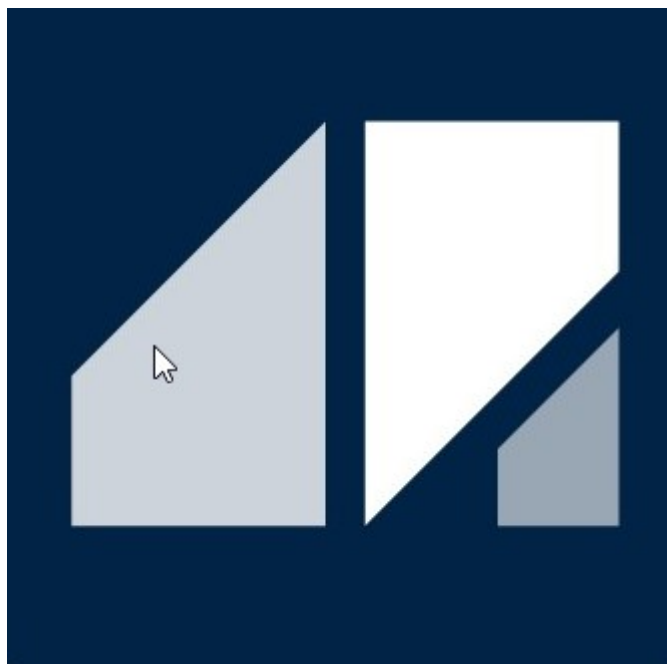


ПРОГРАММНЫЙ КОМПЛЕКС ASTRAREGUL



РГДП.58.29.14.000-001-03 РП

Базы данных

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Редакция 4

Соответствует релизу ПК AstraRegul версии 2024.06.03.00

СПИСОК ИЗМЕНЕНИЙ

| Редакция | Список изменений |
|------------|--|
| Редакция 4 | <ul style="list-style-type: none">- Обновлено описание файловой структуры базы данных в разделе Astra.Historian.- Обновлено описание сигналов виртуальных таблиц для компонента Astra.Rmap в разделах Таблицы данных и Таблицы событий. |
| Редакция 3 | <ul style="list-style-type: none">- Обновлено описание структуры базы данных Astra.Historian.- Добавлен раздел SQL-запросы с рекомендациями по работе с SQL-запросами компонента Astra.RMap.- Добавлено описание работы с компонентом Astra.RMap в LibreOffice Calc.- Добавлено описание работы с БД MySQL. |
| Редакция 2 | <ul style="list-style-type: none">- Добавлен раздел Структура базы данных. |

ОГЛАВЛЕНИЕ

| | |
|--|----|
| СПИСОК ИЗМЕНЕНИЙ | 2 |
| 1. Базы данных | 5 |
| 1.1. Astra.Historian | 6 |
| 1.1.1. Настройка | 13 |
| 1.1.1.1. Вычисление значения атрибута MaxPoolMemorySize | 16 |
| 1.1.1.2. Атрибуты баз данных | 19 |
| 1.1.1.2.1. Атрибуты TCP-сервера | 20 |
| 1.1.1.2.2. Атрибуты хранения | 21 |
| 1.1.1.2.3. Атрибуты для ограничений на хранение данных..... | 22 |
| 1.1.1.2.4. Атрибуты для сжатия архивированных данных | 25 |
| 1.1.2. Структура базы данных..... | 27 |
| 1.1.2.1. Общий формат записи и оценка её размера | 29 |
| 1.1.2.1.1. Размер тела записи истории значений | 30 |
| 1.1.2.1.2. Размер тела записи истории событий..... | 32 |
| 1.1.2.2. Коэффициент сжатия | 33 |
| 1.1.2.3. Общая оценка размера базы данных | 34 |
| 1.1.2.4. Оценка суточного объема данных от сервера | 38 |
| 1.1.2.4.1. Оценка суточного объема истории значений..... | 39 |
| 1.1.2.4.2. Оценка суточного объема истории событий | 40 |
| 1.1.2.4.3. Пример расчета | 41 |
| 1.1.2.5. Оценка объема временного хранилища очереди данных..... | 44 |
| 1.1.2.6. Оценка объема заполнения дискового пространства | 46 |
| 1.1.3. Диагностика работы | 49 |
| 1.1.4. Администрирование..... | 54 |
| 1.1.4.1. Перенос баз данных | 56 |
| 1.1.4.2. Сохранение части архива базы данных | 57 |
| 1.1.4.3. Восстановление сохраненных архивных данных для просмотра | 58 |
| 1.2. Astra.RMap..... | 59 |
| 1.2.1. Настройка | 60 |
| 1.2.1.1. Windows..... | 61 |
| 1.2.1.2. AstraLinux | 65 |

| | |
|--|-----|
| 1.2.1.3. РЕД ОС | 72 |
| 1.2.2. Виртуальные таблицы | 75 |
| 1.2.2.1. Таблицы данных | 76 |
| 1.2.2.2. Таблицы событий..... | 81 |
| 1.2.3. SQL-запросы | 84 |
| 1.2.3.1. Примеры..... | 86 |
| 1.2.4. Подключение по TCP | 92 |
| 1.2.5. Предоставление данных в MS Excel..... | 94 |
| 1.2.6. Предоставление данных в LibreOffice Calc | 99 |
| 1.2.6.1. Windows..... | 100 |
| 1.2.6.2. AstraLinux | 110 |
| 1.3. Работа с БД MySQL..... | 122 |
| 1.3.1. MySQL | 123 |
| 1.3.1.1. Установка | 124 |
| 1.3.1.2. Создание таблицы | 144 |
| 1.3.1.3. Настройка пользователя | 146 |
| 1.3.1.4. Создание источника данных..... | 151 |
| 1.3.2. Настройка в Astra.IDE | 154 |
| 1.3.2.1. Логика функционального блока для внешней БД | 155 |
| 1.3.2.2. Использование функционального блока | 160 |
| 1.3.2.3. Проверка работы программы | 164 |
| 1.3.3. Настройка в Astra.HMI..... | 166 |
| 1.3.3.1. Настройка переменных окружения для взаимодействия с SQL базой данных на Linux | 167 |
| 1.3.3.2. Создание проекта Astra.HMI..... | 169 |
| 1.3.3.3. Проверка работы | 181 |

1. Базы данных

| Компонент | Версия | Описание |
|---------------------------------|----------|--------------------------------|
| Astra.Historian | 1.1.10.1 | Проприетарная нереляционная БД |
| Astra.RMAP | 1.2.6.1 | Расширение СУБД PostgreSQL |

1.1. Astra.Historian

Astra.Historian – программный компонент для хранения истории изменений значений сигналов и уведомлений о происходивших событиях.

Функции:

- › сбор и хранение оперативных значений параметров технологического процесса.
- › сбор и хранение истории событий и тревог технологического процесса.
- › предоставление исторических данных клиентам.

Функционирует в виде:

- › службы Astra.Historian.Server на ОС Windows;
- › сервиса astra.historian.server.service на ОС Linux.

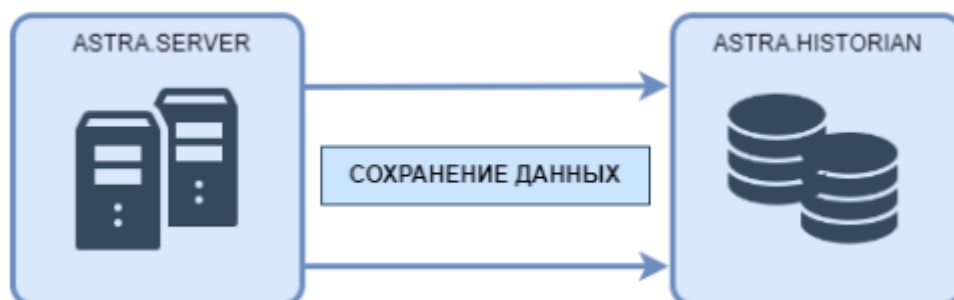


На компьютере может быть установлен только один экземпляр сервера истории.

Сохранение и получение данных

Источниками данных являются экземпляры сервера ввода/вывода Astra.Server.

Astra.Server обеспечивает сбор, фильтрацию и сохранение данных (событий и значений сигналов) в БД Astra.Historian через модуль истории.



Модуль истории в составе Astra.Server выполняет временное хранение данных на стороне сервера и передачу информации в хранилище Astra.Historian.

Буфер временного хранения данных на стороне Astra.Server располагается на жестком диске, что предотвращает потерю данных при аварийном отключении компьютера сервера технологических данных. При следующем старте сервера переданные данные будут повторно отправлены в Astra.Historian. Буферизация данных позволяет также сгладить пиковые нагрузки при большой интенсивности получения данных.

Astra.Historian предназначен для управления базами данных и предоставления хранимой исторической информации клиентам. Сервер может обслуживать несколько БД одновременно.

Базы данных хранят данные, предоставляемые модулем истории. Используется БД, управляемая Astra.Historian. В одну БД могут сохраняться данные с нескольких независимых источников.

Обработка данных

Данные, сохраняемые в сервер истории, попадают во внутренний журнал хранилища. Это способствует ускорению процесса перемещения данных из очереди источника.

Для каждого хранилища в рамках транзакций сервер истории выполняет последовательную обработку поступивших данных с целью размещения их в основной области. Обработка данных происходит в фоновом режиме и не препятствует операциям чтения и записи.

Сервер истории обрабатывает данные порциями в том порядке, в котором они были сохранены в очередь. Сервер истории фильтрует сохраняемые данные и отбрасывает устаревшие (метка времени лежит левее текущей нижней границы для всего хранилища) и архивные (метка времени лежит в текущих границах архивной области). Все остальные данные успешно сохраняются в основную область хранилища.

Сохраняемые данные записываются в соответствующие фрагменты оперативной области хранилища, если подходящего фрагмента нет, то он создается. В файл фрагмента попадают записи временных рядов по всем элементам, информация по которым хранится в истории.

Данные в БД хранятся в закрытом бинарном формате. Сервер допускает лишь добавление новых записей. При этом механизм обновления ранее сохраненной истории, реализованный в Astra.Imitator, не модифицирует существующие записи, а добавляет новые с пометками о перезаписи поверх ранее сохраненных, формируя таким образом новый слой записей. В текущей реализации пользовательские инструменты позволяют просмотреть записи только из самого последнего сохраненного слоя. Для получения записей из предыдущих слоев в настоящее время необходимо обратиться к разработчику. В дальнейшем средства доступа к историческим данным будут доработаны таким образом, чтобы было возможным чтение в том числе и перезаписанных данных.

Содержимое источника представляет собой совокупность пространств, каждое из которых является совокупностью элементов. С течением времени происходят связанные с элементом события. Соответственно событиям строится временной ряд, который упорядочен по метке времени. Каждая запись временного ряда соответствует событию элемента.

По мере поступления сохраняемых данных в хранилище накапливается информация об источниках, пространствах и элементах, история значений которых сохраняется в данное хранилище, если таковой информации нет, то сервер истории создает ее.

Хранение данных

В сервере истории данные хранятся в фрагментах - файлах, содержащих данные за сутки. У каждого фрагмента определяется нижняя граница времени для фрагмента - начало суток по времени UTC.

После поступления в сервер истории, данные последовательно проходят несколько стадий хранения:

- › в активной области;
- › в архивной области.

Активная область предназначена для формирования поступающих данных во временные ряды: это обеспечивает высокую скорость обработки запроса данных. Каждому источнику данных соответствует своя активная область. Длительность хранения фрагментов в активной области задается в настройках БД.

Фрагменты, нижняя граница которых оказывается левее предела времени хранения в активной области, переносятся в **архивную область**. Фрагменты из всех активных областей переносятся в единую архивную область, при этом фрагменты из разных активных областей, имеющие одинаковую нижнюю границу времени сливаются в один файл фрагмента. Для уменьшения занимаемого места на диске, фрагменты в архивной области сжимаются с параметрами сжатия, указанными в настройках БД.

Хранение данных ведется в суточных файлах данных для увеличения скорости доступа к данным. Сервер реализует механизмы сохранения и поиска необходимых данных, направленные на обеспечение максимальной производительности работы с дисковой подсистемой компьютера.

Глубина хранения данных ограничена размерами дискового пространства. Скорость записи и чтения данных не зависит от глубины хранения. Запись в сервер – транзакционная. Сервер обеспечивает высокую плотность записи хранимых данных на диск, уменьшая таким образом объемы читаемых с диска данных.

Резервирование

Astra.Historian позволяет формирование резервируемых хранилищ данных.

Резервирование представляет собой параллельное сохранение исторических данных источником данных в несколько баз данных, принадлежащих разным серверам истории.

При работе с резервируемыми хранилищами, данные из источника не удаляются, пока не пройдет запись во все хранилища.



Сохранение данных в несколько баз данных, принадлежащих одному серверу истории не решает задачу резервирования и не рекомендуется.

Предоставление данных клиентам

Предоставление данных, хранящихся в базах данных сервера истории, выполняет модуль истории в составе Astra.Server или Astra.AccessPoint.

Предоставление данных клиентам осуществляется по проприетарному протоколу на базе TCP.



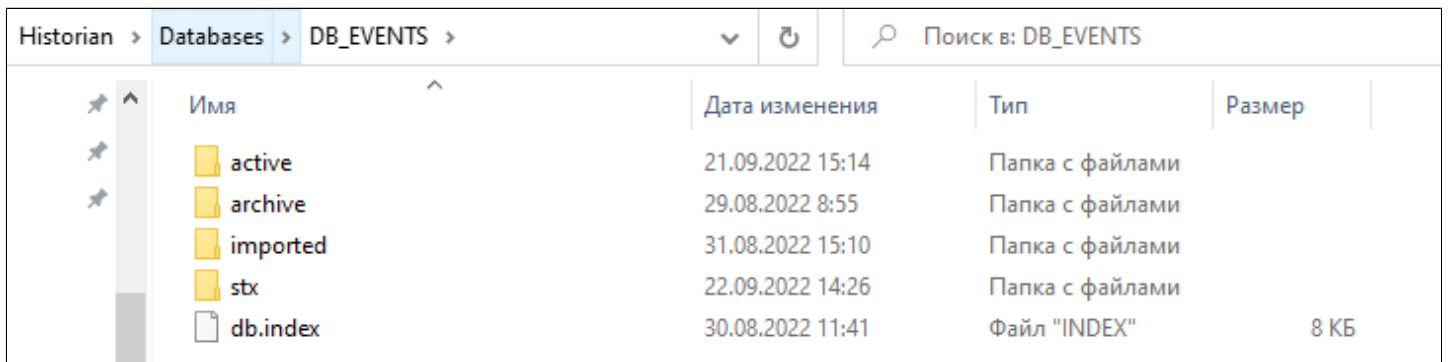
Характеристики

- Производительность записи: до 750 000 изменений значений в секунду.
- Производительность чтения: до 1 000 000 изменений значений в секунду.

Файловая структура базы данных

В файловой системе база данных содержит следующие папки и файл:

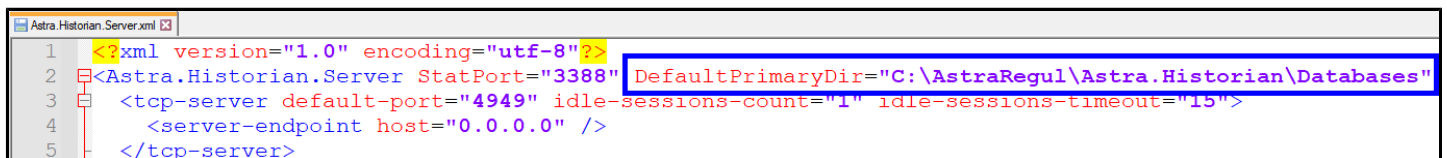
- › active
- › archive
- › imported
- › stx
- › db.index



| Имя | Дата изменения | Тип | Размер |
|----------|------------------|-----------------|--------|
| active | 21.09.2022 15:14 | Папка с файлами | |
| archive | 29.08.2022 8:55 | Папка с файлами | |
| imported | 31.08.2022 15:10 | Папка с файлами | |
| stx | 22.09.2022 14:26 | Папка с файлами | |
| db.index | 30.08.2022 11:41 | Файл "INDEX" | 8 КБ |

Папки и файл создаются автоматически, когда сервер истории создаёт базу данных. Их расположение в файловой системе зависит от описания базы данных в файле конфигурации:

- › Если для базы данных не указаны основной и архивный каталоги, её папки и файл будут храниться в каталоге по умолчанию (атрибут "DefaultPrimaryDir" сервера истории).



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <Astra.Historian.Server StatPort="3388" DefaultPrimaryDir="C:\AstraRegul\Astra.Historian\Databases"
3 <tcp-server default-port="4949" idle-sessions-count="1" idle-sessions-timeout="15">
4   <server-endpoint host="0.0.0.0" />
5 </tcp-server>
```

- › Если для базы данных указан основной каталог, папки и файл будут храниться в нём (атрибут "PrimaryDir" сервера истории).

```
Astra.Historian.Server.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <Astra.Historian.Server StatPort="3388" DefaultPrimaryDir="C:\AstraRegul\Astra.Historian\Databases" MaxPoolMemorySize="512" EnableDCOM="0">
3 <tcp-server default-port="4949" idle-sessions-count="1" idle-sessions-timeout="15">
4 <server-endpoint host="0.0.0.0" />
5 </tcp-server>
6 <Bases>
7 <Base Alias="DB_EVENTS" PreferredCommonCacheLimit="128" PrimaryDir="C:\AstraRegul\Astra.Historian\Custom_Databases" ActiveStorageDepth="3" StorageDepth="365" VolumeLimit="2000" />
8 <Base Alias="DB_HISTORY" PreferredCommonCacheLimit="128" PrimaryDir="C:\AstraRegul\Astra.Historian\Custom_Databases" ActiveStorageDepth="3" StorageDepth="365" VolumeLimit="2000" />
9 <Base Alias="DB_IMIT" PreferredCommonCacheLimit="128" PrimaryDir="C:\AstraRegul\Astra.Historian\Custom_Databases" ActiveStorageDepth="3" StorageDepth="365" VolumeLimit="2000" />
10 </Bases>
```

➤ Если для базы данных указаны и основной и архивный каталоги (атрибуты "PrimaryDir" и "ArchiveDir" сервера истории), папки archive и imported будут храниться в архивном каталоге, остальные папки и файл – в основном каталоге.

```
Astra.Historian.Server.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <Astra.Historian.Server StatPort="3388" DefaultPrimaryDir="C:\AstraRegul\Astra.Historian\Databases" MaxPoolMemorySize="512" EnableDCOM="0">
3 <tcp-server default-port="4949" idle-sessions-count="1" idle-sessions-timeout="15">
4 <server-endpoint host="0.0.0.0" />
5 </tcp-server>
6 <Bases>
7 <Base Alias="DB_EVENTS" PreferredCommonCacheLimit="128" PrimaryDir="C:\AstraRegul\Astra.Historian\Custom_Databases" ArchiveDir="C:\AstraRegul\Astra.Historian\Archive_Databases" ActiveStorageDepth="3" StorageDepth="365" VolumeLimit="2000" />
8 <Base Alias="DB_HISTORY" PreferredCommonCacheLimit="128" PrimaryDir="C:\AstraRegul\Astra.Historian\Custom_Databases" ActiveStorageDepth="3" StorageDepth="365" VolumeLimit="2000" />
9 <Base Alias="DB_IMIT" PreferredCommonCacheLimit="128" PrimaryDir="C:\AstraRegul\Astra.Historian\Custom_Databases" ActiveStorageDepth="3" StorageDepth="365" VolumeLimit="2000" />
10 </Bases>
```

➤ Если для базы данных указан только архивный каталог (атрибут "ArchiveDir" сервера истории), папки archive и imported будут храниться в нём, а остальные папки и файл – в каталоге по умолчанию (атрибут "DefaultPrimaryDir" сервера истории).

```
Astra.Historian.Server.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <Astra.Historian.Server StatPort="3388" DefaultPrimaryDir="C:\AstraRegul\Astra.Historian\Databases" MaxPoolMemorySize="512" EnableDCOM="0">
3 <tcp-server default-port="4949" idle-sessions-count="1" idle-sessions-timeout="15">
4 <server-endpoint host="0.0.0.0" />
5 </tcp-server>
6 <Bases>
7 <Base Alias="DB_EVENTS" PreferredCommonCacheLimit="128" ArchiveDir="C:\AstraRegul\Astra.Historian\Archive_Databases" ActiveStorageDepth="3" StorageDepth="365" VolumeLimit="2000" />
8 <Base Alias="DB_HISTORY" PreferredCommonCacheLimit="128" PrimaryDir="C:\AstraRegul\Astra.Historian\Custom_Databases" ActiveStorageDepth="3" StorageDepth="365" VolumeLimit="2000" />
9 <Base Alias="DB_IMIT" PreferredCommonCacheLimit="128" PrimaryDir="C:\AstraRegul\Astra.Historian\Custom_Databases" ActiveStorageDepth="3" StorageDepth="365" VolumeLimit="2000" />
10 </Bases>
```

1.1.1. Настройка


Вы можете настроить Astra.Historian в среде разработки Astra.IDE, используя плагин AstraRegul, либо вручную, отредактировав файл конфигурации **Astra.Historian.Server.xml** в папке установки.

Конфигурационный файл



```
<?xml version="1.0" encoding="utf-8"?>
<Astra.Historian.Server StatPort="3388" DefaultPrimaryDir="c:
\Historian\Databases" MaxPoolMemorySize="512" EnableDCOM="0">
  <tcp-server default-port="4949" idle-sessions-count="1" idle-
sessions-timeout="15">
    <server-endpoint host="0.0.0.0" />
  </tcp-server>
  <Bases>
    <Base Alias="DB_EVENTS" PreferredCommonCacheLimit="128"
ActiveStorageDepth="3" StorageDepth="365" VolumeLimit="10000"/>
    <Base Alias="DB_HISTORY" PreferredCommonCacheLimit="128"
ActiveStorageDepth="3" StorageDepth="365" VolumeLimit="10000"/>
  </Bases>
</Astra.Historian.Server>
```

Атрибуты

| Атрибут | Описание | Значение по умолчанию |
|-------------------|--|--------------------------|
| StatPort | Номер порта, по которому будет открыт доступ к статистике сервера. Обязательный атрибут. | 3388 |
| DefaultPrimaryDir | Путь до основного каталога файлов БД. Используется в качестве значения по умолчанию для БД, в которых не переопределен | "C:\Historian\Databases" |
| MaxPoolMemorySize | Максимальный объем оперативной памяти, используемой в качестве кеша для работы с базами данных (Мб). 0 - без ограничения. В 64-битном варианте верхняя граница - 128 Гб. Значение данного атрибута вычисляется по формуле . | 512 |
| EnableDCOM | Поддержка DCOM. Значения: 0 – отключен 1 – включен <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> В новых проектах необходимо отключать поддержку DCOM.</div> | 1 |



Чтобы изменения вступили в силу, перезапустите службу (сервис) Astra.Historian.

1.1.1.1. Вычисление значения атрибута MaxPoolMemorySize

Для вычисления значения атрибута используется следующая формула:

$$f \quad MaxPoolMemory \geq \sum_i (ServerTagsCount_i * R_i) * 1.2 * 4KB + \sum_j DBCache_j * 1MB,$$

где i – отдельный экземпляр Astra.Server, для которого настроено сохранение истории значений сигналов в базы данных данного сервера истории;

ServerTagsCount – количество сигналов, сохраняемых i -м экземпляром Astra.Server;

R – если i -й Astra.Server работает в составе резервной пары, то значение равно 2, иначе - 1;

j – отдельная БД данного сервера истории;

DBCACHE – размер оперативной памяти, выделенной для j -й базы данных в качестве кеша (параметр PreferredCommonCacheLimit в файле конфигурации).



Количество сохраняемых сигналов можно посмотреть в приложении Статистика.

The screenshot shows the 'Statistics' application interface. On the left, a tree view shows the 'HistoryModule' selected under 'TcpServer'. On the right, the 'Parameters' window displays a list of configuration items. The 'Additional parameters' section is highlighted, and the 'Number of signals to be saved' parameter is set to 5.

| Имя | Значение |
|---|----------|
| Дополнительные параметры | |
| Количество сохраняемых сигналов | 5 |
| Количество зарегистрированных изменений значен... | 0 |
| Количество отброшенных фильтрами значений | 0 |
| Количество значений, которые не удалось принять ... | 0 |
| Количество значений, принятых к сохранению | 0 |
| Количество событий, принятых к сохранению | 0 |

Пример

Сервер истории управляет двумя базами данных и хранит историю значений для трех экземпляров Astra.Server:

- › ProcessVals (значение PreferredCommonCacheLimit – 256);
- › ControlVals (значение PreferredCommonCacheLimit не указано, берем значение по умолчанию - 128).



```
<?xml version="1.0" encoding="utf-8"?>
<Astra.Historian.Server StatPort="3388" DefaultPrimaryDir="c:
\Astra.Historian\Databases">
  <tcp-server default-port=4949 idle-sessions-count=1 idle-sessions-
timeout=15>
    <server-endpoint host="0.0.0.0" />
  </tcp-server>
  <Bases>
    <Base Alias="ProcessVals" PreferredCommonCacheLimit="256" />
    <Base Alias="ControlVals" />
  </Bases>
</Astra.Historian.Server>
```

Данный сервер истории хранит историю значений для трех экземпляров Astra.Server:

- › Первый Astra.Server сохраняет значения 15000 сигналов.
- › Второй Astra.Server сохраняет значения 10000 сигналов и работает в составе резервной пары.
- › Третий Astra.Server сохраняет значения 7500 сигналов.



Для вычисления значения MaxPoolMemory неважно, в какую базу данных сервера истории записывает значения сигналов каждый из экземпляров Astra.Server.

Подставив значения в формулу, получим значение атрибута:



$$\text{MaxPoolMemory} = (15000 + 10000 * 2 + 7500) * 1.2 * 4\text{КБ} + (256 + 128) * 1\text{МБ} = 588 \text{ МБ}$$

Поскольку значение атрибута должно быть не меньше вычисленного значения, то в файле конфигурации сервера истории можно указать вычисленное значение с округлением вверх.



```
<?xml version="1.0" encoding="utf-8"?>
<Astra.Historian.Server StatPort="3388" DefaultPrimaryDir="c:
\Astra.Historian\Databases" MaxPoolMemorySize="600">
  <tcp-server default-port=4949 idle-sessions-count=1 idle-sessions-
timeout=15>
    <server-endpoint host="0.0.0.0" />
  </tcp-server>
  <Bases>
    <Base Alias="ProcessVals" PreferredCommonCacheLimit="256" />
    <Base Alias="ControlVals" />
  </Bases>
</Astra.Historian.Server>
```

1.1.1.2. Атрибуты баз данных

Базы данных перечислены в элементе Bases.

После первой установки список баз данных пуст. Чтобы добавить базу данных, в элемент Bases добавьте элемент Base и укажите его атрибуты – параметры базы данных. Сервер истории создаст базу данных при перезапуске.

- › [Атрибуты TSP-сервера](#)
- › [Атрибуты хранения](#)
- › [Атрибуты для ограничений на хранение данных](#)
- › [Атрибуты для сжатия архивированных данных](#)

1.1.1.2.1. Атрибуты TCP-сервера

| Атрибут | Описание | Значение по умолчанию |
|-----------------------|--|-----------------------|
| default-port | Порт по умолчанию, используется для дочерних элементов server-endpoint, у которых не указан порт. | 4949 |
| idle-sessions-count | Количество одновременно удерживаемых неактивных сессий. Значение по умолчанию – 0 (не ограничено). | 1 |
| idle-sessions-timeout | Время ожидания активации неактивных сессий (в секундах). Значение по умолчанию – 0 (не ограничено). | 15 |
| server-endpoint | Адрес, по которому доступен сервер. | 0.0.0.0 |

Атрибуты server-endpoint

В элемент tcp-server можно добавить любое количество дочерних элементов server-endpoint.

| Атрибут | Описание | Значение по умолчанию |
|---------|---|-----------------------|
| port | Порт точки доступа к серверу. Если не указан, используется порт default-port родительского элемента. | default-port |
| host | Хост точки доступа к серверу. | 0.0.0.0 (любой адрес) |

1.1.1.2.2. Атрибуты хранения

| Название | Описание | Значение по умолчанию |
|------------|--|-----------------------|
| Alias | Короткое имя БД (псевдоним), должно быть уникальным для сервера истории Обязательный атрибут. | — |
| PrimaryDir | Путь до основного каталога БД. Если не указан, то совпадает с DefaultPrimaryDir из настроек сервера | — |
| ArchiveDir | Путь до архивного каталога БД. Если не указан, то совпадает с PrimaryDir | — |

1.1.1.2.3. Атрибуты для ограничений на хранение данных

| Название | Описание | Значение по умолчанию |
|---------------------------|--|-----------------------|
| PreferredCommonCacheLimit | Предпочтительный объем оперативной памяти в мегабайтах, используемой в качестве кеша для запроса данных из базы данных. 0 - без ограничений. Чем больше значение данного параметра, тем больше данных будет находиться в кэше, обращение к которому выполняется значительно быстрее, чем запрос данных с диска. | 128 |
| ActiveStorageDepth | Длительность хранения данных в активной области в сутках. Минимальное значение – 1. | 3 |
| StorageDepth | Максимальная глубина хранения данных в БД до удаления в сутках. 0 - без ограничений; если не 0, то должно быть больше или равно значения ActiveStorageDepth. | 0 |
| VolumeLimit | Предельный объем БД в мегабайтах; при превышении предела выполняется очистка архивных данных до снижения объема ниже предела, если это возможно. 0 – без ограничений. | 0 |



Атрибуты `ActiveStorageDepth` и `StorageDepth` отвечают за бракование данных, которые не попали в базу данных из-за устаревшей или опережающей метки времени.

Для каждого активного раздела учитываются следующие временные границы (слева-направо в направлении оси времени):

$\text{OutdatedBound} < \text{ActiveLowerBound} < \text{FrontBound} < \text{CurrentHistorianServerTime}$,

где **OutdatedBound** – это общая нижняя граница по общей глубине хранения, все, что ниже нее, признается устаревшим и сразу удаляется (застойные данные, например, когда база долго была в простое и ничего не передавалось в архив).

ActiveLowerBound – это нижняя граница активности для данной конкретной активной области:



$\text{ActiveLowerBound} = \text{FrontBound} - \text{ActiveStorageDepth}$

Все, что поступает на запись левее нее, также признается устаревшим и отбрасывается. А то, что уже сохранено (имеющиеся фрагменты левее нижней границы) запечатываются и передаются в архив.

FrontBound – фронтальная граница, соответствующая начальной границе самого "молодого" фрагмента в этом разделе. Смещается при появлении очередного самого "молодого" фрагмента, который, в свою очередь, появляется, когда по одному из тегов накапливаются данные, примерно, на 4 КБ. При этом если «хвост» данных приходится на следующие сутки (по UTC), то граница `FrontBound` смещается на следующие сутки. При этом если это будут не прям следующие сутки, а +N суток, то создастся такой самый "молодой" фрагмент, что граница `FrontBound` сместится правее на +N суток, что в итоге приведет к соответствующему смещению `ActiveLowerBound`.

CurrentHistorianServerTime – текущее время на сервере (Astra.Historian).
Используется для фильтрации записей "из будущего": если метка времени сохраняемой записи > CurrentHistorianServerTime + 1 hour, то она отбрасывается как опережающая.

1.1.1.2.4. Атрибуты для сжатия архивированных данных

| Название | Описание | Значение по умолчанию |
|-------------------------|---|-----------------------|
| ArchiveCompression | Алгоритм сжатия архивных файлов: <ul style="list-style-type: none"> › none – без сжатия; › lzma – сжатие алгоритмом LZMA. | none |
| BlocksPerCompressedPack | Архивные файлы сжимаются группами по несколько файлов в один сжатый файл. Значение данного параметра задает максимальное количество архивных файлов в группе. Чем больше значение данного параметра, тем выше коэффициент сжатия и тем медленнее чтение данных из сжатых файлов. | 10 |
| Recompress | Стратегия выбора файлов для сжатия: none – сжатие не применяется; uncompressed – сжатие применяется только к несжатым файлам; any – сжатие применяется ко всем архивированным файлам: сжатые файлы сжимаются с новыми параметрами сжатия. Файлы, параметры сжатия ArchiveCompression и BlocksPerCompressedPack которых совпадают с новыми параметрами сжатия, пересжиматься не будут. | none |



Размер сжатых данных точно определить нельзя, так как коэффициент сжатия зависит от самих данных. На практике он варьируется от 2.5 до 3.5.

1.1.2. Структура базы данных

База данных состоит из следующих разделов:

- › **active/<uuid>** – множество активных разделов, каждый из которых создается для отдельного источника, выполняющего сохранение данных.
- › **archive** – единый архивный раздел, в который данные передаются из активных разделов по мере накопления.

Также база данных содержит служебный файл:

- › **db.index** – основной индекс базы данных.

Активный раздел

Активный раздел содержит:

- › Служебные файлы:
 - › **dbpart.index** – индекс элементов раздела.
 - › **dbpart.workset** – рабочее множество раздела.
 - › **dbpart.tx** – файл журнала состояния раздела (данный файл имеет небольшой размер (< 1 МиБ), и не учитывается в оценках).
- › **ahdb-<Date>.store** – множество фрагментов данных.

Архивный раздел

Архивный раздел содержит:

- › **ahdb-<Date>.store** – множество фрагментов данных.

Размер основного индекса базы данных от общего числа элементов в базе данных и среднего размера идентификатора элементов (зависят от источника). Размеры индекса и рабочего множества активных разделов зависят от количества элементов от соответствующего источника.

Размер фрагментов данных зависит от числа содержащихся в них записей и их размеров.



Фрагменты в активных и архивном разделах имеют одинаковый внутренний формат.

Число фрагментов в активных разделах зависит от параметра `ActiveStorageDepth`, определяющего глубину активного хранения (с возможностью записи) для базы данных. Число фрагментов в архивном разделе – от параметра `StorageDepth`, определяющего общую глубину хранения для базы данных.

В активных разделах фрагменты всегда хранятся в несжатом виде, в архивном – в зависимости от настроек сжатия.

Сжатые файлы (в архивной папке `archive`) визуально не отличаются от несжатых. Отличие состоит в их внутреннем содержимом (формат которого определяется по служебной информации, сохраняемой в файл).

`Astra.Historian` выполняет не сжатие архиватором всего файла, а сжатие отдельных блоков содержимого при поддержании одинаковых системных и индексных структур.

При перемещении фрагмента данных из активной области в архивную выполняется его дефрагментация для улучшения характеристики кластеризации данных по тегам (для повышения эффективности чтения). Это приводит к уплотнению данных и изменению итогового размера файла.



Размеры файлов индексов, рабочего множества и фрагментов данных в активных разделах выравниваются на 16 МБ (связано с необходимостью упреждающего выделения внешней памяти для сокращения издержек).



Если источник резервируется, то активный раздел создается для каждого его экземпляра.

1.1.2.1. Общий формат записи и оценка её размера

В базах данных записи хранятся в виде пар < Timestamp, Body >:

- › Timestamp – 8-байтная метка времени записи;
- › Body – последовательность байтов, содержащая тело записи в том виде, в каком она была сохранена источником.

Метка времени и необходимая для хранения записи служебная информация образуют ее "константную" часть, совокупный размер которой составляет 12 байт.

Тело записи, размер которого зависит от источника, составляет ее "переменную" часть. Ее размер хранится в служебной информации (учтен в 12-байтной константной части).

Astra.Server сохраняет исторические данные двух видов:

- › история значений – события изменения значений тегов (сигналов);
- › история алармов – история событий, связанных с алармами: активация, деактивация, квитирование и др.

Размер отдельной записи в несжатой части базы данных вычисляется по следующей общей формуле:



$$\text{RecordSize}(x) = 12 + \text{BodySize}(x),$$

где x – некоторая запись;

$\text{BodySize}(x)$ – размер тела записи, зависящий от источника и самой записи (тело может иметь фиксированную или переменную длину в зависимости от типа записи).

1.1.2.1.1. Размер тела записи истории значений

Тело записи содержит значение, качество, метку времени сервера (в б.х серверах сохраняется опционально) и необходимую служебную информацию.

Размер тела записи истории значений зависит от типа значения:

- для булевского, числовых типов и метки времени – фиксированный размер;
- для строк – в зависимости от размера строки.

В таблице ниже приведен размер элементов тела записи для различных типов данных:

| Тип значения | Метка времени и необходимая служебная информация | | Тело записи | | |
|--------------|--|--------------------------------------|----------------|--------------------------------|----------------|
| | Метка времени, байт | Служебная информация внутри БД, байт | Качество, байт | Дескриптор типа значения, байт | Значение, байт |
| bool | 8 | 4 | 4 | 1 | 0 |
| int1 | | | | | 1 |
| uint1 | | | | | 1 |
| int2 | | | | | 2 |
| uint2 | | | | | 2 |
| int4 | | | | | 4 |
| uint4 | | | | | 4 |
| int8 | | | | | 8 |
| uint8 | | | | | 8 |
| float | | | | | 4 |
| double | | | | | 8 |
| string* | | | | | 4 + N |

* – В случае строкового типа N – размер тела строки строки.

Соответствие размеров тела записи типам с учетом и без учета метки времени приведено в следующей таблице:

| Тип значения | Размер тела записи без учета метки времени, байт | Размер тела записи с учетом метки времени, байт |
|--------------|--|---|
| bool | 5 | 17 |
| int1 | 6 | 18 |
| uint1 | 6 | 18 |
| int2 | 7 | 19 |
| uint2 | 7 | 19 |
| int4 | 9 | 21 |
| uint4 | 9 | 21 |
| int8 | 13 | 25 |
| uint8 | 13 | 25 |
| float | 9 | 21 |
| double | 13 | 25 |
| string* | 9 + N | 21 + N |

* – В случае строкового типа N – размер тела строки строки.



Строки сохраняются в кодировке UTF-8, соответственно, при его вычислении необходимо учитывать следующее:

- латинские символы занимают 1 байт;
- кириллические символы - 2 байта.

1.1.2.1.2. Размер тела записи истории событий

История алармов от одного источника Astra.Server сохраняется в один общий временной ряд.

Записи истории алармов имеют переменный размер, так как:

- › содержат строковые данные переменной длины: сообщение, условие, подусловие, сообщение квитирования;
- › могут содержать переменное количество атрибутов.

В историю алармов сохраняются не только события активации (срабатывания алармов), но и последующие деактивации, события квитирования (которые могут быть множественными), события установки и снятия блокировок и подавлений. Поэтому оценка суточного объема истории алармов зависит от конкретной конфигурации Astra.Server и особенностей ее работы.

Количество байт, которые приходятся на одно сообщение, зависит от содержимого сообщения:

| Содержимое | Средний размер на одно сообщение, байт |
|-------------------|--|
| Цифры | 174 |
| Кириллица и цифры | 200 |
| Латиница и цифры | 179 |



На практике среднее значение объема одного исходного события составляет 200-300 байт.

1.1.2.2. Коэффициент сжатия

Для сжатия содержимого фрагментов архивного раздела используется алгоритм LZMA. Коэффициент сжатия в значительной степени зависит от исходных данных. На практике он варьируется в пределах 2-5 крат (выводится в журнал в сообщении о завершении архивации фрагмента).

1.1.2.3. Общая оценка размера базы данных

Обозначения принятые при расчете размера базы данных:

- › **Source[i]** – множество источников, сохраняющих данные.
- › **Align16MiB(x)** - операция выравнивания размера до 16 МиБ:

$$f \quad \text{Align16MiB}(x) = \text{ceil}(x / 16 \text{ MiB}) * 16 \text{ MiB},$$

где **ceil** – оператор округления вверх до ближайшего целого;

- › **SUM(expr)** – сумма подвыражений expr по множеству.

Общий размер базы данных вычисляется по следующей формуле:

$$f \quad [\text{Общий размер базы данных}] = [\text{Размер хранимых данных}] + [\text{Размер служебных данных}]$$

Размер хранимых данных (РХД):

$$f \quad \text{РХД} = \text{РАР} + \text{РХДАР},$$

где РАР – размер архивного раздела;
РХДАР – размер хранимых данных в активных разделах.

Размер служебных данных (РСД):

$$f \quad \text{РСД} = \text{РФОИ} + \text{РСДАО},$$

где РФОИ – размер файла основного индекса;
РСДАО – размер служебных данных в активных областях.

Размер архивного раздела PAP:



$$PAP = \text{SUM}(\text{Source}[i].\text{PerDayVolumeSizeEstimation}) * (\text{StorageDepth} - \text{ActiveStorageDepth}) / \text{CompressRatio},$$

Source[i].PerDayVolumeSizeEstimation – суточный объем данных по источникам;

StorageDepth – общая глубина хранения, задаваемая в настройках БД;

ActiveStorageDepth – глубина активного хранения, задаваемая в настройках БД;

CompressRatio – оценка коэффициента сжатия данных (1.0, если сжатие отключено).



В оценке размера архивного раздела из общей глубины вычтена глубина активного хранения, поскольку данные по мере переноса в архивный раздел удаляются из исходных активных разделов.

Размер хранимых данных активного раздела РХДАР:



$$РХДАР = \text{SUM}(\text{Align16MiB}(\text{Source}[i].\text{PerDayVolumeSizeEstimation})) * \text{ActiveStorageDepth},$$

Source[i].PerDayVolumeSizeEstimation – суточный объем данных по источникам;

ActiveStorageDepth – глубина активного хранения, задаваемая в настройках БД.



Оценка совокупного размера данных в активных областях основана на предположении, что резервированные источники сохраняют данные с взаимоисключением: в каждый момент времени сохранение выполняет активный источник.

Размер файла основного индекса РФОИ:



$$РФОИ = Align16MiB(SUM(Source[i].NumItems) * (4 + AvgItemIdSize)),$$

Source[i].NumItems – число элементов, по которым ведется сохранение;

AvgItemIdSize – средний размер идентификатора элемента.

Размер хранимых данных активного раздела РСДАО:



$$РСДАО = SUM(Source[i].RedundancyScaleFactor * (Align16MiB(Source[i].NumItems * 16) + Align16MiB(Source[i].NumItems * 4096))),$$

Source[i].RedundancyScaleFactor – множитель резервирования источника: 1 - не резервируется, 2 - 2-кратное резервирование, 3 - 3-кратное и т.д;

Source[i].NumItems – элементы, по которым ведется сохранение.



Оценка включает размер служебных данных.

В общем виде общий размер базы данных (ОРБД) определяется:



$$ОРБД = \frac{SUM(Source[i].PerDayVolumeSizeEstimation) * (StorageDepth - ActiveStorageDepth)}{CompressRatio} + SUM(Align16MiB(Source[i].PerDayVolumeSizeEstimation)) * ActiveStorageDepth + Align16MiB(SUM(Source[i].NumItems) * (4 + AvgItemIdSize)) + SUM(Source[i].RedundancyScaleFactor * (Align16MiB(Source[i].NumItems * 16) + Align16MiB(Source[i].NumItems * 4096))),$$

Source[i].PerDayVolumeSizeEstimation – суточный объем данных по источникам;

StorageDepth – общая глубина хранения, задаваемая в настройках БД;

ActiveStorageDepth – глубина активного хранения, задаваемая в настройках БД;

CompressRatio – оценка коэффициента сжатия данных (1.0, если сжатие отключено).

1.1.2.4. Оценка суточного объема данных от сервера

Оценка суточного объема данных от Astra.Server выполняется для вычисления суточного объема данных от источников `Source[i].PerDayVolumeSizeEstimation`.

Для получения оценки совокупного суточного размера исторических данных от Astra.Server необходимо сложить оценки суточного объема истории значений и истории событий. При этом необходимо учитывать, как ведется сохранение истории значений и алармов: совместно или отдельно (в зависимости от настроек Astra.Server/HistoryModule).

Размер идентификатора элемента `AvgItemIdSize`

В случае с Astra.Server полный идентификатор каждого отдельного элемента истории (для истории значений – отдельно на каждый сохраняемый тег, для истории алармов – один) имеет фиксированный размер 40 байт.

1.1.2.4.1. Оценка суточного объема истории значений

История значений сохраняется отдельно по каждому тегу (каждому тегу, сохраняемому в историю, в базе данных соответствует отдельный элемент и отдельный временной ряд, соответственно). Сохранение выполняется событийно с учетом настроек чувствительности по значению, метки времени и принудительной повторной записи.



Для оценки совокупного суточного объема истории значений необходимо сложить оценки по всем сохраняемым тегам.

Оценка суточного объема данных по отдельному тегу имеет следующий вид:



$$PerDayTagValueDataVolumeSize = 86400 * AvgTagFrequencyEstimation * (12 + TagValueRecordBodySize),$$

где 86400 – число секунд в сутках,

AvgTagFrequencyEstimation – оценка средней частоты сохранения значений по тегу в историю (после фильтрации по чувствительности и с учетом принудительного повторного сохранения);

TagValueRecordBodySize – размер тела записи.

1.1.2.4.2. Оценка суточного объема истории событий

Оценка совокупного суточного объема истории алармов, исходя из оценки среднего удельного размера на один аларм, может быть вычислена следующим образом:

f

$PerDayTagValueDataVolumeSize = 86400 * AvgTagFrequencyEstimation * AvgAlarmDataSizeEstimation,$

где 86400 – число секунд в сутках,

AvgTagFrequencyEstimation – оценка средней частоты сохранения значений по тегу в историю (после фильтрации по чувствительности и с учетом принудительного повторного сохранения);

AvgAlarmDataSizeEstimation – оценка среднего удельного размера данных на один аларм.

1.1.2.4.3. Пример расчета



Пусть источник Source - нерезервированный (RedundancyScaleFactor = 1) экземпляр Astra.Server, который сохраняет историю значений по 1000 тегам (Source.NumItems = 1000) типа double. В сутки в совокупности (по всем тегам) в среднем сохраняется 100 000 000 записей (в среднем ~1 157 записей/с). История алармов не сохраняется. Срок активного хранения (DbSettings.ActiveStorageDepth) – 3 суток, общий срок хранения (DbSettings.StorageDepth) – 1000 суток. Сжатие отключено (CompressRatio = 1.0). Размер полного идентификатора элемента (AvgItemIdSize) от Astra.Server – 40 байт.

Запись истории значений для типа double занимает в БД 25 байт. Соответственно, **оценка суточного объема данных в байтах от источника:**



$Source.PerDayVolumeSizeEstimation = 100\,000\,000 * 25 = 2\,500\,000\,000$
 $\approx 2\,384.2$ МиБ ≈ 2.3 ГиБ.

Общий размер базы данных ОРБД:



$ОРБД = РХД + РСД = 2\,384\,247,4 + 48 = 2\,384\,295,4$ МиБ $\approx 2\,328$ ГиБ $\approx 2,273$ ТиБ.

Размер хранимых данных РХД:



$РХД = РАР + РХДАР = 2\,377\,047,4 + 7200 = 2\,384\,247,4$ МиБ.

Размер архивного раздела РАР:

$$f \quad \text{РАР} = \text{Source.PerDayVolumeSizeEstimation} * (\text{DbSettings.StorageDepth} - \text{DbSettings.ActiveStorageDepth}) * \text{CompressRatio} = 2384,2 * (1000 - 3) / 1,0 = 2\,377\,047,4 \text{ МиБ.}$$

Размер хранимых данных активного раздела РХДАР:

$$f \quad \text{РХДАР} = \text{SUM}(\text{Align16MiB}(\text{Source.PerDayVolumeSizeEstimation})) * \text{DbSettings.ActiveStorageDepth} = \text{Align16MiB}(2384,2) * 3 = 2400 * 3 = 7200 \text{ МиБ.}$$

Размер служебных данных РСД:

$$f \quad \text{РСД} = \text{РФОИ} + \text{РСДАО} = 16 + 32 = 48 \text{ МиБ}$$

Размер файла основного индекса РФОИ:

$$f \quad \text{РФОИ} = \text{Align16MiB}(\text{SUM}(\text{Source.NumItems}) * (4 + \text{AvgItemIdSize})) = \text{Align16MiB}(1000 * (4 + 40)) = \text{Align16MiB}(44000) = 16 \text{ МиБ.}$$

Размер хранимых данных активного раздела РСДАО:

$$f \quad \text{РСДАО} = \text{SUM}(\text{Source.RedundancyScaleFactor} * (\text{Align16MiB}(\text{Source.NumItems} * 16) + \text{Align16MiB}(\text{Source.NumItems} * 4096))) = 1 * (\text{Align16MiB}(1000 * 16) + \text{Align16MiB}(1000 * 4096)) = 1 * (\text{Align16MiB}(16000) + \text{Align16MiB}(4096000)) = 1 * (16 + 16) = 32 \text{ МиБ.}$$



При использовании сжатия при среднем коэффициенте сжатия 3.5 крат, размер архивного раздела при тех же параметрах сократится:

 f

$$PAP = Source.PerDayVolumeSizeEstimation * (DbSettings.StorageDepth - DbSettings.ActiveStorageDepth) * CompressRatio = 2384,2 * (1000 - 3) / 3,5 = 679\ 156,4 \text{ МиБ.}$$

Тогда оценка совокупного размера БД будет следующей:

$$\text{ОРБД} = 679\ 156,4 + 7200 + 16 + 32 = 686\ 404,4 \text{ МиБ} \approx 671 \text{ ГиБ} \approx 0,655 \text{ ТиБ.}$$

1.1.2.5. Оценка объема временного хранилища очереди данных

В файловые очереди попадают записи, соответствующие параметру «Тип данных хранилища»: «история значений», «история событий», «история значений и событий», «имитационные данные». Если выбран вариант «история значений и событий», то в файловые очереди по этому хранилищу (для каждой БД она своя) пишутся и значения, и события.

Для оценки размера файловой очереди, в которой накоплены данные за промежуток времени (EQFS - Estimated Queue File Size), используется следующее выражение:

f

$$EQFS = Duration * (AVHRI * AVHRS + AEHRI * AEHRS) + SegmentSize,$$

где **Duration** – длительность временного интервала, за который данные накапливаются в файловой очереди;

AVHRS (Average Value History Record Size) – оценка среднего размера записи истории значений в файловой очереди;

AVHRI (Average Value History Record Intensity) – оценка средней интенсивности потока записей истории значений, сохраняемых в файловую очередь;

AEHRS (Average Event History Record Size) – оценка среднего размера записи истории событий в файловой очереди;

AEHRI (Average Event History Record Intensity) – оценка средней интенсивности потока записей истории событий, сохраняемых в файловую очередь;

SegmentSize – размер сегмента файловой очереди, на данный момент 32 МиБ;



Следует учитывать, что если для хранилища Astra.Historian настроено N баз данных, то приведенную оценку следует умножить на N, т.к. для каждой БД в текущей реализации создается отдельная файловая очередь.

Оценка среднего размера записи файловой очереди (AVHRS и AEHRS):



$AVHRS = QRHS + AvgValueHistoryBodySize;$
 $AEHRS = QRHS + AvgEventHistoryBodySize,$

где **QRHS** – заголовок записи файловой очереди константного размера 54 байта (содержит полный идентификатор элемента Astra.Server, по которому сохраняется запись + служебные данные);

AvgValueHistoryBodySize - средний размер сериализованного тела записи истории значений;

AvgEventHistoryBodySize - средний размер сериализованного тела записи истории событий.



На одно значение в файловой очереди при отключенной метке времени приходится примерно 60 байт, на одно событие – 350 байт.

1.1.2.6. Оценка объема заполнения дискового пространства

Вычисление объема заполнения дискового пространства за период времени хранения без сжатия:

f

$$D = \sum (r * T * v),$$

где D – суммарный объем данных без сжатия, байт.

r – размер одной записи тега, байт.

T – время хранения, с.

v – частота изменения значения тега, Гц.

Пример



10 тегов типа `double` ($r = 25$ байт) получают значения с частотой 1 Гц.
50 тегов типа `int4` ($r = 21$ байт) получают значения с частотой 2 Гц.

За 1 год диск заполнится на:

$$D = \sum (r * T * v) = 10 * (25 * 31\,536\,000 * 1) + 50 * (21 * 31\,536\,000 * 2) = 7884000000 + 66225600000 = 74109600000 \text{ байт} \approx 75 \text{ ГБ} \approx 69 \text{ ГиБ},$$

где $T = 86\,400$ (число секунд в 1 сутках) * 365 (число дней в году) = 31 536 000 с.



Приведенные расчеты применимы к расчету размера архива (содержимое папки `archive`). К фрагментам в активных областях (`active/{...}`) эти расчеты также можно применить, однако следует иметь в виду, что их размер выравняется вверх на 16 МиБ с целью оптимизации записи (для сокращения издержек на обновление метаданных файловой системы производится упреждающее увеличение размеров активных фрагментов блоками по 16 МиБ).



В приведенной методике расчета не учитывается размер дополнительных служебных данных, используемых для поддержки внутренней структуры фрагментов и индексации их содержимого. Издержки на хранение этих структур составляют не более 2% от полезного объема содержимого.



Следует учитывать, что все фрагменты базы данных имеют заголовок фиксированного размера 12 КБ.

Расчет дискового пространства для параметров, у которых не указана зона нечувствительности по времени зависит от средней совокупности интенсивности поступления записей.

Пример



Имеется 100 000 сигналов, по которым раз в секунду приходит 5000 значений и эти записи формата float4 ($r = 21$ байт). Рассчитать итоговый вес несжатых данных на диске за сутки.

$$D = \sum (r * T * v) = 5000 * (21 * 86400 * 1) = 9072000000 \text{ байт} = 9,1 \text{ ГБ} \approx 8,5 \text{ ГиБ},$$

где $T = 86\,400$ с (число секунд в 1 сутках).

Расчет дискового пространства для событий.

Типичный размер одного события примерно 200-300 байт (объем сильно зависит от размера сообщения, размера полного пути тега, по которому событие сработало).

Пример



Средний размер события составляет 300 байт, средняя генерации – 1 раз в секунду. Рассчитать итоговый вес несжатых данных на диске за сутки.

$$D = \sum (r * T * v) = \sum r * (T * v) = 300 * (86400 * 1) = 25920000 \text{ байт} \approx 26 \text{ МБ} \approx 24,72 \text{ МиБ},$$

где $\sum r$ – средний размер события;

$T = 86\,400$ с (число секунд в 1 сутках).

1.1.3. Диагностика работы

Для просмотра статистической информации Astra.Historian используется сервисное приложение Статистика.



Для подключения к серверу истории используется порт StatPort, указанный в файле конфигурации.

По умолчанию: 3388

Узел статистики Хранилище содержит статистическую информацию о работе сервера истории.

The screenshot shows the 'Статистика' application window. The left pane displays a tree view with the following structure:

- Historian на 127.0.0.1
 - Интерфейс статистики
 - Хранилище**
 - DB_EVENTS
 - DB_HISTORY
 - DB_LIMIT

The right pane displays a table with the following data:

| Имя | Значение |
|--|----------|
| Объём памяти, выделенный пулом, МБ | 8 |
| Использование диска, Б/с | 0 |
| Средний размер операции записи/чтения, Б | 4 096 |

| Параметр | Описание |
|------------------------------------|--|
| Объём памяти, выделенный пулом, МБ | Объём оперативной памяти, используемой в качестве кэша для работы с базами данных в текущий момент времени |
| Использование диска, Б/с | Нагрузка на диск. Рассчитывается средствами сервера истории. |

| | |
|-----------------------------------|---|
| Средний размер операции чтения, Б | Средний объем данных, запрашиваемый/записываемый в рамках одной операции чтения/записи с диска. |
|-----------------------------------|---|

Чтобы посмотреть статистические данные о подключенной БД, выберите узел статистики

Хранилище → <Название БД>.

| Имя | Значение |
|---|---|
| Состояние | База данных находится в рабочем состоянии |
| Глубина хранения с возможностью записи, сутки | 3 |
| Общая глубина хранения, сутки | 365 |
| Предельный объем базы данных, МБ (0 - не задан) | 2 000 |
| Количество открытых рабочих сессий | 1 |
| Количество открытых каналов обмена данными с очер... | 1 |
| Общее количество элементов | 19 818 |
| Количество активных элементов | 3 |
| Объем базы данных, МБ | 422 |
| Сохранение данных - текущая нагрузка, записей/с | 0 |
| Сохранение данных - максимальная нагрузка, записей... | 88 |
| Всего сохранено записей | 241 |
| Отброшено инверсных записей | 0 |
| Отброшено опережающих записей | 6 703 |
| Отброшено некорректных записей | 0 |
| Чтение данных - текущая нагрузка, записей/с | 0 |
| Чтение данных - максимальная нагрузка, записей/с | 2 |
| Средняя скорость чтения, записей/с | 0 |

| Параметр | Описание |
|---|--|
| Состояние | Состояние связи сервера истории с БД: БД находится в рабочем состоянии БД завершает работу БД не инициализирована |
| Глубина хранения с возможностью записи, сутки | Значение атрибута ActiveStorageDepth базы данных в файле конфигурации |
| Общая глубина хранения, сутки | Значение атрибута StorageDepth базы данных в файле конфигурации |

| | |
|--|--|
| Предельный объем базы данных, МБ | Значение атрибута VolumeLimit базы данных в файле конфигурации. 0 - не задан |
| Количество открытых рабочих сессий | Количество транзакций записи в БД, выполняемых в текущий момент времени |
| Количество открытых каналов обмена данными с очередью БД | Количество каналов, открытых клиентами на запись в БД. Канал открывается клиентом при первой транзакции записи в сервер истории (все дальнейшие транзакции записи выполняются клиентом в открытом канале), закрывается при потере соединения клиента с сервером истории (в этом случае для продолжения записи будет открыт новый канал) |
| Общее количество элементов | Количество элементов, по которым в БД хранится информация |
| Количество активных элементов | Количество элементов, участвующих в транзакциях записи в текущий момент времени |
| Объем базы данных, МБ | Текущий объем места на диске, занимаемого базой данных |
| Сохранение данных - текущая нагрузка, записей/с | Количество записей, сохраненных в БД за секунду в текущий момент времени |
| Сохранение данных - максимальная нагрузка, записей/с | Максимальное значение предыдущего параметра за время функционирования сервера истории |
| Отброшено инверсных записей | Количество записей, которые не были записаны в БД из-за устаревшей метки времени (метка времени записи меньше метки времени последней сохраненной записи этого элемента) |
| Отброшено опережающих записей | Количество записей, которые не были записаны в БД из-за метки времени, опережающей текущее время сервера истории |

| | |
|--|--|
| Отброшено некорректных записей | Количество некорректных записей, которые не были записаны в БД |
| Чтение данных - текущая нагрузка, записей/с | Количество записей, переданных в рамках всех транзакций чтения за секунду в текущий момент времени |
| Чтение данных - максимальная нагрузка, записей/с | Максимальное значение предыдущего параметра за время функционирования сервера истории |
| Средняя скорость чтения, записей/с | Средняя продолжительность выполнения запросов на чтение из БД |

Журнал работы

Для просмотра системных событий по компоненту используется приложение EventLogViewer.

The screenshot displays the Windows Event Viewer interface. The main window is titled "Журнал приложений" (Application Log) and contains a table of log entries. The table has three columns: "Источник" (Source), "Время" (Time), and "Сообщение" (Message). The entries are from the "Astra.Historian.Server" source, with timestamps ranging from 08:06:52 to 08:25:57 on 05.04.2023. The messages are identical: "Лицензия не обнаружена или ограничена. Использование Сервера Astr...".

On the right side, the "Настройка фильтра" (Filter Settings) panel is open. It shows a list of sources with checkboxes. The "Источники" (Sources) section is checked, and "Astra.Historian.Server" is selected. Other sources listed include AdobeARMService, Application Error, Astra.AStudio, Astra.Diagnostics.Hashcheck, Astra.Licensing.Agent, Astra.Server, Astra.Server.AeServer, Astra.Server.DaServer, Astra.Server.HistoryModule, Astra.Server.OpcUaClient, Astra.Server.SNMP Manager, Astra.Server.TcpServer, and Astra.Server.HsServer.

At the bottom of the log list, it says "Показаны 6306 событий из 31829" (Showing 6306 events out of 31829).

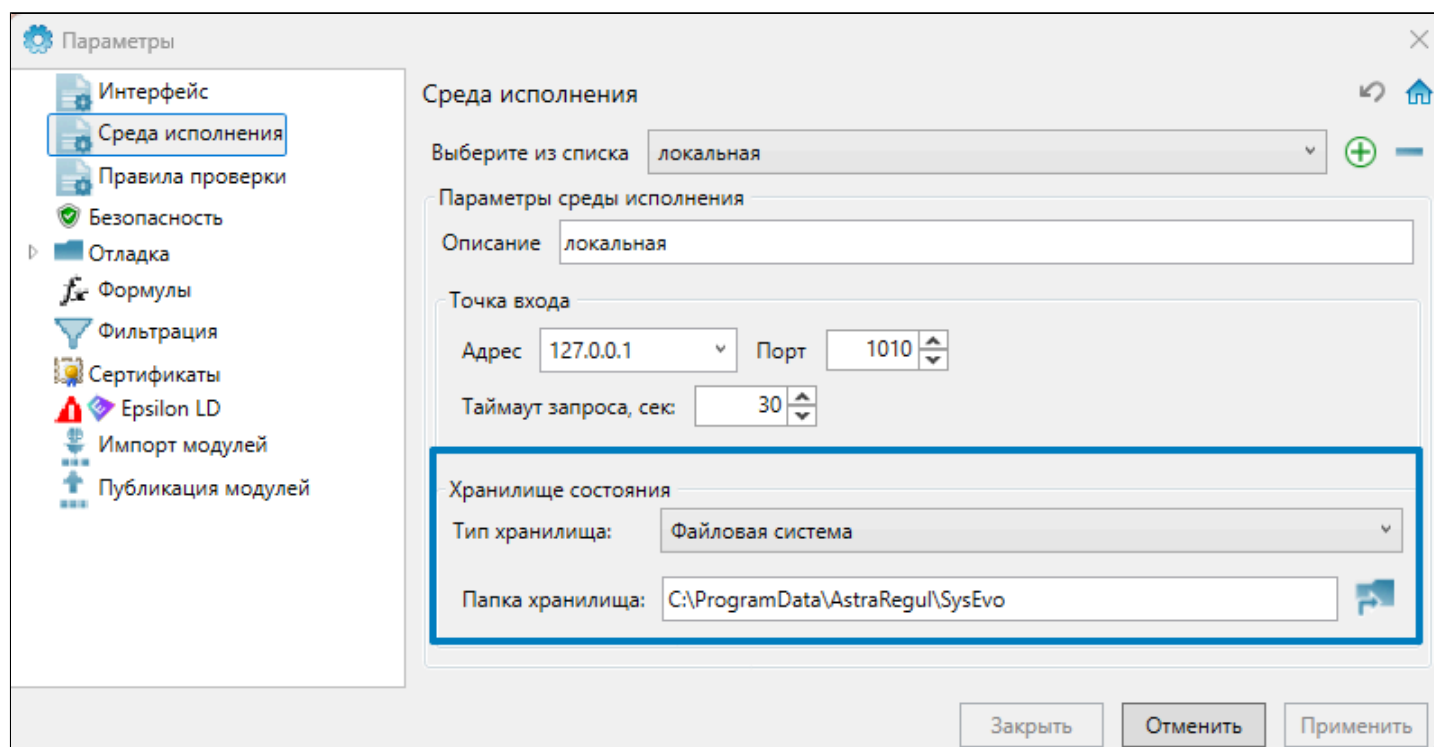
1.1.4. Администрирование

Данные сохраняются и запрашиваются в базу данных по внутренним идентификаторам сигналов.

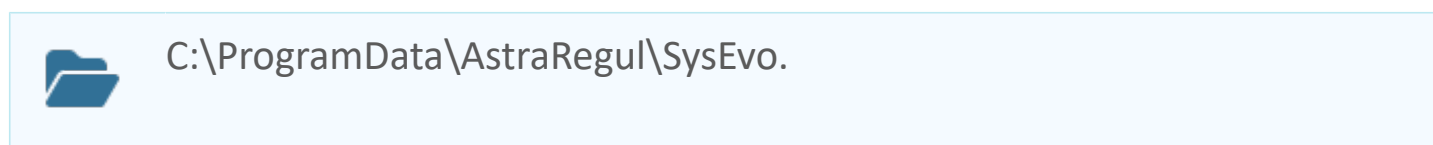
Чтобы после переноса/восстановления базы данных из нее можно было получить данные, необходимо:

- Сохранить проект Astra.AStudio обязательно вместе с хранилищем состояния.

Расположение хранилища состояния указано в Astra.AStudio: Меню Файл → Параметры → Среда исполнения.



По умолчанию:



- Сохранить конфигурацию экземпляра Astra.Server, который сохранял в нее данные.

После переноса/восстановления базы данных нужно применить сохраненную конфигурацию к тому экземпляру Astra.Server, который будет запрашивать сохраненные данные из базы данных.

1.1.4.1. Перенос баз данных

Чтобы перенести базу данных на другой компьютер:

1. У сервера истории, в который будет перенесена база данных, в файл конфигурации `Astra.Historian.Server.xml` добавьте описание новой базы данных.

Любые параметры новой базы данных могут отличаться от параметров исходной базы данных.



Если глубина хранения (атрибут `StorageDepth`) новой базы данных будет меньше, чем у исходной, то после переноса данные, которые окажутся старше новой глубины хранения, будут удалены.

2. Перезапустите службу (сервис) `Astra.Historian`.



При запуске сервер истории создаст структуру папок и файл новой базы данных.

3. Остановите службу (сервис) `Astra.Historian`, чтобы файл базы данных не был заблокирован для замены.

4. Скопируйте с заменой файл и папки исходной базы данных в новую базу данных в соответствии со структурой новой базы данных.

5. Если переносили базу данных в рамках одного сервера истории, из файла конфигурации удалите описание исходной базы данных.

6. Запустите службу (сервис) `Astra.Historian`.

1.1.4.2. Сохранение части архива базы данных

Архив или его части можно сохранить в отдельную папку или на внешний носитель информации. Сохраненные таким образом данные впоследствии можно восстановить.

Чтобы создать копию архивных данных:

1. Создайте папку.
2. Скопируйте в нее файл `db.index` из структуры базы данных.
3. Скопируйте в нее папку `archive` из структуры базы данных.

В ней данные сгруппированы в подпапки по датам и источникам, если источников более одного. В копии архива можно оставить папки только от интересующих вас источников и за нужные вам даты.

После копирования, из папки `archive` базы данных можно удалить скопированные подпапки, если для текущей работы они не нужны.



Файл `db.index` и папку `archive` базы данных удалять **нельзя**.

Чтобы дополнить созданную ранее копию архивных данных:

1. Скопируйте с заменой файл `db.index` из базы данных в копию архива.
2. В копию архива скопируйте подпапки с данными от интересующих вас источников и за нужные Вам даты.

1.1.4.3. Восстановление сохраненных архивных данных для просмотра

Чтобы восстановить сохраненные архивные данные для просмотра, выполните следующие действия:

1. В файл конфигурации сервера истории `Astra.Historian.Server.xml` добавьте новую базу данных.



Для нее не указывайте глубину хранения (по умолчанию – не ограничена) или укажите неограниченную глубину хранения: **`StorageDepth="0"`**. В противном случае сервер истории при запуске может удалить данные, имеющие большую глубину относительно текущей даты.

2. Перезапустите службу (сервис) `Astra.Historian`.

В результате в файловой системе будут созданы файл и папки для новой базы данных.

3. Остановите службу (сервис) `Astra.Historian`.

4. Замените в папке базы данных файл `db.index` одноименным файлом из сохраненного архива.

5. Из папки `archive` сохраненного архива скопируйте папки интересующих вас источников и за нужные для просмотра даты в одноименную папку базы данных.

6. Запустите службу (сервис) `Astra.Historian`.

После этого восстановленные архивные данные можно запрашивать из добавленной базы данных.

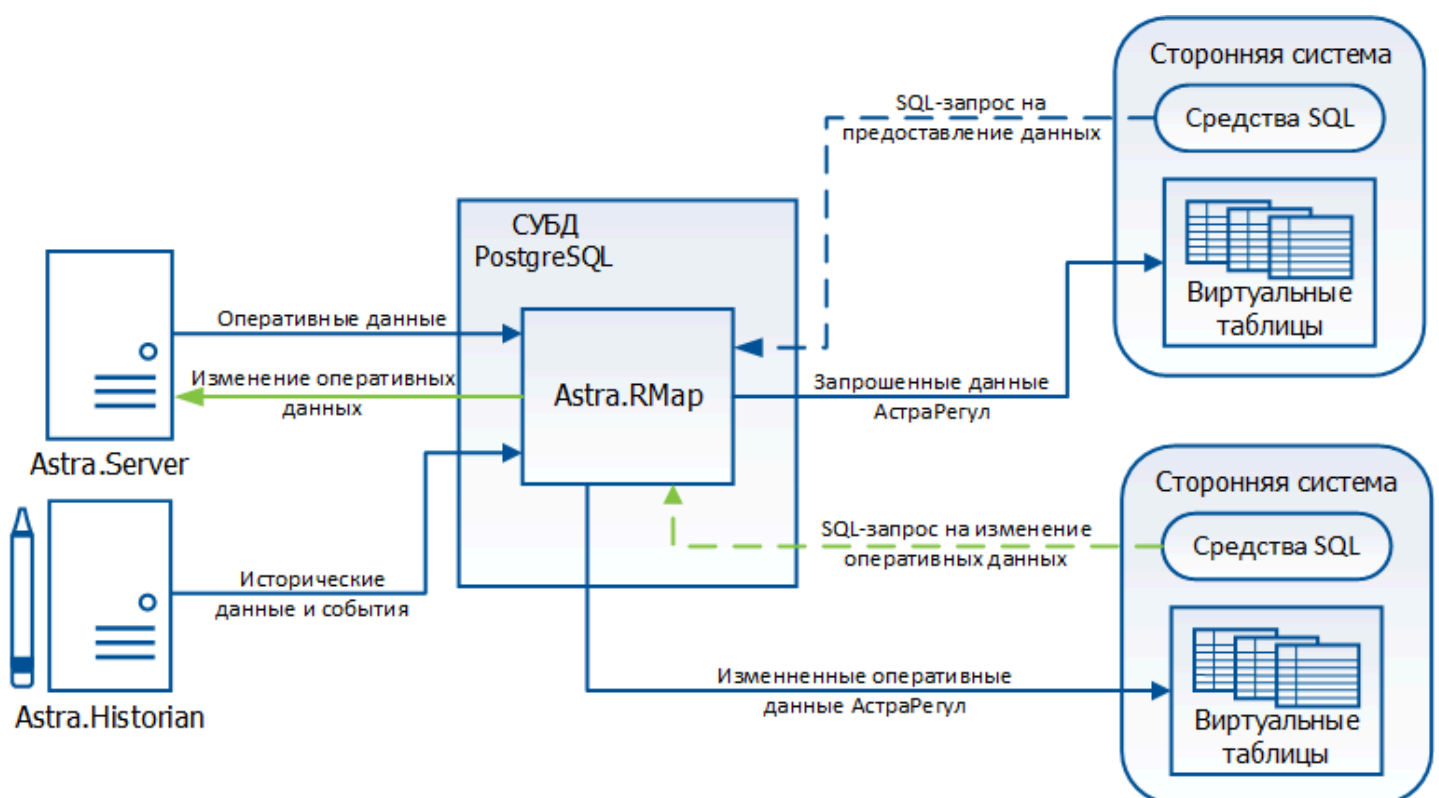
1.2. Astra.RMap

Astra.RMap – расширение СУБД PostgreSQL, с помощью которого возможно представление оперативных значений, истории значений и событий в реляционном виде.

Компонент предоставляет SQL-доступ к данным ПТК AstraRegul с возможностью изменения оперативных значений.

Оперативные значения для Astra.RMap предоставляет Astra.Server, историю значений и событий - Astra.Historian.

Полученные данные Astra.RMap предоставляет СУБД PostgreSQL. СУБД PostgreSQL предоставляет полученные данные сторонним системам (например, Microsoft Excel, 1С, системам отчетности, SQL-клиентам и т.п.), в которых данные отображаются в виде виртуальных таблиц.



1.2.1. Настройка

[Windows](#)

[AstraLinux](#)

[РЕД ОС](#)

1.2.1.1. Windows

Настройка драйвера ODBC

Настройку драйвера ODBC для PostgreSQL выполните в следующем порядке:

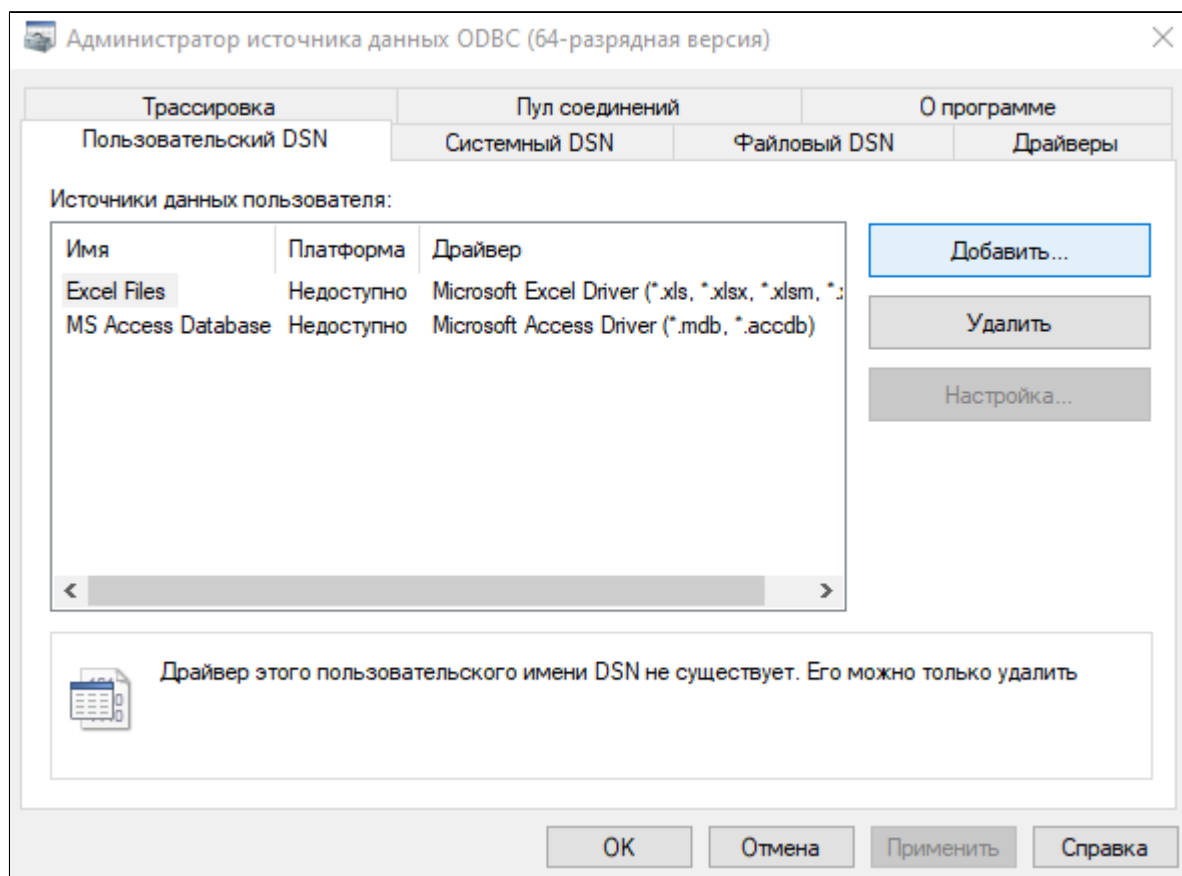
1. Выполните команду Пуск → Служебные → Панель управления → Система безопасности → Администрирование → Источники данных ODBC. Аналогичное действие можно выполнить, запустив файл `odbcad32.exe`, расположенный в папке `C:\Windows\System32`.

Приведенные способы актуальны в случае если разрядность ОС Windows совпадает с разрядностью программного продукта, в который будут предоставляться данные.

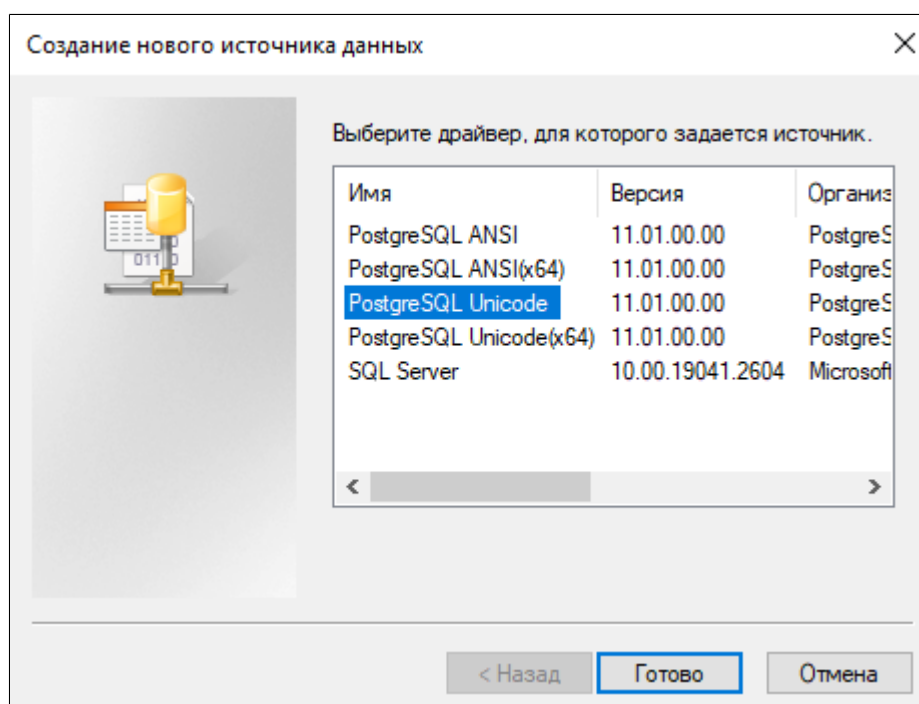


Если разрядность ОС Windows - 64 бита, а разрядность приложения - 32 бита, запустите файл `odbcad32.exe`, расположенный в папке `C:\Windows\SysWOW64`

2. В окне Администратор источников данных ODBC перейдите на вкладку Пользовательский DSN и нажмите кнопку Добавить...

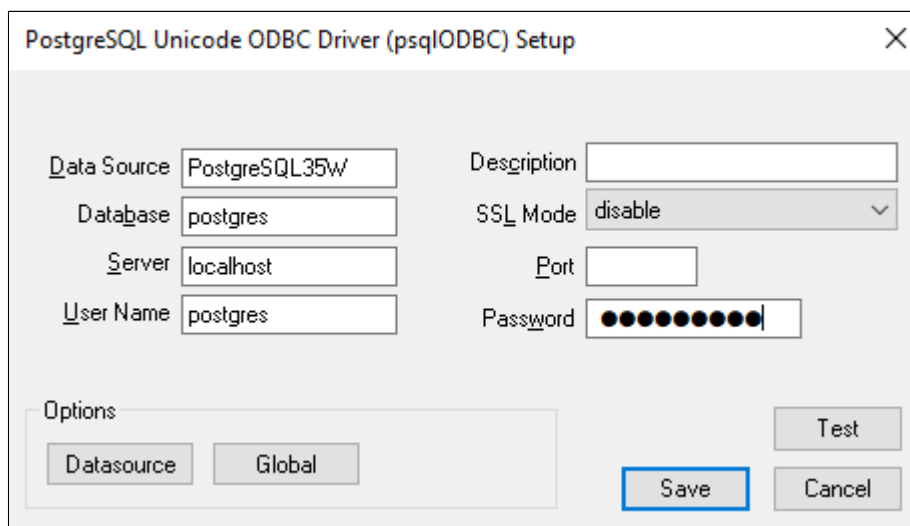


В появившемся окне Создание нового источника данных выберите драйвер «PostgreSQL Unicode» и нажмите кнопку Готово.

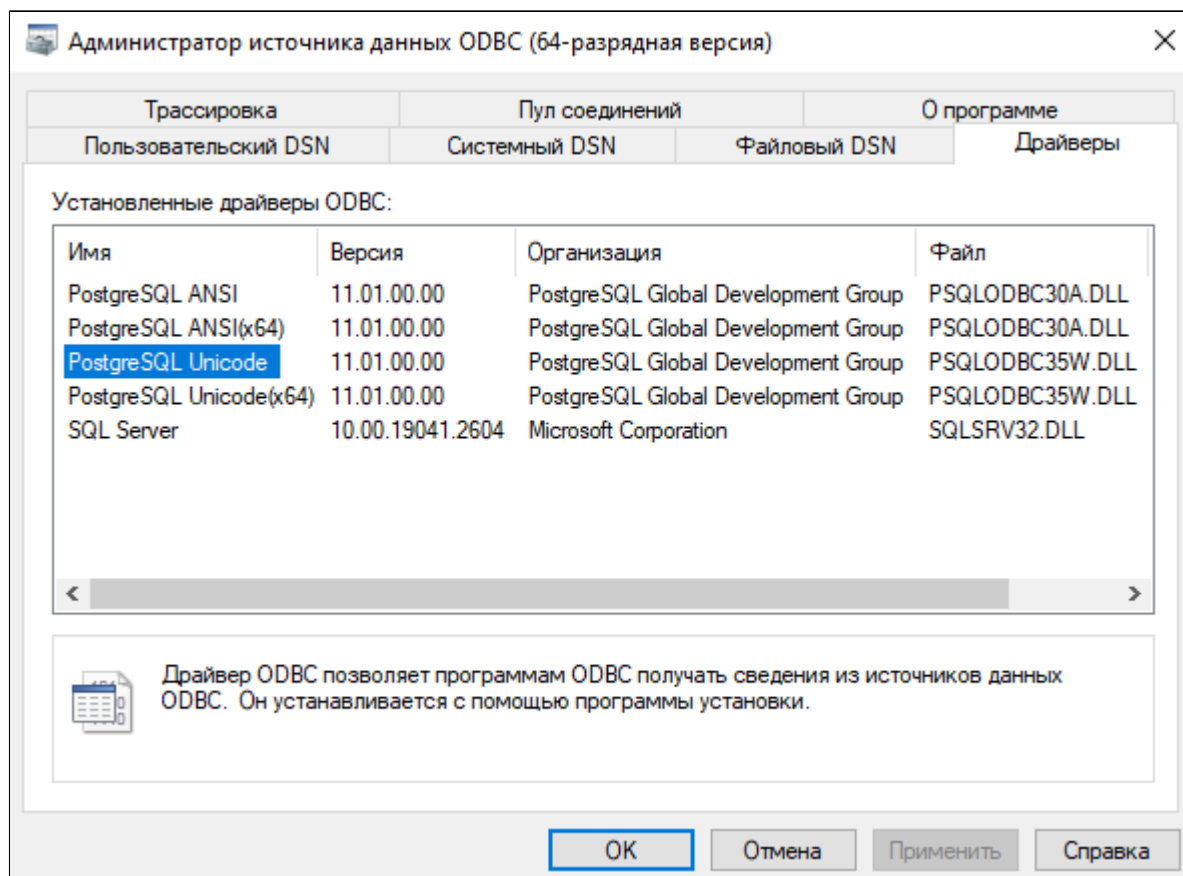


3. В появившемся окне PostgreSQL Unicode ODBC Driver (psqlODBC) Setup укажите:

- › Database - имя базы данных PostgreSQL;
- › Server - адрес сервера;
- › User Name - имя пользователя для подключения к серверу PostgreSQL;
- › Password - пароль пользователя.



4. После произведенных настроек драйвер для Postgres будет готов к работе:



Настройка Astra.RMAP



Настройки для работы Astra.RMap указываются в файле скрипта `init_rmap_history.sql`, который при выполнении формирует виртуальные таблицы для отображения данных.

1. Откройте файл скрипта `init_rmap_history.sql` любым текстовым редактором.

Каталог по умолчанию:



`C:\Program Files\AstraRegul\Astra.RMap\Initialize`

2. Если пользователей СУБД несколько, пропишите команды SQL `CREATE USER MAPPING` для каждого пользователя:

```
init_rmap_history — Блокнот
Файл  Правка  Формат  Вид  Справка
ActorId      text NOT NULL,
AckComment   text
)
SERVER history_server;

CREATE USER MAPPING FOR "postgres"
SERVER history_server;

CREATE USER MAPPING FOR "admin"
SERVER history_server;

CREATE USER MAPPING FOR "user"
SERVER history_server;
```


1.2.1.2. AstraLinux

Настройка драйвера ODBC

1. Откройте файл конфигурации драйвера с помощью команды



```
sudo nano /etc/odbcinst.ini
```

```
user@user:~$ sudo nano /etc/odbcinst.ini
```

2. Файл должен содержать следующие записи о драйверах PostgreSQL ANSI (если требуется кодировка ANSI) и PostgreSQL Unicode (если требуется кодировка Unicode).



```
[PostgreSQL ANSI]
```

```
Description=PostgreSQL ODBC driver (ANSI version)
```

```
Driver=/usr/lib/x86_64-linux-gnu/odbc/psqlodbc.a.so
```

```
Setup=libodbcpsqlS.so
```

```
Debug=0
```

```
CommLog=1
```

```
UsageCount=1
```

```
[PostgreSQL Unicode]
```

```
Description=PostgreSQL ODBC driver (Unicode version)
```

```
Driver=/usr/lib/x86_64-linux-gnu/odbc/psqlodbcw.a.so
```

```
Setup=libodbcpsqlS.so
```

```
Debug=0
```

```
CommLog=1
```


```
UsageCount=1
```

Если их нет, то создайте их.

```
[PostgreSQL ANSI]
Description=PostgreSQL ODBC driver (ANSI version)
Driver=psqlodbc.a.so
Setup=libodbcpsqlS.so
Debug=0
CommLog=1
UsageCount=1

[PostgreSQL Unicode]
Description=PostgreSQL ODBC driver (Unicode version)
Driver=/usr/lib/x86_64-linux-gnu/odbc/psqlodbcw.so
Setup=libodbcpsqlS.so
Debug=0
CommLog=1
UsageCount=1
```

3. Создайте подключение к базе данных. Откройте файл `odbc.ini` с помощью команды :

 `sudo nano /etc/odbc.ini`

```
user@user:~$ sudo nano /etc/odbc.ini
```

Файл должен содержать следующие записи.



```
[PostgreSQL-connector]
Description      = PostgreSQL connection
Driver           = PostgreSQL Unicode
Database         = user_db //укажите имя БД
Servername       = localhost
Username         = user
Password         = MyPassword1 //укажите пароль
Port             = 5432
ReadOnly         = No
RowVersioning   = No
ShowSystemTables = No
ShowOidColumn   = No
FakeOidIndex     = No
ConnSettings     =
```

Если их нет, то создайте их.

```
[PostgreSQL-connector]
Description = PostgreSQL connection
Driver = PostgreSQL Unicode
Database = postgres
Servername = localhost
Username = test_user
Password = 123
Port = 5432
Protocol = 9.6
ReadOnly = No
RowVersioning = No
ShowSystemTables = No
ShowOidColumn = No
FakeOidIndex = No
ConnSettings =
```

4. Откройте файл `pg_hba.conf` с помощью команды:



```
sudo nano /etc/postgresql/11/main/pg_hba.conf
```

5. Файл `pg_hba.conf` должен содержать следующую запись:



```
host all all ::1/128 md5
```



Если данная запись отсутствует, ее необходимо добавить в конец файла и перезапустить PostgreSQL.

```
local all postgres peer
# TYPE DATABASE USER ADDRESS METHOD
# "local" is for Unix domain socket connections only
local all all peer
# IPv4 local connections:
#host all all 127.0.0.1/32 md5
#host all all 0.0.0.0/0 md5
# IPv6 local connections:
host all all ::1/128 md5
# Allow replication connections from localhost, by a user with the
# replication privilege.
local replication all peer
host replication all 127.0.0.1/32 md5
host replication all ::1/128 md5
host mydb1 test_user all md5
```

6. После этого перезапустите PostgreSQL с помощью команды:



```
sudo systemctl restart postgresql.service
```

```
user@user:~$ sudo systemctl restart postgresql.service
user@user:~$ █
```

Настройка Astra.RMAP



Настройки для работы Astra.RMap указываются в файле скрипта `init_rmap_history.sql`, который при выполнении формирует виртуальные таблицы для отображения данных.

1. Произведите настройку Astra.RMap для этого откройте файл `init_rmap_history.sql` с помощью следующей команды.



```
sudo nano /usr/share/postgresql/11/extension/init_rmap_history.sql
```

```
user@user:~$ sudo nano /usr/share/postgresql/11/extension/init_rmap_history.sql
user@user:~$ █
```

Если пользователей СУБД несколько, пропишите команды SQL "CREATE USER MAPPING" для каждого пользователя:

```
CREATE USER MAPPING FOR "postgres"
SERVER history_server;

CREATE USER MAPPING FOR "admin"
SERVER history_server;

CREATE USER MAPPING FOR "user"
SERVER history_server; █
```

2. Настройте подключение к Astra.Server:

```
CREATE SERVER history_server TYPE 'astra.rmap.historian.rprovider'
FOREIGN DATA WRAPPER rmap_fdw
OPTIONS (
    SourceLocation '127.0.0.1',
    --SourceDB 'Astra.OPCDA_Server',
    SourceTcpPort '4388',
    HistorianLocation '127.0.0.1',
    --HistorianName 'Astra.Historian.Server',
    HistorianTcpPort '4949',
    HistorianDB 'default',
    --SourceAE 'Astra.OPCAE_Server',
    SourceAETcpPort '4388',
    AeHistorianLocation '127.0.0.1',
    --AeHistorianName 'Astra.Historian.Server',
    AeHistorianTcpPort '4949',
    AeHistorianDB 'aebase'
);
```

› SourceLocation – адрес Astra.Server;

› SourceTcpPort и SourceAETcpPort – порт модуля TCP Server, указанный в параметре конфигурации модуля Номер TCP порта. По умолчанию порт 4388.

3. Настройте подключение к Astra.Historian:

```
CREATE SERVER history_server TYPE 'astra.rmap.historian.rprovider'  
FOREIGN DATA WRAPPER rmap_fdw  
OPTIONS (  
    SourceLocation '127.0.0.1',  
    --SourceDA 'Astra.OPCDA Server',  
    SourceTcpPort '4388',  
    HistorianLocation '127.0.0.1',  
    --HistorianName 'Astra.Historian.Server',  
    HistorianTcpPort '4949',  
    HistorianDB 'default',  
    --SourceAE 'Astra.OPCAE Server',  
    SourceAETcpPort '4388',  
    AeHistorianLocation '127.0.0.1',  
    --AeHistorianName 'Astra.Historian.Server',  
    AeHistorianTcpPort '4949',  
    AeHistorianDB 'aebase'  
);
```

- › HistorianLocation и HistorianTcpPort – адрес и порт Astra.Historian, в котором хранятся значения;
- › HistorianDB – имя базы данных, в которой хранятся значения;
- › AeHistorianLocation и AeHistorianTcpPort – адрес и порт Astra.Historian, в котором хранятся события;
- › AeHistorianDB – имя базы данных, в которой хранятся события.

Символы комментирования "--" перед атрибутами HistorianTcpPort и AeHistorianTcpPort должны отсутствовать.

Атрибуты HistorianName и AeHistorianName должны быть закомментированы символами "--".

4. Для получения истории с помощью модуля TCP Server в значениях атрибутов HistorianTcpPort и AeHistorianTcpPort укажите порт доступа к истории, заданный в настройках модуля TCP Server в Astra.AStudio в свойстве Номер TCP порта группы Настройки доступа к истории.

| Настройки доступа к истории | |
|------------------------------|------|
| Номер TCP порта | 4949 |
| Время удержания сессии | 10 |
| Ограничение ожидающих сессий | 64 |

5. Если в настройках модуля порт не задан, то установите в соответствующем параметре модуля любой свободный порт.

```
CREATE SERVER history_server TYPE 'astra.rmap.historian.rprovider'  
FOREIGN DATA WRAPPER rmap_fdw  
OPTIONS (  
    SourceLocation '127.0.0.1',  
    --SourceDA 'Astra.OPCDAServer',  
    SourceTcpPort '4388',  
    HistorianLocation '127.0.0.1',  
    --HistorianName 'Astra.Historian.Server',  
    HistorianTcpPort '4949',  
    HistorianDB 'default',  
    --SourceAE 'Astra.OPCAEServer',  
    SourceAETcpPort '4388',  
    AeHistorianLocation '127.0.0.1',  
    --AeHistorianName 'Astra.Historian.Server',  
    AeHistorianTcpPort '4949',  
    AeHistorianDB 'aebase'  
);
```

6. Запустите консольный клиент PostgreSQL. Для этого от суперпользователя root выполните команду:



```
sudo -u postgres psql
```

```
user@user:~$ sudo -u postgres psql  
could not change directory to "/home/user": Отказано в гоступе  
psql (11.17 (Debian 11.17-astra.sel+b1))  
Type "help" for help.
```

```
postgres=#
```

7. Выполните команду инициализации скрипта для настройки Astra.RMAP:



```
\i /usr/share/postgresql/x.x/extension/init_rmap_history.sql
```

```
postgres=# \i /usr/share/postgresql/11/extension/init_rmap_history.sql  
psql:/usr/share/postgresql/11/extension/init_rmap_history.sql:1: ЗАМЕЧАНИЕ: расширение "rmap_fdw" не существует, пропускается  
DROP EXTENSION  
CREATE EXTENSION  
CREATE SERVER  
CREATE FOREIGN TABLE  
CREATE FOREIGN TABLE  
CREATE FOREIGN TABLE  
CREATE FOREIGN TABLE  
CREATE FOREIGN TABLE  
CREATE FOREIGN TABLE  
CREATE FOREIGN TABLE  
CREATE FOREIGN TABLE  
CREATE FOREIGN TABLE  
CREATE USER MAPPING  
postgres=#
```

1.2.1.3. РЕД ОС

Настройка драйвера ODBC

1. Откройте файл конфигурации драйвера с помощью команды:

```
sudo nano /etc/odbcinst.ini
```

2. Файл должен содержать следующие записи о драйверах PostgreSQL ANSI (если требуется кодировка ANSI) и PostgreSQL Unicode (если требуется кодировка Unicode).



```
[PostgreSQL]  
Description=ODBC for PostgreSQL  
Driver=/usr/pgsql-x.x/lib/psqlodbcw.so  
Setup=/usr/lib64/libodbcpsqlS.so  
FileUsage=1
```



Если их нет, то создайте их.

3. Создайте подключение к базе данных. Откройте файл `odbc.ini` с помощью команды:

```
sudo nano /etc/odbc.ini
```


Файл должен содержать следующие записи.



```
[PostgreSQL-connector]
Description = PostgreSQL connection
Driver = PostgreSQL Unicode
Database = user_db //укажите имя БД
Servername = localhost
Username = user
Password = MyPassword1 //укажите пароль
Port = 5432 ReadOnly = No
RowVersioning = No
ShowSystemTables = No
ShowOidColumn = No
FakeOidIndex = No
ConnSettings =
```



Если их нет, то создайте их.

4. Откройте файл `pg_hba.conf` с помощью команды:



```
sudo nano /var/lib/pgsql/11/data/pg_hba.conf
```

5. Файл `pg_hba.conf` должен содержать следующую запись:



```
host    all    all    ::1/128    md5
```



Если данная запись отсутствует, ее необходимо добавить в конец файла и перезапустить PostgreSQL.

6. После этого перезапустите PostgreSQL с помощью команды:



```
sudo systemctl restart postgresql-11.service
```

7. Выполните команду инициализации скрипта для настройки Astra.RMAP:



```
\i /usr/pgsql-11/share/extension/init_rmap_history.sql
```

1.2.2. Виртуальные таблицы

При выполнении скрипта `init_rmap_history.sql` формируются виртуальные таблицы для отображения данных.

Типы данных

| Тип данных | Описание |
|-------------------------------|---|
| <code>bigint</code> | Знаковое целое из 8 байт $[-9 \times 10^{-18}; 9 \times 10^{18}]$ |
| <code>boolean</code> | Логическое значение <code>[true; false]</code> |
| <code>double precision</code> | Значение с плавающей точкой из 8 байт $[\pm 5.0 \times 10^{-324}; \pm 1.7 \times 10^{308}]$, точность 15-17 цифр |
| <code>integer</code> | Знаковое целое из 4 байт <code>[-2 147 483 648; 2 147 483 647]</code> |
| <code>smallint</code> | Знаковое целое из 2 байт <code>[32 768; 32 767]</code> |
| <code>text</code> | Символьная строка переменной длины |
| <code>timestamp</code> | Дата и время |

1.2.2.1. Таблицы данных

К виртуальным таблицам данных Astra.RMap относятся:

- › nodes – сигналы;
- › nodes_values – оперативные значения;
- › nodes_history – история значений;
- › nodes_attributes – свойства сигналов.

Связи таблиц

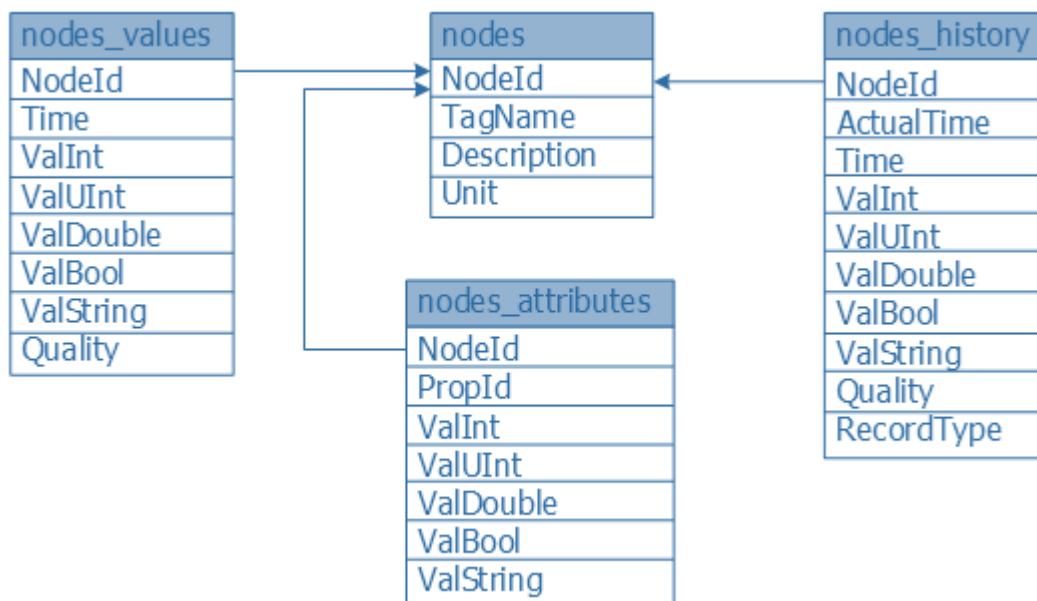
В таблицах nodes_values, nodes_history и nodes_attributes поля NodeId являются внешними ключами и ссылаются на таблицу nodes.



При подключении Astra.RMap к Astra.AccessPoint по TCP, Appld соответствует идентификатору источника, с которого Astra.AccessPoint получает данные.



При подключении по DCOM Appld соответствует идентификатору источника, к которому подключается Astra.RMap.



nodes

Отображает сигналы, по которым возможен просмотр данных.

| Столбец | Тип данных | Описание |
|-------------|------------|-------------------------|
| NodId | bigint | Идентификатор сигнала |
| AppId | bigint | Идентификатор источника |
| TagName | text | Имя сигнала |
| Description | text | Описание сигнала |
| Unit | text | Единицы измерения |

nodes_values

Отображает оперативные значения сигналов.

| Столбец | Тип данных | Описание |
|-----------|------------|---------------------------------------|
| NodId | bigint | Идентификатор сигнала |
| AppId | bigint | Идентификатор источника |
| Time | timestamp | Дата и время по UTC |
| ValInt | bigint | Значение сигнала (знаковое) |
| ValUInt | bigint | Значение сигнала (беззнаковое) |
| ValDouble | double | Значение сигнала (с плавающей точкой) |
| ValBool | boolean | Значение сигнала (логическое) |
| ValString | text | Значение сигнала (текстовое) |
| Quality | integer | Качество |



В зависимости от типа данных значение сигнала отображается в соответствующем столбце.

nodes_history

Отображает историю значений сигналов.

| Столбец | Тип данных | Описание |
|------------|------------|--|
| Nodeld | bigint | Идентификатор сигнала |
| Appld | bigint | Идентификатор источника |
| ActualTime | timestamp | Действительная метка времени сигнала. Для всех значений внутри временного интервала совпадает с полем Time, для граничных точек содержит действительную метку времени границы |
| Time | timestamp | Дата и время |
| ValInt | bigint | Значение сигнала (знаковое) |
| ValUInt | bigint | Значение сигнала (беззнаковое) |
| ValDouble | double | Значение сигнала (с плавающей точкой) |
| ValBool | boolean | Значение сигнала (логическое) |
| ValString | text | Значение сигнала (текстовое) |
| Quality | integer | Качество |
| RecordType | text | Запрашиваемый тип значения. Принимаемые значения: <ul style="list-style-type: none">› lbound - значение является левой граничной точкой› ubound - значение является правой граничной точкой› inner - значение является точкой внутри интервала |



В зависимости от типа данных значение сигнала отображается в соответствующем столбце.

nodes_attributes

Отображает свойства сигналов.

| Столбец | Тип данных | Описание |
|-----------|------------|--|
| Nodeld | bigint | Идентификатор сигнала |
| Appld | bigint | Идентификатор источника |
| PropId | bigint | Идентификатор свойства сигнала |
| ValInt | bigint | Значение свойства (знаковое) |
| ValUInt | bigint | Значение свойства (беззнаковое) |
| ValDouble | double | Значение свойства (с плавающей точкой) |
| ValBool | boolean | Значение свойства (логическое) |
| ValString | text | Значение свойства (текстовое) |



В зависимости от типа данных значение свойства сигнала отображается в соответствующем столбце.



Astra.RMap преобразует значения свойства CDT (Канонический тип данных, PropId = 1), получаемые от Astra.Server, к собственным кодам типов.

В таблице приведено соответствие значений свойства 1 (CDT) кодам типов в Astra.RMap:

| Тип сигнала | Int1 | UInt1 | Int2 | UInt2 | Int4 | UInt4 | Int8 | UInt8 | Float | Double | Bool | String | Time |
|------------------|------|-------|------|-------|------|-------|------|-------|-------|--------|------|--------|------|
| Значение 1 (CDT) | 1 | 3 | 9 | 8 | 7 | 6 | 13 | 12 | 14 | 15 | 5 | 17 | 18 |
| Код | 16 | 17 | 2 | 18 | 3 | 19 | 20 | 21 | 4 | 5 | 11 | 8 | 7 |

1.2.2.2. Таблицы событий

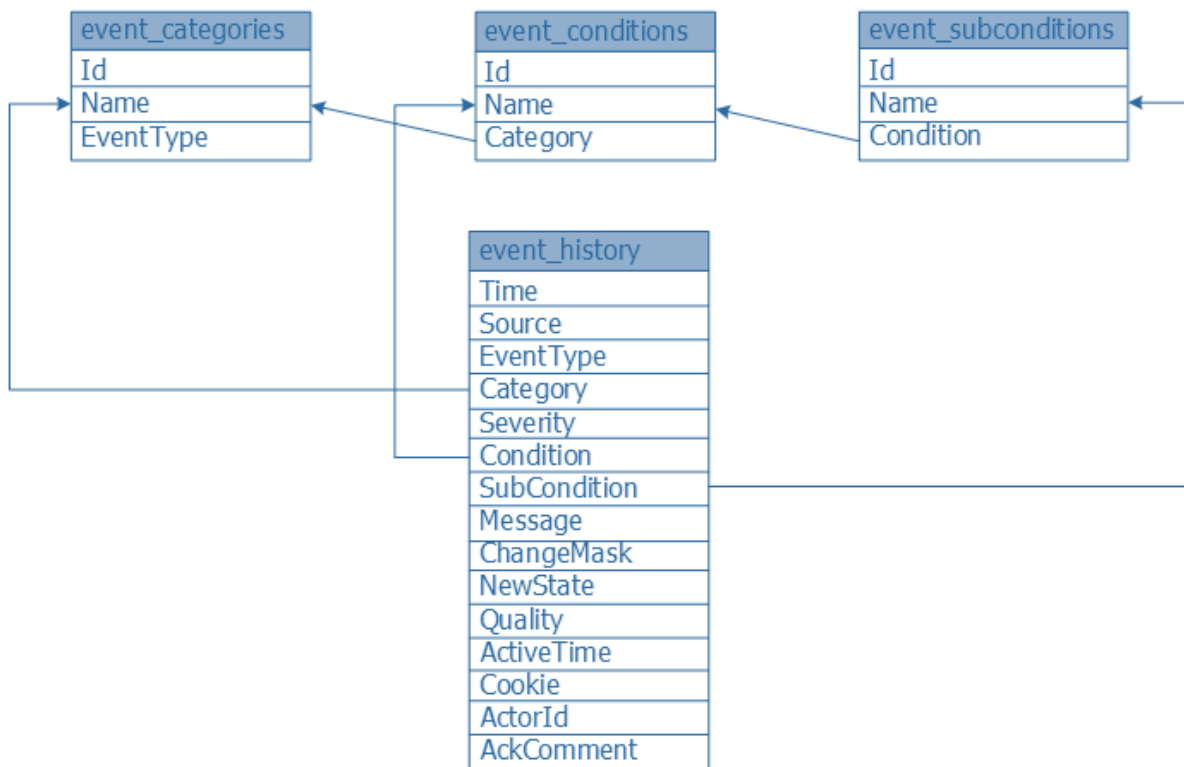
К виртуальным таблицам событий Astra.RMap относятся:

- › event_categories – доступные категории событий;
- › event_conditions – доступные условия событий;
- › event_subconditions – доступные подусловия событий;
- › event_history – история событий.

Связи таблиц

В таблице event_history поле Category является внешним ключом и ссылается на поле Name таблицы event_categories. Аналогично, поля Condition и SubCondition ссылаются на поля Name таблиц event_conditions и event_subconditions соответственно.

В таблице event_subconditions поле Condition ссылается на поле Name таблицы event_conditions, в которой поле Category ссылается на поле Name таблицы event_categories.



event_categories

Отображает доступные категории событий.

| Столбец | Тип данных | Описание |
|-----------|------------|---|
| Id | bigint | Идентификатор категории событий |
| Name | text | Название категории событий |
| EventType | text | Тип события, к которому относится категория |

event_conditions

Отображает доступные условия событий.

| Столбец | Тип данных | Описание |
|----------|------------|--|
| Id | bigint | Идентификатор условия событий |
| Name | text | Название условия событий |
| Category | text | Категория, к которой относится условие |

event_subconditions

Отображает доступные подусловия событий.

| Столбец | Тип данных | Описание |
|-----------|------------|--|
| Id | bigint | Идентификатор подусловия событий |
| Name | text | Название подусловия событий |
| Condition | text | Условие генерации события, к которому относится подусловие |

event_history

Отображает историю событий.

| Столбец | Тип данных | Описание |
|--------------|------------|--|
| Time | timestamp | Время, когда сгенерировано уведомление о событии |
| Source | text | Полное имя сигнала, от которого произошло событие |
| EventType | text | Тип события |
| Category | text | Категория события |
| Severity | integer | Приоритет – важность данного события |
| Condition | text | Имя условия генерации события |
| SubCondition | text | Имя подусловия генерации события |
| Message | text | Сообщение – текст, который описывает событие |
| ChangeMask | smallint | Маска изменений |
| NewState | smallint | Индикатор нового состояния условия |
| Quality | integer | Качество |
| ActiveTime | timestamp | Время возникновения события |
| Cookie | integer | Текстовая информация о событии, хранящаяся в сервере |
| ActorId | text | Идентификатор OPC клиента, который квитирует событие |
| AckComment | text | Комментарий квитирования |
| NodeId | bigint | Идентификатор сигнала |
| AppId | bigint | Идентификатор источника |

1.2.3. SQL-запросы

Общие рекомендации по построению SQL-запросов

1. При построении SQL-запросов с объединением таблиц рекомендуется использовать оператор "JOIN" без дополнительных операндов. Общий вид запроса:



```
SELECT {столбцы} FROM {таблица_1} JOIN {таблица_2} ON  
{условие_объединения} WHERE {условие_запроса}
```

2. Объединять таблицы рекомендуется в следующем порядке:

- в качестве {таблица_1} использовать таблицы nodes_values при запросе оперативных значений и nodes_history при запросе истории значений;
- в качестве {таблица_2} использовать таблицу nodes.

3. Для получения оперативных значений или истории значений по имени сигнала рекомендуется использовать запрос следующего вида:



```
SELECT {столбцы} FROM {таблица_1} JOIN nodes ON  
nodes.NodeId={таблица_1}.NodeId WHERE nodes.TagName =  
'{Имя_сигнала}'
```

SQL-запросы к таблицам данных

Для изменения оперативных значений используется запрос "UPDATE" к таблице nodes_values. Например, запрос на изменение значения сигнала "NPS.BIK172_6.BB97n.Cmd" на значение "10" типа "integer" имеет вид:



```
UPDATE nodes_values SET valint=10 FROM nodes WHERE  
nodes.nodeid=nodes_values.nodeid AND nodes.TagName =  
'NPS.BIK172_6.BB97n.Cmd'
```



Если новое значение сигнала не совпадает с типом сигнала, происходит попытка преобразования типа. Если преобразование невозможно, запрос завершается ошибкой.

Для вывода данных таблиц используется запрос "SELECT". Например, запрос на получение текущего значения сигнала "NPS.BIK172_6.BB97n.Cmd" имеет вид:



```
SELECT * FROM nodes_values JOIN nodes ON nodes.NodeId =  
nodes_values.NodeId WHERE  
nodes.TagName='NPS.BIK172_6.BB97n.Cmd'
```

Запрос на получение истории значений сигнала "NPS.BIK172_6.BB97n.Cmd" имеет вид:



```
SELECT * FROM nodes_history JOIN nodes ON  
nodes.NodeId = nodes_history.NodeId WHERE nodes.TagName =  
'NPS.BIK172_6.BB97n.Cmd'
```

SQL-запросы к таблицам событий

Для вывода данных таблиц используется запрос "SELECT". Например, запрос на получение истории событий, у которых условия генерации события "DISCRETE", имеет вид:



```
SELECT * FROM event_history JOIN event_conditions ON  
event_history.Condition=event_conditions.Name WHERE  
Name='DISCRETE'
```

1.2.3.1. Примеры

1. Запрос на получение истории изменения значений сигнала "NPS.MNS1.PT001_1.Value".



```
SELECT * FROM nodes_history h JOIN nodes n ON n.NodeId=h.NodeId  
WHERE n.TagName='NPS.MNS1.PT001_1.Value'
```

Результат запроса в MS Excel (значения сигнала отображаются в столбце "valdouble"):

| nodeid | actualtime | time | valint | valuint | valdouble | valbool | valstring | quality | recordtype | nodeid2 | tagname | description | unit |
|--------|-----------------|-----------------|--------|---------|-----------|---------|-----------|---------|------------|---------|----------------------------|---------------------------|------|
| 457 | 23.06.2021 4:47 | 23.06.2021 4:47 | | | 23 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:47 | 23.06.2021 4:47 | | | 25 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:48 | 23.06.2021 4:48 | | | 69 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:48 | 23.06.2021 4:48 | | | 24 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:49 | 23.06.2021 4:49 | | | 44 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:49 | 23.06.2021 4:49 | | | 55 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:49 | 23.06.2021 4:49 | | | 23 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:49 | 23.06.2021 4:49 | | | 24 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:49 | 23.06.2021 4:49 | | | 63 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:49 | 23.06.2021 4:49 | | | 0 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:49 | 23.06.2021 4:49 | | | 24 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:49 | 23.06.2021 4:49 | | | 21 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |

2. Запрос на получение истории изменения значений сигнала "NPS.MNS1.PT001_1.Value" за промежуток времени с 04:48 до 04:49 23 июня 2021 года (включая граничные значения):



```
SELECT * FROM nodes_history h JOIN nodes n ON n.NodeId=h.NodeId  
WHERE (n.TagName='NPS.MNS1.PT001_1.Value') AND (h.time BETWEEN  
'2021-06-23 04:48' AND '2021-06-23 04:49')
```

Результат запроса в MS Excel (в столбце "recordtype" отображается тип значения - граничная точка или значение внутри интервала):

| nodeid | actualtime | time | valint | valuint | valdouble | valbool | valstring | quality | recordtype | nodeid2 | tagname | description | unit |
|--------|-----------------|-----------------|--------|---------|-----------|---------|-----------|---------|------------|---------|----------------------------|---------------------------|------|
| 457 | 23.06.2021 4:47 | 23.06.2021 4:48 | | | 25 | | | 216 | lbound | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:48 | 23.06.2021 4:48 | | | 69 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:48 | 23.06.2021 4:48 | | | 24 | | | 216 | inner | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:49 | 23.06.2021 4:49 | | | 44 | | | 216 | ubound | | 457 NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |

3. Запрос на получение истории изменения значений сигнала "NPS.MNS1.PT001_1.Value" за промежуток времени с 04:48 до 04:49 23 июня 2021 года (без граничных значений):



```
SELECT * FROM nodes_history h JOIN nodes n ON n.NodeId=h.NodeId
WHERE (n.TagName='NPS.MNS1.PT001_1.Value') AND (h.time BETWEEN
'2021-06-23 04:48' AND '2021-06-23 04:49') AND (recordType !=
'ubound') AND (recordType != 'lbound')
```

Результат запроса в MS Excel:

| nodeid | actualtime | time | valint | valuint | valdot | valbool | valstring | quality | recordtype | nodeid | tagname | description | unit |
|--------|-----------------|-----------------|--------|---------|--------|---------|-----------|---------|------------|--------|------------------------|---------------------------|------|
| 457 | 23.06.2021 4:48 | 23.06.2021 4:48 | | | 69 | | | 216 | inner | 457 | NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |
| 457 | 23.06.2021 4:48 | 23.06.2021 4:48 | | | 24 | | | 216 | inner | 457 | NPS.MNS1.PT001_1.Value | Давление ДТ на приёме МНС | |

4. Для получения истории изменений состояний задвижки "ZDV_007_8_1" за промежуток времени с 04:30 до 04:50 29 января 2021 года создайте запрос с выборкой по столбцам "Сообщение" и "Время возникновения события":



```
SELECT message, activetime FROM event_history eh WHERE
(eh.source='NPS.UZA8.ZDV_007_8_1.Msgs') AND (eh.time BETWEEN
'2021-01-29 04:30' AND '2021-01-29 04:50')
```

Результат запроса в MS Excel:

| message | activetime |
|---|-----------------|
| УЗА №8. Задвижка №007.8.1. Открывается | 29.01.2021 4:30 |
| УЗА №8. Задвижка №007.8.1. Открытие невозможно. Нет напряжения в схеме управления | 29.01.2021 4:30 |
| УЗА №8. Задвижка №007.8.1. Открыта. Выполнение команды не требуется | 29.01.2021 4:31 |
| УЗА №8. Задвижка №007.8.1. Открывается в режиме имитации | 29.01.2021 4:32 |
| УЗА №8. Задвижка №007.8.1. Заккрытие невозможно. Авария задвижки | 29.01.2021 4:32 |
| УЗА №8. Задвижка №007.8.1. Открывается | 29.01.2021 4:33 |

5. Запрос на получение текущего значения сигнала "NPS.BIK172_6.BB97n.Cmd":



```
SELECT * FROM nodes_values v JOIN nodes n ON n.NodeId=v.NodeId
WHERE n.TagName='NPS.BIK172_6.BB97n.Cmd'
```

Результат запроса в MS Excel (текущее значение сигнала отображается в столбце "valuint"):

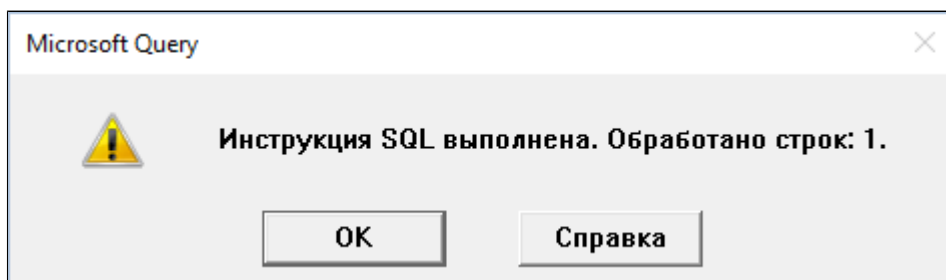
| nodeid | time | valint | valuint | valdouble | valbool | valstring | quality | nodeid2 | tagname |
|--------|-----------------|--------|---------|-----------|---------|-----------|---------|---------|---------|
| 43071 | 01.02.2021 8:39 | | 30 | | | | | 216 | |

6. Запрос на изменение текущего значения сигнала "NPS.BIK172_6.BB97n.Cmd" на значение "100":



```
UPDATE nodes_values v SET valint=100 FROM nodes n WHERE n.nodeid=v.nodeid AND n.TagName='NPS.BIK172_6.BB97n.Cmd'
```

После выполнения данной команды программа, через которую осуществляется SQL-доступ, выдает сообщение о выполнении инструкции SQL. Пример сообщения программы Microsoft Query:



Результат запроса на получение текущего значения сигнала "NPS.BIK172_6.BB97n.Cmd" (текущее (измененное) значение сигнала отображается в столбце "valuint"):

| nodeid | time | valint | valuint | valdouble | valbool | valstring | quality | nodeid2 | tagname |
|--------|-----------------|--------|---------|-----------|---------|-----------|---------|---------|---------|
| 43071 | 01.02.2021 8:52 | | 100 | | | | | 216 | |

7. Для получения истории изменений состояний всех задвижек узла "UZA12 НПС" создайте запрос с выборкой по столбцам "Время возникновения события", "Имя сигнала" и "Сообщение":



```
SELECT activetime, source, message FROM event_history WHERE source LIKE '%UZA12.%'
```


Результат запроса в MS Excel:

| activetime | source | message |
|--------------------|-----------------------------|---|
| 20.02.2021 8:12:51 | NPS.UZA12.ZDV_007_12_4.Msgs | УЗА №12. Задвижка №007.12.4. Открывается |
| 20.02.2021 8:12:56 | NPS.UZA12.ZDV_007_12_6.Msgs | УЗА №12. Задвижка №007.12.6. В промежуточном положении |
| 20.02.2021 8:13:33 | NPS.UZA12.ZDV_007_12_4.Msgs | УЗА №12. Задвижка №007.12.4. Открытие невозможно. Авария задвижки |
| 20.02.2021 8:13:37 | NPS.UZA12.ZDV_007_12_6.Msgs | УЗА №12. Задвижка №007.12.6. Открытие невозможно. Нет напряжения в схеме управления |
| 20.02.2021 8:14:30 | NPS.UZA12.ZDV_007_12_4.Msgs | УЗА №12. Задвижка №007.12.4. Открыта. Выполнение команды не требуется |
| 20.02.2021 8:14:34 | NPS.UZA12.ZDV_007_12_6.Msgs | УЗА №12. Задвижка №007.12.6. Открыта. Выполнение команды не требуется |
| 21.02.2021 7:52:00 | NPS.UZA12.ZDV_007_12_3.Msgs | УЗА №12. Задвижка №007.12.3. Открыта. Выполнение команды не требуется |
| 21.02.2021 7:52:05 | NPS.UZA12.ZDV_007_12_5.Msgs | УЗА №12. Задвижка №007.12.5. Открыта. Выполнение команды не требуется |
| 21.02.2021 7:52:13 | NPS.UZA12.ZDV_007_12_3.Msgs | УЗА №12. Задвижка №007.12.3. Резерв |
| 21.02.2021 7:52:18 | NPS.UZA12.ZDV_007_12_5.Msgs | УЗА №12. Задвижка №007.12.5. Открывается |
| 21.02.2021 7:52:25 | NPS.UZA12.ZDV_007_12_4.Msgs | УЗА №12. Задвижка №007.12.4. Открыта в режиме имитации |

8. Для получения истории событий по задвижкам, находящимся в состоянии Открыта, за промежуток времени с 00:00 20 февраля 2021 года до 23:00 21 февраля 2021 года создайте запрос с выборкой по столбцам "Время возникновения события", "Имя сигнала" и "Сообщение":



```
SELECT activetime, source, message FROM event_history eh WHERE (eh.time BETWEEN '2021-02-20 00:00' AND '2021-02-21 23:00') AND (message LIKE '%Открыта');
```

Результат запроса в MS Excel:

| activetime | source | message |
|--------------------|-----------------------------|--------------------------------------|
| 20.02.2021 8:13:45 | NPS.UZA8.ZDV_007_8_4.Msgs | УЗА №8. Задвижка №007.8.4. Открыта |
| 21.02.2021 8:09:24 | NPS.UZA8.ZDV_007_8_1.Msgs | УЗА №8. Задвижка №007.8.1. Открыта |
| 21.02.2021 8:09:29 | NPS.UZA12.ZDV_007_12_3.Msgs | УЗА №12. Задвижка №007.12.3. Открыта |
| 21.02.2021 8:09:33 | NPS.UZA12.ZDV_007_12_5.Msgs | УЗА №12. Задвижка №007.12.5. Открыта |
| 21.02.2021 8:09:37 | NPS.UZA8.ZDV_007_8_6.Msgs | УЗА №8. Задвижка №007.8.6. Открыта |
| 21.02.2021 8:09:42 | NPS.UZA12.ZDV_007_12_6.Msgs | УЗА №12. Задвижка №007.12.6. Открыта |
| 21.02.2021 8:09:47 | NPS.UZA13.ZDV_007_13_2.Msgs | УЗА №13. Задвижка №007.13.2. Открыта |
| 21.02.2021 8:09:59 | NPS.UZA12.ZDV_007_12_4.Msgs | УЗА №12. Задвижка №007.12.4. Открыта |

Добавление данных в БД

В этом примере добавляется новая запись в таблицу "testtables" для столбцов "Id", "Value", "Category", "Text" с указанными значениями.



```
Table_1.TableModel_1.SqlQuery_1.Connect();// Установка соединения
с базой данных
if ( Table_1.TableModel_1.SqlQuery_1.ConnectionState == 1 ) { //
Проверка соединения с базой данных
    Table_1.TableModel_1.SqlQuery_1.Text = "INSERT INTO testtables
(Id, Value, Category, Text) VALUES (1, 6, 'Even', 'Six')"; // Установка
текста SQL-запроса
    Table_1.TableModel_1.SqlQuery_1.Execute( ); // Выполнение
запроса INSERT
}
else
    Text_1.Text="ConnectError"
```

Обновление данных в БД

В этом примере обновляется значение столбца "Value" в строке, где значение столбца "Id" равно "2".



```
Table_1.TableModel_1.SqlQuery_1.Connect();// Установка соединения
с базой данных
if (Table_1.TableModel_1.SqlQuery_1.ConnectionState == 1) { //
Проверка соединения с базой данных
    Table_1.TableModel_1.SqlQuery_1.Text = "UPDATE testtables SET
Value = 343 WHERE Id = 2"; // Установка текста SQL-запроса
    Table_1.TableModel_1.SqlQuery_1.Execute( ); // Выполнение
запроса UPDATE
}
else
    Text_1.Text="ConnectError"
```

Получение данных из БД

В этом примере получены все данные из таблицы "testtables".



```
Table_1.TableModel_1.SqlQuery_1.Connect();// Установка соединения
с базой данных
if ( Table_1.TableModel_1.SqlQuery_1.ConnectionState == 1 ) { //
Проверка соединения с базой данных
    Table_1.TableModel_1.SqlQuery_1.Text = "SELECT * from
testtables"; // Установка текста SQL-запроса
    Table_1.TableModel_1.SqlQuery_1.Execute( ); // Выполнение
запроса SELECT
    Table_1.TableModel_1.BeginReadAsync( ); // Полученными
данными заполняем таблицу
}
else
    Text_1.Text="ConnectError"
```

Удаление данных из БД

В этом примере удаляются все строки из таблицы "testtables", где значение столбца "Category" равно "Even".



```
Table_1.TableModel_1.SqlQuery_1.Connect();// Установка соединения
с базой данных
if ( Table_1.TableModel_1.SqlQuery_1.ConnectionState == 1 ) { //
Проверка соединения с базой данных
    Table_1.TableModel_1.SqlQuery_1.Text = "DELETE FROM testtables
WHERE Category = 'Even'";
// Установка текста SQL-запроса
    Table_1.TableModel_1.SqlQuery_1.Execute( ); // Выполнение
запроса DELETE
}
else
    Text_1.Text="ConnectError"
```

1.2.4. Подключение по TCP

Чтобы подключиться к Astra.Server и Astra.Historian по TCP, укажите значения следующих атрибутов параметра OPTIONS команды SQL CREATE SERVER в файле скрипта init_rmap_history.sql:

- › SourceLocation – адрес Astra.Server.
- › SourceTcpPort – порт, на который настроен модуль TCP Server.
- › HistorianLocation – адрес Astra.Historian, в котором хранятся значения.
- › HistorianTcpPort – порт Astra.Historian, в котором хранятся значения, или порт доступа к истории модуля TCP Server;
- › HistorianDB – имя базы данных, в которой хранятся значения.
- › SourceAETcpPort – порт, на который настроен модуль TCP Server.
- › AeHistorianLocation – адрес Astra.Historian, в котором хранятся события.
- › AeHistorianTcpPort – порт Astra.Historian, в котором хранятся события, или порт доступа к истории модуля TCP Server.
- › AeHistorianDB – имя базы данных, в которой хранятся события.



Символы комментирования "--" перед атрибутами SourceTcpPort, HistorianTcpPort, SourceAETcpPort, AeHistorianTcpPort должны отсутствовать.



Атрибуты SourceDA, HistorianName, SourceAE, AeHistorianName должны быть закоментированы символами "--".

Параметры Astra.Historian, необходимые для настройки Astra.RMap, прописаны в конфигурационном файле Astra.Historian.Server.xml. Номера портов модуля TCP Server указаны в конфигурации Astra.Server.

The screenshot shows the 'Parameters of the module configuration node' window in pgAdmin. Under the '2. Additional' section, the following parameters are listed:

| | |
|--|--------|
| Номер TCP порта | 4388 |
| Анонимные клиенты могут изменять значения сигналов | Да |
| Максимальный размер очередей данных | 500000 |
| Максимальный размер очереди сообщений | 100000 |

The 'init_rmap_history' SQL script is shown in a separate window. The relevant lines are:

```

--HistorianName 'Astra.Historian.Server',
HistorianTcpPort '4949',
HistorianDB 'ValuesDB',
--SourceAE 'Astra.OPCAEServer',
SourceAETcpPort '4388',
AeHistorianLocation '127.0.0.1',
--AeHistorianName 'Astra.Historian.Server'
AeHistorianTcpPort '4949',
AeHistorianDB 'EventsDB'

```

The 'Astra.Historian.Server' configuration window shows the following XML snippet:

```

<tcp-server default-port=4949 idle-sessions-count=1
idle-sessions-timeout=15>
  <server-endpoint host="0.0.0.0" />
</tcp-server>
<Bases>
  <Base Alias="ValuesDB" />
  <Base Alias="EventsDB" />
</Bases>
</Astra.Historian.Server>

```

Для получения истории с помощью модуля TCP Server в значениях атрибутов HistorianTcpPort и AeHistorianTcpPort укажите порт, заданный в параметре Порт доступа к истории модуля TCP Server. Если в настройках модуля порт не задан, установите в параметре Порт доступа к истории любой свободный порт.

The screenshot shows the 'Parameters of the module configuration node' window in pgAdmin. Under the '3. History access settings' section, the following parameters are listed:

| | |
|------------------------------|------|
| Порт доступа к истории | 8085 |
| Время удержания сессии | 10 |
| Ограничение ожидающих сессий | 64 |

The 'init_rmap_history' SQL script is shown in a separate window. The relevant lines are:

```

--HistorianName 'Astra.Historian.Server',
HistorianTcpPort '8085',
HistorianDB 'ValuesDB',
--SourceAE 'Astra.OPCAEServer',
SourceAETcpPort '4388',
AeHistorianLocation '127.0.0.1',
--AeHistorianName 'Astra.Historian.Server'
AeHistorianTcpPort '8085',
AeHistorianDB 'EventsDB'

```

The 'Astra.Historian.Server' configuration window shows the following XML snippet:

```

<tcp-server default-port=4949 idle-sessions-count=1
idle-sessions-timeout=15>
  <server-endpoint host="0.0.0.0" />
</tcp-server>
<Bases>
  <Base Alias="ValuesDB" />
  <Base Alias="EventsDB" />
</Bases>
</Astra.Historian.Server>

```

После внесения изменений файл init_rmap_history.sql сохраните и исполните данный скрипт с помощью среды разработки и администрирования pgAdmin.

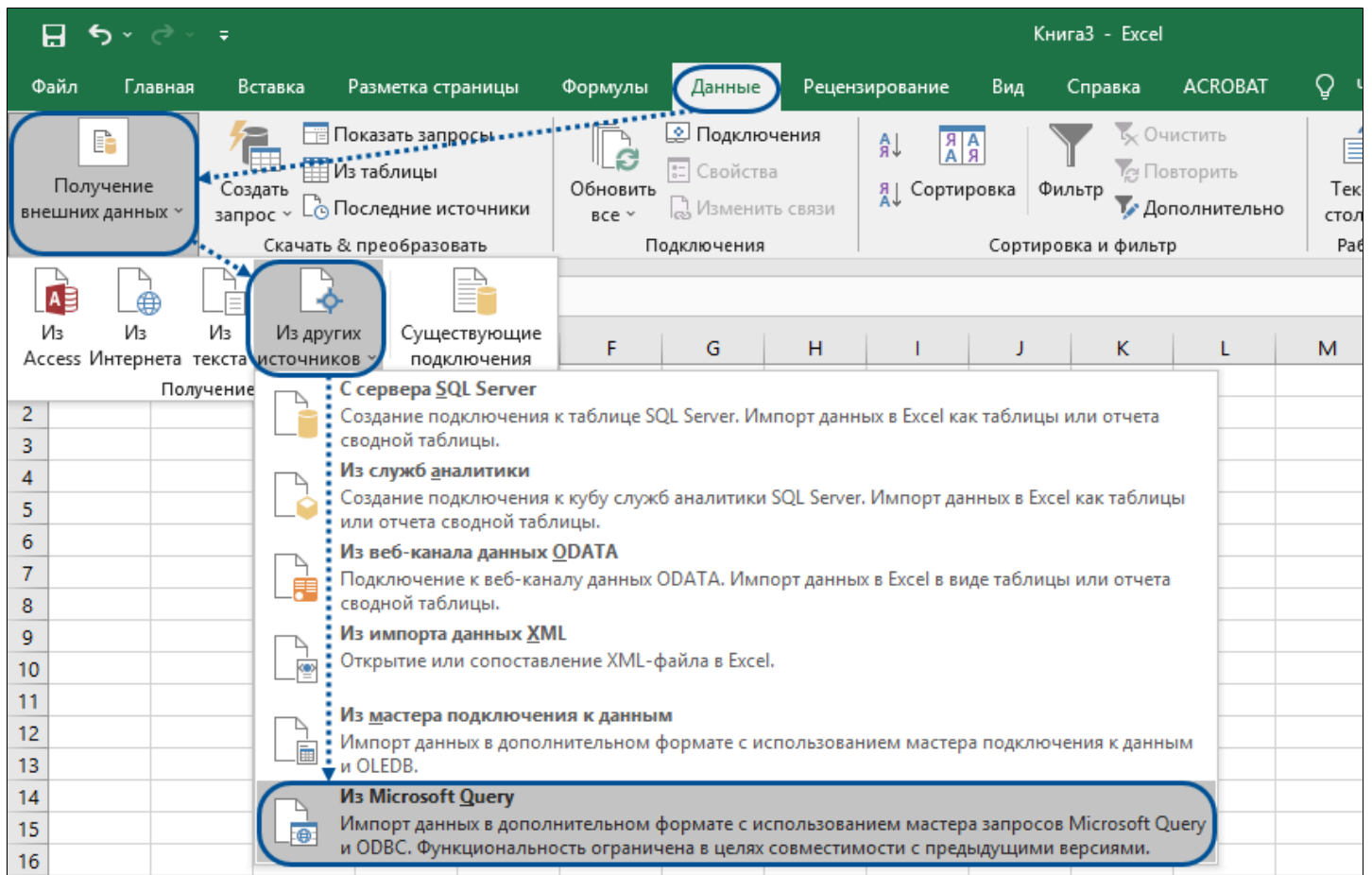
1.2.5. Предоставление данных в MS Excel



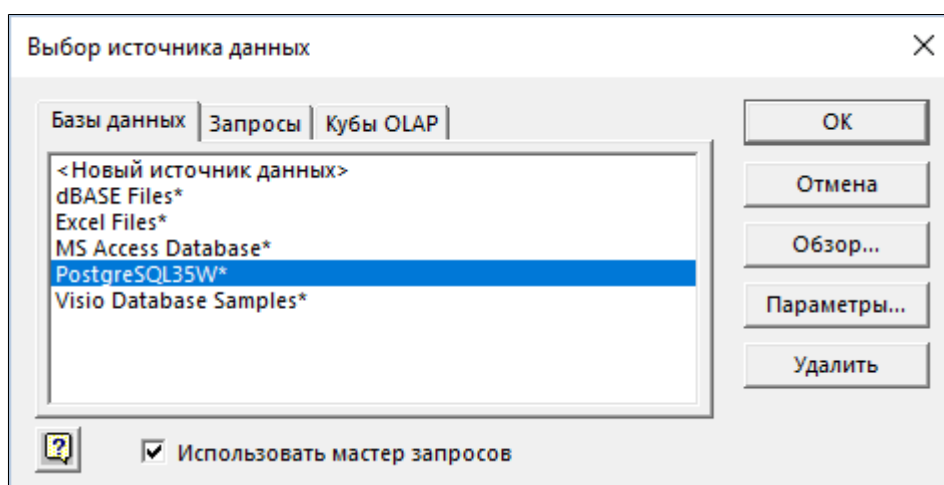
Для представления данных в MS Excel через Astra.RMap должен быть установлен драйвер ODBC для PostgreSQL.

Порядок действий:

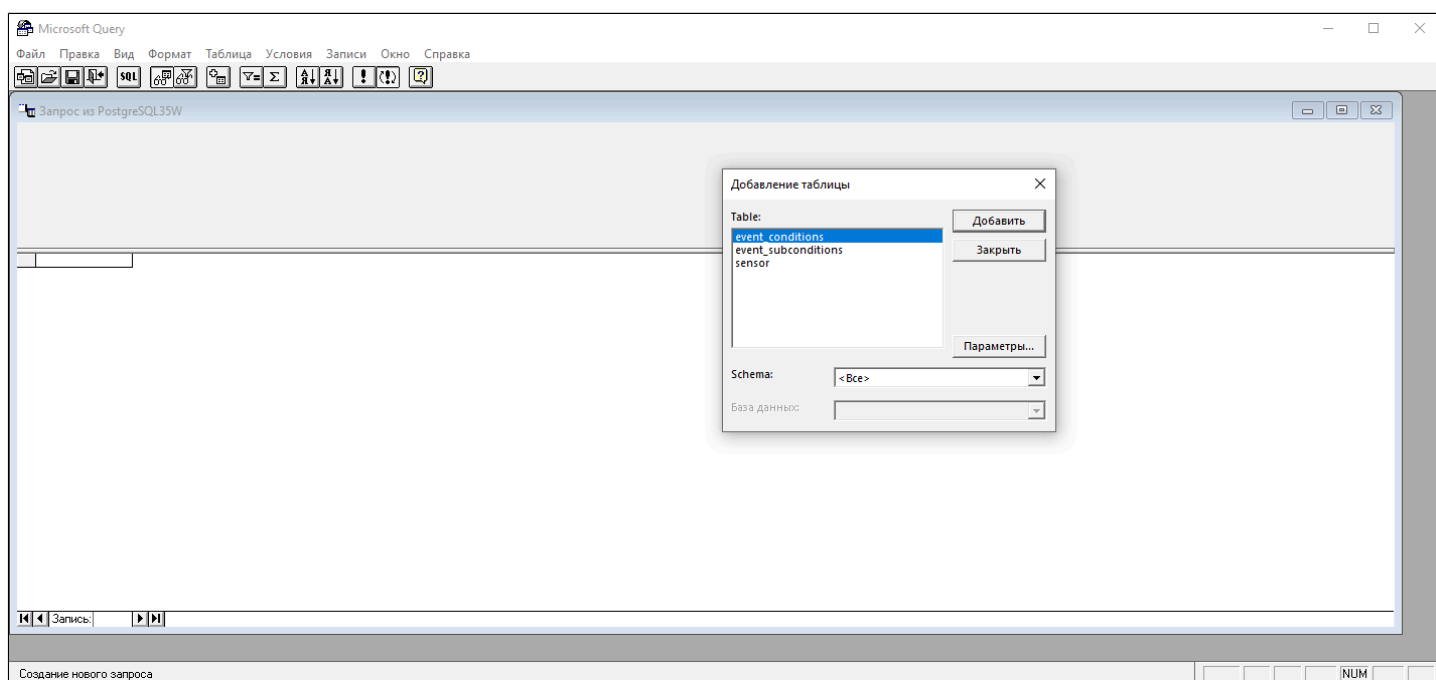
1. Запустите программу Microsoft Excel.
2. Создайте новую Книгу.
3. На вкладке "Данные" в области "Получение внешних данных" нажмите кнопку "Из других источников" и в выпадающем списке выберите "Из Microsoft Query".



4. В появившемся окне "Выбор источника данных" на вкладке "Базы данных" выберите базу "PostgreSQL" и уберите флаг "Использовать мастер запросов". Нажмите кнопку "OK".

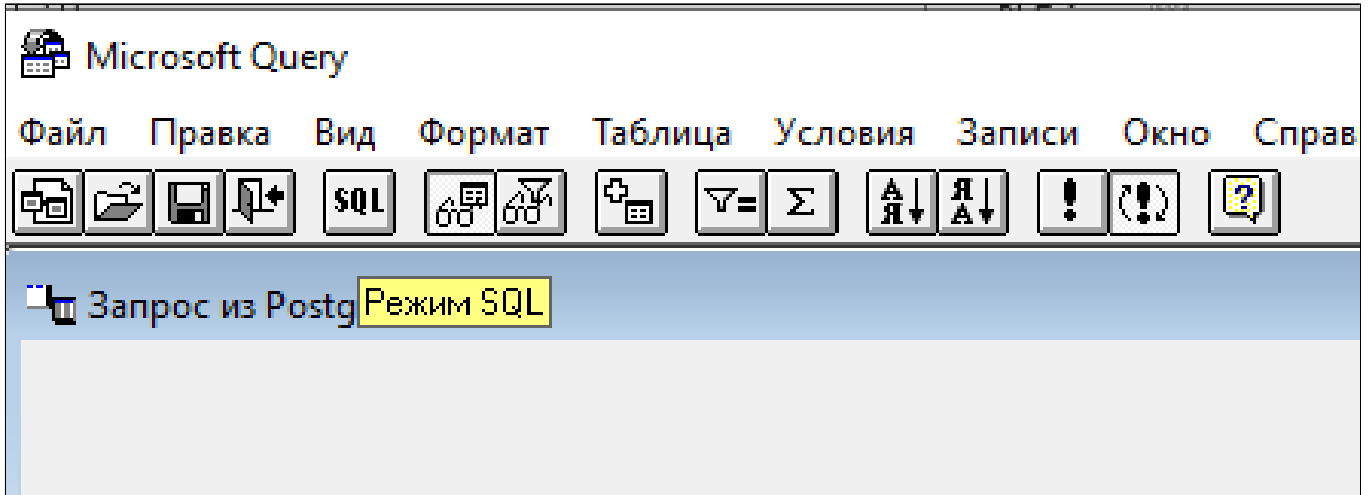


В результате будет запущена программа Microsoft Query.

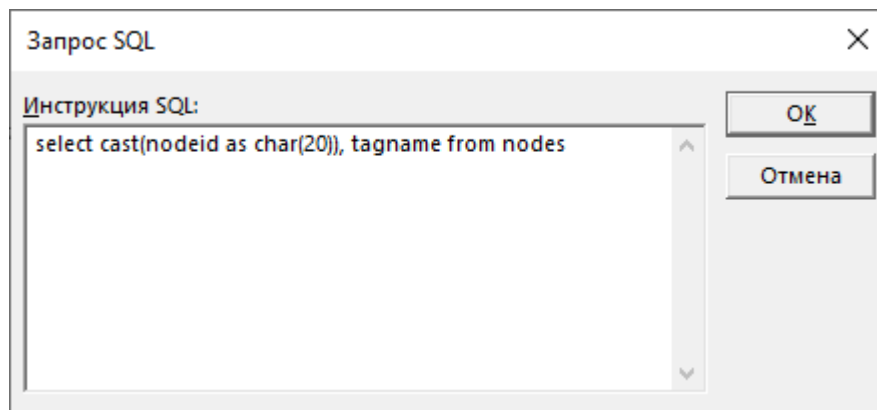


5. Закройте окно "Добавление таблицы".

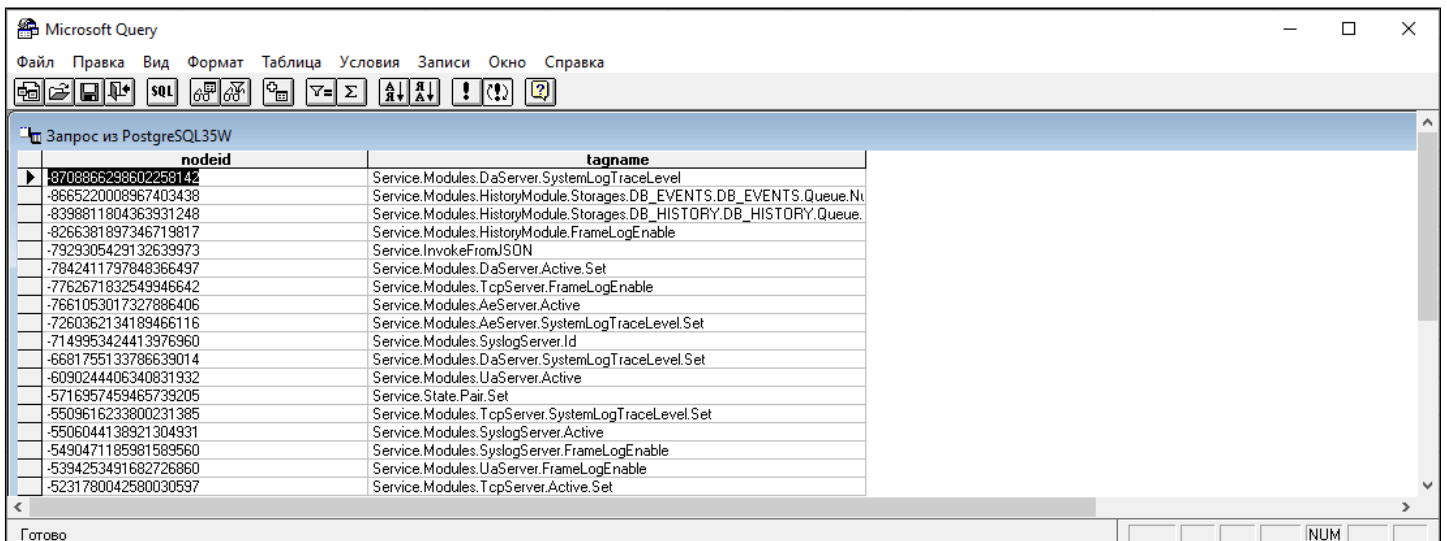
6. В окне программы Microsoft Query нажмите кнопку Режим SQL .



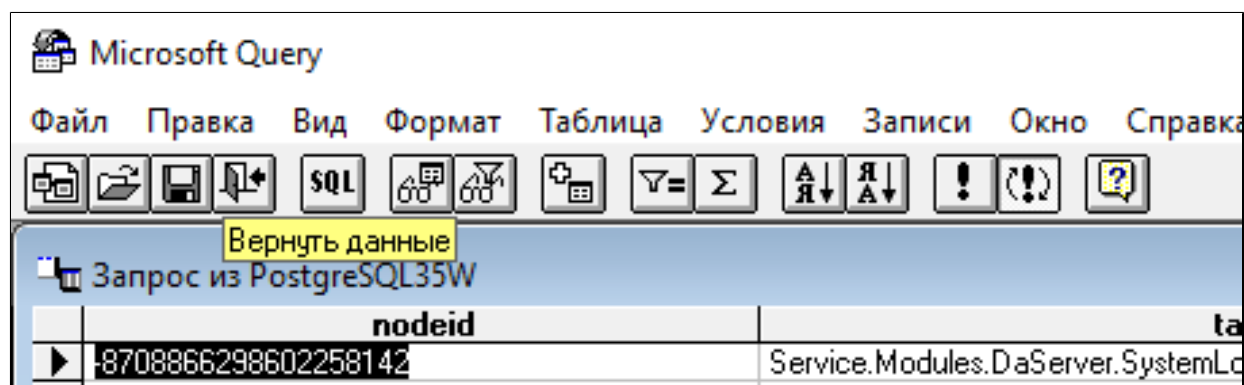
7. В появившемся окне "Запрос SQL" напишите SQL-запрос и нажмите кнопку "OK".



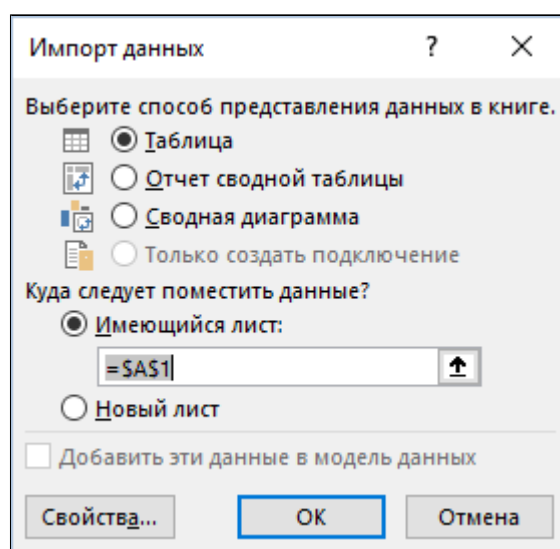
Результат запроса будет выведен в окне программы "Microsoft Query".



8. Нажмите кнопку "Вернуть данные".



9. В появившемся окне "Импорт данных" укажите параметры импорта данных и нажмите кнопку "OK".



Результаты SQL-запроса будут импортированы в Microsoft Excel.

| nodeid | tagname |
|----------------------|--|
| -8708866298602258142 | Service.Modules.DaServer.SystemLogTraceLevel |
| -8665220008967403438 | Service.Modules.HistoryModule.Storages.DB_EVENTS.DB_EVENTS.Queue.NumRecords |
| -8398811804363931248 | Service.Modules.HistoryModule.Storages.DB_HISTORY.DB_HISTORY.Queue.CurrentWriteQueueSize |
| -8266381897346719817 | Service.Modules.HistoryModule.FrameLogEnable |
| -7929305429132639973 | Service.InvokeFromJSON |
| -7842411797848366497 | Service.Modules.DaServer.Active.Set |
| -7762671832549946642 | Service.Modules.TcpServer.FrameLogEnable |
| -7661053017327886406 | Service.Modules.AeServer.Active |
| -7260362134189466116 | Service.Modules.AeServer.SystemLogTraceLevel.Set |
| -7149953424413976960 | Service.Modules.SyslogServer.Id |
| -6681755133786639014 | Service.Modules.DaServer.SystemLogTraceLevel.Set |
| -6090244406340831932 | Service.Modules.UaServer.Active |
| -5716957459465739205 | Service.State.Pair.Set |
| -5509616233800231385 | Service.Modules.TcpServer.SystemLogTraceLevel.Set |
| -5506044138921304931 | Service.Modules.SyslogServer.Active |
| -5490471185981589560 | Service.Modules.SyslogServer.FrameLogEnable |
| -5394253491682726860 | Service.Modules.UaServer.FrameLogEnable |
| -5231780042580030597 | Service.Modules.TcpServer.Active.Set |
| -5094472086410317985 | Service.Modules.HistoryModule.Storages.Historian.DB_EVENTS.DB_EVENTS.IntervalLockHigh |

1.2.6. Предоставление данных в LibreOffice Calc

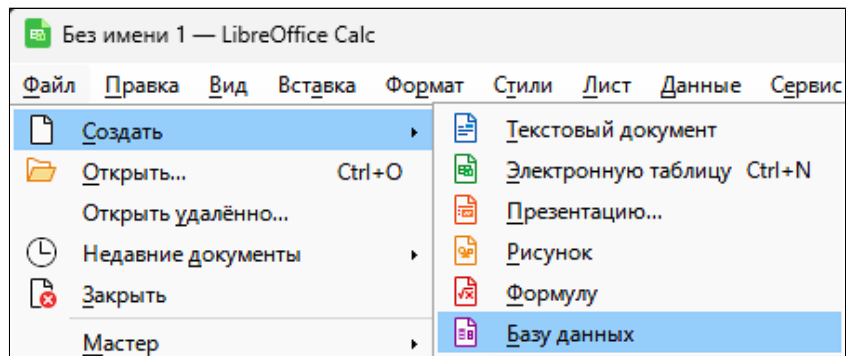
[Windows](#)

[AstraLinux](#)

1.2.6.1. Windows

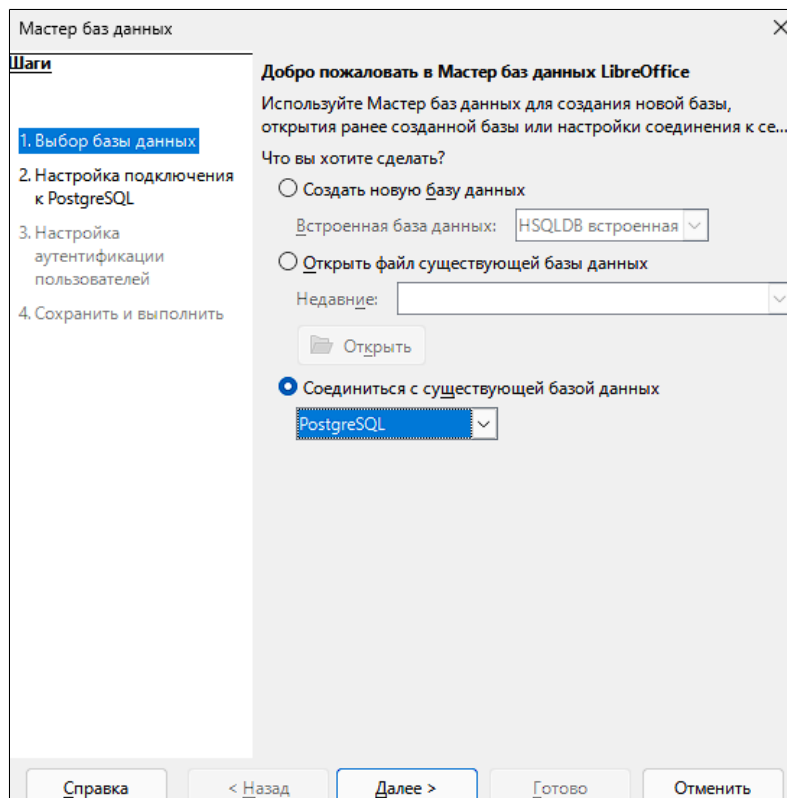
Для предоставления данных запустите LibreOffice Calc и выполните следующие действия:

1. Создайте базу данных. Для этого выполните команду меню "Файл" → "Создать" → "Базу данных".



2. Укажите БД для подключения:

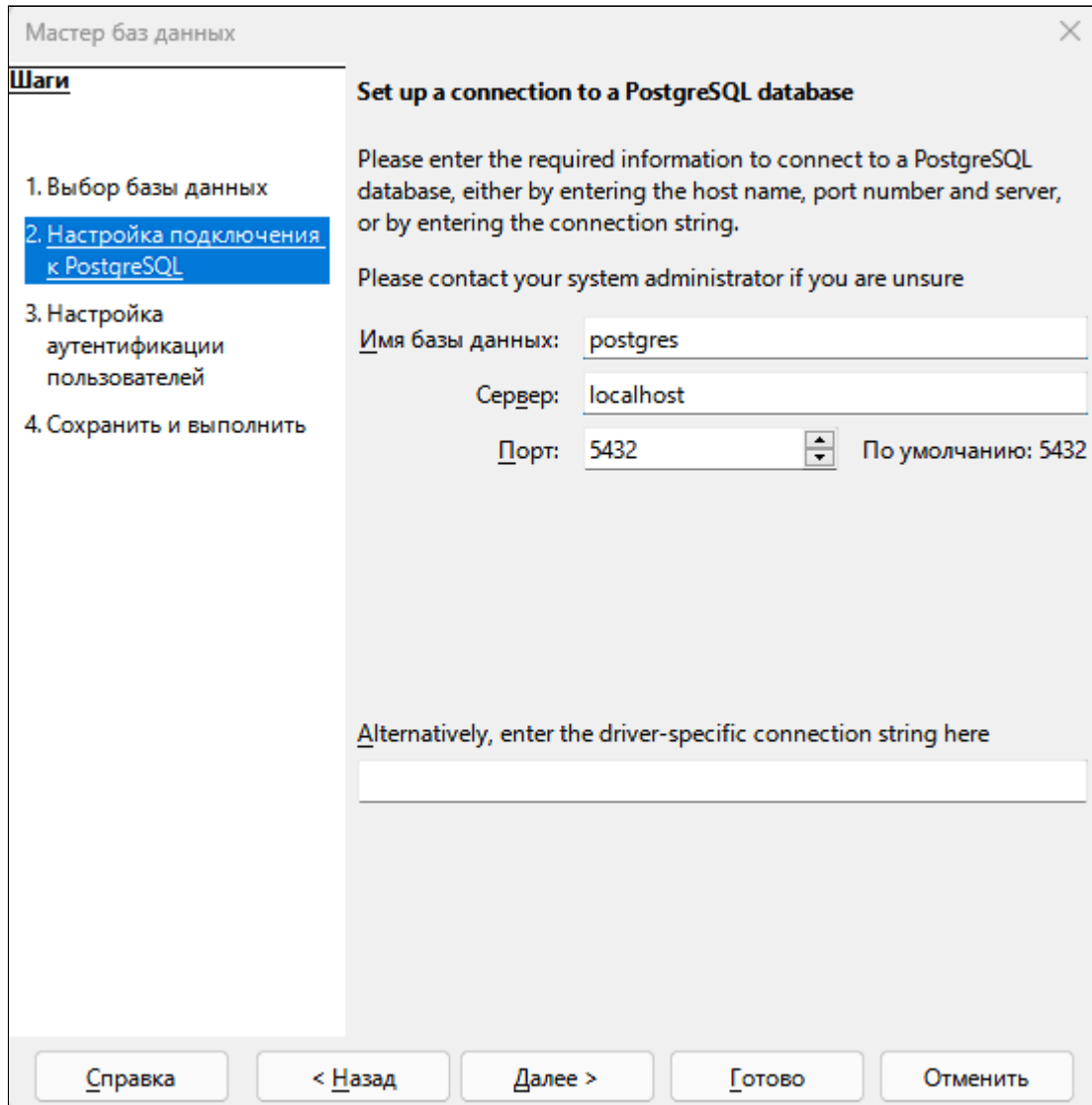
- Выберите пункт "Соединиться с существующей базой данных";
- В выпадающем списке выберите "PostgreSQL".



Нажмите кнопку "Далее".

3. Укажите параметры подключения к БД:

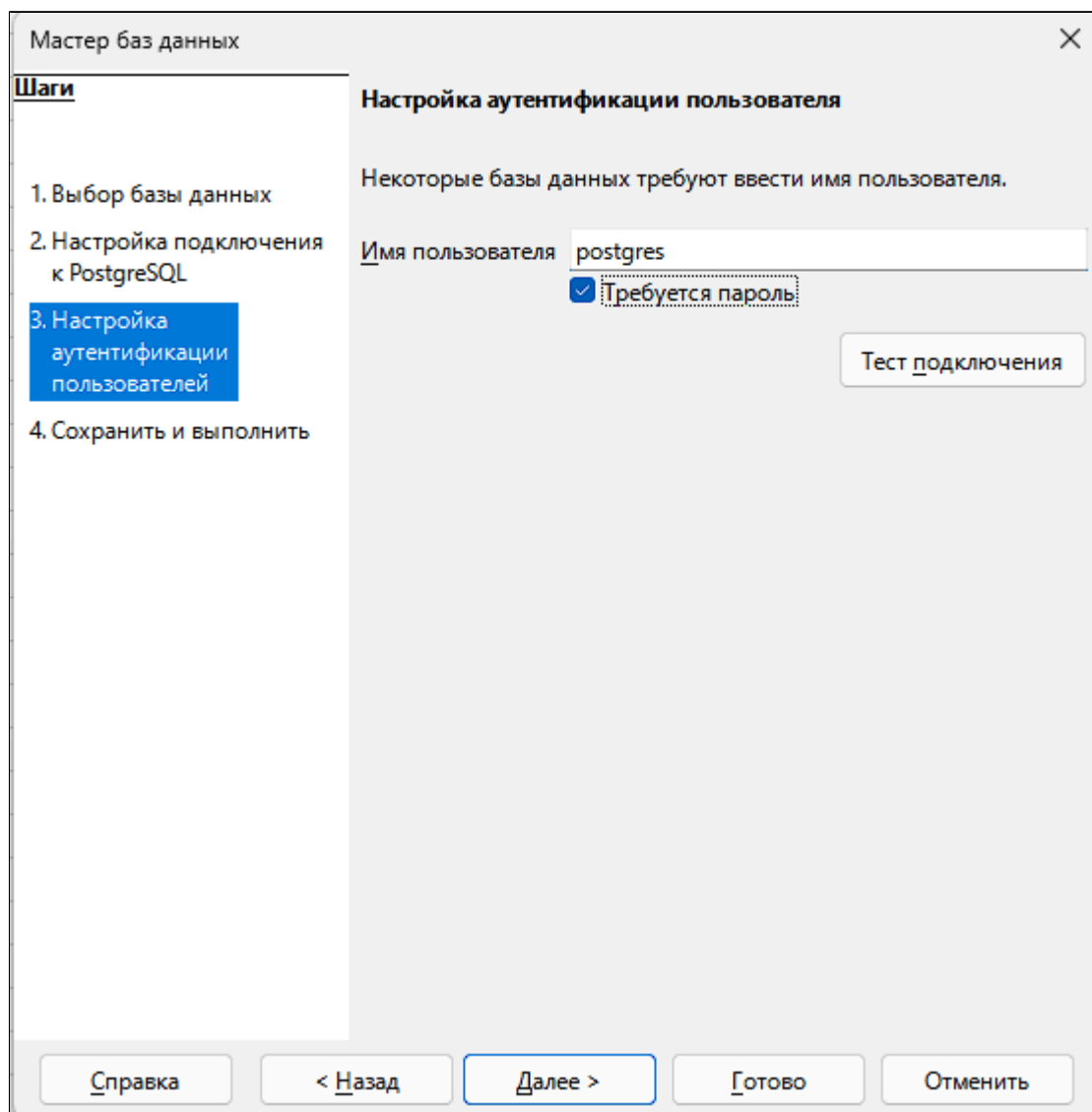
- › «Имя базы данных» - имя базы данных PostgreSQL;
- › «Сервер» - адрес сервера PostgreSQL;
- › «Порт» - порт подключения к серверу PostgreSQL.



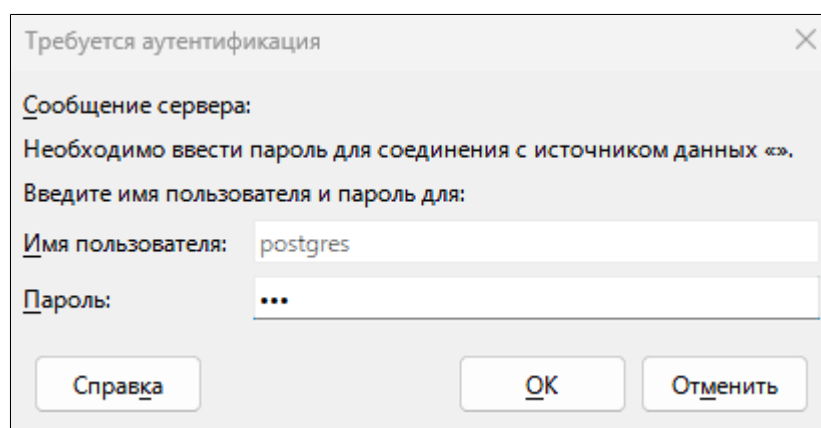
The screenshot shows a window titled "Мастер баз данных" (Master of Databases) with a close button in the top right corner. On the left, a "Шаги" (Steps) sidebar lists four steps: 1. Выбор базы данных, 2. Настройка подключения к PostgreSQL (highlighted in blue), 3. Настройка аутентификации пользователей, and 4. Сохранить и выполнить. The main area is titled "Set up a connection to a PostgreSQL database" and contains the following text: "Please enter the required information to connect to a PostgreSQL database, either by entering the host name, port number and server, or by entering the connection string." Below this, it says "Please contact your system administrator if you are unsure". There are three input fields: "Имя базы данных:" with the value "postgres", "Сервер:" with the value "localhost", and "Порт:" with a spinner control showing "5432" and the text "По умолчанию: 5432". At the bottom, there is a text box for an alternative connection string with the label "Alternatively, enter the driver-specific connection string here". At the very bottom, there are five buttons: "Справка", "< Назад", "Далее >", "Готово", and "Отменить".

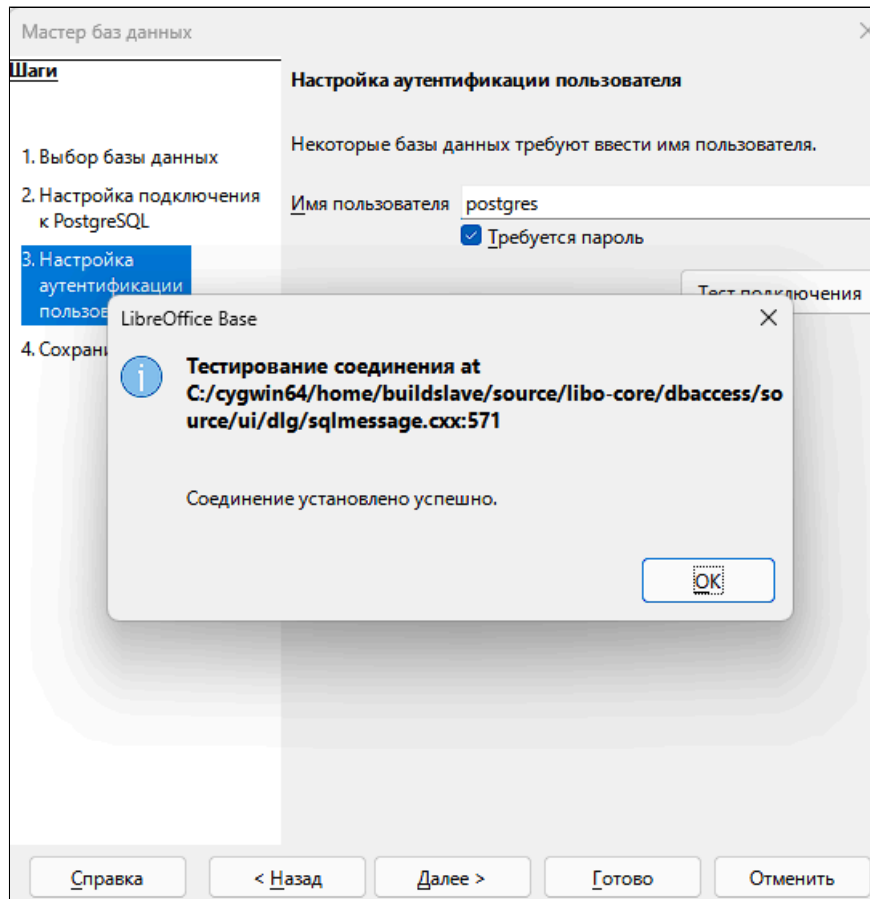
Нажмите кнопку "Далее".

4. Укажите имя пользователя для подключения к серверу "PostgreSQL" и установите флаг "Требуется пароль".

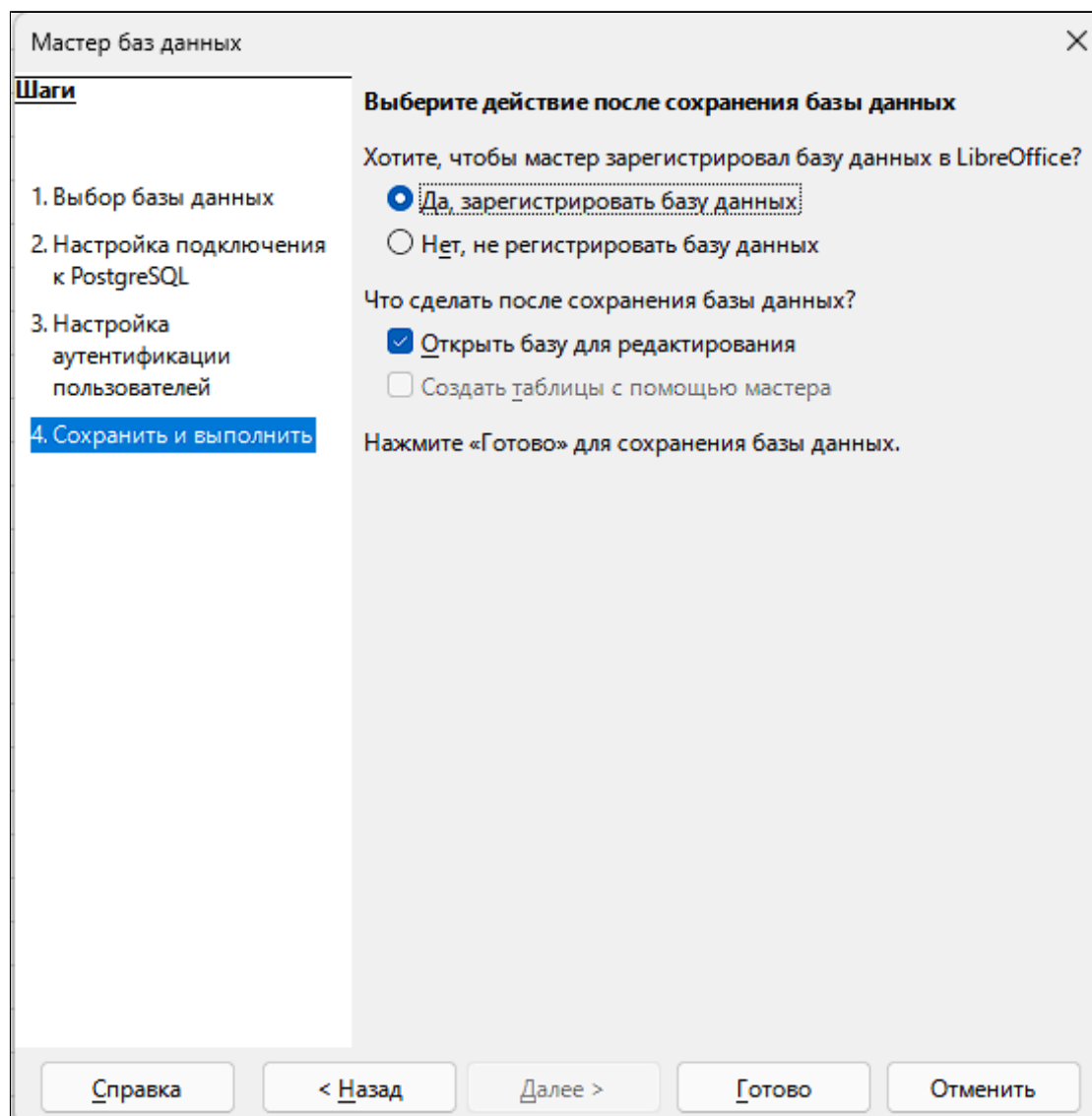


5. Нажмите кнопку "Тест подключения", введите пароль и нажмите кнопку "OK".

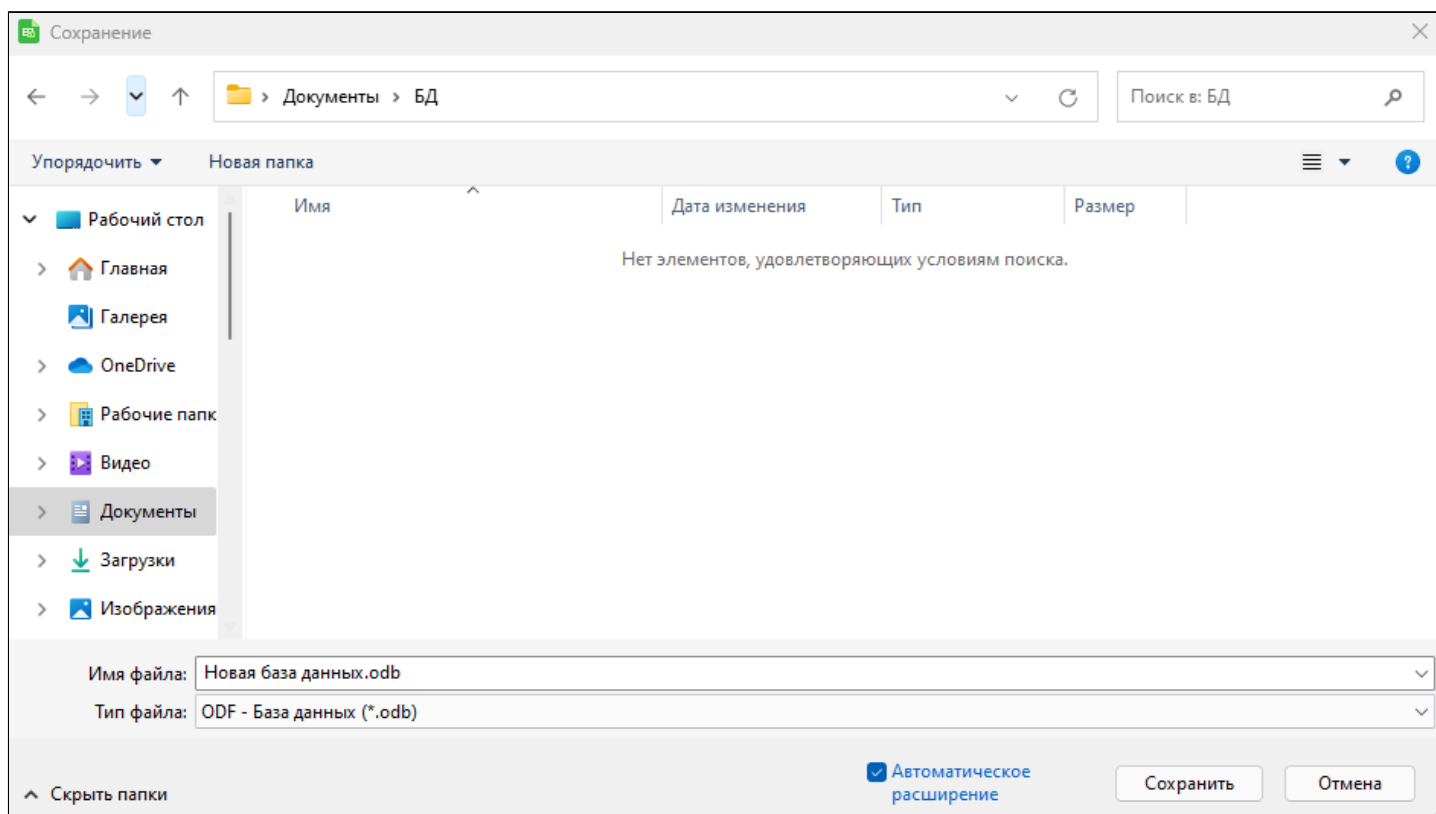




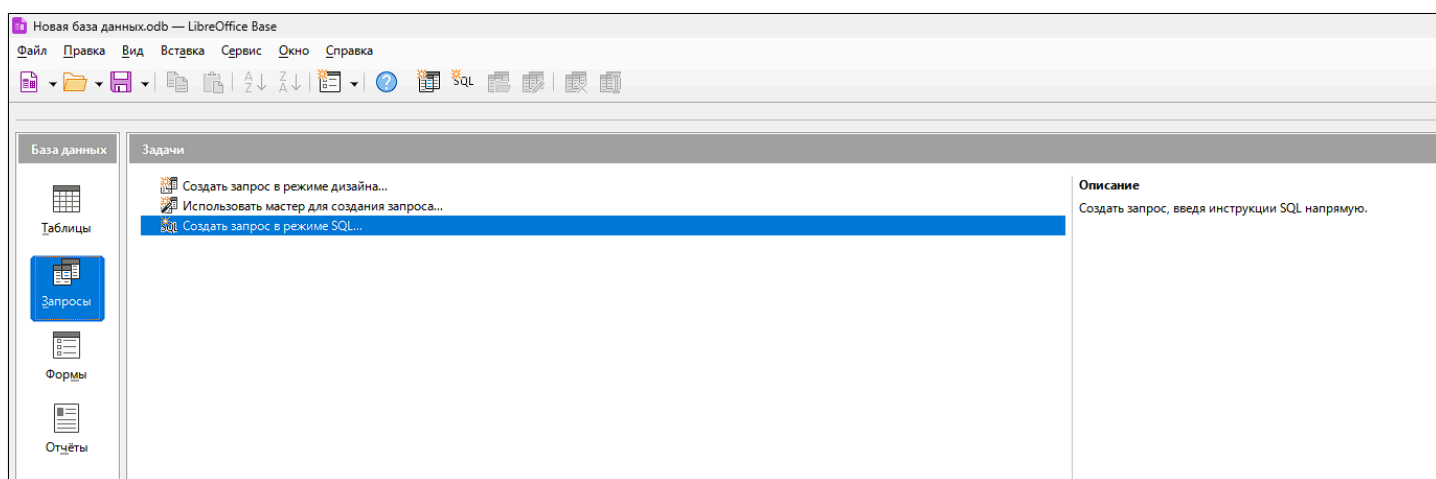
6. Установите флаги "Да, зарегистрировать базу данных" и "Открыть базу для редактирования", и нажмите кнопку "Готово".



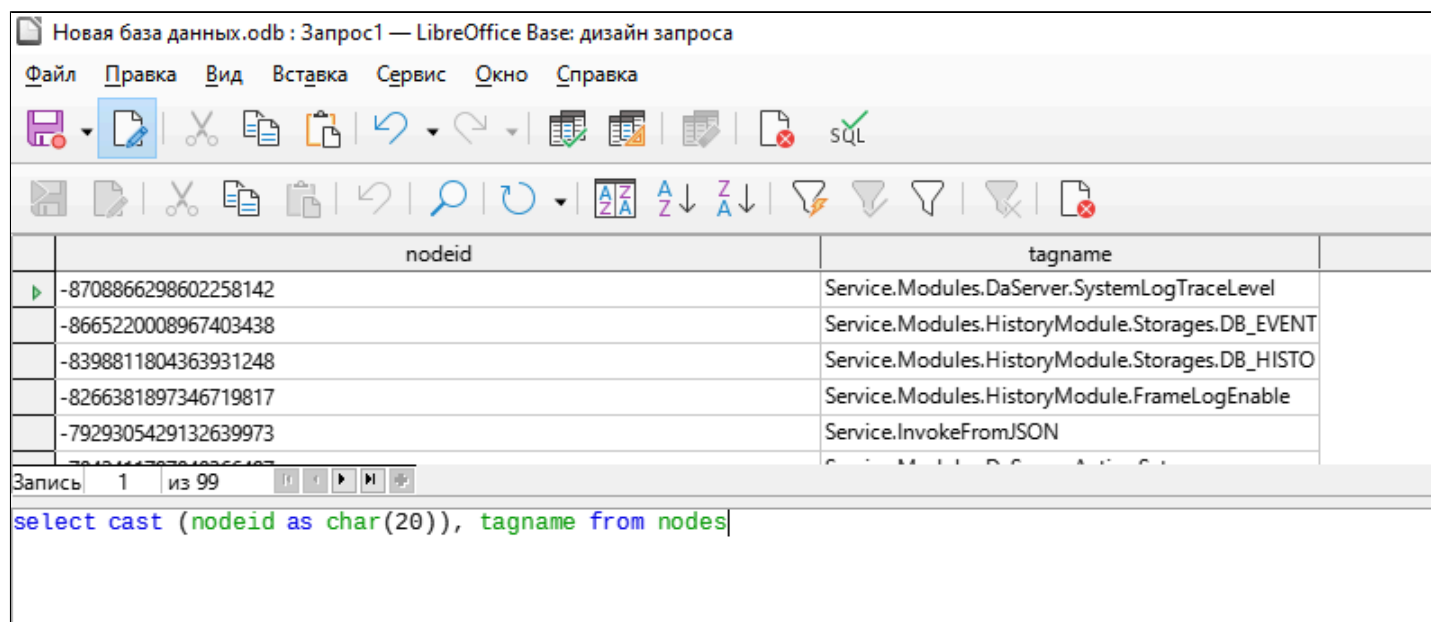
7. В окне проводника при необходимости переименуйте БД и сохраните в необходимой директории.



8. В открывшемся окне выберите "Запросы" → "Создать запрос в режиме SQL".



9. Введите требуемый SQL-запрос и выполните его, нажав кнопку "Выполнить запрос".



Новая база данных.oddb : Запрос1 — LibreOffice Base: дизайн запроса

Файл П_равка В_ид В_ставка С_ервис О_кно С_правка

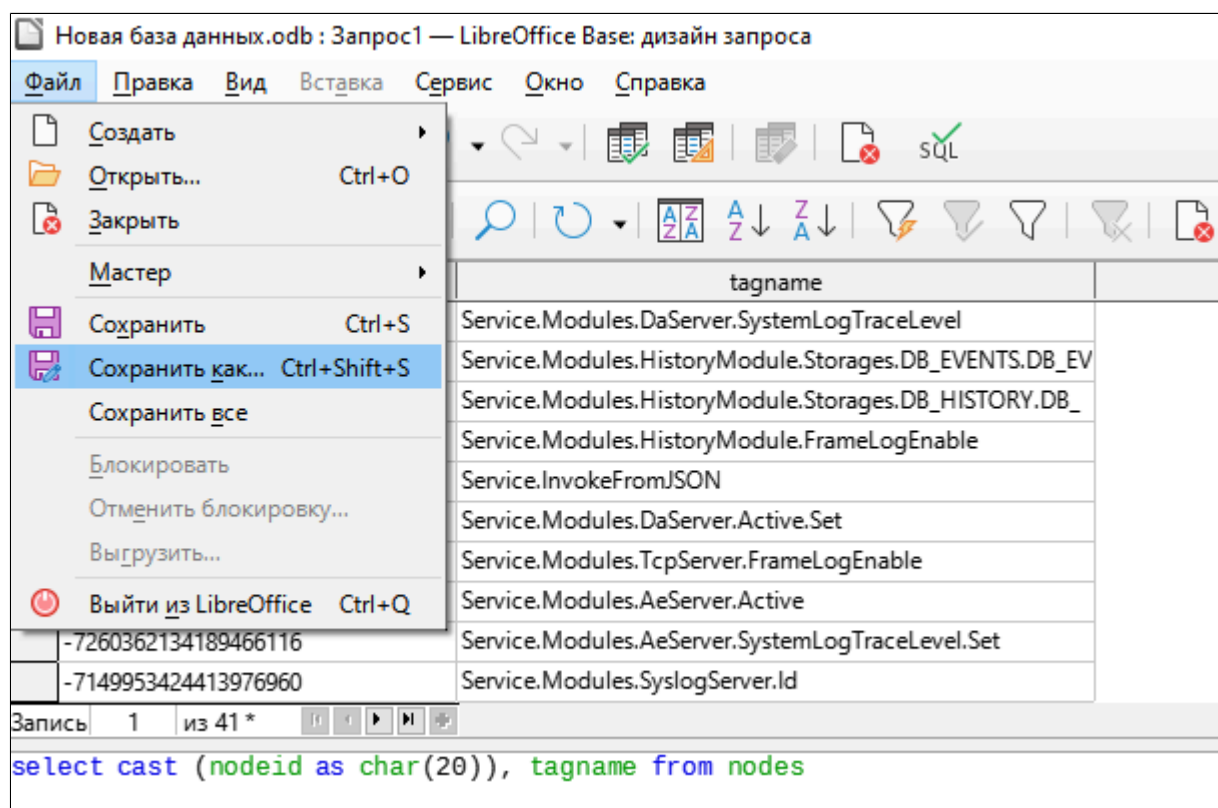
SQL

| nodeid | tagname |
|----------------------|---|
| -8708866298602258142 | Service.Modules.DaServer.SystemLogTraceLevel |
| -8665220008967403438 | Service.Modules.HistoryModule.Storages.DB_EVENT |
| -8398811804363931248 | Service.Modules.HistoryModule.Storages.DB_HISTO |
| -8266381897346719817 | Service.Modules.HistoryModule.FrameLogEnable |
| -7929305429132639973 | Service.InvokeFromJSON |

Запись 1 из 99

```
select cast (nodeid as char(20)), tagname from nodes
```

10. Сохраните запрос, выполнив команду меню "Файл" → "Сохранить как...". Сохраненные запросы отображаются в области "Запросы".



Новая база данных.oddb : Запрос1 — LibreOffice Base: дизайн запроса

Файл П_равка В_ид В_ставка С_ервис О_кно С_правка

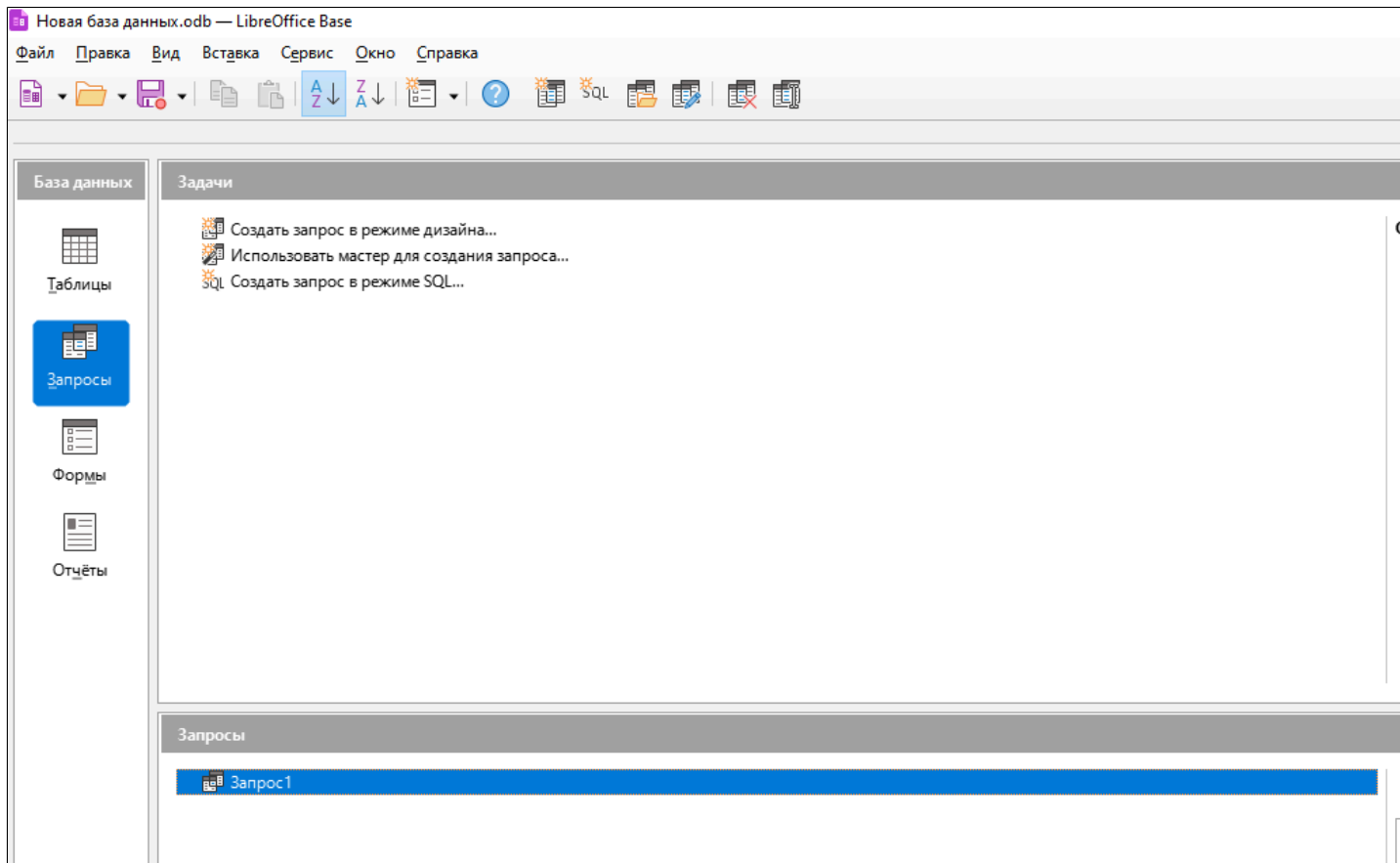
- Создать
- Открыть... Ctrl+O
- Закрыть
- Мастер
- Сохранить Ctrl+S
- Сохранить как... Ctrl+Shift+S**
- Сохранить все
- Блокировать
- Отменить блокировку...
- Выгрузить...
- Выйти из LibreOffice Ctrl+Q

SQL

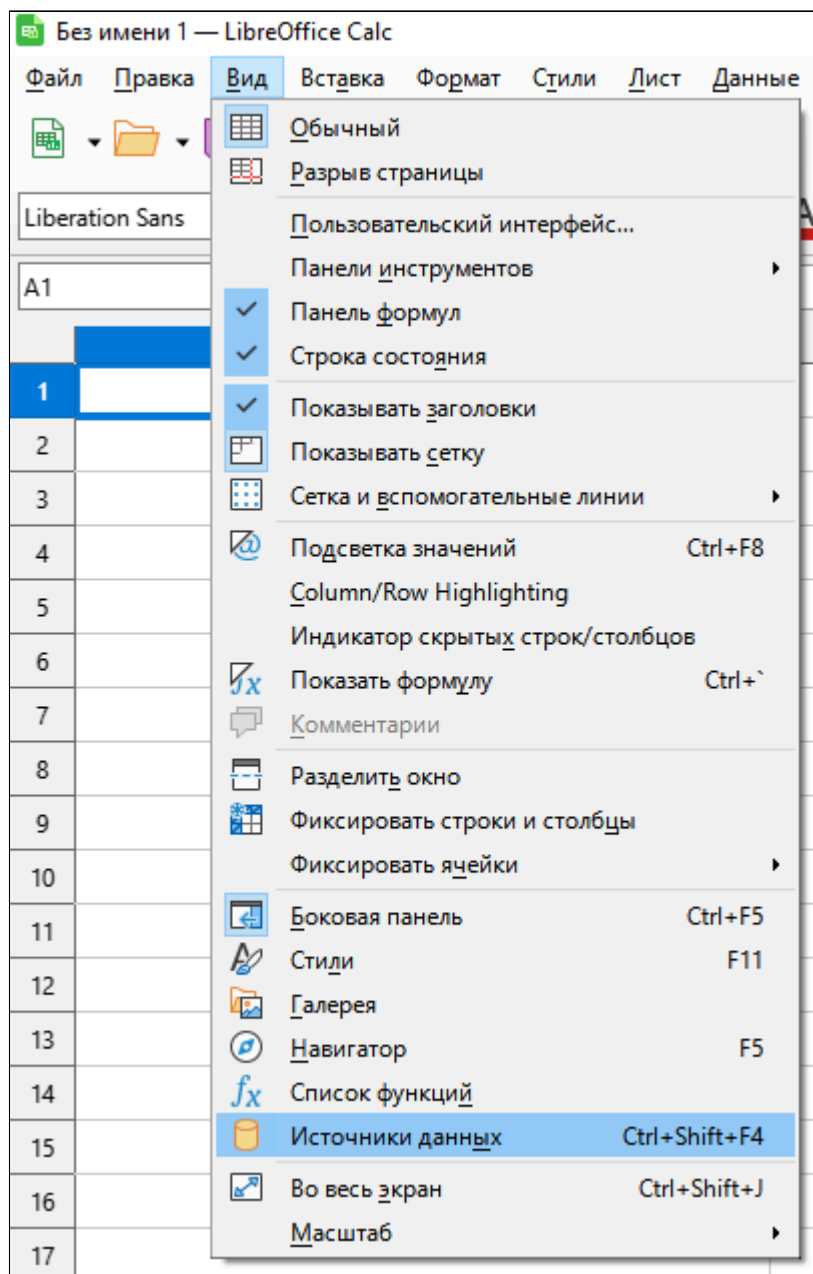
| tagname |
|--|
| Service.Modules.DaServer.SystemLogTraceLevel |
| Service.Modules.HistoryModule.Storages.DB_EVENTS.DB_EV |
| Service.Modules.HistoryModule.Storages.DB_HISTORY.DB_ |
| Service.Modules.HistoryModule.FrameLogEnable |
| Service.InvokeFromJSON |
| Service.Modules.DaServer.Active.Set |
| Service.Modules.TcpServer.FrameLogEnable |
| Service.Modules.AeServer.Active |
| Service.Modules.AeServer.SystemLogTraceLevel.Set |
| Service.Modules.SyslogServer.Id |

Запись 1 из 41 *

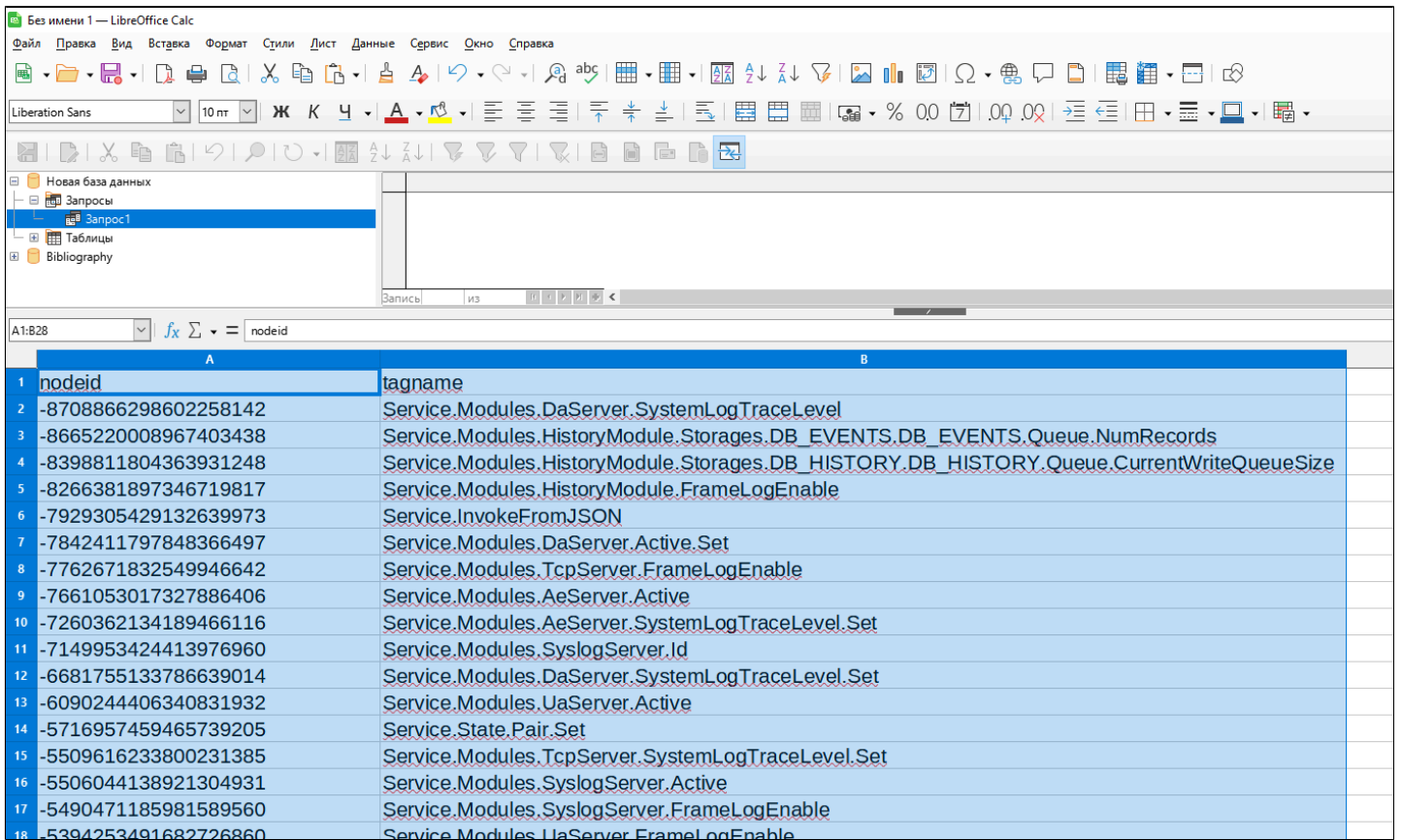
```
select cast (nodeid as char(20)), tagname from nodes
```



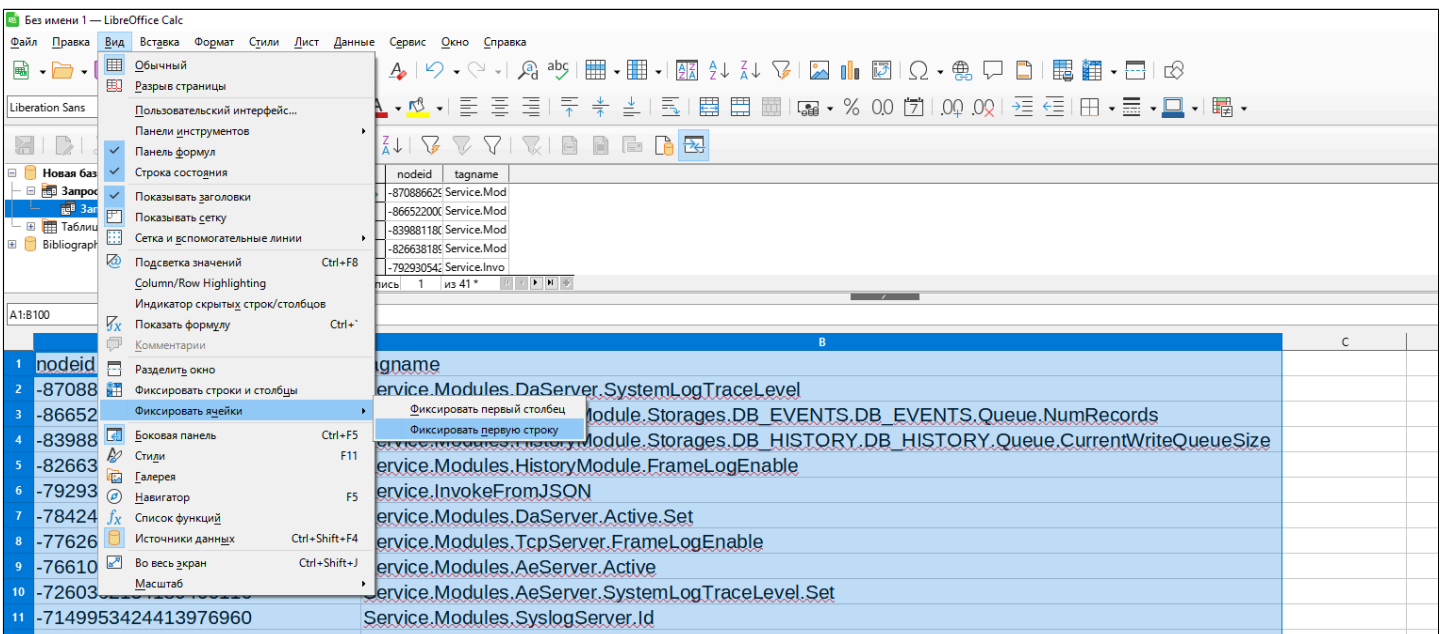
11. В окне LibreOffice Calc выберите пункт меню "Вид" → "Источники данных".



12. В списке запросов выберите сохраненный ранее запрос и перетяните его в область таблицы.



13. Для отображения названий столбцов при прокручивании таблицы выполните команду меню "Вид" → "Фиксировать ячейки" → "Фиксировать первую строку".



1.2.6.2. AstraLinux

Для предоставления данных в LibreOffice Calc через Astra.RMap необходима установка пакетов "libreoffice-base" и "libreoffice-sdbc-postgresql", которые расположены на диске со средствами разработчика.

Для установки выполните команду:

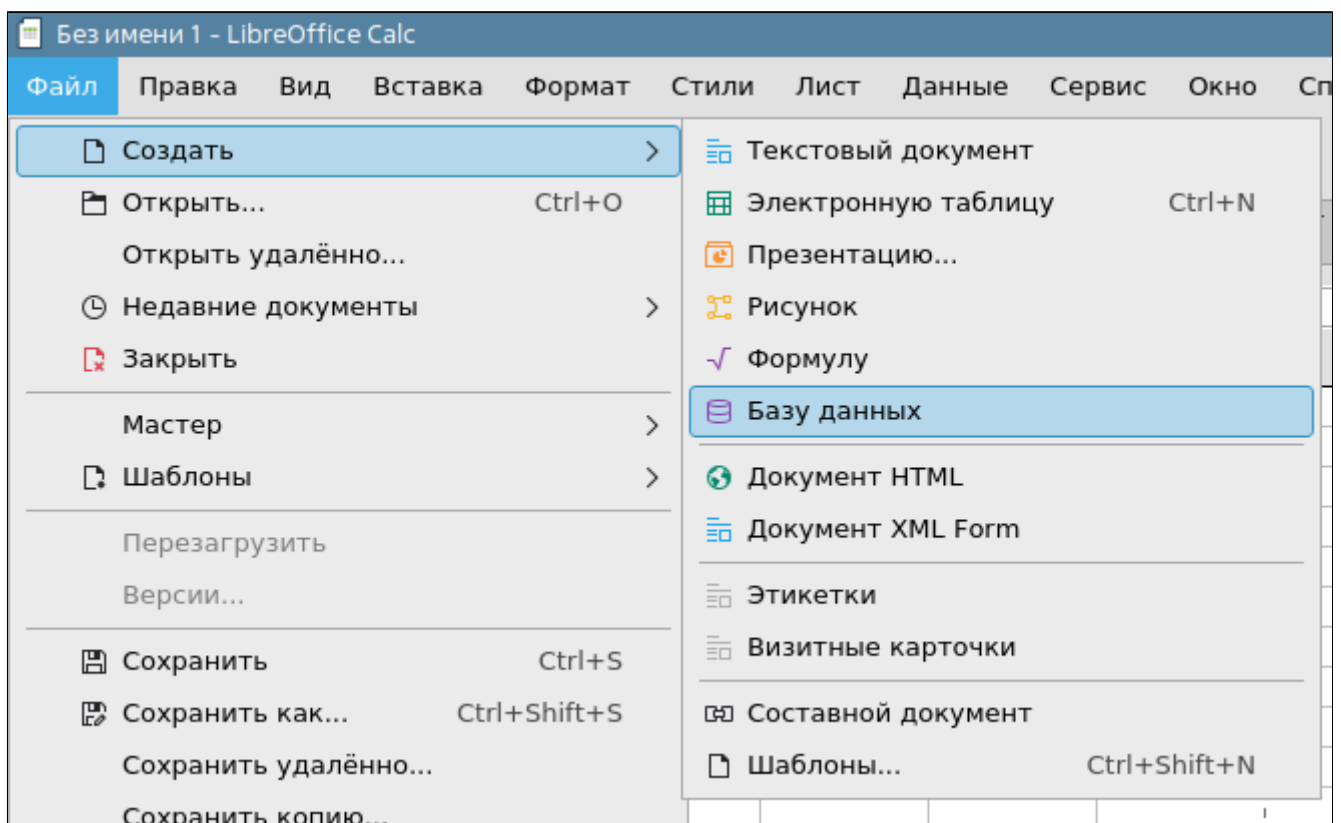


```
sudo apt install libreoffice-base libreoffice-sdbc-postgresql
```

```
astraregul@astraregul:~$ sudo apt install libreoffice-base libreoffice-sdbc-postgresql
```

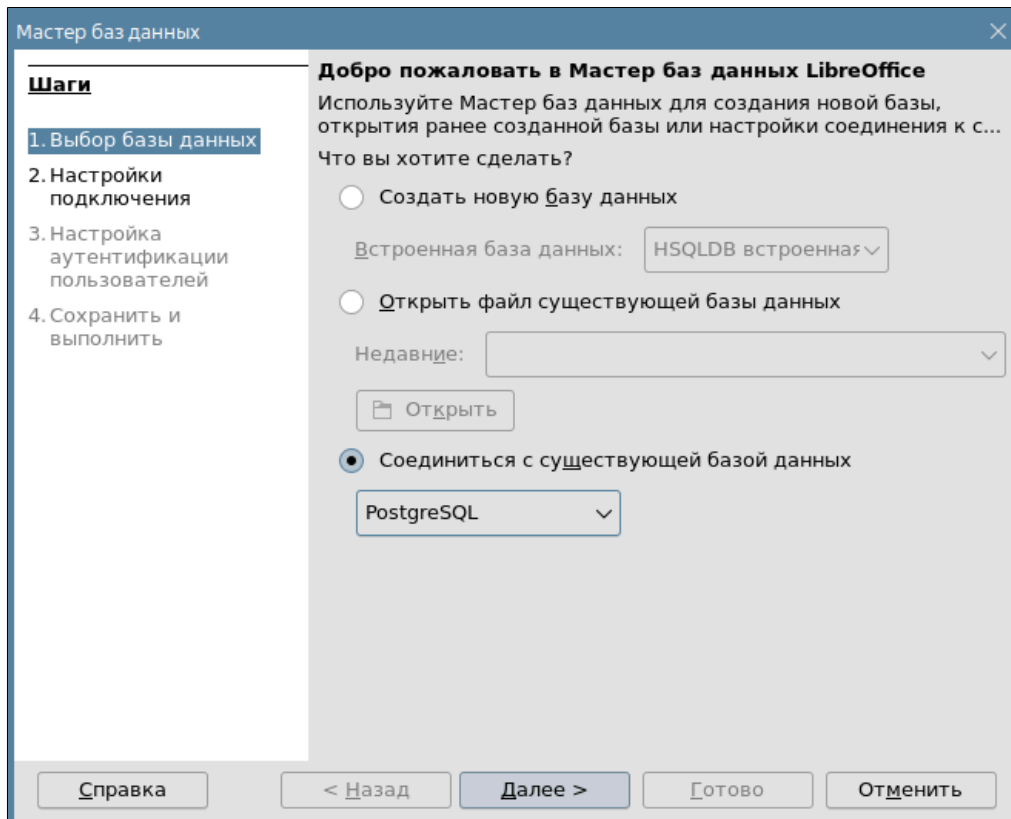
Для предоставления данных запустите LibreOffice Calc и выполните следующие действия:

1. Создайте базу данных. Для этого выполните команду меню "Файл" → "Создать" → "Базу данных".



2. Укажите БД для подключения:

- › выберите пункт Соединиться с существующей базой данных;
- › в выпадающем списке выберите PostgreSQL.



Нажмите кнопку "Далее".

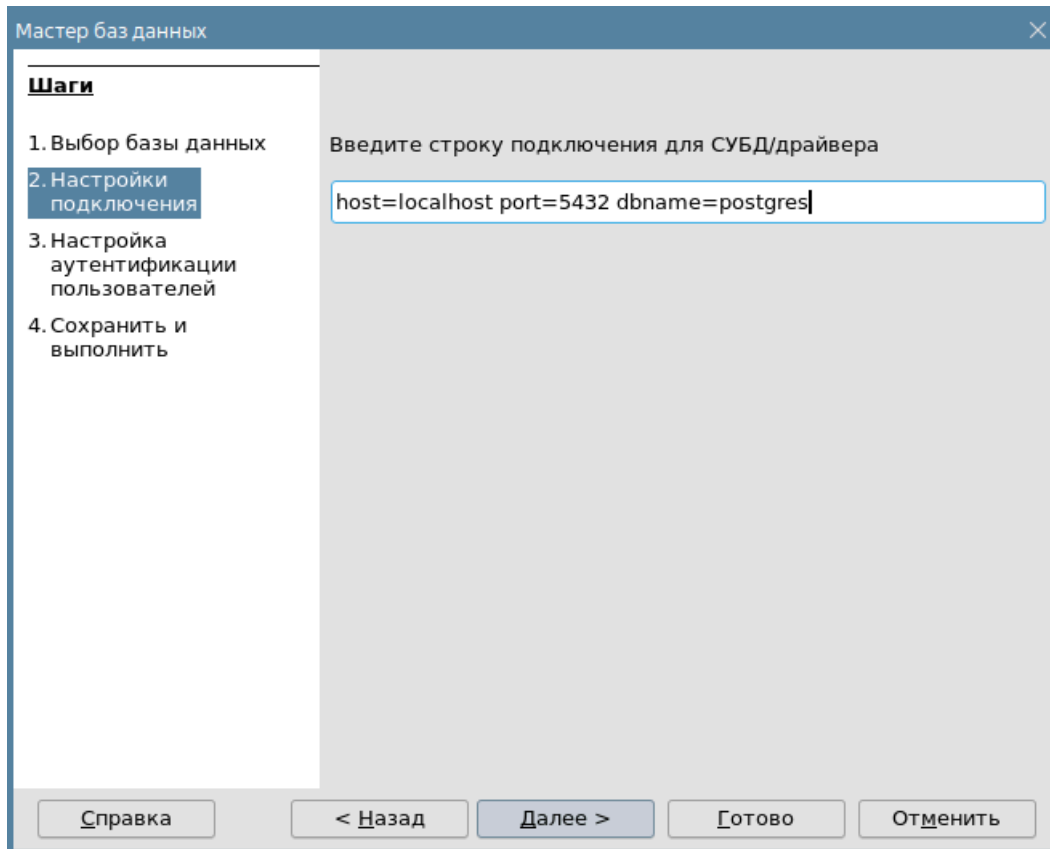
3. Укажите параметры подключения к БД:



```
host=localhost port=5432 dbname=postgres
```

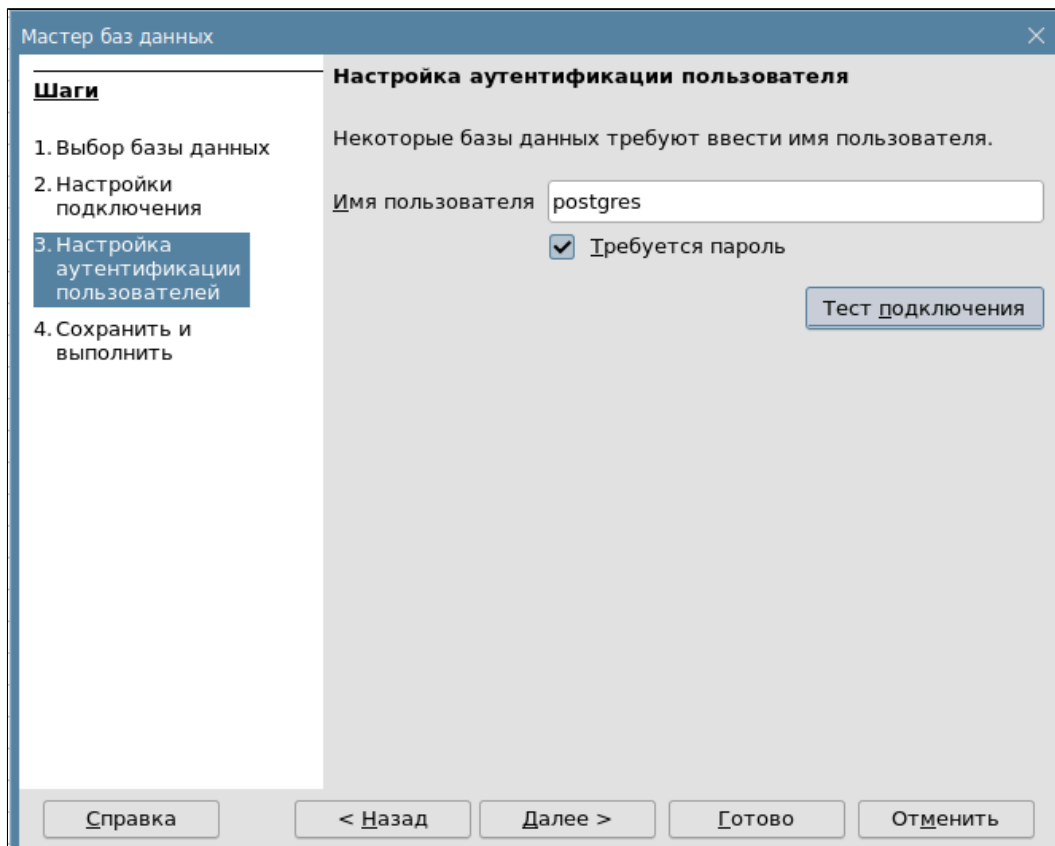
где:

- › «host» - адрес сервера PostgreSQL;
- › «port» - порт подключения к серверу PostgreSQL;
- › «dbname» - имя базы данных PostgreSQL.

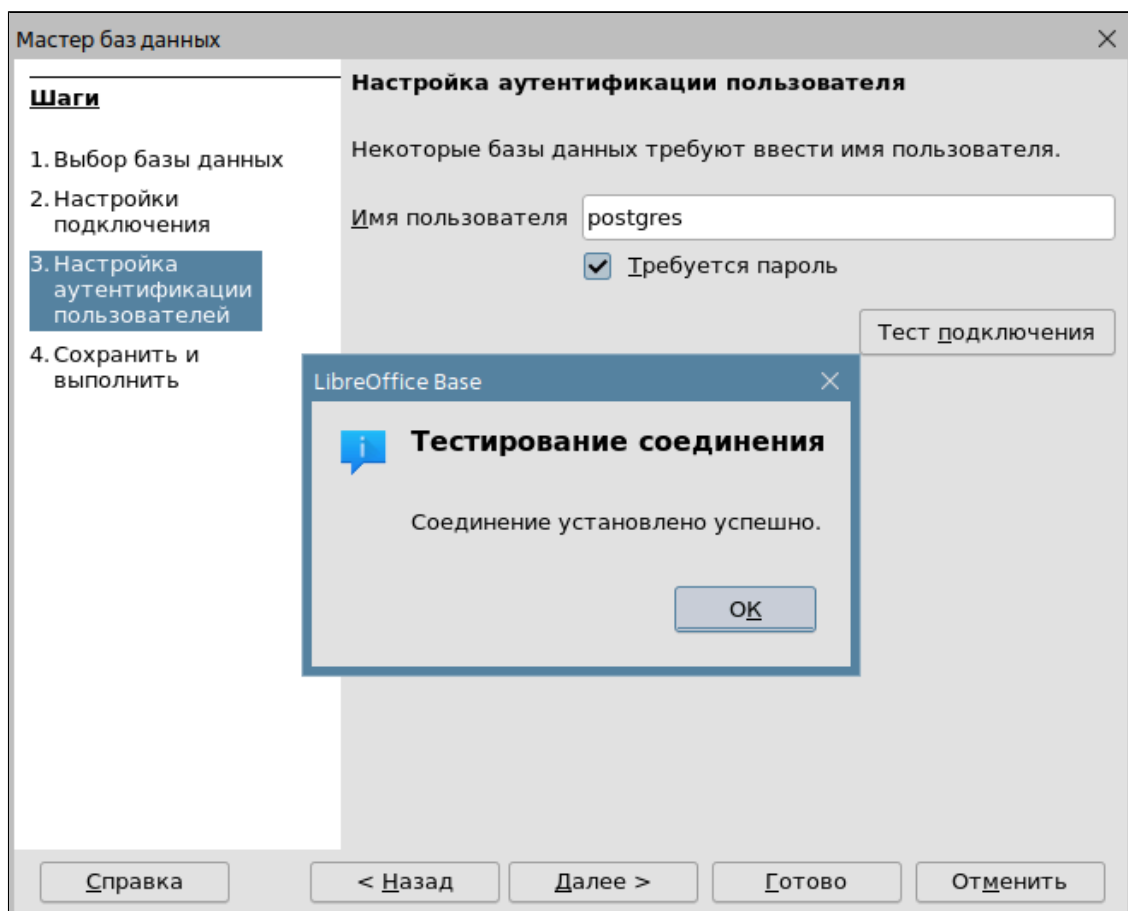


Нажмите кнопку "Далее".

4. Укажите имя пользователя для подключения к серверу PostgreSQL и установите флаг "Требуется пароль".

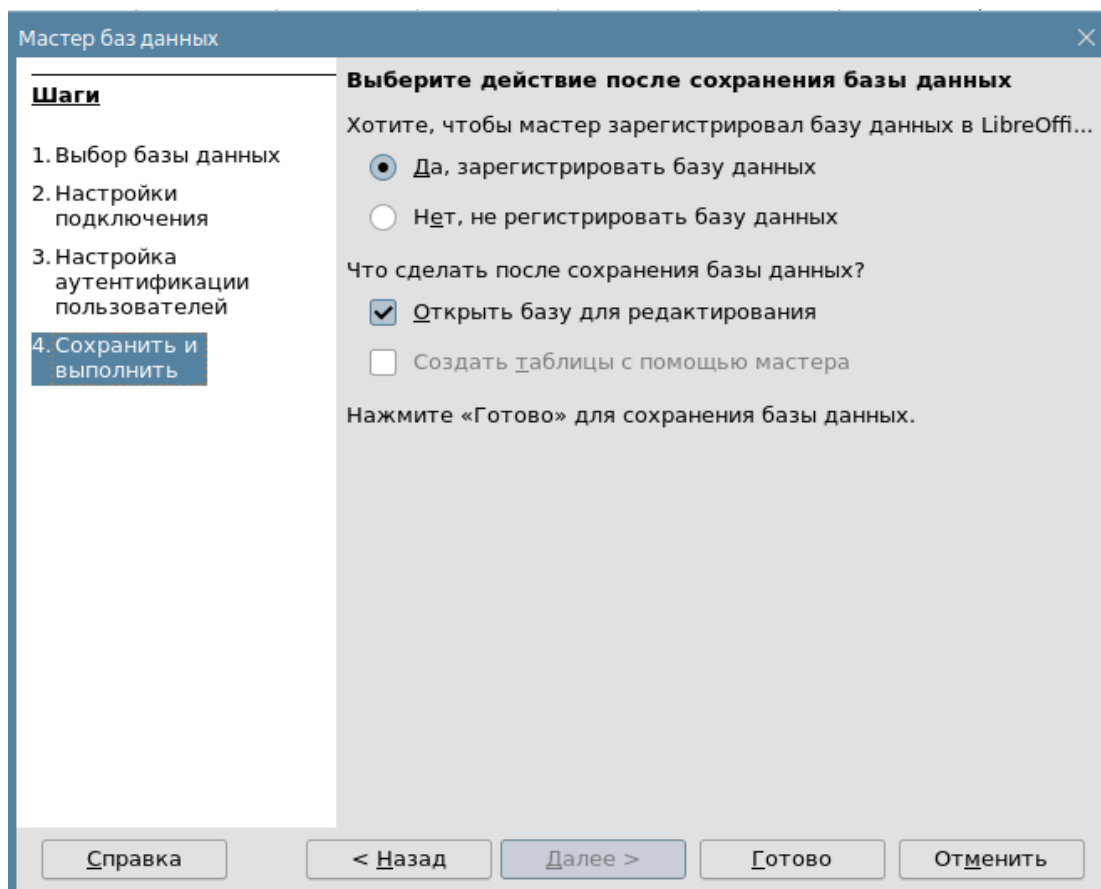


5. Нажмите кнопку "Тест подключения", введите пароль и нажмите кнопку "ОК".

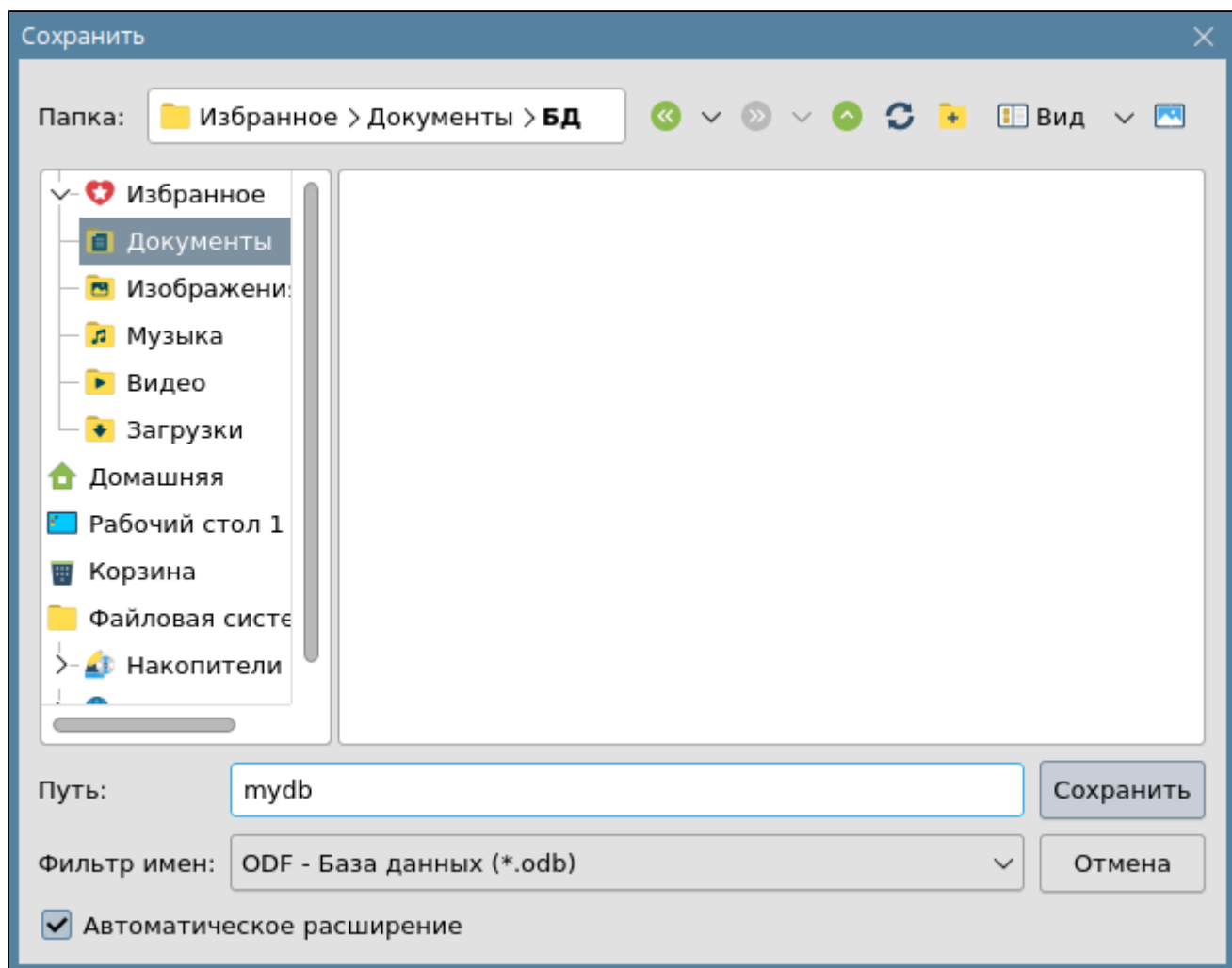


Нажмите кнопку "Далее".

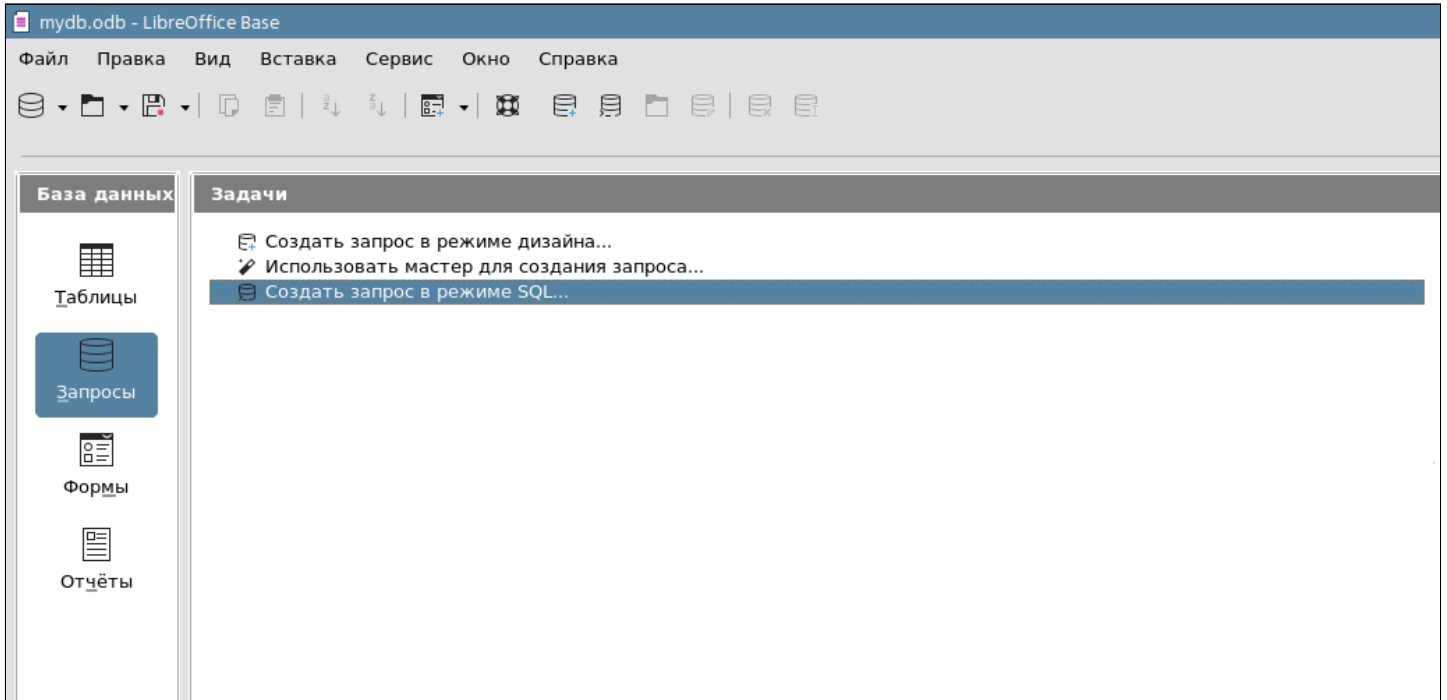
6. Установите флаги "Да, зарегистрировать базу данных" и "Открыть базу для редактирования", и нажмите кнопку "Готово".



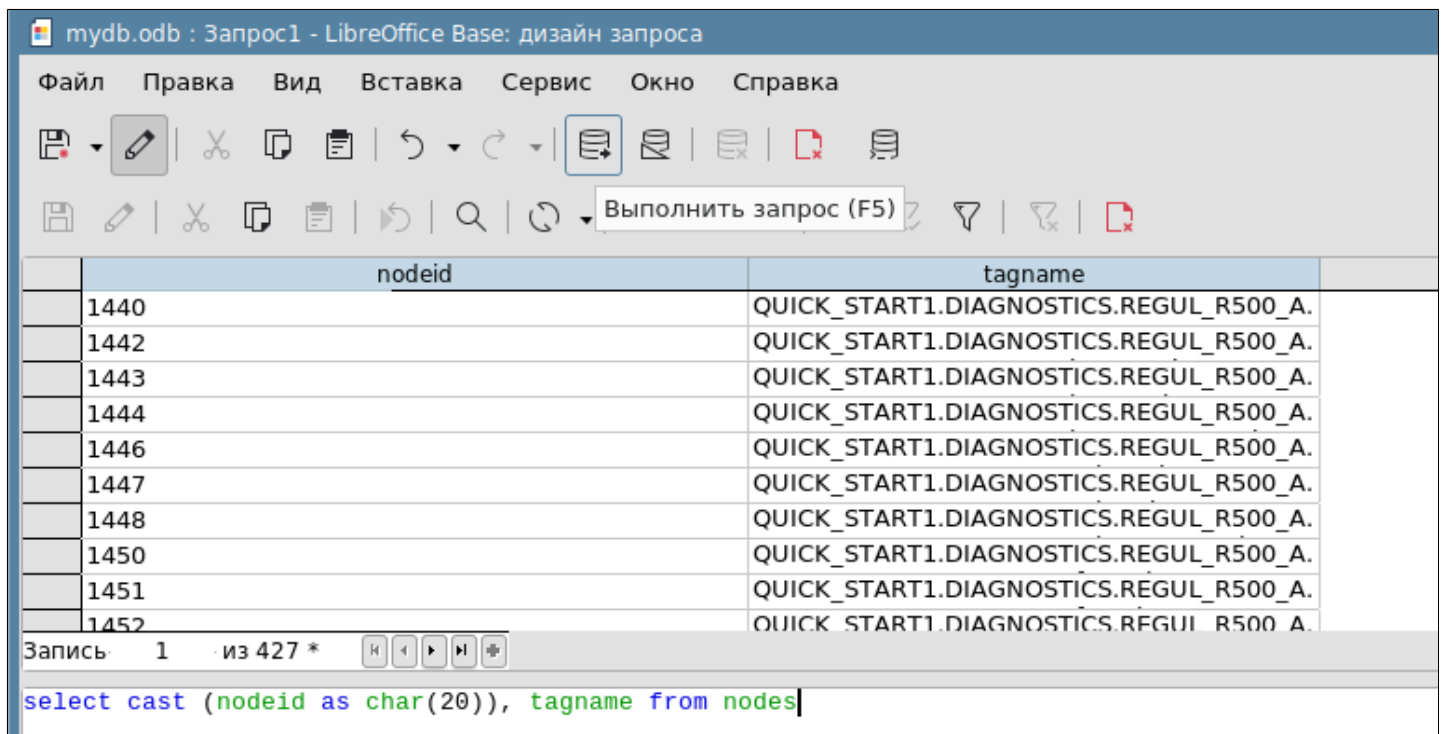
7. В окне проводника при необходимости переименуйте БД и сохраните в необходимой директории.



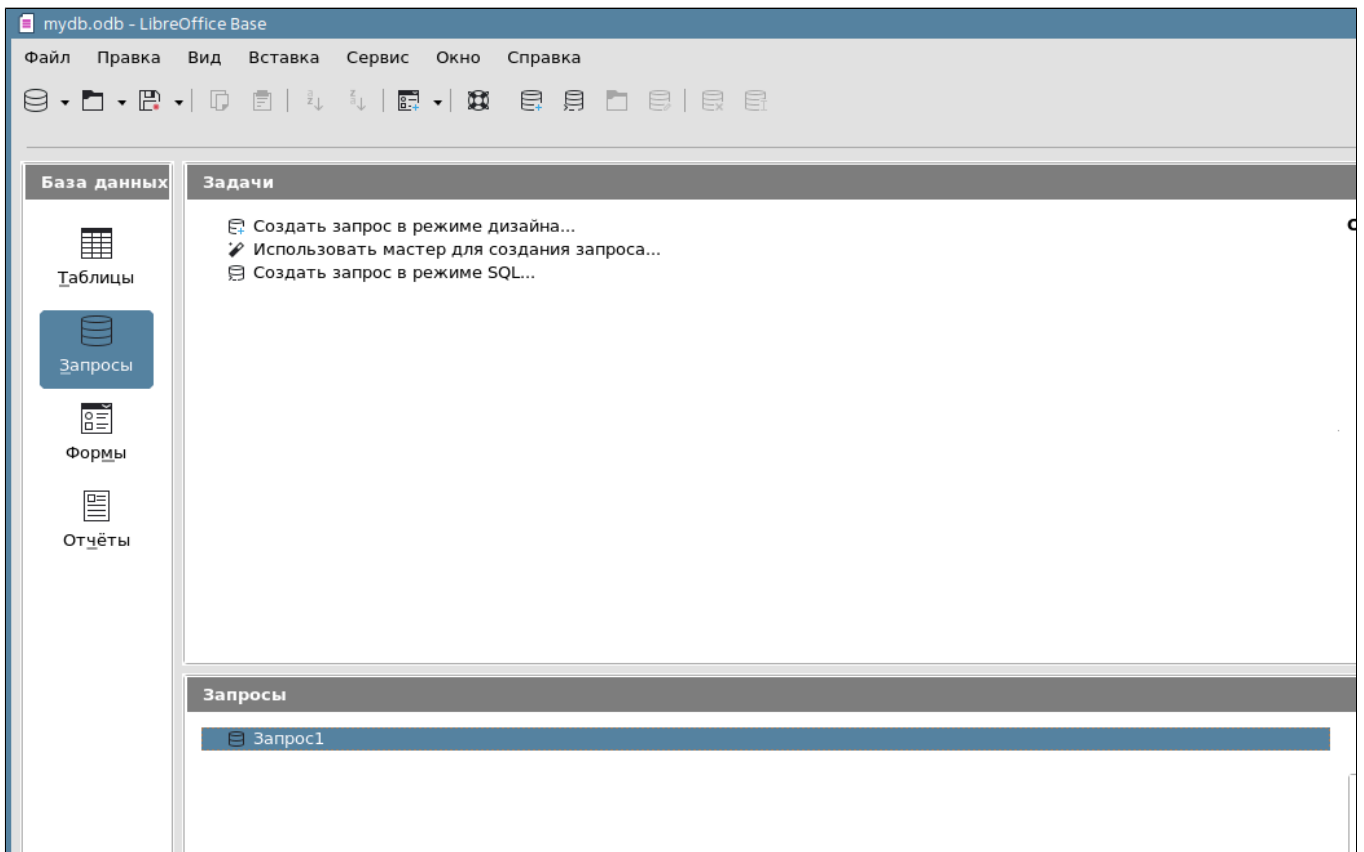
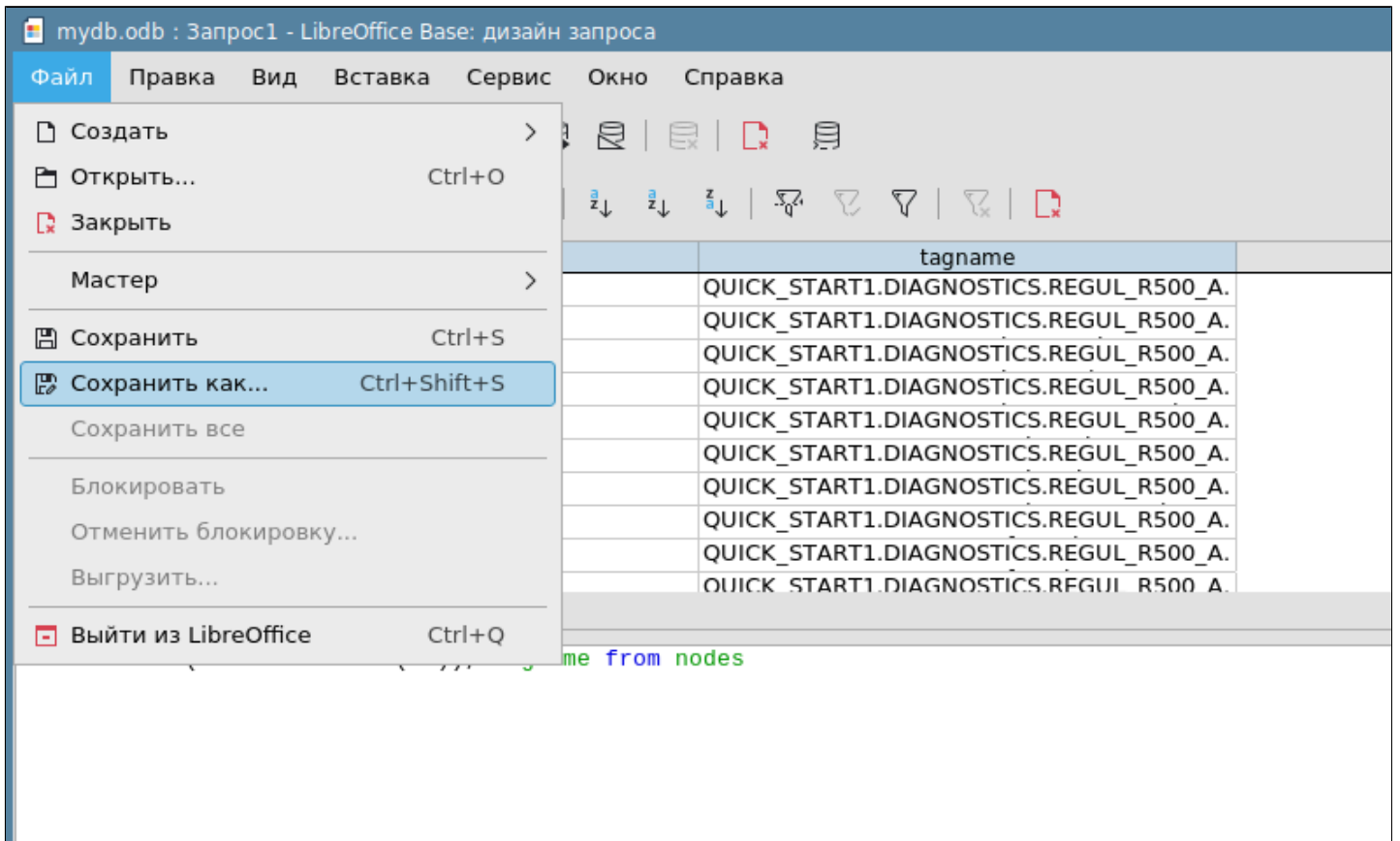
8. В открывшемся окне выберите "Запросы" → "Создать запрос в режиме SQL".



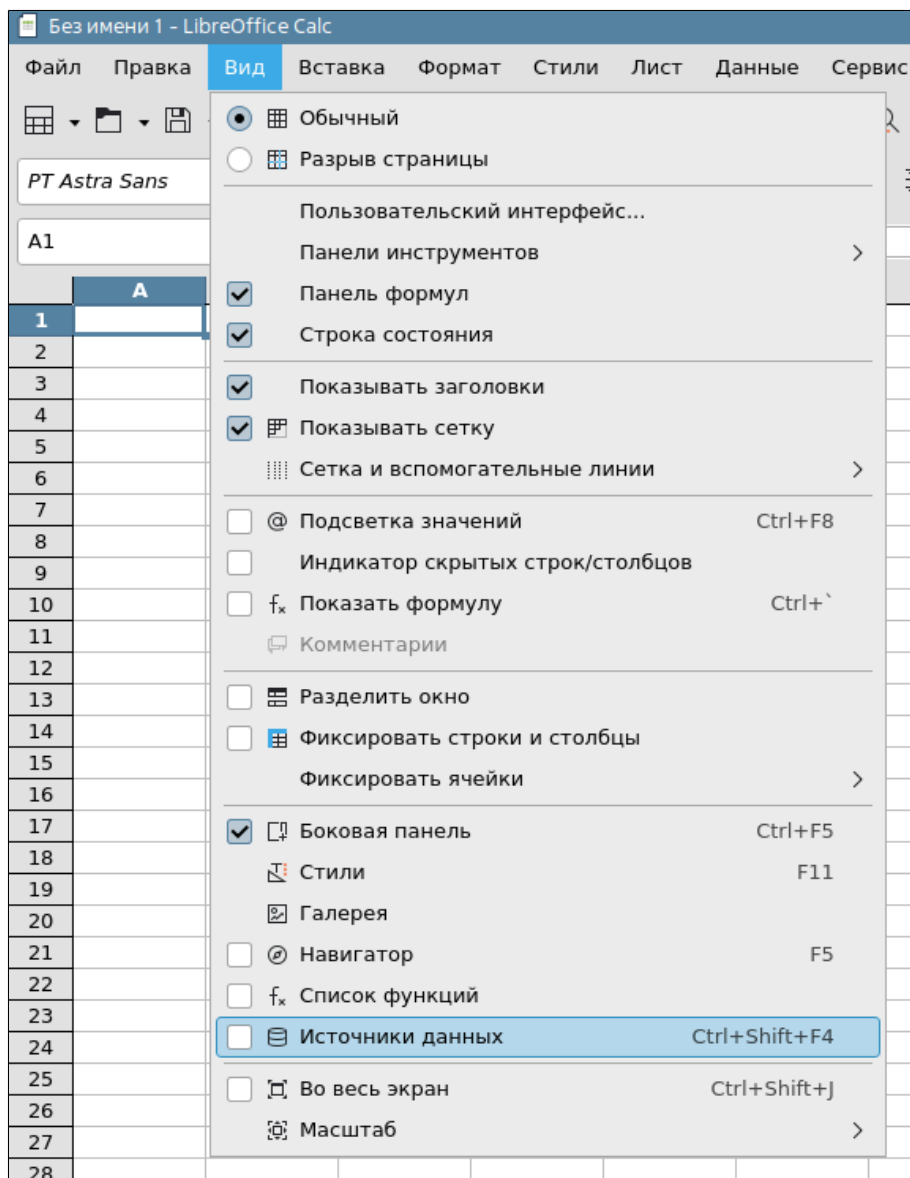
9. Введите требуемый SQL-запрос и выполните его нажав кнопку "Выполнить запрос".



10. Сохраните запрос, выполнив команду меню "Файл" → "Сохранить как...". Сохраненные запросы отображаются в области "Запросы".



11. В окне LibreOffice Calc выберите пункт меню "Вид" → "Источники данных".



12. В списке запросов выберите сохраненный ранее запрос и перетяните его в область таблицы.

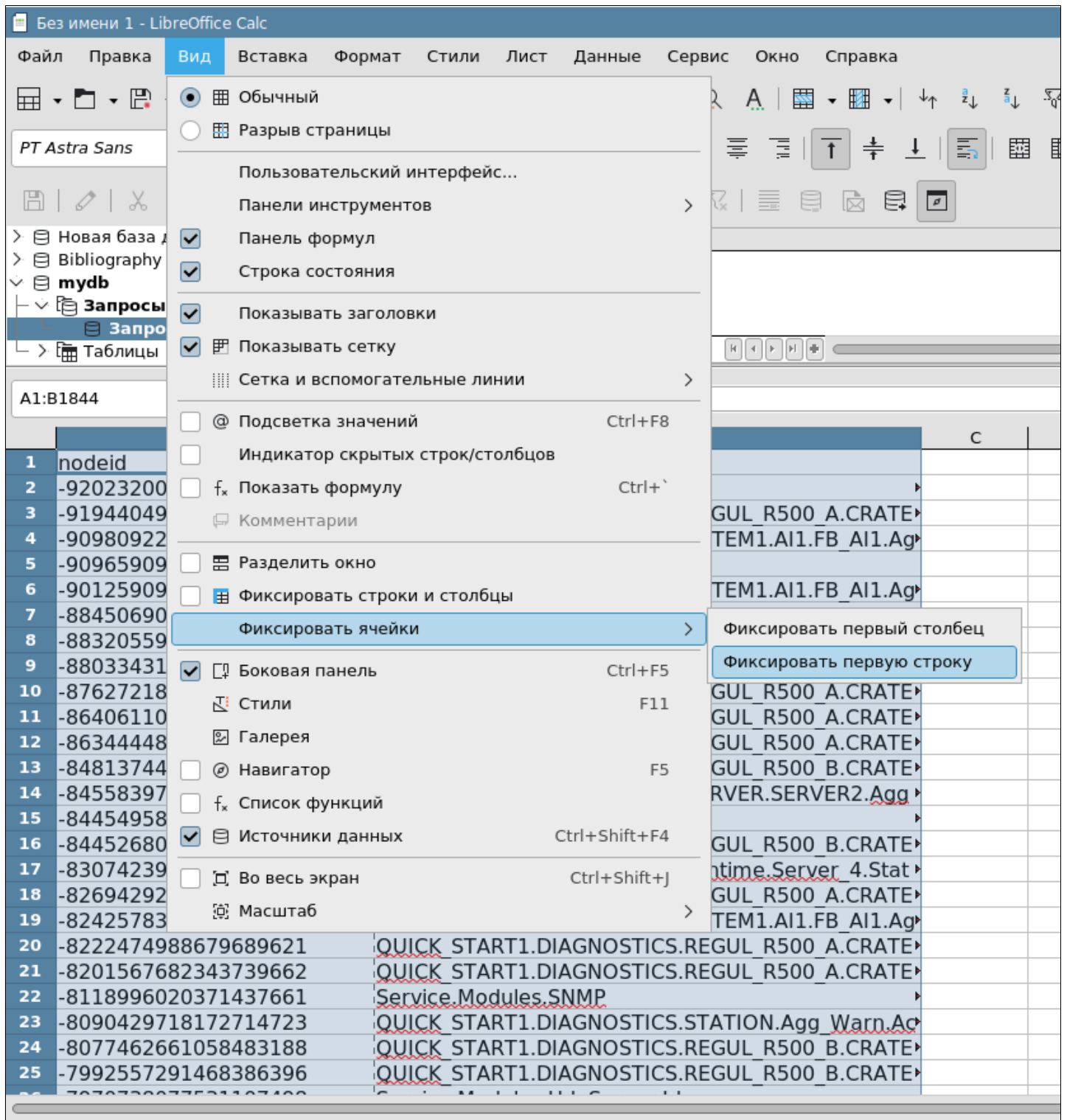
The screenshot shows the LibreOffice Calc application window. The title bar reads "Без имени 1 - LibreOffice Calc". The menu bar includes "Файл", "Правка", "Вид", "Вставка", "Формат", "Стили", "Лист", "Данные", "Сервис", "Окно", and "Справка". The toolbar contains various icons for file operations, editing, and formatting. The sidebar on the left shows a tree view with the following structure:

- Новая база данных
- Bibliography
- mydb
 - Запросы
 - Запрос1
 - Таблицы

The main window displays a table with the following data:

| | A | B | C | D | E | F |
|----|----------------------|---|---|---|---|---|
| 1 | nodeid | tagname | | | | |
| 2 | -9202320016856573880 | Service.Modules.SNMP | | | | |
| 3 | -9194404986514661827 | QUICK_START1.DIAGNOSTICS.REGUL_R500_A.CRATE | | | | |
| 4 | -9098092237553157711 | QUICK_START1.REGUL_R500.SYSTEM1.AI1.FB_AI1.Ag | | | | |
| 5 | -9096590950352657846 | Service.State.Pair.Set | | | | |
| 6 | -9012590996443393586 | QUICK_START1.REGUL_R500.SYSTEM1.AI1.FB_AI1.Ag | | | | |
| 7 | -8845069086842652079 | QUICK_START1.DIAGNOSTICS.REGUL_R500_A.CU_00 | | | | |
| 8 | -8832055984210759293 | Service.Modules.HistoryModule.FrameLogEnable | | | | |
| 9 | -8803343185187948247 | Service.Modules.UaServer.Active.Set | | | | |
| 10 | -8762721818425268731 | QUICK_START1.DIAGNOSTICS.REGUL_R500_A.CRATE | | | | |
| 11 | -8640611039435677964 | QUICK_START1.DIAGNOSTICS.REGUL_R500_A.CRATE | | | | |
| 12 | -8634444834111281462 | QUICK_START1.DIAGNOSTICS.REGUL_R500_A.CRATE | | | | |
| 13 | -8481374492237721385 | QUICK_START1.DIAGNOSTICS.REGUL_R500_B.CRATE | | | | |
| 14 | -8455839740122579786 | QUICK_START1.DIAGNOSTICS.SERVER.SERVER2.Ag | | | | |
| 15 | -8445495879906855905 | Service.Modules.SNMP | | | | |
| 16 | -8445268098591730919 | QUICK_START1.DIAGNOSTICS.REGUL_R500_B.CRATE | | | | |
| 17 | -8307423938209487493 | Service.Modules.OpcUaClient.Runtime.Server_4.Stat | | | | |
| 18 | -8269429274205822495 | QUICK_START1.DIAGNOSTICS.REGUL_R500_A.CRATE | | | | |
| 19 | -8242578327226020075 | QUICK_START1.REGUL_R500.SYSTEM1.AI1.FB_AI1.Ag | | | | |
| 20 | -8222474988679689621 | QUICK_START1.DIAGNOSTICS.REGUL_R500_A.CRATE | | | | |

13. Для отображения названий столбцов при прокручивании таблицы выполните команду меню "Вид" → "Фиксировать ячейки" → "Фиксировать первую строку".



1.3. Работа с БД MySQL

MySQL - свободная реляционная система управления базами данных.

1.3.1. MySQL

- › [Установка](#)
- › [Создание таблицы](#)
- › [Настройка пользователя](#)
- › [Создание источника данных](#)

1.3.1.1. Установка

MySQL

Для установки MySQL выполните следующие действия:

1. Скачайте установщик MySQL. Для этого можно воспользоваться [ССЫЛКОЙ](#). Нажмите вторую кнопку "Download" для скачивания дистрибутива установки mysql-installer-community-8.x.xx.x.msi.

General Availability (GA) Releases Archives ⓘ

MySQL Installer 8.0.36

Note: MySQL 8.0 is the final series with MySQL Installer. As of MySQL 8.1, use a MySQL product's MSI or Zip archive for installation. MySQL Server 8.1 and higher also bundle MySQL Configurator, a tool that helps configure MySQL Server.

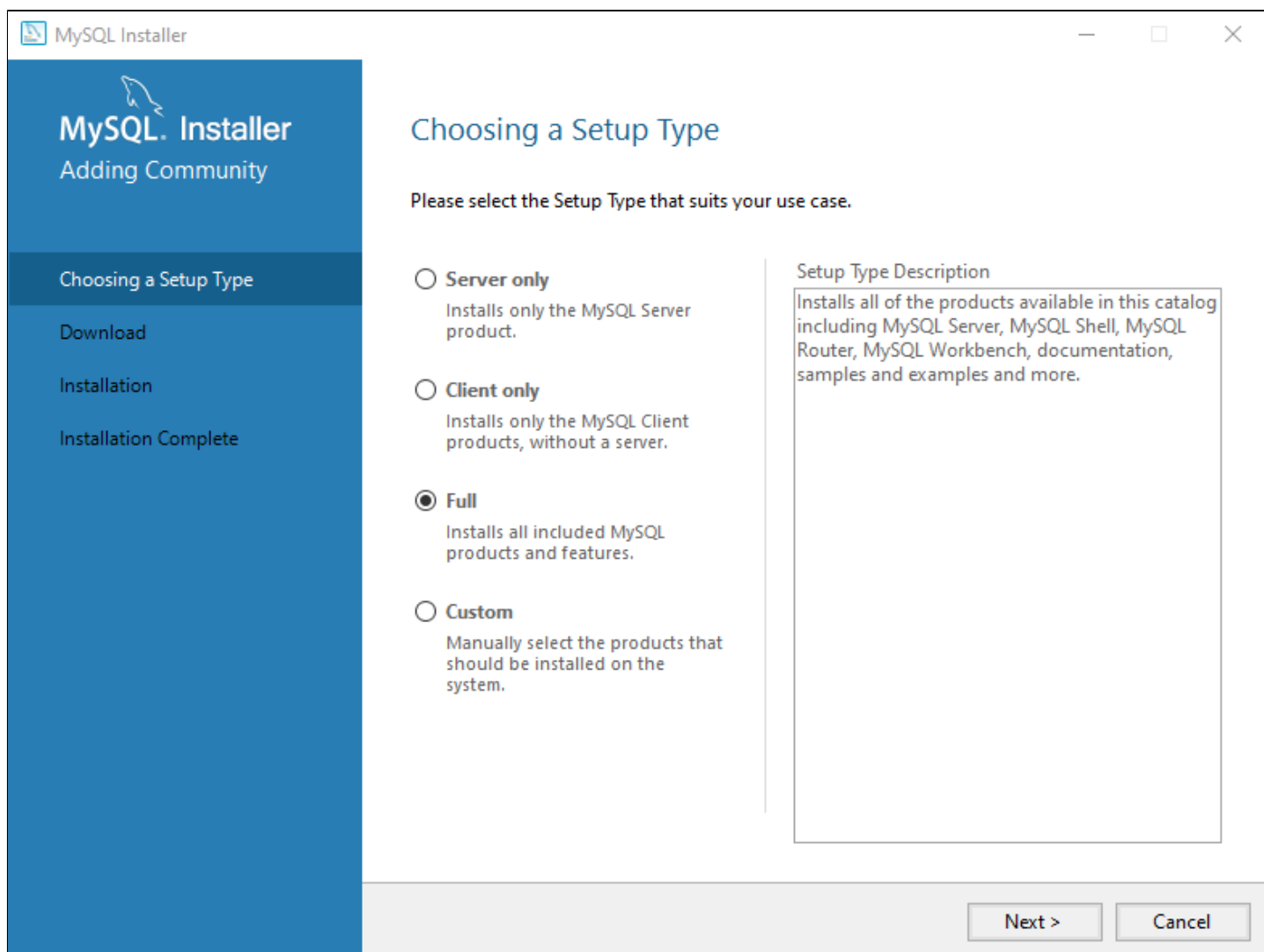
Select Version:
8.0.36

Select Operating System:
Microsoft Windows

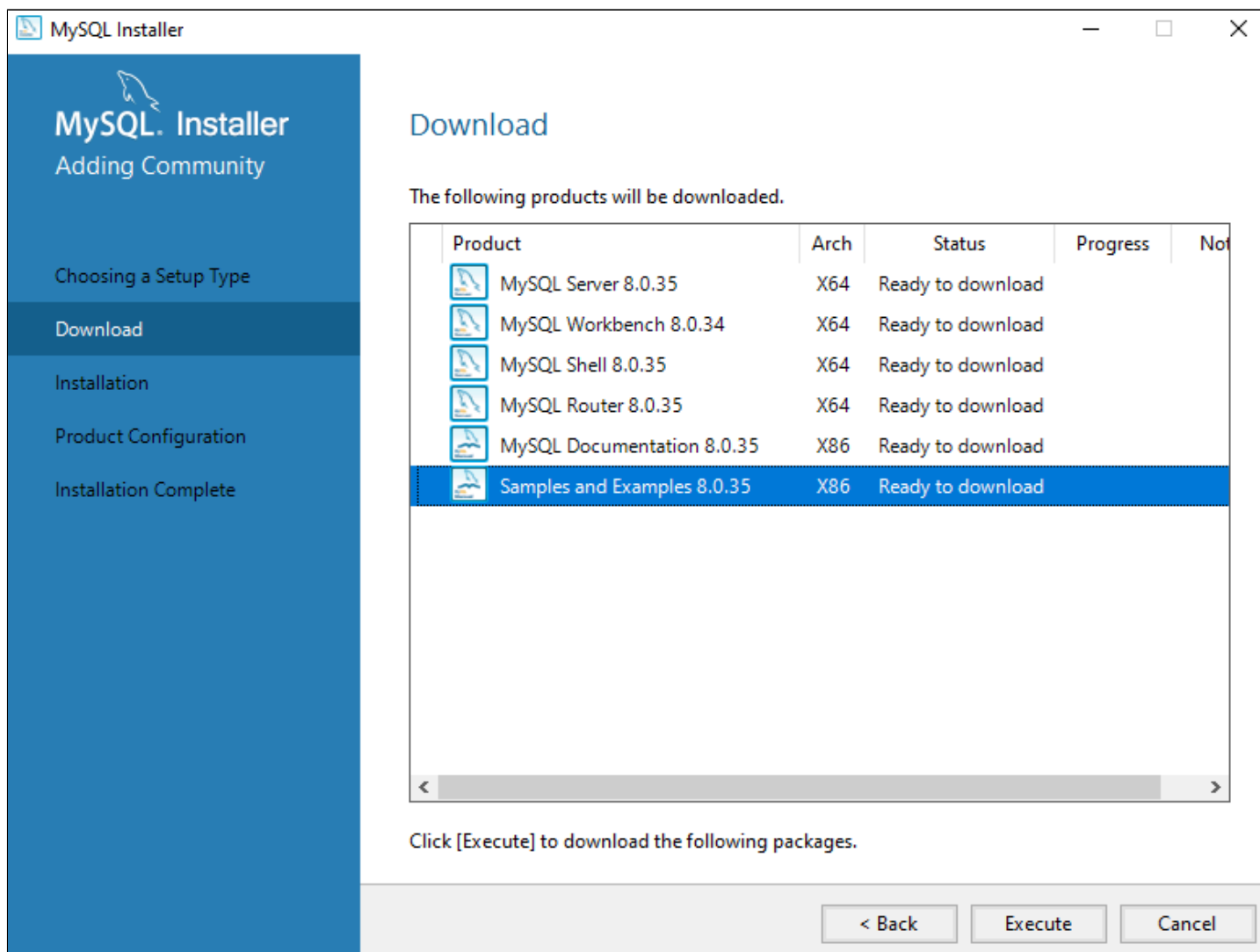
| | | | |
|---|--------|--------|--------------------------|
| Windows (x86, 32-bit), MSI Installer (mysql-installer-web-community-8.0.36.0.msi) | 8.0.36 | 2.1M | Download |
| Windows (x86, 32-bit), MSI Installer (mysql-installer-community-8.0.36.0.msi) | 8.0.36 | 285.3M | Download |

We suggest that you use the [MD5 checksums](#) and [GnuPG signatures](#) to verify the integrity of the packages you download.

2. Запустите дистрибутив установки `mysql-installer-community-8.x.xx.x.msi` и выберите полную установку MySQL и нажмите кнопку "Next".



3. Откроется окно со списком дистрибутивов к установке. Нажмите "Execute".



Начнется установка дистрибутивов. Дождитесь полной установки.

MySQL Installer












MySQL. Installer

Adding Community

- Choosing a Setup Type
- Download**
- Installation
- Product Configuration
- Installation Complete

Download

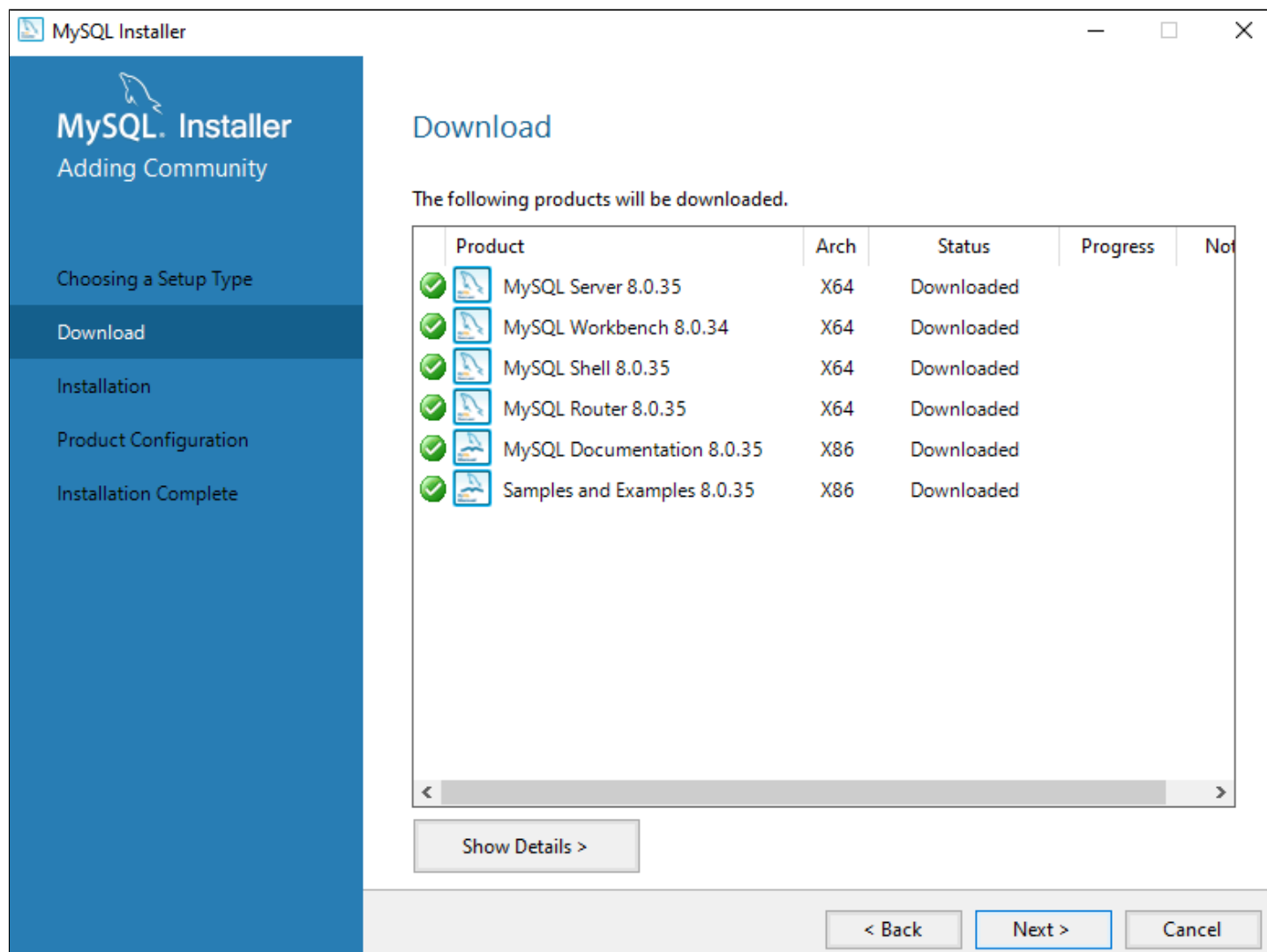
The following products will be downloaded.

| | Product | Arch | Status | Progress | Not |
|---|---|------|-------------------|----------|-----|
|  |  MySQL Server 8.0.35 | X64 | Downloading... | 4% | |
|  |  MySQL Workbench 8.0.34 | X64 | Downloading... | 10% | |
|  |  MySQL Shell 8.0.35 | X64 | Downloading... | 8% | |
|  |  MySQL Router 8.0.35 | X64 | Downloading... | 16% | |
|  |  MySQL Documentation 8.0.35 | X86 | Downloading... | 10% | |
| |  Samples and Examples 8.0.35 | X86 | Ready to download | | |

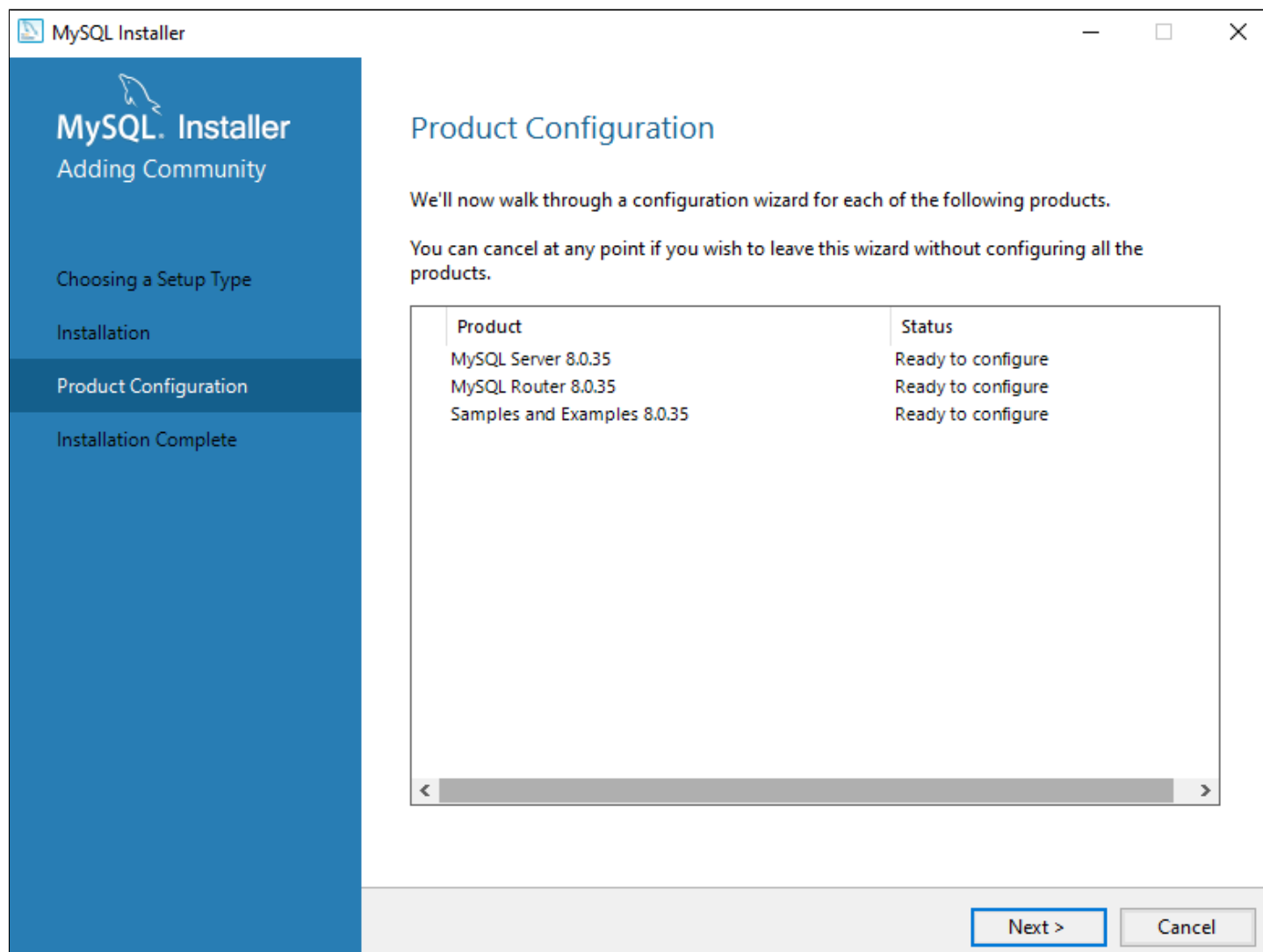
[Show Details >](#)

[< Back](#) [Execute](#) [Cancel](#)

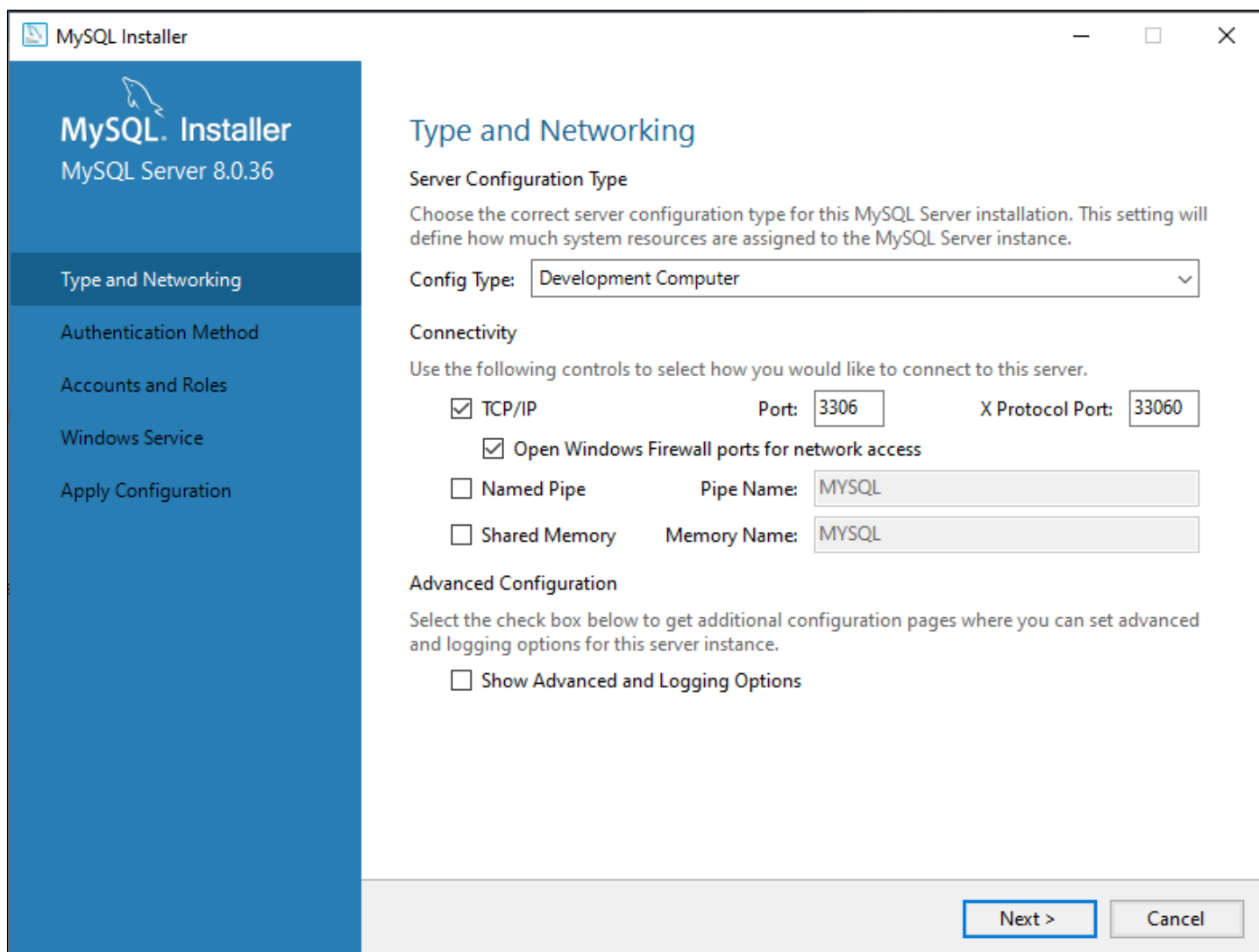
4. После успешной установки нажмите "Next".



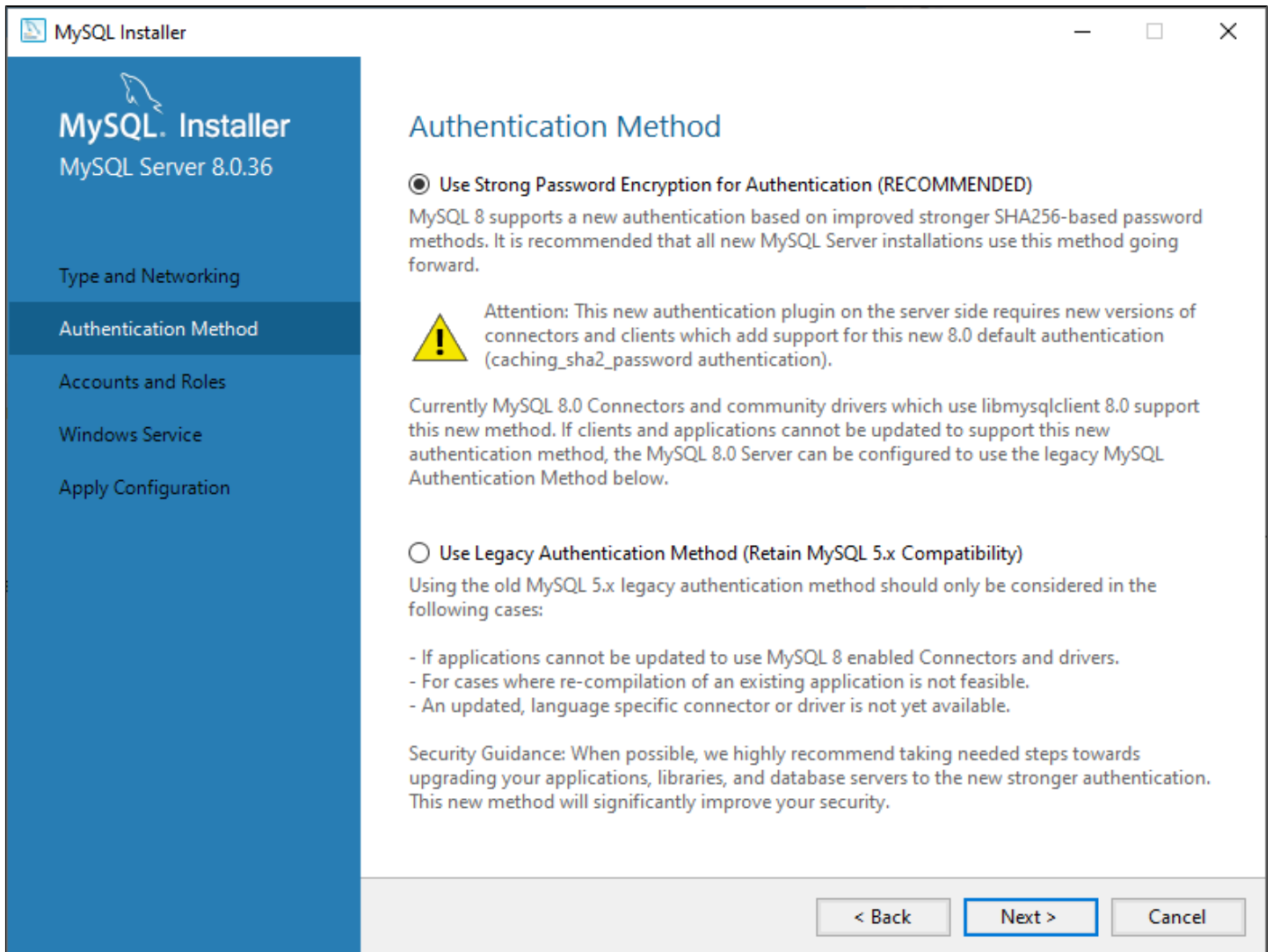
5. На вкладке конфигурации нажмите "Next".



6. На вкладке "Type and networking" оставьте значения по умолчанию и нажмите кнопку "Next".



7. На вкладке "Authentication Method" оставьте значения по умолчанию и нажмите кнопку "Next".



8. На вкладке "Accounts and Roles" введите пароль для пользователя root и нажмите кнопку "Next".

Пример: 1234

MySQL Installer

MySQL. Installer

MySQL Server 8.0.35

- Type and Networking
- Authentication Method
- Accounts and Roles**
- Windows Service
- Server File Permissions
- Apply Configuration

Accounts and Roles

Root Account Password
Enter the password for the root account. Please remember to store this password in a secure place.

MySQL Root Password:

Repeat Password:

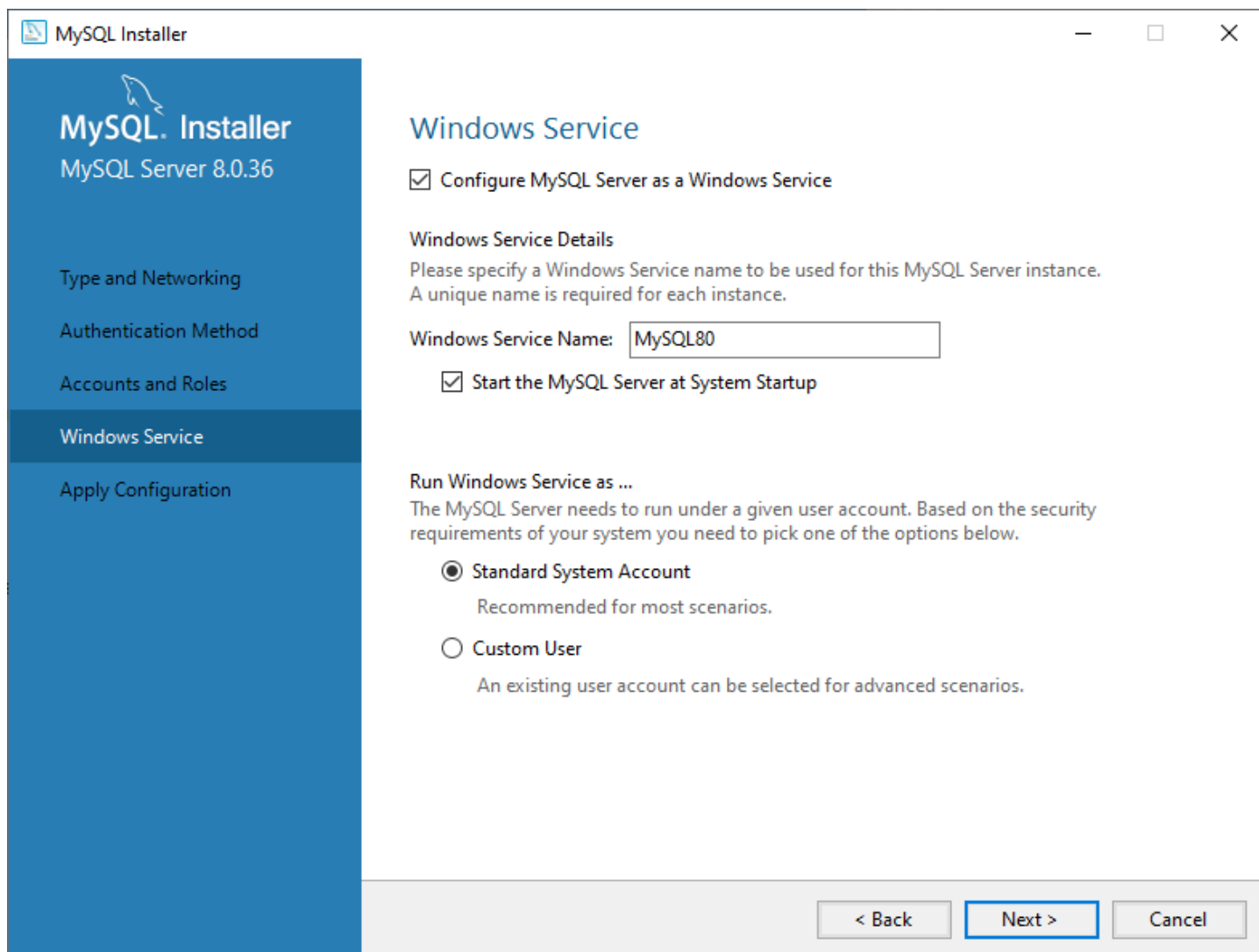
Password strength: **Weak**

MySQL User Accounts
Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.

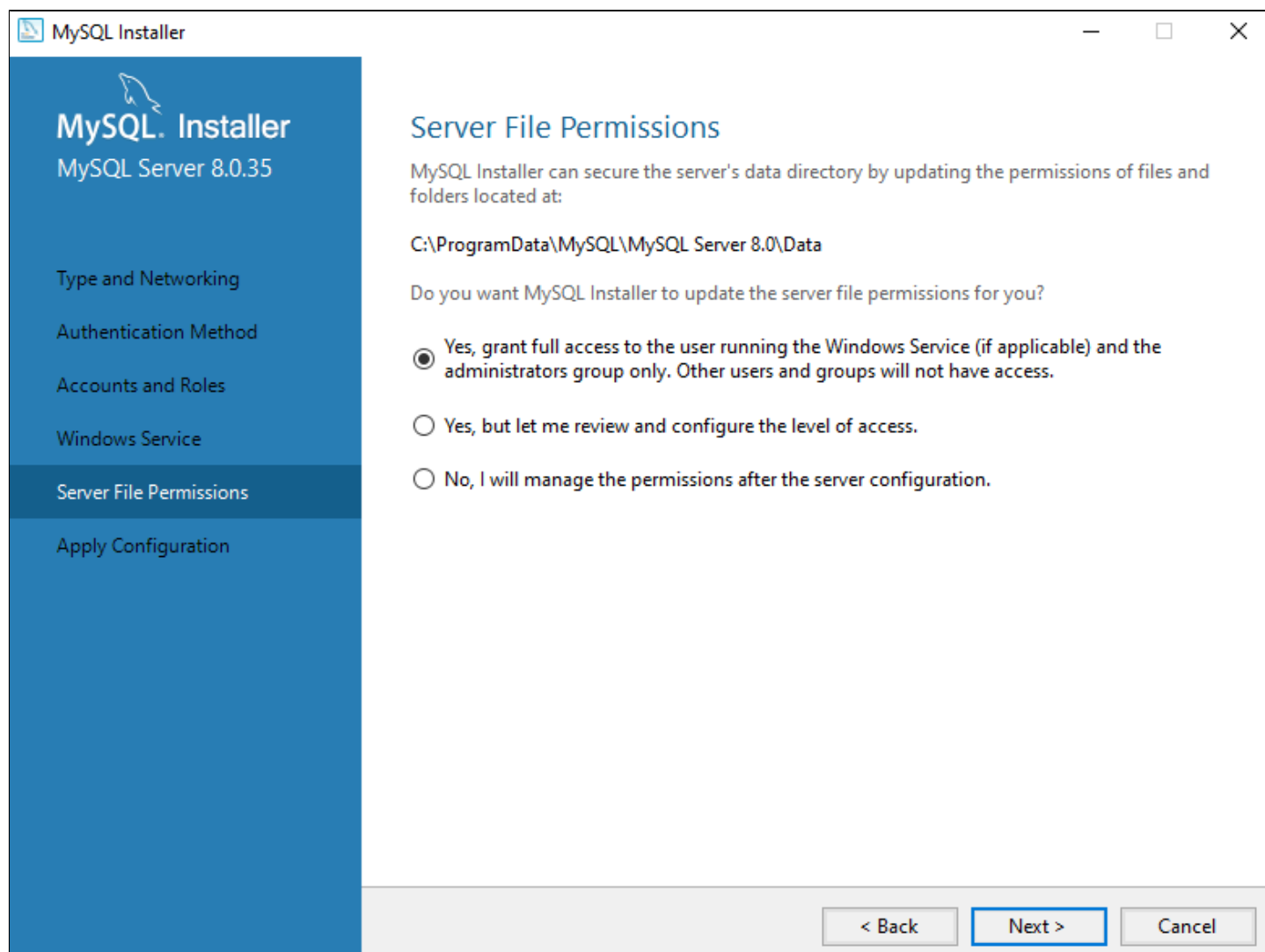
| MySQL User Name | Host | User Role |
|-----------------|------|-----------|
|-----------------|------|-----------|

< Back Next > Cancel

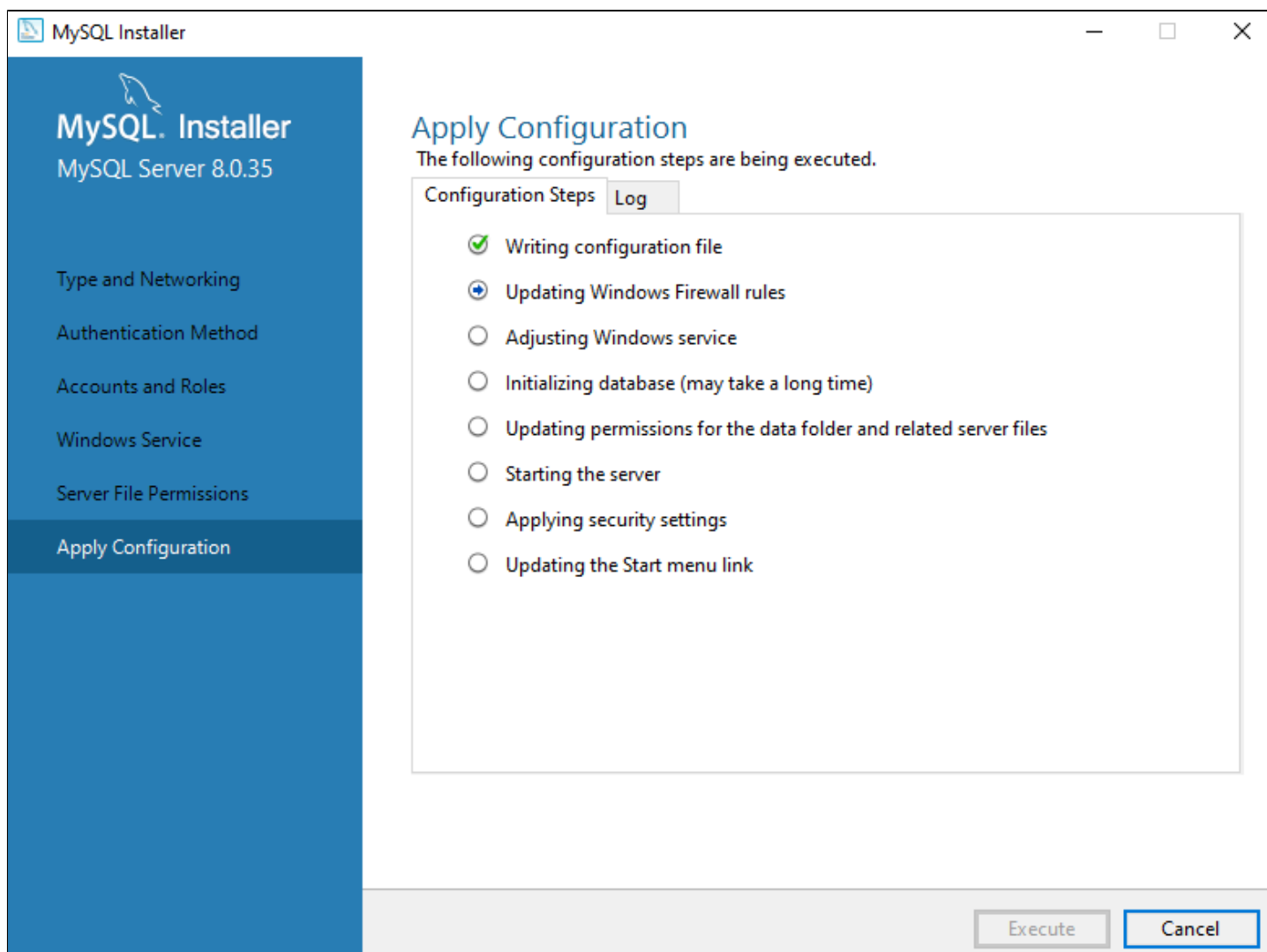
9. На вкладке "Windows Service" оставьте значения по умолчанию.



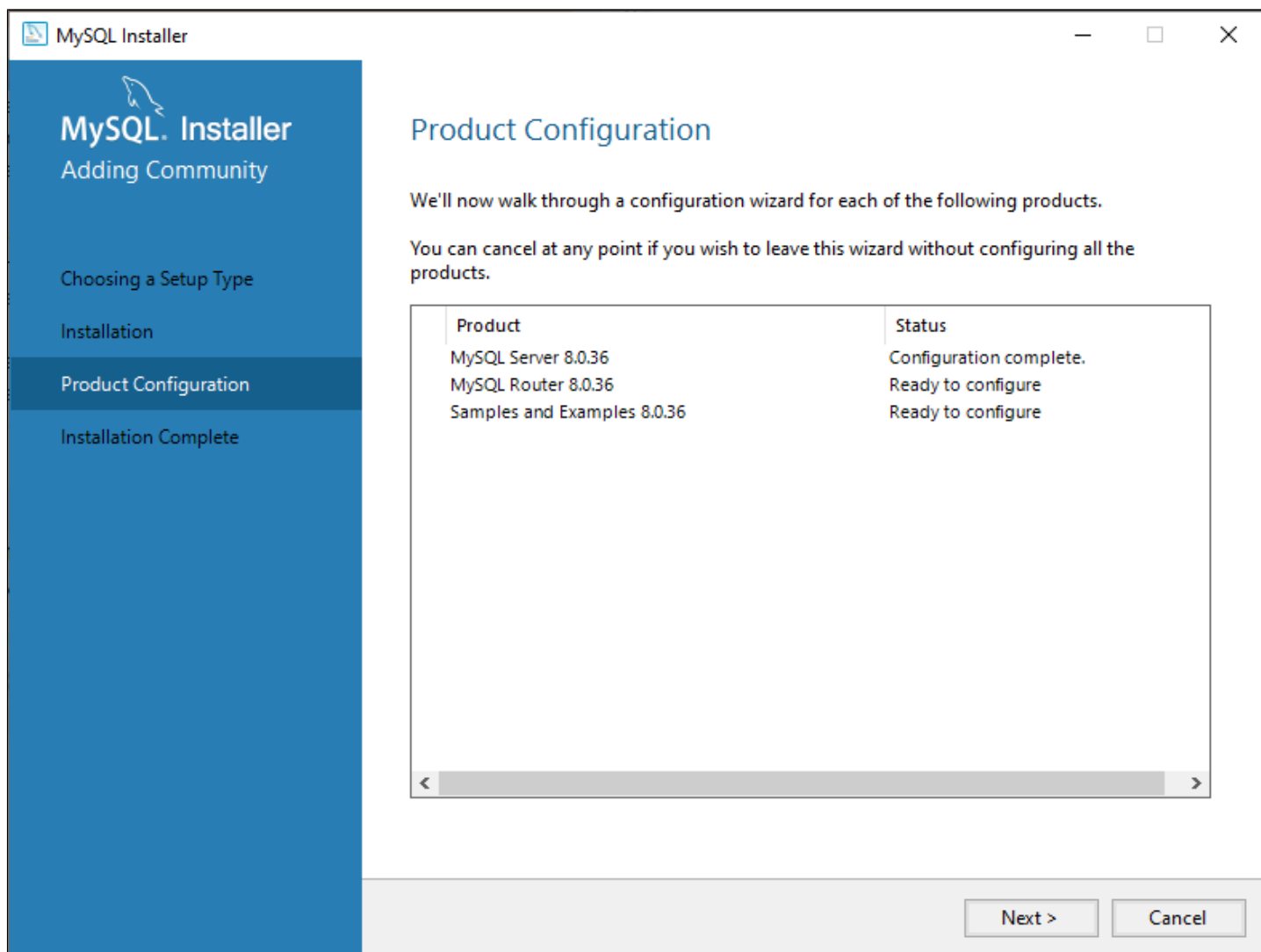
10. На вкладке "Server File Permissions" оставьте значения по умолчанию и нажмите кнопку "Next".



11. На вкладке "Apply Configuration" нажмите "Execute" и ожидайте применение конфигурации. После этого нажмите кнопку "Finish".



12. На вкладке конфигурации продукта нажмите кнопку "Next".



13. На вкладке "MySQL Router Configuration" оставьте значения по умолчанию и нажмите кнопку "Finish".

MySQL Installer

MySQL Router 8.0.36

MySQL Router Configuration

MySQL Router Configuration

Bootstrap MySQL Router for use with InnoDB Cluster

This wizard can bootstrap MySQL Router to direct traffic between MySQL applications and InnoDB Cluster. Applications that connect to the router will be automatically directed to an available read/write or read-only member of the cluster.

The bootstrapping process requires a connection to InnoDB Cluster. In order to register the MySQL Router for monitoring, use the current Read/Write instance of the cluster.

Hostname:

Port:

Management User:

Password:

MySQL Router requires specification of a base port (between 80 and 65532). The first port is used for classic read/write connections. The other ports are computed sequentially after the first port. If any port is indicated to be in use, please change the base port.

Classic MySQL protocol connections to InnoDB Cluster:

Read/Write:

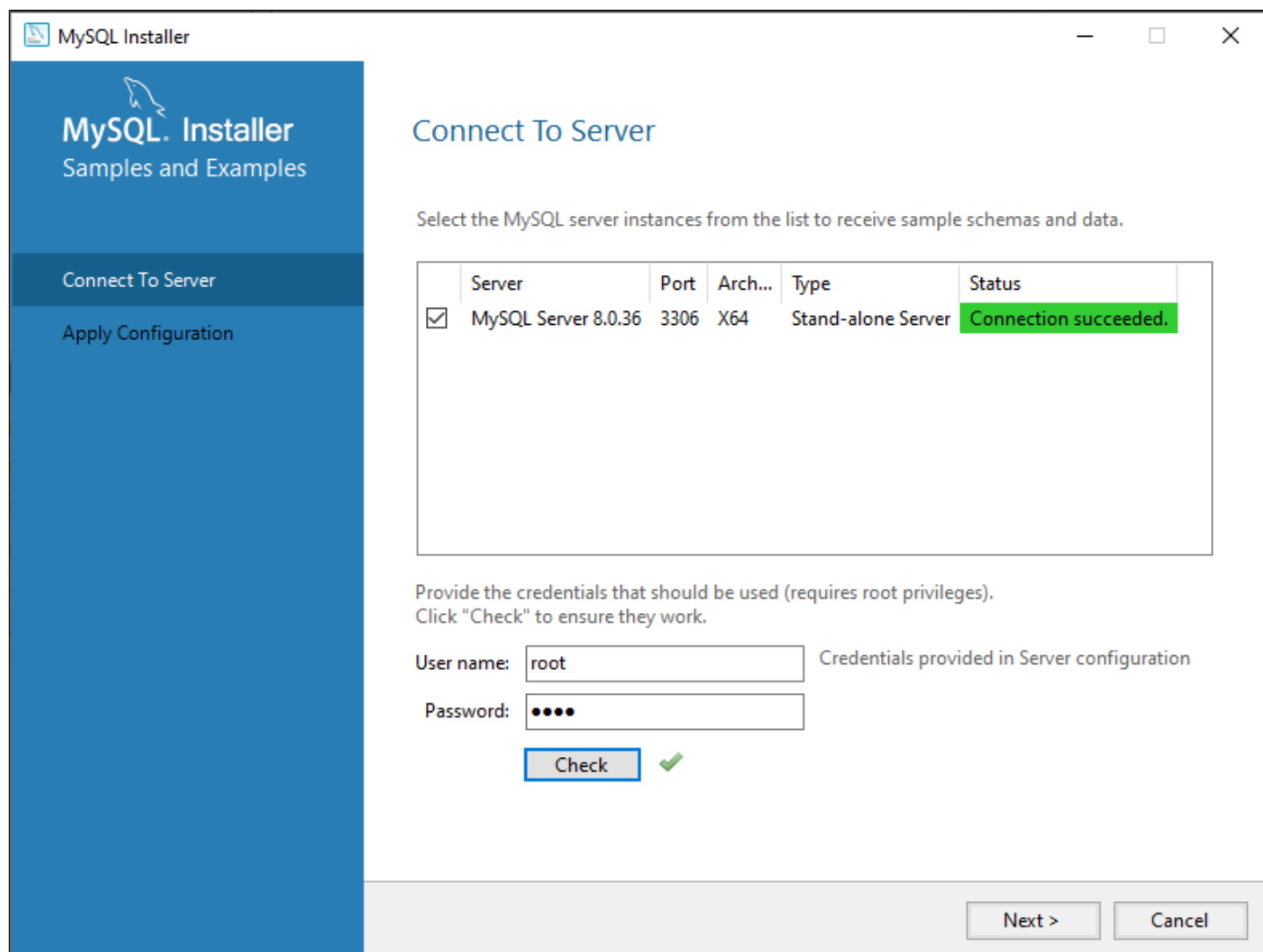
Read Only:

X Protocol connections to InnoDB Cluster:

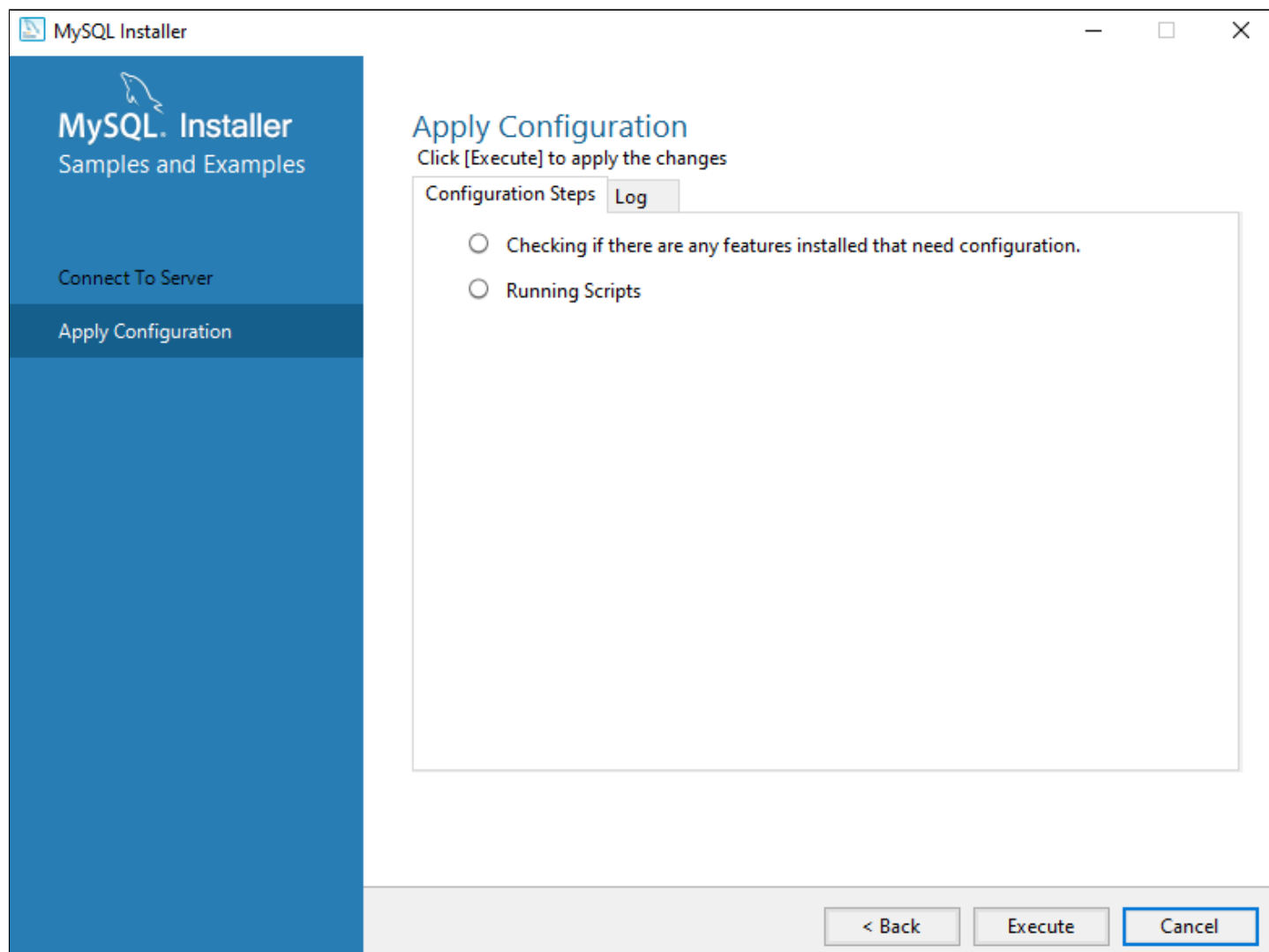
Read/Write:

Read Only:

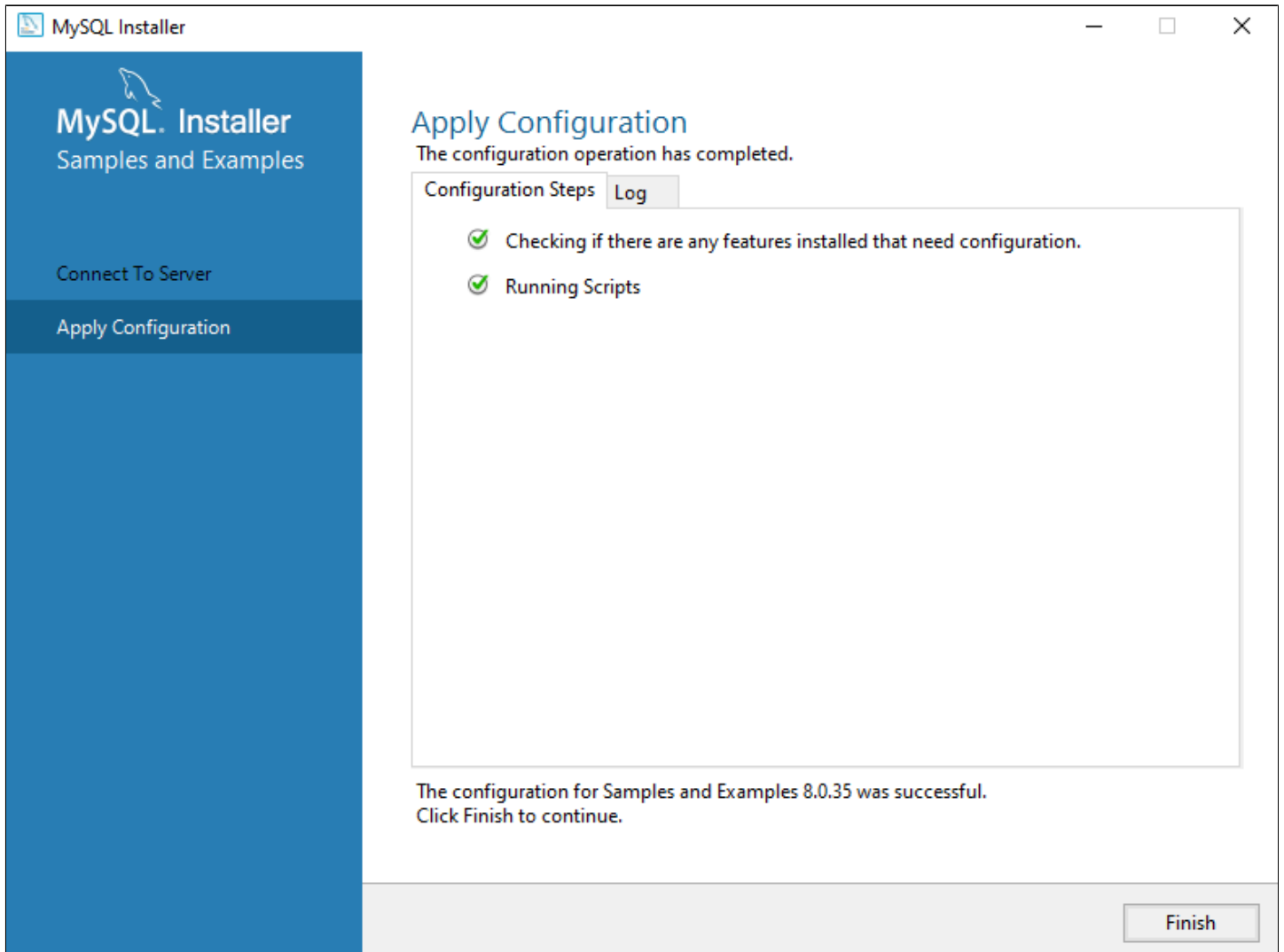
14. Проверьте подключение к серверу и нажмите кнопку "Next". Пароль был задан в пункте [8](#).



15. Нажмите на кнопку "Execute".



16. Нажмите на кнопку "Finish".

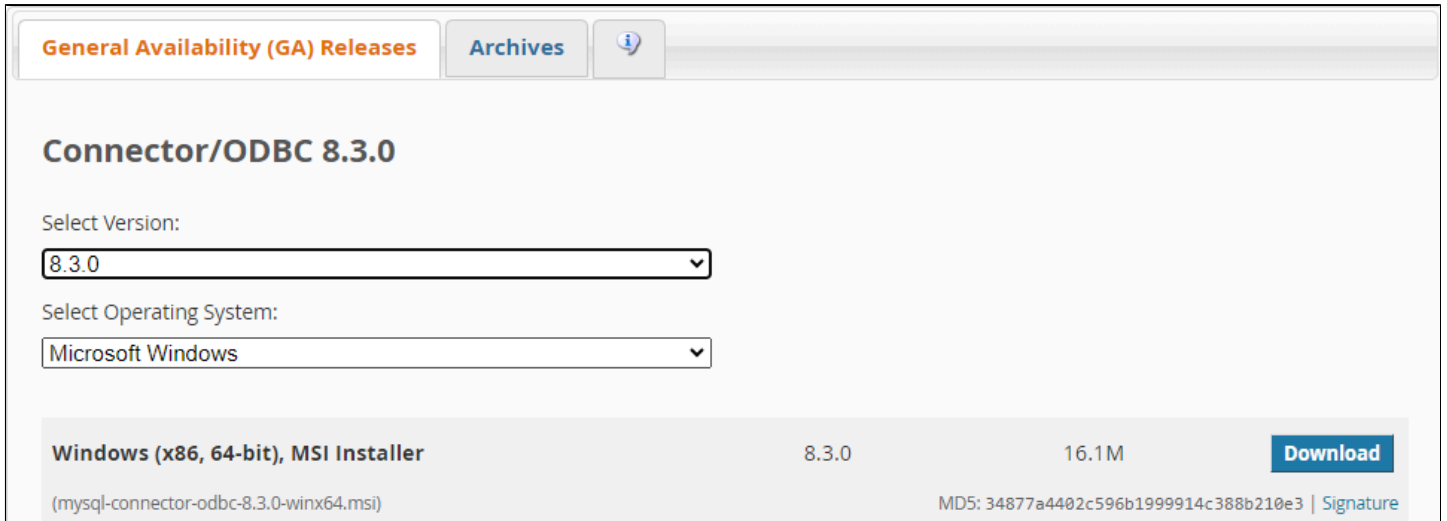


У вас могут быть дополнительные вкладки конфигурации. На них следует оставлять значения по умолчанию и нажимать "Next".

ODBC Connector

Дополнительно вам понадобится ODBC Connector.

1. Скачайте установщик по [ссылке](#).



General Availability (GA) Releases Archives ⓘ

Connector/ODBC 8.3.0

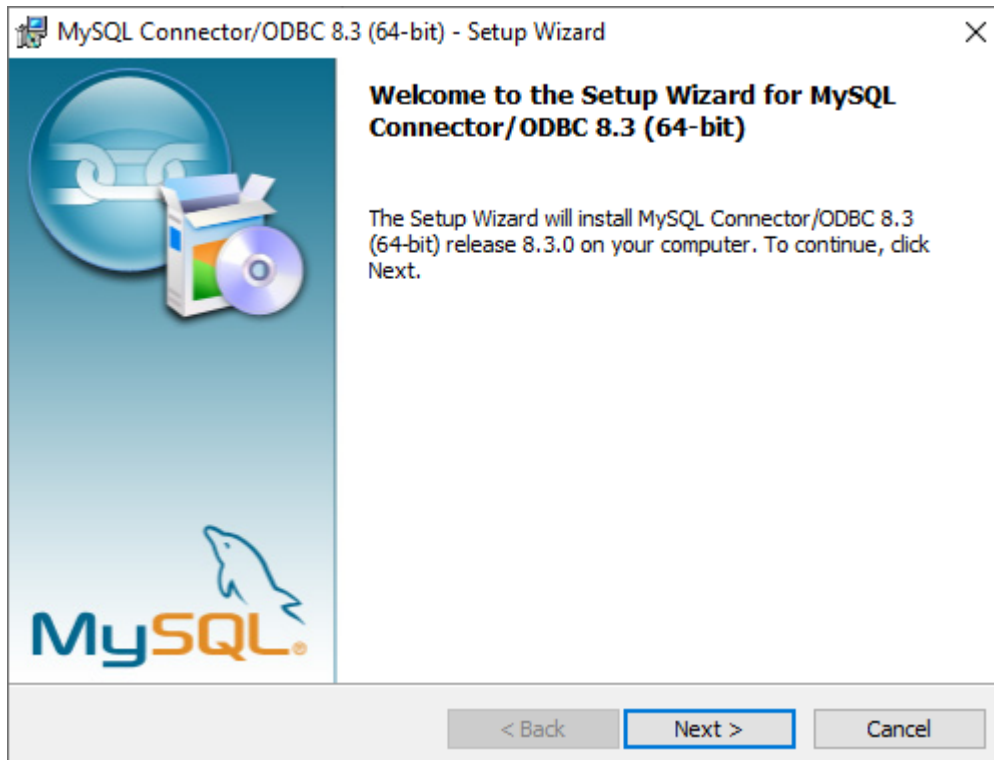
Select Version:
8.3.0

Select Operating System:
Microsoft Windows

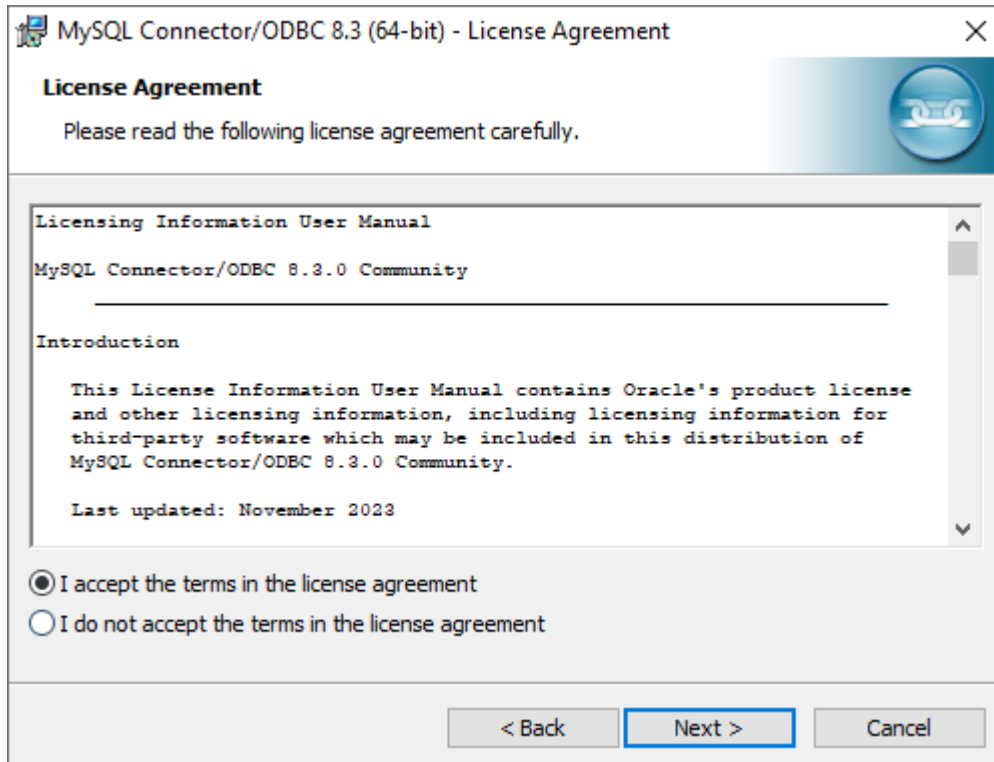
| | | | |
|--|-------|-------|--------------------------|
| Windows (x86, 64-bit), MSI Installer <small>(mysql-connector-odbc-8.3.0-winx64.msi)</small> | 8.3.0 | 16.1M | Download |
|--|-------|-------|--------------------------|

MD5: 34877a4402c596b1999914c388b210e3 | [Signature](#)

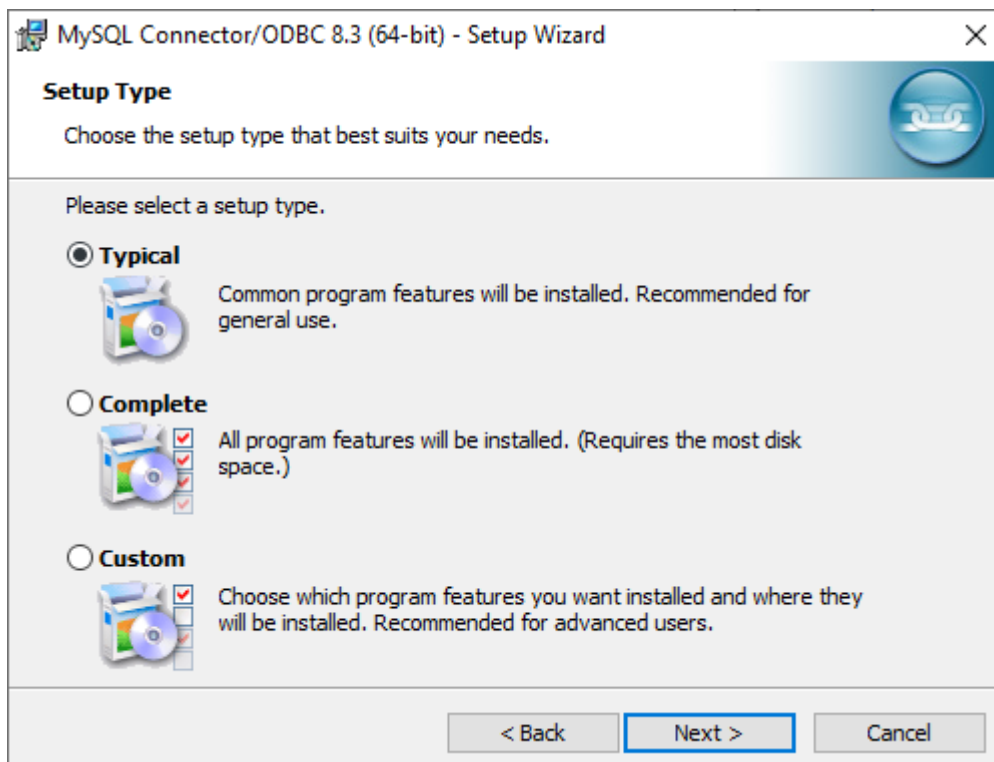
2. Запустите дистрибутив установки mysql-connector-odbc-x.x.x-winx64.msi.



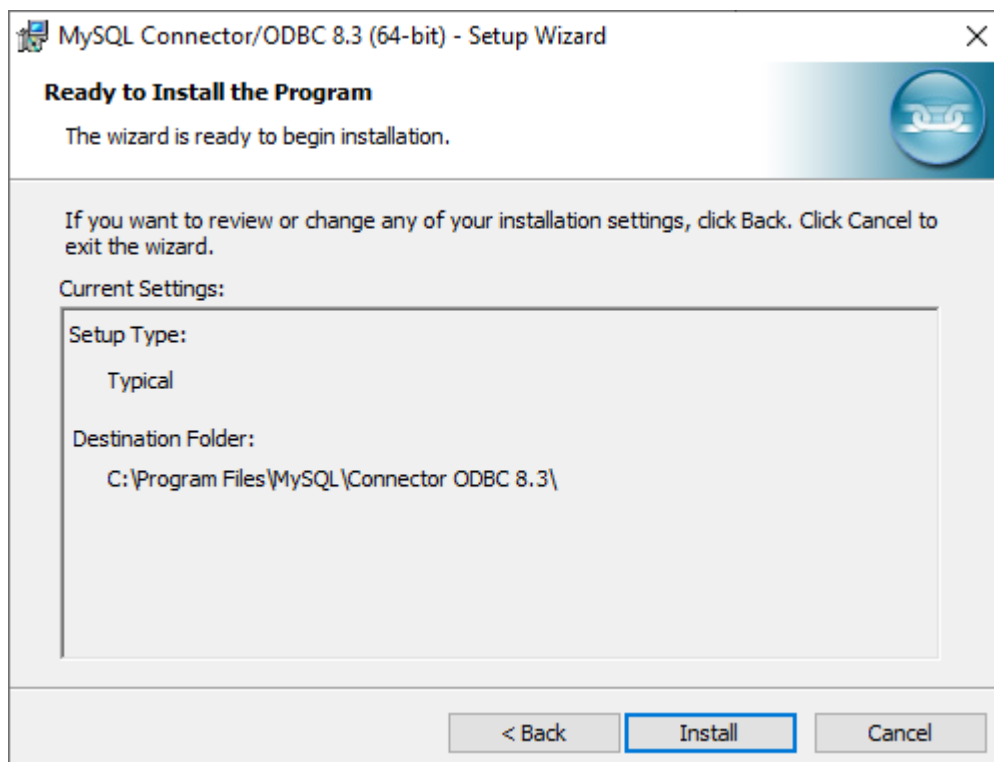
3. Внимательно прочитайте лицензию, примите ее условия и нажмите "Next".



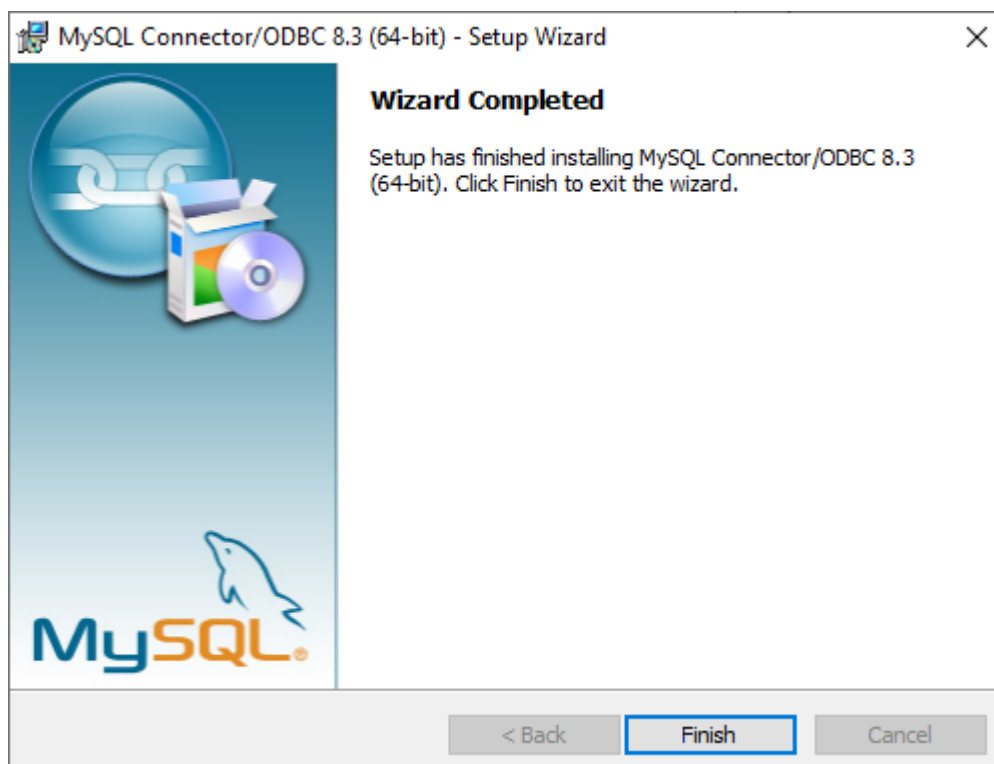
4. Выберите типичную установку.



5. Нажмите "Install".



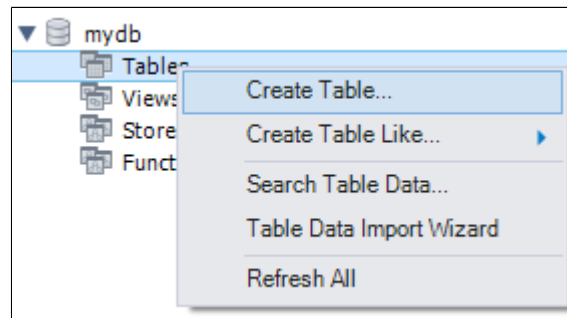
6. Нажмите "Finish".



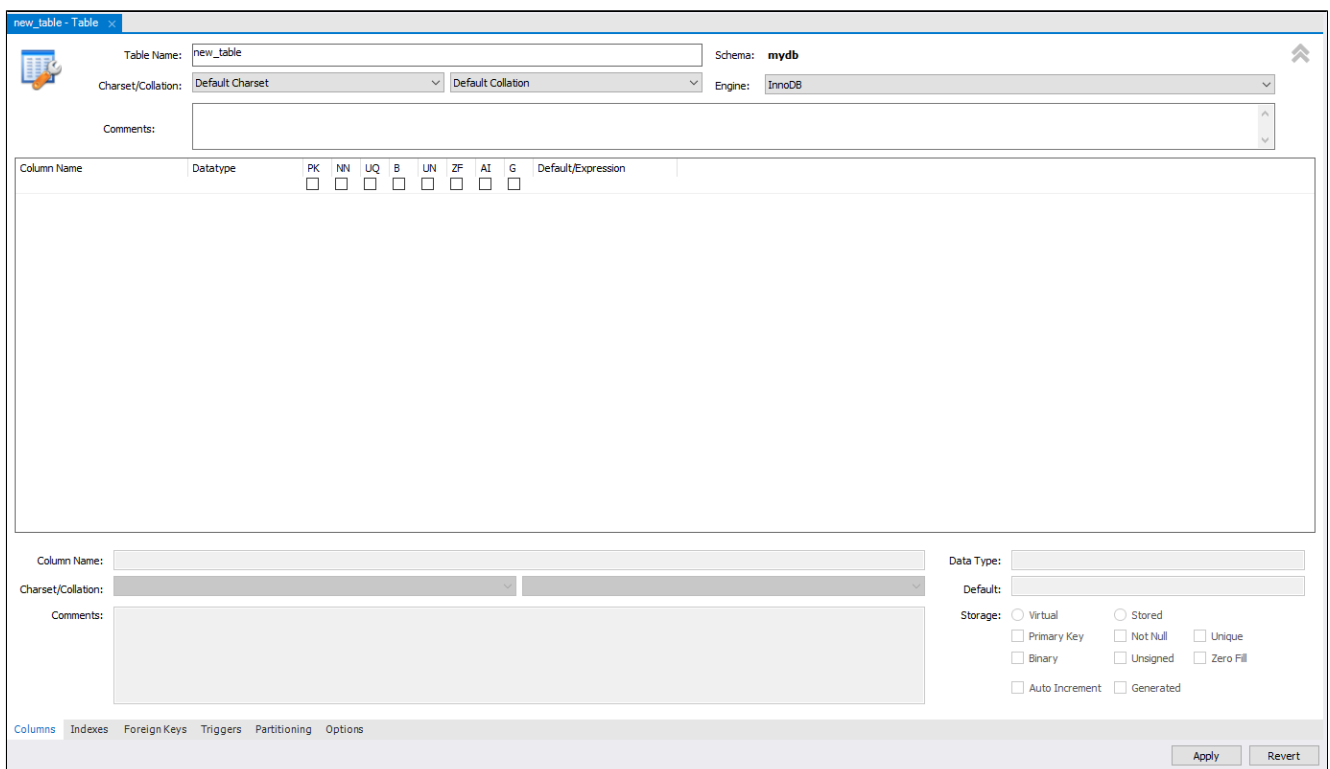
1.3.1.2. Создание таблицы

Для создания таблицы откройте MySQL Workbench. В примере таблица будет создана в базе данных mydb.

1. Нажмите правой кнопкой мыши на вкладку "Tables" в "mydb".



2. Нажмите "Create Table" и начните создание таблицы.



3. Назовите таблицу "t_sensor". Имена колонок назовите "ID", "NAME", "VALUE", "DATE", где "ID" - primary key.

Подробные настройки указаны на скриншоте.

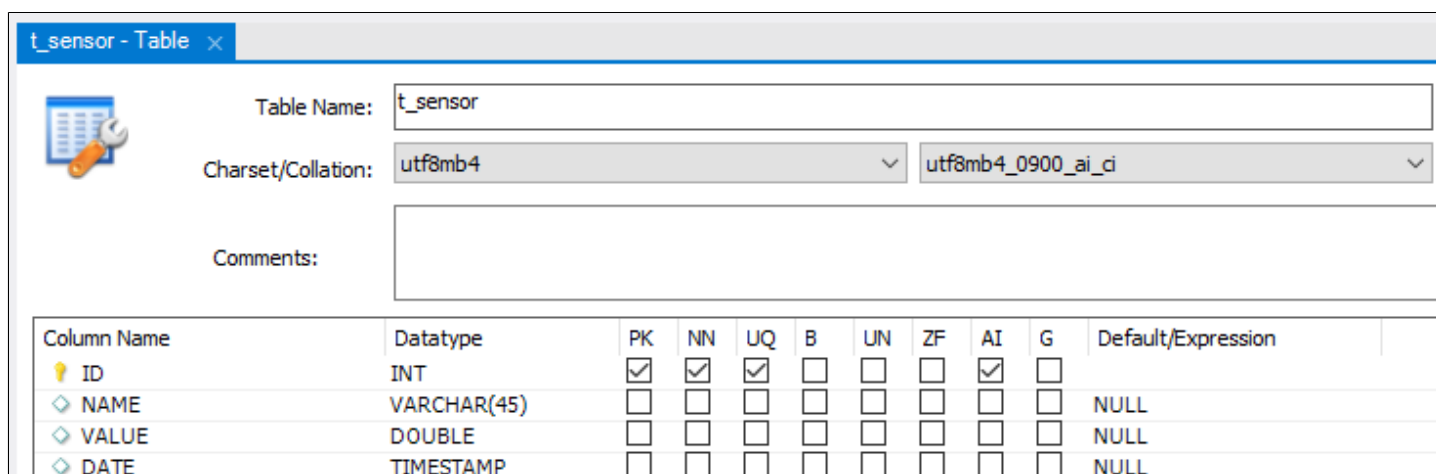


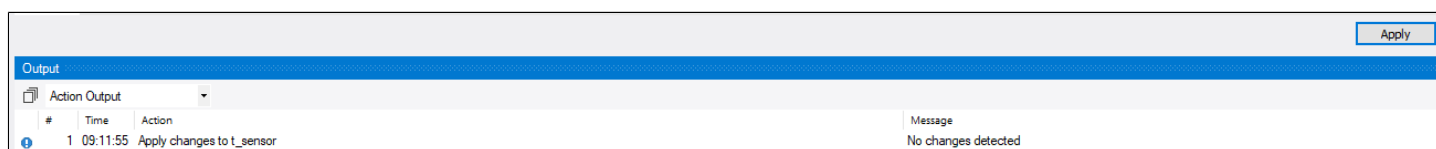
Table Name:

Charset/Collation:

Comments:

| Column Name | Datatype | PK | NN | UQ | B | UN | ZF | AI | G | Default/Expression |
|-------------|-------------|-------------------------------------|-------------------------------------|-------------------------------------|--------------------------|--------------------------|--------------------------|-------------------------------------|--------------------------|--------------------|
| ID | INT | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | |
| NAME | VARCHAR(45) | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | NULL |
| VALUE | DOUBLE | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | NULL |
| DATE | TIMESTAMP | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | NULL |

4. Сохраните таблицу нажав на кнопку Apply.



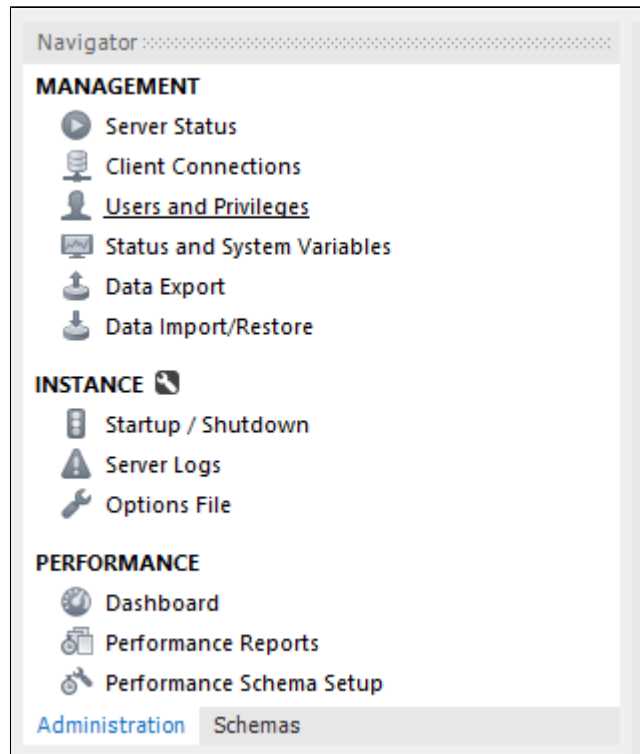
Output

Action Output

| # | Time | Action | Message |
|---|----------|---------------------------|---------------------|
| 1 | 09:11:55 | Apply changes to t_sensor | No changes detected |

1.3.1.3. Настройка пользователя

1. В навигаторе перейдите на вкладку "Administration" и выберите "User and Privileges".



2. Нажмите "Add Account".

The screenshot shows the 'Users and Privileges' page in the MySQL Administration interface. The user 'test_connect' is logged in. The 'User Accounts' table is visible on the left, and the 'Details for account newuser@%' form is on the right.

| User | From Host |
|------------------|-----------|
| mysql.infoschema | localhost |
| mysql.session | localhost |
| mysql.sys | localhost |
| newuser | % |
| root | localhost |

Details for account newuser@%

Login Account Limits Administrative Roles Schema Privileges

Login Name: You may create multiple accounts with the same name to connect from different hosts.

Authentication Type: For the standard password and/or host based authentication, select 'Standard'.

Limit to Hosts Matching: % and _ wildcards may be used

Password: Type a password to reset it.

Consider using a password with 8 or more characters with mixed case letters, numbers and punctuation marks.

Confirm Password: Enter password again to confirm.

3. Создайте пользователя "user" и введите пароль.

Details for account newuser@%

Login Account Limits Administrative Roles Schema Privileges

Login Name: You may create multiple accounts with the same name to connect from different hosts.

Authentication Type: For the standard password and/or host based authentication, select 'Standard'.

Limit to Hosts Matching: % and _ wildcards may be used

Password: Type a password to reset it.

Weak password.

Confirm Password: Enter password again to confirm.

4. Во вкладке "Administrative Roles" отметьте "DBA" и нажмите "Apply".

Details for account newuser@%

Login Account Limits Administrative Roles Schema Privileges

| Role | Description |
|--|--|
| <input checked="" type="checkbox"/> DBA | grants the rights to perform all tasks |
| <input checked="" type="checkbox"/> MaintenanceAdmin | grants rights needed to maintain server |
| <input checked="" type="checkbox"/> ProcessAdmin | rights needed to assess, monitor, and kill any user proce... |
| <input checked="" type="checkbox"/> UserAdmin | grants rights to create users logins and reset passwords |
| <input checked="" type="checkbox"/> SecurityAdmin | rights to manage logins and grant and revoke server an... |
| <input checked="" type="checkbox"/> MonitorAdmin | minimum set of rights needed to monitor server |
| <input checked="" type="checkbox"/> DBManager | grants full rights on all databases |
| <input checked="" type="checkbox"/> DBDesigner | rights to create and reverse engineer any database sche... |
| <input checked="" type="checkbox"/> ReplicationAdmin | rights needed to setup and manage replication |
| <input checked="" type="checkbox"/> BackupAdmin | minimal rights needed to backup any database |

Global Privileges

- ALTER
- ALTER ROUTINE
- CREATE
- CREATE ROUTINE
- CREATE TABLESPACE
- CREATE TEMPORARY TABLES
- CREATE USER
- CREATE VIEW
- DELETE
- DROP
- EVENT
- EXECUTE
- FILE

5. Перейдите на вкладку "Schema Privileges" и нажмите "Add Entry".

The screenshot shows the 'Details for account user@%' window with the 'Schema Privileges' tab selected. The window has a header with tabs for 'Login', 'Account Limits', 'Administrative Roles', and 'Schema Privileges'. Below the tabs is a large empty table with columns for 'Schema' and 'Privileges'. Below the table, there is a note: 'Schema and Host fields may use % and _ wildcards. The server will match specific entries before wildcarded ones.' To the right of this note are three buttons: 'Revoke All Privileges', 'Delete Entry', and 'Add Entry...'. Below the table are three sections of checkboxes: 'Object Rights' (SELECT, INSERT, UPDATE, DELETE, EXECUTE, SHOW VIEW), 'DDL Rights' (CREATE, ALTER, REFERENCES, INDEX, CREATE VIEW, CREATE ROUTINE), and 'Other Rights' (GRANT OPTION, CREATE TEMPORARY TABLES, LOCK TABLES). At the bottom right are 'Revert' and 'Apply' buttons.

6. Выберите "All Schema" и нажмите "OK".

The screenshot shows the 'New Schema Privilege Definition' dialog box. The title bar says 'New Schema Privilege Definition'. The main text says 'Select the Schema for which the user 'user' will have the privileges you want to define.' Below this is a 'Schema' section with three radio button options: 'All Schema (%)', 'Schemas matching pattern:', and 'Selected schema:'. The 'All Schema (%)' option is selected. To the right of these options are three explanatory paragraphs: 'This rule will apply to any schema name.', 'This rule will apply to schemas that match the given name or pattern. You may use _ and % as wildcards in a pattern. Escape these characters with \ in case you want their literal value.', and 'Select a specific schema name for the rule to apply to.' The 'Selected schema:' option has a text box containing 'information_schema' and a dropdown arrow. At the bottom right are 'Cancel' and 'OK' buttons.

7. Нажмите на кнопку "Select "ALL"" для предоставления всех привилегий пользователю и нажмите "Apply".

Details for account user@%

Login Account Limits Administrative Roles Schema Privileges

| Schema | Privileges |
|--------|--|
| % | ALTER, ALTER ROUTINE, CREATE, CREATE ROUTINE, CREATE TEMPORARY TABLES, CREATE VIEW, DELETE, DROP, EVENT, EXECUTE, INDEX, INSERT, LO... |

Schema and Host fields may use % and _ wildcards.
The server will match specific entries before wildcarded ones.

The user 'user'@'%' will have the following access rights to any schema:

Object Rights

- SELECT
- INSERT
- UPDATE
- DELETE
- EXECUTE
- SHOW VIEW

DDL Rights

- CREATE
- ALTER
- REFERENCES
- INDEX
- CREATE VIEW
- CREATE ROUTINE
- ALTER ROUTINE
- EVENT
- DROP
- TRIGGER

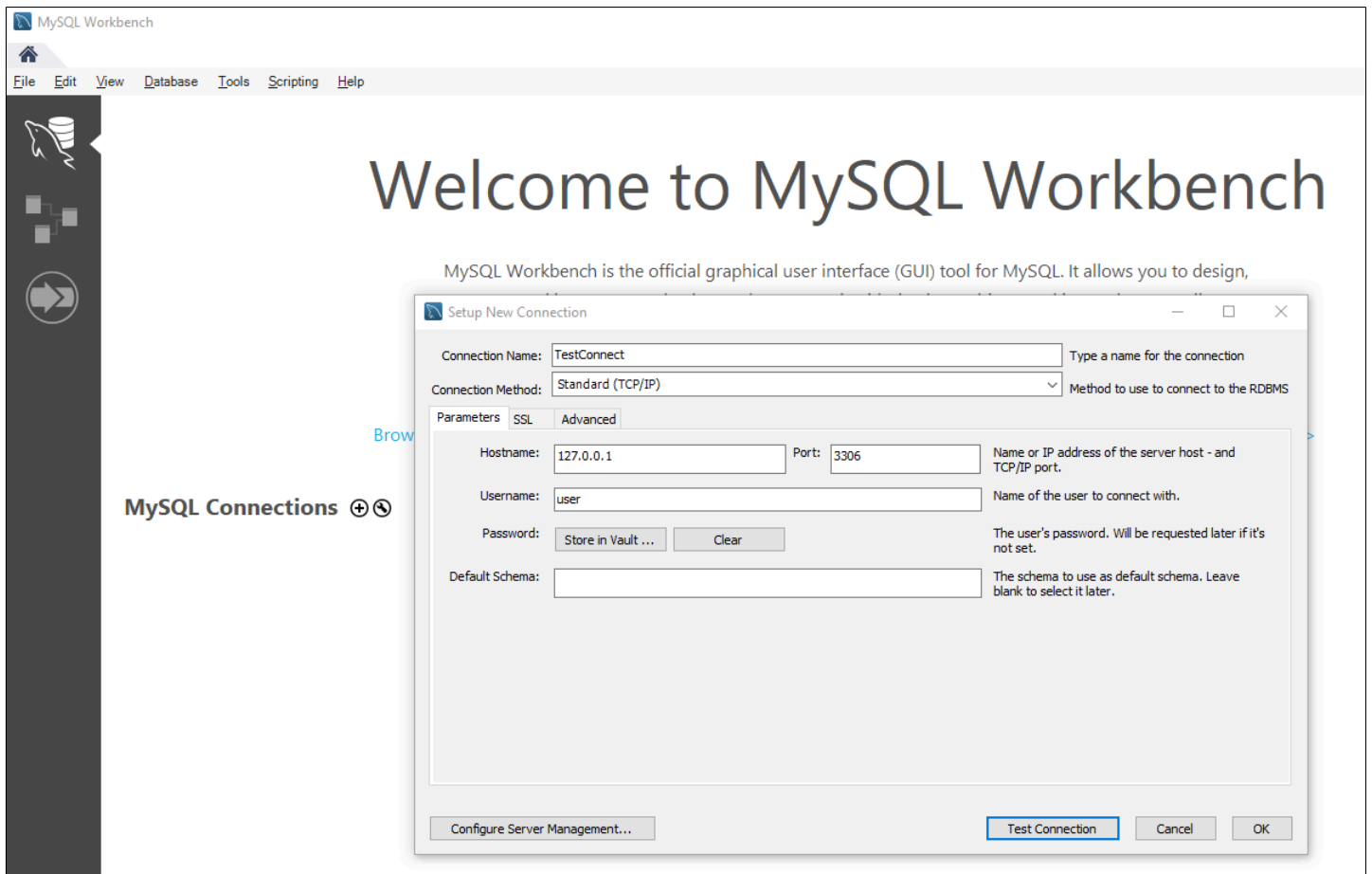
Other Rights

- GRANT OPTION
- CREATE TEMPORARY TABLES
- LOCK TABLES

Unselect All Select "ALL"

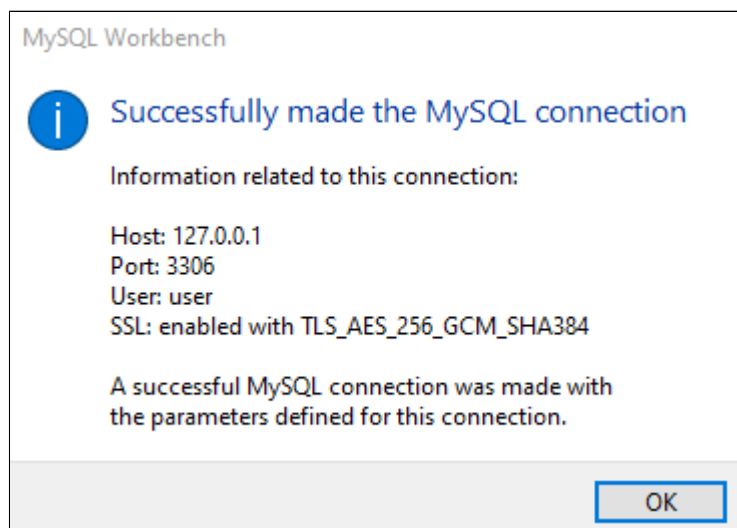
Revert Apply

8. На вкладке добавления подключений проверьте подключение пользователя "user" к БД. Нажмите на "+" в области "MySQL Connections". Откроется новое окно. Введите имя подключения и имя пользователя.



Нажмите на кнопку "Test Connection", в открывшемся окне введите пароль для пользователя "user".

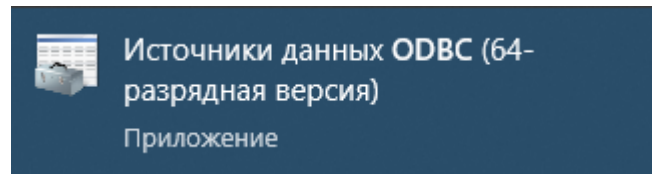
В случае успешного подключения у вас высветится следующее окно:



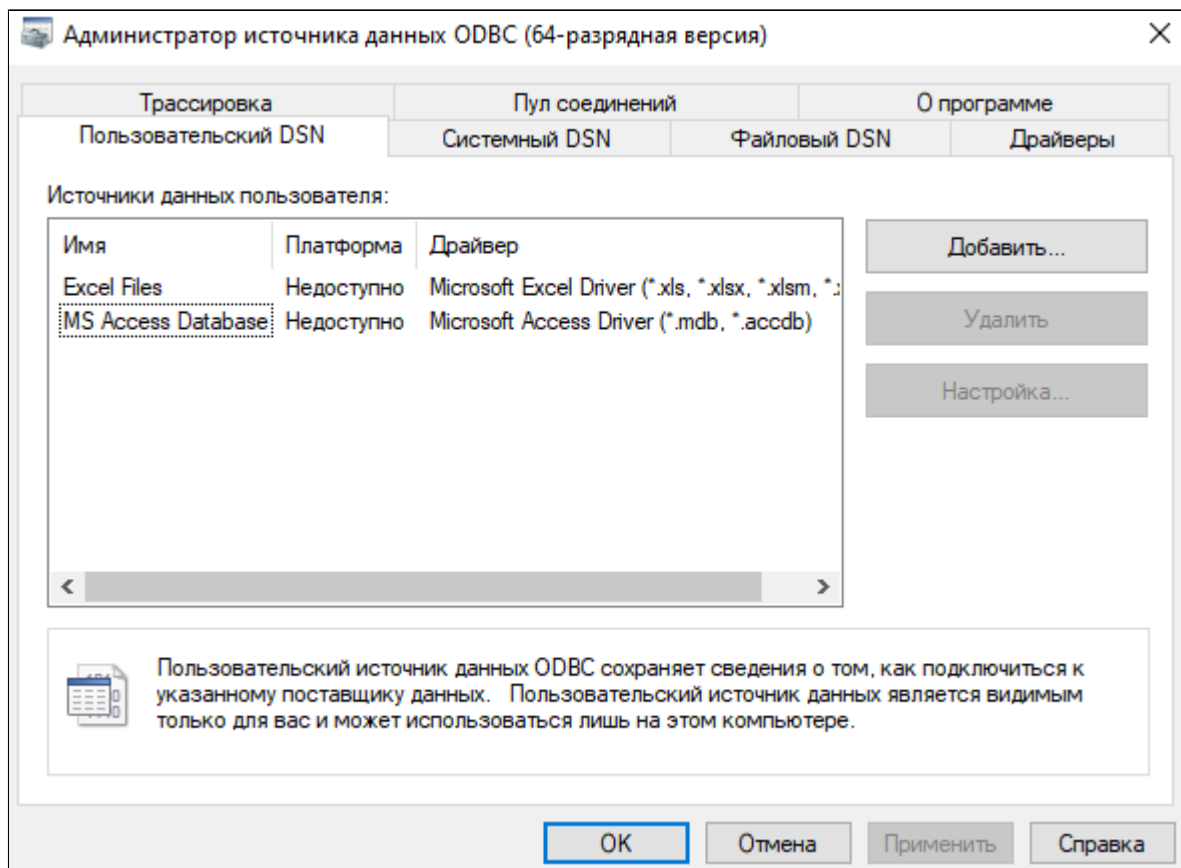
1.3.1.4. Создание источника данных

Для подключения Astra.HMI к MySQL необходимо настроить источник данных.

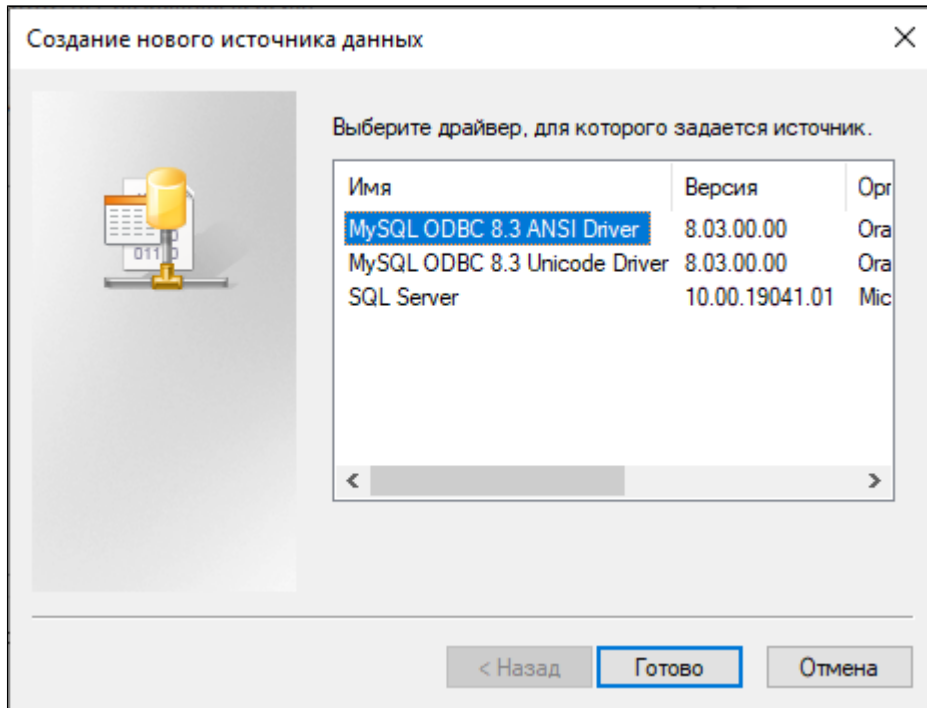
1. Откройте утилиту источники данных ODBC



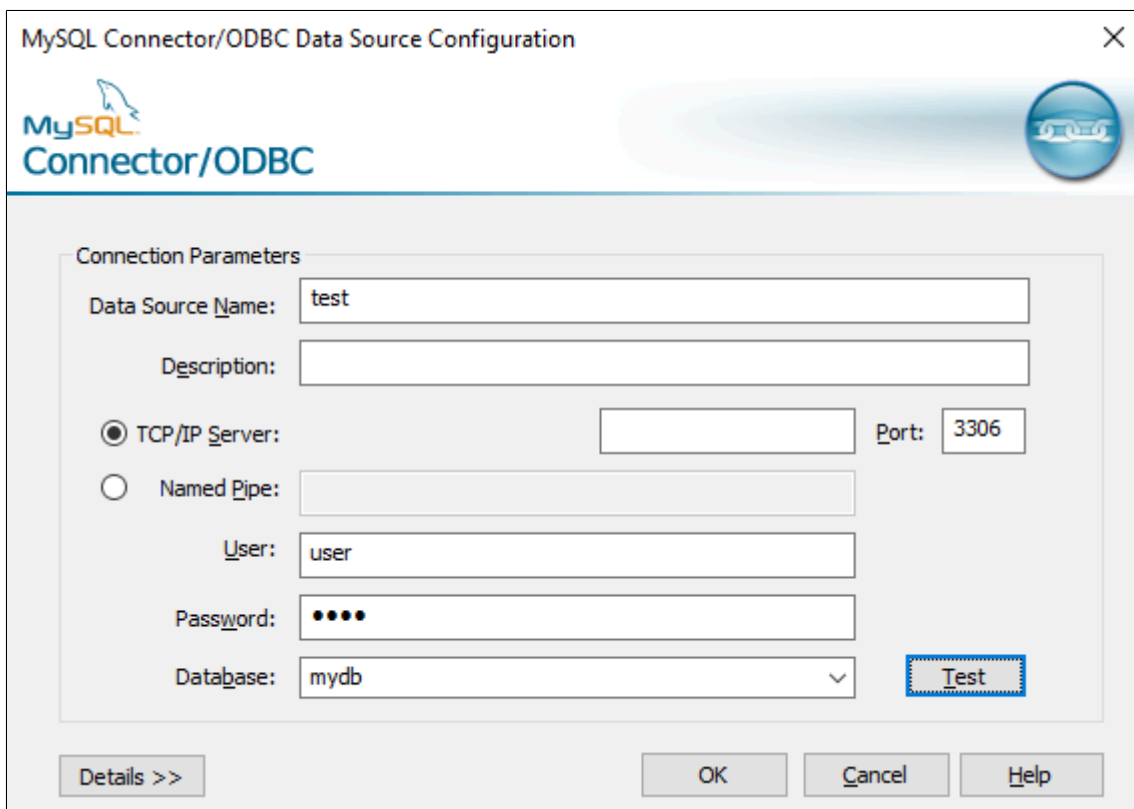
2. В открывшемся окне нажмите на кнопку "Добавить".



3. Выберите "MySQL ODBC 8.x ANSI Driver" и нажмите "Готово".

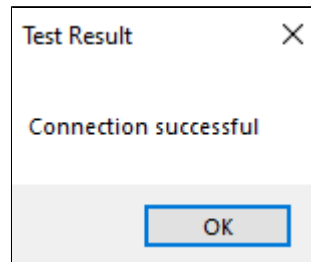


4. Настройте подключение к MySQL. Введите название подключения, имя пользователя, пароль и название базы данных.

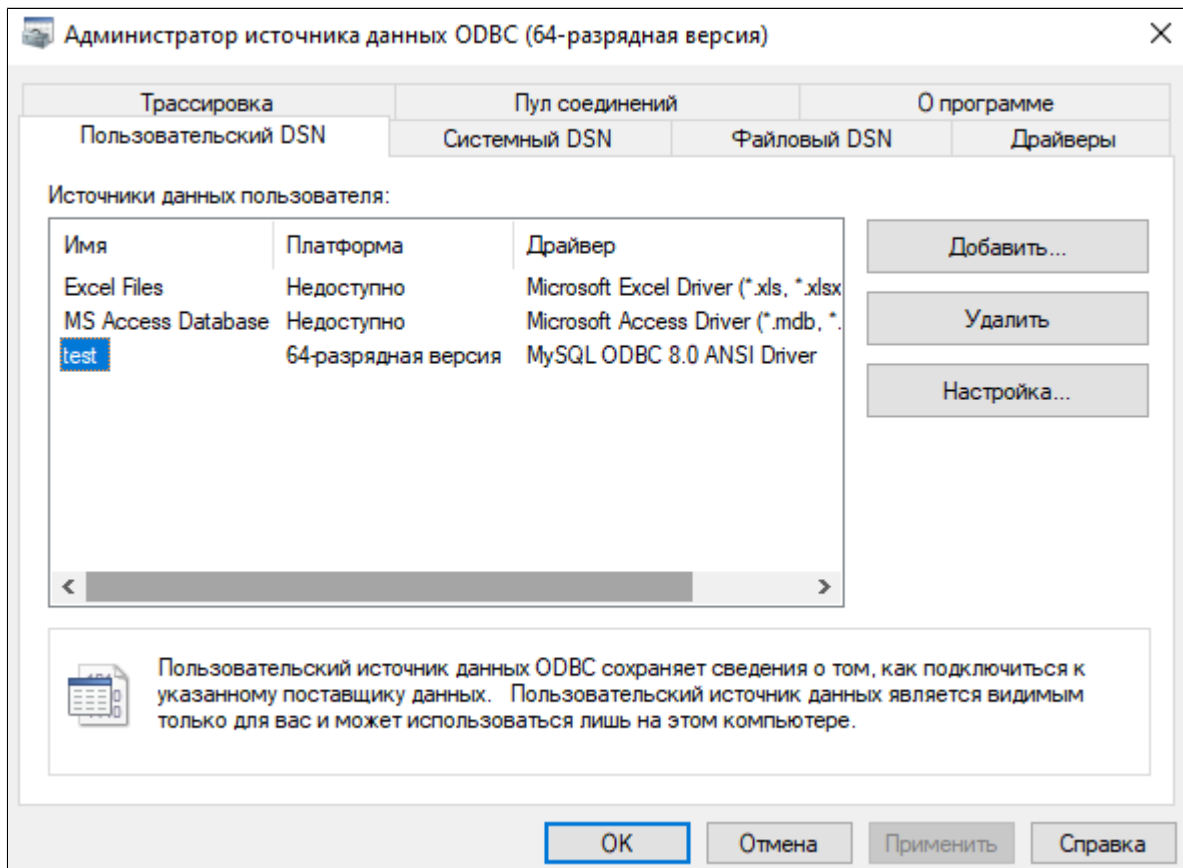


5. Проверьте подключение нажав на кнопку Test.

В случае успеха высветится появится окно:



Нажмите "OK" и сохраните подключение снова нажав на кнопку "OK" во вкладке настройки подключения.



Источник данных должен появится с списке источников данных пользователя.

1.3.2. Настройка в Astra.IDE

- › [Логика функционального блока для внешней БД](#)
- › [Использование функционального блока](#)
- › [Проверка работы программы](#)

1.3.2.1. Логика функционального блока для внешней БД

Для работы с внешней базой данных MySQL никаких дополнительных настроек контроллера не требуется.

Создайте функциональный блок, который будет подключаться к MySQL и записывать в нее значения. Для этого:

1. Создайте POU типа "Функциональный блок" на языке ST и задайте имя "DB".

Добавить POU

Создать новый POU (компонент организации программы)

Имя
DB

Тип

Программа

Функциональный блок

Extends

Implements

Окончательный Абстрактный

Спецификатор доступа

Язык реализации метода:
Структурированный текст (ST)

Функция

Тип возвращаемого значения

Язык реализации
Структурированный текст (ST)

Добавить Отмена

2. Объявите его входные и выходные значения.

```
FUNCTION_BLOCK DB
VAR_INPUT
  DB_TYPE      : DB_TYPES;           //Тип базы данных
  DB_NAME      : STRING              := 'MyDB'; //Имя базы данных
  HOST         : STRING              := 'localhost'; //IP хоста
  PORT        : UDINT                := 0; //Номер порта
  USER        : STRING              := 'user'; //Имя пользователя
  PASSWORD    : STRING              := ''; //Пароль
  CHARSET     : STRING              := 'cp1251'; //Кодировка
  ASYNC       : BOOL                 := FALSE; //Синхронные/асинхронные запросы
END_VAR
VAR_OUTPUT
  RESULT      : BOOL;                //Успешное выполнение кода
  ERROR       : MySQLError;          //Последняя ошибка базы данных
END_VAR
VAR
  MYSQL       : MySql;               //Переменная для работы с запросами
  CONN        : MySqlConnection;     //Переменная для настройки подключения к БД
  ERRTYPE     : MysqlError;          //Переменная показывающая ошибки при подключении
  I           : INT;                 //Итератор
  FIRST       : INT                  := 0; //Проверка на первый запуск
  LINES       : ARRAY [0..50] OF STRING; //Массив результатов одной строки
END_VAR
```

3. Напишите метод "INIT" который будет подключаться к внешней базе данных. В данном случае при инициализации блока будут передаваться настроечные параметры переменной "MySQLConn" и выполняться проверка подключения. При успехе будет возвращена единица, а иначе – выведена ошибка в переменную "ERROR".

```
//Метод инициализации внутренней БД или полключения к внешней БД
METHOD INIT : BOOL
VAR
END_VAR

IF DB_TYPE=DB_TYPES.INTERNAL_DB THEN
    //pass
ELSE
    conn.reconnect:= TRUE;                //Переподключение при неуспехе

    //Настройки внешней базы данных
    conn.host:= host;
    conn.port:= port;
    conn.user:= user;
    conn.password:= password;
    conn.database:= DB_NAME;
    conn.charset:= charset;
    mysql.xAsync:= async;

    IF NOT mysql.connect(conn) THEN      //Действия если БД не подключена
        ERROR:= mysql.Error;
    ELSE
        INIT:=1;
    END_IF
END_IF
```

4. Напишите метод, который будет добавлять значения в базу данных. Для этого создайте метод "INSERT", который будет принимать на вход название таблицы, массив названий столбцов и массив значений у столбцов, которые необходимо будет записать в базу данных. На выход метод будет отдавать сформированный запрос MySQL и число использованных столбцов.

```
//Метод добавления данных в БД
METHOD INSERT : BOOL
VAR_INPUT
    TABLE_NAME      : STRING                := 'DbTable'; //Название таблицы
    COLUMN_NAME      : ARRAY [1..50] OF STRING; //Массив названий столбцов
    COLUMN_VALUE     : ARRAY [1..50] OF STRING; //Массив значений у столбцов
END_VAR
VAR
    QUERY            : STRING(500); //Переменная для записи запроса
    COUNT            : INT; //Число используемых столбцов
END_VAR
```

5. Реализуйте логику этого метода.

С помощью конкатенации будем склеивать значения из массивов, чтобы получился запрос вида:



```
INSERT INTO `<название БД>.<название таблицы>` (`<имена столбцов>`, ...) VALUES (`<значения столбцов>`, ...)
```

Далее посылаем запрос в базу данных и ловим ошибки если они есть.

```
COUNT:= COUNTER(COLUMN_NAME);

QUERY:=CONCAT(Query, 'INSERT INTO ');

IF DB_TYPE=DB_TYPES.EXTERNAL_DB THEN
    QUERY:=CONCAT(QUERY, DB_NAME);
    QUERY:=CONCAT(QUERY, '`.`');
END_IF
QUERY:=CONCAT(QUERY, TABLE_NAME);
QUERY:=CONCAT(QUERY, '` (');

FOR i:=1 TO COUNT DO
    QUERY:=CONCAT(QUERY, ' ');
    QUERY:=CONCAT(QUERY, COLUMN_NAME[i]);

    IF i<COUNT THEN
        QUERY:=CONCAT(QUERY, ', ');
    ELSE
        QUERY:=CONCAT(QUERY, ')');
    END_IF
END_FOR

QUERY:=CONCAT(QUERY, ' VALUES (');

FOR i:=1 TO COUNT DO
    QUERY:=CONCAT(QUERY, ' ');
    QUERY:=CONCAT(QUERY, COLUMN_VALUE[i]);

    IF i<COUNT THEN
        QUERY:=CONCAT(QUERY, ', ');
    ELSE
        QUERY:=CONCAT(QUERY, ')');
    END_IF
END_FOR

mysql.query(QUERY, 0, 0); //Запрос в БД

IF mysql.Completed AND mysql.Error=EOK THEN
    INSERT:= TRUE;
ELSE
    ERROR:= mysql.Error;
END_IF
```

1.3.2.2. Использование функционального блока

Реализуем пример работы с внешней базой данных.

Ранее мы создали базу данных MySQL добавим в нее значения:

1. Создайте POU типа "Программа" на языке "CFC - постранично" и назовите его "TEST_DB".

Добавить POU

Создать новый POU (компонент организации программы)

Имя
TEST_DB

Тип

Программа

Функциональный блок

Extends ...

Implements ...

Окончательный Абстрактный

Спецификатор доступа
▼

Язык реализации метода:
Структурированный текст (ST) ▼

Функция

Тип возвращаемого значения ...

Язык реализации
Непрерывные функциональные схемы (CFC) - постранично ▼

Добавить Отмена

2. Объявите переменные с которыми вы будете работать в программе:

```
PROGRAM TEST_DB
VAR
  FB_EXTERNAL_DB: DB; //Внешняя БД

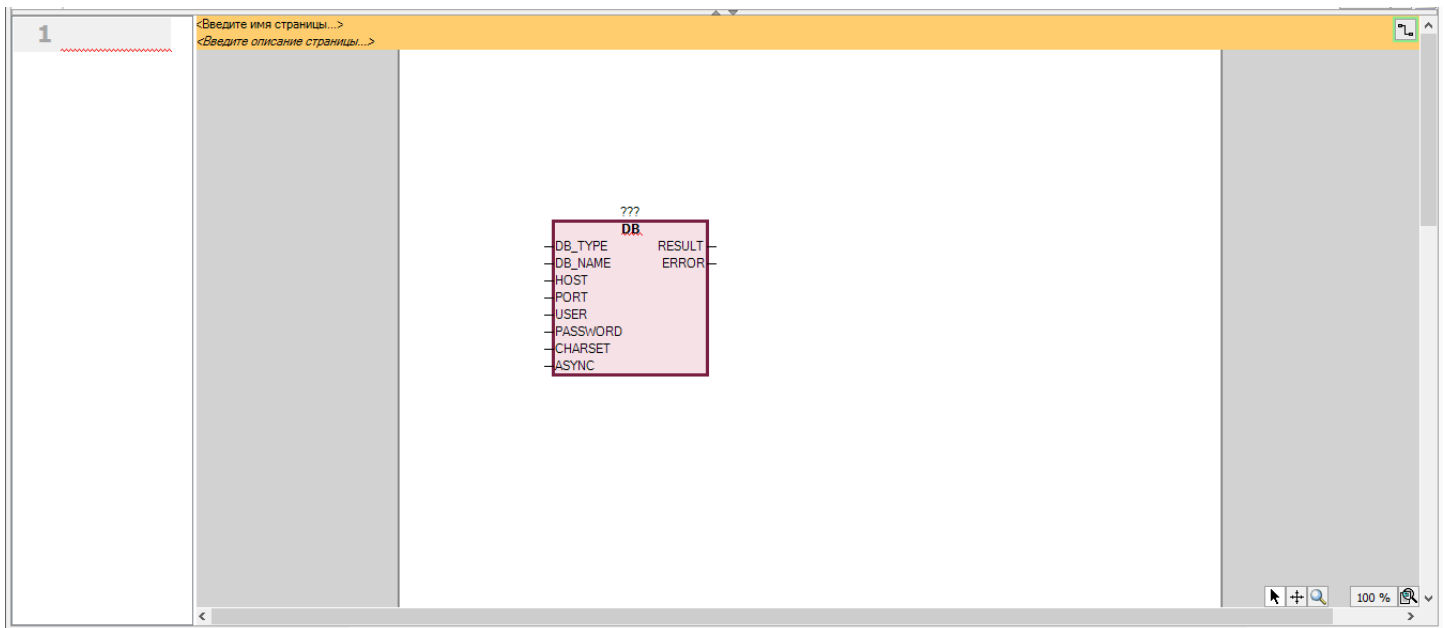
  ARRAY_OF_NAMES : ARRAY [1..50] OF STRING := ['NAME','VALUE','DATE']; //Массив названий столбцов
  ARRAY_OF_TYPES : ARRAY [1..50] OF STRING := ['varchar(45)','DOUBLE','DATETIME']; //Массив типов столбцов
  ARRAY_OF_VALUES : ARRAY [1..50] OF STRING; //Массив значений столбцов

  RESULTS : ARRAY [0..50, 0..9] OF STRING; //Массив результатов
  TABLE_NAME : STRING := 't_sensor'; //Имя таблицы

  IS_WRITE : BOOL := FALSE; //Разрешение на запись в БД
  VALUE : INT; //Переменная, которая имитирует изменение значений для записи в БД
  I : INT; //Счетчик
  RES : RIS_IEC_RESULT; //Переменная для получения времени
  DATE_ST : STRING; //Результат получения времени в виде строки
  DATE_DT : DATE_AND_TIME; //Результат получения времени в виде DT
  dwSysTimeRtsGetUTC : DWORD; //Результат получения времени UTC
  dwSysTimeRtsGetLOCAL: DWORD; //Результат получения местного времени

END VAR
VAR OUTPUT
  EXTERNAL_RESULT : BOOL; //Результат инициализации ФБ для внешней БД
  EXTERNAL_ERROR : MySQLError; //Последняя ошибка внешней БД
  RESULT : BOOL; //Результат инициализации ФБ
END VAR
```

3. Перетяните на страницу функциональный блок для работы с БД.

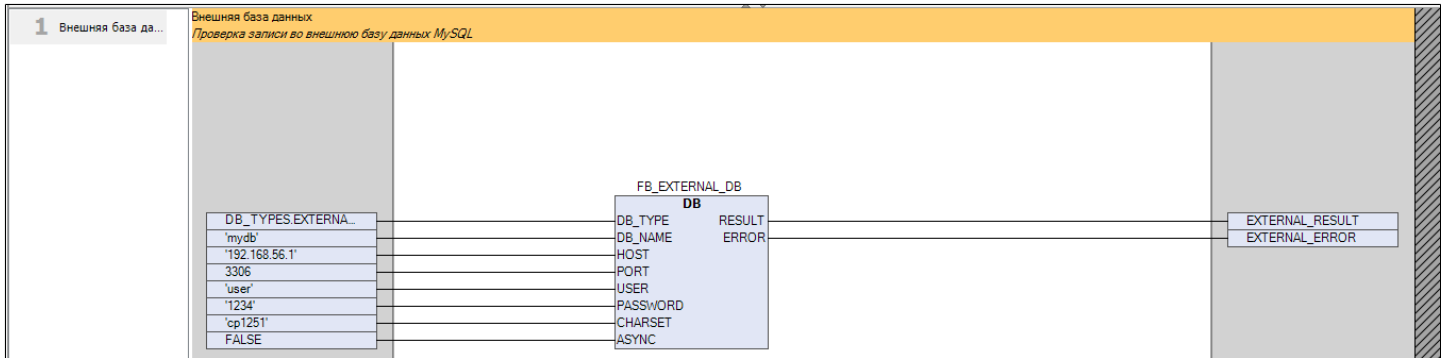


4. Дайте имя ФБ, а так же укажите значения для входов и выходов.



Необходимо указать IP машины, на которой находится сервер базы данных.

Для определения адреса введите в командную строку: "ipconfig" – для Windows, "ip a" – для Linux.



На скриншоте выше приведен пример подключения к внешней БД.

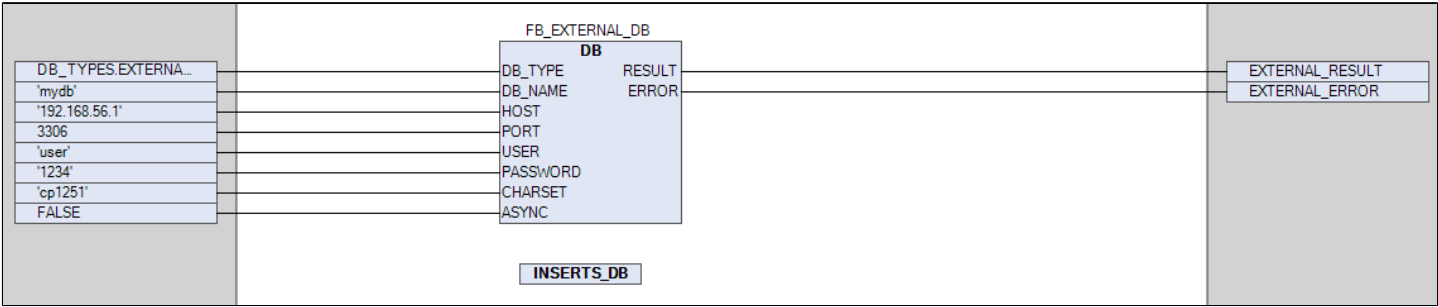
5. Добавьте действие, которое будет добавлять в БД значения.

Ниже приведена логика действия, которое добавляет в столбец "NAME" значение "Number <номер итерации>", в столбец "VALUE" – синус от номера итерации, а в столбец "DATE" – текущую дату и время.

```
//Запись во внешнюю БД данных
IF IS_WRITE= TRUE THEN
  //Заполняем массив значений данными
  ARRAY_OF_VALUES[1]:=CONCAT('Number ', INT_TO_STRING(VALUE));
  ARRAY_OF_VALUES[2]:=LREAL_TO_STRING(SIN(VALUE));
  //Получаем текущее время и тоже добавляем в массив
  dwSysTimeRtsGetUTC:= SysTimeRtsGet(res);
  SysTimeRtsConvertUtcToLocal(dwSysTimeRtsGetUTC, dwSysTimeRtsGetLOCAL);
  DATE_DT:=DWORD_TO_DT(dwSysTimeRtsGetLOCAL);
  DATE_ST:=DT_TO_STRING(DATE_DT);
  DATE_ST:=DELETE(DATE_ST, 3, 1);
  ARRAY_OF_VALUES[3]:=DATE_ST;
  I:=I+1;
  //Каждые 30 повторений добавляем во внешнюю и внутреннюю БД значения и обновляем SELECT
  IF I=30 THEN
    FB_EXTERNAL_DB.INSERT(TABLE_NAME, ARRAY_OF_NAMES, ARRAY_OF_VALUES);

    VALUE:= VALUE+1;
    I:=0;
  END_IF
END_IF
```

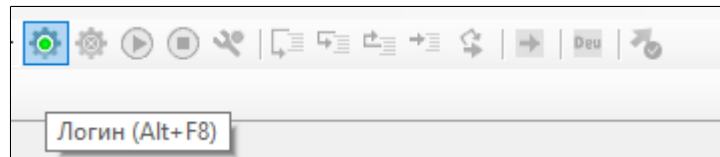
6. Чтобы действие вызывалось в программе, перетяните действие в основную программу.



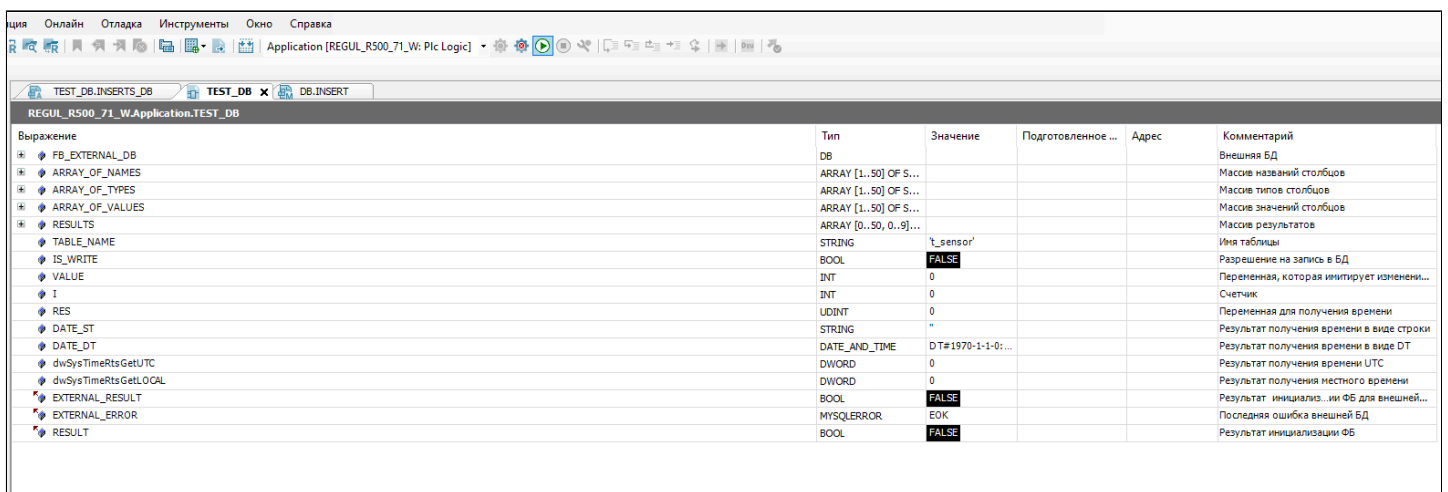
1.3.2.3. Проверка работы программы

Проверьте работу программы. Для этого сделайте следующие действия:

1. Нажмите на кнопку "Логин" и подключитесь к контроллеру.



2. Перейдите на вкладку с программой "TEST_DB" и запустите контроллер.



| Выражение | Тип | Значение | Подготовленное ... | Адрес | Комментарий |
|----------------------|------------------------|-------------------|--------------------|-------|--|
| FB_EXTERNAL_DB | DB | | | | Внешняя БД |
| ARRAY_OF_NAMES | ARRAY [1..50] OF S... | | | | Массив названий столбцов |
| ARRAY_OF_TYPES | ARRAY [1..50] OF S... | | | | Массив типов столбцов |
| ARRAY_OF_VALUES | ARRAY [1..50] OF S... | | | | Массив значений столбцов |
| RESULTS | ARRAY [0..50, 0..9]... | | | | Массив результатов |
| TABLE_NAME | STRING | 't_sensor' | | | Имя таблицы |
| IS_WRITE | BOOL | FALSE | | | Разрешение на запись в БД |
| VALUE | INT | 0 | | | Переменная, которая инициализирует изменени... |
| I | INT | 0 | | | Счетчик |
| RES | UDINT | 0 | | | Переменная для получения времени |
| DATE_ST | STRING | - | | | Результат получения времени в виде строки |
| DATE_DT | DATE_AND_TIME | DT#1970-1-1-0:... | | | Результат получения времени в виде DT |
| dwSysTimeRtsGetUTC | DWORD | 0 | | | Результат получения времени UTC |
| dwSysTimeRtsGetLOCAL | DWORD | 0 | | | Результат получения местного времени |
| EXTERNAL_RESULT | BOOL | FALSE | | | Результат инициализ...ии ФБ для внешней... |
| EXTERNAL_ERROR | MYSQLError | EOK | | | Последняя ошибка внешней БД |
| RESULT | BOOL | FALSE | | | Результат инициализации ФБ |

3. После запуска при отсутствии ошибок подключения на выходе "ERROR" и у переменной "EXTERNAL_ERROR" будет значение "EOK".





4. Найдите переменную "IS_WRITE" и запишите в нее значение "true".



| | | |
|------------|--------|------------|
| TABLE_NAME | STRING | 't_sensor' |
| IS_WRITE | BOOL | TRUE |

5. Перейдите к таблице в MySQL и убедитесь, что данные записываются.

```
1 • SELECT * FROM mydb.t_sensor;
```

Result Grid | Filter Rows: | Edit:  

| | ID | NAME | VALUE | DATE |
|---|----|-----------|----------------------|---------------------|
| ▶ | 35 | Number 0 | 0 | 2023-10-23 10:07:34 |
| | 36 | Number 1 | 0.841470984807897 | 2023-10-23 10:07:34 |
| | 37 | Number 2 | 0.909297426825682 | 2023-10-23 10:07:35 |
| | 38 | Number 3 | 0.141120008059867 | 2023-10-23 10:07:36 |
| | 39 | Number 4 | -0.756802495307928 | 2023-10-23 10:07:37 |
| | 40 | Number 5 | -0.958924274663138 | 2023-10-23 10:07:38 |
| | 41 | Number 6 | -0.279415498198926 | 2023-10-23 10:07:39 |
| | 42 | Number 7 | 0.656986598718789 | 2023-10-23 10:07:40 |
| | 43 | Number 8 | 0.989358246623382 | 2023-10-23 10:07:41 |
| | 44 | Number 9 | 0.412118485241757 | 2023-10-23 10:07:42 |
| | 45 | Number 10 | -0.54402111088937 | 2023-10-23 10:07:43 |
| | 46 | Number 11 | -0.999990206550704 | 2023-10-23 10:07:44 |
| | 47 | Number 12 | -0.536572918000435 | 2023-10-23 10:07:45 |
| | 48 | Number 13 | 0.420167036826641 | 2023-10-23 10:07:46 |
| | 49 | Number 14 | 0.99060735569487 | 2023-10-23 10:07:47 |
| | 50 | Number 15 | 0.650287840157117 | 2023-10-23 10:07:47 |
| | 51 | Number 16 | -0.287903316665065 | 2023-10-23 10:07:48 |
| | 52 | Number 17 | -0.961397491879557 | 2023-10-23 10:07:49 |
| | 53 | Number 18 | -0.750987246771676 | 2023-10-23 10:07:50 |
| | 54 | Number 19 | 0.149877209662952 | 2023-10-23 10:07:51 |
| | 55 | Number 20 | 0.912945250727628 | 2023-10-23 10:07:52 |
| | 56 | Number 21 | 0.836655638536056 | 2023-10-23 10:07:53 |
| | 57 | Number 22 | -0.00885130929040... | 2023-10-23 10:07:54 |
| | 58 | Number 23 | -0.846220404175171 | 2023-10-23 10:07:55 |
| | 59 | Number 24 | -0.905578362006624 | 2023-10-23 10:07:56 |
| | 60 | Number 25 | -0.132351750097773 | 2023-10-23 10:07:57 |
| | 61 | Number 26 | 0.762558450479603 | 2023-10-23 10:07:58 |

1.3.3. Настройка в Astra.HMI

- › [Настройка переменных окружения для взаимодействия с SQL базой данных на Linux](#)
- › [Создание проекта Astra.HMI](#)
- › [Проверка работы](#)

1.3.3.1. Настройка переменных окружения для взаимодействия с SQL базой данных на Linux

При взаимодействии с SQL базой данных (PostgreSQL, MySQL, Microsoft SQL Server и другие) через Astra.HMI на операционных системах семейства Linux необходимо предварительно настроить переменные окружения "ODBCSYSINI" и "LD_LIBRARY_PATH". Эти переменные окружения необходимы для того, чтобы система знала, где искать конфигурационные файлы для корректной работы с SQL базой данных через ODBC (интерфейс между базой данных и приложением, взаимодействующим с ней).

Самый распространенный способ настройки переменных окружения на Linux – это добавить переменные в системный файл ".bashrc".

Для добавления переменных окружения в файл ".bashrc" откройте терминал и выполните следующие действия:

1. Выполните команду для открытия файла ".bashrc" для редактирования.

```
nano ~/.bashrc
```

2. Добавьте следующие строки в конец файла:

```
export ODBCSYSINI="/etc"
```

```
export LD_LIBRARY_PATH="/usr/lib/x86_64-linux-gnu/odbc"
```

```
export ODBCSYSINI="/etc"  
export LD_LIBRARY_PATH="/usr/lib/x86_64-linux-gnu/odbc"
```

4. Сохраните файл, нажав сочетание клавиш "Ctrl + X", затем "Y", затем клавишу "Enter".

Обновите текущий сеанс оболочки, чтобы изменения вступили в силу:

```
▶ source ~/.bashrc
```

```
astraregul@astraregul:~$ source ~/.bashrc
```

5. Для запуска проекта Astra.HMI выполните команду:

```
▶ astra.hmi.viewer <путь_до_файла>
```

Если необходимо открыть дизайнер Astra.HMI для редактирования/создания проекта и его дальнейшего запуска, выполните следующую команду:

```
▶ astra.hmi.designer
```



Для дальнейшей работы с SQL базами данных на ОС Linux необходимо запускать проект/дизайнер Astra.HMI только через терминал.

1.3.3.2. Создание проекта Astra.HMI

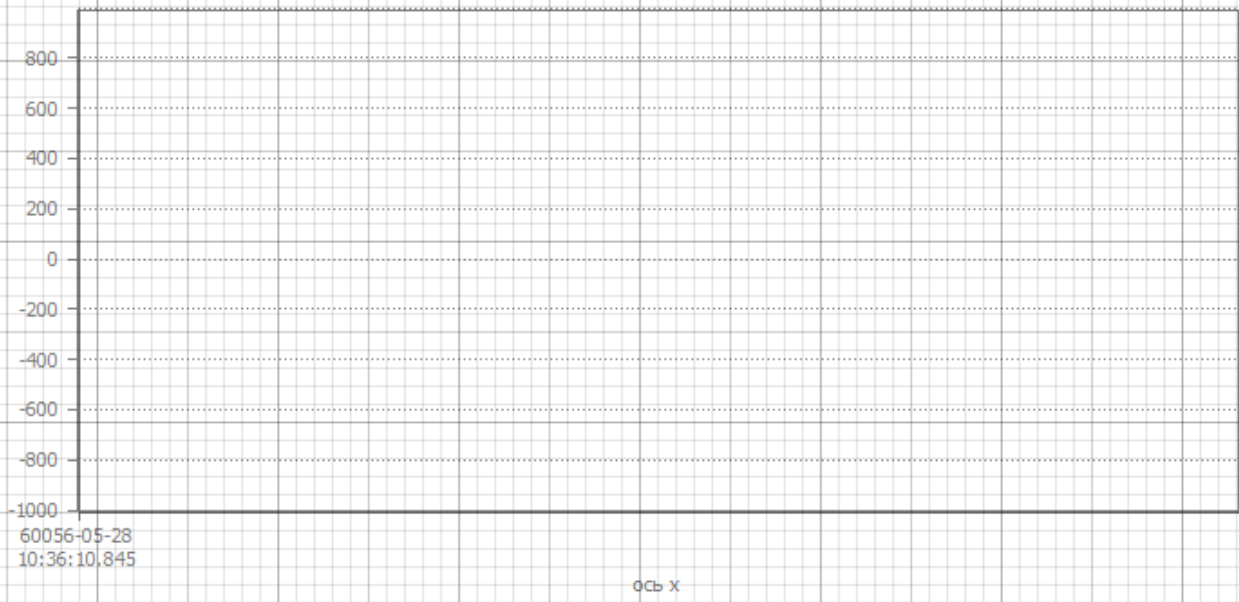


Для корректного отображения графиков с данными из MySQL необходимо, чтобы версия Astra.HMI.Charts была не ниже 2.0.3.

1. Создайте следующую структуру объектов в форме "MainForm":

| Структура объекта | |
|-----------------------|------------------------|
| 🔍 | |
| Имя | Описание |
| ▼ MainForm | Тип на основе Форма |
| ▼ Графические объекты | |
| ▼ Chart_1 | График: поле графика |
| ▼ Данные | |
| Axis_Y | График: ось |
| Axis_X | График: ось |
| ▼ DataSource_1 | График: данные |
| ▼ Данные | |
| ▼ TableModel_1 | Таблица: модель данных |
| ▼ Данные | |
| SqlQuery_1 | Запрос SQL |
| Line_1 | График: линия |
| Debug | Средство отладки |
| Button_1 | Системная кнопка |
| ▼ Функции | |
| Refresh_Table | Функция |

На мнемосхеме это должно выглядеть следующим образом:



Построить график

2. В объектах "График: ось" укажите их свойства: ориентация оси, формат меток, название и коэффициент масштабирования.

Axis_Y

График: ось

Axis_X

График: ось

Редактор свойств ☰

🔍

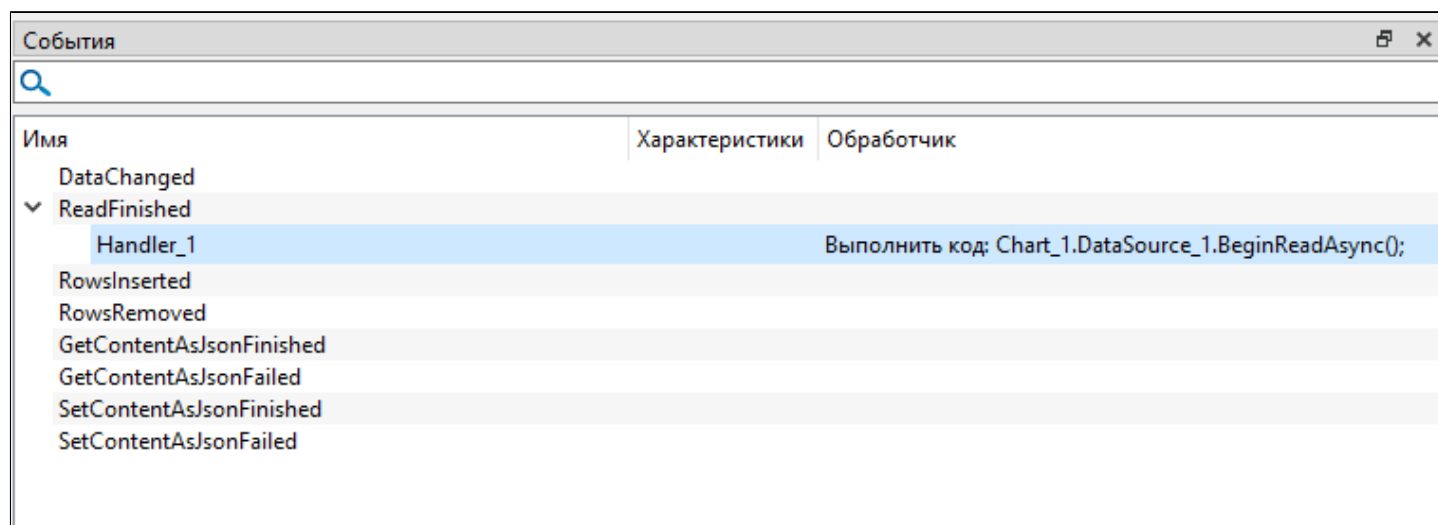
| Свойство | Харак | Значение |
|---|-------|----------------------------|
| i8 Кардинальное число | | 1 |
| > u4 Цвет | R W | 4286611584 |
| > u4 Ориентация оси | R W | 2 |
| > u4 Выравнивание | R W | 1 |
| > u4 Тип данных | R W | 0 |
| > u4 Количество делений | R W | 10 |
| > B Показывать граничные значения | R W | true |
| > u4 Тип масштабирования | R W | 2 |
| > B Логарифмическая шкала | R W | false |
| > S Единицы измерения | R W | |
| > u4 Положение единиц измерения | R W | 1 |
| > S Формат меток | R W | %.4g |
| > S Формат метки времени | R W | hh:mm:ss.zzz yyyy-MM-dd |
| > v8R Минимум | R W | -1000 |
| > v8R Максимум | R W | 1000 |
| > S Название оси | R W | |
| > B Показывать единицы измерения | R W | true |
| > u4 Положение названия оси | R W | 0 |
| > B Видимость оси | R W | true |
| > S Шрифт | R W | |
| > u4 Коэффициент масштабирования, % | R W | 5 |
| > B Автоматическое назначение ширины оси | R W | false |
| > u4 Ширина оси | R W | 65 |

| | | |
|---|--------|----------------------------|
| | Axis_Y | График: ось |
| | Axis_X | График: ось |
| Редактор свойств | | |
| <input type="text"/> | | |
| Свойство | Харак | Значение |
| <input type="checkbox"/> Кардинальное число | | 1 |
| > <input type="checkbox"/> Цвет | R W | 4286611584 |
| > <input type="checkbox"/> Ориентация оси | R W | 1 |
| > <input type="checkbox"/> Выравнивание | R W | 1 |
| > <input type="checkbox"/> Тип данных | R W | 1 |
| > <input type="checkbox"/> Количество делений | R W | 10 |
| > <input type="checkbox"/> Показывать граничные значения | R W | true |
| > <input type="checkbox"/> Тип масштабирования | R W | 2 |
| > <input type="checkbox"/> Логарифмическая шкала | R W | false |
| > <input type="checkbox"/> Единицы измерения | R W | |
| > <input type="checkbox"/> Положение единиц измерения | R W | 1 |
| > <input type="checkbox"/> Формат меток | R W | %f |
| > <input type="checkbox"/> Формат метки времени | R W | yyyy-MM-dd hh:mm:ss.zzz |
| > <input type="checkbox"/> Минимум | R W | 18446744073708050000 |
| > <input type="checkbox"/> Максимум | R W | 1500000 |
| > <input type="checkbox"/> Название оси | R W | ось x |
| > <input type="checkbox"/> Показывать единицы измерения | R W | true |
| > <input type="checkbox"/> Положение названия оси | R W | 1 |
| > <input type="checkbox"/> Видимость оси | R W | true |
| > <input type="checkbox"/> Шрифт | R W | |
| > <input type="checkbox"/> Коэффициент масштабирования, % | R W | 5 |
| > <input type="checkbox"/> Автоматическое назначение ширины оси | R W | false |
| > <input type="checkbox"/> Ширина оси | R W | 65 |

3. Привяжите оси и модель данных к объекту "График: линия", а также номер колонки для значений в таблице:

| Line_1 | | График: линия |
|-----------------------------|-------|---------------------------|
| Debug | | Средство отладки |
| Редактор свойств | | |
| <input type="text"/> | | |
| Свойство | Харак | Значение |
| Отображаемое имя | | Line_1 |
| Кардинальное число | | 1 |
| AxisX | R ↗ → | here.Chart_1.Axis_X |
| AxisY | R ↗ → | here.Chart_1.Axis_Y |
| DataSource | R ↗ → | here.Chart_1.DataSource_1 |
| > Цвет | R W | 0xff0000ff |
| > Название | R W | |
| > Единица измерения | R W | |
| > Толщина линии | R W | 3 |
| > Размер точки | R W | 15 |
| > Стиль линии | R W | Сплошная заливка |
| > Прозрачность | R W | 1 |
| > Номер символа точки | R W | 0 |
| > Видимость точек | R W | true |
| > Шрифт символов точек | R W | |
| > Видимость | R W | true |
| > Приоритет | R W | 100 |
| > Тип графика | R W | 0 |
| > Номер колонки значений X | R W | 3 |
| > Номер колонки значений Y | R W | 2 |

4. В объекте "Таблица: модель данных" в событии "ReadFinished" выполните функцию асинхронного чтения данных "BeginReadAsync()";

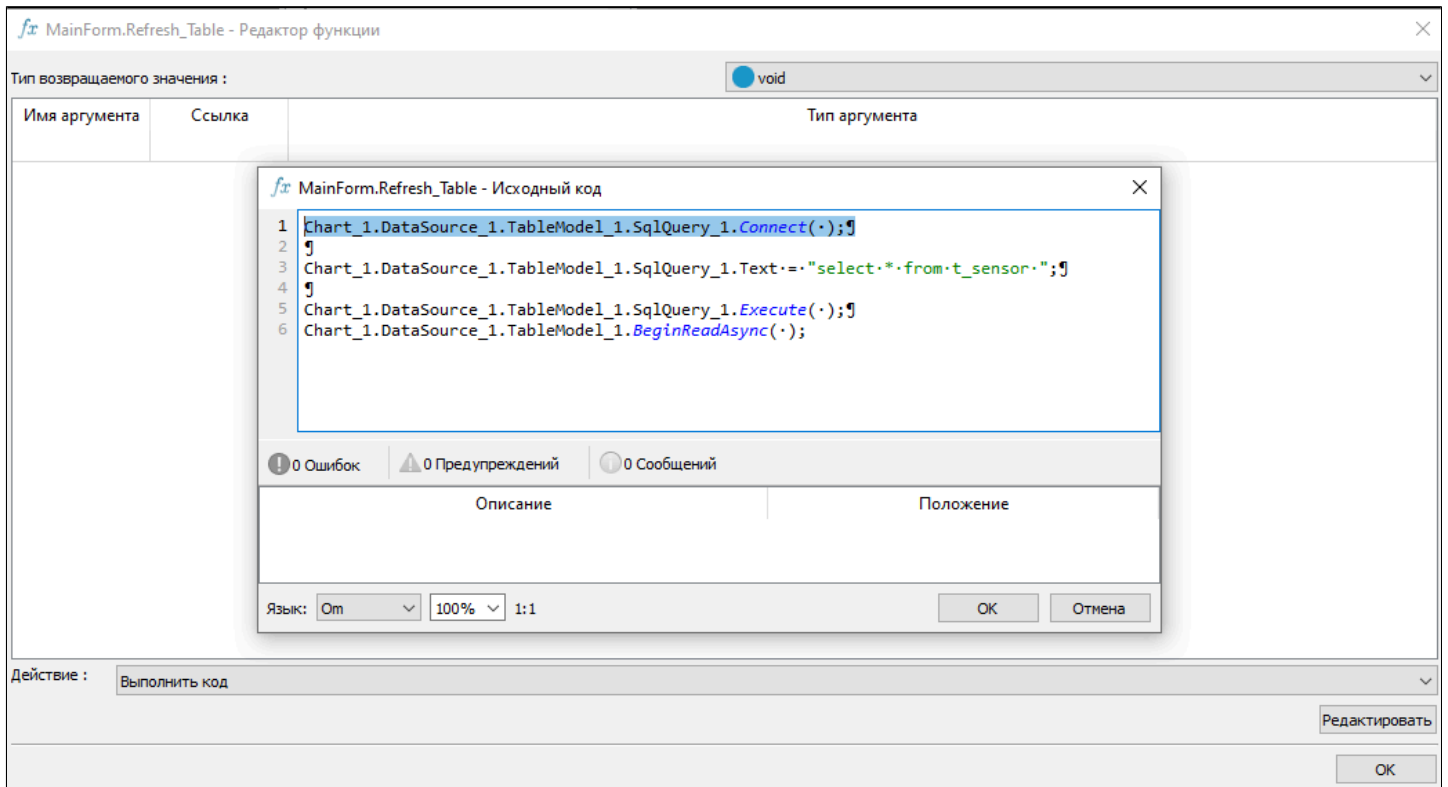


5. Настройте объект "Запрос SQL". В поле строка подключения запишите "DSN=<название источника данных>".

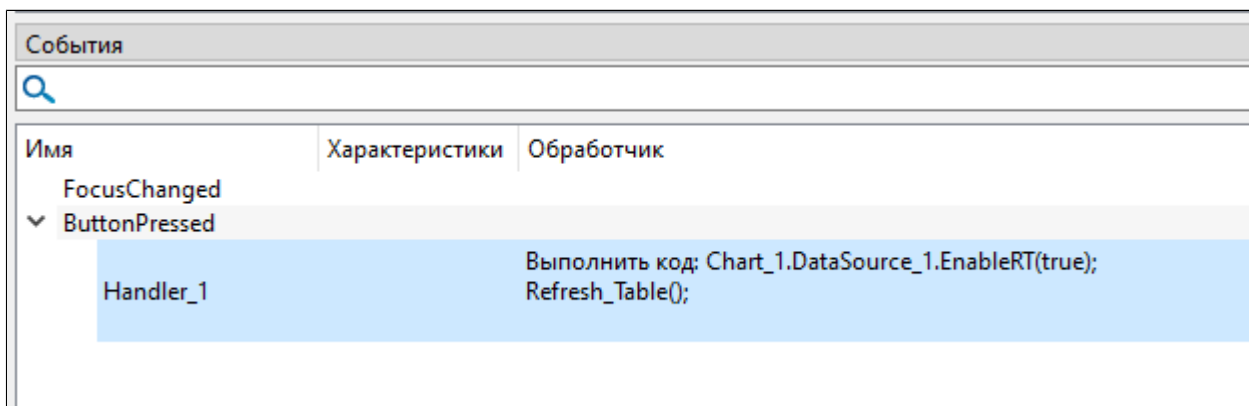
Настройка источника данных описана в разделе ["Создание источника данных"](#).

| Свойство | Харак | Значение |
|--------------------|-------|------------|
| Отображаемое имя | | SqlQuery_1 |
| Кардинальное число | | 1 |
| Строка подключения | R W | DSN=test |
| Текст запроса | R W | |

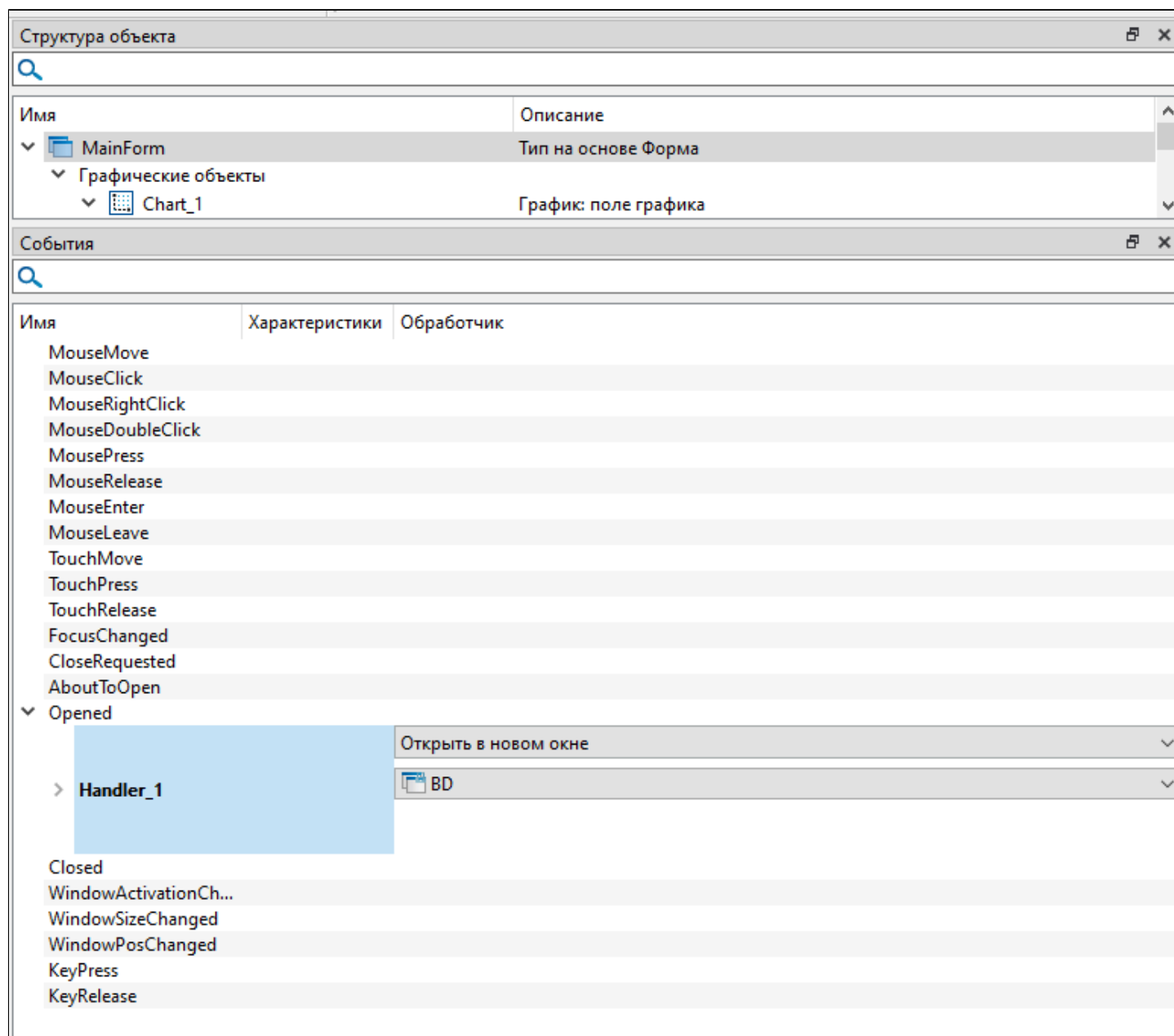
6. В функции "Refresh_Table" реализуйте логику подключения к внешней БД, выбор всех значений из таблицы и начало построение графика.



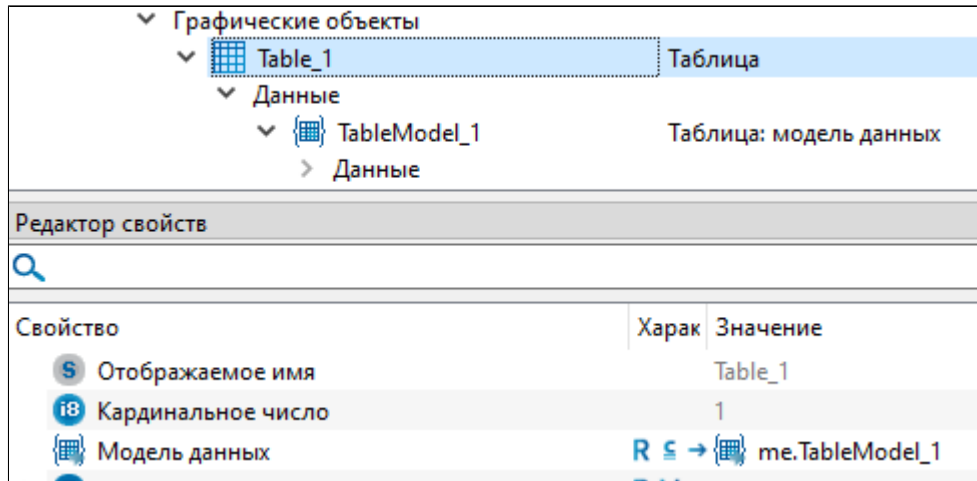
7. Добавьте событие нажатия на кнопку под графиком. По нему должен включаться режим "реалтайм" у графика и вызываться функция "Refresh_Table".



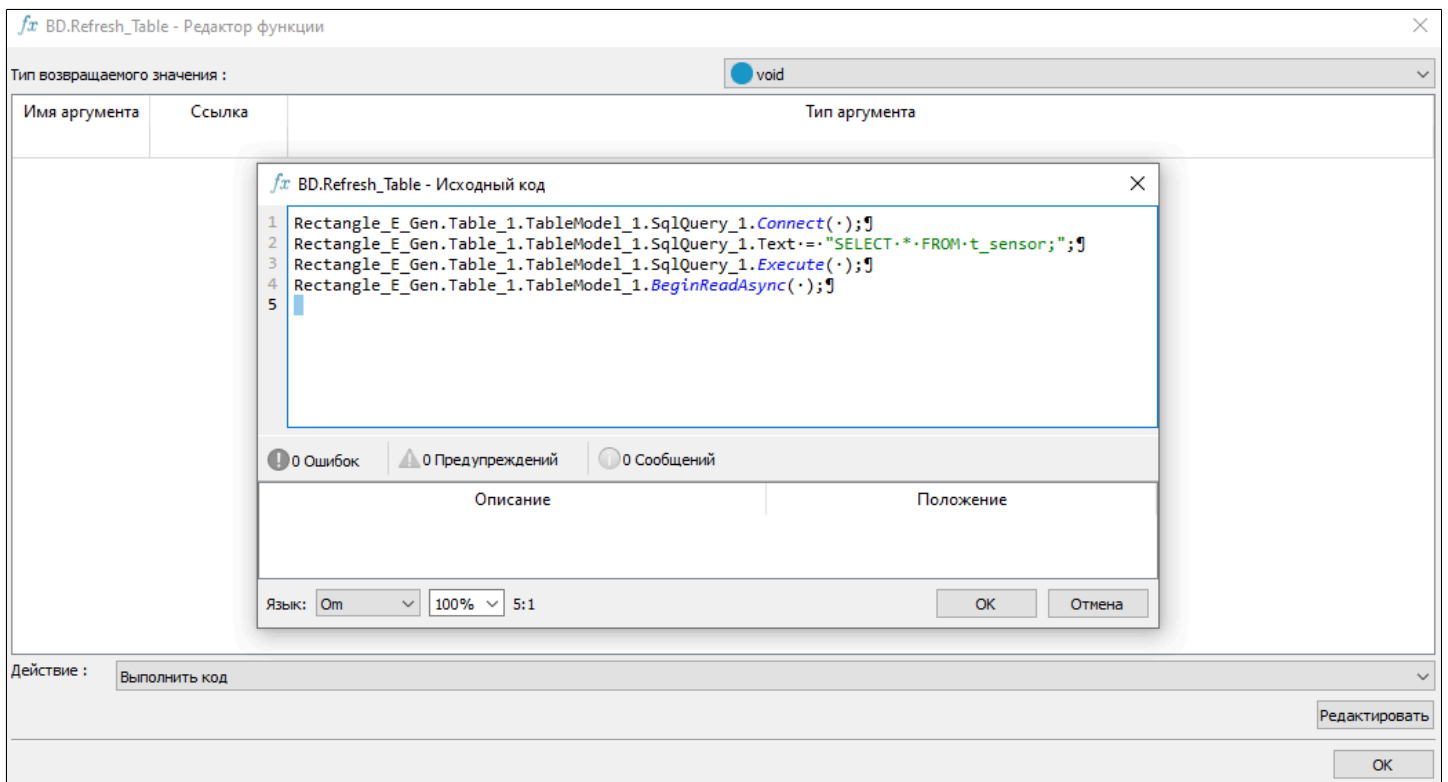
9. Создайте в "MainForm" событие "Opened" для открытия формы "BD" в новом окне.



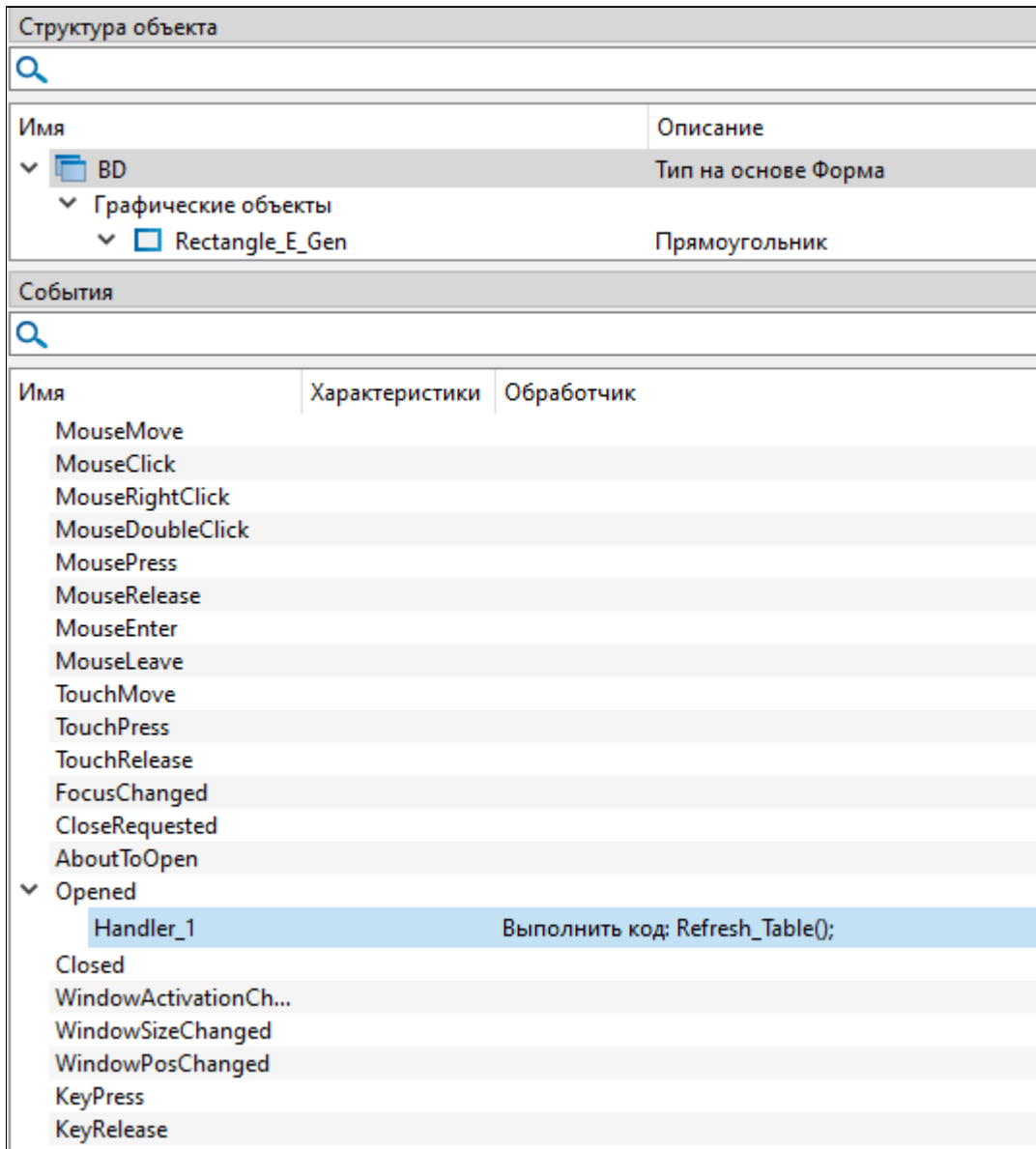
10. В объекте "Таблица" укажите модель данных.



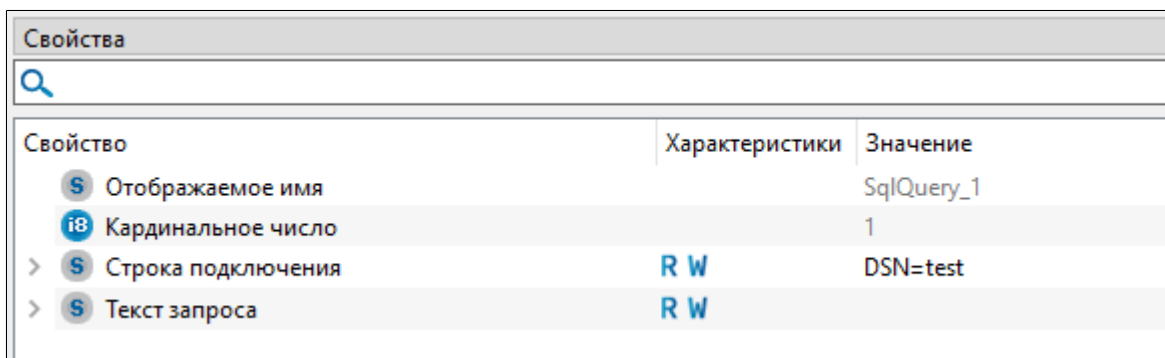
11. Перейдите на форму "BD" и реализуйте логику функции "Refresh_Table". Она аналогична логике в главной форме.



12. Создайте в "BD" событие "Opened" и вызовите в нем функцию "Refresh_Table".



13. Настройка объекта "Запрос SQL" полностью аналогична настройке в главной форме.



14. В свойствах "Заголовок" и "Идентификатор" для объектов "TableColumn_LineName", "TableColumn_LineName_1" и т.д. введите соответственно имена колонок "ID", "NAME", "VALUE", "DATE".

The screenshot shows a software interface with a list of objects and a properties editor. The list includes:

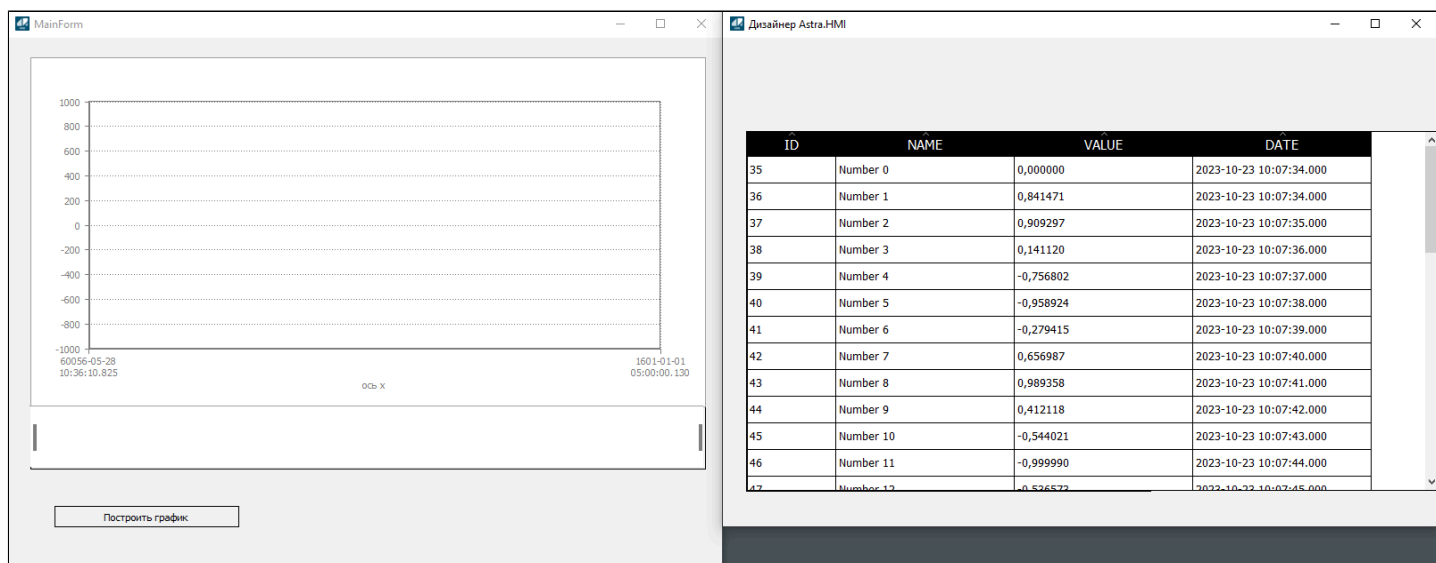
- SqlQuery_1 (Запрос SQL)
- TableColumn_LineName (Таблица: столбец)
- TableColumn_LineName_1 (Таблица: столбец) - selected
- TableColumn_LineName_2 (Таблица: столбец)
- TableColumn_LineName_3 (Таблица: столбец)

Below the list is a section labeled "Функции" (Functions) with a dropdown arrow. Below that is a "Редактор свойств" (Properties Editor) section with a search icon. The properties editor contains a table with the following data:

| Свойство | Харак | Значение |
|--------------------|-------|------------------------|
| Отображаемое имя | | TableColumn_LineName_1 |
| Кардинальное число | | 1 |
| > Заголовок | R W | NAME |
| > Описание | R W | |
| > Идентификатор | R W | NAME |

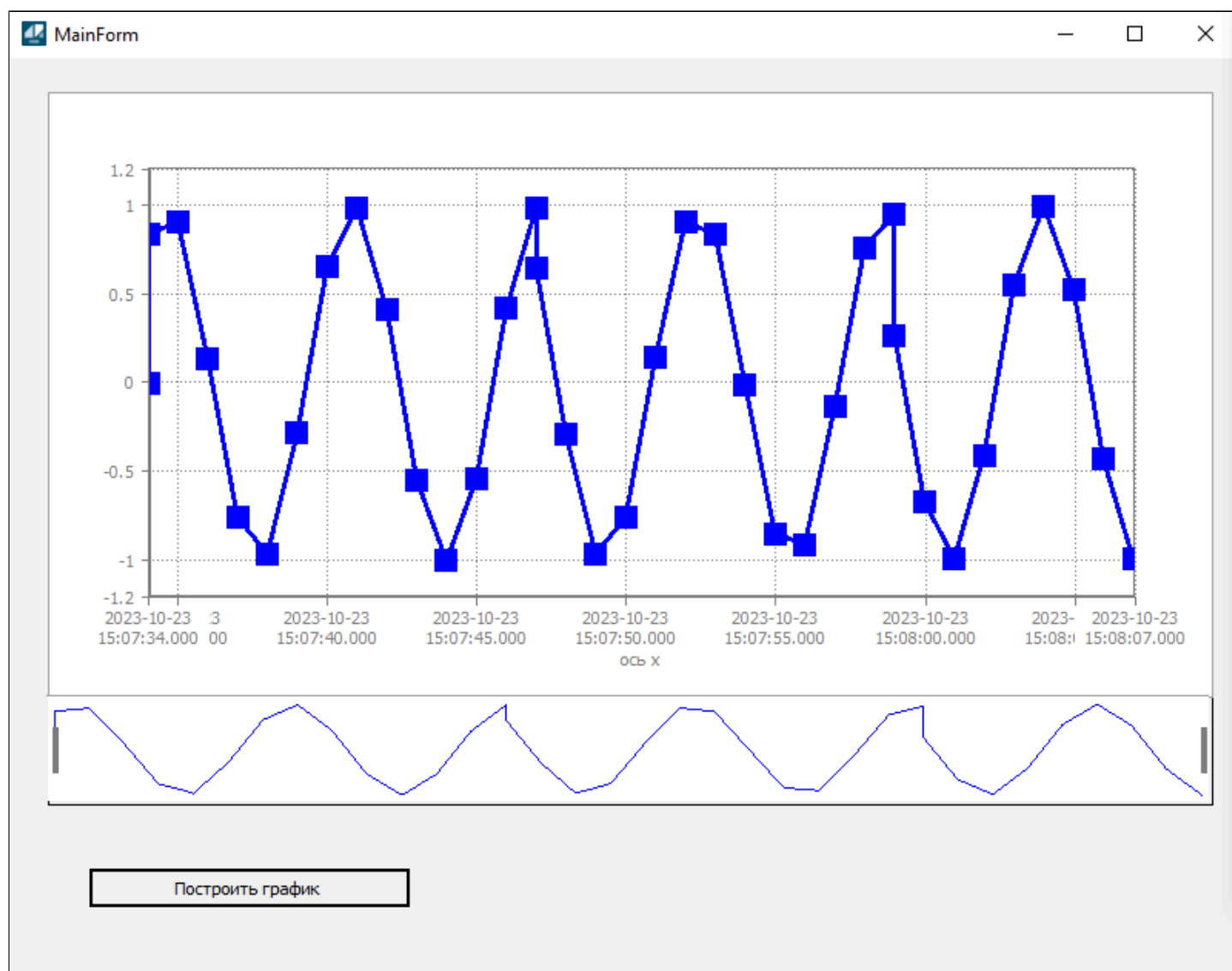
1.3.3.3. Проверка работы

1. Запустите проект нажатием на клавишу "F9".



Если вы правильно создали проект Astra.HMI, то у вас появится два окна, а таблица будет заполнена значениями из базы данных MySQL.

2. Нажмите на кнопку "Построить график".



Отобразился график, построенный на основе данных из MySQL.