Применение баз данных
В MasterSCADA

Часть 2

Методическое пособие

Москва 24 ноября 2011

Содержание

1. Использование хранимых процедур

Зачастую требуется интеграция SCADA с другими системами – MES, программами бухгалтерского учета. Как правило, взаимодействие с такими системами осуществляется через базу данных. Для этого SCADA должна записывать или считывать данные, используя SQL запросы.

В MasterSCADA взаимодействие с базами данных реализовано через *хранимые процедуры*. *Хранимые процедуры* – это объекты базы данных, представляющих собой набор SQL-инструкций. Хранимые процедуры могут также содержать *переменные* (*входные*, *выходные*, *локальные*) и набор команд на языке высокого уровня для обработки данных. Хранимые процедуры создаются в базе, используя средства администрирования. Выполнять хранимые процедуры можно на всех поддержанных в MasterSCADA базах данных – ***MSSQL***, ***Oracle, Interbase/Firebird***, ***Sybase***, ***MySQL***.

Рассмотрим работу с хранимыми процедурами на нескольких примерах. В качестве СУБД будем использовать ***MSSQL***.

Сначала необходимо добавить *БД-коннектор:* ***«MSSQL»***, и произвести настройку связи с сервером и базой данных. Задавать какие-либо настройки на *закладке* ***Использование*** не нужно. Хранимая процедура добавляется через *контекстное меню БД-коннектора* (Рисунок 1-1).

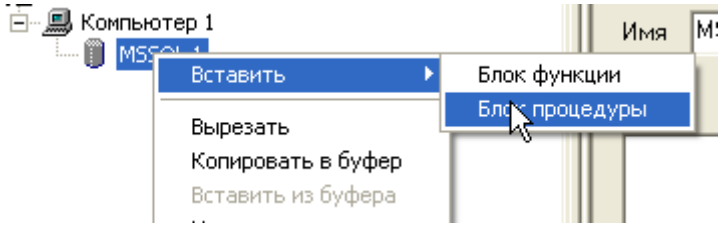


Рисунок 1-1

После этого нужно перейти на *закладку* ***Параметры*** *(рисунок 1-2)*. В верхней части закладки находится *поле для ввода имени* хранимой процедуры – в него нужно ввести *имя*, которое имеет процедура в базе. Процедура содержит *входы* и *выходы* (хотя может и не содержать), для их добавления есть специальная кнопка. Имена входов и выходов могут быть произвольными (не соответствовать именам в хранимой процедуре), но их число, тип и порядок следования должны соответствовать заданным в процедуре. Направление передачи данных через параметр (вход или выход) задается соответствующими *галочками* (Рисунок 1-2).

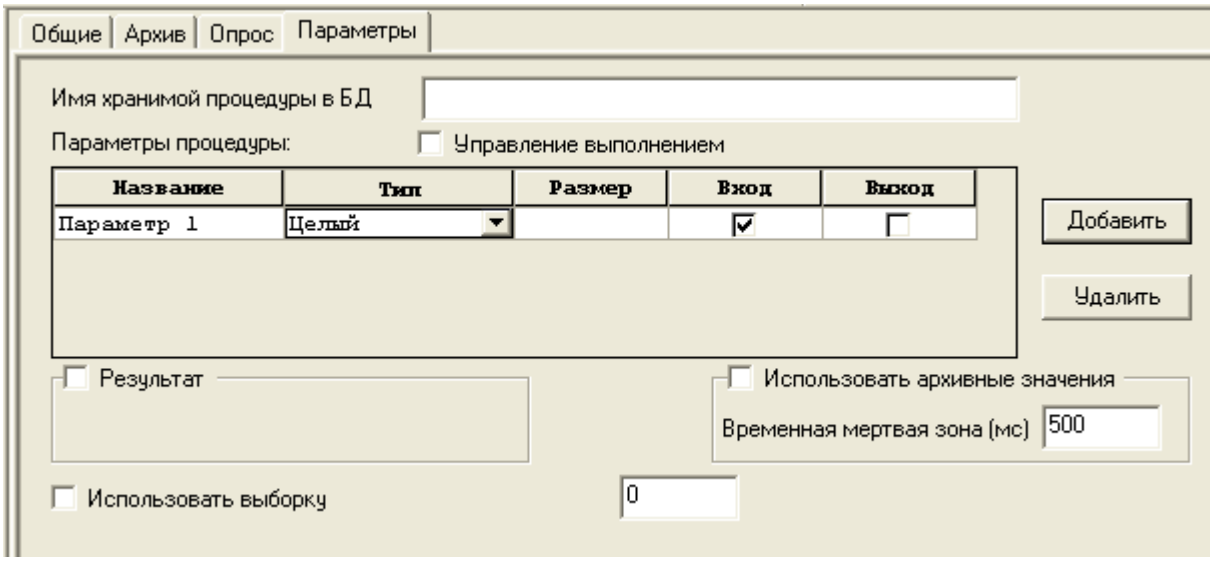


Рисунок 1-2

1.1. Запись в БД

Рассмотрим работу с хранимой процедурой на конкретном примере. Например, требуется записывать в базу значения по определенному событию.

Сначала создадим базу, используя ***Management Studio*** – аналогично, когда мы создавали базу для хранения данных архива MasterSCADA, назовем ее ***«First_example»*** (данное имя нужно также прописать в *БД-коннекторе* в MasterSCADA), добавим *таблицу* через *контекстное меню*, назовем ее ***«MyTable»*** (Рисунок 1-3).

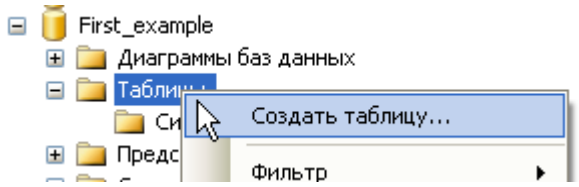


Рисунок 1-3

Мы будем записывать три значения и метку времени, поэтому нам нужно четыре столбца ***«Time_val»***, ***«Val1»***, ***«Val2»***, ***«Val3»***. Кроме того нам также будет нужен *первичный ключ*, в данном случае мы будем использовать инкрементируемое поле, назовем столбец ***«ID»*** (Рисунок 1-4).

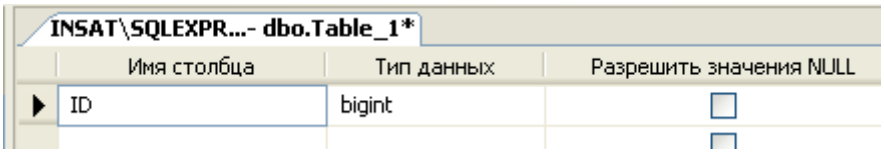


Рисунок 1-4

Нужно чтобы поле ***«ID»*** было инкрементируемым. В нижней части окна есть таблица ***Свойства столбца*** (Рисунок 1-5), найдем в нем параметр ***Спецификация идентификатора*** и установим его в ***«Да»***. Настройки начального значения и шага можно оставить без изменения – теперь при добавлении данных значение поле ***«ID»*** будет увеличиваться на единицу.

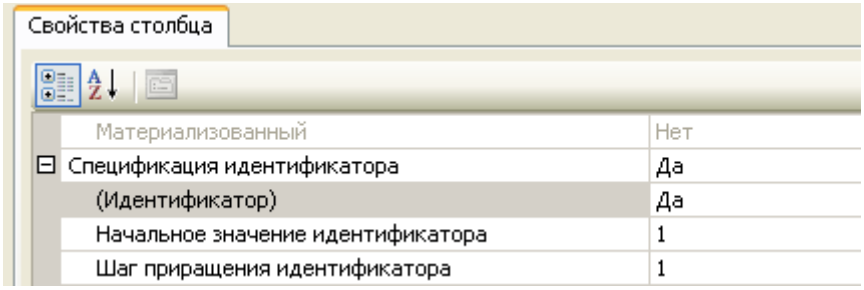


Рисунок 1-5

Вызовем *контекстное меню* столбца и выберем пункт: ***Задать первичный ключ*** (Рисунок 1-6).

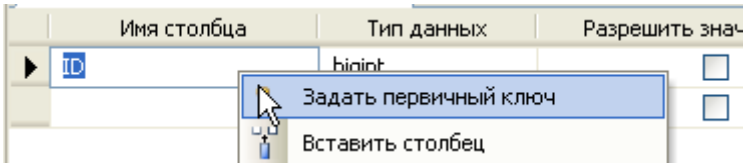


Рисунок 1-6

Столбец будет иметь метку в виде ключа. Добавим еще четыре столбца, один – типа ***DateTime*** (дата-время), остальные – ***int*** (целый).

Столбец: ***Разрешить значения Null*** - разрешает ввод пустых значений (Рисунок 1-7). Запретим ввод пустых значений для столбца ***«Time_val»***. В итоге таблица будет иметь следующий вид:

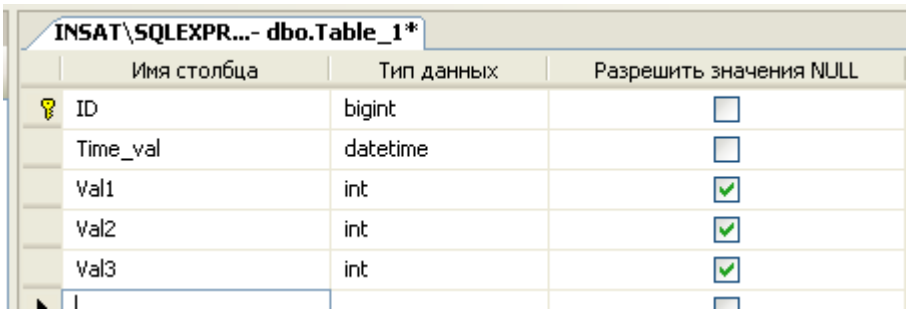


Рисунок 1-7

Теперь создадим хранимую процедуру, которая будет записывать данные в базу. Для этого в объекте базы ***«Программирование»*** есть специальный раздел ***«Хранимые процедуры»*** - вызовем *контекстное меню* и добавим хранимую процедуру. Введем в появившемся окне код процедуры.

```
CREATEPROCEDUREAddVal
@Time_valDateTime,
@Val1int,
@Val2int,
@Val3int

AS
BEGIN
      InsertintoMyTable(Time_val,Val1,Val2,Val3)values
      (@Time_val,@Val1,@Val2,@Val3);
END
      GO
```

В первой строчке объявляется имя процедуры – ***«AddVal»***. Затем задаются переменные, для упрощения мы сделали их имена аналогичные столбцам в таблице ***«MyTable»***. Обратите внимание, что перед переменными стоит знак ***«@»*** - это идентификатор переменной.

В секции ***Begin – End*** должен располагаться непосредственно код процедуры. В данном случае код состоит из одного SQL запроса на запись. В таблицу ***«MyTable»***, в столбцы ***«Time_val», «Val1», «Val2», «Val3»*** записываются значения из переменных процедуры ***@Time_val, @Val1, @Val2, @Val3***. Столбец ***«ID»***, как мы говорили ранее, будет заполняться самостоятельно – увеличивая свое значение.

Теперь настроим взаимодействие с MasterSCADA. Для этого добавим процедуру и настроим ее на *закладке* ***Параметры*** (Рисунок 1-8).

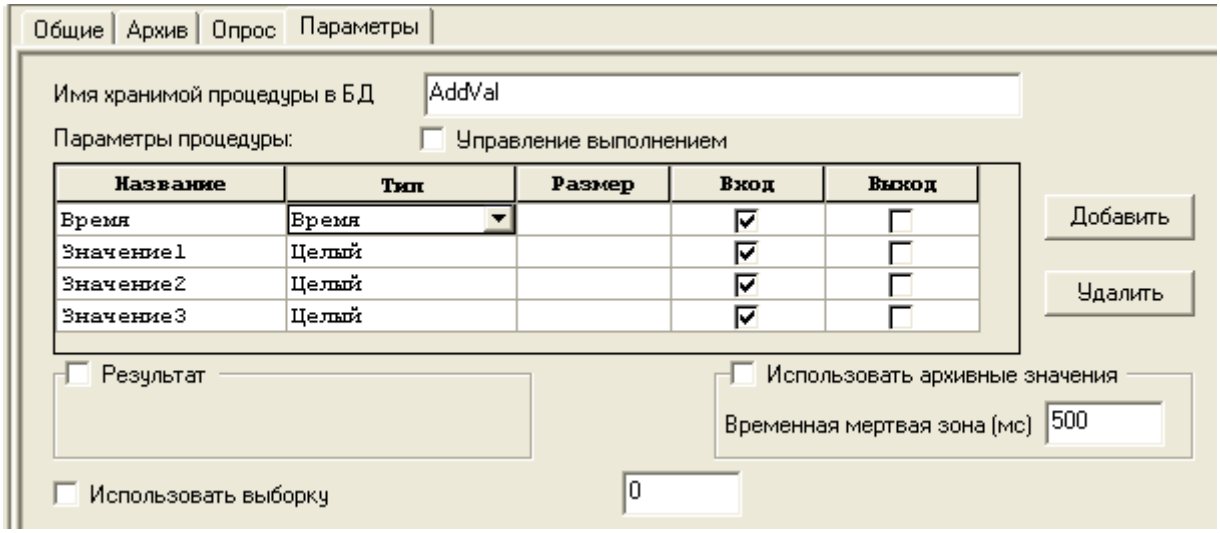


Рисунок 1-8

На стандартной *закладке Опрос* настраивается режим выполнения процедуры. По умолчанию установлен режим ***«По изменению»***. Это значит, что процедура будет выполняться, когда измениться какой-либо его из параметров. Но нам необходимо записывать данные по определенному событию. Для этого на *закладке* ***Параметры*** поставим *галочку* ***Управление выполнением***. Появится дополнительный вход ***«Выполнять»*** - когда на нем ***«Ложь»*** процедура перестает выполняться (Рисунок 1-9).

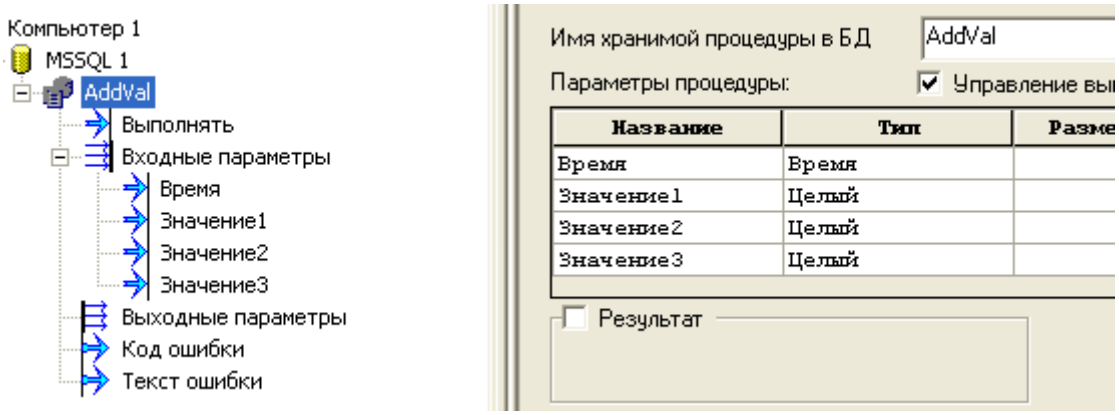


Рисунок 1-9

Теперь добавим объект в *дерево объектов*, а в него добавим необходимые переменные – модуль *расчет* с функцией ***DateTime( )*** (будет выдавать текущее время) и 3 *команды* типа ***«Целый»*** в режиме имитации. Также добавим один *ФБ* ***«Пульсатор»*** который будет выдавать импульс для

выполнения процедуры. Установим связи с соответствующими переменными в *дереве системы* (Рисунок 1-10).

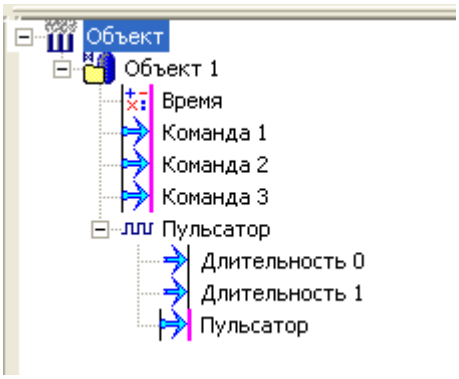


Рисунок 1-10

Теперь можно запустить режим исполнения. В случае возникновения ошибок на выходах ***«Код ошибки»*** и ***«Текст ошибки»*** появятся сообщения (Рисунок 1-11).

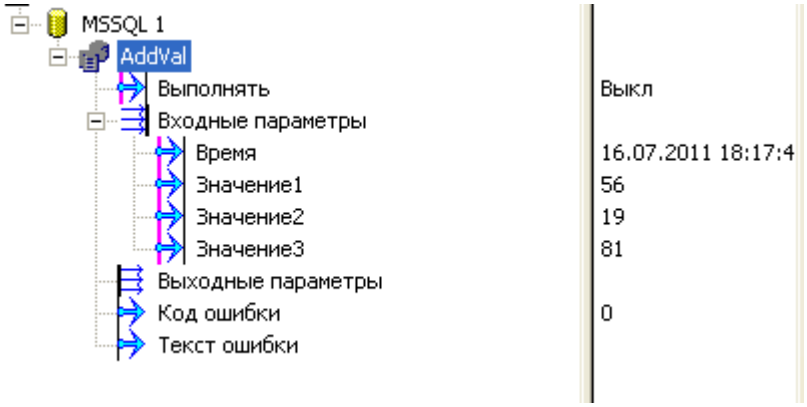


Рисунок 1-11

Остановим режим исполнения. Проверим, что данные в базу записаны. Для этого снова перейдем в ***Management Studio***, вызовем *контекстное меню* таблицы, и выберем пункт: ***Выбрать первую 1000 строк*** (Рисунок 1-12).

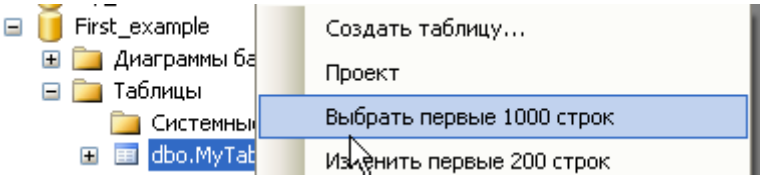
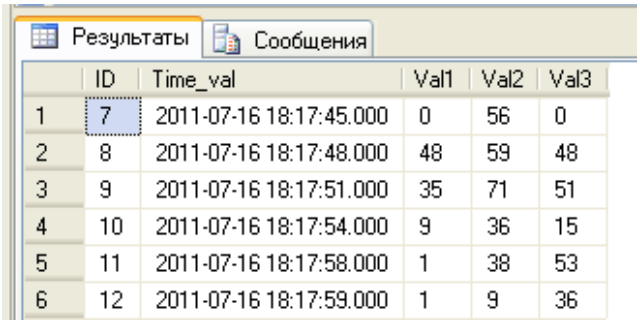


Рисунок 1-12

Будет выполнен SQL запрос, и мы сможем увидеть сохраненные в базе данные (Рисунок 1-13).



Результаты | Сообщения

	ID	Time_val	Val1	Val2	Val3
1	7	2011-07-16 18:17:45.000	0	56	0
2	8	2011-07-16 18:17:48.000	48	59	48
3	9	2011-07-16 18:17:51.000	35	71	51
4	10	2011-07-16 18:17:54.000	9	36	15
5	11	2011-07-16 18:17:58.000	1	38	53
6	12	2011-07-16 18:17:59.000	1	9	36

Рисунок 1-13

Существует также режим записи архивных значений. В этом режиме в базу передаются не действующие, а архивные значения и метка времени. При очередном выполнении процедуры определяется ближайшее время архивного значения, которое еще не было передано ранее (по всем архивным параметрам). Каждый цикл вызова хранимой процедуры в базу записывается строка из архива данных, таким образом если данные не изменялись, то и в базу они записаны не будут. В этом режиме также исключена возможность потери данных – если процедура какое-то время не будет выполняться, то впоследствии данные все равно будут записаны.

Для корректной работы необходимо поставить *галочку*: ***Использовать архивные значения***, а также добавить параметр типа ***«Время»***, указать его номер в соответствующем поле (подсоединять на его вход в дереве системы ничего не нужно!) (Рисунок 1-14). При этом *способ опроса* у хранимой процедуры нужно установить на ***Периодический***.

Если изменилось несколько параметров в пределах диапазона, заданного в поле ***Временная мертвая зона***, то эти параметры записываются в базу одним запросом.

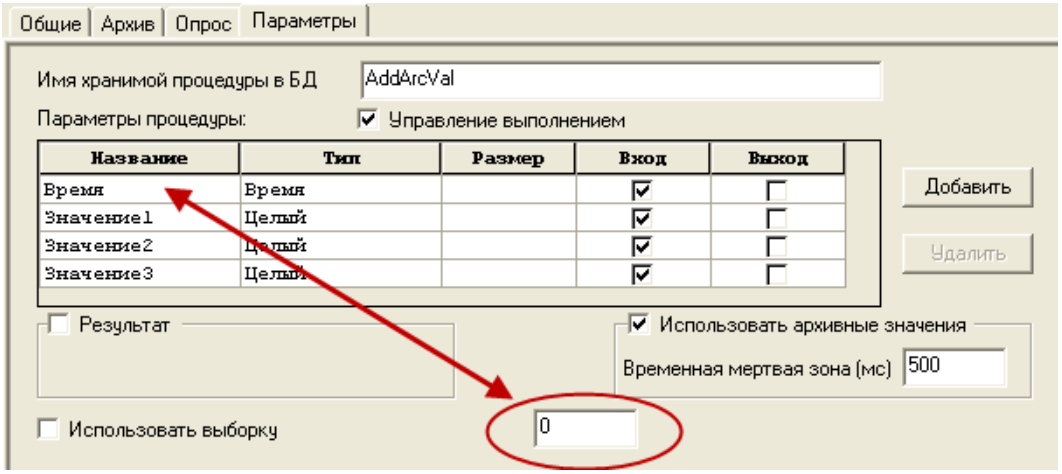
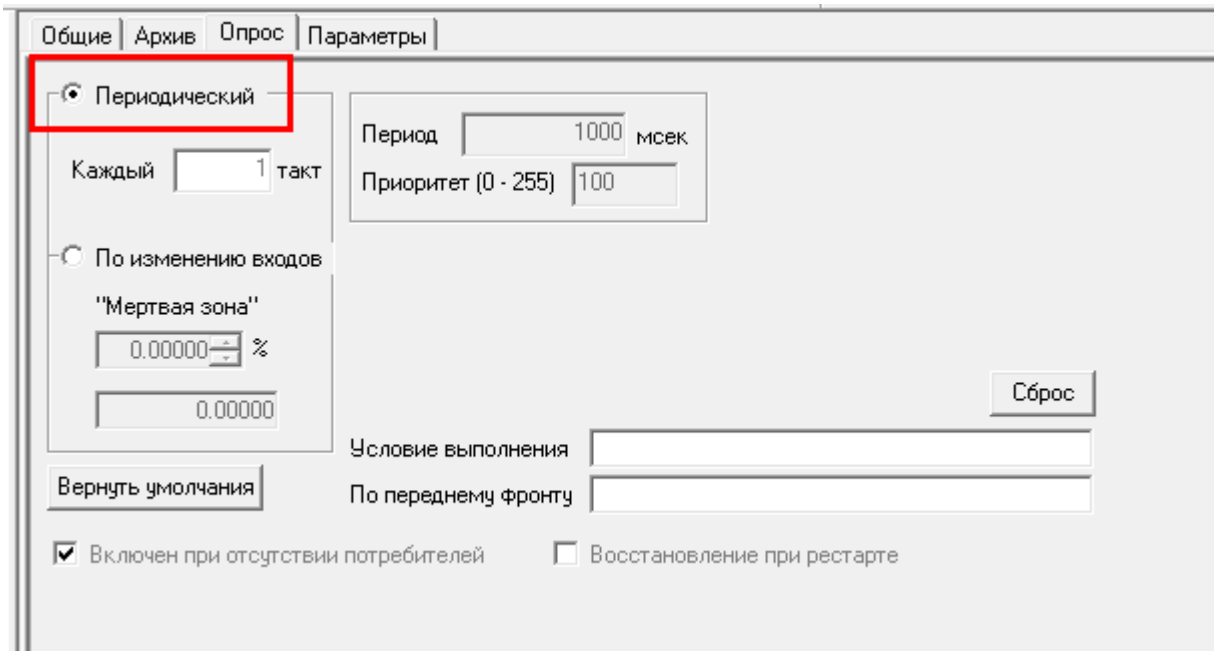


Рисунок 1-14

Кроме этого, у хранимой процедуры необходимо включить периодический опрос на вкладке ***«Опрос»***:



Хранимая процедура в таком режиме работы никаких отличий не имеет. Для данного примера она будет иметь такой же вид, как и в предыдущем примере (запись ведется в базу ***«MyTable1»***)

```
CREATEPROCEDUREAddArcVal
@Time_valDateTime,
@Val1int,
@Val2int,
@Val3int
AS
BEGIN
      InsertintoMyTable1(Time_val,Val1,Val2,Val3)values
      (@Time_val,@Val1,@Val2,@Val3);
END
```

Теперь в таблицу будут записываться значения из архива (Рисунок 1-15).

	ID	Time_val	Val1	Val2	Val3
10	10	2011-07-21 13:27:54.000	81	81	59
11	11	2011-07-21 13:27:56.000	48	35	90
12	12	2011-07-21 13:27:58.000	82	75	17
13	13	2011-07-21 13:28:00.000	86	71	51
14	14	2011-07-21 13:28:02.000	30	1	9
15	15	2011-07-21 13:28:04.000	0	56	19
16	16	2011-07-21 13:28:06.000	81	59	48
17	17	2011-07-21 13:31:23.000	10	20	30
18	18	2011-07-21 13:31:38.000	10	29	30
19	19	2011-07-21 13:31:46.000	10	29	37

Рисунок 1-15

1.2. Чтение из базы

Процедура для выполнения запроса чтения строится аналогичным образом. Вот пример процедуры выполняющей запрос чтения:

```
CREATEPROCEDURESelDate
@Time_valdatetimeout,
@Val1intout,
@Val2intout,
@Val3intout
AS
BEGIN

      SELECT@Time_val=MyTable.Time_val,@Val1=MyTable.Val1,
      @Val2=MyTable.Val2,@Val3=MyTable.Val3
      fromMyTable;
END
```

Однако нужно иметь ввиду, что запрос чтения может вернуть несколько строк, а на выходы процедуры MasterSCADA поступит только одна. Поэтому нужно создавать запрос, который установит на выходах только один набор значений. Для этого нужно использовать ограничение выборки оператором ***WHERE*** – он задает критерии поиска. В данном коде он будет искать только те записи, где ***«ID»*** записи равно заданному. Поскольку ***«ID»*** в данной таблице это первичный ключ, а он всегда уникален, то процедура будет возвращать только один набор значений (или ни одного).

```
ALTERPROCEDURE[dbo].[SelDate]
@Time_valdatetimeout,
@Val1intout,
@Val2intout,
@Val3intout,
@numberint
AS
BEGIN
      SELECT@Time_val=MyTable.Time_val,@Val1=MyTable.Val1,
      @Val2=MyTable.Val2,@Val3=MyTable.Val3
      fromMyTablewhereID=@number;
END
```

В этом случае окно настройки процедуры в MasterSCADA будет выглядеть как на Рисунок 1-16.

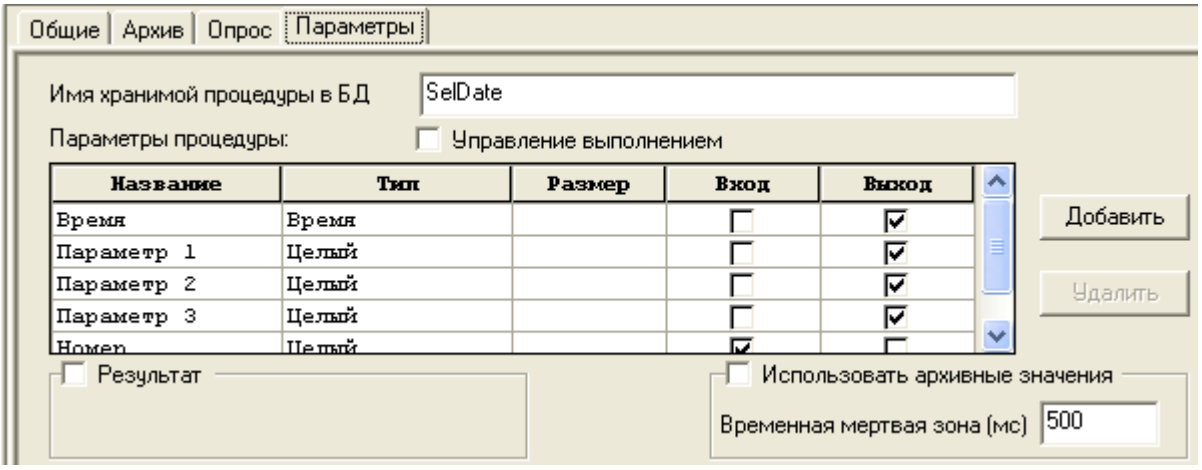


Рисунок 1-16

Но как быть, если необходимо в результате запроса чтения получить несколько значений? В этом случае необходимо использовать *режим выборки*. В этом режиме MasterSCADA запрашивает данные из базы, а полученные на выходы значения записывает в архив. В итоге с данными запроса можно работать как с архивными данными.

Рассмотрим пример. На сетевом компьютере работает SCADA-система, которая записывает свои данные в ***MSSQL***, необходимо получать эти данные в MasterSCADA.Таблица (***«MyTable»***) имеет следующую структуру (Рисунок 1-17):

ID	Time_val	val1	val2	val3	qual
219	2011-07-19 13:02:08.000	99,51781	45,74419	99,80163	1
220	2011-07-19 13:02:13.000	62,80709	60,41444	45,16129	1
221	2011-07-19 13:02:18.000	90,50874	24,28663	18,89401	1
222	2011-07-19 13:02:23.000	57,60369	86,77328	91,15574	1

Рисунок 1-17

Столбец ***«ID»*** (первичный ключ) тип ***int***, столбец ***«Time_val»*** времени в формате ***DateTime***, 3 столбца значений (***«val1», «val2», «val3»***) в формате ***real*** (вещественный) и поле признака достоверности – если признак качества хороший то в поле 1, если плохой – 0.

Сначала создадим в базе хранимую процедуру

```
CREATEPROCEDURESelectValue
AS
BEGIN
      SELECTTime_val,val1,val2,val3,(qual*192)  FromMyTable
END
```

Данная процедура будет возвращать все записи перечисленных столбцов из таблицы ***«MyTable»***. Признак качества умножается на 192 – это необходимо, что привести его к значениям качества OPC, с которыми работает MasterSCADA. Если запрос вернет значение признака качества 0, то MasterSCADA также получит ноль (статус ***«Ошибка»***), а если запрос вернет 1, то MasterSCADA получит признак качества 192 (статус ***«Хороший»***).

Теперь настроим MasterSCADA – добавим *БД-коннектор*, настроим в нем связи, добавим в него процедуру. Чтобы включить режим выборки нужно поставить соответствующую галочку. Появится дополнительная таблица, в которую нужно добавить возвращаемые процедурой переменные. Кроме того появились два дополнительных параметра – ***Временной параметр*** (с ним мы работали когда производили запись архивных значений) и ***Параметр качества***, с их помощью мы будем получать и назначать переменными метку времени и признак качества. Установим ***Временной параметр*** – 0, ***параметр качества*** – 4 (нумерация идет с нуля) (Рисунок 1-18).

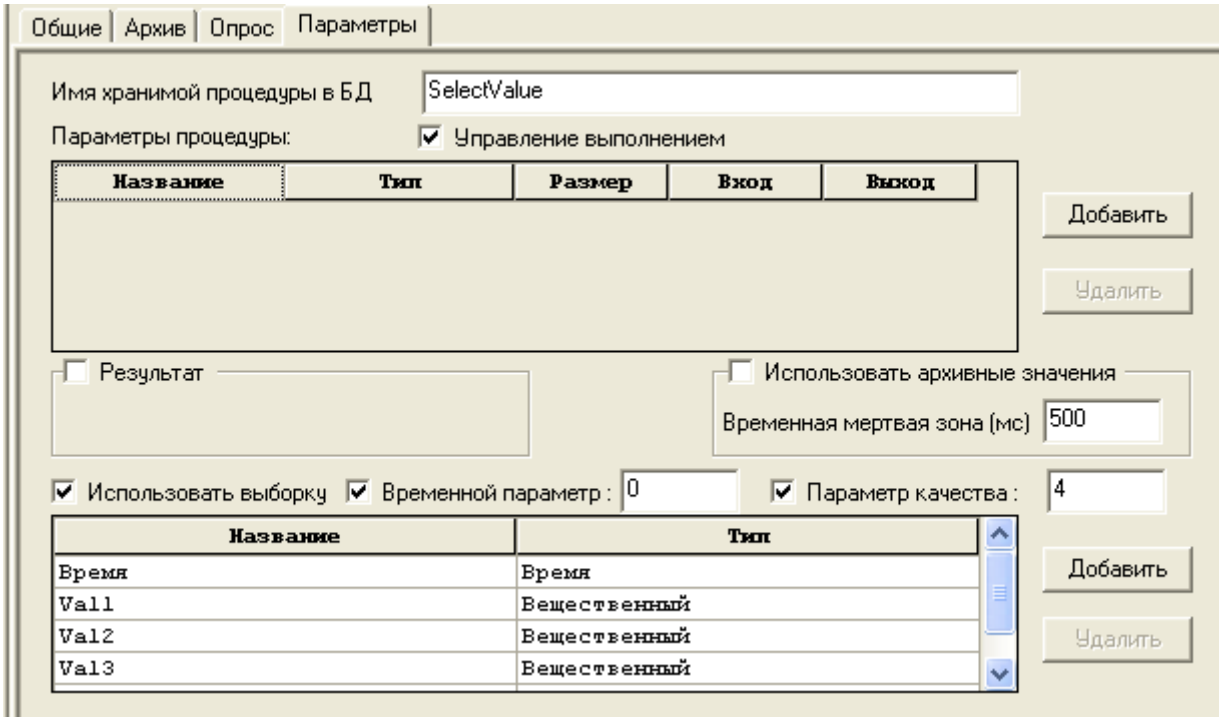


Рисунок 1-18

У выходов в *дереве системы* ***«Val1»***, ***«Val2»*** и ***«Val3»*** поставим *флаг архивирования*.

Запустим режим исполнения. Данные начали поступать в архив (Рисунок 1-19).

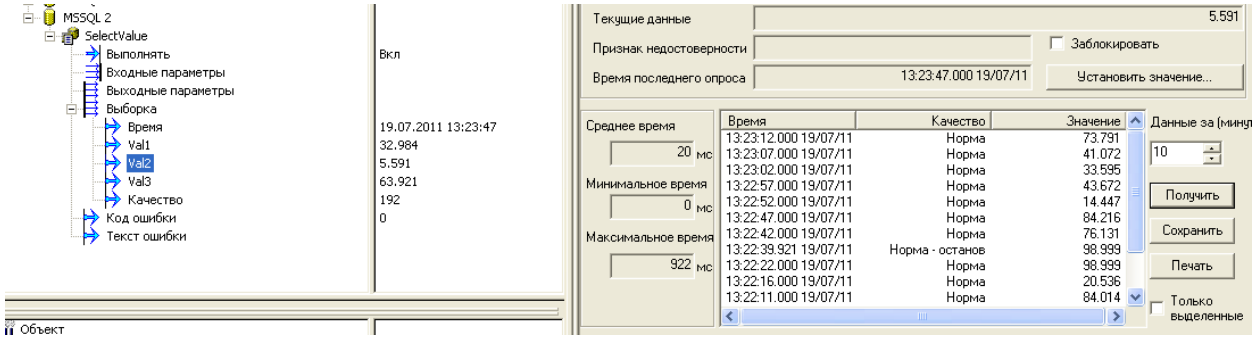


Рисунок 1-19

Но есть проблема – даже если данные не обновляются (вторая SCADA система ничего не пишет) данные все равно постоянно считываются – это видно в поле ***Текущие данные*** (каждый цикл опроса данные в нем быстро меняются). Таким образом, каждый запрос мы выбираем данные из всей таблицы. Конечно, MasterSCADA отбрасывает уже имеющиеся данные и не записывает их в свой архив, но по мере увеличения объема базы будет увеличиваться и объем получаемых данных, что приведет к нагрузке на локальную сеть и компьютеры. Чтобы избежать этой проблемы нужно ограничить глубину выборки запроса.

Мы создадим запрос, в который будем передавать метку времени – время последнего полученного значения, а запрос построим таким образом, чтобы он не возвращал данные имеющие более позднюю метку (не возвращал значения, которые уже есть).

Изменим процедуру:

```
ALTERPROCEDURE[dbo].[SelectValue]
@TimeValDateTime
AS
BEGIN

      SELECTTime_val,val1,val2,val3,(qual*192)  FromMyTable
      whereTime_val>@TimeVal
END
```

Итак, процедура имеет вход времени, а в условие запроса ***WHERE*** указано выбирать только те данные, время которых больше заданной метки. Таким образом, старые данные поступать в MasterSCADA не будут.

Теперь настроим процедуру в MasterSCADA –добавим вход типа «Время» (Рисунок 1-20).

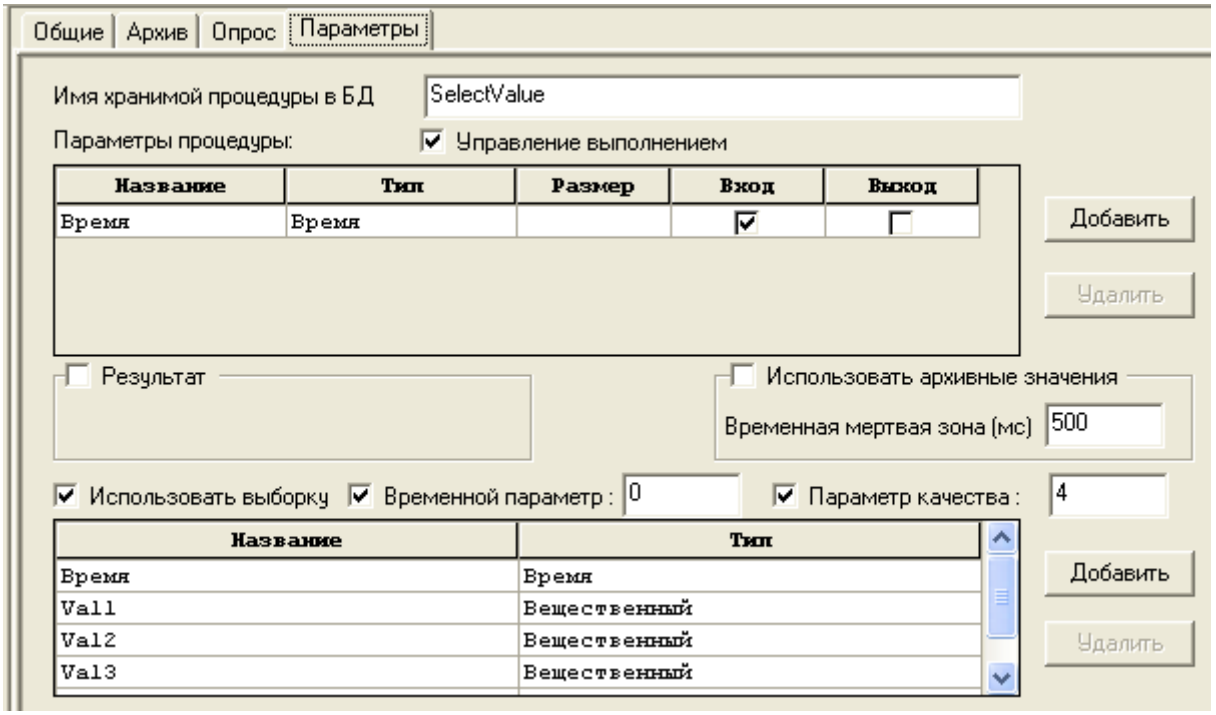


Рисунок 1-20

Теперь создадим *объект* в дереве объектов, добавим в него расчет, назовем его ***Метка времени*** сделаем, чтобы расчет возвращал метку времени какого-нибудь из выходов процедуры (Рисунок 1-21).

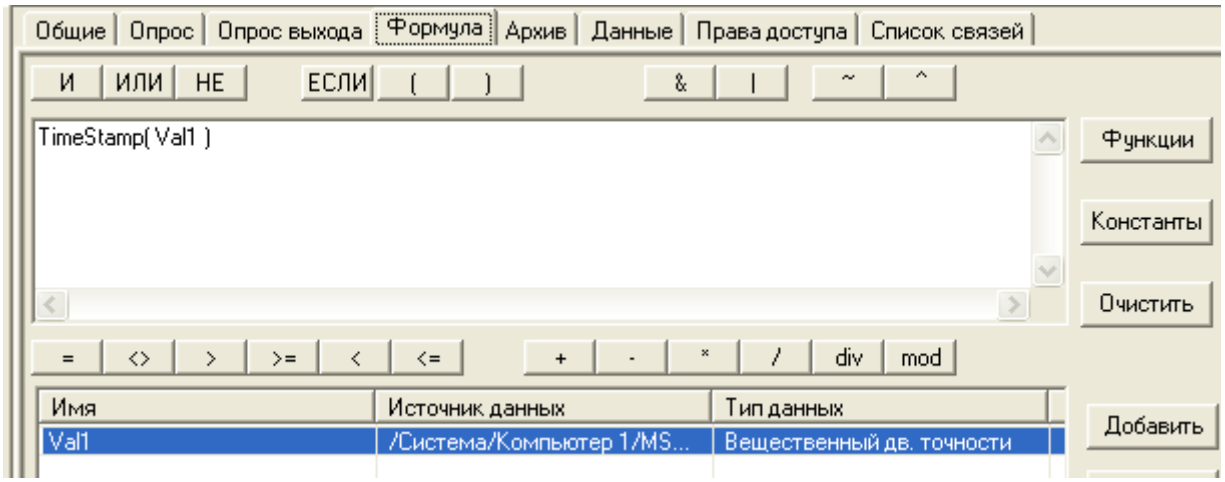


Рисунок 1-21

Для корректной работы на *закладке* ***Опрос выхода*** поставим *галочку* ***Восстановление при рестарте*** и поставим *значение до опроса* (Рисунок 1-22).

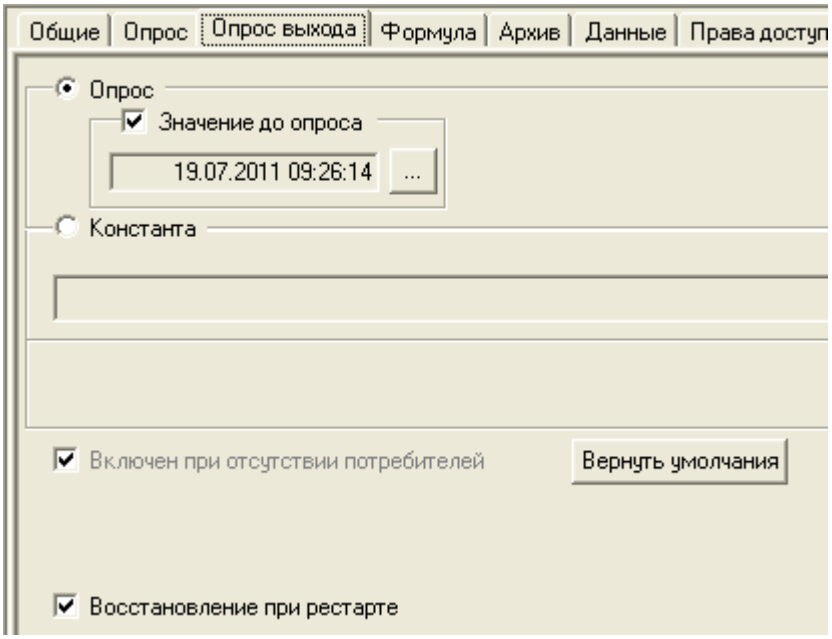


Рисунок 1-22

Установим связь между модулем ***Расчет*** и входом процедуры ***«Время»***.

Запустим режим исполнения - теперь в MasterSCADA поступают только свежие данные.

Если признак качества переменной вам не нужен, то снимите *галочку* ***Параметр качества*** на *закладке* ***Параметры*** в настройках хранимой процедуры MasterSCADA. В этом случае поступающие данные всегда будут иметь признак качества ***«хороший» (192)***.

Выборкой можно получать данные, не имеющие поля с меткой времени – для этого нужно снять *галочку* ***Временной параметр***. В этом случае поступающие данные будут получать метку времени в виде текущего времени, т.е. времени на момент получения данных.

Необходимо знать, что в режиме работы без метки времени MasterSCADA не может отбросить старые данные, поэтому необходимо ограничивать глубину выборки в процедуре. Например, если таблица имеет первичный ключ в виде инкрементирующего поля (ID), то для ограничения выборки можно использовать его – передавать в запрос выборки последнее полученное значение ID, и выбирать только те записи, где ID больше заданного.

1.3. Создание связанного сервера.

MasterSCADA взаимодействует с СУБД через хранимые процедуры. Однако иногда возникают ситуации, когда использовать хранимые процедуры невозможно – например, их создание

заблокировано администратором, но при этом существует возможность подключиться к базе и выполнить SQL-запрос. Обойти это ограничение поможет использование *связанного сервера*.

Идея связанного сервера в следующем. Необходимо установить на локальный компьютер (на компьютер с MasterSCADA) ***MSSQL***, создать в нем хранимую процедуру, добавить *связанный удаленный сервер* и получать из него данные, используя хранимую процедуру с локального сервера. В этом случае локальная СУБД ***MSSQL*** будет играть роль шлюза – через него, используя хранимую процедуру, мы будем получать данные с другого сервера.

Рассмотрим пример. На удаленном компьютере установлен ***MSSQL***, в ней находится *база данных* ***«First_example»*** с *таблицей* ***«MyTable»***. Хранимую процедуру на сервере создать нельзя. Необходимо получить данные из этой таблицы.

Установим на наш компьютер ***MSSQL Express*** и ***Management Studio***. Авторизуемся и зайдем на сервер. Развернем ***Объекты сервера***, а затем вызовем *контекстное меню* ***Связанные серверы*** и выберем пункт – ***Создать связанные сервер*** (Рисунок 1-23).

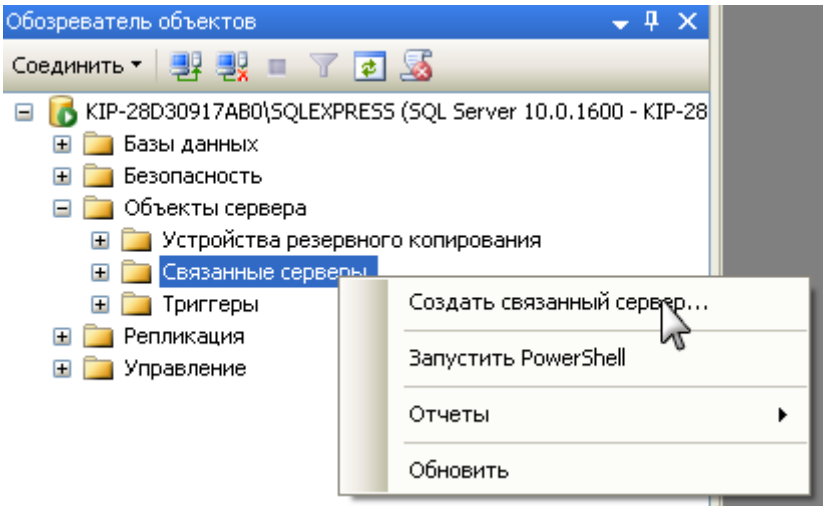


Рисунок 1-23

Появится окно ***Создание связанного сервера***. Установим *переключатель* ***Тип сервера*** в ***«Сервер SQL Server»***, в этом случае в поле ***Связанный сервер*** нужно ввести путь к удаленному MSSQL серверу (Рисунок 1-24).

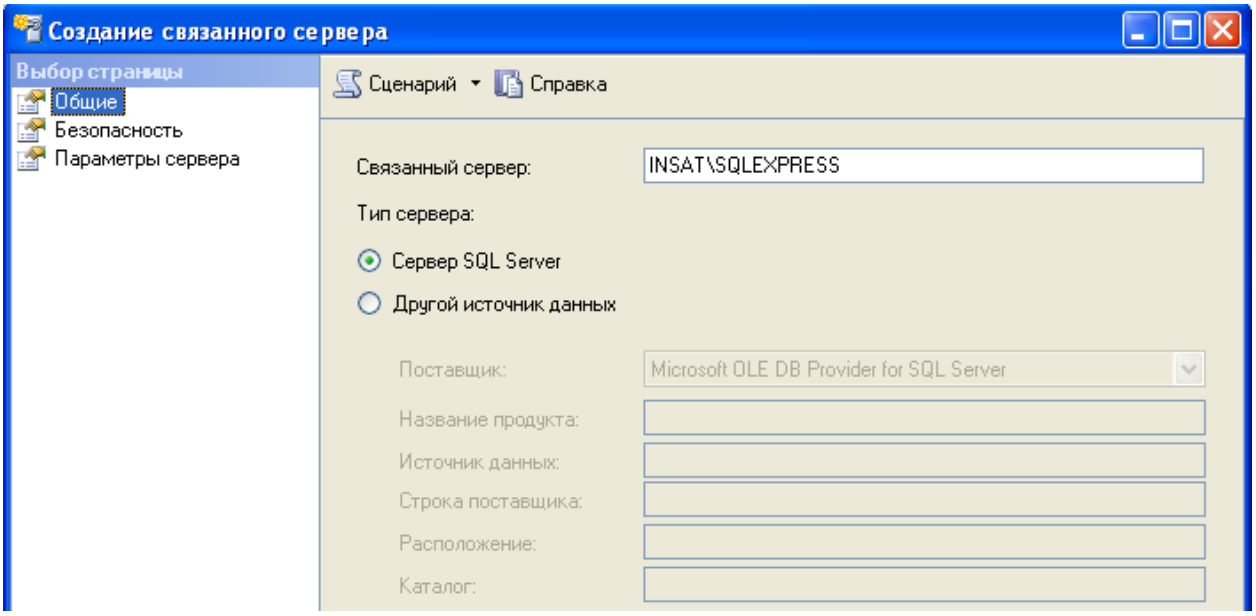


Рисунок 1-24

Перейдем в *раздел* ***Безопасность***. В нем необходимо задать пользователя для входа на удаленный сервер. Добавим имя входа. Сначала выберем из раскрывающегося списка ***Локальное имя входа*** локального пользователя, затем введем в поля ***Удаленный пользователь*** и ***Пароль для удаленного пользователя*** имя и пароль пользователя, которые нам предоставлены на удаленном сервере, – через эти имена локальный сервер подключиться к удаленному (Рисунок 1-25).

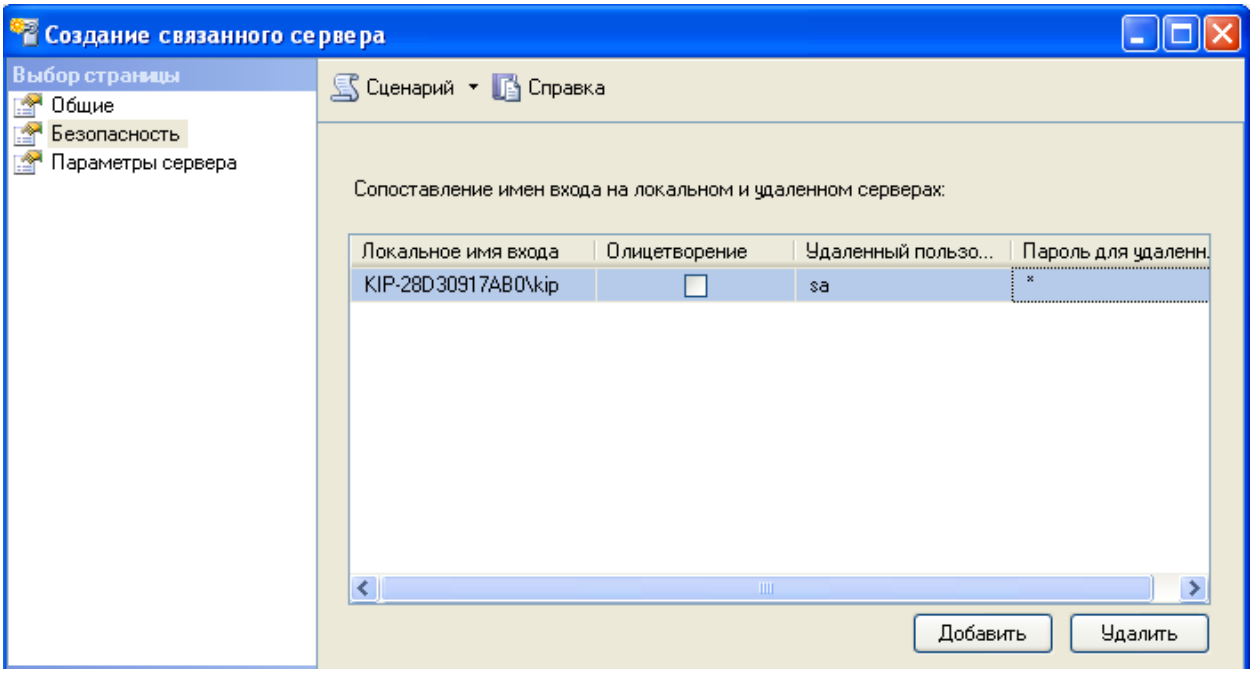


Рисунок 1-25

Нажмем ***ОК*** внизу окна. Если связь установится корректно, то в ***«Связанные серверы»*** добавится наш сервер, и мы сможем увидеть нужные нам таблицы (Рисунок 1-26).

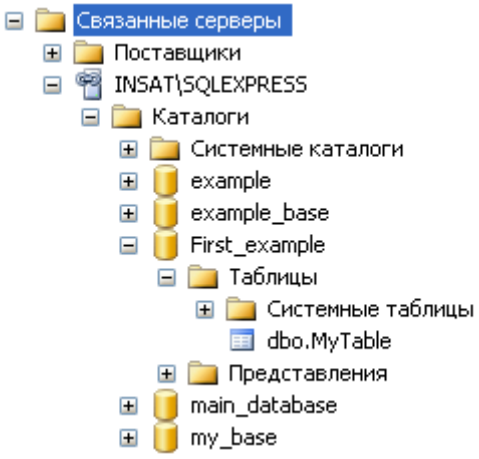


Рисунок 1-26

Теперь получим данные из *таблицы* ***«MyTable»***. Для этого сначала создадим на локальном сервере *базу данных*, назовем ее ***«MyBase»*** (Рисунок 1-27), добавлять в нее таблицы не обязательно.

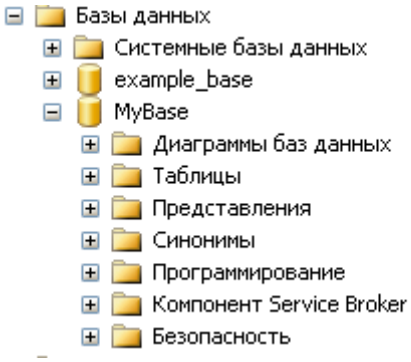


Рисунок 1-27

Теперь добавим хранимую процедуру. Процедура будет иметь такой текст:

```
CREATEPROCEDUREMyProc
AS
BEGIN
      Select*     From[INSAT\SQLEXPRESS].First_example.dbo.MyTable;
END
```

Данный запрос выбирает все данные из таблицы ***«MyTable»***. Обратите внимание, что сначала указывается *удаленный сервер* (в квадратных скобках), затем уже можно работать с *базой* и *таблицей*. Выполним процедуру:(Рисунок 1-28).

Результаты | Сообщения

	ID	Time_val	Val1	Val2	Val3
1	7	2011-07-16 18:17:45.000	0	56	0
2	8	2011-07-16 18:17:48.000	48	59	48
3	9	2011-07-16 18:17:51.000	35	71	51
4	10	2011-07-16 18:17:54.000	9	36	15
5	11	2011-07-16 18:17:58.000	1	38	53
6	12	2011-07-16 18:17:59.000	1	9	36

Рисунок 1-28

Теперь можно использовать процедуру в MasterSCADA. Для этого нужно добавить *БД-коннектор*, настроить его на связь с *локальной базой данных* и добавить в коннектор хранимую процедуру. Теперь MasterSCADA будет получать данные с удаленного сервера через локальный.

Связанный сервер можно использовать и для других целей. Например, с его помощью можно получить данные из СУБД, которая не поддержана в MasterSCADA (Линтер, DB2 и другие). Для этого при добавлении связанного сервера нужно выбрать: ***Другой источник данных*** и выбрать в раскрывающемся списке доступный OLEDB коннектор (Рисунок 1-29).

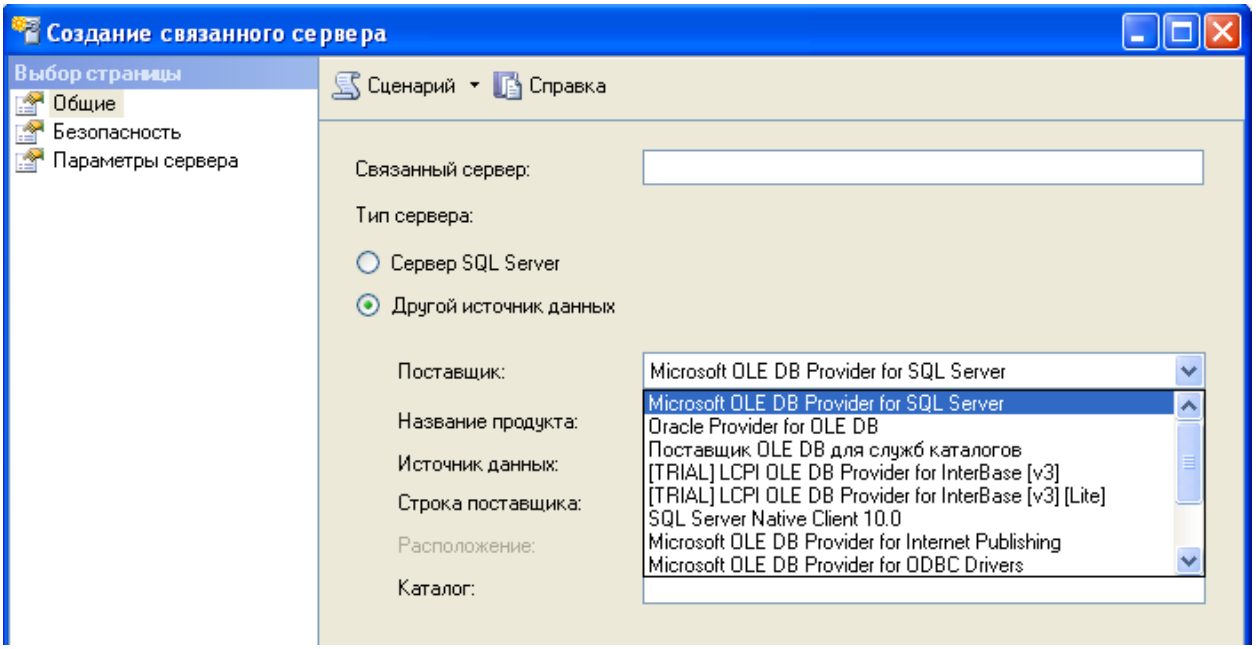


Рисунок 1-29

2. Использование других СУБД

Для выполнения хранимых процедур можно использовать все поддержанные базы данных. Для этого необходимо приобрести опцию связи с нужной базой данных или использовать архивный сервер ***MAS***. Способ создания и настройки ничем не отличается от рассмотренного примера.

Существуют некоторые особенности при работе с базами данных ***Firebird*** и ***Interbase***. На данный момент опция связи одна на две базы ***MSRT-Interbase-Firebird-SQL:***http://www.insat.ru/prices/info.php?pid=396

Это связано с тем, что ***Firebird*** основа на открытом коде ***Interbase*** версии 6, т.е. базы схожи. Для того чтобы выполнять хранимые процедуры в данных СУБД необходимо установить ***IBProvider:***http://www.ibprovider.com/rus/download.html

Для версий ***Interbase 6*** и ***Firebird 2.0*** (и ранних) можно использовать ***Free*** версию провайдера, для новых версий необходима ***Pro*** версия. Независимо от того, какая будет использоваться база данных для выполнения хранимых– ***Firebird*** или ***Interbase***, необходимо добавлять в компьютер *БД-коннектор*:***Interbase***.

В будущем, для ***Firebird*** мы планируем поддержать бесплатный: ***Firebird ADO.NET Data Provider***, что позволит обходиться без ***IBProvider***.

В настройках связи *БД-коннектора* нужно указать полный путь к базе данных, имя пользователя и пароль (Рисунок 2-1).

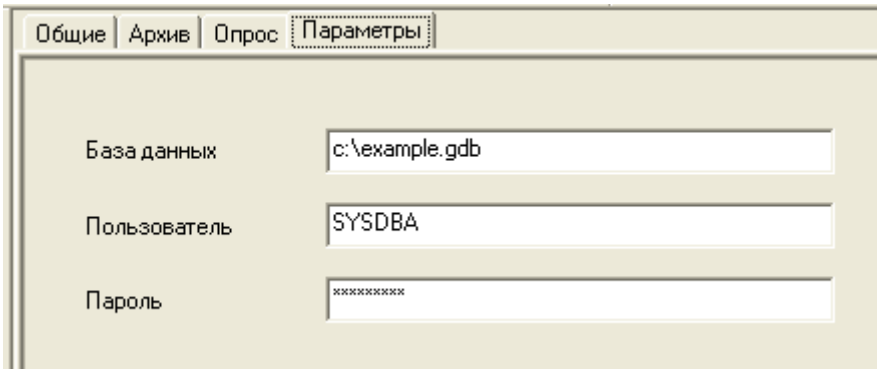


Рисунок 2-1

Дальнейшая работа с хранимыми процедурами не отличается от рассмотренных примеров.

3. Использование редактора отчетов для выполнения запросов

Данные из SQL баз можно получать не только через БД-коннекторы и хранимые процедуры, но также используя редактор отчетов MasterReport. В этом случае данные из SQL будут

использоваться только в создаваемом отчете. С подробной документацией по редактору отчетов можно ознакомится здесь:

http://www.insat.ru/services/support/art_step_by_step/Otchet.ZIP

Редактор отчетов MasterSCADA имеет возможность получения данных для отчета из различных источников. Стандартным *источником* является ***«Архив MasterSCADA»*** – при помощи него можно получить данные накопленные в SCADA системе.

В некоторых случаях данные необходимо получить другими путями – через файлы таблиц ***csv*** или, например, напрямую из базы данных – используя ***SQL*** запрос.

Создадим пример, в котором, используя редактор отчетов, мы получать данные из SQL-сервера за определенный промежуток времени. Отчет должен содержать таблицу следующего вида: «***ID»*** (уникальный номер записи), ***«Время»***, ***«Величина значения»***. Все данные мы будет получать из SQL-сервера, переменные ***«Начало»*** и ***«Конец»,*** для задания диапазона выборки значения, мы будем получать из дерева объектов MasterSCADA.Для таких отчетов необходимо использовать *источник данных* ***«Данные из SQL соединения»***. В качестве SQL сервера будем использовать Microsoft SQL Server 2008. Сначала создадим базу данных и наполним ее данными.

Используя администратор баз ***Microsoft SQL Server Management Studio***, создадим новую базу данных (Рисунок 3-1).

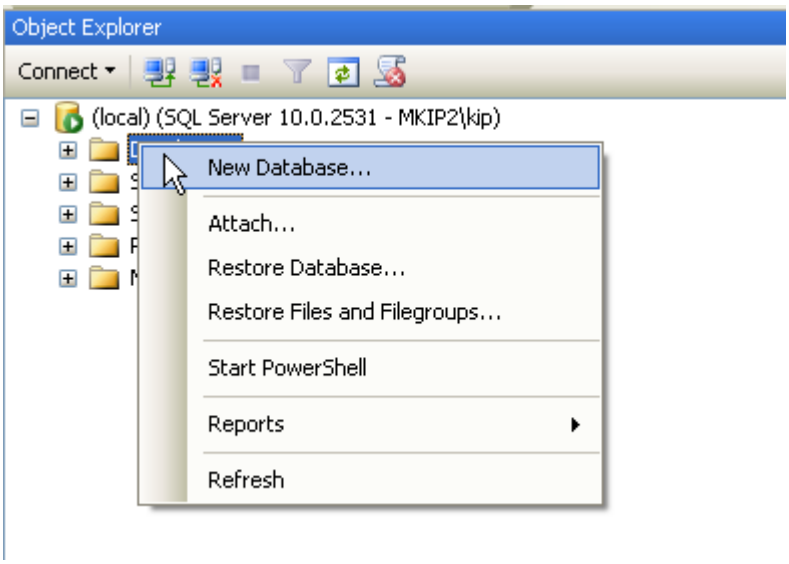


Рисунок 3-1

Назовем ее ***«report_database»*** (Рисунок 3-2)**.**

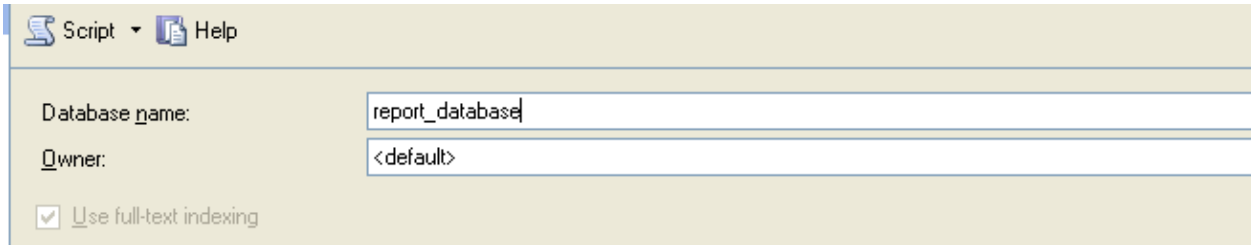


Рисунок 3-2

Добавим новую таблицу. Назовем ее ***«my_table»***. Сделаем три поля: уникальный номер поля – ***«ID»*** (тип *bigint*, поле не может быть *null*), значение переменное – ***«Value»*** (тип *int*), и время переменной – ***«Time»*** (тип *datetime*).

Наполним таблицу данными: (Рисунок 3-3):

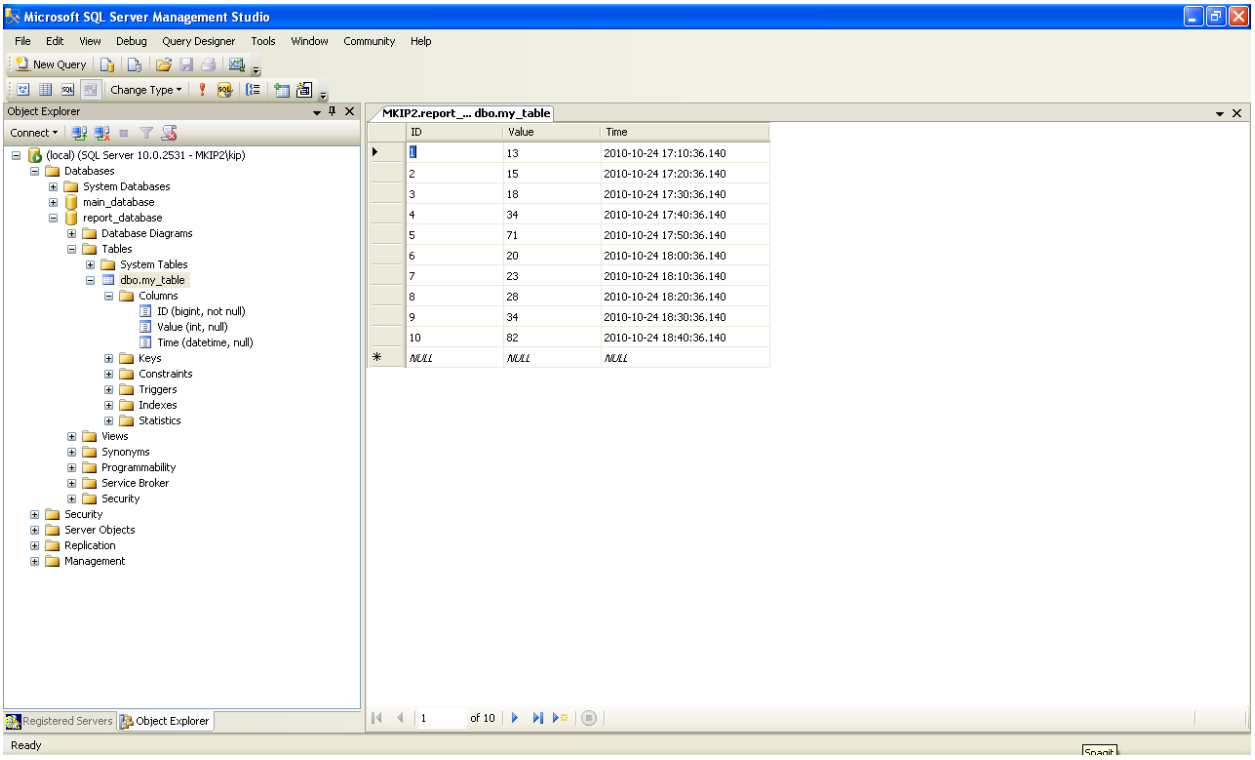


Рисунок 3-3

Создание базы завершено – закроем администратор баз.

Запустим MasterSCADA и создадим проект. В проект добавим объект, в объект добавим две переменные типа *время* – ***«начало»*** и ***«конец»***. На *закладке* ***Опрос*** зададим переменным параметр ***Значение до опроса*** - в этом случае мы сможем просмотреть отчет в режиме предварительного

просмотра (не запуская систему в режим исполнения). Этими переменными мы будем ограничивать диапазон времени выборки переменных из базы данных.

На закладке объекта ***Отчеты*** добавим отчет. Перетащим в словарь отчета переменные ***«начало»*** и ***«конец»*** из дерева объектов. Переменные появятся в словаре (Рисунок).

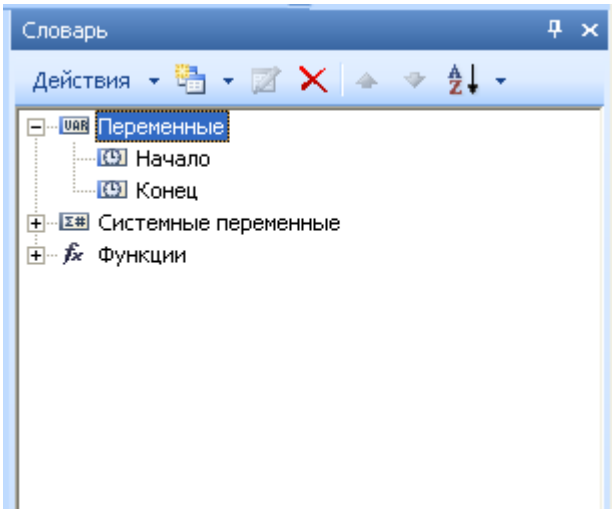


Рисунок 3-4

Теперь получим данные из базы данных.

Для этого сначала нужно установить соединение с базой данных. Вызовем *контекстное меню* словаря и выберем пункт: ***Новое соединение*** (Рисунок 3-5).

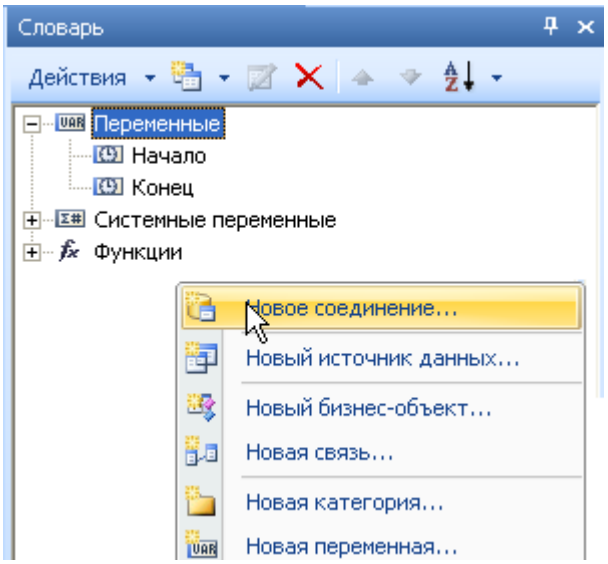


Рисунок 3-5

В появившемся окне выберем – ***SQL соединение*** (Рисунок 3-6).

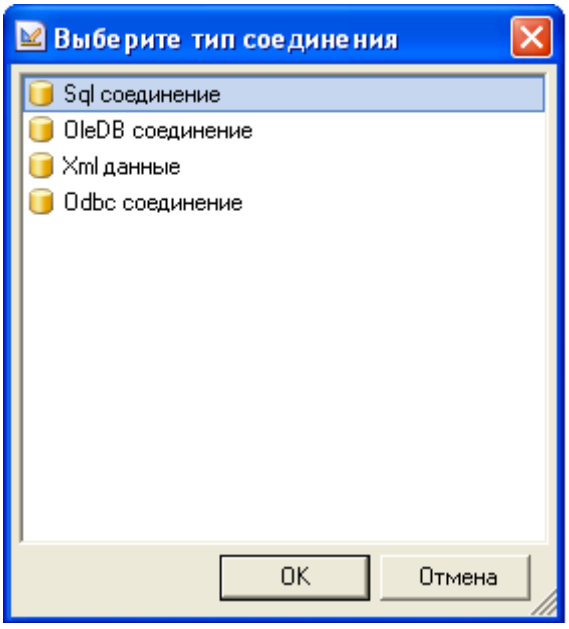


Рисунок 3-6

В появившемся окне введем имя соединения – ***«SQL_connection»*** (Рисунок 3-7).

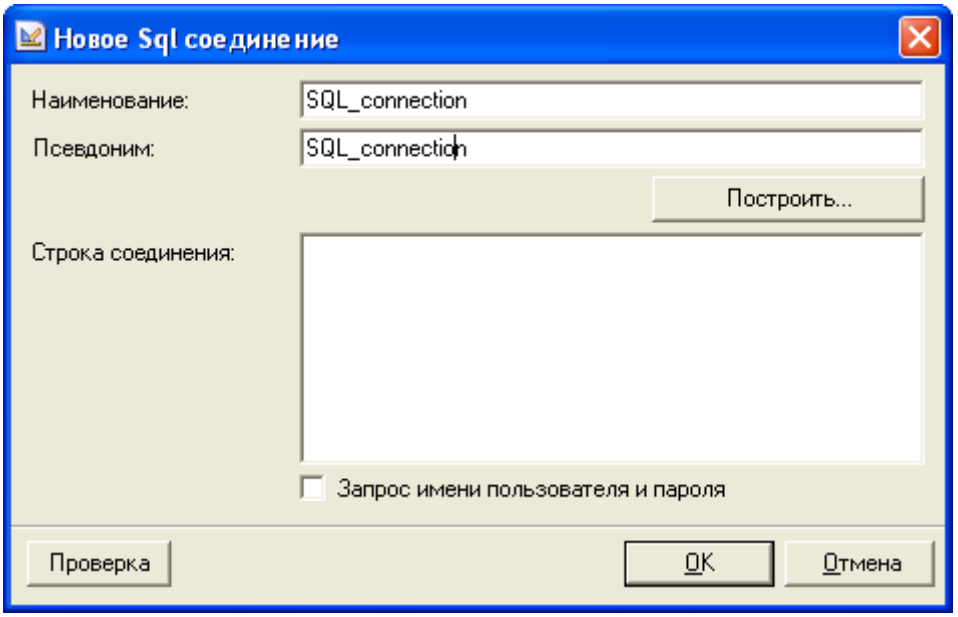


Рисунок 3-7

В *поле* ***Строка соединения*** нужно ввести параметры соединения – *имя базы данных*, *имя пользователя*, *пароль*. Можно это сделать вручную, но лучше воспользоваться автоматическим построением. Для этого нажмем на кнопку: ***Построить***.

В появившемся окне (Рисунок 3-8) введем ***имя соединения*** с базой данных, и выберем нужную нам базу данных и нажмем ***ОК***.

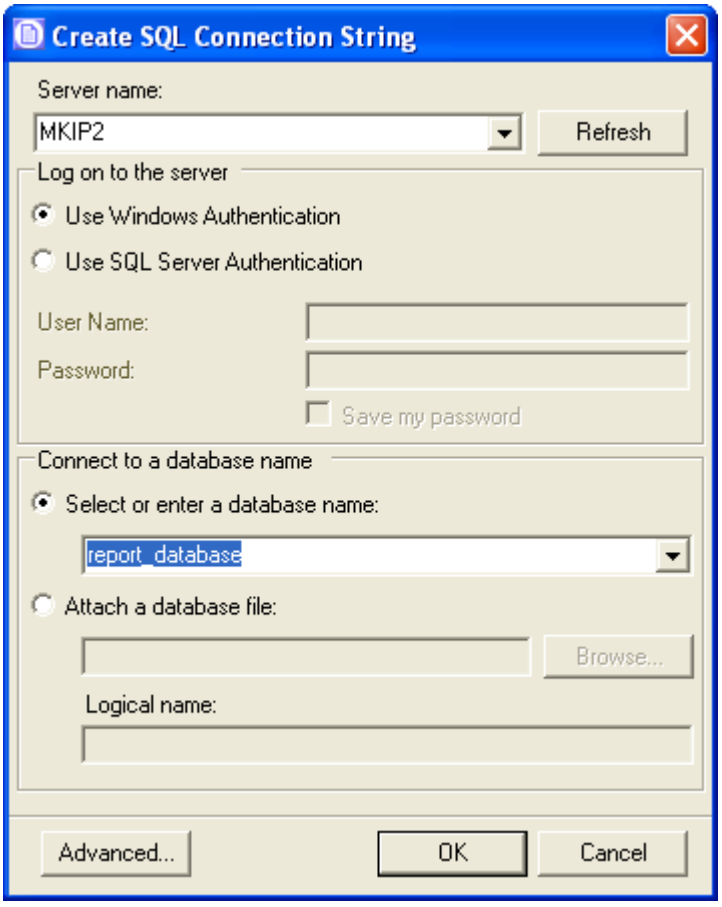


Рисунок 3-8

Строка соединения пропишется в поле (Рисунок 3-9)

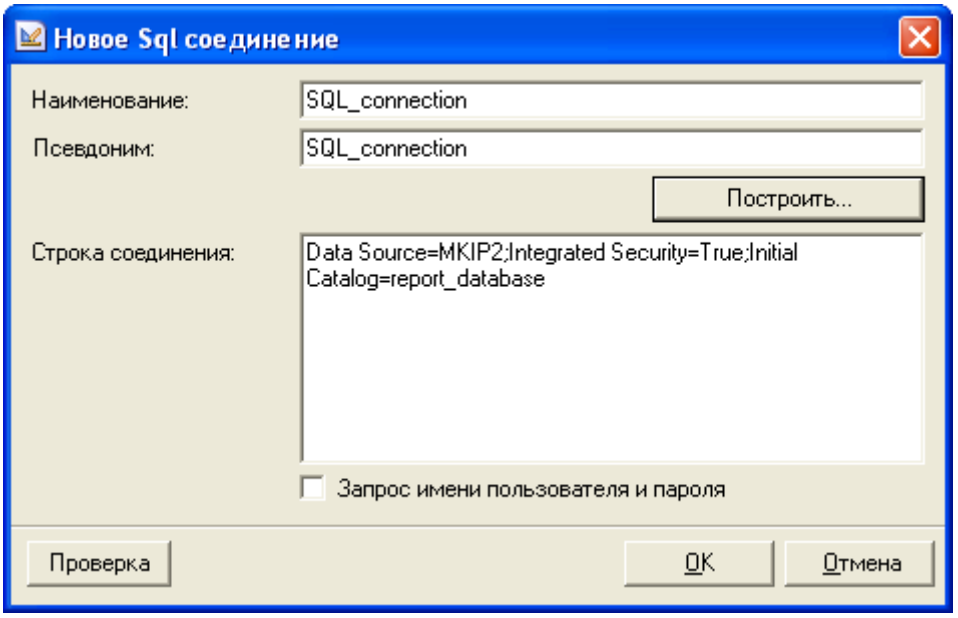


Рисунок 3-9

Проверим соединение – нажмем на кнопку ***Проверка*** (Рисунок 3-10).

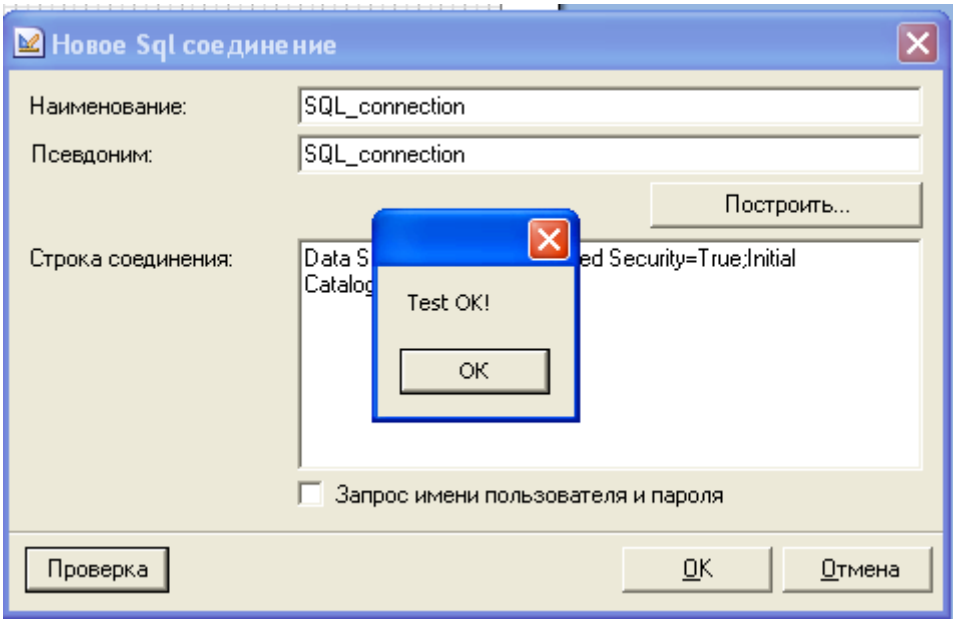


Рисунок 3-10

Соединение успешно установлено. Нажмем на ***ОК*** сообщения, и на ***ОК*** окна.

Соединения появилось в словаре (Рисунок 3-11):

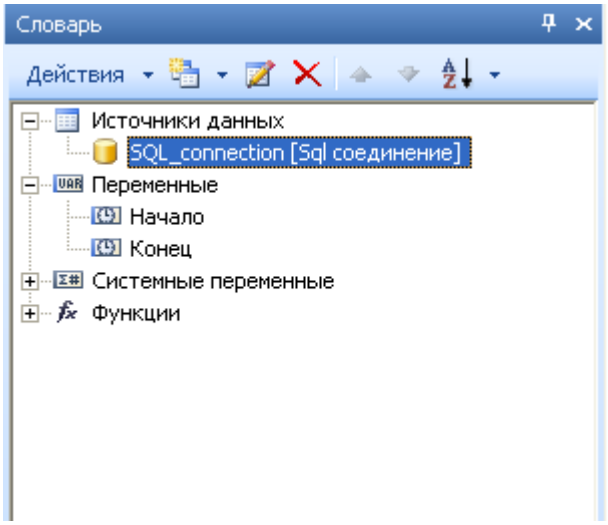


Рисунок 3-11

Теперь добавим *источник данных*. Вызовем *контекстное меню* и выберем пункт ***Данные из SQL соединения*** (Рисунок 3-12).

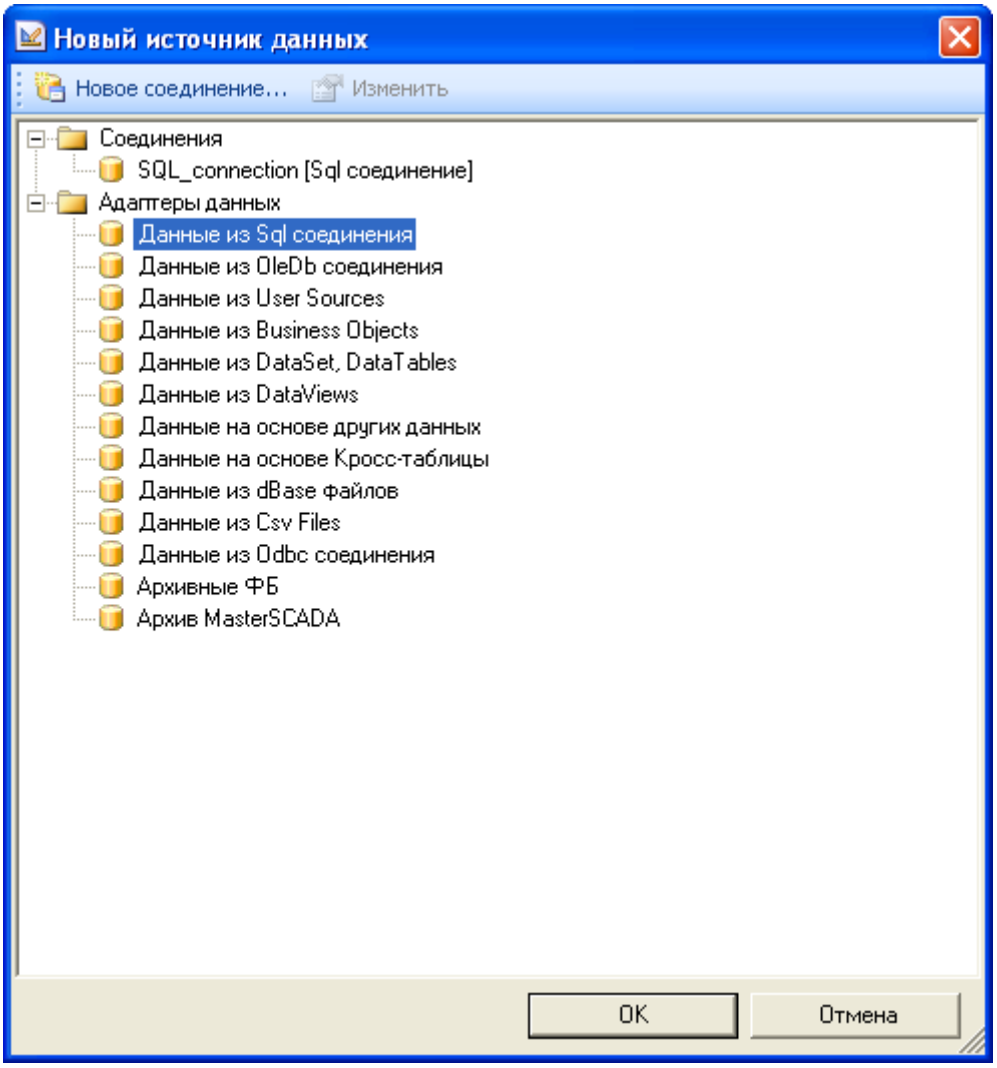


Рисунок 3-12

Появится окно настройки. В верхнем поле ***Наименование в источнике*** вводится имя SQL соединения, из которого необходимо получить данные. У нас соединение одно, поэтому оно прописалось в строку автоматически.

Введем имя данных – ***«Данные»*** (Рисунок 3-13).

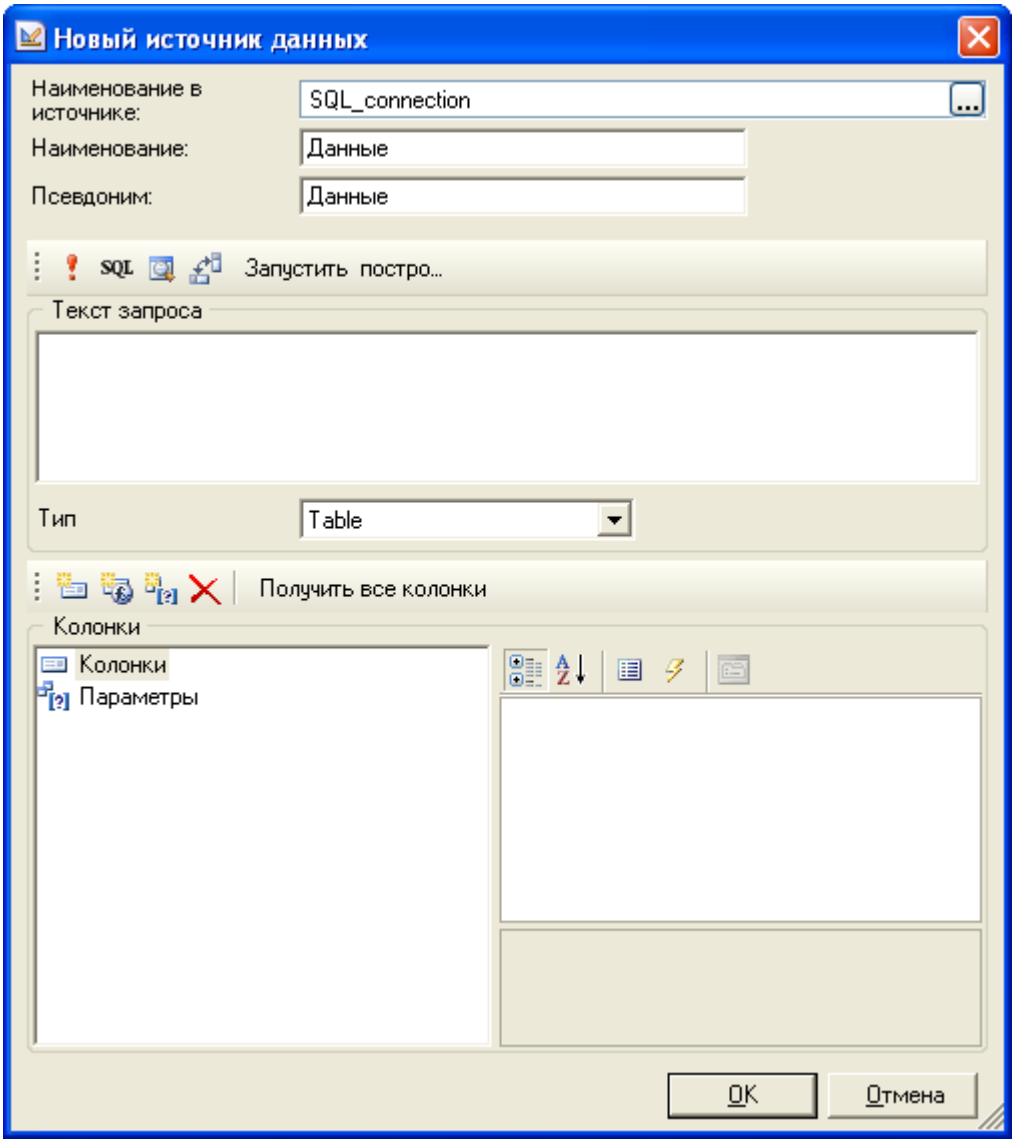


Рисунок 3-13

Теперь необходимо настроить получение и выборку данных. Для этого в поле ***Текст запроса*** введем следующий SQL запрос:

Select*Frommy_table

Этот запрос выберет из базы все колонки. Нажмем на *кнопку с восклицательным знаком* (выполнить запрос) (Рисунок 3-14).

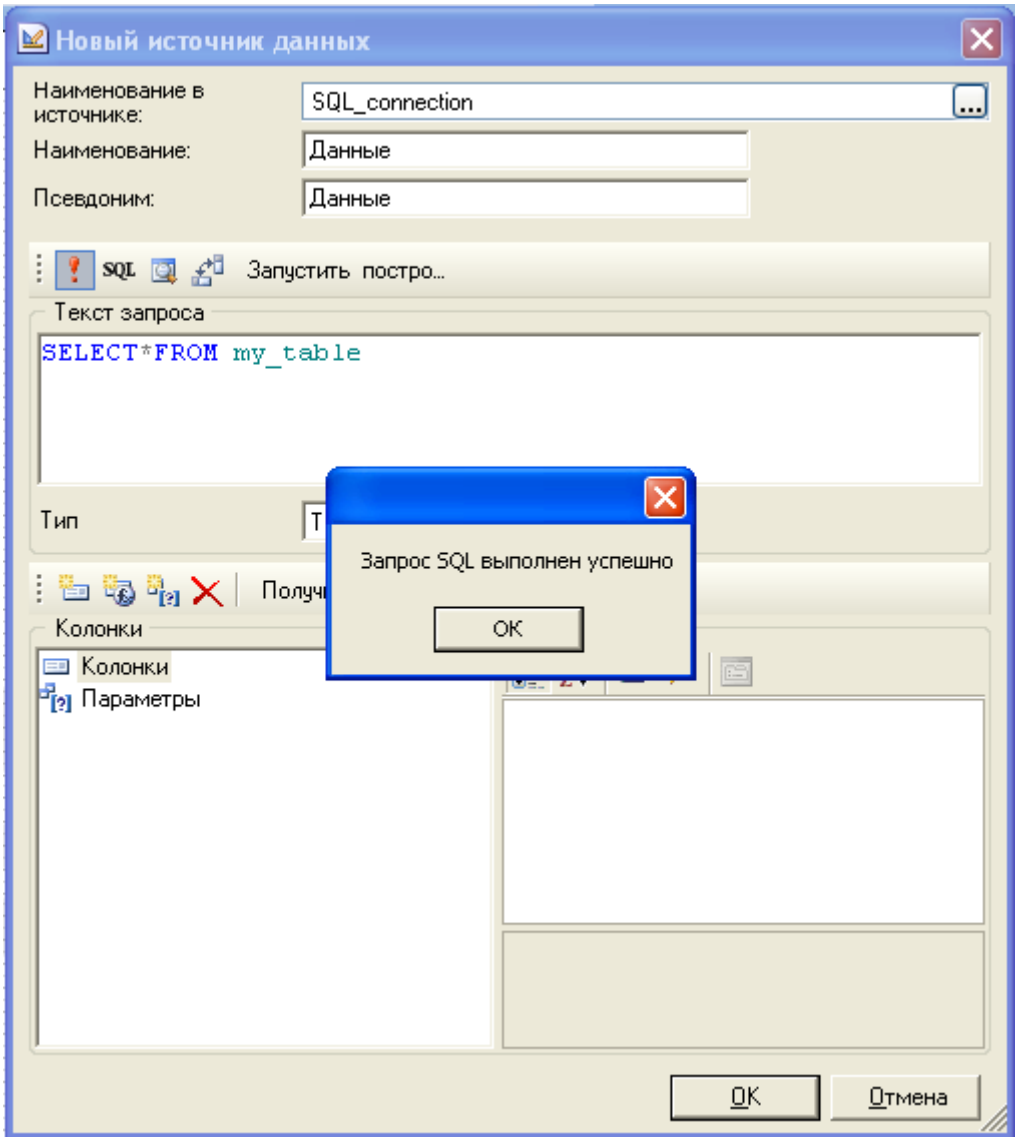


Рисунок 3-14

Нажмем на ***ОК*** сообщения, а затем нажмем на кнопку: ***Получить все колонки***. Колонки добавятся в окно (Рисунок 3-15)

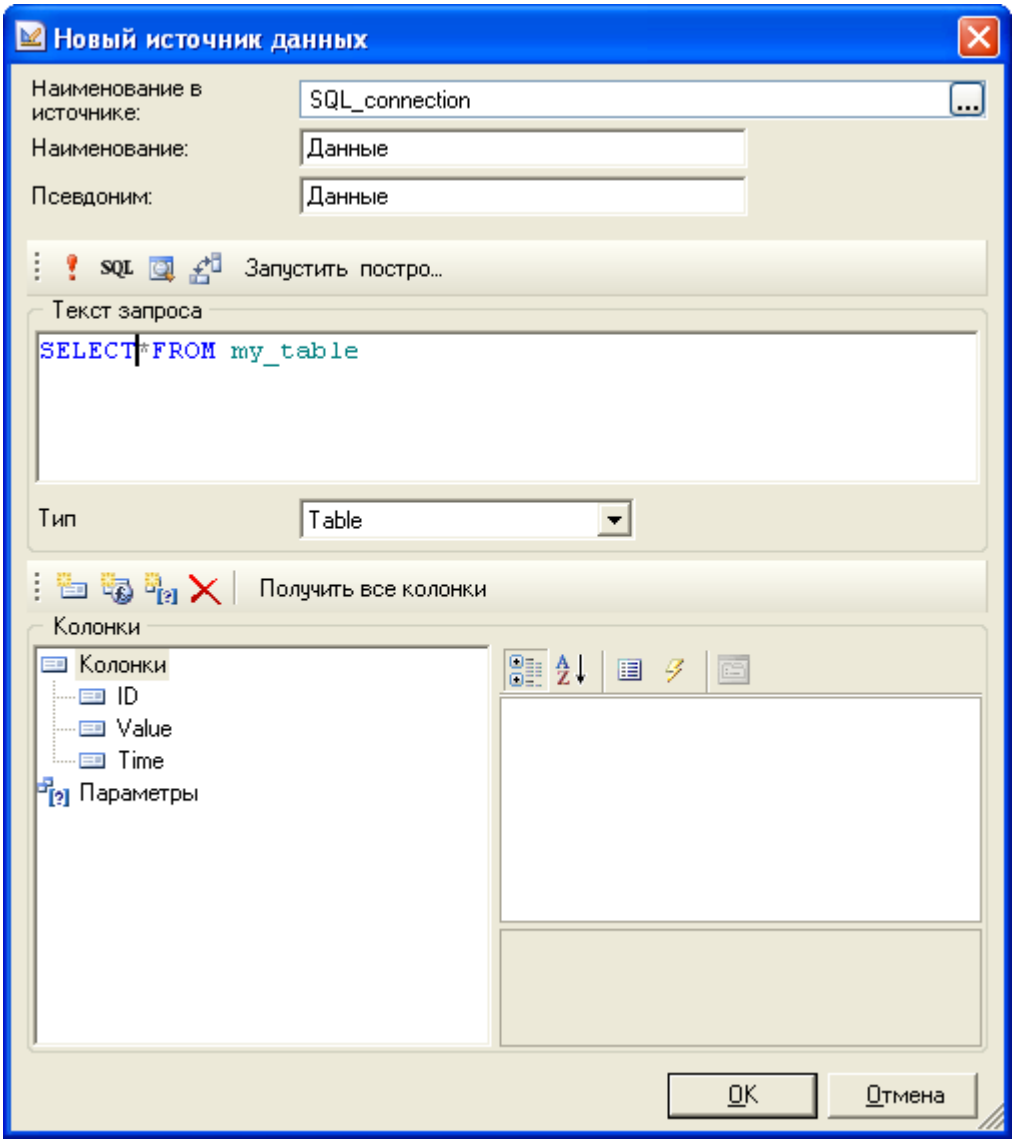


Рисунок 3-15

Нам необходимо ограничить выборку при помощи двух параметров времени – ***«начало»*** и ***«конец»***. Добавим эти параметры.

Для этого щелкнем правой кнопкой мыши по объекту ***«Параметры»*** и выберем пункт ***Новый параметр*** (Рисунок 3-16)

.

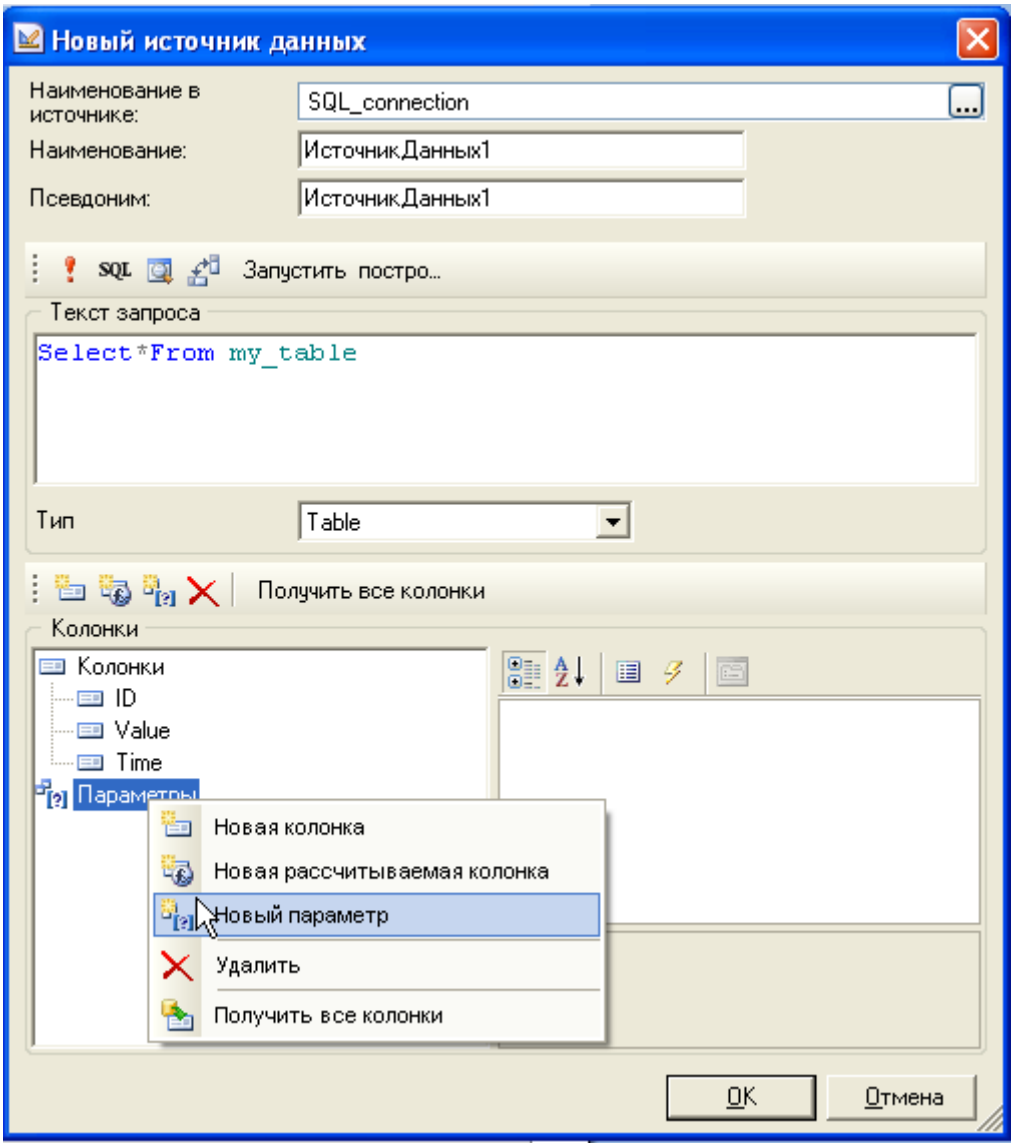


Рисунок 3-16

Зададим имя параметру – ***«Начало»*** и тип – *datetime* (Рисунок 3-17).

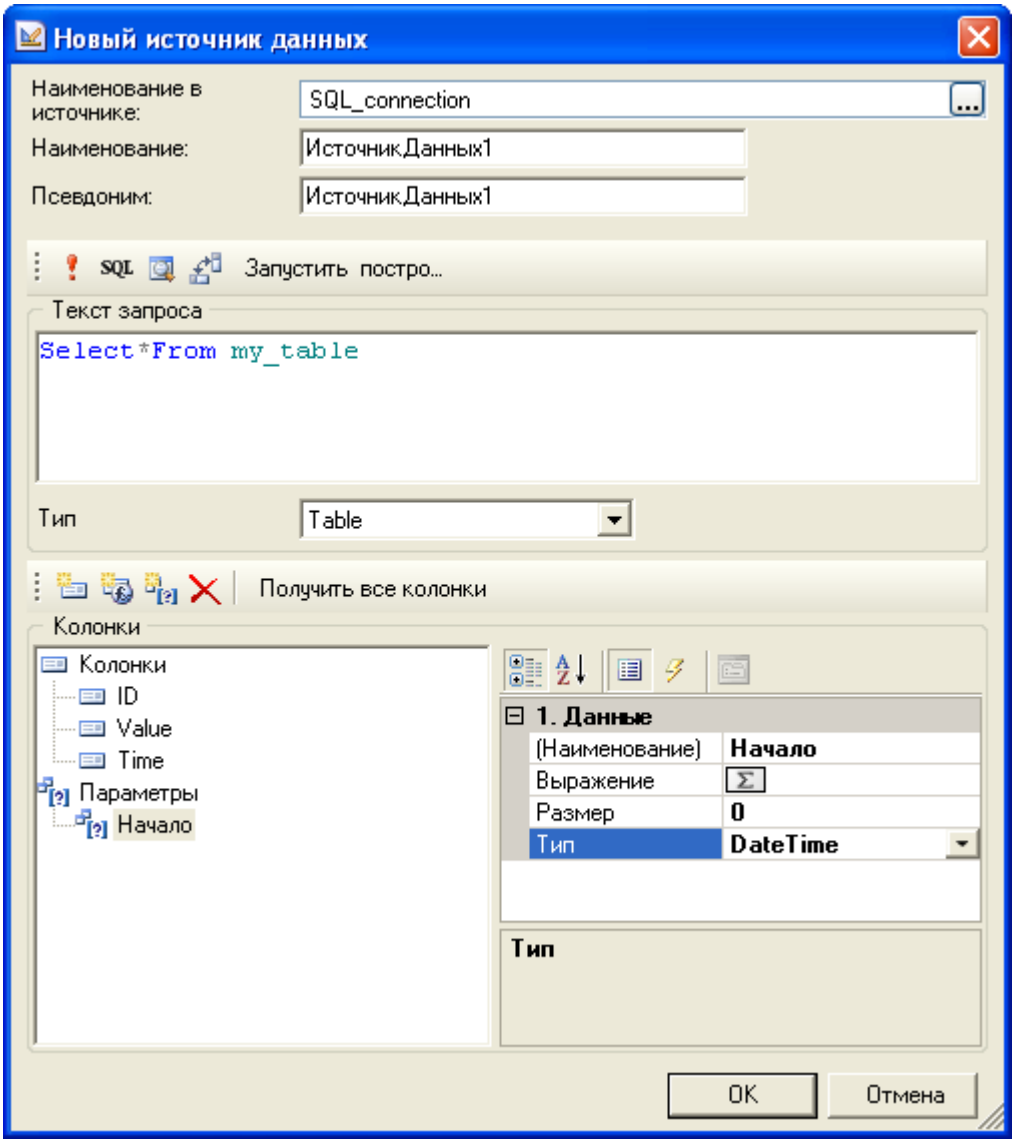


Рисунок 3-17

Аналогично создадим *параметр* ***«Конец»***.

Теперь нам необходимо построить запрос выборки при помощи этих двух параметров. Для того чтобы использовать в запросе параметр, перед его именем нужно поставить символ ***@***.

Подправим наш запрос:

Select*From my_table where Time>@Начало and Time<@Конец

Этот запрос будет выбирать только те значения, время которых больше параметра ***«Начало»,*** и меньше параметра ***«Конец»*** (Рисунок 3-18).

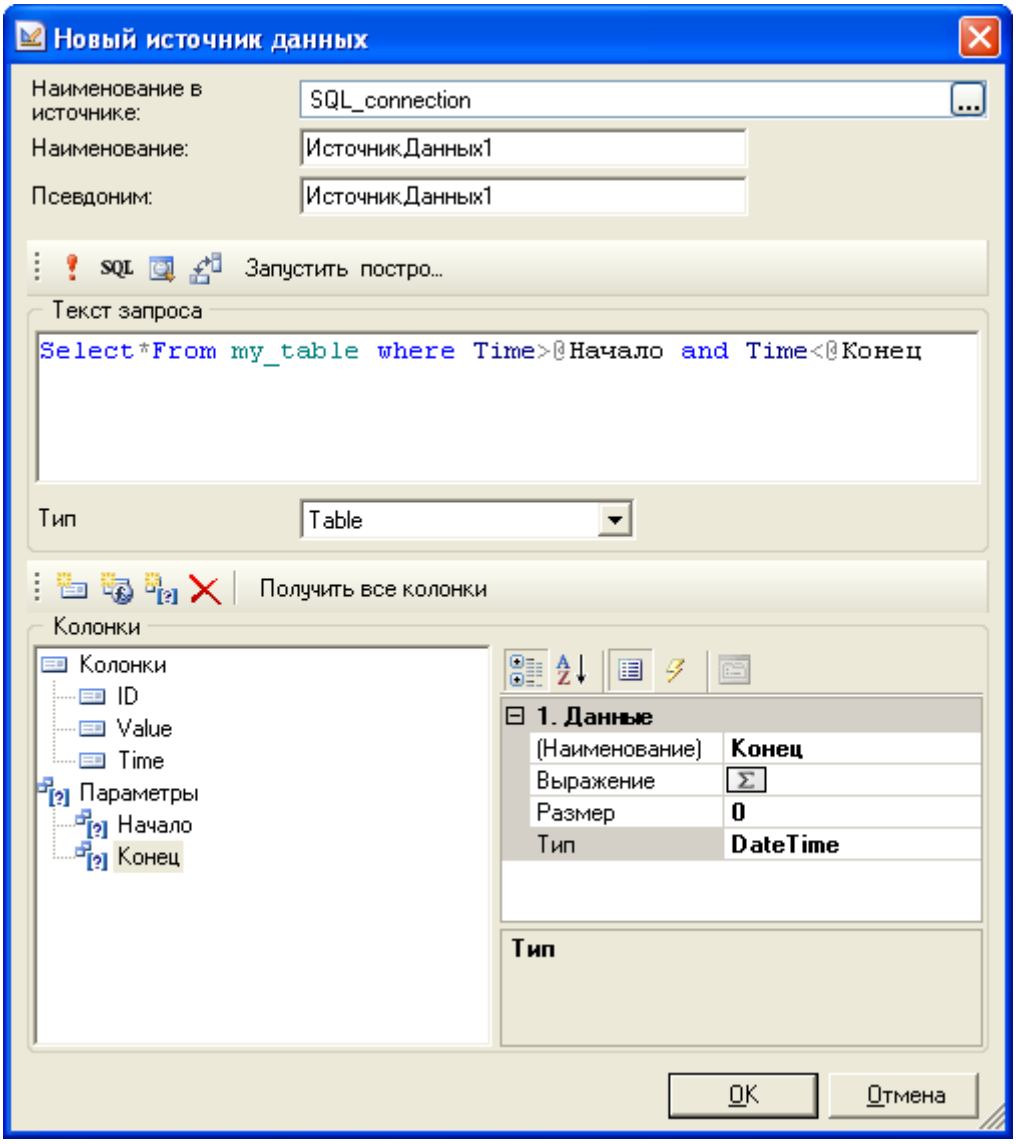


Рисунок 3-18

Можно удостоверится в правильности составления запроса и снова нажать на кнопку: ***Выполнить запрос***.

После этого нажмем ***ОК***.

В словарь добавился новый источник, с тремя колонками и двумя параметрами (Рисунок 3-19).

.

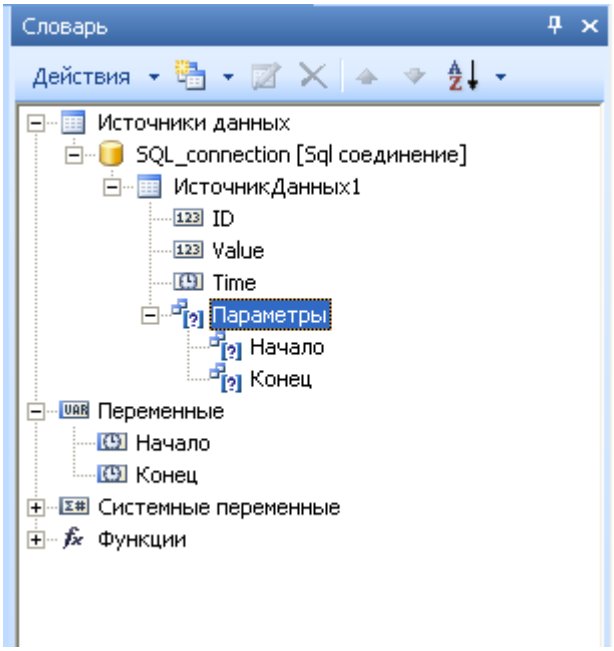


Рисунок 3-19

Теперь присвоим параметрам значения переменных.

Для этого выберем параметр ***«Начало»*** и перейдем на *закладку* ***Свойства***. Щелкнем мышью в поле ***Выражение*** и нажмем на кнопку с тремя точками (Рисунок 3-20).

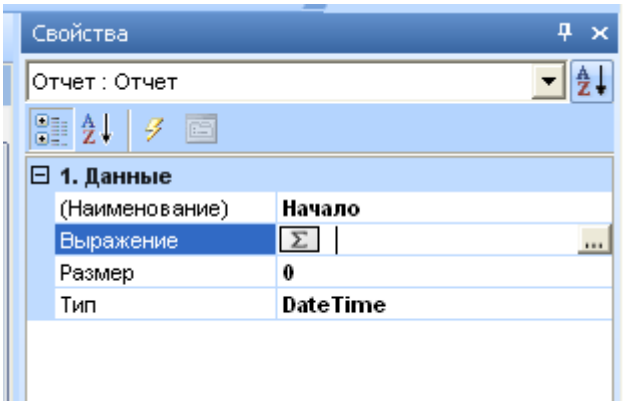


Рисунок 3-20

В появившемся окне введем имя переменной ***«Начало»*** (можно перетянуть ее из объекта ***Переменные***), и нажмем ***ОК*** (Рисунок 3-21).

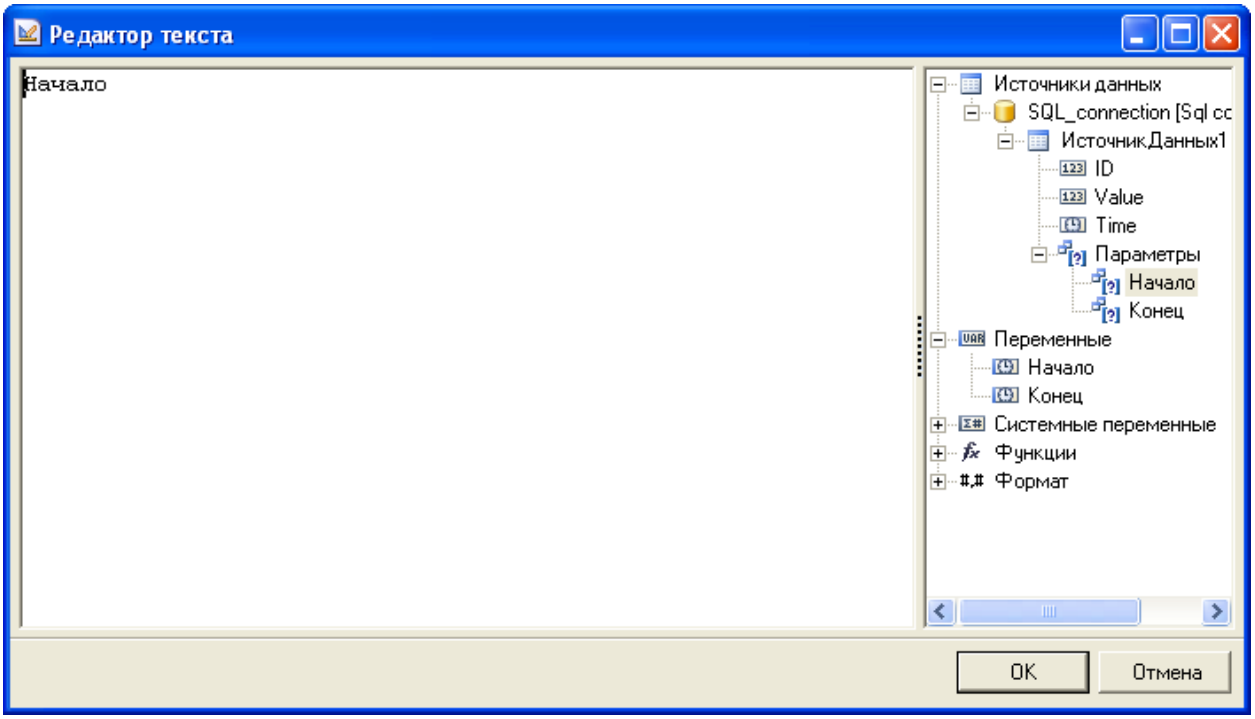


Рисунок 3-21

Проделаем аналогичные действия с параметром ***«Конец»***.

Теперь перетащим источник на рабочее поле страницы – создастся *бэнд**«Данные»***, в окне настроек (Рисунок 3-22) укажем нужные переменные и их порядок отображения. Нажмем: ***ОК***.

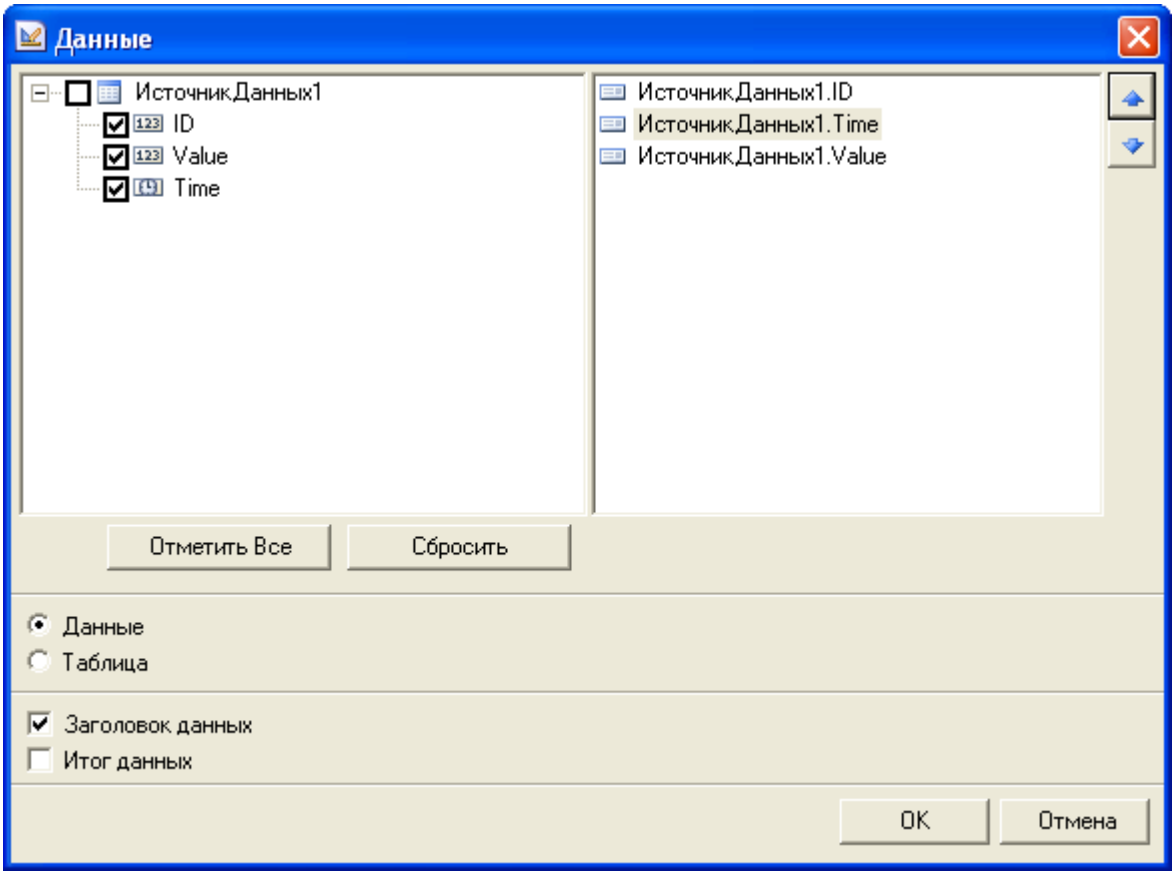


Рисунок 3-22

Отформатируем поля – зададим выравнивание по центру, более крупный шрифт и сделаем у полей границу (Рисунок 3-23).

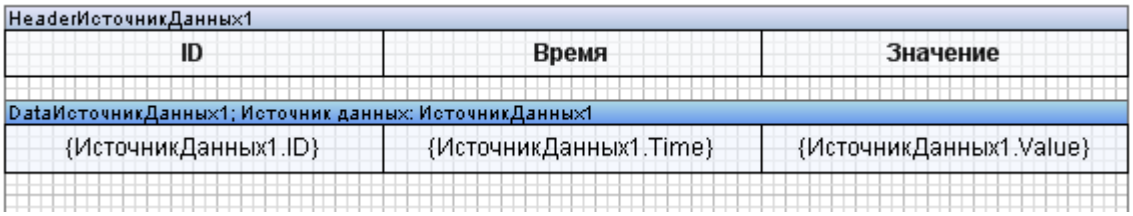


Рисунок 3-23

Теперь запустим предварительный просмотр и посмотрим на результат (Рисунок 3-24).

ID	Время	Значение
3	24.10.2010 17:30:36	18
4	24.10.2010 17:40:36	34
5	24.10.2010 17:50:36	71
6	24.10.2010 18:00:36	20
7	24.10.2010 18:10:36	23
8	24.10.2010 18:20:36	28

Рисунок 3-24

Данные выбраны из базы данных и отфильтрованы.