Пример создания модуля «с нуля» (Быстрый старт)

Задача, которую преследует данный документ, - дать необходимый минимум сведений, но, возможно, самых главных, которых достаточно для того, чтобы сесть и написать свое первое приложение на RP-платформе. Уже, обладая первичным опытом разработки, значительно легче и продуктивнее углубляться в детальное изучение нужных разделов общей документации и особенностей встроенного языка RP Server.

Мы попытаемся решить данную задачу на фоне достаточно простого примера. В рамках этого, примера мы не будем претендовать на обучение созданию «промышленного» ПО, т.к. это обычно требует детальной проработки бизнес процессов, интерфейса пользователя, логики работы программы и схемы данных.
В данном документе мы дадим иллюстрацию того как на основе RP платформы можно в кратчайшие сроки, буквально минут за 30 создать полностью работающее клиент-серверное приложение, выполняющее вполне реальную прикладную функцию и автоматически предоставляющее развитый набор сервисов конечным пользователям.

Итак, какова, например, несложная бизнес задача, требующая срочной автоматизированной поддержки?
Предположим, мы имеем дело с деловыми партнерами, для которых производим отгрузку продукции или для которых выполняем какие-либо работы. Т.е. для нас они являются заказчиками. Нам требуется срочно обеспечить автоматизированный учет наших заказов. И, мы помним, что у нас есть в запасе полчаса (☺) на реализацию.

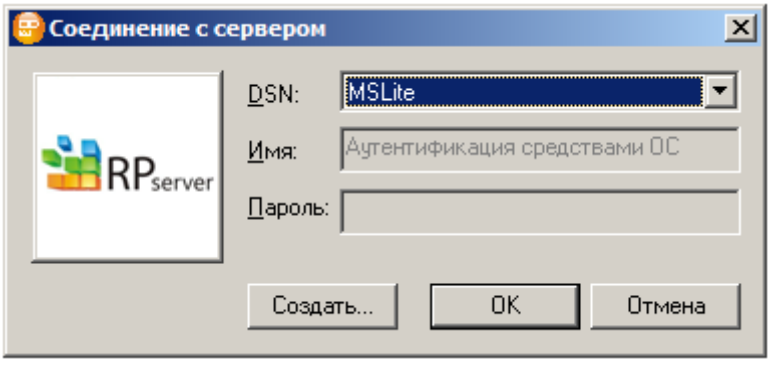
Приступим к работе….

I. Создание новой базы данных.

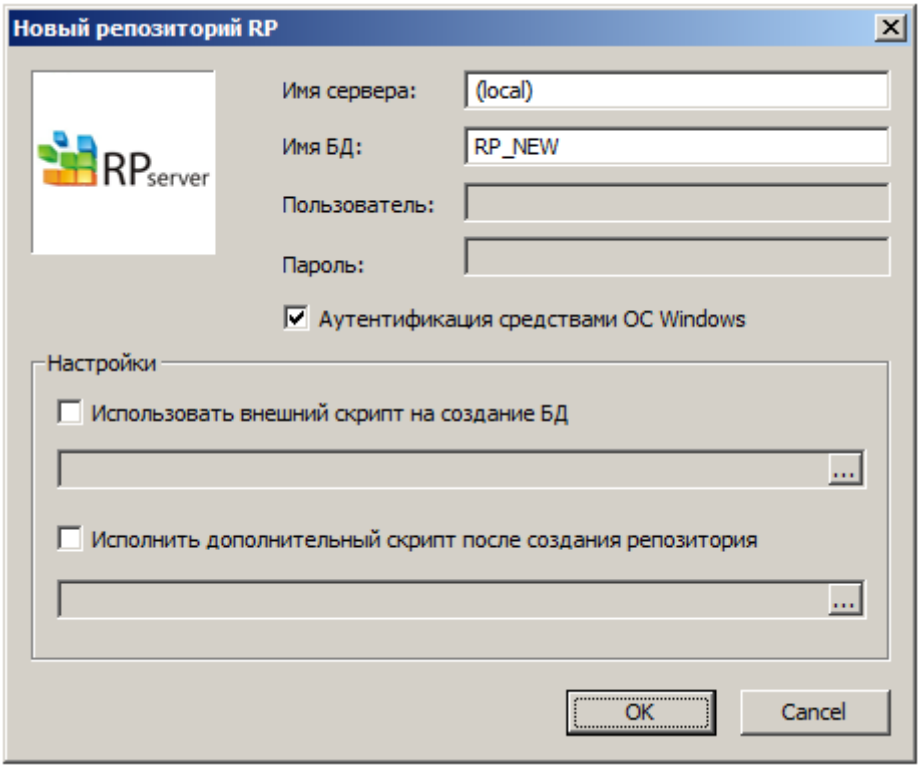
Если у вас еще не создана база данных для проведения разработки, то сделаем это прямо сейчас. Если ODBC источник и база данных уже созданы, то можно сразу перейти к пункту II.

Запускаем ДИЗАЙНЕР:

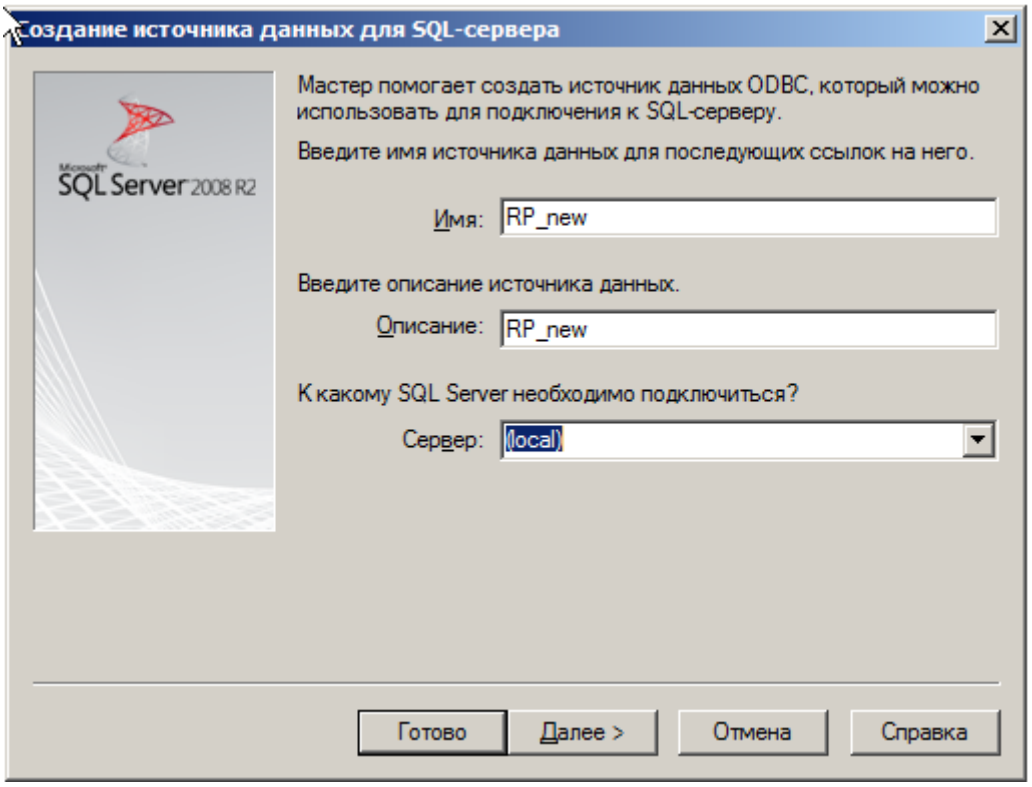
В появившемся диалоге нажимаем пункт «Создать»:



В открывшемся диалоге указываем имя нашего Microsoft SQL Server и имя базы данных, которую нужно создать. Остальные поля оставляем по умолчанию.
Если SQL Server установлен локально на том же компьютере, где запускается RP Designer, и только в одном экземпляре, то имя сервера может быть указано как (local), либо должно быть прописано одно из имен доступных SQL серверов.
Нажимаем «OK».

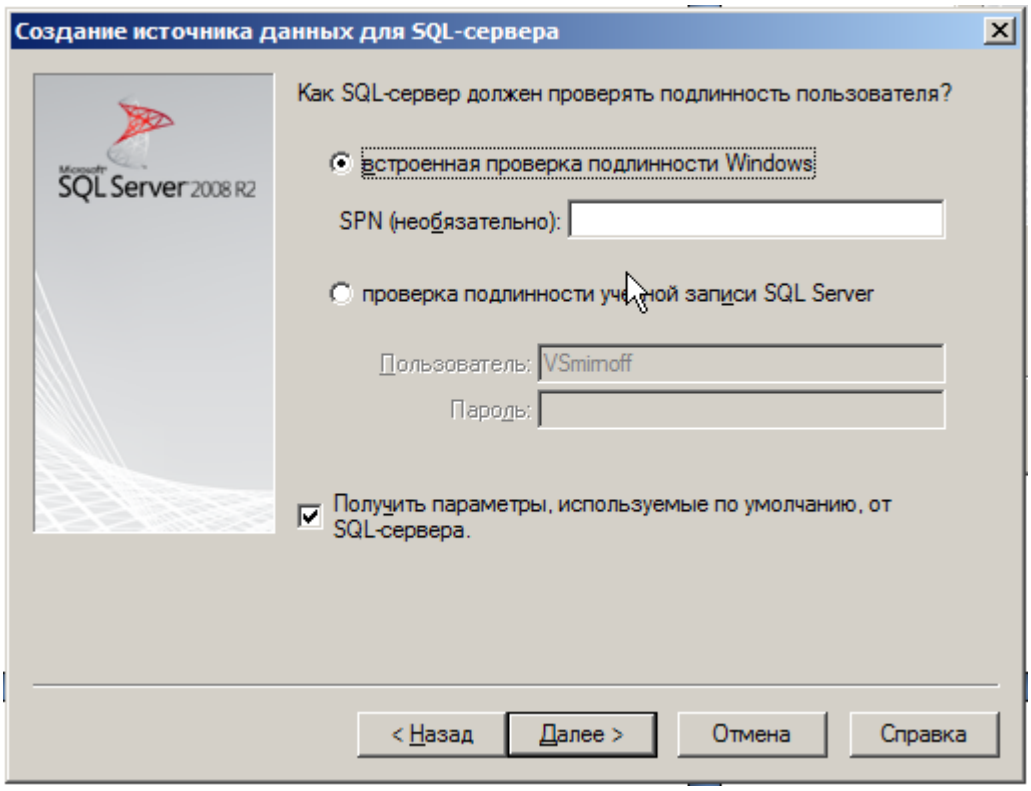


В следующем диалоге указываем название ODBC источника, через который мы будем работать с нашей базой данных.

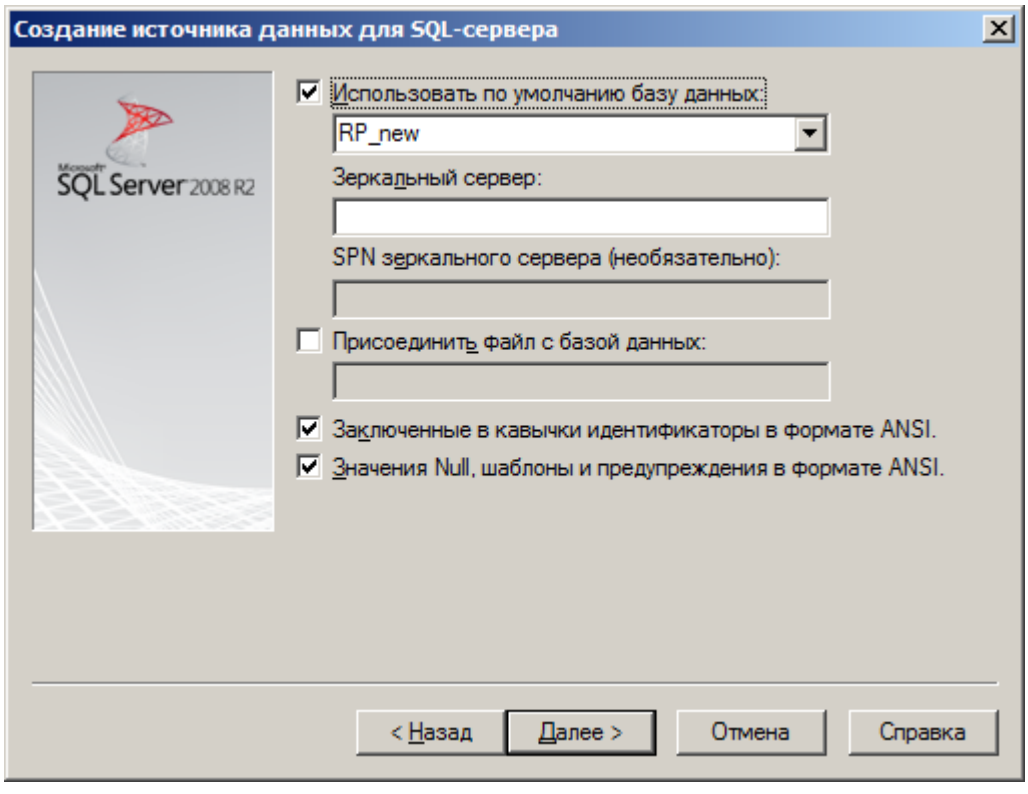


Имя ODBC источника может быть любым, в том числе может совпадать с именем базы данных. Нажимаем «Далее».

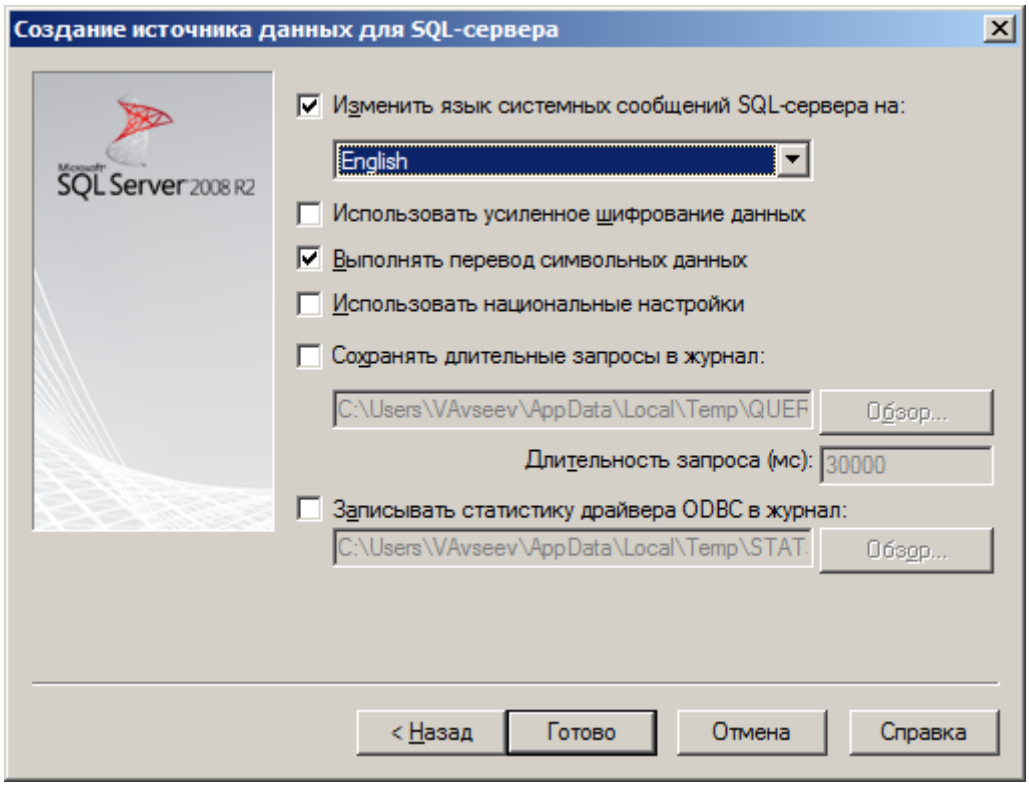
В следующем диалоге оставляем поля по умолчанию и нажимаем «Далее».



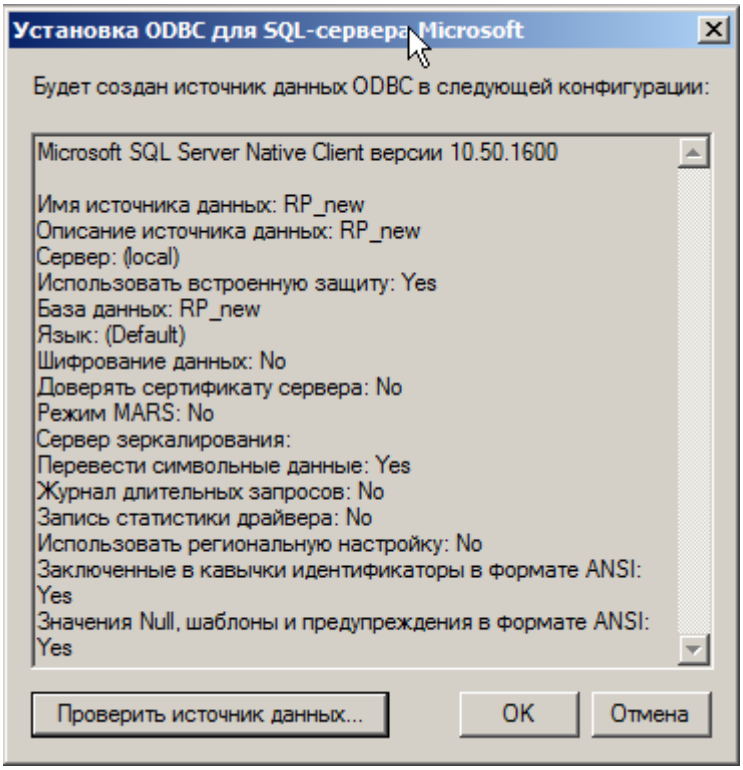
После этого выбираем нашу базу данных как базу по умолчанию и нажимаем «Далее».



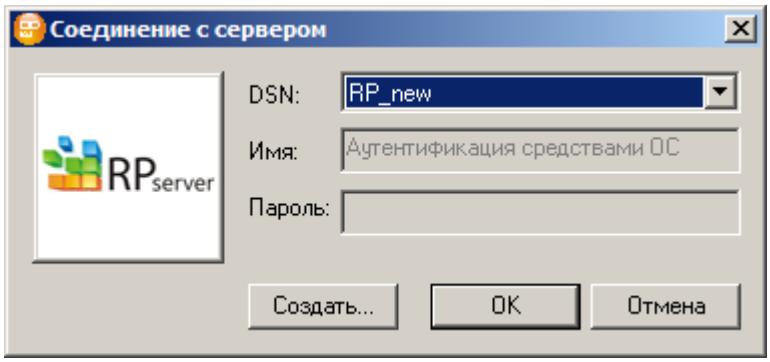
В этом диалоге можно указать, язык системных сообщений сервера. Лучше поднять флаг «Изменить язык…» и указать English. Нажимаем «Готово».



И, наконец, в последнем диалоге мастер настройки показывает нам все установленные параметры. Можно сразу нажать на «ОК», можно проверить источник данных.



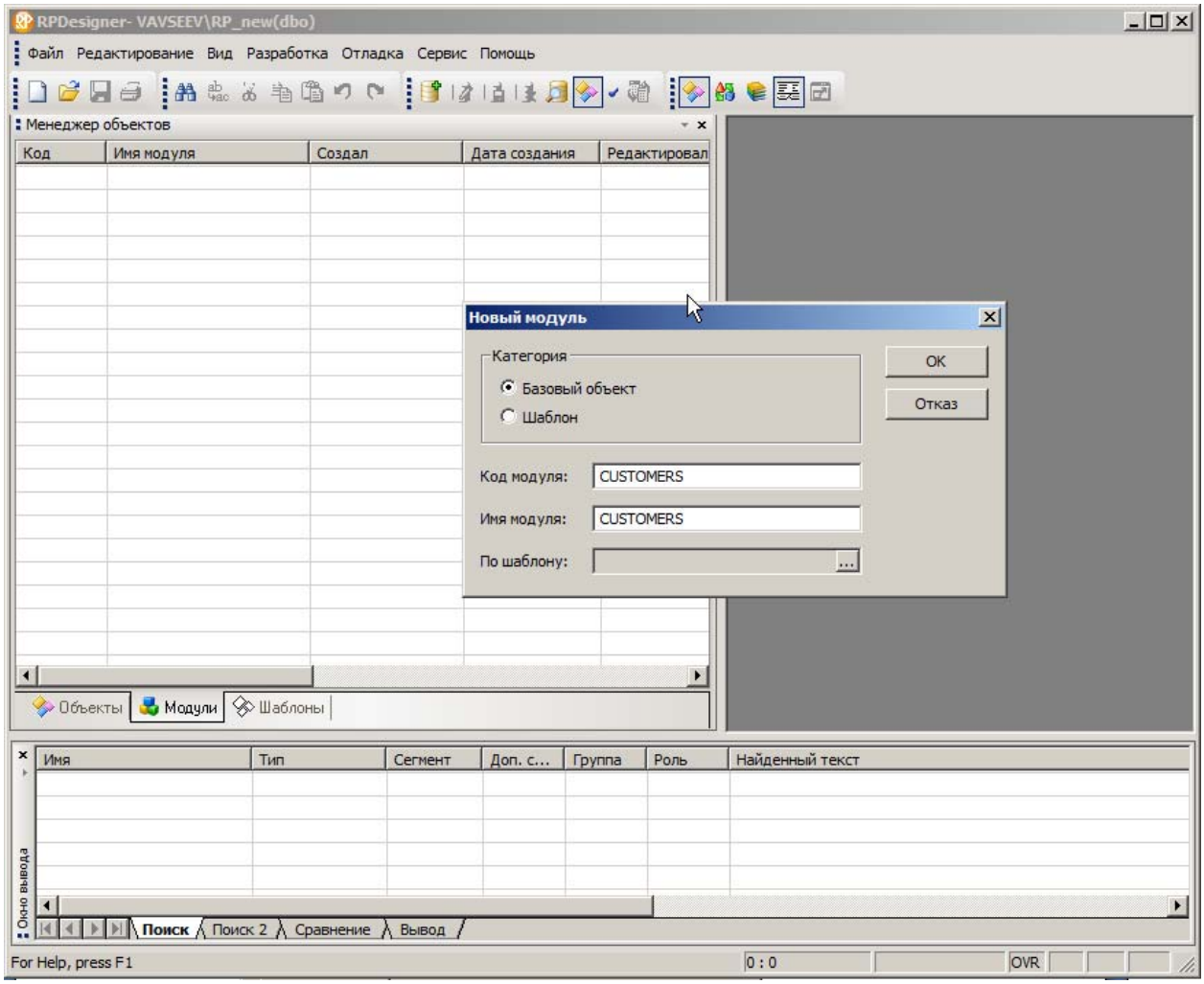
После этого программа снова предлагает диалог соединения с сервером, в котором теперь можно выбрать только что созданное нами ODBC соединение и нажать ОК:



Откроется среда разработки RP Designer.

II. Создание конструктора модуля.

Модулем называется любое приложение на RP платформе. Модуль – это совокупность объектов (процедур, списков, диалогов, отчетов и т.п.), предназначенных для выполнения какой-либо единой задачи создаваемой прикладной системы. Как правило, все эти объекты объединены одним общим меню пользователя.
В среде RP Designer выбираем закладку «Модули» и нажимаем <Ins> на клавиатуре или через правую кнопку меню. Раз мы собираемся автоматизировать заказы, то назовем модуль «CUSTOMERS».

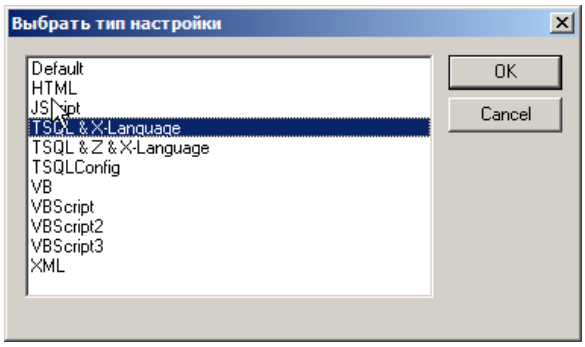


Теперь вызываем созданный модуль на редактирование и в секции «Конструктор» вписываем строку:

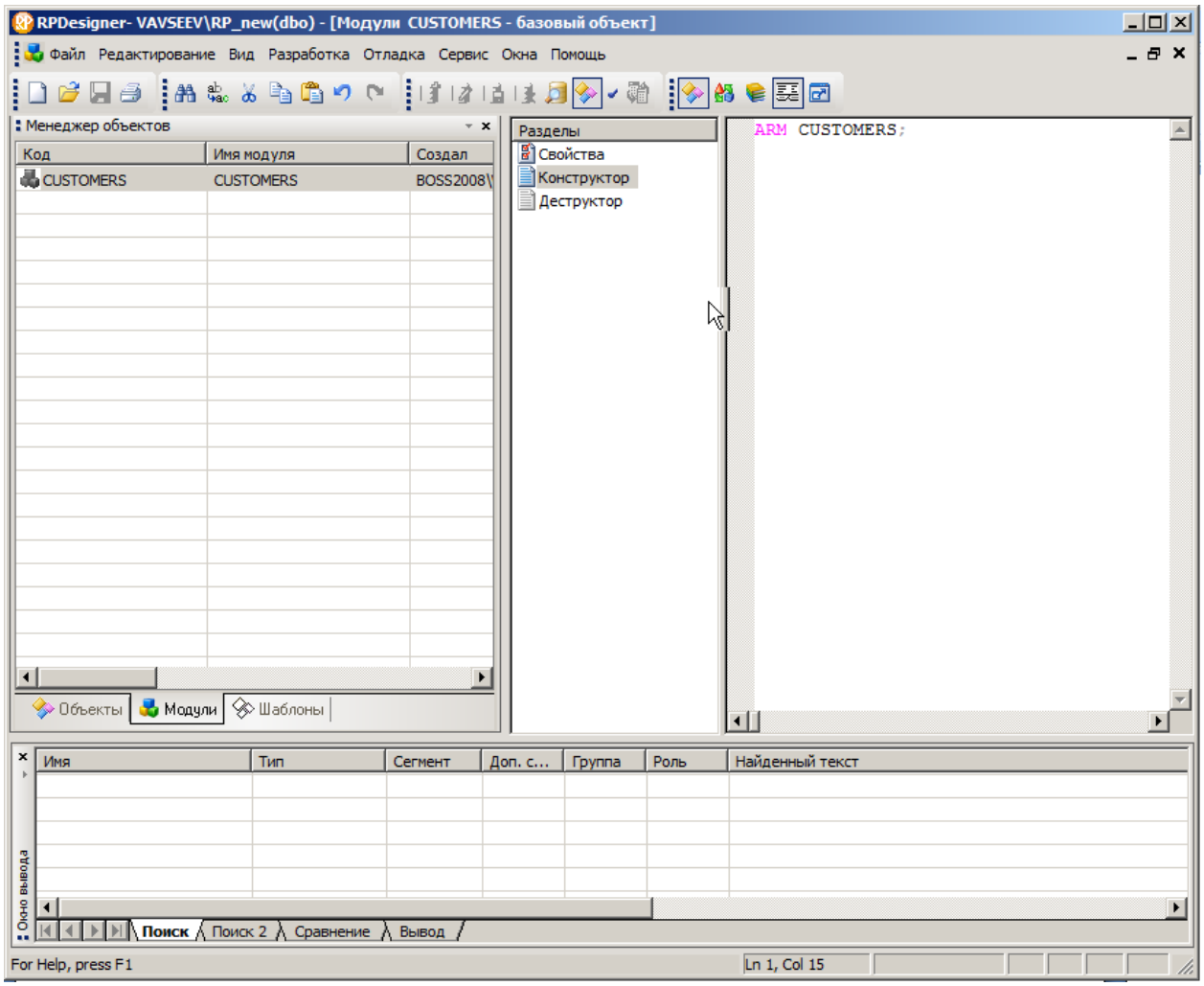
```
ARM CUSTOMERS;
```

На самом деле конструктор модуля – это специальная процедура, которая выполняется при запуске модуля на исполнение и она может содержать код любой степени сложности на внутреннем языке RP платформы (X-языке). Но в нашем случае мы всего лишь указали, что хотим вызывать меню модуля, которое называется «CUSTOMERS».

При входе в редактор конструктора модуля может появиться вот такой диалог с вопросом о том, какой тип настройки редактора нам требуется:



Здесь и далее можно смело выбирать вариант «TSQL & X-Language».

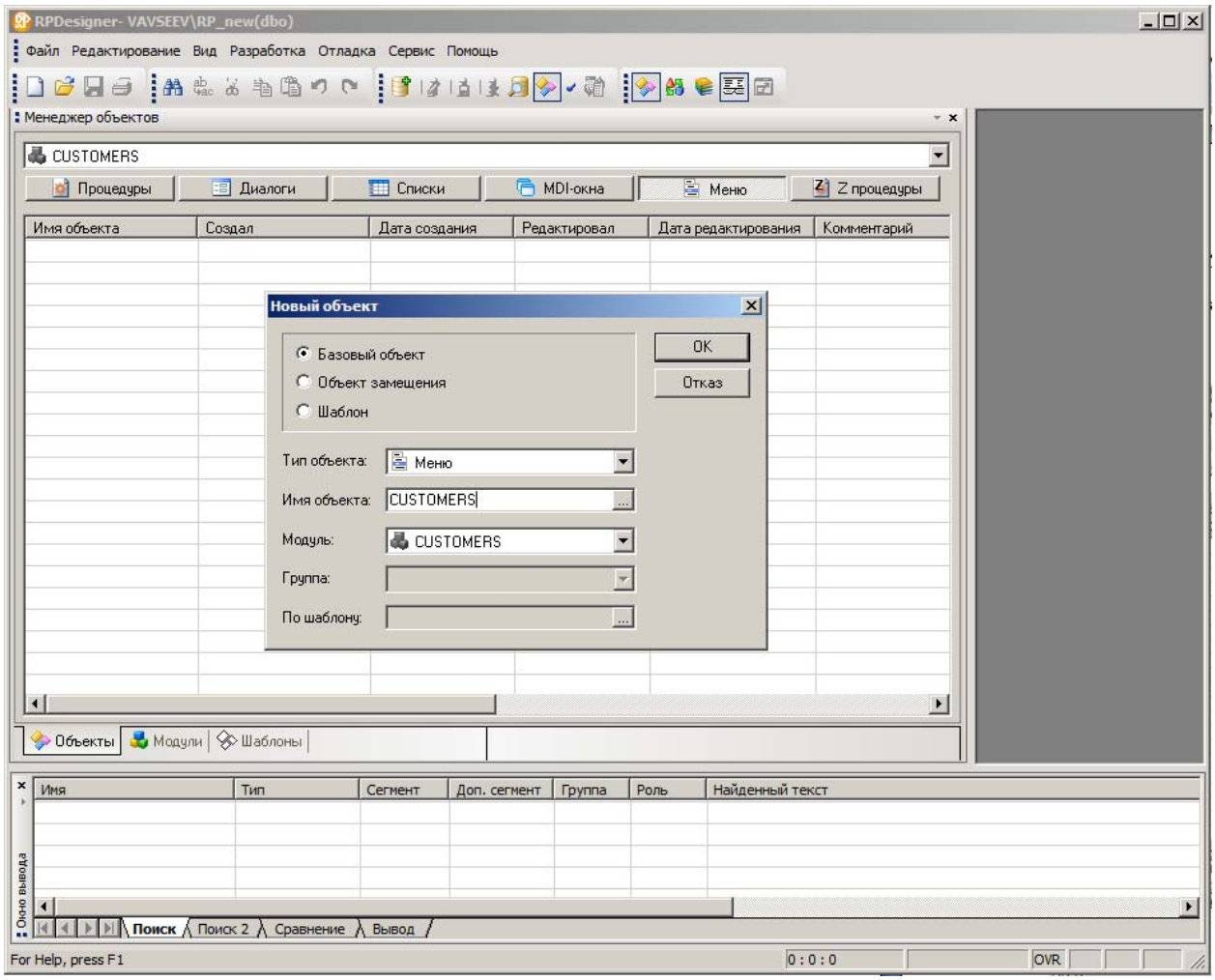


Сохраняем созданный модуль, нажав на иконку  .

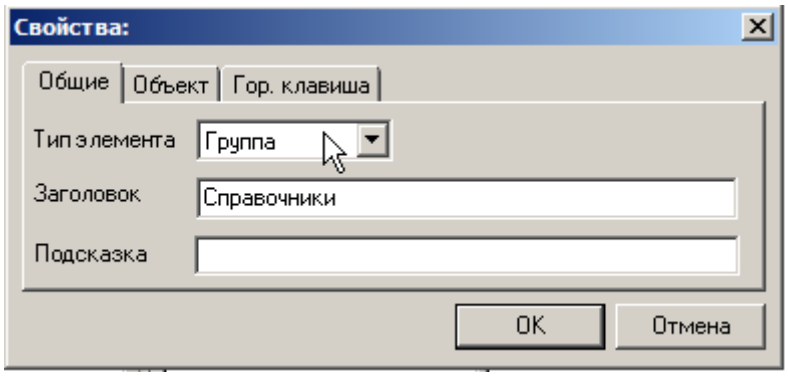
III Создание меню.

Переключаемся на закладку «Объекты» и выбираем тип объекта «Меню». В верхней части окна «Менеджер объектов» указываем, что текущий модуль – CUSTOMERS.
Нажимаем <Ins>.

!!! Обратите внимание.
Новый объект автоматически привязывается к тому модулю, который указан как текущий. Для удобства работы все создаваемые нами объекты лучше сразу привязывать к модулю CUSTOMERS. Хотя, это можно сделать и позже, например, по правой кнопке мыши в списке объектов выбрать пункт «Включить в модуль».

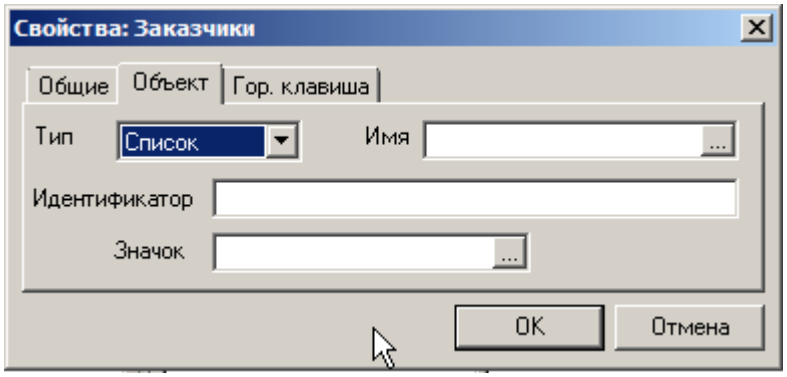


Создаем на верхнем уровне меню 2 пункта – «Справочники» и «Данные»



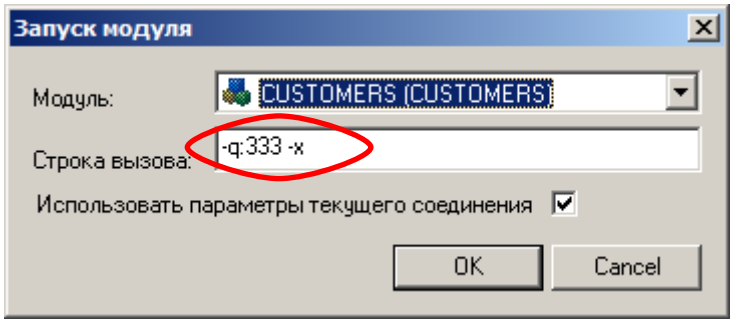
Теперь добавим выпадающие пункты меню, как показано на рисунках.

При создании выпадающего меню программа попросит указать тип элементов. Пока можно указать, что тип – «Список»:



Сохраняем созданное таким образом двухуровневое меню, нажав на иконку  .

С этого момента созданный модуль доступен для запуска из среды RP Designer с целью отладки. Нажимаем <F7> и говорим «ОК» в появившемся диалоге:



Обратите внимание на параметры в строке вызова. Их следует указать как отмечено красным кружком.

В запустившемся приложении мы увидим сформированное нами меню. Можно даже понажимать на его пункты. Но, как и следует ожидать, они пока пустые.

Выходим из запущенного приложения.

IV. Создаем справочник «Заказчики»

Для начала надо создать таблицу базы данных:

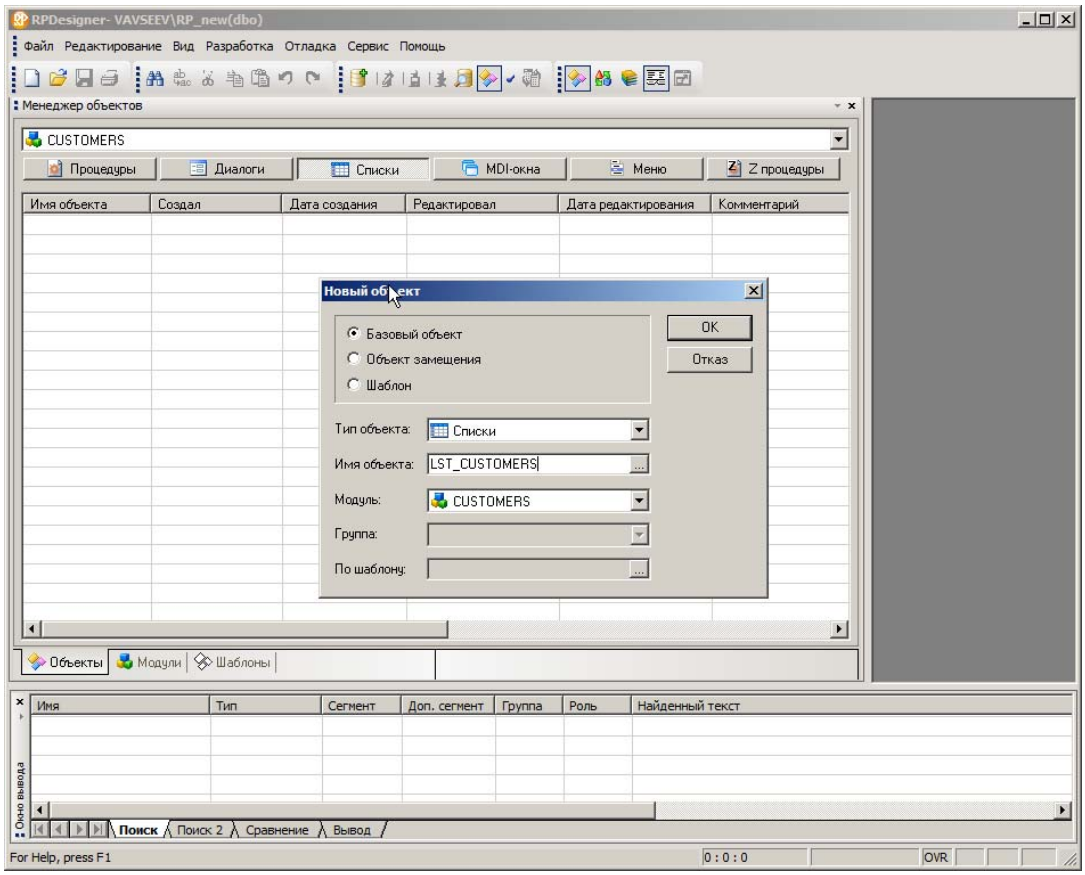
```
CREATE TABLE CU_CUSTOMERS
(
PRIMARY KEY (ID),                       -- первичный ключ
ID INTEGER IDENTITY(1,1),
KOD VARCHAR(25),                        -- код
INN VARCHAR(25),                        -- ИНН
NAME VARCHAR(255),                      -- наименование
ADR VARCHAR(255),                       -- адрес
ACC VARCHAR(25),                        -- счет в банке
BANK VARCHAR(255)                       -- банк
)
GO

GRANT ALL ON CU_CUSTOMERS TO public
GO
```

!!! Обратите внимание.
Таблицу в БД можно создать несколькими способами, в том числе используя соответствующие интерфейсы SQL Server Management Studio (в составе утилит Microsoft SQL Server).
Так же создавать таблицы можно и не выходя из RP Designer, например, в режиме отладки приложения. Для этого нужно запустить любое приложение, а пока оно у нас единственное (клавиша <F7>, см. предыдущий раздел) и нажать комбинацию клавиш SHIFT-F1. При этом появляется дополнительное окно отладки, предназначенное в том числе для выполнения любого скрипта. В это окно просто переносим (вставляем) указанный выше код на создание таблицы CU_CUSTOMERS, выделяем его целиком и выполняем по клавише <F8>. Теперь можем закрыть и это окно отладки, и запущенное приложение.

В БД появилась соответствующая таблица.

Переключимся в менеджере объектов среды RP Designer на закладку «Списки» и добавим по клавише <Ins> список, который назовем, например, «LST_CUSTOMERS».



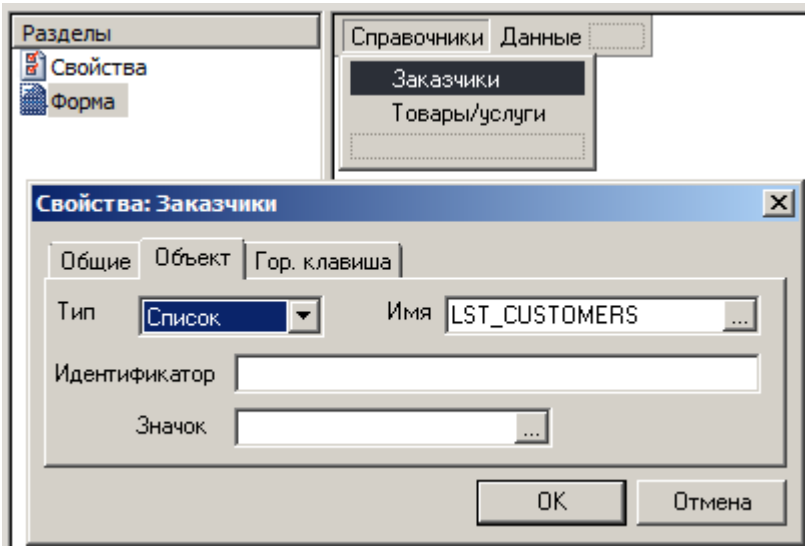
В секции «SQL-запрос» создаваемого списка напишем вот такой несложный оператор SELECT, позволяющий извлечь данные из созданной только что таблицы. Лучше сразу дать этому запросу имя, например, @LST_CUSTOMERS, чтобы потом можно было свободно обращаться к этому «поименованному» запросу (строго говоря, конечно, это имя не запроса, как такового, а того буфера, куда попадут данные, возвращаемые этим запросом).

```
@LST_CUSTOMERS
SELECT id, kod[]"Код", inn[]"ИНН", name[]"Наименование", adr[]"Адрес", acc[]"Счет", bank[]"Банк"
  FROM CU_CUSTOMERS
ORDER BY kod;
```

Сохраним созданный список, нажав на иконку  .

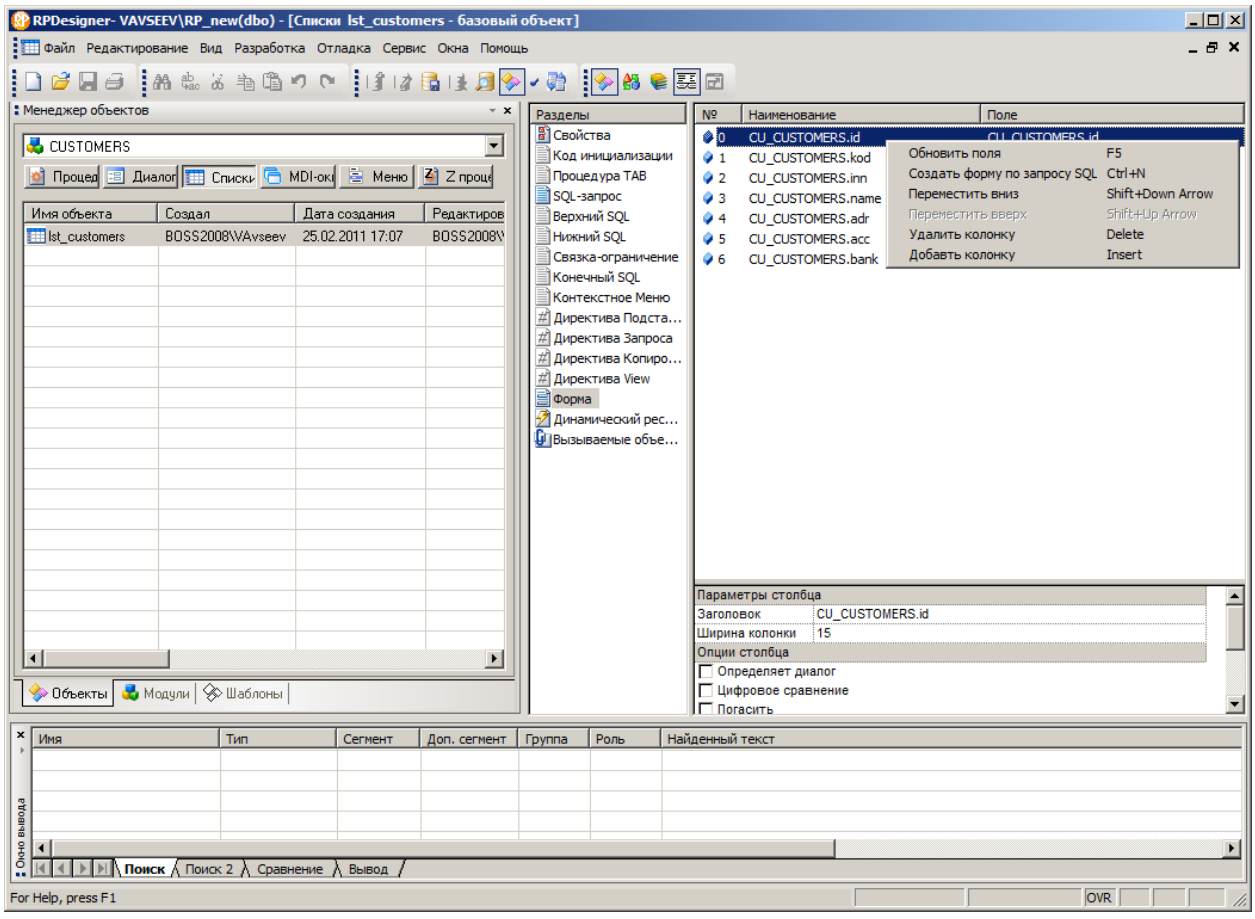
!!! Обратите внимание.
Если в данный момент запустить наше приложение (по клавише <F7>), то непосредственно в среде RP Designer автоматически становятся доступны средства отладки. Например, выделив введенный SQL запрос (непосредственно в секции «SQL-запрос» списка) и нажав на клавишу F5 мы получим автоматически сформированный список с содержимым таблицы CU_CUSTOMERS или сообщение об ошибке, если запрос написан не правильно.

Теперь нужно связать созданный список с пунктом меню.
На закладке «Меню» выбираем наше меню и вызываем его на редактирование.
В секции диалога редактирования меню «Форма» выбираем пункт меню «Заказчики» и привязываем объект список LST_CUSTOMERS:



Сохраняем изменения в меню, нажав на иконку  .

По клавише <F7> проверим работу созданного объекта. Он должен вызываться при нажатии на пункт меню «Заказчики». Обратите внимание, что все столбцы (кроме ID) имеют русские названия. Это произошло потому, что мы их прописали непосредственно в SQL запросе списка. А вот столбец ID следует просто погасить. Сделать это можно так: в диалоге редактирования списка заходим в секцию «Форма» (приложение при этом запущено), встаем на список полей и нажимаем Ctrl-N. Встав на первое поле ID внизу в его свойствах, поднимаем флажок «Погасить».



Сохраняем изменения в меню, нажав на иконку  .

Сразу же после этого в окне нашего приложения (даже не закрывая его) можно заново запустить список и убедиться, что в нем отразилось внесенное изменение. Поле ID погашено.

V. Создаем справочник «Товары/услуги»

Создадим таблицу товаров:

```
CREATE TABLE CU_GOODS
(
PRIMARY KEY (ID),                       -- первичный ключ
ID INTEGER IDENTITY(1,1),
KOD VARCHAR(25),                        -- код
NAME VARCHAR(255),                      -- наименование
PRICE NUMERIC(19,2),                    -- цена
MEASURE VARCHAR(25)                     -- единица измерения
)
GO

GRANT ALL ON CU_GOODS TO public
GO
```

В менеджере объектов среды RP Designer выбираем закладку «Списки» и добавим список «LST_GOODS».

В секции «SQL-запрос» этого списка напишем вот такой поименованный оператор SELECT, позволяющий извлечь данные из таблицы Товары/услуги:

```
@LST_GOODS
SELECT id, kod[]"Код", name[]"Наименование", PRICE[]"Цена", MEASURE[]"Е/и"
 FROM CU_GOODS
ORDER BY kod;
```

Аналогично предыдущему пункту подключим созданный список к пункту меню «Товары/услуги», а также создадим список полей этого списка и для первого поля ID отметим «Погасить».

Проверим его работу в приложении.

VI. Создаем список «Заказы».

Создадим таблицу товаров:

```
CREATE TABLE CU_ORDERS
(
PRIMARY KEY (ID),                          -- первичный ключ
ID INTEGER IDENTITY(1,1),
NAME VARCHAR(255),                         -- наименование
ORDER_DATE datetime,                       -- дата заказа
CUSTOMER_ID INTEGER,                       -- ссылка на покупателя
status INTEGER                             -- статус заказа
)
GO

GRANT ALL ON CU_ORDERS TO public
GO
```

На закладке «Списки» добавим список «LST_ORDERS».

Для секции TAB списка напишем такой код:

```
MSG 2041, @LST_ORDER_LINES;
```

Эта команда в будущем станет перерисовывать строки заказа при перемещении по списку заказов, когда мы реализуем список «Строки заказа».

В секции «SQL-запрос» списка напишем вот такой оператор SELECT:

```
@LST_ORDERS
SELECT CU_ORDERS.id, CU_ORDERS.name[]"Наименование заказа", ORDER_DATE[%x] "Дата",
CU_CUSTOMERS.name[]"Заказчик"
  FROM CU_ORDERS _HINTBROWSER JOIN CU_CUSTOMERS ON CU_CUSTOMERS.id =
CU_ORDERS.CUSTOMER_ID
  ORDER BY CU_CUSTOMERS.name, ORDER_DATE;
```

Подключим полученный список к пункту меню «Заказы». При этом не забудем создать перечень полей списка и для первого поля ID отметим «Погасить». Перезапустим приложение и проверим его работу.

VII. Создаем список «Строки заказа», связанный со списком «Заказы»

Создадим таблицу строк заказа:

```
CREATE TABLE CU_ORDER_LINES
(
PRIMARY KEY (ID),                                   -- первичный ключ
ID INTEGER IDENTITY(1,1),
ORDER_ID INTEGER,                                   -- ссылка на заказ
GOODS_ID INTEGER,                                   -- ссылка на товар/услугу
COUNT NUMERIC(19,2)                                 -- количество
)
GO

GRANT ALL ON CU_ORDER_LINES TO public
GO
```

На закладке «Списки» добавим список «LST_ORDER_LINES».

В секции «SQL-запрос» напишем:

```
@LST_ORDER_LINES
SELECT  CU_ORDER_LINES.ID,
                ORDER_ID,
                CU_GOODS.NAME[]"Товар",
                CU_GOODS.PRICE[%12.2f] "Цена",
                CU_GOODS.MEASURE[]"Ед. Изм. ",
                COUNT[%12.2f] "Количество",
                CU_GOODS.PRICE * COUNT AS total[%12.2f] "ВСЕГО"
  FROM CU_ORDER_LINES, CU_GOODS
 WHERE CU_ORDER_LINES.GOODS_ID = CU_GOODS.id
   AND ORDER_ID = @LST_ORDERS:ID
 ORDER BY ID;
```

В квадратных скобках указаны маски полей ввода. В данном случае это ввод дробного числа с двумя знаками после запятой.

Не забудем создать перечень полей списка и для полей ID и ORDER_ID отметим признак «Погасить».

Мы предполагаем, что список строк заказа должен работать совместно со списком заказов и быть подчиненным по отношению к нему. Для этого в запросе строк заказа имеется ссылка на текущую строку списка заказов ORDER_ID = @LST_ORDERS:ID. Поэтому чтобы наш запрос списка срок заказа отработал, а соответственно, чтобы получить корректный список его полей в секции «Форма», запустим наше приложение и откроем в нем уже имеющийся список «Заказы». Тогда RP Designer корректно обработает ссылку на текущую строку этого списка @LST_ORDERS:ID.

VIII. Создаем MDI окно «Заказы+строки».

В менеджере объектов среды RP Designer выбираем закладку «MDI-окна» и добавляем объект – MDI окно под названием «**ORDERS_MDI».**

В секции «Список объектов» вписываем вот такой код:

```
BROWSER LST_ORDERS;
BROWSER LST_ORDER_LINES;
{-1, 2, 1, 1}
```

Что означает этот код?
Мы перечислили объекты, которые должны быть отображены в MDI окне (в данном случае – два уже известных нам списка – заказы и их строки). Параметры в фигурных скобках определяют лишь как списки должны быть взаимно ориентированы относительно друг друга.

Сохраняем объект.
Подключим созданное MDI окно к пункту меню «Заказы+строки»
Через клавишу <F7> традиционно можно проверить работоспособность созданного объекта, т.е. посмотреть как запускается созданное MDI окно.

То, что мы сделали, это пока еще не законченное приложение. Мы создали списки просмотра. И если бы в базе данных была информация, то мы могли бы ее видеть через эти списки. Причем у конечного пользователя автоматически доступны все сервисы поиска, фильтрации данных, индивидуальной настройки опций списков, выгрузки данных в популярные форматы Word, Excell и многое другое. Но нам необходима функциональность и для редактирования (добавления/удаления/изменения) данных. Для этого мы должны к спискам подключить диалоги редактирования.

IX. Диалог «Заказчики».

В менеджере объектов среды RP Designer выбираем закладку «Диалоги» и создаем диалог «DLG_CUSTOMERS».

В секции «SQL-запрос» прописываем запрос для диалога, также присвоив ему индивидуальное имя:

```
@DLG_CUSTOMERS
SELECT id, kod, inn, name, adr, acc, bank
  FROM CU_CUSTOMERS;
```

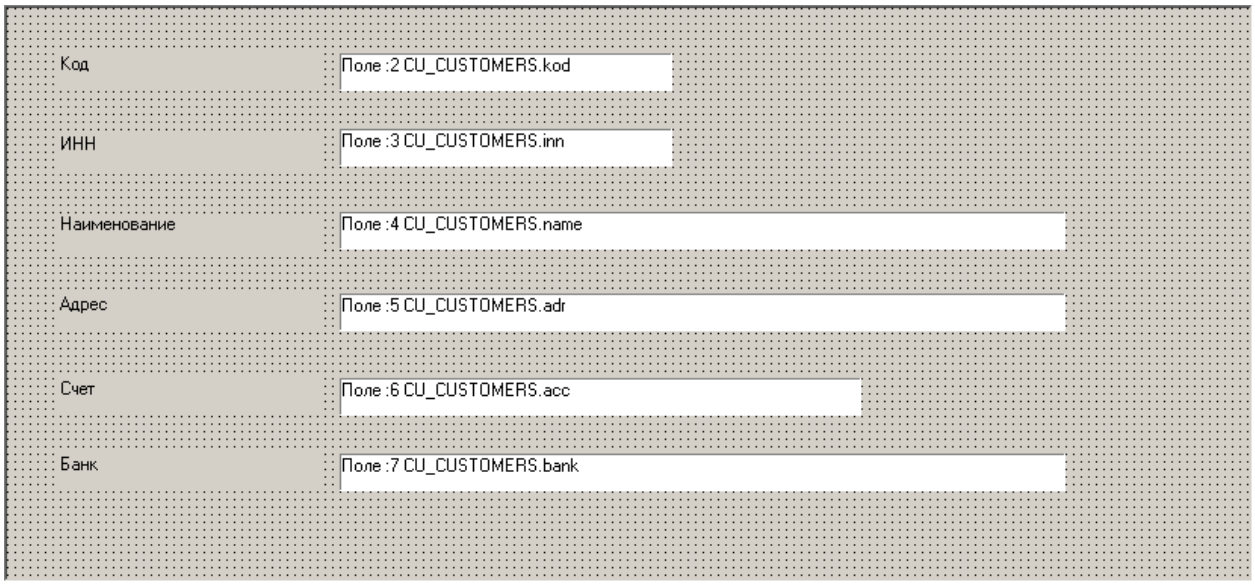
Далее переключаемся на секцию «Форма».

С помощью команды «Добавить поле редактирования»  создаем на диалоге 6 полей.

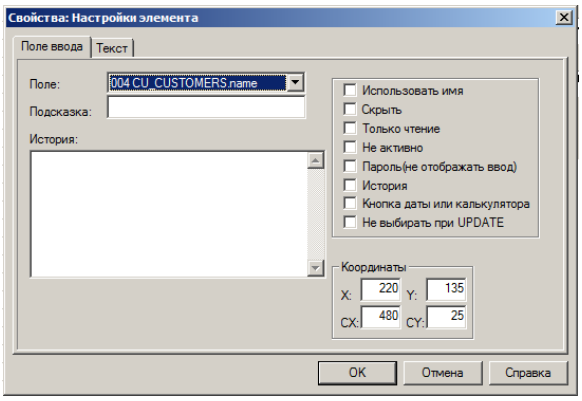
!!! Обратите внимание.
В этот момент в другом окне наше созданное приложение должно быть запущено по клавише <F7>.

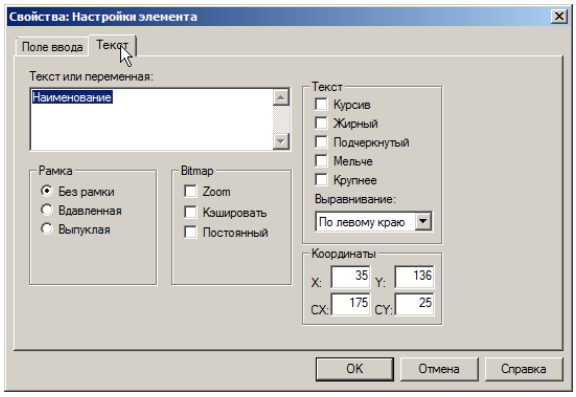
Поля можно разместить на экране, выровнять и определить их размер с помощью управляющих элементов в линейке инструментов «Layout»:

Поля диалога нужно привязать к полям запроса диалога и прописать для них соответствующие заголовки:



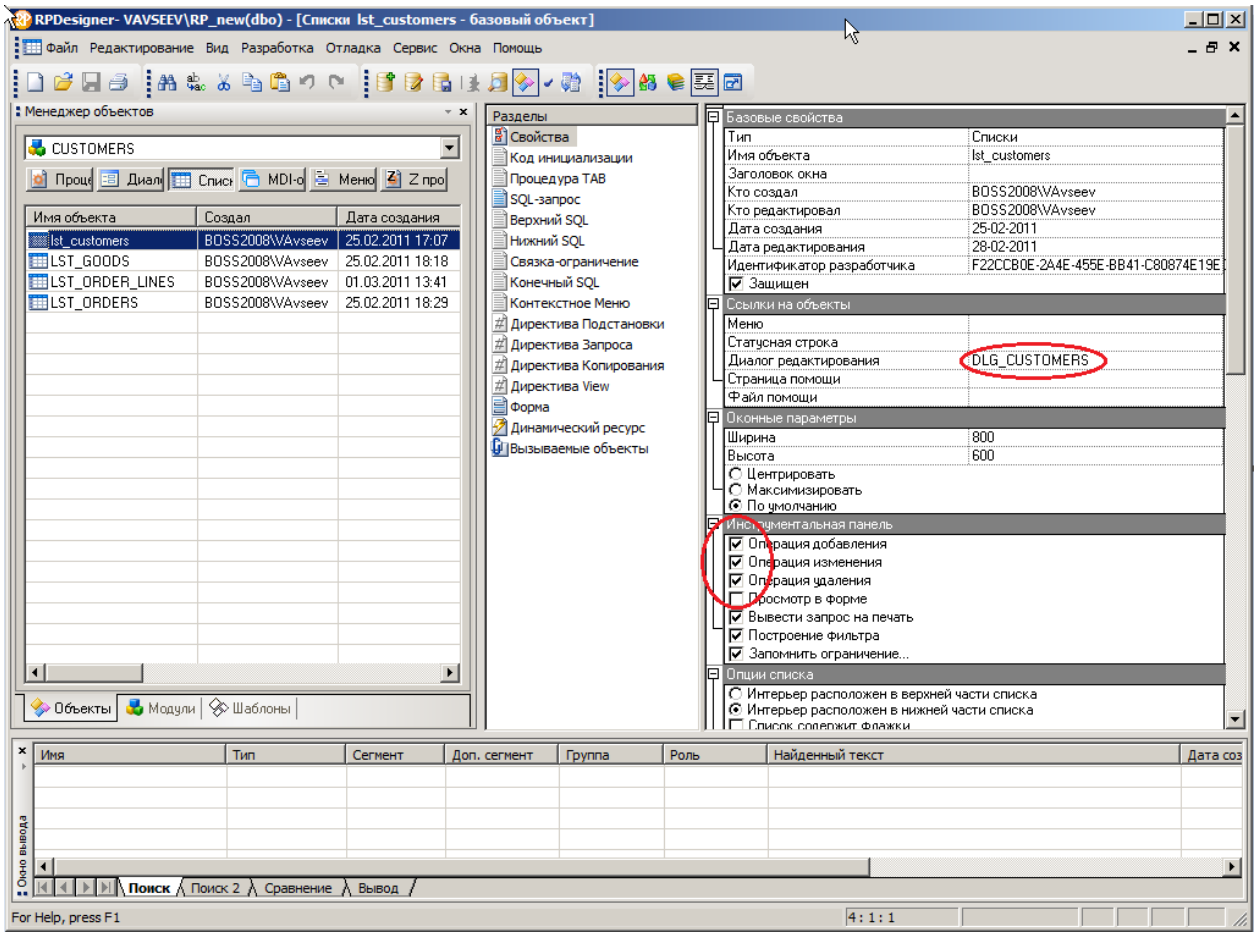
Для привязки полей диалога и изменения их свойств нужно дважды щелкнуть мышкой на поле:





Сохраняем диалог в базе данных -  .

Следующим шагом привязываем диалог к списку «LST_CUSTOMERS». Для этого в режиме редактирования списка «LST_CUSTOMERS» указываем в его свойствах, что диалогом редактирования для этого списка является диалог «DLG_CUSTOMERS» и что в списке разрешены операции добавления/изменения/удаления информации:



По кнопке  сохраняем изменения в списке..

Теперь мы можем проверить работу нашего приложения уже и в части редактирования данных о заказчиках. Данные в этом списке теперь можно добавлять и удалять через диалог редактирования. Заведем и отредактируем несколько записей в качестве теста.

!!! Обратите внимание.
Мы получили полностью работающий функционал по редактированию справочника заказчиков, не написав ни одной строки программного кода, описывающего как эти операции должны исполняться на сервере.
Единственное что потребовалось, это знать структуру данных в базе и корректно написать запрос SELECT для извлечения данных.

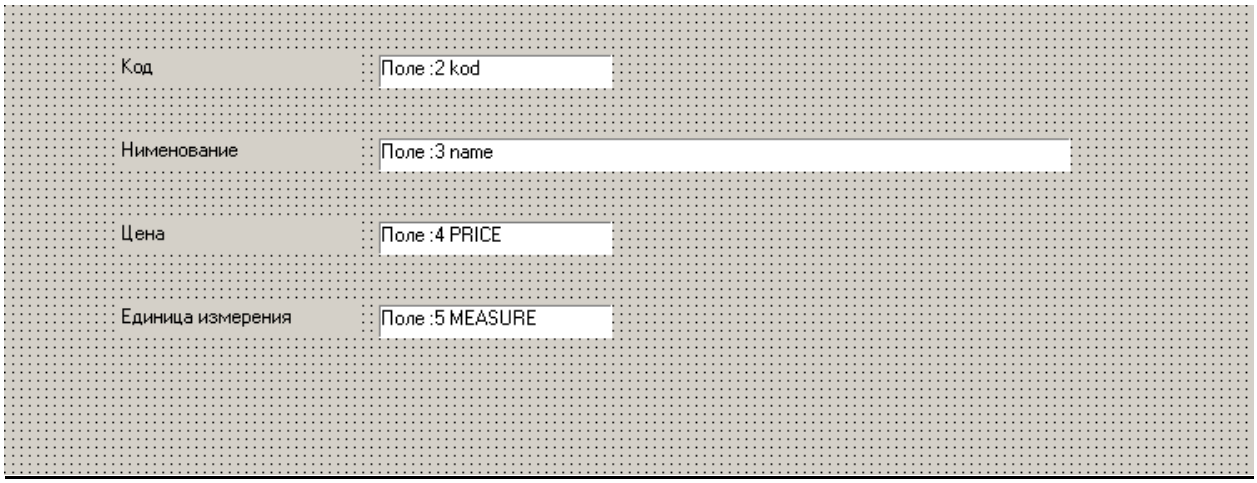
X. Диалог «Товары/услуги».

Аналогично предыдущему диалогу создаем новый диалог под именем «DLG_GOODS».

В секции «SQL-запрос» прописываем запрос для диалога:

```
@DLG_GOODS
SELECT id, kod, name, PRICE, MEASURE
 FROM CU_GOODS;
```

Заполняем форму диалога полями редактирования:



Подключим этот диалог к списку «LST_GOODS» и так же укажем, что в списке разрешены операции добавления/изменения/удаления информации.

XI. Диалог «Заказы».

Аналогично предыдущему диалогу создаем еще один новый диалог под именем «DLG_ORDERS».

В секции «SQL-запрос» прописываем запрос для диалога:

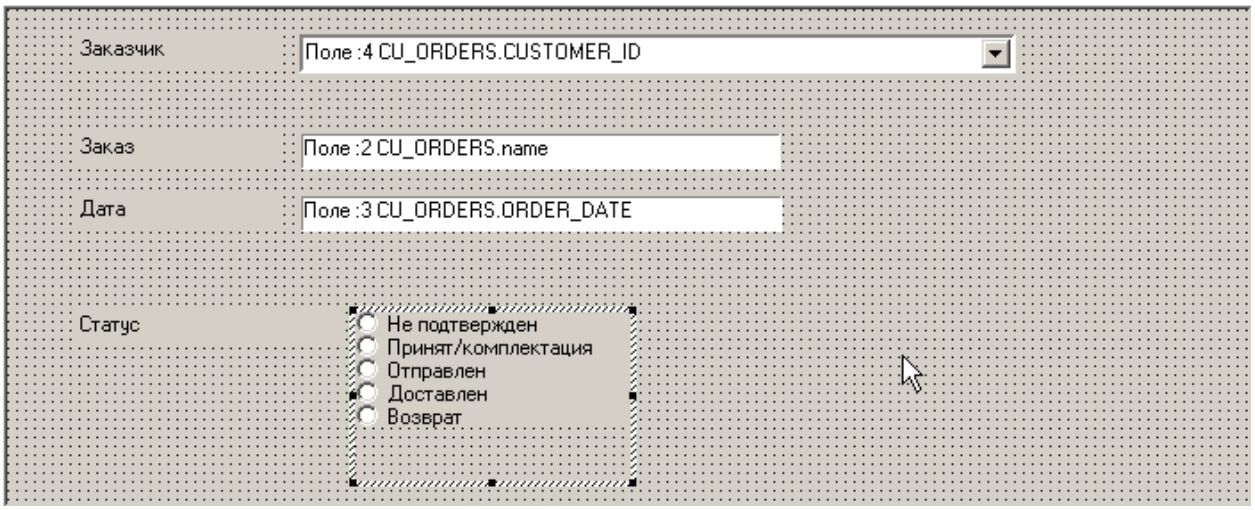
```
@DLG_ORDERS
SELECT id, name, ORDER_DATE, CUSTOMER_ID, status
  FROM CU_ORDERS;
```

В дополнение в секции «LET инициализация» пропишем вот такой код:

```
ORDER_DATE = getdate();
STATUS = 0;
```

В этом коде мы просто ссылаемся на поля диалога и определяем их значение по умолчанию, определяя поведение диалога при его открытии на добавление новой записи в список.

Сам диалог должен выглядеть вот таким образом:



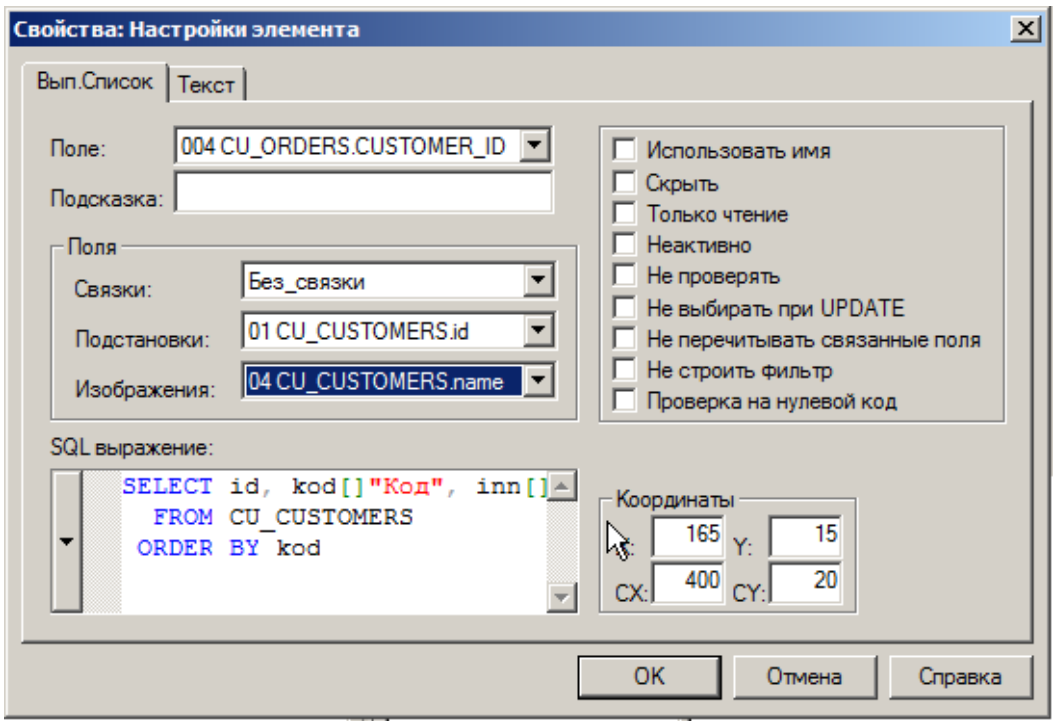
Поля «Заказ» и «Дата» являются простыми полями редактирования, как это было в предыдущих диалогах. В вот поля «Заказчик» и «Статус» имеют другой тип. Поле «Заказчик» имеет тип «Выпадающий список», а поле «Статус» - имеет тип «Переключатели».

Для поля «Заказчик» делаем вот такую настройку его свойств:

В поле «SQL выражение» (свойства элемента «Заказчик») вписываем запрос:

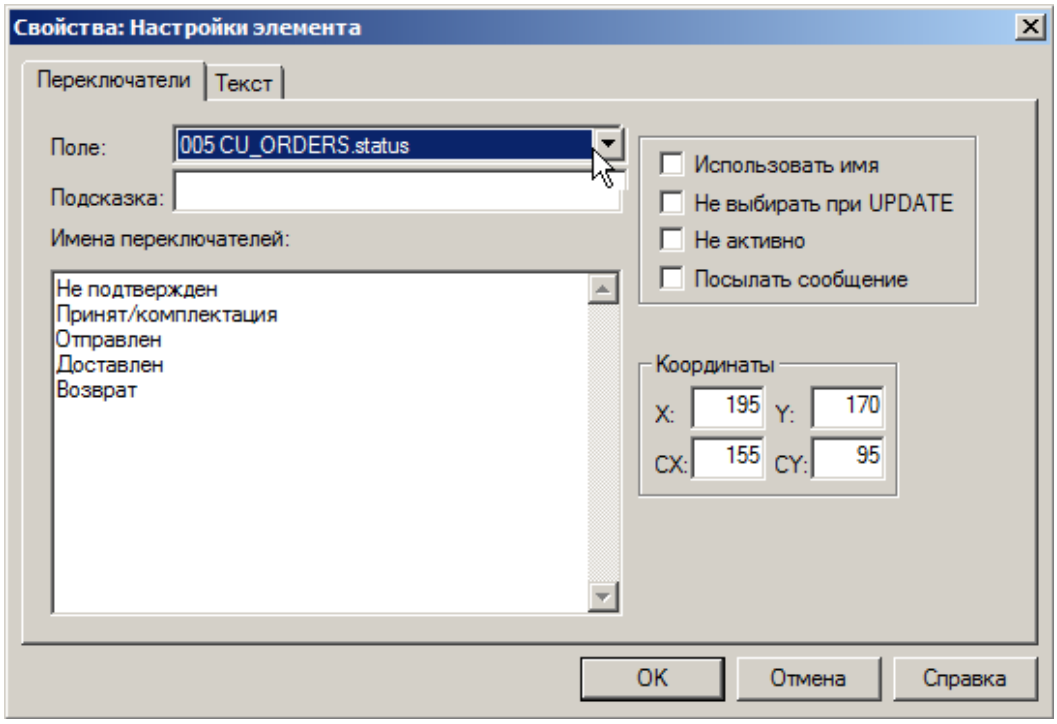
```
SELECT id, kod[]"Код", inn[]"ИНН", name[]"Наименование", adr[]"Адрес", acc[]"Счет", bank[]"Банк"
  FROM CU_CUSTOMERS
 ORDER BY kod;
```

Выбираем в качестве подстановки поле CUSTOMER_ID.



При такой настройке указанное SQL выражение позволяет развернуть на экране для выбора подстановки список заказчиков и выбрать из них нужного. При этом в базе данных будет сохранено поле ID (ссылка на заказчика), а в диалоге будет отображаться название заказчика.

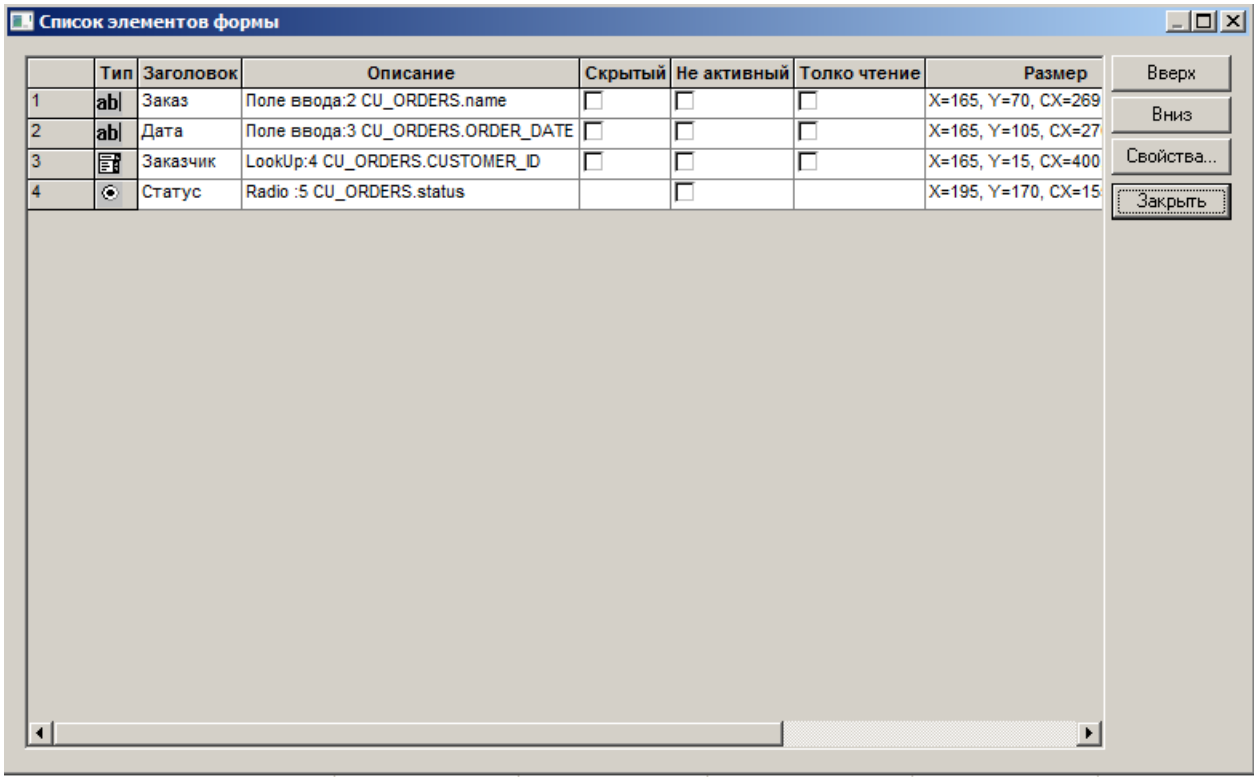
Для поля «Статус» делаем следующую настройку:



Подключаем созданный диалог к списку «LST_ORDERS», не забыв указать, что в нем разрешены операции добавления/изменения/удаления информации.

Если щелкнуть на поле редактирования диалога правой кнопкой мыши, то можно перейти в список элементов. Можем проверить, соответствует ли получившийся у нас порядок полей правильному порядку их обхода сверху вниз.

Если это необходимо, изменим порядок обхода клавишами «Вверх» или «Вниз»:



XII. Диалог «Заказы+строки».

Аналогично предыдущему диалогу создаем новый диалог под именем «DLG_ORDER_LINES».

В секции «SQL-запрос» прописываем запрос для диалога:

```
@DLG_ORDER_LINES
SELECT ID, ORDER_ID, GOODS_ID, "COUNT"
  FROM CU_ORDER_LINES;
```

В секции «LET инициализация» пропишем вот такой код:

```
-- Присваиваем ссылку на текущий заказ.
ORDER_ID = @LST_ORDERS:ID;
```

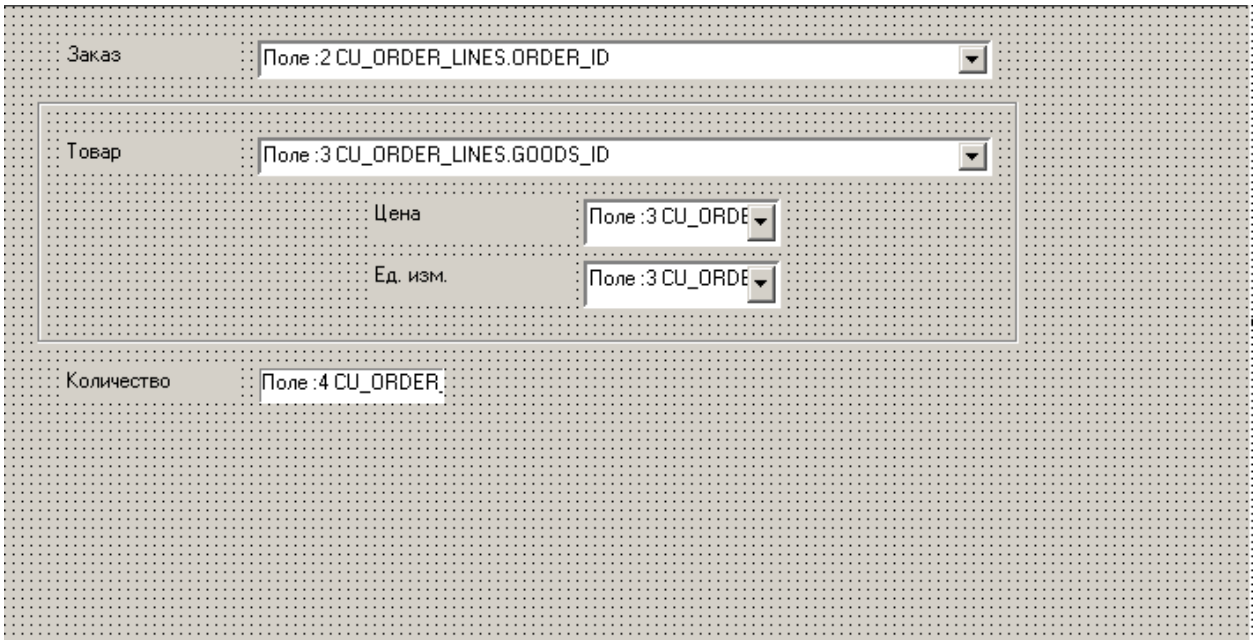
Это означает, что при открытии диалога на вставку в поле «заказ» будет подставлена ссылка на текущий заказ из списка заказов.

А в секции «Конечный SQL» напишем:

```
-- При изменении информации о строке заказа перечитать список заказов
MSG 2041, @LST_ORDERS;
```

Это означает, что при изменении информации о строке заказа обязательно нужно перечитать список заказов (команда перечитки, посылаемая соответствующему списку).

Сам диалог должен выглядеть вот таким образом:

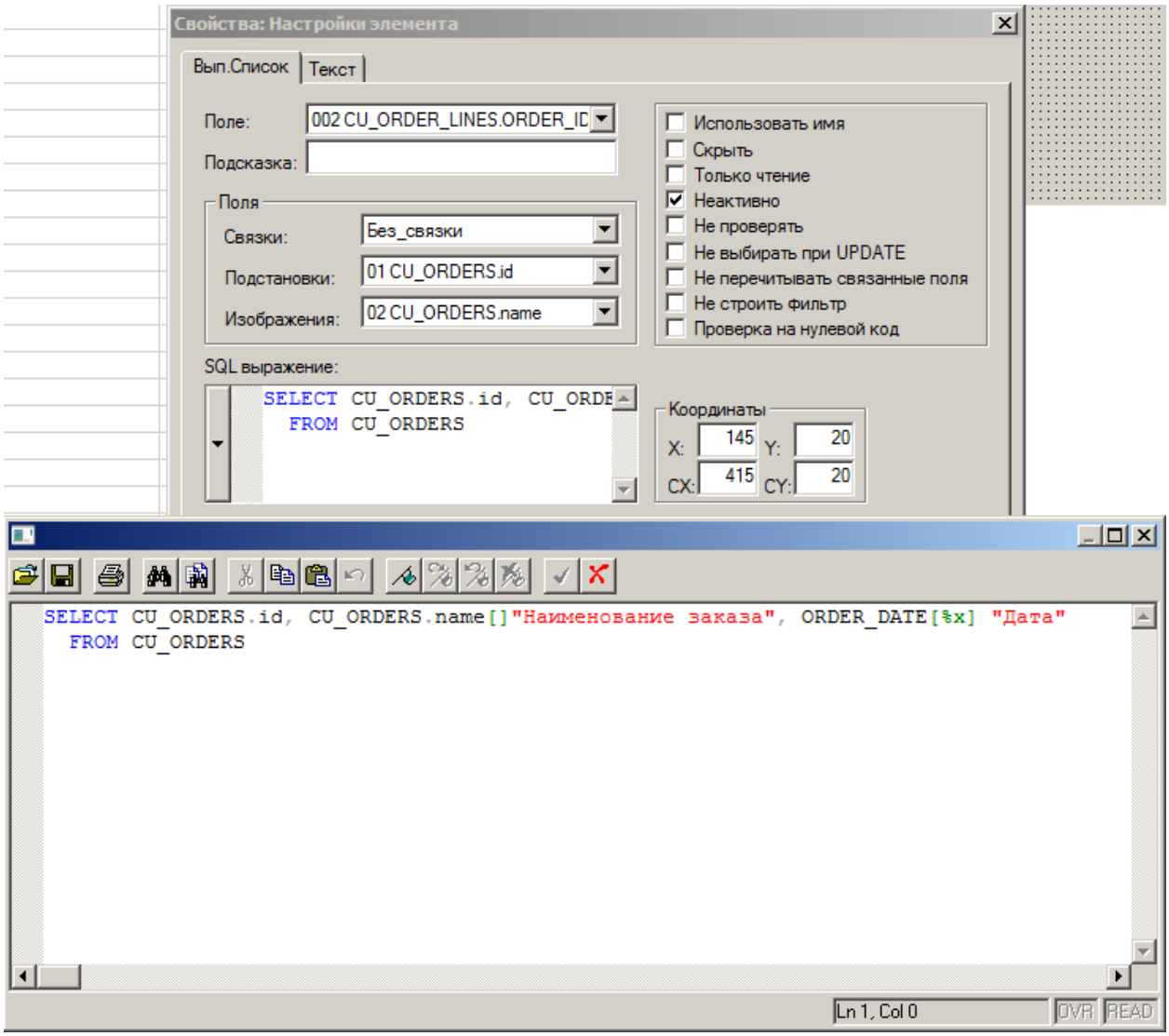


Поле «Количество» - простое поле редактирования.
Поле «Заказ» – это выпадающий список, запрос которого выглядит вот таким образом:

```
SELECT CU_ORDERS.id, CU_ORDERS.name[]"Наименование заказа", ORDER_DATE[%x] "Дата"
```

FROM CU_ORDERS

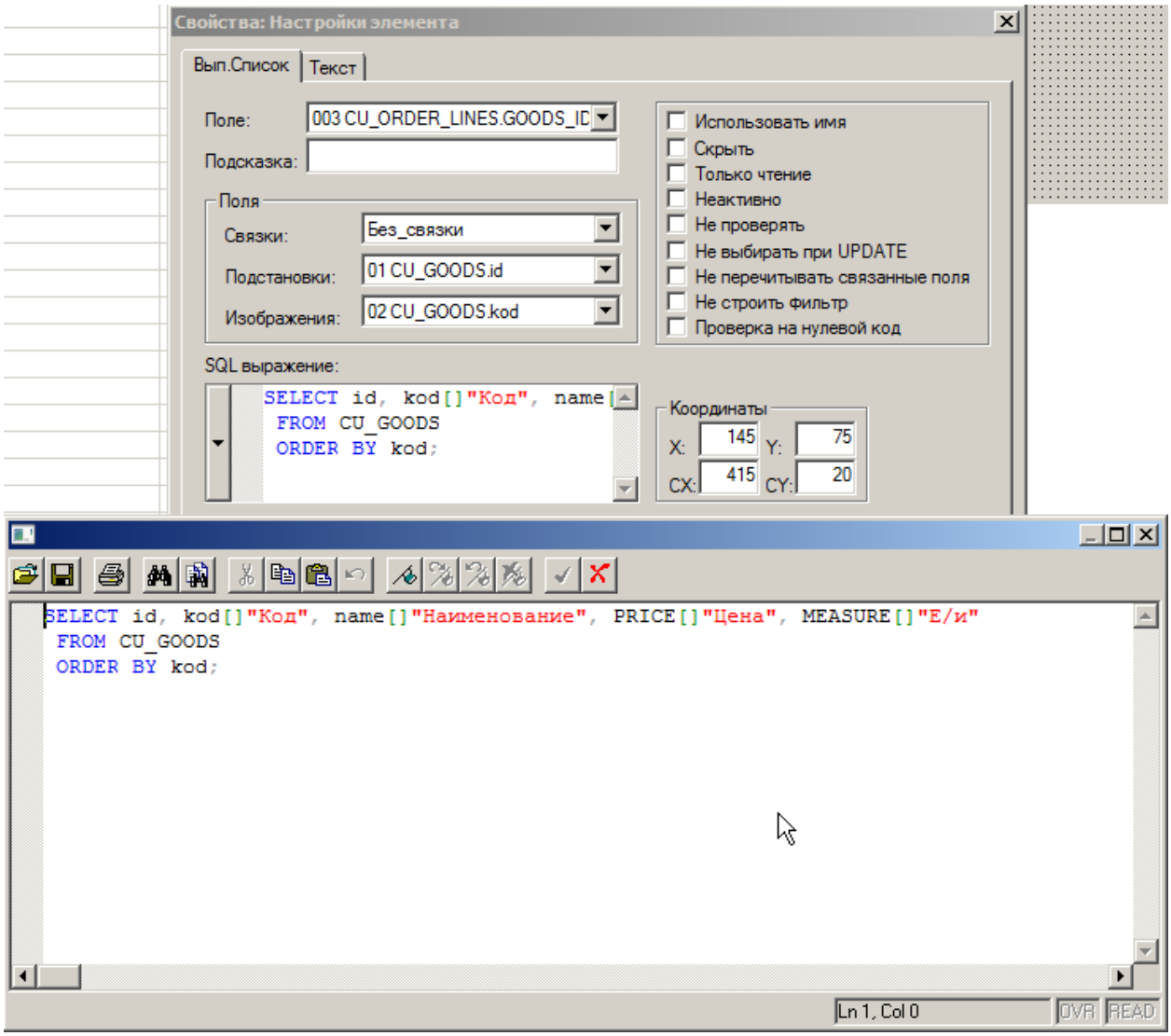
Настройки поля «Заказ», соответственно, такие:



Поле «Товар» - тоже выпадающий список. Вот его запрос:

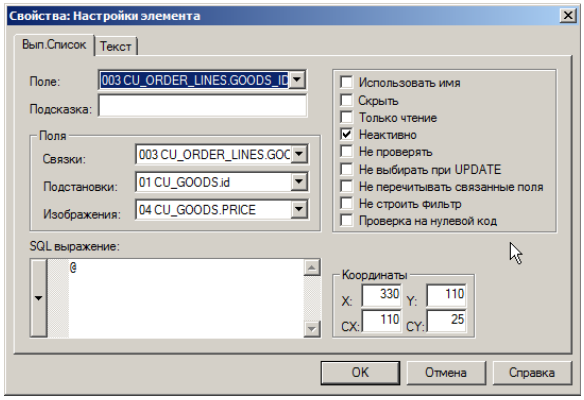
```
SELECT id, kod[]"Код", name[]"Наименование", PRICE[]"Цена", MEASURE[]"Е/и"
 FROM CU_GOODS
 ORDER BY kod;
```

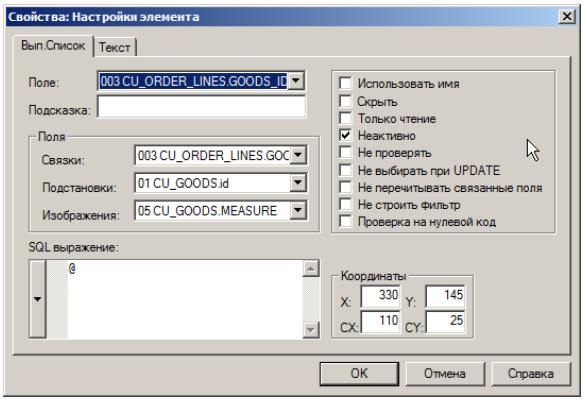
Настройки поля «Товар», соответственно, такие:



Немного дополним диалог.
Мы выбираем товар из выпадающего списка. В поле «Товар» мы видим название товара/услуги. Но у товара/услуги есть еще ряд полей, которые было бы интересно видеть в диалоге. Это поля «Цена» и «Единица измерения». Будем выводить их в диалоге под названием выбранного товара, но без возможности их редактирования.

Для этого под полем «Товар» добавляем 2 поля типа «Выпадающий список», указав для них поле связки «GOODS.ID». Вот пример настройки свойств этих полей:





Подключаем созданный диалог к списку «LST_ORDER_LINES», не забыв указать, что в списке разрешены операции добавления/изменения/удаления информации.

XIII. Теперь займемся “украшательством”.

1. Пропишем заголовки окон.

У каждого окна в системе может быть заголовок. Это относится и к спискам, и к диалогам, и к MDI окнам. Пройдитесь по свойствам объектов и определите их заголовки. У списков и MDI окон это свойство называется «Заголовок окна». У диалогов это свойство называется «Добавить к заголовку».

!!! Обратите внимание.
Заголовки диалогов следует указывать в винительном падеже, поскольку к введенному нами заголовку система сама автоматически будет добавлять слова «Добавить»/«Переписать»/ «Удалить» в зависимости от выполняемого пользователем действия.

2. Введем отображение статусов заказов

Для этого отредактируем запрос списка заказов (дополнив его тем, что выделено красным):

```
@LST_ORDERS
SELECT CU_ORDERS.id, CU_ORDERS.name[]"Наименование заказа", ORDER_DATE[%x] "Дата",
CU_CUSTOMERS.name[]"Заказчик",
    (CASE status WHEN 0 THEN 'Не подтвержден'
                                    WHEN 1 THEN 'Принят/комплектация'
                                    WHEN 2 THEN 'Отправлен'
                                    WHEN 3 THEN 'Доставлен'
                                    WHEN 4 THEN 'Возврат' END) []"Статус",
    (CASE status WHEN 0 THEN 239
                                    WHEN 1 THEN 251
                                    WHEN 2 THEN 229
                                    WHEN 3 THEN 228
                                    WHEN 4 THEN 227 END)
 FROM CU_ORDERS _HINTBROWSER JOIN CU_CUSTOMERS ON CU_CUSTOMERS.id =
CU_ORDERS.CUSTOMER_ID
 ORDER BY CU_CUSTOMERS.name, ORDER_DATE;
```

Далее зайдем в секцию «Форма» этого списка и 2 раза нажмем <Ins> (“Добавить колонку”) и <F5>. Для последнего из 2-х появившихся новых полей поднимем флажки «Погасить» и «Определяет цвет» (т.е. само не светится в списке, но определяет цвет всей строки).
Если у нас в списке заказов есть строки, то, изменяя их статус, мы теперь будете получать разное выделение строк цветом (цвет определяется значением числа поля, определяющего цвет).

3. Выведем итоговые показатели в заказах.

Займемся дальнейшей редакцией запроса списка заказов. Еще дополним запрос (выделено красным):

```
@LST_ORDERS
SELECT CU_ORDERS.id, CU_ORDERS.name[]"Наименование заказа", ORDER_DATE[%x] "Дата",
CU_CUSTOMERS.name[]"Заказчик",
```

```
    (CASE status WHEN 0 THEN 'Не подтвержден'
                                        WHEN 1 THEN 'Принят/комплектация'
                                        WHEN 2 THEN 'Отправлен'
                                        WHEN 3 THEN 'Доставлен'
                                        WHEN 4 THEN 'Возврат' END) []"Статус",
    (CASE status WHEN 0 THEN 239
                                        WHEN 1 THEN 251
                                        WHEN 2 THEN 229
                                        WHEN 3 THEN 228
                                        WHEN 4 THEN 227 END),
        (SELECT SUM(PRICE * COUNT)
                FROM CU_ORDER_LINES, CU_GOODS
                 WHERE CU_ORDER_LINES.GOODS_ID = CU_GOODS.ID
                   AND ORDER_ID = CU_ORDERS.ID) AS TOTAL[%19.2f]"ИТОГО",
        (SELECT COUNT(*)
                FROM CU_ORDER_LINES
                 WHERE ORDER_ID = CU_ORDERS.ID) AS STROK[%19.0f]"строк"
  FROM CU_ORDERS _HINTBROWSER JOIN CU_CUSTOMERS ON CU_CUSTOMERS.id =
CU_ORDERS.CUSTOMER_ID
  ORDER BY CU_CUSTOMERS.name, ORDER_DATE;
```

Опять зайдем в секцию «Форма» этого списка и 2 раза нажмем <Ins> (“Добавить колонку”) и <F5>. Теперь при изменении состава заказа его общая стоимость и число строк заказа пересчитываются автоматически.

ИТАК, МЫ СОЗДАЛИ ПРИЛОЖЕНИЕ, АВТОМАТИЗИРУЮЩЕЕ УЧЕТ ЗАКАЗОВ НАШЕГО ПРЕДПРИЯТИЯ!

После того как мы (или администратор SQL Server, если это в его обязанностях) предоставим доступ соответствующим лицам к БД, на которой мы только что создали систему, ПОЛЬЗОВАТЕЛИ МОГУТ ПРИСТУПАТЬ К ПОЛНОЦЕННОЙ РАБОТЕ !

XIV. ПРИЛОЖЕНИЕ.

Диаграмма схемы данных нашей базы данных выглядит вот таким образом: