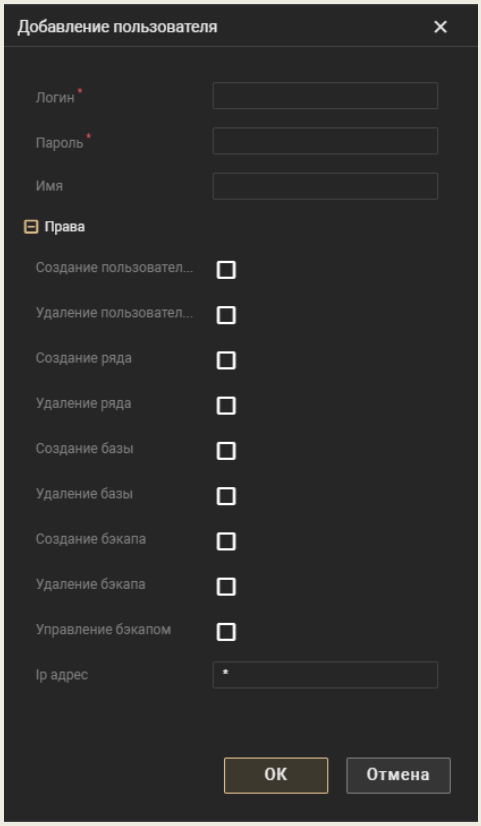
Для получения доступа к Серверу архивирования любой клиент должен ввести логин и пароль. Клиент получает те права, которые установлены для введенного логина.

Создание пользователей и настройка их прав доступа выполняется в клиенте администрирования. Кроме того, управлять пользователями можно с помощью специальных команд по протоколам AVADS TCP и AVADS WEB (описание протоколов приведено ниже).

Чтобы добавить пользователя в клиенте администрирования надо нажать правую кнопку мыши на разделе **Пользователи** панели управления сервером. На экране появится меню в котором надо выбрать пункт **Добавить пользователя**.



После этого на экране появится следующий диалог.



В этом диалоге следует задать логин, пароль, имя и права нового пользователя. Кроме того, в этом диалоге можно задать IP адрес, с которого данному пользователю разрешено подключаться к серверу архивирования. Если оставить в этом поле *, то подключаться можно будет с любого IP адреса.

Если требуется отредактировать свойства пользователя, то надо кликом мыши выбрать его в списке. При этом в панели свойств будут выведены и доступна для редактирования параметры выбранного пользователя.

Для удаления пользователя надо кликнуть правой кнопкой мыши на его имени в панели управления сервером и в появившемся меню выбрать **Удалить пользователя**.

Для управления пользователями можно использовать специальные команды протоколов AVADS TCP и AVADS WEB (см. ниже).

Лицензирование

Сервер архивирования лицензируется по количеству хранимых временных рядов и по числу обслуживаемых клиентов. Для лицензионной защиты сервера архивирования используется программный ключ.

ДЕМО-версии

Чтобы пользователи могли полноценно протестировать сервер архивирования имеется ДЕМО-версия. Она не имеет ограничений на число временных рядов и на число клиентов. Но она работает не более суток подряд. Через сутки она перестает сохранять данные и требует перезапуска. По запросу может быть предоставлена ДЕМО-версия, которая работает более продолжительное время.

ДЕМО-версия собрана для операционных систем Windows и Linux на платформе X86. Они доступны для скачивания с сайта. Для получения ДЕМО-версии под другие операционные системы и аппаратные платформы обращайтесь в службу техподдержки.

Платные лицензии

Платные версии сервера архивирования имеют следующую градацию по числу тегов: 100, 500, 1000, 2500, 10000, 60000, Unlim.

Сервер архивирования может вести неограниченное число баз данных. Однако суммарное число тегов, сохраняемых в этих базах, не может быть больше лицензионного ограничения.

Ограничений на создание новых баз данных и тегов не налагается. Однако при нарушении лицензионного ограничения по числу тегов по вновь созданным тегам данные не будут сохраняться в базы данных.

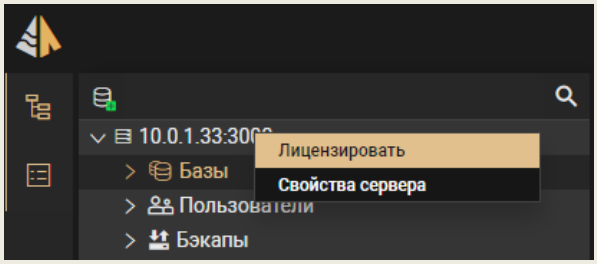
По умолчанию сервер архивирования позволяет подключить 2 клиента. Для подключения большего числа надо купить другую версию сервера архивирования. Существуют следующие градации по числу подключаемых клиентов: 2, 10, 100, Unlim.

Активация лицензии

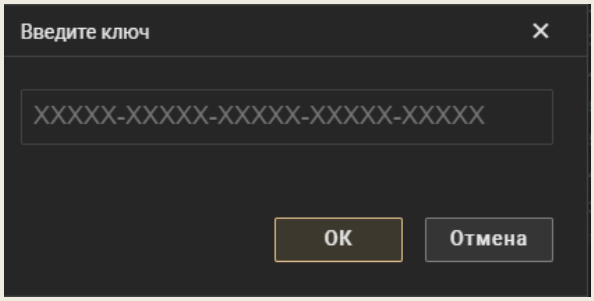
При покупке коммерческой версии Сервера архивирования пользователь получает код активации. Чтобы сервер архивирования начал работать в соответствии с купленной лицензией код активации необходимо ввести в систему.

> ***Внимание**! Для выполнения процедуры активации компьютер, на который устанавливается Сервер архивирования должен быть подключен к Интернет. После завершения активации компьютер можно отключить от Интернета.*

Чтобы ввести код активации надо кликнуть правой кнопкой мыши на адресе сервера в дереве панели управления и в появившемся меню выбрать команду **Активировать**.



После этого на экране появится форма для ввода кода активации.



После ввода ключа активации Сервер архивирования связывается через интернет с сервером компании АВАДС Софт и тот присылает обратно электронный ключ, который привязывает данную копию Сервера архивирования к компьютеру. После успешного завершения этой процедуры на экране появится сообщение об активации продукта.

Описание протокола AVADS TCP

Протокол AVADS TCP обеспечивает API к Серверу архивирования через стек TCP/IP. Взаимодействие с Сервером архивирования выполняется в режиме запрос-ответ. Запрос содержит команду и данные, состав которых определяется командой. В ответ сервер передает статус выполнения команды и данные, соответствующие выполнению посланной команды.

Формат запроса

A [B] [C]

Где
A – код команды (1 байт);
B – длина пакета (4 байта). Необязательный параметр. Его наличие определяется командой **A**;
C – пакет данных длинной **B** байт. Необязательной параметр. Его наличие определяется командой **A**.

Формат ответа

D [E] [F]

Где
D – статус выполнения команды **A** (1 байт);
E – длина пакета (4 байта). Необязательный параметр. Его наличие определяется командой **A**;
F – пакет данных длинны **E**, соответствующий команде **A**. Необязательный параметр.

Общие правила передачи данных

1. Числовые значения укладываются в последовательности big-endian.
2. Строковые переменные передаются в следующем формате:

   L T

   где

   L – (int32) длинна текста в байтах;
   T – массив байтов, содержащий текст.
   Для пустой текстовой строки передается только **L,** равное 0, а массив байт с текстом отсутствует.

Перечень команд

Код команды	Описание команды
1	Создать базу данных
2	Открыть базу данных
3	Получить информацию о базе данных
4	Получить список баз данных
5	Удалить базу данных

Код команды	Описание команды
6	Обновить параметры базы данных
7	Закрыть базу данных
8	Создать временной ряд (тег)
9	Удалить временной ряд (тег)
10	Обновить параметры временного ряда
11	Получить список всех временных рядов базы данных
12	Получить информацию о временном ряде по его имени
13	Получить список пользователей
14	Получить информацию о пользователе
15	Создание пользователя
16	Удаление пользователя
17	Изменение свойств пользователя
18	Получение списка свойств сервера
19	Получение информации о работе сервера
20	Изменение значения свойства сервера
21	Получение меток времени первой и последней записи временного ряда
22	Получение указателя на запись ближайшую к указанному времени
23	Получение заданного количества записей от указанной записи в заданном направлении по времени
24	Получение записей от указанной записи до заданного времени
25	Получение записей из указанного диапазона времени
26	Добавление записи во временной ряд
27	Удаление записи из временного ряда
28	Удаление записей из временного ряда в указанном диапазоне времени
29	Групповое добавление записей
30	Получение значения в указанной точке времени
31	Получение вычисляемых данных в заданном временном диапазоне
32	Групповое добавление записей с подтверждением после сохранения в базу
33	Получить текущее (последнее полученное) значение
34	Получить информацию о временном ряде по его идентификатору
35	Получить список ошибок, возникших при выполнении функции 29
40	Добавить группу
41	Изменить параметры группы
42	Удалить группу

Код команды	Описание команды
43	Получить список групп
44	Получить список тегов, входящих в группу
45	Добавить теги в группу
46	Удалить теги из группы

Перечень параметров команд и ответов

Параметр	Тип	Описание параметра
A	byte	Код команды
B	Int32	Длина пакета запроса в байтах
D	byte	Статус выполнения команды: 0 – ОК, не 0 – ошибка. В последнем случае в ответе посылается строка Err с описанием ошибки;
E	Int32	Длина пакета ответа сервера архивирования в байтах
Err	string	Описание ошибки в ответе сервера архивирования
ID	Int64	Числовой идентификатор базы данных. Задается при открытии базы. Сохраняется в рамках текущей сессии до закрытия соединения с базой. При следующем открытии базы данных ID можно задать другим.
Name	string	Имя базы данных
Comment	string	Комментарий к базе данных/группе
Path	string	Путь сохранения базы данных. Если **Path** равен «» или null, то путь устанавливается по настройкам, заданными «по умолчанию»
FsType	string	Тип хранения базы данных: • fs – база данных пишется единым файлом; • fs_mp – база данных пишется файлами по 1 гб; • mem_fs – база данных создается в оперативной памяти.
DbSize	string	Ограничение на размер базы данных. Задается в виде **<число><размерность>**. Например, • 100gb – 100 гигабайт; • 200mb – 200 мегабайт; • 800kb – 800 килобайт.
AutoAddS	bool	Автоматическое создание временных рядов. Если равен **true** и пришла команда на запись данных по отсутствующему в базе временному ряду, то этот ряд будет автоматически создан.
AutoSave	bool	Если равен **true**, то включается автосохранение буферов из оперативной памяти.

Параметр	Тип	Описание параметра
Duration	string	Время простоя – максимальный интервал времени от последней записи в буфер, когда автосохранение не выполняется. Задается в виде **<число><размерность>**. Например, • 100ms – 100 миллисекунд; • 10s – 10 секунд.
Interval	string	Период автосохранения буферов из оперативной памяти. Задается в виде **<число><размерность>**. Например, • 10s – 10 секунд; • 5m – 5 минут; • 1h – 1 час.
Cycle	byte	Тип зацикливания базы данных: • 0 – наследуется с верхнего уровня иерархии (для базы от сервера, для тега от базы данных) «по умолчанию»; • 1 – зацикливание не включено; • 2 – включено зацикливание по времени (глубине) хранения; • 3 – включено зацикливание по размеру файла базы данных;
Lt	string	Период хранения данных (строка) в формате «DdHhMm», где D – число дней (целое число); H – число часов (целое число); M – число минут (целое число). Несколько примеров: • Два года – «730d»; • Полтора часа – «1h30m»; • Полтора дня – «1d12h»;
Status	Int64	Статус базы данных. Каждый бит статуса передает информации о активации соответствующего состояния базы данных: • 1-й бит = 1 – имеются активные подключения к базе; • 2-й бит = 1 – база повреждена и не пригодна к работе; • 3-й бит = 1 – база занята системным процессом, никакие операции невозможны.
N	Int64	Число баз данных/записей/временных рядов/пользователей
Tid	Int64	Числовой идентификатор временного ряда (тега)
Class	byte	Класс данных: 0 – атомарные данные, 1 - blob
Tname	string	Имя временного ряда (тега)
Type	Int64	Тип значений временного ряда (тега): BOOL 0 SINT 1 INT 2 DINT 3 LINT 4 USINT 5 UINT 6 UDINT 7 ULINT 8

Параметр	Тип	Описание параметра
		REAL 9 LREAL 10 TIME 11 LTIME 12 DATE 13 LDATE 14 TIME_OF_DAY или TOD 15 LTIME_OF_DAY или LTOD 16 DATE_AND_TIME или DT 17 LDATE_AND_TIME или LDT 18 STRING 19 WSTRING 20 CHAR 21 WCHAR 22 BYTE 23 WORD 24 DWORD 25 LWORD 26 SANY 27 BANY 28
Tcomment	string	Комментарий к временному ряду
Ttime	byte	Тип представления метки времени: 0 – с точностью 1 нс, 1 – с точностью 1 мс
T1	Int64	Метка времени первой записи ряда / начало временного диапазона
Tend	Int64	Метка времени последней записи ряда / конец временного диапазона
Count	Int64	Число записей во временном ряду
AD1	string	Указатель на первую запись временного ряда/выборки
ADend	string	Указатель на последнюю запись временного ряда/выборки
AD	string	Указатель на запись в базе данных
Cont	bool	наличие не переданных записей: **true** – есть не переданные записи, **false** – все записи переданы
Time	Int64	Время/временная метка
Direction	byte	Направление считывания записей из базы данных: 1 – вперед по времени, 2 – назад по времени
Np	Int16	Число интервалов, на которые делится диапазон времени при прореживании. Для каждого интервала передаются 4 записи: первая, последняя, с максимальным и с минимальным значение тега. Если **Np**=0, то прореживание при запросе данных не выполняется
Value	byte[8]	Значение тега, сохраненное в записи/значение параметра
Q	uint32	Признак качества значения тега в записи
Login	string	Логин пользователя
Uname	string	Имя пользователя
Pass	string	Пароль пользователя
Ack	bool	Признак блокирования пользователя (если true, то блокирован)
PaddU	bool	Право добавлять и редактировать пользователей
PdelU	bool	Право удалять пользователей

Параметр	Тип	Описание параметра
PaddDB	bool	Право добавлять и редактировать базы данных
PdelDB	bool	Право удалять базы данных
PaddTS	bool	Право добавлять и редактировать временные ряды
PdelTS	bool	Право удалять временные ряды
Pbackup	bool	Право выполнять бэкап из баз данных
IP	string	Расширенный IP-адрес для подключения (если задан, то накладывает ограничение на подключение)
Key	string	Название настройки сервера
address	string	Адрес для прослушивания TCP/IP подключения. Если значение "" то слушает на всех доступных адресах
port	Int64	Порт для TCP/IP подключения
db_path	string	Путь к месту расположения баз данных по умолчанию
cpu	Int64	Разрешенное для использования количество ядер процессора
Read_ceche_size	Int64	Размер кэша на чтение в блоках. Минимальный размер 100
AddWEB	string	Адрес для прослушивания WEB подключения. Если значение "" то слушает на всех доступных адресах
PortWEB	Int64	Порт для WEB подключения
Access	bool	доступность WEB-API/WEB-клиента. Если **true**, то доступен
Timeout	string	Ограничение времени выполнения команд запроса данных. Если за указанный период команда не закончит выполняться, то она прерывается. Задается в виде **<число><размерность>**. Например, • 10s – 10 секунд; • 5m – 5 минут; • 1h – 1 час.
Gname	string	Имя группы
idGroup	string	Идентификатор группы
Lid	string	Идентификатор ссылки на тег

Подключение к Серверу архивирования

Чтобы начать взаимодействовать с Сервером архивирования надо сначала к нему подключиться. Для этого используются две команды. Эти команды имеют коды 0 и 1. С помощью первой команды клиент передает Серверу архивирования логин, с которым он будет подключаться. В ответ сервер присылает клиенту параметры шифрования пароля. После этого с помощью второй командой клиент должен послать Серверу архивирования зашифрованный пароль. Если проверка пароля прошла успешно, то Сервер архивирования пришлет в ответ ключ сессии.

> ***Внимание!*** *Не стоит путать коды команд подключения с кодами команд перечисленными выше. Последние воспринимаются Сервером архивирования только после подключения к нему клиента.*

Среди команд, которые Сервер архивирования воспринимает до подключения имеются еще две: 2 – восстановить сессию, 254 – версия протокола.

Описание команд подключения

Команда передачи логина имеет следующий формат:

A B Login

где
A – (byte) код команды. Для команды передачи логина равен 0;
B – (int32) длина пакета команды;
Login – (string) логин пользователя.

Ответ Сервера архивирования:

D E [Solt Key] [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;
E – (int32) длина пакета ответа;
если D=0 то в ответ присылаются параметры шифрования:
Solt – (string) соль;
Key – (string) ключ
если D отличен от 0, то в ответ присылается:
Err – (string) описание ошибки.

Команда передачи пароля выглядит следующим образом:

A B PassHash

где
A – (byte) код команды. Для команды передачи пароля равен 1;
B – (int32) длина пакета команды;
PassHash – (string) зашифрованный пароль. Этот параметр должен формироваться по формуле **md5(md5(Pass+Salt)+Key)** и передаваться в виде строки каждые 2 символа которой передают значение соответствующего байта сформированного значения в шестнадцатеричном формате.

Ответ Сервера архивирования:

D E [sessionUid] [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;
E – (int32) длина пакета ответа;
если D=0 то в ответ присылаются параметры шифрования:
sessionUid – (string) ключ сессии;
если D отличен от 0, то в ответ присылается:
Err – (string) описание ошибки.

Восстановление взаимодействия с сервером

С помощью ключа сессии можно восстановить взаимодействие с Сервером архивирования после кратковременном нарушении связи.

Команда восстановления сессии выглядит следующим образом:

A B sessionUid

где
A – (byte) код команды. Для команды восстановления сессии равен 2;
B – (int32) длина пакета команды;

sessionUid – (string) ключ сессии.

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;
E – (int32) длина пакета ответа;
Err – (string) описание ошибки. Посылается если D отличен от 0.

Версия протокола

До подключения к Серверу архивирования можно запросить у него версию протокола взаимодействия. Для этого используется команда, имеющая следующий формат:

A

где
A – (byte) код команды. Для команды запроса версии протокола равен 254;

Ответ Сервера архивирования:

D E [Version] [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;
E – (int32) длина пакета ответа;
если D=0 то в ответ присылаются параметры шифрования:
Version – (byte) текущая версия протокола;
если D отличен от 0, то в ответ присылается:
Err – (string) описание ошибки.

Работа с базами данных

Для управления базами данных используются следующие команды API сервера архивирования.

Открытие базы данных

Команда открытия базы данных имеет следующий формат:

A B ID Name

где
A – (byte) код команды. Для команды открытия базы данных равен 2;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
Name – (string) имя базы данных.

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;
E – (int32) длина пакета ответа;
Err – (string) описание ошибки. Посылается если D отличен от 0.

Создание базы данных

Для создания базы данных используется следующая команда:

A B Name Comment Path FsType DbSize Cycle Lt AutoAddS AutoSave Duration Interval

где
A – (byte) код команды. Код команды создания базы данных равен 1;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;
Comment – (string) комментарий к базе данных;
Path – (string) путь, куда сохраняется база данных. Если **Path** равен «» или null, то путь устанавливается в соответствии с настройками, заданными «по умолчанию»;
FsType – (string) тип размещения базы данных;
DbSize – (string) ограничение на размер базы данных;
Cycle – (byte) тип зацикливания базы данных;
Lt – (string) глубина хранения данных;
AutoAddS – (bool) разрешение на автоматическое добавление временных рядов при поступлении данных по отсутствующим в базе временным рядам;
AutoSave – (bool) признак включения автосохранения буферов из оперативной памяти на диск;
Duration – (string) время от последней записи, превышение которого делает обязательным сохранение буфера данного временного ряда из памяти на диск при автосохранении;
Interval – (string) период выполнения автосохранения буферов из оперативной памяти на диск.

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;
E – (int32) длина пакета ответа;
Err – (string) описание ошибки. Посылается если D отличен от 0.

Удаление базы данных

При выполнении этой команды выполняется удаление базы данных из списка баз сервера архивирования и удаляются файлы базы данных по месту их размещения.

Для удаления базы данных используется следующая команда:

A B Name

где
A – (byte) код команды. Для команды удаления базы данных равен 5;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;
E – (int32) длина пакета ответа;
Err – (string) описание ошибки. Посылается если D отличен от 0.

Получить информацию о базе данных

Для запроса информации о базе данных используется следующая команда:

A B Name

где

A – (byte) код команды. Для команды запроса информации о базе данных равен 3;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;

Ответ Сервера архивирования:

D E [Name Path Comment Status Cycle Lt DbSize FsType AutoAddS AutoSave Duration Interval] [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D=0, то передаются следующие параметры:
Name – (string) имя базы данных;
Path – (string) путь, куда сохраняется база данных. Если **Path** равен «» или null, то путь устанавливается в соответствии с настройками, заданными «по умолчанию»;
Comment – (string) комментарий к базе данных;
Status – (int64) статус базы данных. Каждый бит статуса передает информации о активации соответствующего состояния базы данных;
Cycle – (byte) тип зацикливания базы данных;
Lt – (string) глубина хранения данных;
DbSize – (string) ограничение на размер базы данных;
FsType – (string) тип размещения базы данных;
AutoAddS – (bool) разрешение на автоматическое добавление временных рядов при поступлении данных по отсутствующим в базе временным рядам;
AutoSave – (bool) признак включения автосохранения буферов из оперативной памяти на диск;
Duration – (string) время от последней записи, превышение которого делает обязательным сохранение буфера данного временного ряда из памяти на диск при автосохранении;
Interval – (string) период выполнения автосохранения буферов из оперативной памяти на диск.

Если D отличен от 0, то передается только один параметр:

Err – (string) описание ошибки.

Получить список баз данных

Для запроса списка базе данных используется команда, содержащая только код:

A

где

A – (byte) код команды. Для команды запроса списка баз данных равен 4;

Ответ Сервера архивирования:

D E [N Name(1) Path(1) Comment(1) Status(1) Cycle(1) Lt(1) DbSize(1) FsType(1) AutoAddS(1) AutoSave(1) Duration(1) Interval(1) … Name(N) Path(N) Comment(N) Status(N) Cycle(N) Lt (N) DbSize(N) FsType(N) AutoAddS(N) AutoSave(N) Duration(N) Interval(N)] [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D=0, то передаются следующие параметры:
N – (int64) число баз данных, обслуживаемых сервером архива;

Далее для каждой базы данных передаются:
Name(i) – (string) имя i-й базы данных;
Path(i) – (string) путь, куда сохраняется i-я база данных. Если **Path** равен «» или null, то путь устанавливается в соответствии с настройками, заданными «по умолчанию»;
Comment(i) – (string) комментарий к i-й базе данных;

Status(i) – (int64) статус i-й базы данных. Каждый бит статуса передает информации о активации соответствующего состояния базы данных;
Cycle(i) – (byte) тип зацикливания i-й базы данных;
Lt (i) – (string) глубина хранения данных i-й базы данных;
DbSize(i) – (string) ограничение на размер i-й базы данных;
FsType(i) – (string) тип размещения i-й базы данных;
AutoAddS(i) – (bool) разрешение на автоматическое добавление временных рядов при поступлении данных по отсутствующим в i-й базе временным рядам;
AutoSave(i) – (bool) признак включения автосохранения буферов из оперативной памяти на диск для i-й базы данных;
Duration(i) – (string) время от последней записи, превышение которого делает обязательным сохранение буфера данного временного ряда из памяти на диск при автосохранении для i-й базы данных;
Interval(i) – (string) период выполнения автосохранения буферов из оперативной памяти на диск для i-й базы данных.

Если D отличен от 0, то передается только один параметр:
Err – (string) описание ошибки.

Обновление параметров базы данных

Для обновления параметров базы данных используется следующая команда:

A B Name NewName Comment Path Cycle Lt DbSize FsType AutoAddS AutoSave Duration Interval

где
A – (byte) код команды. Для команды обновления параметров базы данных равен 6;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;
NewName – (string) новое имя базы данных (если имя не менялось, то равно Name)
Path – (string) путь, куда сохраняется база данных. Если **Path** равен «» или null, то путь устанавливается в соответствии с настройками, заданными «по умолчанию»;
Comment – (string) комментарий к базе данных;
Cycle – (byte) тип зацикливания базы данных;
Lt – (string) глубина хранения данных;
DbSize – (string) ограничение на размер базы данных;
FsType – (string) тип размещения базы данных;
AutoAddS – (bool) разрешение на автоматическое добавление временных рядов при поступлении данных по отсутствующим в базе временным рядам;
AutoSave – (bool) признак включения автосохранения буферов из оперативной памяти на диск;
Duration – (string) время от последней записи, превышение которого делает обязательным сохранение буфера данного временного ряда из памяти на диск при автосохранении;
Interval – (string) период выполнения автосохранения буферов из оперативной памяти на диск.

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;
E – (int32) длина пакета ответа;
Err – (string) описание ошибки. Посылается если D отличен от 0.

Закрытие базы данных

Для закрытия соединения с базой данных используется следующая команда:

A B ID

где
A – (byte) код команды.Для команды закрытия базы данных равен 7;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;
E – (int32) длина пакета ответа;
Err – (string) описание ошибки. Посылается если D отличен от 0.

Работа с временными рядами (тегами)

Создание временного ряда

Для создания временного ряда используется следующая команда:

A B Name Tid Tname Type Ttime Tcomment Cycle Lt

где
A – (byte) код команды. Для команды создания временного ряда равен 8;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;
Tid – (int64) числовой идентификатор временного ряда;
Tname – (string) имя временного ряда;
Type – (int64) тип значений временного ряда;
Ttime – (byte) тип представления времени: 0 – с точностью 1 нс, 1 – с точностью 1 мс;
Tcomment – (string) комментарий к временному ряду;
Cycle – (byte) тип зацикливания базы данных;
Lt – (string) глубина хранения данных;

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;
E – (int32) длина пакета ответа;
Err – (string) описание ошибки. Посылается если D отличен от 0.

Удаление временного ряда

Для удаления временного ряда используется команда:

A B Name Tid

где
A – (byte) код команды. Для команды удаления временного ряда равен 9;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;
Tid – (int64) числовой идентификатор временного ряда;

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;

E – (int32) длина пакета ответа;
Err – (string) описание ошибки. Посылается если D отличен от 0.

Обновление параметров временного ряда

Для обновления параметров временного ряда используется следующая команда:

A B Name Tid Tname Class Type Ttime Tcomment Cycle Lt

где
A – (byte) код команды. Для команды обновления параметров временного ряда равен 10;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;
Tid – (int64) числовой идентификатор временного ряда;
Tname – (string) имя временного ряда;
Class – (byte) класс данных временного ряда: 0 – атомарные данные, 1 – blob;
Type – (int64) тип значений временного ряда;
Ttime – (byte) тип представления времени: 0 – с точностью 1 нс, 1 – с точностью 1 мс;
Tcomment – (string) комментарий к временному ряду;
Cycle – (byte) тип зацикливания временного ряда;
Lt – (string) глубина хранения данных временного ряда;

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа. Если не равен 0, то ошибка;
E – (int32) длина пакета ответа;
Err – (string) описание ошибки. Посылается если D отличен от 0.

Получение списка временных рядов базы данных

Для получения списка временных рядов базы данных используется следующая команда:

A B Name

где
A – (byte) код команды. Для команды получения списка временных рядов равен 11;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;

Ответ Сервера архивирования:

D E [N Tid(1) Tname(1) Class(1) Type(1) Ttime(1) Tcomment(1) Cycle(1) Lt(1) … Tid(N) Tname(N) Class(N) Type(N) Ttime(N) Tcomment(N) Cycle(N) Lt(N)] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

если **D**=0, то ответ содержит следующие данные:
N – (int64) число временных рядов в базе данных
Tid(i) – (int64) числовой идентификатор i-го временного ряда;
Tname(i) – (string) имя i-го временного ряда;
Class(i) – (byte) класс данных i-го временного ряда: 0 – атомарные данные, 1 – blob;
Type(i) – (int64) тип значений i-го временного ряда;
Ttime(i) – (byte) тип представления времени i-го временного ряда: 0 – с точностью 1 нс, 1 – с точностью 1 мс;
Tcomment(i) – (string) комментарий к i-му временному ряду;
Cycle(i) – (byte) тип зацикливания i-го временного ряда;
Lt(i) – (string) глубина хранения данных i-го временного ряда;

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Запрос информации о временном ряде по имени

Для запроса параметров временного ряда используется команда:

A B Name Tname

где
A – (byte) код команды. Для команды запроса параметров временного ряда равен 12;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;
Tname – (string) имя временного ряда;

Ответ Сервера архивирования:

D E [Tid Tname Class Type Ttime Tcomment Cycle Lt] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

если D=0, то ответ содержит следующие данные:
Tid – (int64) числовой идентификатор временного ряда;
Tname – (string) имя временного ряда;
Class – (byte) класс данных временного ряда: 0 – атомарные данные, 1 – blob;
Type – (int64) тип значений временного ряда;
Ttime – (byte) тип представления времени: 0 – с точностью 1 нс, 1 – с точностью 1 мс;
Tcomment – (string) комментарий к временному ряду;
Cycle – (byte) тип зацикливания базы данных;
Lt – (string) глубина хранения данных;

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Запрос информации о временном ряде по идентификатору

Для запроса параметров временного ряда используется команда:

A B Name Tid

где
A – (byte) код команды. Для команды запроса параметров временного ряда равен 12;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;
Tid – (int64) идентификатор временного ряда;

Ответ Сервера архивирования:

D E [Tid Tname Class Type Ttime Tcomment Cycle Lt] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

если D=0, то ответ содержит следующие данные:
Tid – (int64) числовой идентификатор временного ряда;
Tname – (string) имя временного ряда;
Class – (byte) класс данных временного ряда: 0 – атомарные данные, 1 – blob;
Type – (int64) тип значений временного ряда;
Ttime – (byte) тип представления времени: 0 – с точностью 1 нс, 1 – с точностью 1 мс;

Tcomment – (string) комментарий к временному ряду;
Cycle – (byte) тип зацикливания базы данных;
Lt – (string) глубина хранения данных;

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Работа с данными

Получение меток времени первой и последней записи временного ряда

Чтобы получить метки времени крайних записей (самой старшей и последней) временного ряда используется следующая команда:

A B ID Tid

где
A – (byte) код команды. Для команды запроса временных меток первой и последней записей временного ряда равен 21;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
Tid – (int64) числовой идентификатор временного ряда.

Ответ Сервера архивирования:

D E [T1 Tend Count AD1 ADend] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

если **D**=0, то ответ содержит следующие данные:
T1 – (int64) метка времени первой записи временного ряда;
Tend – (int64) метка времени последней записи временного ряда;
Count – число записей в ряду
AD1 – (string) указатель на первую запись временного ряда в базе данных;
ADend – (string) указатель на последнюю запись временного ряда в базе данных;

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Получение указателя на запись ближайшую к указанному времени

Данная команда позволяет получить указатель на запись во временном ряду, метка времени которой ближе всего находится к указанному времени. Она имеет следующий формат:

A B ID Tid Time

где
A – (byte) код команды. Для команды получения указателя на запись, ближайшую к указанному времени равен 22;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
Tid – (int64) числовой идентификатор временного ряда.
Time – (int64) время.

Ответ Сервера архивирования:

D E [AD] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

если **D**=0, то ответ содержит следующие данные:

AD – (string) указатель на запись, метка времени которой ближе всего **Time**.

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Получение заданного числа записей от указанной записи

Эта команда позволяет сделать выборку заданного числа записей из базы данных начиная с указанной записи. Причем выборка может выполняться как в сторону увеличения времени, так и в сторону уменьшения времени. Для указания направления выборки используется специальный параметр. Формат команды:

A B ID Tid AD N

где
A – (byte) код команды. Для команды считывания заданного числа записей равен 23;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
AD – (string) указатель на первую запись в выборке;
Direction – (byte) направление чтения;
N – (int64) число считываемых записей.

Ответ Сервера архивирования:

D E [AD1 ADend Cont N Time(1) Value(1) Q(1) … Time(N) Value(N) Q(N)] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

если **D**=0, то ответ содержит следующие данные:
AD1 – (string) указатель в базе данных на первую запись в выборке;
ADend – (string) указатель в базе данных на последнюю запись в выборке;
Cont – (bool) наличие не переданных записей: true – есть не переданные записи, false – все данные переданы;
N – (int64) число записей в выборке;
Time(i) – (int64) метка времени i-й записи;
Value(i) – (byte[8] если атомарные данные, или Lblob(int32)+ byte[Lblob] если данные класса blob) значение i-й записи;
Q(i) – (uint32) признак качества i-й записи;

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Получение записей из заданного временного диапазона

Для выполнения этой задачи предусмотрены две команды 25 и 24. Первая из них (25) предназначена непосредственно для выполнения запроса на выдачу записей из указанного временного диапазона.

Использование второй команды (24) обусловлено ограничением на число считываемых за одну команду записей. Наличие такого ограничения связано с защитой сервера от перегрузки при запросе большого количества записей. Это ограничение является параметром команды. Можно устанавливать любое ограничение (но не более 10000) исходя из возможностей сервера (производительность процессора, объем оперативной памяти) и загрузки его по количеству обслуживаемых баз данных и суммарному числу временных рядов в них.

Таким образом команда (24) используется, когда количество записей в указанном временном диапазоне превосходит число допустимых для считывания за одну команду записей. В этом случае при выполнении команды 25 сервер архива возвращает число записей равное заданному ограничению и указатель на последнюю считанную запись. Чтобы дочитать все оставшиеся записи из заданного временного диапазона используется команда 24. С ней в сервер передается указатель на последнюю считанную запись и временной диапазон. Если число оставшихся для считывания записей превосходит лимит, то сервер передаст допустимое число записей и указатель на последнюю считанную запись, а также установит в true параметр **Cont** в ответе, указывающий на наличие не переданных записей. Далее можно опять применять команду 24 пока не будут считаны все записи.

Команда 25 имеет следующий формат:

A B ID Tid Direction N T1 Tend Np

где
A – (byte) код команды. Для команды считывания записей из заданного временного интервала равен 25;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
Tid – (int64) числовой идентификатор временного ряда;
Direction – (byte) направление чтения;
N – (int64) ограничение на число считываемых записей;
T1 – (int64) начало временного диапазона;
Tend – (int64) конец временного диапазона;
Np – (int16) число интервалов, используется для прореживания. Делит заданный диапазон времени на **Np** интервалов. Для каждого интервала передаются 4 записи: первая, последняя, с максимальным значение тега и с минимальным значение тега. Если **Np**=0, то прореживание не выполняется.

Ответ Сервера архивирования:

D E [AD1 ADend Cont N Time(1) Value(1) Q(1) … Time(N) Value(N) Q(N)] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

если **D**=0, то ответ содержит следующие данные:
AD1 – (string) указатель в базе данных на первую запись в выборке;
ADend – (string) указатель в базе данных на последнюю запись в выборке;
Cont – (bool) наличие не переданных записей: **true** – есть не переданные записи, **false** – все записи переданы;
N – (int64) число записей в выборке;
Time(i) – (int64) метка времени i-й записи;
Value(i) – (byte[8] если атомарные данные, или Lblob(int32)+ byte[Lblob] если данные класса blob) значение i-й записи;
Q(i) – (uint32) признак качества i-й записи;

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Команда 24 имеет следующий формат:

A B ID AD Direction N T1 Tend Np

где
A – (byte) код команды. Для команды дочитывания записей из заданного временного интервала равен 24;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
AD – (string) указатель на последнюю запись, считанную предыдущей командой (25 или 24);
Direction – (byte) направление чтения;
N – (int64) ограничение на число считываемых записей;
T1 – (int64) начало временного диапазона;
Tend – (int64) конец временного диапазона;
Np – (int16) число интервалов, используется для прореживания. Делит заданный диапазон времени на **Np** интервалов. Для каждого интервала передаются 4 записи: первая, последняя, с максимальным значение тега и с минимальным значение тега. Если **Np**=0, то прореживание не выполняется.

Ответ Сервера архивирования:

D E [AD1 ADend Cont N Time(1) Value(1) Q(1) … Time(N) Value(N) Q(N)] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

если **D**=0, то ответ содержит следующие данные:
AD1 – (string) указатель в базе данных на первую запись в выборке;
ADend – (string) указатель в базе данных на последнюю запись в выборке;
Cont – (bool) наличие не переданных записей: **true** – есть не переданные записи, **false** – все записи переданы;
N – (int64) число записей в выборке;
Time(i) – (int64) метка времени i-й записи;
Value(i) – (byte[8] если атомарные данные, или Lblob(int32)+ byte[Lblob] если данные класса blob) значение i-й записи;
Q(i) – (uint32) признак качества i-й записи;

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Добавление записи во временной ряд

Команда добавления записи во временной ряд выглядит следующим образом:

A B ID Tid Class Time [Value] [Lblob Vblob] Q

где
A – (byte) код команды. Для добавления записи во временной ряд код равен 26;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
Tid – (int64) числовой идентификатор временного ряда;
Class – (byte) класс данных временного ряда: 0 – атомарные данные, 1 – blob;
Time – (int64) метка времени записи;
Q – (uint32) признак качества записи.

Если поле Class равно 0, то передается:
Value – (byte[8]) значение записи;
Если поле Class равно 1, то передается:
Lblob – (int32) длина данных в теле blob;
Vblob – (byte[Lblob]) тело данных blob;

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа.

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Удаление записи из временного ряда

Команда удаления записи из временного ряда выглядит следующим образом:

A B ID Tid Time

где
A – (byte) код команды. Для команды удаления записи код равен 27;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
Tid – (int64) числовой идентификатор временного ряда;
Time – (int64) метка времени записи.

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа.

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Удаление записей из временного ряда в заданном диапазоне времени

Команда удаления записей из временного ряда в заданном временном диапазоне выглядит следующим образом:

A B ID Tid T1 Tend

где
A – (byte) код команды. Для команды удаления записи во временном диапазоне код равен 28;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
Tid – (int64) числовой идентификатор временного ряда;
T1 – (int64) начало временного диапазона
Tend – (int64) конец временного диапазона

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа.

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Групповое добавление записей

Эта команда позволяет передать серверу архивирования сразу много записей для разных временных рядов. Она имеет следующий формат:

A B ID Tid(1) Class(1) Time(1) [Value(1)] [Lblob(1) Vblob(1)] Q(1) … Tid(N) Class(N) Time(N) [Value(N)] [Lblob(N) Vblob(N)] Q(N)

где
A – (byte) код команды. Для команды группового добавления записей код равен 29;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
N – число записей в пакете;
Tid(i) – (int64) числовой идентификатор временного ряда i-й записи в передаваемом пакете;
Class(i) – (byte) признак класса данных: 0 – атомарные, 1 - blob;
Time(i) – (int64) метка времени i-й записи;

Если поле Class(i) равно 0, то передается:
Value(i) – (byte[8]) значение i-й записи;
Если поле Class(i) равно 1, то передается:
Lblob(i) – (int32) длина данных в теле blob i-й записи;
Vblob(i) – (byte[Lblob(i)]) тело данных blob i-й записи;

Value(i) – (byte[8]) значение i-й записи;
Q(i) – (uint32) признак качества i-й записи.

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа.

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Получение списка ошибок, возникших при выполнении функции 29

Данная команда позволяет получить информацию о выполнении команд группового добавления записей (29) в рамках текущей сессии.

Для каждого клиента, подключенного к Серверу архивирования, создается циклический буфер. В него помещается информация об ошибках, возникающих при добавлении записей групповым способом по команде 29 в рамках текущей сессии. При выполнении команды запроса списка ошибок после передачи информации клиенту буфер, хранящий ошибки очищается. При переполнении буфера новые записи об ошибках замещают самые старые.

Если сессия была прервана, а затем восстановлена с помощью команды восстановления взаимодействия с сервером (команда 2 в рамках начала взаимодействия), буфер с ошибками восстанавливается.

Формат команды:

A B

где
A – (byte) код команды. Для команды получения списка ошибок код равен 35;

B – (int32) длина пакета команды;

Ответ Сервера архивирования:

D E N [Nerr(1) Terr(1) … Nerr(N) Terr(N)] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа.

Если D равен 0, то ответ содержит следующие данные:
N – (int64) число ошибок в буфере;
Если **N** больше 0, то передается:
Nerr(i) – (int64) код ошибки;
Terr(i) – (string) описание ошибки.

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки выполнение данной команды.

Групповое добавление записей с подтверждением после сохранения в базу

Эта команда позволяет передать серверу архивирования сразу много записей для разных временных рядов. Она отличается от команды 29 тем, что ответ сервер посылает только после завершения сохранения записей в базу, а не по получении их. Эта команда имеет следующий формат:

A B ID Tid(1) Class(1) Time(1) [Value(1)] [Lblob(1) Vblob(1)] Q(1) … Tid(N) Class(N) Time(N) [Value(N)] [Lblob(N) Vblob(N)] Q(N)

где
A – (byte) код команды. Для команды группового добавления записей с подтверждением после сохранения код равен 32;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
N – число записей в пакете;
Tid(i) – (int64) числовой идентификатор временного ряда i-й записи в передаваемом пакете;
Class(i) – (byte) признак класса данных: 0 – атомарные, 1 - blob;
Time(i) – (int64) метка времени i-й записи;

Если поле Class(i) равно 0, то передается:
Value(i) – (byte[8]) значение i-й записи;
Если поле Class(i) равно 1, то передается:
Lblob(i) – (int32) длина данных в теле blob i-й записи;
Vblob(i) – (byte[Lblob(i)]) тело данных blob i-й записи;

Value(i) – (byte[8]) значение i-й записи;
Q(i) – (uint32) признак качества i-й записи.

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа.

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Получение значения тега на указанный момент времени

Эта команда позволяет получить значение тега в указанный момент времени. Причем на запрос посылается значение первой записи метка времени которой меньше указанного времени. Она имеет следующий формат:

A B ID Tid Time

где
A – (byte) код команды. Для команды получения значения на указанный момент времени код равен 30;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
Tid – (int64) числовой идентификатор временного ряда;
Time – (int64) время, на которое запрашивается значение.

Ответ Сервера архивирования:

D E [Time Value Q] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа.

Если D равен 0, то ответ содержит следующие данные:
Time – (int64) метка времени найденной записи;
Value – (byte[8] если атомарные данные, или Lblob(int32)+ byte[Lblob] если данные класса blob) значение тега;
Q – (uint32) признак качества.

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Получение последнего полученного значения тега

Эта команда позволяет получить актуальное (последнее полученное) значение тега. Она имеет следующий формат:

A B ID Tid

где
A – (byte) код команды. Для команды получения значения на указанный момент времени код равен 33;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
Tid – (int64) числовой идентификатор временного ряда.

Ответ Сервера архивирования:

D E [Time Value Q] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа.

Если D равен 0, то ответ содержит следующие данные:
Time – (int64) метка времени найденной записи;
Value – (byte[8] если атомарные данные, или Lblob(int32)+ byte[Lblob] если данные класса blob) значение тега;
Q – (uint32) признак качества.

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Получение вычисляемых данных в заданном временном диапазоне

Эта команда позволяет получить следующие вычисляемые по данным временного ряда параметры за заданный интервал времени:

- минимальное значение тега;
- максимальное значение тега;
- интеграл;
- среднее значение – значение интеграла, деленное на длительность интервала;
- наработка - суммарное время, когда значение было не равно 0;
- процент наработки. Вычисляется делением наработки на длительность интервала.

  ***Внимание**! Эта команда применима только к временным рядам с атомарными данными.*

Данная команда имеет следующий формат:

A B ID Tid T1 Tend Alg

где
A – (byte) код команды. Для команды получения вычисляемых данных код равен 31;
B – (int32) длина пакета команды;
ID – (int64) числовой идентификатор базы данных;
Tid – (int64) числовой идентификатор временного ряда;
T1 – (int64) время начала анализируемого интервала;
Tend – (int64) время конца анализируемого интервала;
Alg – (int64) параметр, задающий набор запрашиваемых вычисляемых параметров. Установка в 1 первых четырех битов этого параметра соответствует запросу следующих параметров:

- 1 – минимальное значение тега;
- 2 – максимальное значение тега;
- 3 – интеграл и среднее значение;
- 4 – наработка и процент наработки.

С помощью этой команды можно запросить все вычисляемые параметры или выбрать только те, которые нужны.

Ответ Сервера архивирования:

D E [“min” Value] [“max” Value] [“integral” Value “average” Value] [“work_time” Value “work_time_percent” Value] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D равен 0, то ответ содержит следующие данные:

Набор пар: <**строка**><**значение**>

Строка передает название параметра. За строкой следует значение соответствующего параметра. Число и типы передаваемых параметров определяются параметром **Alg** в запросе.

В ответе могут присутствовать следующие строки, за которыми передаются значения следующих параметров:
min – минимальное значение тега. Тип Value – int64
max – максимальное значение тега. Тип Value – int64
integral – интеграл значения тега. Тип Value – float64
average – среднее значения тега. Тип Value – float64

work_time – наработка. Тип Value – int64
work_time_percent – наработка в процентах. Тип Value – int64

Если **D** отличен от 0, то присылается:

Err – (string) описание ошибки.

Работа с пользователями

Получение списка пользователей

Команда получения списка пользователей выглядит следующим образом:

A

где

A – (byte) код команды. Для Команды запроса списка пользователей код равен 13;

Ответ Сервера архивирования:

D E [N Login(1) Uname(1) Ack(1) PaddU(1) PdelU(1) PaddDB(1) PdelDB(1) PaddTS(1) PdelTS(1) Pbackup(1) IP(1) … Login(N) Uname(N) Ack(N) PaddU(N) PdelU(N) PaddDB(N) PdelDB(N) PaddTS(N) PdelTS(N) Pbackup(N) IP(N)] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D=0, то передаются следующие параметры:

N – (int64) число пользователей;

Далее для каждого пользователя передается:
Login(i) – (string) логин i-го пользователя;
Uname(i) – (string) имя i-го пользователя;
Ack(i) – (bool) признак блокировки i-го пользователя (если true, то пользователь блокирован);
PaddU(i) – (bool) право добавлять и редактировать пользователей у i-го пользователя;
PdelU(i) – (bool) право удалять пользователей у i-го пользователя;;
PaddDB(i) – (bool) право добавлять и редактировать базы данных у i-го пользователя;
PdelDB(i) – (bool) право удалять базы данных у i-го пользователя;
PaddTS(i) – (bool) право добавлять и редактировать временные ряды у i-го пользователя;
PdelTS(i) – (bool) право удалять временные ряды у i-го пользователя;
Pbackup(i) – (bool) право создавать бэкапы из баз данных у i-го пользователя;
IP(i) – (string) расширенный IP-адрес для подключения (если задан, то накладывает ограничение на подключение).

Если D отличен от 0, то передается только один параметр:

Err – (string) описание ошибки.

Получение информации о пользователе

Команда получения информации о пользователе выглядит следующим образом:

A B Login

где
A – (byte) код команды. Для команды запроса информации о пользователе код равен 14;
B – (int32) длина пакета команды;
Login – (string) логин пользователя.

Ответ Сервера архивирования:

D E [Login Uname Ack PaddU PdelU PaddDB PdelDB PaddTS PdelTS Pbackup IP] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D=0, то передаются следующие параметры:
Login – (string) логин пользователя;
Uname – (string) имя пользователя;
Ack – (bool) признак блокировки пользователя (если true, то пользователь блокирован);
PaddU – (bool) право добавлять и редактировать пользователей;
PdelU – (bool) право удалять пользователей;
PaddDB – (bool) право добавлять и редактировать базы данных;
PdelDB – (bool) право удалять базы данных;
PaddTS – (bool) право добавлять и редактировать временные ряды;
PdelTS – (bool) право удалять временные ряды;
Pbackup – (bool) право создавать бэкапы из баз данных;
IP – (string) расширенный IP-адрес для подключения (если задан, то накладывает ограничение на подключение).

Если D отличен от 0, то передается только один параметр:

Err – (string) описание ошибки.

Создание нового пользователя

Команда создания нового пользователя выглядит следующим образом:

A B Login Pass Uname Ack PaddU PdelU PaddDB PdelDB PaddTS PdelTS Pbackup IP

где
A – (byte) код команды. Для команды создания пользователя код равен 15;
B – (int32) длина пакета команды;
Login – (string) логин пользователя;
Pass – (string) пароль пользователя
Uname – (string) имя пользователя;
Ack – (bool) признак блокировки пользователя (если true, то пользователь блокирован);
PaddU – (bool) право добавлять и редактировать пользователей;
PdelU – (bool) право удалять пользователей;
PaddDB – (bool) право добавлять и редактировать базы данных;
PdelDB – (bool) право удалять базы данных;
PaddTS – (bool) право добавлять и редактировать временные ряды;
PdelTS – (bool) право удалять временные ряды;
Pbackup – (bool) право создавать бэкапы из баз данных;
IP – (string) расширенный IP-адрес для подключения (если задан, то накладывает ограничение на подключение).

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D отличен от 0, то передается один параметр:
Err – (string) описание ошибки.

Изменение свойств пользователя

Команда изменения свойств пользователя выглядит следующим образом:

A B Login NewLogin NewPass Uname Ack PaddU PdelU PaddDB PdelDB PaddTS PdelTS Pbackup IP

где
A – (byte) код команды. Для команды изменения свойств пользователя код равен 16;
B – (int32) длина пакета команды;
Login – (string) логин пользователя;
NewLogin – (string) новый логин пользователя;
NewPass – (string) новый пароль пользователя
Uname – (string) имя пользователя;
Ack – (bool) признак блокировки пользователя (если true, то пользователь блокирован);
PaddU – (bool) право добавлять и редактировать пользователей;
PdelU – (bool) право удалять пользователей;
PaddDB – (bool) право добавлять и редактировать базы данных;
PdelDB – (bool) право удалять базы данных;
PaddTS – (bool) право добавлять и редактировать временные ряды;
PdelTS – (bool) право удалять временные ряды;
Pbackup – (bool) право создавать бэкапы из баз данных;
IP – (string) расширенный IP-адрес для подключения (если задан, то накладывает ограничение на подключение).

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа.

Если D отличен от 0, то передается один параметр:

Err – (string) описание ошибки.

Удаление пользователя

Команда изменения свойств пользователя выглядит следующим образом:

A B Login

где
A – (byte) код команды. Для команды удаления пользователя код равен 17;
B – (int32) длина пакета команды;
Login – (string) логин пользователя;

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D отличен от 0, то передается один параметр:

Err – (string) описание ошибки.

Работа с настройками сервера

Получение списка свойств сервера

Команда получения списка свойств сервера имеет следующий формат:

A

Где

A – (byte) код команды. Для команды получения списка свойств сервера код равен 18;

Ответ Сервера архивирования:

D E [address port cpu db_path read_cache_size Cycle Lt AddWEB PortWEB Access] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D равен 0, то передаются следующие параметры:
address – (string) адрес для прослушивания TCP/IP подключения. Если значение "" то слушает на всех доступных адресах;
port – (int64) порт для TCP/IP подключения;
cpu – (int64) разрешенное для использования количество ядер процессора;
db_path – (string) путь к месту расположения баз данных по умолчанию;
read_cache_size – (int64) размер кэша на чтение в условных единицах (блоках). Минимальный размер 100;
Cycle – (byte) тип зацикливания для всех баз (значение по умолчанию для любой базы);
Lt – (string) глубина хранения данных (значение по умолчанию для любой базы);
AddWEB – (string) адрес для прослушивания WEB подключения. Если значение "" то слушает на всех доступных адресах;
PortWEB – (int64) порт для WEB подключения;
Access – (bool) доступность WEB-API/WEB-клиента. Если **true**, то доступен.

Если D отличен от 0, то передается один параметр:

Err – (string) описание ошибки.

Чтение указанного свойства сервера

Команда чтения указанного свойства сервера имеет следующий формат:

A B Key

где
A – (byte) код команды. Для команды запроса свойства сервера код равен 19;
B – (int32) длина пакета команды;
Key – (string) ключевое слово для выбора запрашиваемого параметра. В запросе можно использовать только одно ключевое слово из приведенных ниже:

- **address** – адрес для прослушивания TCP/IP подключения;
- **port** – порт для TCP/IP подключения;
- **cpu** – разрешенное для использования количество ядер процессора;
- **db_path** – путь к месту расположения баз данных по умолчанию;
- **read_cache_size** – размер кэша на чтение;
- **lopping** – параметры зацикливания баз;
- **web_client_api** – параметры WEB-API.

Ответ Сервера архивирования:

D E [параметры] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D равен 0, то передаются следующие параметры:

Параметры – набор параметров, соответствующих ключевому слову **Key** в запросе:

- address:

address – (string) адрес для прослушивания TCP/IP подключения. Если значение "" то слушает на всех доступных адресах

- port:
  port – (int64) порт для TCP/IP подключения;
- cpu:
  cpu – (int64) разрешенное для использования количество ядер процессора;
- db_path:
  db_path – (string) путь к месту расположения баз данных по умолчанию;
- read_cache_size
  read_cache_size – (int64) размер кэша на чтение в условных единицах (блоках). Минимальное значение 100;
- lopping:
  Cycle – (byte) тип зацикливания для всех баз (значение по умолчанию для любой базы);
  Lt – (string) глубина хранения данных;
- web_client_api:
  AddWEB – (string) адрес для прослушивания WEB подключения. Если значение "" то слушает на всех доступных адресах;
  PortWEB – (int64) порт для WEB подключения;
  Access – (bool) доступность WEB-API/WEB-клиента (true доступен).

Если D отличен от 0, то передается один параметр:

Err – (string) описание ошибки.

Изменение значения свойства сервера

Команда изменения значения свойства сервера выглядит следующим образом:

A B Key Параметры

где
A – (byte) код команды. Для команды изменения значения свойства сервера код равен 20;
B – (int32) длина пакета команды;
Key – (string) ключевое слово для выбора запрашиваемого параметра. В запросе можно использовать только одно ключевое слово из приведенных ниже:

- **address** – адрес для прослушивания TCP/IP подключения;
- **port** – порт для TCP/IP подключения;
- **cpu** – разрешенное для использования количество ядер процессора;
- **db_path** – путь к месту расположения баз данных по умолчанию;
- **read_cache_size** – размер кэша на чтение;
- **lopping** – параметры зацикливания баз;
- **web_client_api** – параметры WEB-API.

Параметры – набор параметров, соответствующих ключевому слову **Key**:

- address:
  address – (string) адрес для прослушивания TCP/IP подключения. Если значение "" то слушает на всех доступных адресах
- port:
  port – (int64) порт для TCP/IP подключения;
- cpu:
  cpu – (int64) разрешенное для использования количество ядер процессора;
- db_path:
  db_path – (string) путь к месту расположения баз данных по умолчанию;
- read_cache_size

read_cache_size – (int64) размер кэша на чтение в условных единицах (блоках). Минимальное значение 100;

- lopping:
  Cycle – (byte) тип зацикливания для всех баз (значение по умолчанию для любой базы);
  Lt – (string) глубина хранения данных;
- web_client_api:
  AddWEB – (string) адрес для прослушивания WEB подключения. Если значение "" то слушает на всех доступных адресах;
  PortWEB – (int64) порт для WEB подключения;
  Access – (bool) доступность WEB-API/WEB-клиента (true доступен);
  Timeout – (string) ограничение времени выполнения команд запроса данных. Например, значение «10s» означает, что через 10 секунд запрос будет прерван.

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D отличен от 0, то передается один параметр:

Err – (string) описание ошибки.

Работа с группами

Добавление группы

Команда добавления группы имеет следующий формат:

A B Name Gname Comment

где
A – (byte) код команды. Для команды добавления группы код равен 40;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;
Gname – (string) имя группы;
Comment – (string) комментарий к группе.

Ответ Сервера архивирования:

D E [idGroup] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D=0, то передается:

idGroup – (string) идентификатор группы;

Если D отличен от 0, то передается только один параметр:

Err – (string) описание ошибки.

Изменение параметров группы

Команда изменения параметров группы выглядит следующим образом

A B Name idGroup Gname Comment

где
A – (byte) код команды. Для команды изменения параметров группы код равен 41;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;
idGroup – (string) идентификатор группы;
Gname – (string) имя группы;
Comment – (string) комментарий к группе.

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D отличен от 0, то передается один параметр:

Err – (string) описание ошибки.

Удаление группы

Запрос на сервер для удаления группы выглядит так:

A B Name idGroup

где
A – (byte) код команды. Для команды удаления группы код равен 42;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;
idGroup – (string) идентификатор группы.

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D отличен от 0, то передается один параметр:

Err – (string) описание ошибки.

Получение списка групп

Для получения списка групп следует отправить на сервер следующий запрос:

A B Name

где
A – (byte) код команды. Для команды получения списка групп код равен 43;
B – (int32) длина пакета команды;
Name – (string) имя базы данных.

Ответ Сервера архивирования

D E [N idGroup(1) Gname(1) Comment(1) … idGroup(N) Gname(N) Comment(N)] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D=0, то передается:

N – (int64) число групп;
idGroup(i) – (string) идентификатор i-й группы;
Gname(i) – (string) имя i-й группы;
Comment(i) – (string) комментарий к i-й группе.

Если D отличен от 0, то передается только один параметр:

Err – (string) описание ошибки.

Получение списка тегов группы

Команда запроса списка тегов (рядов) группы имеет следующий формат:

A B Name idGroup

где
A – (byte) код команды. Для команды получения списка тегов группы код равен 44;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;
idGroup – (string) идентификатор группы.

Ответ Сервера архивирования

D E [N Lid(1) Tid(1) Comment(1) … idGroup(N) Gname(N) Comment(N)] [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D=0, то передается:

N – (int64) число тегов в группе;
Lid(i) – (string) идентификатор ссылки на i-й тег группы;
Tid(i) – (int64) идентификатор i-го тега группы;

Если D отличен от 0, то передается только один параметр:

Err – (string) описание ошибки.

Добавление тегов в группу

Команда добавления тегов в группу позволяет одним запросом добавить несколько тегов. Эта команда имеет следующий формат:

A B Name idGroup N Tid(1) … Tid(N)

где
A – (byte) код команды. Для команды добавления тегов в группу код равен 45;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;
idGroup – (string) идентификатор группы;
Tid(i) – (int64) идентификатор i-го добавляемого тега.

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D отличен от 0, то передается один параметр:

Err – (string) описание ошибки.

Удаление тегов из группы

Команда удаления тегов из группы позволяет одним запросом удалить несколько тегов. Эта команда имеет следующий формат:

A B Name idGroup N Tid(1) … Tid(N)

где
A – (byte) код команды. Для команды удаления тегов из группы код равен 45;
B – (int32) длина пакета команды;
Name – (string) имя базы данных;
idGroup – (string) идентификатор группы;
Tid(i) – (int64) идентификатор i-го добавляемого тега.

Ответ Сервера архивирования:

D E [Err]

где
D – (byte) код ответа: если не равен 0, то ошибка;
E – (int32) длина пакета ответа;

Если D отличен от 0, то передается один параметр:

Err – (string) описание ошибки.

Описание протокола AVADS WEB

В рамках протокола AVADS WEB основным способом взаимодействия клиента и сервера является WebSocket.

В приведенном ниже описании все запросы к серверу и его ответы представлены в виде примеров, снабженных комментариями.

Получение ключей аутентификации

Чтобы установить соединение клиента с сервером необходимо получить ключ сессии и произвести аутентификацию пользователя. Для этого существует предварительный этап установки соединения, аутентификации и авторизации. На этом этапе используется протокол HTTP и стандартные GET запросы.

Для получения ключа аутентификации надо послать следующий запрос:

```
GET http://*.*.*.*/login?login={login}
```

где
.*.*. - IP-адрес сервера;
{login} - логин пользователя.

Формат ответа (пример):

```
{
    "status": "ok",     // String статус авторизации (ok или error)
    "salt": "f68022286" // String соль для хеша пароля
    "key": "k68022286"  // String ключ для хеша пароля
}
```

Дальше необходимо послать данные для аутентификации. Для этого надо выполнить следующий запрос:

```
GET http://*.*.*.*/login?login={login}&pass={pass}
```

где
login – логин пользователя
pass – хеш-функция MD5 от (MD5 от (пароль + salt) + key)

Для получения хэш-пароля нужно взять хэш функцию MD5 от конкатенации хэша от реального пароля и salt с ключом key, полученных в первом логин запросе.

```
{
    "status": "ok",            // String статус авторизации (ok,error)
    "session-uid": "f68022286" // String ключ сессии для WebSocket соединения
}
```

При успешной аутентификации, полученный в ответе ключ сессии (uid) используется для установки WebSocket соединения.

Взаимодействие по протоколу WEBSOCKET

Для получения данных серверу посылаются запросы, состоящие из следующих полей:

- cmd – команда запроса;
- id – уникальный идентификатор запроса. При формировании запроса клиент должен сгенерировать это поле. Сервер в ответе на запрос вернет это же значение;
- p – данные запроса.

Формат ответа сервера такой же, но имеет дополнительные поля:

- code – код ошибки, если равен 0 – нормальное завершение, если не 0 – ошибка;
- desc – строка, описание ошибки.

Широковещательные команды

Клиент через WebSocket может принимать широковещательные сообщения сервера (broadcast,). Эти сообщения не требуют ответа. Они имеют свой тип и отличаются по формату. Широковещательные сообщения состоят из двух объектов:

- event – объект, дублирует поля стандартного ответа сервера;
- type – число, соответствует команде из списка широковещательных команд.

С помощью этого механизма запуск некоторых команд или изменений в одном клиенте, передается другим пользователям, подключенным к данному серверу. Это обеспечивает многопользовательский режим.

Список широковещательных команд

```
AddBase      = 0 // добавлена база
RemoveBase   = 1 // удалена база
UpdateBase   = 2 // изменились свойства базы
AddSeries    = 3 // добавлен временной ряд
RemoveSeries = 4 // удален временной ряд
UpdateSeries = 5 // обновлены параметры временного ряда
AddUser      = 6 // добавлен пользователь
RemoveUser   = 7 // удален пользователь
UpdateUser   = 8 // обновлены параметры пользователя
UpdateProps  = 9 // обновлены параметры сервера
BaseStatus   = 10 // изменился статус базы
AddTask      = 11 // добавлена задача бэкапа
RemoveTask   = 12 // удалена задача бэкапа
UpdateTask   = 13 // обновлена задача бэкапа
TaskStatus   = 14 // изменился статус задачи бэкапа
```

Ниже приведен пример широковещательного сообщения о изменении статуса задачи бэкапа.

```
{
    "type": 14,               // сообщение о изменении статуса задачи бэкапа
    "event": {
        "cmd": "",
        "id": "",
        "p": {
            "Id": 2,          // числовой идентификатор задачи бэкапа
            "Status": "Start" // новый статус задачи бэкапа
        },
        "code": 0,
        "desc": ""
    }
}
```

Команды протокола AVADS WEB

Команда	Описание команды
addBase	Создать базу данных
connectBase	Открыть базу данных
getBase	Получить информацию о базе данных
getBaseList	Получить список баз данных

Команда	Описание команды
removeBase	Удалить базу данных
updateBase	Обновить параметры базы данных
disconnectBase	Закрыть базу данных
addSeries	Создать временной ряд (тег)
removeSeries	Удалить временной ряд (тег)
updateSeries	Обновить параметры временного ряда
getSeriesList	Получить список всех временных рядов базы данных
getSeries	Получить информацию о временном ряде по имени
getUserList	Получить список пользователей
getUser	Получить информацию о пользователе
addUser	Создание пользователя
removeUser	Удаление пользователя
updateUser	Изменение свойств пользователя
getPropsList	Получение свойств сервера
ServerInfo	Получение информации о работе сервера
updateProps	Изменение значения свойства сервера
registerLicense	Активация сервера (лицензия)
getBoundary	Получение меток времени первой и последней записи временного ряда
getCP	Получение указателя на запись ближайшую к указанному времени
getFromCP	Получение заданного количества записей от указанной записи
getRangeFromCP	Получение записей от указанной записи до заданного времени
getRange	Получение записей из указанного диапазона времени
addRow	Добавление записи во временной ряд
deletRow	Удаление записей из временного ряда в указанном диапазоне времени
getValue	Получение значения в указанной точке времени
getMath	Получение вычисляемых данных в заданном временном диапазоне
getTaskList	Получение списка задач бэкапа
getTask	Получить информацию по задаче бэкапа
addTask	Добавить задачу бэкапа
updateTask	Изменить параметры задачи бэкапа
removeTask	Удалить задачу бэкапа
stateTask	Управление состоянием задачи бэкапа

Перечень параметров команд и ответов

Поля	Тип	Описание параметра
cmd	строка	Имя команды
id	строка	Уникальный ключ запроса
p	структура	Дополнительные параметры запроса/ответа
code	число	Статус выполнения команды: 0 – ОК, не 0 – ошибка. В последнем случае в поле desc посылается описание ошибки
desc	строка	Описание ошибки в ответе сервера архивирования
BaseList	массив структур	Массив структур BaseInst с описаниями баз данных

BaseInst	структура	Структура, описывающая базу данных. Содержит следующие параметры:
		• name – (строка) имя базы данных
		• comment – (строка) комментарий к базе данных
		• path – (строка) путь сохранения базы данных. Если **Path** равен «» или null, то путь устанавливается по настройкам, заданными «по умолчанию»
		• data_size – (число) число записей в одном блоке записей
		• status – (число) Статус базы данных. Каждый бит статуса передает информации о активации соответствующего состояния базы данных:
		  • 1-й бит = 1 – имеются активные подключения к базе;
		  • 2-й бит = 1 – база повреждена и не пригодна к работе;
		  • 3-й бит = 1 – база занята системным процессом, никакие операции невозможны.
		• looping – (структура) настройки зацикливания
		• db_size – (строка) ограничение на размер базы денных. Например, строка “500gb” означает ограничение на размер азы в 500 гигабайт, а “250mb” – 250 мегабайт
		• fs_type" – (строка) тип хранения базы данных:
		  • fs – база данных пишется единым файлом;
		  • fs_mp – база данных пишется файлами по 1 гб;
		  • mem_fs – база данных создается в ОЗУ.
		• auto_add_series – (bool) если true, то автоматическое создание временного ряда при запросе на сохранение данных по не существующему временному ряду;
		• auto_save – (bool) признак автоматического сохранения буферов из оперативной памяти на диск
		• auto_save_duration – (строка) максимальный интервал времени от последней записи в буфер, когда автосохранение не выполняется. Например,
		  • 100ms – 100 миллисекунд;
		  • 10s – 10 секунд.

Поля	Тип	Описание параметра

		• auto_save_interval – (строка) период автосохранения буферов из оперативной памяти. Например,
		  • 10s – 10 секунд;
		  • 5m – 5 минут;
		  • 1h – 1 час.

looping	структура	Структура описывающая зацикливание. Может являться параметром при работе с серверами, базами и временными рядами. Содержит следующие поля:
		• type – (число) тип зацикливания базы данных:
		  • 0 – наследуется с верхнего уровня иерархии (для базы от сервера, для тега от базы данных) «по умолчанию»;
		  • 1 – зацикливание не включено;
		  • 2 – включено зацикливание по времени (глубине) хранения;
		  • 3 – включено зацикливание по размеру файла базы данных;
		• lt – (строка) период хранения данных (строка) в формате «DdHhMm»
		где
		D – число дней (целое число);
		H – число часов (целое число);
		M – число минут (целое число).
		Несколько примеров:
		Два года – «730d»;
		Полтора часа – «1h30m»;
		Полтора дня – «1d12h»;

baseName	строка	Имя базы данных
SeriesList	массив структур	Массив структур SeriesInst с описаниями временных рядов
SeriesInst	структура	Структура, описывающая временной ряд. Содержит следующие параметры: • name – (строка) имя временного ряда • type – (число) тип значений временного ряда: BOOL 0 SINT 1 INT 2 DINT 3 LINT 4 USINT 5 UINT 6 UDINT 7 ULINT 8 REAL 9 LREAL 10 TIME 11 LTIME 12 DATE 13 LDATE 14 TIME_OF_DAY или TOD 15 LTIME_OF_DAY или LTOD 16

Поля	Тип	Описание параметра

		DATE_AND_TIME или DT 17
		LDATE_AND_TIME или LDT 18
		STRING 19
		WSTRING 20
		CHAR 21
		WCHAR 22
		BYTE 23
		WORD 24
		DWORD 25
		LWORD 26
		SANY 27
		BANY 28
		• id – (число) числовой идентификатор временного ряда
		• comment – (строка) комментарий к временному ряду
		• view_time_mod – (число) точность представления временных меток:
		  • 0 – одна наносекунда
		  • 1 – одна миллисекунда
		• looping – (структура) настройки зацикливания
		• class – (число) класс данных:
		  • 0 – атомарные данные
		  • 1 – blob

seriesName	строка	Имя временного ряда
seriesId	число	Числовой идентификатор временного ряда
UserList	список объектов	Список структур описаний пользователей
UserInst	структура	Структура, описывающая свойства пользователя. Она содержит следующие параметры • id – (число) числовой идентификатор пользователя • login – (строка) логин пользователя • name – (строка) имя для отображения • disabled – (bool) признак временной блокировки пользователя • permission – (структура) права пользователя
permission	структура	Структура, описывающая права пользователя. Она описывает следующие права: • create_user – (bool) создание/изменение пользователя • delete_user – (bool) удаление пользователя • create_series – (bool) создание/изменение временного ряда • delete_series – (bool) удаление временного ряда • create_base – (bool) создание/изменение базы данных • delete_base – (bool) удаление базы данных • createTask – (bool) создание/изменение задач • deleteTask – (bool) удаление задач • createBackup – (bool) создание/изменение бэкапов • controlBackup – (bool) управление бэкапами • deleteBackup – (bool) удаление бэкапа

Поля	Тип	Описание параметра
		• ip – (строка) ip-адрес, с которого разрешено подключение пользователя. Если “*”, то с любого адреса
PropsList	список объектов	Список структур описаний серверов
PropsInst	структура	Структура, описывающая свойства Сервера БД. Она содержит следующие поля: • addres – (строка) IP-адрес для подключения по протоколу AVADS TCP • port – (число) порт для подключения по протоколу AVADS TCP • cpu – (число) число ядер процессора, доступных для Сервера БД • path – (строка) путь размещения сервера на диске • read_cache_size – (число) размер кэша на чтение в условных единицах. • looping – (структура) настройки зацикливания • web_client_api – (структура) параметры подключения по протоколу AVADS WEB

web_client_api	структура	Структура, описывающая параметры для подключения к серверу БД по протоколу AVASD WEB:
		• addres – (строка) IP-адрес для подключения
		• port – (число) порт для подключения
		• enable – (bool) доступность подключения
		• timeout – (строка) Ограничение времени выполнения команд запроса данных. Если за указанный период команда не закончит выполняться, то она прерывается. Задается в виде **<число><размерность>**. Например,
		  • 10s – 10 секунд;
		  • 5m – 5 минут.

ServerInfo	структура	Структура, описывающая лицензионные ограничения и другие свойства Сервера БД. Она содержит следующие поля: • CountOpenedBases – (число) количество открытых баз на сервере • ConnectionsToBases – (число) количество подключений к базам данных • TcpConnectionsCount – (число) количество подключенных клиентов по AVADS TCP • WebApiConnections – (число) количество подключенных клиентов по AVADS WEB • License – (структура) описывающая лицензионные ограничения Сервера БД
License	структура	Структура, описывающая лицензионные ограничения Сервера БД. Она содержит следующие поля: • ValidVer – (строка) версия, на которой работает данная лицензия. Если «*», то работает на всех версиях. Если

Поля	Тип	Описание параметра
		«1.*», то работает на всех релизах версии 1. Если «1.2.*», то работает на всех релизах версии 1.2 • ValidData – (строка) дата, по которую действительна лицензия, если «*», то ограничения нет • ValidConnections – (число) ограничение по числу клиентов • ValidTags – (число) ограничение по числу тегов (временных рядов) • OrganizationOwner – (строка) название организации – владельца лицензии • OrganizationSeller – (строка) название организации, продавшей лицензию
RecT	структура	Структура полей записи данных временного ряда. Она содержит следующие параметры • Time – (число) временная метка записи с точностью до одной миллисекунды • NsOfTime – (число) расширение точности времени до одной наносекунды. Этот параметр передается если для временного ряда задана точность времени до одной наносекунды (параметр **view_time_mod**) • Vflue – (массив байт) значение записи • Q – (число) признак качества данных в соответствии со стандартом OPC
Recs	Массив структур	Массив структур RecT (записи данных), передаваемый на запрос данных
limit	число	Ограничение на число записей, считываемых за один запрос
direct	число	Направление считывания записей из базы данных: 1 – вперед по времени, 2 – назад по времени
dpi	число	Число интервалов, на которые делится диапазон времени при прореживании. Для каждого интервала передаются 4 записи: первая, последняя, с максимальным и с минимальным значение тега. Если **Np**=0, то прореживание при запросе данных не выполняется
HasContinuation	bool	Признак наличия непереданных записей
CP	строка	Указатель на запись данных
StartCP	строка	Указатель на первую запись в выборке
EndCP	строка	Указатель на последнюю запись в выборке
timeMin	число	Время начала диапазона запроса
timeMax	число	Время конца диапазона запроса
algorithm	число	Применяемые алгоритмы математической обработки. Первые 4 бита определяют какие будут выполняться алгоритмы: • 1 бит – поиск минимального значения • 2 бит – поиск максимального значения • 3 бит – вычисление среднего и интеграла • 4 бит – вычисление наработки (время и %)

Поля	Тип	Описание параметра
TaskList	список структур	Список структур TaskItemT описаний задач бэкапа

TaskInstT	структура	Структура, описывающая свойства задачи бэкапа. Она содержит следующие поля:
		• id – (число) числовой идентификатор задачи бэкапа
		• name – (строка) имя задачи бэкапа
		• comment – (строка) комментарий к задаче бэкапа
		• next_start – (число) время последнего выполнения бэкапа
		• laxt_start – (число) время последнего выполнения бэкапа
		• state – (число) состояние задачи бэкапа
		  • 0 – остановлена
		  • 1 – запущена
		  • 2 – поставлена на паузу
		  • 3 – прервана
		  • 4 – завершена
		• typ – (число) тип задачи бэкапа
		  • 0 – задача создания бэкапа;
		  • 1 – задача восстановления бэкапа;
		• parameters – (структура) содержащая дополнительные параметры задачи бэкапа

parameters	структура	Структура, содержащая дополнительные параметры задачи бэкапа:
		• beckupName – (строка) имя файла бэкапа. Может содержать лексемы:
		  • "%y" - год
		  • "%m" - месяц
		  • "%d" - день
		  • "%t" - время
		  • "%base" - название базы
		• beckupPath – (строка) путь к файлу бэкапа. Может содержать лексемы:
		  • "%y" - год
		  • "%m" - месяц
		  • "%d" - день
		  • "%t" - время
		  • "%base" - название базы
		• baseName – (строка) имя базы данных, из которой сохраняется (в которую восстанавливается) бэкап
		• series – список временных рядов. При задании списка временных рядов, сохраняемых в бэкап, можно использовать следующие способы:
		  1. «*» - сохранение всех временных рядов из базы данных;
		  2. a,b,c,… - перечисление через запятую числовых идентификаторов. Например, если записать **3, 7, 15** , то в бэкап будут сохранены временные ряды с числовыми идентификаторами 3, 7 и 15;

Поля	Тип	Описание параметра
		3. m-n – указание диапазона числовых идентификаторов. Например, чтобы указать все временные ряды с 5 по 27 надо написать **5-27**; 4. Комбинация второго и третьего способа. Например, запись **1,5,7-15, 28** означает, что в бэкап будут сохранены времянные ряды с идентификаторами 1, 5, с 7 по 15 и 28. • rangeType – тип временного диапазона формирования бэкапа: • 0 – диапазон отсутствует, выбираются все данные по указанным временным рядам • 1 – диапазон времени от последнего сохранения до текущего времени • 2 – диапазон времени задается параметрами startTime и endTime • startTime – время начала временного диапазона • endTime – время конца временного диапазона • lastTime – время последнего сохранения
db_path	string	Путь к месту расположения баз данных по умолчанию
cpu	Int64	Разрешенное для использования количество ядер процессора
Read_ceche_size	Int64	Размер кэша на чтение в блоках. Минимальный размер 100

Работа с базами данных

Получение списка баз данных

В приведенном примере выполняется запрос списка баз данных. В ответ сервер сообщает, что он обслуживает две базы: Base_1 и Base_2. Они имеют комментарии «Комментарий к Base_1» и «Комментарий к Base_2» соответственно. Обе базы находятся в рабочем состоянии. База Base_1 размещена по адресу “./db/BasesTest/” и она зациклена по глубине хранения в 2 года, а база Base_2 размещена по адресу “./db/” и наследует настройки зацикливания от сервера.

```
{
    "cmd": "getBaseList", // команда получения списка баз
    "id": "32277b53",     // уникальный ключ запроса (создает клиент)
    "p": {}               // дополнительные параметры отсутствуют
}
```

Ответ

```
{
    "cmd": "getBaseList", // команда запроса getBaseList
    "id": "32277b53",     // ключ запроса
    "p": {
        "BaseList": [     // массив описаний баз данных
            {
                "name": "Base_1",                   // имя базы
                "comment": "Комментарий к Base_1", // комментарий к базе
                "path": "./db/BasesTest/",          // путь к базе
                 "data_size": 200, // число записей в одном блоке записей
```

```
                "status": 0,      // статус базы, см. описание параметра status
                "looping": {      // параметры зацикливания базы
                     "type": "2", // тип зацикливания по времени
                     "lt": "730d" // глубина хранения 730 дней = 2 года
                 }
                "db_size": "500gb", // ограничение на размер базы денных
                "fs_type": "fs",    // тип хранения базы данных
                                    //   fs - единым файлом
                                    //   fs_mp - файлами по 1 гб
                                    //   mem_fs - база данных создается в ОЗУ
                "auto_add_series": false, // если true, то автоматическое создание
                                          // временного ряда при запросе на сохранение
                                          // данных по не существующему временному ряду
                "auto_save": false,  // признак автоматического сохранения буферов
                                     // из оперативной памяти на диск
            "auto_save_duration": "10ms", // максимальный интервал времени от
                                          // последней записи в буфер, когда
                                          // автосохранение не выполняется
            "auto_save_interval": "1m"    // период автосохранения буферов из ОЗУ
            },
            {
                "name": "Base_2",                   // имя базы
                "comment": "Комментарий к Base_2", // комментарий к базе
                "path": "./db/",  // путь к базе
                 "data_size": 200, // число записей в одном блоке записей
                "status": 0,      // статус базы, см. описание параметра status
                "looping": {      // параметры зацикливания базы
                     "type": "0", // тип зацикливания по заданному для сервера
                     "lt": ""     // глубина хранения не задана
                "db_size": "300gb", // ограничение на размер базы денных
                "fs_type": "fs",    // тип хранения базы данных
                                    //   fs - единым файлом
                                    //   fs_mp - файлами по 1 гб
                                    //   mem_fs - база данных создается в ОЗУ
                "auto_add_series": false, // если true, то автоматическое создание
                                          // временного ряда при запросе на сохранение
                                          // данных по не существующему временному ряду
                "auto_save": false,  // признак автоматического сохранения буферов
                                     // из оперативной памяти на диск
                "auto_save_duration": "10ms", // максимальный интервал времени от
                                              // последней записи в буфер, когда
                                              // автосохранение не выполняется
                 "auto_save_interval": "1m" // период автосохранения буферов из ОЗУ
                 }
            }

        ]
    },
    "code": 0, // код ошибки (если 0, то нормальное завершение)
    "desc": "" // текстовое описание ошибки если code отличен от 0
}
```

Получение информации о базе данных

В этой команде, в отличии от предыдущей указывается имя конкретной базы. В ответ сервер присылает аналогичную информацию, но только именно для указанной базы.

```
{
    "cmd": "getBase",  // команда запроса информации о базе
    "id": "-653ccaf6", // ключ запроса
    "p": {
        "baseName": "Base_1" // имя базы
    }
}
```

Ответ

```
{
```

```
    "cmd": "getBase",  // команда запроса информации о базе
    "id": "-653ccaf6", // ключ запроса
    "p": {
        "BaseInst": {         // Информация о базе данных
            "name": "Base_1", // Имя
            "comment": " Комментарий к Base_1", // комментарий
            "path": "./db/BasesTest/",          // путь к базе
            "data_size": 200, // число записей в одном блоке записей
            "status": 0,      // статус базы, см. описание параметра status
            "looping": {      // параметры зацикливания базы
                 "type": "2", // тип зацикливания по времени
                 "lt": "730d" // глубина хранения 730 дней
                 }
            "db_size": "500gb", // ограничение на размер базы денных
            "fs_type": "fs",    // тип хранения базы данных
                                //   fs - единым файлом
                                //   fs_mp - файлами по 1 гб;
                                //   mem_fs - база данных создается в ОЗУ.
            "auto_add_series": false, // если true, то автоматическое создание
                                      // временного ряда при запросе на сохранение
                                      // данных по не существующему временному ряду
            "auto_save": false,  // признак автоматического сохранения буферов из
                                 // оперативной памяти на диск
            "auto_save_duration": "10ms", // максимальный интервал времени от
                                          // последней записи в буфер, когда
                                          // автосохранение не выполняется
            "auto_save_interval": "1m"    // период автосохранения буферов из ОЗУ

        }
    },
    "code": 0, // код ошибки (если 0, то нормальное завершение)
    "desc": "" // текстовое описание ошибки если code отличен от 0
}
```

Добавление новой базы данных

В этом примере демонстрируется добавление базы данных с именем «Base_2», с комментарием «Комментарий к Base_2», которая расположена по пути «./db/», параметры зацикливания которой наследуются от сервера

```
{
    "cmd": "addBase", // команда добавления базы данных
    "id": "5ca6ce22", // ключ запроса
    "p": {
        "baseName": "Base_2",     // уникальное имя базы
        "comment":  "Комментарий к Base_2", // текстовый комментарий, необ. поле
        "path": "./db/"           // место расположения базы, необязательное поле
        "looping": {              // параметры зацикливания базы
                 "type": "0",     // тип зацикливания наследуется от сервера
                 "lt": ""         // глубина хранения наследуется от сервера
                 }

    }
}
```

Ответ

```
{
    "cmd": "addBase", // команда добавления базы данных
    "id": "5ca6ce22", // ключ запроса
    "p": {
        "BaseInst": { // структура описания базы
            "name": "Base_2",
            "comment": "Комментарий к Base_2",
            "path": "./db/",
            "data_size": 200, // число записей в одном блоке записей
            "status": 0,
            "looping": {              // параметры зацикливания базы
```

```
                    "type": "0",     // тип зацикливания наследуется от сервера
                    "lt": ""         // глубина хранения наследуется от сервера
                      },
          "db_size": "500gb", // ограничение на размер базы денных
          "fs_type": "fs",    // тип хранения базы данных
                              //   fs - единым файлом
                              //   fs_mp - файлами по 1 гб;
                              //   mem_fs - база данных создается в ОЗУ.
          "auto_add_series": false, // если true, то автоматическое создание
                                    // временного ряда при запросе на сохранение
                                    // данных по не существующему временному ряду
          "auto_save": false,  // признак автоматического сохранения буферов из
                               // оперативной памяти на диск
          "auto_save_duration": "10ms", // максимальный интервал времени от
                                        // последней записи в буфер, когда
                                        // автосохранение не выполняется
          "auto_save_interval": "1m"    // период автосохранения буферов из ОЗУ

       }
    },
    "code": 0, // код ошибки (если 0, то нормальное завершение)
    "desc": "" // текстовое описание ошибки если code отличен от 0
}
```

Удаление базы данных

При выполнении команды удаления базы данных выполняется ее отключение от сервера и физическое удаление файла базы данных. Ниже приведен пример команды удаления базы «Test_2».

```
{
    "cmd": "removeBase", // команда удаления базы данных
    "id": "6e4e2c40",    // ключ запроса
    "p": {
        "baseName": "Test_2" // имя удаляемой базы данных
    }
}
```

Ответ

```
{
    "cmd": "removeBase", // команда удаления базы данных
    "id": "6e4e2c40",    // ключ запроса
    "p": {
        "baseName": "Test_2"
    },
    "code": 0,           // если 0 то удаление прошло нормально
    "desc": ""           // текстовое описание ошибки если code отличен от 0
}
```

Изменение параметров базы данных

В приведенном примере у базы данных «Test_2» меняется имя на «Test_3» и меняется комментарий с «Комментарий к Test_2» на «Новый комментарий».

> ***Внимание!*** *В теле запроса в структуре* ***BaseInst*** *можно ввести только те параметры, которые надо изменить. Однако в ответе придет структура со значениями всех параметров.*

После выполнения команды изменения параметров базы данных ответ будет послан всем клиентам, подключенным к серверу. Это позволит им внести соответствующие изменения в представление информации об измененной базе данных. Поэтому в поле baseName ответа будет присутствовать имя базы данных, которое было до внесения изменений.

```
{
    "cmd": "updateBase", // команда изменения параметров базы данных
```

```
    "id": "23f3c707",    // ключ запроса
    "p": {
        "baseName": "Test_2", // имя редактируемой базы данных
        "BaseInst": {         // параметры базы данных
            "name": "Test_3", // новое имя базы данных
            "comment": "Новый комментарий", // новый комментарий
            "path": "./db/"
            "looping": {      // параметры зацикливания базы
                     "type": "0",
                     "lt": ""
                       },
            "db_size": “500gb”, // ограничение на размер базы денных
            “fs_type”: “fs”,    // тип хранения базы данных
                                //   fs – единым файлом
                                //   fs_mp – файлами по 1 гб;
                                //   mem_fs – база данных создается в ОЗУ.
            “auto_add_series”: false, // если true, то автоматическое создание
                                      // временного ряда при запросе на сохранение
                                      // данных по не существующему временному ряду
            “auto_save”: false,  // признак автоматического сохранения буферов из
                                 // оперативной памяти на диск
            “auto_save_duration”: “10ms”, // максимальный интервал времени от
                                          // последней записи в буфер, когда
                                          // автосохранение не выполняется
            “auto_save_interval”: “1m”    // период автосохранения буферов из ОЗУ
        }
    }
```

Ответ

```
{
    "cmd": "updateBase", // команда изменения updateBase
    "id": "23f3c707",    // ключ запроса
    "p": {
        "BaseInst": {    // параметры базы данных
            "name": "Test_3", // новое имя базы данных
            "comment": "Новый комментарий", // новый комментарий
            "path": "./db/"
            "looping": {      // параметры зацикливания базы
                     "type": "0",
                     "lt": ""
                       },
            "db_size": “500gb”, // ограничение на размер базы денных
            “fs_type”: “fs”,    // тип хранения базы данных
                                //   fs – единым файлом
                                //   fs_mp – файлами по 1 гб;
                                //   mem_fs – база данных создается в ОЗУ.
            “auto_add_series”: false, // если true, то автоматическое создание
                                      // временного ряда при запросе на сохранение
                                      // данных по не существующему временному ряду
            “auto_save”: false,  // признак автоматического сохранения буферов из
                                 // оперативной памяти на диск
            “auto_save_duration”: “10ms”, // максимальный интервал времени от
                                          // последней записи в буфер, когда
                                          // автосохранение не выполняется
            “auto_save_interval”: “1m”    // период автосохранения буферов из ОЗУ
        },
        "baseName": "Test_2", // имя базы данных в которую внесены изменения.
                              // Посылается имя, которое было у базы на момент
                              // подачи команды
    },
    "code": 0,
    "desc": ""
}
```

Внимание! *С помощью данной команды можно изменить все параметры базы данных кроме* ***data_size****. Этот параметр задается при создании базы данных и далее изменен быть не может.*

Открытие базы данных

В приведенном примере выполняется открытие базы с именем «GENERATED».

```
{
     "cmd": "connectBase", // команда открытия базы данных
     "id": "5c87a3cc",     // ключ запроса
     "p": {
            "baseName": "GENERATED" // имя открываемой базы данных
          }
}
```

Ответ:

```
{
    "cmd": "connectBase", // команда открытия базы данных
    "id": "5c87a3cc",     // ключ запроса
    "p": {},
    "code": 0,
    "desc": ""
}
```

Закрытие базы данных

Закрытие базы данных иллюстрируется примером, в котором закрывается база с именем «GENERATED».

```
{
     "cmd": "disconnectBase", // команда закрытия базы данных
     "id":  "-5a01038a",      // ключ запроса
     "p": {
            "baseName": "GENERATED" // имя закрываемой базы данных
          }
}
```

Ответ:

```
{
    "cmd": "disconnectBase", // команда закрытия базы данных
    "id": "-5a01038a",       // ключ запроса
    "p": {},
    "code": 0,
    "desc": ""
}
```

Работа с временными рядам

Получение списка временных рядов

В примере запрашивается список временных рядов базы данных с именем «Test_1». Эта база данных содержит только один временной ряд, имеющий имя «0» и числовой идентификатор 0, пустой комментарий и ненастроенные параметры зацикливания.

```
{
    "cmd": "getSeriesList", // команда получение списка рядов
    "id": "19fa46e9",       // ключ запроса
    "p": {
        "baseName": "Test_1" // имя базы
    }
}
```

Ответ

```
{
    "cmd": "getSeriesList",
    "id": "19fa46e9",
    "p": {
        "SeriesList": [        // массив структур с описанием временных рядов
```

```
            {
                "name": "0",    // имя временного ряда
                "type": 27,     // тип значений временного ряда: SANY
                "id": 0,        // числовой идентификатор временного ряда
                "comment": "", // комментарий к временному ряду
                "view_time_mod": 1, // точность представления временных меток:
                                    //   0 - одна наносекунда
                                    //   1 - одна милисекунда
                "looping": {   // параметры зацикливания
                    "type": "",
                    "lt": ""
                },
                "class": 0     // класс данных:
                               //   0 - атомарные данные
                               //   1 - blob
            }
        ]
    },
    "code": 0,
    "desc": ""
}
```

Получение свойств временного ряда по имени

В примере показан запрос свойств временного ряда с именем «0» из базы данных «Test_1».

```
{
    "cmd": "getSeries",       // команда запроса свойств временного ряда по имени
    "id": "d1eae80",          // ключ запроса
    "p": {                    // параметры запроса
        "baseName": "Test_1", // имя базы данных
        "seriesName": "0"     // имя временного ряда
    }
}
```

Ответ

```
{
    "cmd": "getSeries",       // команда запроса свойств временного ряда по имени
    "id": "d1eae80",          // ключ запроса
    "p": {                    // параметры запроса
        "SeriesInst": {       // структура описания свойств временного ряда
            "name": "0",
            "type": 27,
            "id": 0,
            "comment": "",
            "view_time_mod": 1, // точность представления временных меток:
                                //   0 - одна наносекунда
                                //   1 - одна милисекунда
            "looping": {
                "type": "",
                "lt": ""
              },
                "class": 0     // класс данных:
                               //   0 - атомарные данные
                               //   1 - blob

        }
    },
    "code": 0,
    "desc": ""
}
```

Добавление временного ряда

В примере демонстрируется команда добавления временного ряда с именем «TRAS-785», типом значений REAL и параметрами зацикливания, наследуемыми от базы данных. Добавление выполняется в базу данных «Test_1».

Важно отметить, что при добавлении временного ряда необязательно задавать его имя, числовой идентификатор, комментарий и тип значений. Если любой из этих параметров не задать, то Сервер архивирования присвоит значения этим параметрам и сообщит их в ответе.

```
{
    "cmd": "addSeries", // команда добавления временного ряда
    "id": "70cc2db1",   // ключ запроса
    "p": {              // параметры запроса
        "baseName": "Test_1",   // имя базы
        "SeriesInst": {         // свойства добавляемого временного ряда
            "id": 3,            // числовой идентификатор (необязательное поле)
            "name": "TRAS-785", // имя (необязательное поле)
            "comment": "",      // комментарий (необязательное поле)
            "view_time_mod": 0, // точность представления временных меток:
                                //   0 – одна наносекунда
                                //   1 – одна милисекунда
            "type": 9,          // тип значений REAL (необязательное поле по
                                // умолчанию устанавливается тип SANY)
                "looping": {    // параметры зацикливания
                    "type": 0,
                    "lt": ""
             }
        }
    }
}
```

Ответ

```
{
    "cmd": "addSeries", // команда добавления временного ряда
    "id": "70cc2db1",   // ключ запроса
    "p": {
        "BaseInst": {   // структура описания базы данных
            "name": "Test_1",
            "comment": "Комментарий к Test_1",
            "path": "./db/BasesTest/",
            "status": 0
                "looping": {    // параметры зацикливания
                    "type": "",
                    "lt": ""
        },
        "SeriesInst": {  // структура описания временного ряда
            "name": " TRAS-785",
            "type": 9,
            "id": 3,
            "comment": "",
            "view_time_mod": 0, // точность представления временных меток:
            "looping": {
                "type": "0",
                "lt": ""
             },
                "class": 0      // класс данных:
                                //   0 – атомарные данные
                                //   1 – blob
        }
    },
    "code": 0,
    "desc": ""
}
```

Удаление временного ряда

Для удаления временного ряда из базы надо в соответствующей команде указать имя базы данных, из которой удаляется временной ряд и его числовой идентификатор. В примере ниже из базы данных «Test_1» удаляется временной ряд с числовым идентификатором 3.

```
{
    "cmd": "removeSeries", // команда удаления временного ряда
    "id": "72d7ed91",      // ключ запроса
    "p": {
        "baseName": "Test_1", // имя базы данных
        "seriesId": 3         // числовой идентификатор временного ряда
    }
}
```

Ответ

```
{
    "cmd": "removeSeries", // команда удаления временного ряда
    "id": "72d7ed91",      // ключ запроса
    "p": {},
    "code": 0,
    "desc": ""
}
```

Изменение свойств временного ряда

Для изменения свойств временного ряда надо указать его числовой идентификатор и имя базы данных. В структуре описания свойств временного ряда надо указать редактируемые параметры и их новые значения. В примере в базе данных «Test_1» меняется имя и тип значений временного ряда с числовым идентификатором 3.

```
{
    "cmd": "updateSeries", // команда изменения свойств временного ряда
    "id": "2b7c095a",      // ключ запроса
    "p": {
        "baseName": "Test_1",  // имя базы данных
        "seriesId": 3,         // числовой идентификатор временного ряда
        "SeriesInst": {        // структура свойств временного ряда
            "name": "PRC-251", // имя временного ряда
            "type": 1          // тип значений SINT
        }
    }
}
```

Ответ

```
{
    "cmd": "updateSeries", // команда изменения свойств временного ряда
    "id": "2b7c095a",      // ключ запроса
    "p": {
        "BaseInst": { // структура свойств базы данных
            "name": "Test_1",
            "comment": "Комментарий к Test_1",
            "path": "./db/BasesTest/",
            "status": 0
        },
        "SeriesInst": { // обновленное описание свойств временного ряда
            "name": " PRC-251",
            "type": 1,
            "id": 3,
            "comment": "",
            "view_time_mod": 0,
            "looping": {
                "type": "",
                "lt": ""
            },
            "class": 0     // класс данных:
                           //   0 - атомарные данные
```

```
                            //   1 - blob
        }
    },
    "code": 0,
    "desc": ""
}
```

Работа с пользователями

Структура, описывающая пользователя

```
type UsrT struct {
      id         uint        // числовой идентификатор пользователя
      login      string      // логин
      name       string      // имя для отображения
      disabled   bool        // признак временной блокировки пользователя
      permission PermissionT // структура, описывающая права пользователя
}
```

Структура, описывающая права доступа

```
type PermissionT struct {
      UrlGet        bool   `json:"url_get"`       // доступ на запрос через url
      UrlSet        bool   `json:"url_set"`       // доступ на управление по url
      CreateUser    bool   `json:"create_user"`   // создание/обновление пользов.
      DeleteUser    bool   `json:"delete_user"`   // удаление пользователей
      CreateSeries  bool   `json:"create_series"` // создание/обновление рядов
      DeleteSeries  bool   `json:"delete_series"` // удаление рядов
      CreateBase    bool   `json:"create_base"`   // создание/обновление баз
      DeleteBase    bool   `json:"delete_base"`   // удаление баз
      CreateTask    bool   `json:"create_task"`   // создание/обновление задач
      DeleteTask    bool   `json:"delete_task"`   // удаление задач
      CreateBackup  bool   `json:"create_backup"` // создание/обновление бэкапов
      ControlBackup bool   `json:"control_backup"`// управление бэкапом
      DeleteBackup  bool   `json:"delete_backup"` // удаление бэкапов
      Ip            string `json:"ip"`            // ip-адрес, с которого
                                                  // разрешено подключение
}
```

Запрос списка пользователей

Получение списка пользователей демонстрируется на следующем примере. В нем сервер в ответ на запрос возвращает информацию о 2 пользователях с логинами admin и user1 – это все пользователя, которые описаны для данного сервера.

При получении списка пользователей, в отличии от запроса списков баз данных или временных рядов, сервер присылает не массив структур, а список объектов. Каждый объект – это структура описания пользователя, и он имеет имя идентичное логину данного пользователя.

```
{
  "cmd": "getUserList",  // команда запроса списка пользователей
  "id": "2ec5795c",      // ключ запроса
  "p": {}
}
```

Ответ

```
{
  "cmd": "getUserList", // команда запроса списка пользователей
  "id": "2ec5795c",     // ключ запроса
  "p": {
    "UserList": {
      "admin": {        // структура описания свойств пользователя с логином admin
        "id": 0,
        "login": "admin",
        "name": "",
        "disabled": false,
        "permission": { // описание прав пользователя
```

```
          "create_user": true,
          "delete_user": true,
          "create_series": true,
          "delete_series": true,
          "create_base": true,
          "delete_base": true,
          "createTask": true,
          "deleteTask": true,
          "createBackup": true,
          "controlBackup": true,
          "deleteBackup": true,
          "ip": "*"
        }
      }
    },
      "user1": {        // структура описания свойств пользователя с логином user1
        "id": 1,
        "login": " user1",
        "name": "",
        "disabled": false,
        "permission": {  // описание прав пользователя
          "create_user": false,
          "delete_user": false,
          "create_series": true,
          "delete_series": true,
          "create_base": true,
          "delete_base": true,
          "createTask": false,
          "deleteTask": false,
          "createBackup": false,
          "controlBackup": false,
          "deleteBackup": false,
          "ip": "*"
        }
      }
    }
  },
  "code": 0,
  "desc": ""
}
```

Запрос информации по конкретному пользователю

В примере запрашиваются параметры пользователя, имеющего логин admin

```
{
  "cmd": "getUser",  // команда запроса параметров пользователя по логину
  "id": "-387526fd", // ключ запроса
  "p": {
    "login": "admin"
  }
}
```

Ответ

```
{
  "cmd": "getUser",  // команда запроса параметров пользователя по логину
  "id": "-387526fd", // ключ запроса
  "p": {
    "UserInst": { // структура, описывающая свойства пользователя
      "id": 0,
      "login": "admin",
      "name": "",
      "disabled": false,
      "permission": {
          "create_user": true,
          "delete_user": true,
```

```
          "create_series": true,
          "delete_series": true,
          "create_base": true,
          "delete_base": true,
          "createTask": true,
          "deleteTask": true,
          "createBackup": true,
          "controlBackup": true,
          "deleteBackup": true,
          "ip": "*"
      }
    }
  },
  "code": 0,
  "desc": ""
}
```

Добавление пользователя

При добавлении нового пользователя все поля кроме логина и пароля не обязательны

```
{
  "cmd": "addUser",  // команда добавление пользователя
  "id": "-7d2062b5", // ключ запроса
  "p": {             // параметры пользователя
    "login": "test",
    "password": "test",
    "name": "",
    "disabled": false,
    "permission": {
      "create_user": true,
      "delete_user": true,
      "create_series": true,
      "delete_series": true,
      "create_base": true,
      "delete_base": true,
      "ip": "*"
    }
  }
}
```

Ответ

```
{
  "cmd": "addUser",   // команда добавление пользователя
  "id": "-7d2062b5",  // ключ запроса
  "p": {
    "UserInst": {     // структура параметров пользователя
      "id": 1,        // числовой идентификатор пользователя
      "login": "test",
      "name": "",
      "disabled": false,
      "permission": {
        "create_user": true,
        "delete_user": true,
        "create_series": true,
        "delete_series": true,
        "create_base": true,
        "delete_base": true,
        "ip": "*"
      }
    }
  },
  "code": 0,
  "desc": ""
}
```

Изменения свойств пользователя

В приведенном примере меняется имя для отображения пользователя. Ему присваивается занчение «Оператор1»

```
{
  "cmd": "updateUser", // команда обновления свойств пользователя
  "id": "759fc615",    // ключ запроса
  "p": {
    "login": "test",   // логин пользователя в котором меняют данные
    "UserInst": {      // Структура свойств пользователя
      "login": "test1",
      "name": "Оператор1" // новое значение имени для отображения
    }
  }
}
```

Ответ

```
{
  "cmd": "updateUser", // команда обновления свойств пользователя
  "id": "759fc615",    // ключ запроса
  "p": {
    "UserInst": {
      "id": 1,
      "login": "test1",
      "name": "Оператор1",  // обновленное значение имени для отображения
      "disabled": false,
      "permission": {
        "create_user": true,
        "delete_user": true,
        "create_series": true,
        "delete_series": true,
        "create_base": true,
        "delete_base": true,
        "ip": "*"
      }
    }
  },
  "code": 0,
  "desc": ""
}
```

Удаление пользователя

Для удаления пользователя в параметрах команды надо указать только логин удаляемого пользователя. В приведенном примере удаляется пользователь с логином «test1».

```
{
  "cmd": "removeUser",
  "id": "-55c9cc58",
  "p": {
    "login": "test1"
  },
}
```

Ответ

```
{
  "cmd": "removeUser",
  "id": "-55c9cc58",
  "p": {
     "login": "test1"
  },
  "code": 0,
  "desc": ""
}
```

Работа со свойствами Сервера БД

Получение настроек Сервера БД

```
{
  "cmd": "getPropsList", // команда получения свойств сервера
  "id": "-d57c7cd",      // ключ запроса
  "p": {}
}
```

Ответ

```
{
  "cmd": "getPropsList", // команда получения свойств сервера
  "id": "-d57c7cd",      // ключ запроса
  "p": {
    "PropsList": {
      "address": "127.0.0.1", // IP-адрес для подключения по AVADS TCP
      "port": 7777,           // порт для работы по протоколу AVADS TCP
      "cpu": 1,               // число ядер процессора, доступных для использования
      "db_path": "./db/",     // путь к серверу на диске
      "read_cache_size": 100, // размер кэша на чтение (в условных единицах)
      "looping": {            // параметры зацикливания
        "type": 2,
        "lt": "730d"
      },
      "web_client_api": {     // настройки доступа по протоколу AVADS WEB
        "enable": true,       // разрешение использования
        "port": 7779,         // порт для работы по протоколу AVADS WEB
        "address": "",        // IP-адрес для работы по протоколу AVADS WEB
        "timeout": "10s"      // ограничение времени на выполнение запроса данных
      }
    }
  },
  "code": 0,
  "desc": ""
}
```

Изменение значений свойств Сервера БД

При формировании команды изменения свойств **updateProps** в структуре PropsInst следует указать только те свойства сервера, которые предполагается изменить. В приведенном примере устанавливается новый IP-адрес для доступа к серверу по TCP/IP. В ответе также присылаются только значения измененных свойств.

```
{
  "cmd": "updateProps", // команда изменения свойств сервера
  "id": "48a71c06",     // ключ запроса
  "p": {
    "PropsInst": {
      "address": "127.0.0.1"
    }
  }
}
```

Ответ

```
{
  "cmd": "updateProps", // команда изменения свойств сервера
  "id": "48a71c06",     // ключ запроса
  "p": {
    "PropsInst": {
```

```
      "address": "127.0.0.1"
    }
  },
  "code": 0,
  "desc": ""
}
```

Активация Сервера архивирования

При активации Сервера архивирования надо подключить компьютер, на который он установлен к Интернет и передать с помощью команды registerLicense серийный номер лицензии серверу. Далее Сервер архивирования свяжется с сервисом регистрации и вернет серверу файл, привязывающий данную лицензию к компьютеру.

```
{
  "cmd": "registerLicense", // команда активации лицензии
  "id": "48a71c06",         // ключ запроса
  "p": {
    ""SerialNumber": "XXXXX-XXXXX-XXXXX-XXXXX-XXXXX"
  }
}
```

Ответ

```
{
  "cmd": "registerLicense", // команда активации лицензии
  "id": "48a71c06",         // ключ запроса
  "code": 0,
  "desc": ""
}
```

Получение информации о Сервере архивирования

С помощью этой команды запрашиваются лицензионные ограничения сервера, а также ряд других параметров сервера, описывающих его текущее состояние.

```
{
  "cmd": "getServerInfo", // команда запроса информации о сервере
  "id": "48a71c06",       // ключ запроса
  "p": {}
}
```

Ответ

```
{
  "cmd": "getServerInfo", // команда запроса информации о сервере
  "id": "48a71c06",       // ключ запроса
  "p": {
    "ServerInfo": {       // структура описания состояния сервера
      "CacheMemSize": "",
      "IndexMemSize": "",
      "TotalBaseMemUsage": "0 B",
      "CountOpenedBases": 0,     // число открытых баз данных
      "ConnectionsToBases": 0,   // число подключений к базам данных
      "TcpConnectionsCount": 0,  // число подключений по AVADS TCP
      "WebApiConnections": 1,    // число подключений по AVADS WEB
      "License": {               // параметры лицензии
        "ValidVer": "*",         // купленная версия продукта
        "ValidData": "*",        // дата действия лицензии
        "ValidConnections": "2", // ограничение по числу клиентов
        "ValidTags": "500",      // ограничение по числу временных рядов
        "OrganizationOwner": "ООО ААА-сервис", // название организации, купившей
                                               // эту лицензию
        "OrganizationSeller": "AVADS SOFT"     // название организации, продавшей
                                               // эту лицензию
      }
    }
  },
```

```
  "code": 0,
  "desc": ""
}
```

Работа с данными

Читать данные можно от метки времени или от указателя на запись. Указатель на запись – это строковое значение, позволяющее быстро позиционироваться в базе данных.

Получение значения в заданное время

Если в указанном времени нет значения, то берется первое значение с меньшим временем, если такого нет, то берется первое значение с большим временем.

В примере запрашивается значение временного ряда с числовым идентификатором 233 из базы данных Test_1 на момент времени 2022-07-14 12:30:00. Время задается как целое 8-байтовое число в соответствии с Unix-форматом.

```
{
    "cmd": "getValue", // команда получения значения в заданное время
    "id": "7b7711c8",  // ключ запроса
    "p": {
        "baseName": "Test_1",       // имя базы данных
        "seriesId": 233,            // числовой идентификатор временного ряда
        "time": 1657801800000000000 // заданное время 2022-07-14 12:30:00
    }
}
```

Ответ

```
{
  "cmd": "getValue", // команда получения значения в заданное время
  "id": "7b7711c8",  // ключ запроса
  "p": {
    "Data": {                       // RecT - структура, описывающая запись данных
      "Time": 1657801790000000000, // метка времени записи 2022-07-14 12:29:59
      "NsOfTime": 982354            // число наносекунд. Если точность представления
                                    // для ряда задана до одной миллисекунды, то
                                    // данный параметр не передается
      "Value": 10.5,                // значение записи
      "Q": 192                      // качество данных в соответствии с OPC
    }
  },
  "code": 0,
  "desc": ""
}
```

Получение временных границ ряда

Эта команда позволяет получить временные метки первой и последней записей временного ряда и указатели для быстрого перехода на эти записи. Кроме того, в ответе возвращается количество записей в данном временном ряду и объем памяти занимаемый ими на диске.

```
{
    "cmd": "getBoundary", // команда получения временных границ ряда
    "id": "284ca5ec",     // ключ запроса
    "p": {
        "baseName": "Test_1",
        "seriesId": 233
    }
}
```

Ответ

```
{
    "cmd": "getBoundary", // команда получения временных границ ряда
```

```
    "id": "284ca5ec",     // ключ запроса
    "p": {
        "Data": {
            "Min": 1657497600000000000, // метка времени первой записи
            "Max": 1658102399000000000, // метка времени последней записи
            "Count": 1332,              // количество записей
            "StartCP": "0.36196.0",     // указатель на первую запись
            "EndCP": "0.18098.499",     // указатель на последнюю запись
            "Size" : "4.04 MB"          // занимаемое на диске пространство
        }
    },
    "code": 0,
    "desc": ""
}
```

Получение записей временного ряда в диапазоне времени

Для выполнения этой задачи предусмотрены две команды **getRange** и **getRangeFromCP**. Первая из них предназначена непосредственно для выполнения запроса на выдачу записей из указанного временного диапазона. Вторая используется, когда записей в заданном диапазоне больше, чем указано в параметре **limit** команды **getRange**. Наличие такого ограничения связано с защитой сервера от перегрузки при запросе большого количества записей. Можно устанавливать любое ограничение (но не более 100000) исходя из возможностей сервера (производительность процессора, объем оперативной памяти) и его загрузки.

Таким образом команда **getRangeFromCP** используется, когда число записей в указанном временном диапазоне превосходит предел для считывания за одну команду записей. В этом случае сервер архива при выполнении команды **getRange** возвращает число записей равное заданному ограничению и указатель на последнюю считанную запись, а также устанавливает значение параметра **HasContinuation** сигнализирующего наличие не переданных записей в true. Чтобы дочитать оставшиеся записи в параметрах команды **getRangeFromCP** передается указатель на последнюю считанную запись и временной диапазон. Если число оставшихся для считывания записей превосходит лимит, то сервер передаст допустимое число записей и указатель на последнюю считанную запись, а также снова установит в true параметр **HasContinuation** в ответе. Далее можно опять применять команду **getRangeFromCP** пока не будут считаны все записи.

В примере выполняется запрос записей временного ряда с числовым идентификатором 250 из базы данных Test_1. Время начала диапазона 2022-07-11 00:00:00, время конца диапазона 2022-07-17 23:59:59, ограничение на число считываемых за раз записей – 3 (такое маленькое ограничение в этом и следующем примере установлено только для демонстрации – реальные значения измеряются тысячами и более).

```
{
    "cmd": "getRange", // команда считывания записей в диапазоне времени
    "id": "7b7711c8",  // ключ запроса
    "p": {
        "baseName": "Test_1", // имя базы данных
        "seriesId": 250,      // числовой идентификатор временного ряда
        "timeMin": 1657497600000000000, // время начала диапазона
        "timeMax": 1658102399000000000, // время конца диапазона
        "limit": 3,           // максимальное количество записей, считываемых за
                              // раз. По умолчанию - 3000, максимум - 100000.
        "direct": 1,          // направление чтения (по умолчанию 1):
                              //   1 – вперед по времени
                              //   2 – назад по времени
        “dpi”: 0              // параметр прореживания. Если число считанных
                              // записей больше dpi, то включается прореживание.
                              // Если dpi=0, то прореживание отключено
    }
}
```

Ответ

```
{
    "cmd": "getRange", // команда считывания записей ряда из диапазона времени
    "id": "7b7711c8",  // ключ запроса
    "p": {
        "Data": {
            "Recs": [ // массив структур RecT, содержащий переданные записи
                {
                    "Time": 1657497602000000000,
                    "NsOfTime": 902384,
                    "Value": 98.5,
                    "Q": 192
                },
                {
                    "Time": 1657497607000000000,
                    "NsOfTime": 852384,
                    "Value": 98.7,
                    "Q": 192
                },
                {
                    "Time": 1657497615000000000,
                    "NsOfTime": 375698,
                    "Value": 99.3,
                    "Q": 192
                }
            ],
            "StartCP": "0.36196.0",  // указатель на первую переданную запрись
            "EndCP": "0.36196.2"     // указатель на последнюю переданную запрись
            "HasContinuation": true  // наличие не переданных записей. Если true,
                                     // то для дочитывания надо использовать
                                     // команду getFromCP
        }
    },
    "code": 0,
    "desc": ""
}
```

Получение записей от указателя в заданном диапазоне

В этом примере продолжается чтение записей из временного ряда с числовым индикатором 250, начатое в предыдущем параграфе командой **getRange**. Здесь ограничение на число считываемых за 1 раз записей установлено равным 2. Поэтому в ответе передаются данные только по двум записям.

```
{
    "cmd": "getRangeFromCP", // команда получения записей от указателя в диапазоне
    "id": "-4f8cf243",       // ключ запроса
    "p": {
        "baseName": "Test_1", // имя базы данных
        "seriesId": 250,      // числовой идентификатор временного ряда
        "CP": "0.36196.0",    // указатель на последнюю переданную запрись
        "direct": 1,          // направление чтения (по умолчанию 1):
                              //   1 – вперед по времени
                              //   2 – назад по времени
        "limit": 2,           // максимальное количество записей, считываемых за
                              // 1 раз. По умолчанию - 3000, максимум - 100000.
        "timeMin": 1657497600000000000, // время начала диапазона
        "timeMax": 1658102399000000000, // время конца диапазона
        "dpi": 0              // параметр прореживания. Если число считанных
                              // записей больше dpi, то включается прореживание.
                              // Если dpi=0, то прореживание отключено
    }
}
```

Ответ

```
{
    "cmd": "getRangeFromCP", // команда получения записей от указателя в диапазоне
```

```
    "id": "-4f8cf243",
    "p": {
        "Data": {
            "Recs": [        // массив структур, содержащий переданные записи
                {
                    "Time": 1657497622000000000,
                    "NsOfTime": 902384,
                    "Value": 105.6,
                    "Q": 192
                },
                {
                    "Time": 1657497634000000000,
                    "NsOfTime": 671549,
                    "Value": 103.7,
                    "Q": 192
                }
            ],
            "StartCP": "0.36196.0",  // указатель на первую переданную запрись
            "EndCP":   "0.36196.1",  // указатель на последнюю переданную запрись
            "HasContinuation": true  // наличие не переданных записей. Если true,
                                     // то для дочитывания надо использовать
                                     // команду getFromCP
        }
    },
    "code": 0,
    "desc": ""
}
```

Получение записей временного ряда от указателя

Эта команда позволяет запросить у сервера заданное число записей начиная с указателя на конкретную запись. Число считываемых записей задается параметром **limit**, а направление считыания (вперед или назад по времени) – параметром **direct**.

```
{
    "cmd": "getFromCP", // команда получения записей от указателя
    "id": "71763f78"    // ключ запроса
    "p": {
        "baseName": "Test_1", // имя базы данных
        "seriesId": 250,      // числовой идентификатор временного ряда
        "cp": "0.36196.0",    // указатель на запись
        "direct": 1,          // направление чтения (по умолчанию 1):
                              //   1 - вперед по времени
                              //   2 - назад по времени
        "limit": 2,           // максимальное количество записей, считываемых за
                              // раз. По умолчанию - 3000, максимум - 100000.
    }
}
```

Ответ

```
{
    "cmd": "getFromCP", // команда получения записей от указателя
    "id": "71763f78",   // ключ запроса
    "p": {
        "Data": {
            "Recs": [         // массив структур, содержащий переданные записи
                {
                    "Time": 1657497622000000000,
                    "NsOfTime": 564879,
                    "Value": 105.6,
                    "Q": 192
                },
                {
                    "Time": 1657497634000000000,
                    "NsOfTime": 781254,
                    "Value": 103.7,
                    "Q": 192
```

```
                }
            ],
            "StartCP": "0.36196.0",
            "EndCP": "0.36196.1"
        }
    },
    "code": 0,
    "desc": ""
}
```

Получение указателя на запись по времени

Данная команда возвращает указатель на запись, метка времени которой ближе всего к указанному в команде времени.

```
{
  "cmd": "getCP", // команда получения указателя на запись по времени
  "id": "7663fcdf", // ключ запроса
  "p": {
    "baseName": "Test_1",        // имя базы данных
    "seriesId": 250,             // числовой идентификатор ряда
    "time": 1657497634000000000  // время
    "NsOfTime": 902384,          // дополнение времени: число наносекунд. Задается
                                 // если точность представления времени для ряда
                                 // задана до одной наносекунды

    }
}
```

Ответ

```
{
    "cmd": "getCP",   // команда получения указателя на запись по времени
    "id": "7663fcdf", // ключ запроса
    "p": {
        "CP": "0.36196.0"    // указатель на ближайшую запись к указанному времени
    },
    "code": 0,
    "desc": ""
}
```

Добавление записи во временной ряд

Эта команда позволяет добавлять записи во временные ряды. Важно, чтобы передаваемые командой значения записей соответствовали типам, заданным для редактируемых временных рядов.

```
{
  "cmd": "addRow",  // команда добавления записи во временной ряд
  "id": "3253dccc", // ключ запроса
  "p": {
    "baseName": "Test_1",        // имя базы данных
    "seriesId": 250,             // числовой идентификатор временного ряда
    "Time": 1657497638000000000, // время
    "NsOfTime": 902384,          // дополнение времени: число наносекунд. Задается
                                 // если точность представления времени для ряда
                                 // задана до одной наносекунды
    "Value": 100.8,              // значение, соответствующее типу ряда
    "Q": 192                     // признак качества
  }
}
```

Ответ

```
{
  "cmd": "addRow",  // команда добавления записи во временной ряд
  "id": "3253dccc", // ключ запроса
```

```
  "p": {},
  "code": 0,
  "desc": ""
}
```

Удаление записей из временного ряда

Эта команда позволяет удалять записи из временных рядов. Из ряда будут удалены все записи временные метки которых находятся внутри заданного диапазона. Если надо удалить одну конкретную запись, то начало и конец временного диапазона следует задать равным временной метке этой записи.

```
{
  "cmd": "deletRow", // команда удаления записей из временного ряда
  "id": "3253dccc",  // ключ запроса
  "p": {
    "baseName": "Test_1",        // имя базы данных
    "seriesId": 250,             // числовой идентификатор временного ряда
    "TimeStart": 1657497638000000000, // начало диапазона времени для удаления
                                      // записей
    "NsOfTime": 902384,          // дополнение времени: число наносекунд. Задается
                                 // если точность представления времени для ряда
                                 // задана до одной наносекунды
    "TimeEnd": 1657500385000000000, // конец диапазона времени для удаления
                                    // записей
    "NsOfTime": 563214           // дополнение времени: число наносекунд. Задается
                                 // если точность представления времени для ряда
                                 // задана до одной наносекунды

  }
}
```

Ответ

```
{
  "cmd": "deletRow", // команда удаления записей из временного ряда
  "id": "3253dccc",  // ключ запроса
  "p": {
    "Data": {
            "count": 58   // число удаленных записей
           }
     }
},
  "code": 0,
  "desc": ""
}
```

Математический анализ данных

Сервер архивирования может вычислять ряд параметров для временных рядов за указанный интервал времени. Такими параметрами являются:

- минимальное значение;
- максимальное значение;
- интеграл;
- среднее значение;
- наработка в единицах времени и в процентах.

Чтобы вычисления выполнялись для временного ряда должен быть задан определенный тип. Для типов SANY и BANY вычисления выполняться не будут. Первые 4 параметра вычисляются для чисел с плавающей точкой или целых чисел. Наработка вычисляется для данных типа

bool. Принципиально она может выполняться и для других типов. В этом случае рабочим состоянием будет считаться любое отличие от 0.

Получение значений мат анализа

Данная команда позволяет получить результаты математической обработки данных за указанное время. Используемые алгоритмы задаются в поле "algorithm". Любой из первых 4 битов этого параметра установленный в 1 включает выполнение соответствующих алгоритмов.

В приведенном примере для временного ряда с числовым идентификатором 250 будут вычисляться среднее значение, интеграл, максимальное и минимальное значение. Наработка для этого ряда вычисляться не будет. Соответственно первые 3 бита параметра "algorithm" будут равны 1, а 4-й будет равен 0. Это соответствует значению 7.

```
{
    "cmd": "getMath",  // команда математической обработки данных
    "id": "7b7711c8",  // ключ запроса
    "p": {
        "baseName": " Test_1",   // имя базы
        "seriesId": 250,         // числовой идентификатор временного ряда
        "timeMin": 1657497600000000000, // начало диапазона
        "timeMax": 1658102399000000000, // конец диапазона
        "algorithm": 7           // применяемые алгоритмы. Первые 4 бита определяют
                                 // какие будут выполняться алгоритмы:
                                 //    1 бит - поиск минимального значения
                                 //    2 бит - поиск максимального значения
                                 //    3 бит - вычисление среднего и интеграла
                                 //    4 бит - вычисление наработки (время и %)
    }
}
```

Ответ

```
{
    "cmd": "getMath",  // команда математической обработки данных
    "id": "7b7711c8",  // ключ запроса
    "p": {
        "Data": {
            "integral": 50781.258,       // интеграл
            "integral_average": 87.532,  // среднее значение
            "max": 123.24,               // максимальное значение
            "min": 30.478,               // минимальное значение

// при запросе вычисления наработки в ответе присутствуют следующие поля:
//          "work_time": "",
//          "work_time_percent": ""
         }
    },
    "code": 0,
    "desc": ""
}
```

Работа с бэкапами

Для работы с бэкапами в Сервере архивирования использует две задачи:

- задача создания бэкапа;
- задача восстановления бэкапа.

Эти задачи имеют тип Task и отличаются значением поля typ:

- typ = 0 – задача создания бэкапа,
- typ = 1 – задача восстановления бэкапа.

При настройке таких параметров как backupName (имя файла бэкапа) и backupPath (путь к файлу бэкапа) можно использовать следующие лексемы:

- "%y" - год
- "%m" - месяц
- "%d" - день
- "%t" - время
- "%base" - название базы

Например, если путь сохранения файла бэкапа для базы данных Base_1 года задать следующим образом **‘”%base”/”%y”/”%m”’**, то путь к файлу бэкапа в сентябре 2022 будет **/Base_1/2022/09.**

Текущее состояние задачи бэкапа определяется параметром state. Он может принимать следующие значения:

- 0 – остановлена;
- 1 – запущена;
- 2 – на паузе;
- 3 – прервана;
- 4 – успешно завершена.

Запрос информации о всех задачах

Эта команда позволяет получить информацию о всех задачах создания и восстановления бэкапов, оформленных в Сервере архивирования. В приведенном примере в рамках сервера описана только одна задача восстановления бэкапа.

```
{
    "cmd":: "getTaskList", // команда математической обработки данных
    "id": "8bbb98",        // ключ запроса
    "p": {}
}
```

Ответ

```
{
    "cmd": "getTaskList", // команда математической обработки данных
    "id": "8bbb98",       // ключ запроса
    "p": {
        "TaskList": [     // массив структур TaskItemT с описанием задач
            {
                "id": 2,  // числовой идентификатор задачи бэкапа
                "name": "Restore Test_1",          // имя задачи бэкапа
                "comment": "Комментарий к Test_1", // комментарий к задаче
                "next_start": 1658102399000000000, // время следующего запуска
                "last_start": 1657497600000000000, // время предыдущего бэкапа
                "state": 0,                        // текущее состояние задачи:
                                                   //   0 – остановлена
                                                   //   1 – запущена
                                                   //   2 – на паузе
                                                   //   3 – прервана
                                                   //   4 – успешно завершена
                "typ": 1,                          // тип задачи
                                                   //   0 – создание бэкапа
                                                   //   1 – восстановление бэкапа
                "parameters": {                    // дополнительные параметры
                    "backupName": "%base_%m.back", // имя файла бэкапа состоит из
                                                   // имени базы, номера текущего
                                                   // месяца и расширения ".back"
                    "backupPath": "./%base_%y",    // путь к файлу бэкапа состоит
                                                   // из имени базы и текущего года
                    "baseName": "BaseTest"         // имя базы, из которой
                                                   // делается бэкап
```

```
                }
            }
        ]
    },
    "code": 0,
    "desc": ""
}
```

Запрос информации по конкретной задаче

Запрос выполняется по числовому идентификатору задачи. В приведенном примере запрашивается информация по задаче с идентификатором 1.

```
{
   "cmd": "getTask",  // команда запроса информации о задаче
    "id": "12c7bcce", // ключ запроса
    "p": {
       "id": 1        // числовой идентификатор задачи
    }
}
```

Ответ

```
{
  "cmd": "getTask", // команда запроса информации о задаче
  "id": "12c7bcce", // ключ запроса
  "p": {
    "TaskInst": {            // структура, описывающая параметры задачи
      "id": 1,               // числовой идентификатор
      "name": "Test_1",      // имя задачи
      "comment": "сохранение бэкапа",    // комментарий к задаче
      "next_start": 1658102399000000000, // время следующего выполнения задачи
      "last_start": 1657497600000000000, // время последнего выполнения задачи
      "state": 0,            // состояние задачи
      "typ": 0,              // тип задачи:
                             //   0 - создание бэкапа
                             //   1 - восстановление бэкапа
      "parameters": {        // структура, описывающая дополнительные параметры
        "backupName": "BaseTest.back",
        "backupPath": "/TestBackup", // путь к файлу бэкапа
        "baseName": "BaseTest HDD",
        "series": "0,1,5-9", // перечень временных рядов. Сохраняются временные
                             // ряды с идентификаторами 0, 1 и с 5-го по 9-й
         "rangeType": 1,     // тип указания диапазона времени для бэкапа:
                             //   0 - диапазон не задан. В бэкап выбираются все
                             //       записи по указанным рядам
                             //   1 - в бэкап помещаются записи от времени
                             //       последнего сохранения lastTime до текущего
                             //       момента времени
                             //   2 - диапазон времени задается параметрами
                             //       startTime и endTime
         "startTime": 0,     // начало диапазона времени сохранения бэкапа
         "endTime": 0,       // конец диапазона времени сохранения бэкапа
         "lastTime": 1657497600000000000 // время последнего сохранения бэкапа
      }
    }
  },
  "code": 0,
  "desc": ""
}
```

Добавить задачу

В примере создается задача создания бэкапа всех временных рядов из базы **abc**.

```
{
    "cmd": "addTask",  // команда добавления задачи бэкапа
```

```
    "id": "-4a6d6feb", // ключ запроса
    "p": {
        "taskName": "abc_backup",  // имя задачи
        "typ": 0,                  // тип задачи: создание бэкапа
        "parameters": {
            "baseName": "abc",     // имя базы данных
            "backupName": "abc_b", // имя файла бэкапа
            "backupPath": "./",    // путь к файлу бэкапа
            "series": "*",         // выбираются все временные ряды
            "rangeType": 0         // диапазон времени не задан. Сохраняются все
                                   // записи выбранных временных рядов
        }
    }
}
```

Ответ

```
{
    "cmd": "addTask",  // команда добавления задачи бэкапа
    "id": "-4a6d6feb", // ключ запроса
    "p": {
        "TaskInst": {
            "id": 0,              // числовой идентификатор задачи
            "name": "abc_backup", // имя задачи
            "comment": "",        // комментарий к задаче
            "next_start": 0,      // время следующего выполнения задачи
            "last_start": 0,      // время предыдущего выполнения задачи
            "state": 0,           // состояние задачи «остановлена»
            "typ": 0,             // тип задачи – создание бэкапа
            "parameters": {
                "backupName": "abc_b", // имя файла бэкапа
                "backupPath": "./",    // путь к файлу бэкапа
                "baseName": "abc",     // имя базы данных
                "series": "*",         // выбираются все временные ряды
                "rangeType": 0         // диапазон времени не задан. Сохраняются
                                       // все записи выбранных временных рядов
            }
        }
    },
    "code": 0,
    "desc": ""
}
```

Изменение параметров задачи

Для изменения параметров задачи в команде указываются числовой идентификатор задачи и те параметры, которые надо изменить. В примере дается команда на изменение имени задачи.

```
{
    "cmd": "updateTask", // команда добавления задачи бэкапа
    "id": "2ad7cce0",    // ключ запроса
    "p": {
        "id": 0,              // числовой идентификатор задачи
        "TaskInst": {
            "name": "abcBack" // новое имя задачи
        }
    }
}
```

Ответ

```
{
    "cmd": "updateTask", // команда добавления задачи бэкапа
    "id": "2ad7cce0",    // ключ запроса
    "p": {
        "TaskInst": {
            "id": 0,
            "name": "abcBack", // измененное имя задачи
            "comment": "",
```

```
            "next_start": 1658102399000000000,
            "last_start": 1657497600000000000,
            "state": 0,
            "typ": 0,
            "parameters": {
                "backupName": "abc_b",
                "backupPath": "./",
                "baseName": "abc",
                "series": "*",
                "rangeType": 0
            }
        }
    },
    "code": 0,
    "desc": ""
}
```

Удалить задачу

В команде надо указать только числовой идентификатор удаляемой задачи.

```
{
    "cmd": "removeTask", // команда удаления задачи бэкапа
    "id": "4fbf65f4",    // ключ запроса
    "p": {
        "id": 0 // числовой идентификатор удаляемой задачи
    }
}
```

Ответ

```
{
    "cmd": "removeTask",  // команда удаления задачи бэкапа
    "id": "4fbf65f4",     // ключ запроса
    "p": {
        "id": 0
    },
    "code": 0,
    "desc": ""
}
```

Управление состоянием задачи (запуск, пауза, стоп)

С помощью команды управления состоянием задачи бэкапа можно запустить задачу, поставить ее на паузу или остановить. Если поставить на паузу, то при следующей команде запуска задача начнет выполняться с того места, где была остановлена, а при остановке – она начнётся сначала. В примере дается команда запуска задачи с числовым идентификатором 2.

```
{
    "cmd": "stateTask", // команда управления состоянием задачи бэкапа
    "id": "-412b2095",  // ключ запроса
    "p": {
        "id": 2,        // числовой идентификатор задачи
        "state": 1      // задаваемое состояние: 0 - стоп, 1 - запуск, 2 - пауза
    }
}
```

Ответ

```
{
    "cmd": "stateTask", // команда управления состоянием задачи бэкапа
    "id": "-412b2095",  // ключ запроса
    "p": {},
    "code": 0,
    "desc": ""
}
```

Работа с группами

Получение списка групп

В приведенном примере посылается запрос на получение списка групп из базы, в которой созданы 2 группы: test_grp_1 и test_grp. В первую из них добавлен 1 тег, а во вторую - ни одного.

```
{
       "cmd": "getTagGroups",      // команда получения списка групп
       "id":  "70c38b22",          // ключ запроса

       "p": {
              "baseName": "new"    // имя базы данных
  }
}
```

Ответ

```
{
  "cmd": "getTagGroups",          // команда получения списка групп
         "id": "70c38b22",        // ключ запроса
  "p": {
    "GroupInst": [                // массив описаний групп
      {
        "id": "g11668551",        // идентификатор группы
        "baseName": "new",        // имя базы данных
        "name": "test_grp_1",     // имя группы
        "comment": "Комментарий", // комментарий к группе
        "TagRefs": [              // массив описаний ссылок на теги группы
      {
        "id": "l6652752",         // идентификатор ссылки на тег
        "baseName": "new",        // имя базы данных
        "seriesId": 0             // идентификатор тега
      }
    ],
      },
      {
        "id": "g30193207",        // идентификатор группы
        "baseName": "new",        // имя базы данных
        "name": "test_grp",       // имя группы
        "comment": "",            // комментарий к группе
        "TagRefs": null           // массив описаний ссылок на теги группы
      }
    ]
  },
       "code": 0,
       "desc": ""
}
```

Добавление группы

В этом примере посылается запрос на создание группы **test_grp** в базе данных **new**.

```
{
       "cmd": "addTagGroup",      // команда добавления группы
       "id": "-639a93c7",         // ключ запроса
       "p": {
              "baseName": "new",  // имя базы данных
              "GroupInst": {
              "name": "test_grp", // имя создаваемой группы
              "comment": ""       // комментарий к создаваемой группе
    }
  }
}
```

ответ

```
{
       "cmd": "addTagGroup",      // команда добавления группы
       "id": "-639a93c7",         // ключ запроса
       "p": {
              "GroupInst": {
              "id": "g85921750",  // идентификатор созданной группы
              "baseName": "new",  // имя базы данных
              "name": "test_grp", // имя группы
              "comment": ""       //комментарий к группе
    }
  },
  "code": 0,
  "desc": ""
}
```

Изменение параметров группы

В примере посылается запрос на изменение комментария к группе.

```
{
       "cmd": "updateTagGroup",   // команда изменения параметров группы
       "id": "271dc4dc",          // команда изменения параметров группы
       "p": {
       "baseName": "new",         // имя базы данных
       "groupId": "g11668551",    // идентификатор группы
       "GroupInst": {             // перечень меняемых параметров
       "comment": "Комментарий"   // комментарий к группе
    }
  }
}
```

ответ

```
{
       "cmd": "updateTagGroup",   // команда изменения параметров группы
       "id": "271dc4dc",          // ключ запроса
  "p": {
    "GroupInst": {                // список обновленных параметров группы
      "id": "g11668551",
      "baseName": "new",
      "name": "test_grp_1",
      "comment": " Комментарий"
    }
  },
  "code": 0,
  "desc": ""
}
```

Удаление группы

В приведённом примере посылается запрос на удаление группы с идентификатором **g18804692**. При успешном удалении группы в ответ присылается объект с параметрами удаленной группы.

```
{
       "cmd": "removeTagGroup",          // команда удаления группы
       "id": "51fe0065",                 // ключ запроса
       "p": {
              "baseName": "new",         // имя базы данных
              "groupId": "g18804692"     // имя базы данных
  }
}
```

ответ

```
{
  "cmd": "removeTagGroup", // команда удаления группы
  "id": "51fe0065",        // ключ запроса
  "p": {
```

```
    "GroupInst": {         // параметры удаленной группы
      "id": " g18804692",
      "baseName": "new",
      "name": "test_grp",
      "comment": ""
    }
  },
  "code": 0,
  "desc": ""
}
```

Получение списка тегов группы

Ниже приведен пример запроса информации о тегах входящих в группу с идентификатором **g11668551** в базе с именем **new**. В группе содержится 2 тега с идентификаторами 0 и 1. В ответ на запрос сервер присылает массив объектов, в каждом из которых содержатся параметры одного из тегов, входящих в группу.

```
{
       "cmd": "getTagRefs",       // команда запроса списка тегов группы
       "id": "-47f403e",          // ключ запроса
  "p": {
       "baseName": "new",         // имя базы данных
       "groupId": "g11668551"     // имя базы данных
  }

}
```

ответ

```
{
  "cmd": "getTagRefs",            // команда запроса списка тегов группы
  "id": "-47f403e",               // ключ запроса
  "p": {
    "TagsInst": [                 // массив объектов, описывающих параметры
      {                           // тегов группы
        "id": "l13165732",        // идентификатор ссылки на тег
        "baseName": "new",        // имя базы
        "seriesId": 0             // идентификатор тега
      },
      {
        "id": "l20306433",
        "baseName": "new",
        "seriesId": 1
      }
    ]
  }
}
```

Добавление тегов в группу

Следующий пример демонстрирует запрос на добавление в группу с идентификатором **g11668551** в базе **new** тега с идентификатором 0. В запросе передается массив объектов **refInst**, каждый элемент которого содержит описание добавляемого тега (в данном примере – один элемент в массиве). В ответ на запрос сервер присылает массив объектов, каждый элемент которого содержит параметры одного из добавленных в группу тегов (в данном примере один элемент в массиве).

```
{
       "cmd": "addTagRefs",       // команда добавления тегов в группу
       "id": "3bfad5fd",          // ключ запроса
       "p": {
        "baseName": "new",        // имя базы
        "groupId": "g11668551",   // идентификатор группы
        "refInst": [              // параметры добавляемых тегов
       {
```

```
        "baseName": "new",
        "seriesId": 0
      }
    ]
  },
}
```

ответ

```
{
  "cmd": "addTagRefs",     // команда добавления тегов в группу
  "id": "3bfad5fd",        // ключ запроса
  "p": {
    "TagsInst": [          // параметры добавленных тегов
      {
        "id": "l6652752",
        "baseName": "new",
        "seriesId": 0
      }
    ],
    "groupId": "g6077759"
  },
  "code": 0,
  "desc": ""
}
```

Удаление тегов из группы

Ниже демонстрируется пример команды на удаление списка тегов (список состоит из 1 тега) из группы с идентификатором **g11668551** в базе данных **new**.

```
{
       "cmd": "removeTagRefs",    // команда удаления тегов из группы
       "id": "-7eb8a931",         // ключ запроса
       "p": {
              "baseName": "new",  // имя базы данных
              "groupId": "g11668551",    // идентификатор группы
               "refInst": [              // массив удаляемых из группы тегов
          {
               "id": "l22610146"         // идентификатор ссылки на тег
        }
    ]
  },
}
```

ответ

```
{
  "cmd": "removeTagRefs",         // команда удаления тегов из группы
  "id": "-7eb8a931",              // ключ запроса
  "p": {
    "TagsInst": [                 // массив описаний удаленных из группы тегов
      {
        "id": " l22610146",
        "baseName": "",
        "seriesId": 0
      }
    ],
    "groupId": "g11668551"        // идентификатор группы
  },
  "code": 0,
  "desc": ""
}
```