Лабораторная работа 5

ГРАФИЧЕСКОЕ ПРЕДСТАВЛЕНИЕ ДАННЫХ, ФОРМИРОВАНИЕ ПРОСТЫХ ОТЧЕТОВ

Цель работы: познакомится с возможностями построения отчетов и графиков на основе хранимой в БД информации.

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Ознакомиться с теоретическими сведениями.
2. Создать необходимые для построения графиков **SQL**-запросы и поместить их в компоненты **ADOQuery**.
3. Организовать настройку и отображения графиков в разрабатываемом приложении.
4. Установить набор компонентов генератора отчетов **FastReport**.
5. Сформировать необходимые отчеты по данным.
6. Ответить на контрольные вопросы.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

В данной лабораторной работе будут рассмотрены аспекты формирование статистической выборки и построение графиков с использованием компонентов **DBChart** и **ChartSeries**. Также познакомимся с функциональными возможностями генератора отчетов **FastReport**.

При этом данные используемые для построения графиков и отчетов, как правило, формируются с применением **SQL**-запросов.

5. Построение графиков и отчетов

Для построения графиков воспользуемся информацией о средней стоимости товаров, данными о количестве товаров на складе, а также сведениями о продажах.

Для того чтобы получить среднюю стоимость товаров в каждой категории сформируйте SQL запрос:

```
SELECT
  view_goods_catalog.tgName,
  AVG(view_goods_catalog.gcCost) AS avgCost
FROM view_goods_catalog
GROUP BY view_goods_catalog.tgName
```

Для вывода данных этого запроса в виде графика, на **DataModule2** добавьте компонент **ADOQuery** (назовите - **Q_avg_cost_by_type**), и в свойство **SQL** пропишите данный код.

Чтобы получить информацию о продажах по определенным датам необходимо сформировать запрос:

```
SELECT
  view_goods_operations.oDateTime,
  view_goods_operations.oFullCost
FROM view_goods_operations
WHERE view_goods_operations.oIsSale = 1
```

Однако при выполнении данного запроса продажи, оформленные в один день будут учитываться как две разные позиции (рис 5.1, а).

Для того чтобы исправить данную ситуацию нам необходимо просуммировать стоимость каждой продажи и сгруппировать результат по дням (рис 5.1, б). Изменим запрос следующим образом:

```
SELECT
  CAST(view_goods_operations.oDateTime AS date) AS oSaleDate,
  SUM(view_goods_operations.oFullCost) AS oDayCost
```

```
FROM view_goods_operations
WHERE view_goods_operations.oIsSale = 1
GROUP BY oSaleDate
```

oDateTime DATETIME	oFullCost DECIMAL(42, 2)
06.05.2021 11:25:00	86295,00
06.05.2021 14:27:00	129287,00
12.05.2021 12:31:54	25499,00
14.05.2021 10:34:23	27955,00
14.05.2021 12:32:51	129980,00
18.05.2021 14:38:45	25499,00
18.05.2021 12:34:17	49999,00
20.05.2021 11:23:41	25499,00
22.05.2021 15:35:16	64990,00

а

oSaleDate DATE	oDayCost DECIMAL(64, 2)
06.05.2021	215582,00
12.05.2021	25499,00
14.05.2021	157935,00
18.05.2021	75498,00
20.05.2021	25499,00
22.05.2021	64990,00

б

Рис. 5.1. Данные по продажам товаров: а) вывод всех данных, б) с группировкой по дате

Измененный запрос пропишите в новом компоненте **ADOQuery,** который назовите **Q_sales_by_days**.

5.1 Добавление графиков в проект

На графике может одновременно располагаться несколько серий. В большинстве случаев их значения строятся по одинаковому закону, при этом две и более серий показываются на графике одновременно для сравнения. Рассмотрим способы добавления серии на график и настройку ее параметров на уровне программного кода.

Рассмотрим примеры создания графиков, которые выводят информацию о стоимости товара и сведения о продажах.

Создайте новую форму (**Form6**), и подключите к ней **Unit2** (**DataModule2**) через меню **File** пункт **Use Unit**. На форме разместим **PageControl** в котором создадим три вкладки.

Для начала создадим график отображающий среднюю стоимость товаров в зависимости от типа, для этого на первой вкладке разместим компонент **DBChart** (по умолчанию имя **DBChart1**).

Войдем в редактор графика (рис 5.2) используя двойное нажатие по компоненту **DBChart**.

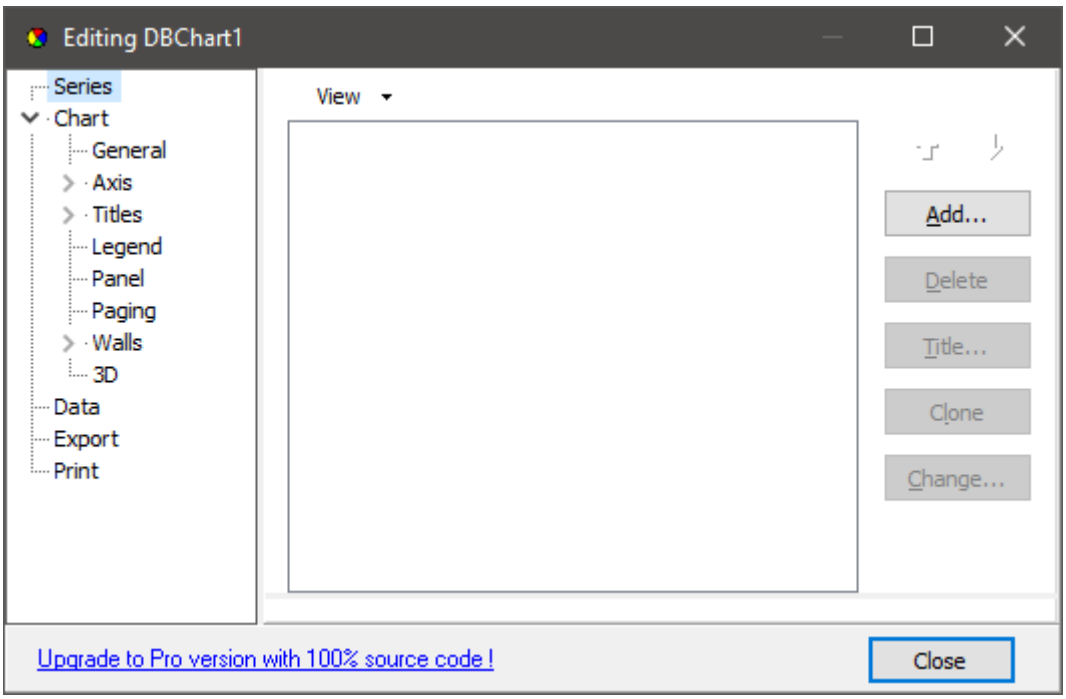


Рис. 5.2. Окно редактора графика

Для добавления серии в график на закладке **Series** нажмем кнопку **Add**, после чего появится окно выбора типа серии. В этом окне представлены базовые типы серий и возможности настройки стиля их отображения. В левом нижнем углу можно увидеть стрелку, при нажатии на которую отобразится галерея всех видов графиков с отображением подтипов (рис. 5.3):

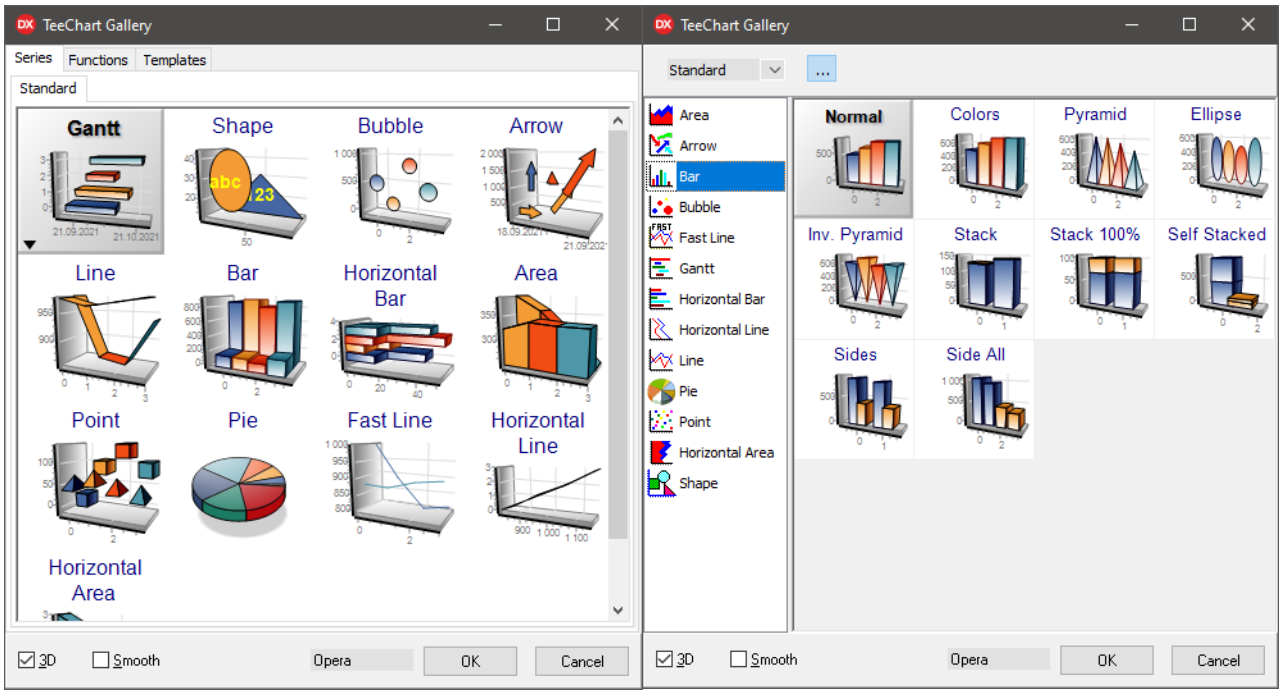


Рис. 5.3. Окно выбора типа серии

Выберем серию типа **Bar** и нажмем **OK**. В списке серий будет показана серия типа **Bar**, а в дереве параметров будет отображено ее название. При выборе в нем созданной серии **Series1** будет отображен блокнот с закладками **Format**, **Stack**, **General**, **Marks**, **DataSource** (рис. 5.4).

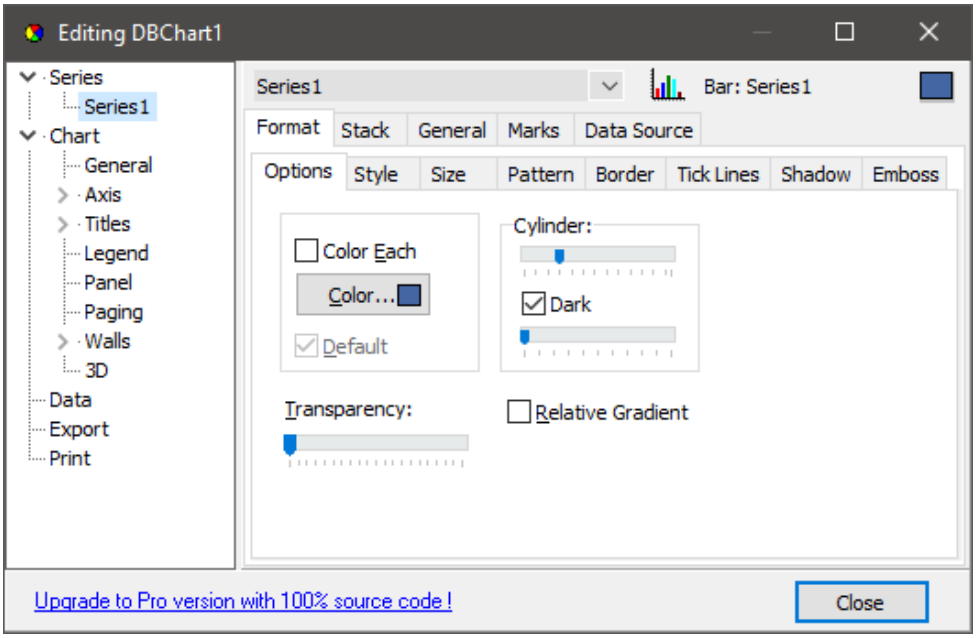


Рис. 5.4. Окно настройки серии

Свойства серии, которые можно установить на страницах, соответствующих этим закладкам:

– **Format** – отвечает за настройку отображения серии. Здесь можно определять свойства палитры, линий графика и т. д.;

– **Stack** – предоставляет доступ к настройкам вида сбора графика при наличии нескольких серий;

– **General** – позволяет задавать форматы данных;

– **Marks** – устанавливает параметры маркеров (значения в рамке над точками серий). На графике маркеры могут показывать координаты узловых точек. Маркеры отображаются на графике, если отмечено свойство **Visible**. Свойство **Style** дает возможность настроить вид марок;

– **DataSource** – определяет источник данных для серии.

После создания серии необходимо выбрать источник данных для нее. Выпадающий список позволяет определить тип источника данных для серии:

– **No Data** – источник данных серии не назначается (в случае, когда это делается программно). Кроме того, заготовленный шаблон серии можно в разное время использовать для показа данных из разных источников, которые переключаются программно во время выполнения приложения;

– **Random Values** – набор случайных чисел, который бывает полезен при формировании заготовки серии;

– **Function** – функция, служит для построения графиков на основании данных в двух или более серий;

– **DataSet** – указывает НД, значения полей (столбцов) которого будут использоваться для формирования точек серии.

Из выпадающего списка **DataSet** выберем компонент **Q_ _avg_cost_by_type.** Укажем, что в качестве меток (**Labels**) будет использоваться поле **tgName,** а поле **avgCost** – значения **Bar**. (рис. 5.5):

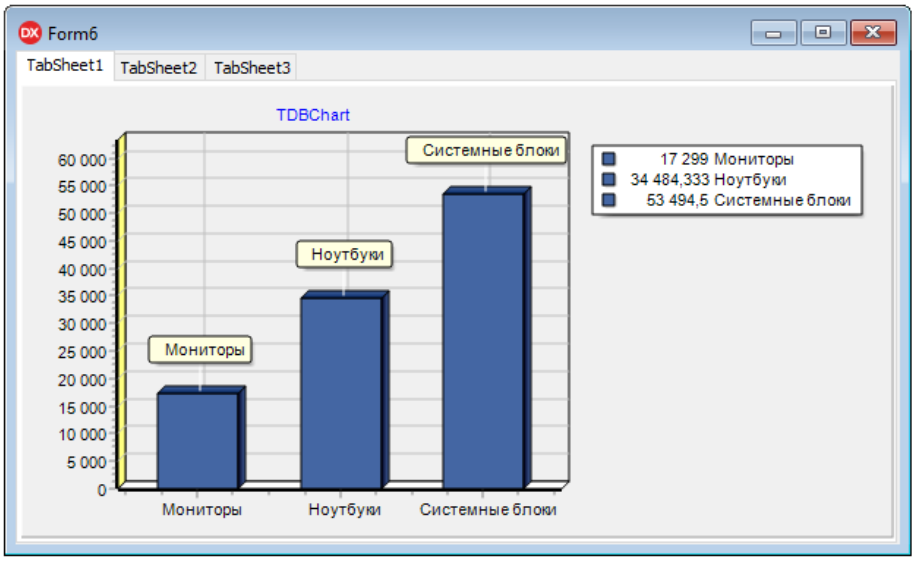
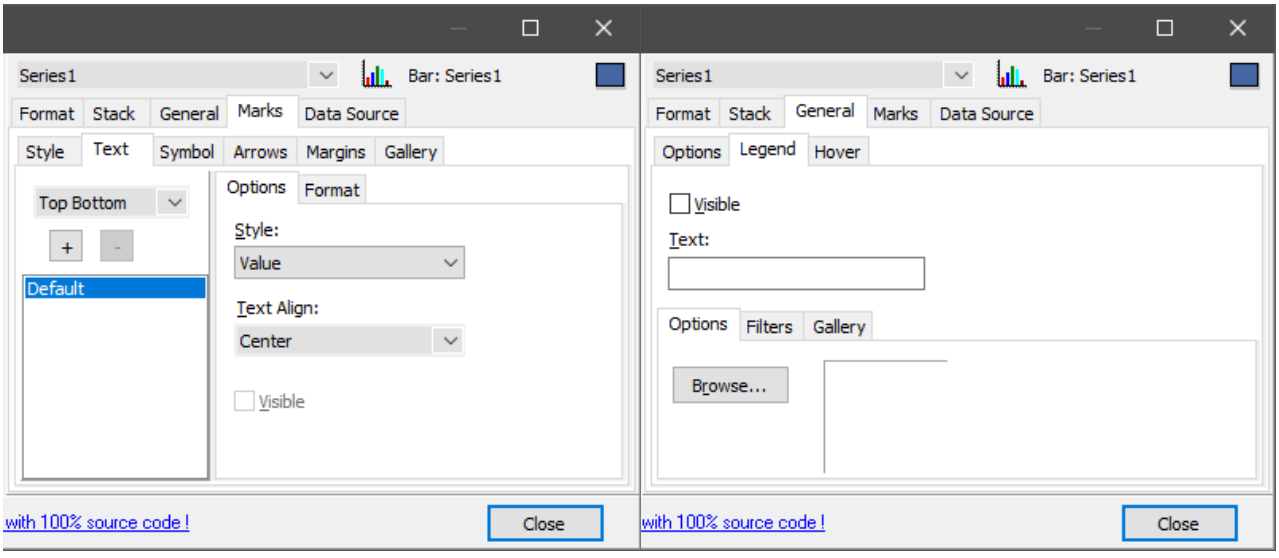


Рис. 5.5. График средней стоимости товаров по категориям

Для данного графика не целесообразно выводить легенду и в маркерах выводить название типа товаров. Для отключения легенды в настройках серии у легенды уберем флажок **Visible** (рис 5.6, а) А выводимый текст в маркерах изменим на конкретное значение (рис 5.6, б)



а б

Рис. 5.6. Пример настройки графика с изменением отображаемых данных: а) настройка легенды б) настройка маркеров.

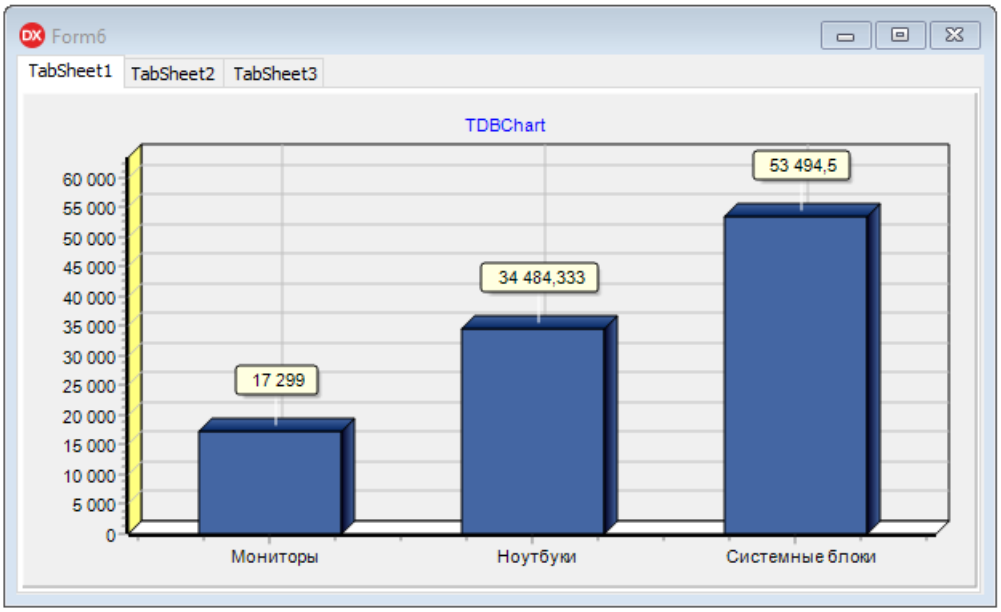
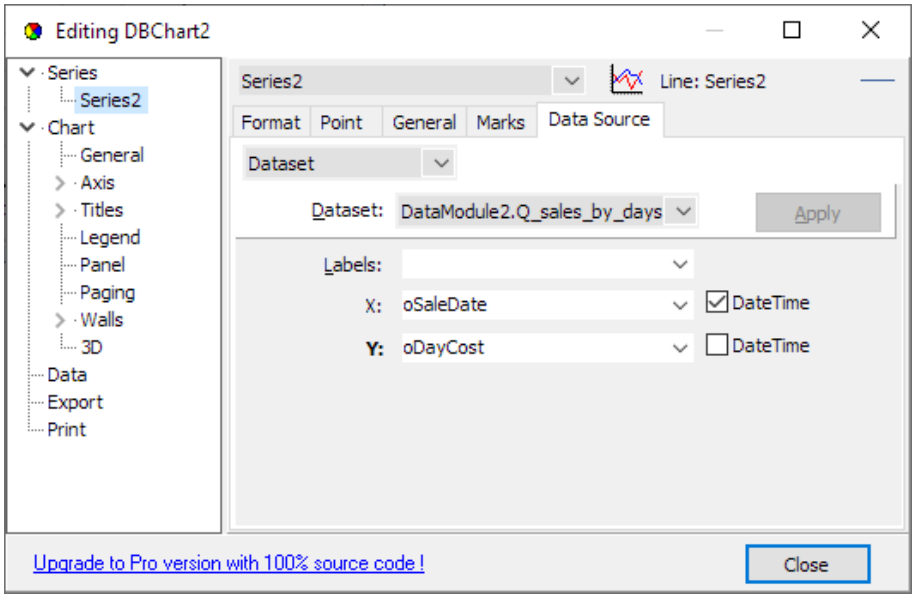


Рис. 5.7. Пример графика с настроенным отображением

Формирование графика для вывода сведений о продажах из **Q_ _sales_by_days** будет происходить схожим образом. Разместим на второй вкладке компонент **DBChart** (имя по умолчанию **DBChart2**). Вызовем редактор, и на этот раз при создании серии вместо типа **Bar** укажем **Line**.

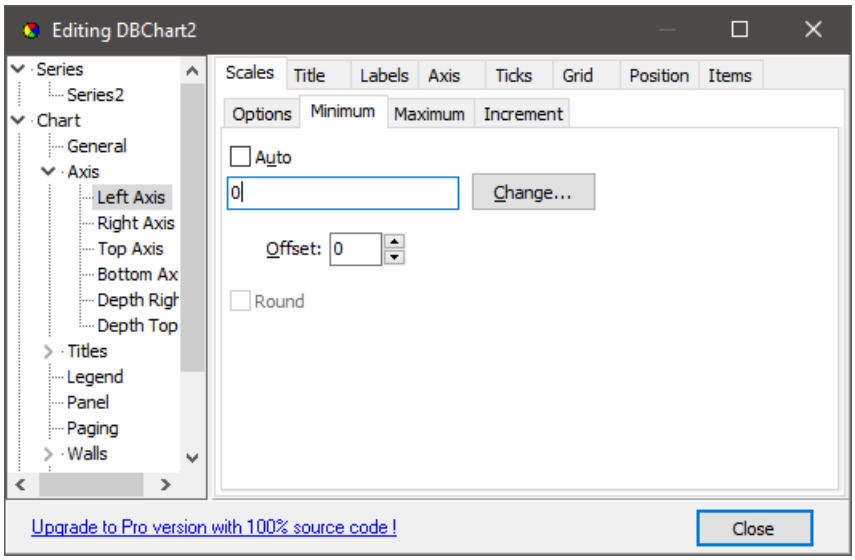
Настройка источника данных будет следующей: из выпадающего списка **DataSet** выберем компонент **Q_ _sales_by_days**. Укажем, что поле **oSaleDate** содержит значения по оси **X**, а поле **oDayCost** – значения по оси **Y**.

Поскольку поле **oSaleDate** является полем типа **TDateField**, то напротив настройки оси **X** необходимо поставить флажок **DateTime** (рис 5.8, а).

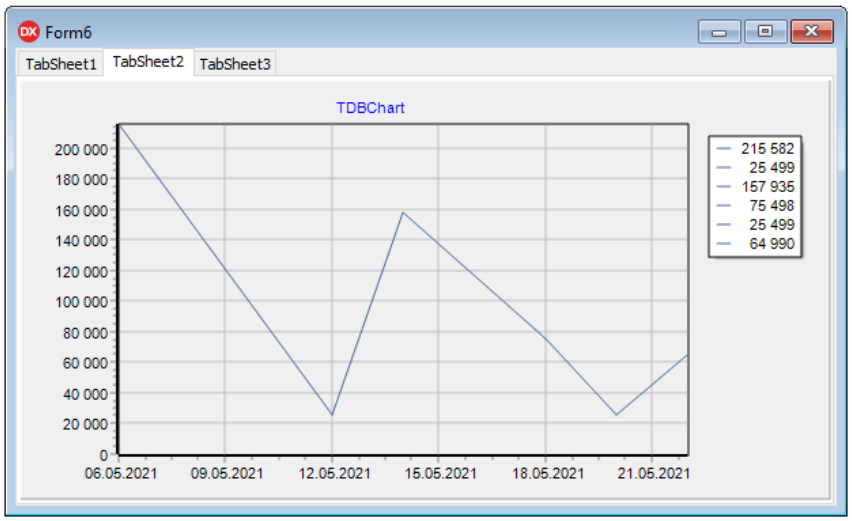


а

Для того чтобы вертикальная ось не выбирала в качестве начального значения минимальное из возможных, то в ручном режиме установим значение **0** (рис 5.8, б).



б



в

Рис. 5.8. График продаж по датам: а) настройка источника данных; б) настройка левой вертикальной оси; в) отображение графика.

Для вызова формы с графиками модифицируем основное меню на главной форме добавив пункт **Аналитические данные** в котором создадим подпункт **Графики**. Для вызова формы с графиками в обработчике **OnClick** этого пункта пропишем:

```
procedure TForm1.N7Click(Sender: TObject);
begin
  Form6.ShowModal;
end;
```

Схожим образом осуществляется формирование всех видов графиков. При выводе графиков их можно обновлять изменяя исходный набор данных.

Рассмотрим данную возможность на примере отображения складских запасов с возможностью фильтрации по типу товара. На **DataModule2** добавьте компонент **ADOQuery** (назовите - **Q_storage_count**).

На **Form6** в третьей вкладке разместим следующие компоненты: **Panel** и **DBChart**. На панель поместим **DBLookupComboBox** для выбора типа товара, **Button** для сброса фильтра и **Label** для пояснения. Поскольку товаров может быть много то целесообразно сделать график типа **Horizontal Bar** (рис 5.9):

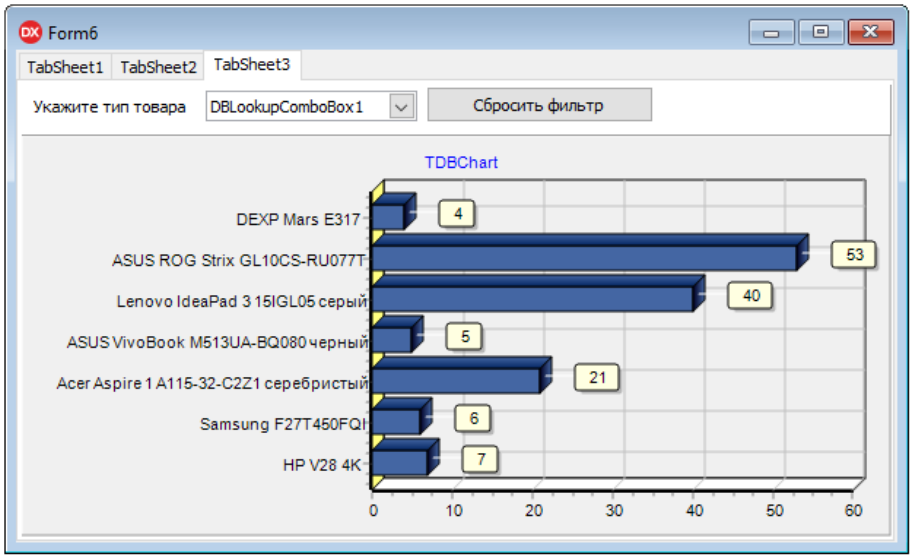


Рис. 5.9. График количества товара на складе.

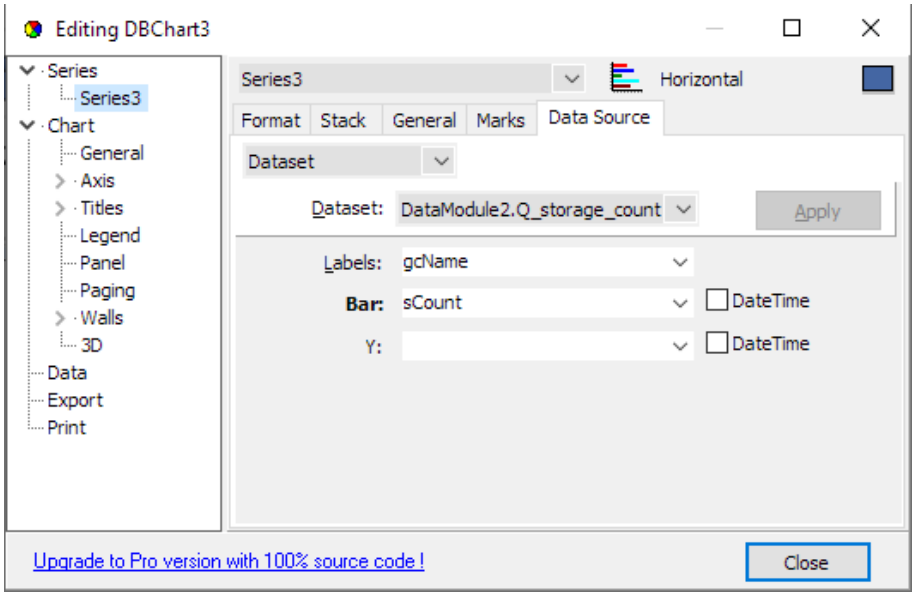


Рис. 5.10. Настройка источника данных для графика.

Для **DBLookupComboBox1:**

Свойство	Значение
ListSource	DS_T_type_goods
ListField	tgName
KeyField	IDtg

Непосредственная организация работы с графиком будет осуществляться с использованием события **OnCloseUp** компонента **DBLookupComboBox1**. Пропишем следующий код в обработчик события:

```
procedure TForm6.DBLookupComboBox1CloseUp(Sender: TObject);
var
  baseSQL, filter_s: string;
begin
  baseSQL := 'SELECT * FROM view_storage ';
  filter_s := '';

  if DBLookupComboBox1.KeyValue <> null then
    filter_s := 'WHERE IDtg = ' + inttostr(DBLookupComboBox1.KeyValue);

  DataModule2.Q_storage_count.sql.text := baseSQL + filter_s;
  DataModule2.Q_storage_count.Active := true;
  DBChart3.RefreshData;
end;
```

Поскольку компонент **DBLookupComboBox** не предусматривает выбор пустого значения, то для сброса фильтра будем использовать событие **OnClick** кнопки **Сбросить фильтр**:

```
procedure TForm6.Button1Click(Sender: TObject);
begin
  // назначаем пустое значение в качестве выбора
  DBLookupComboBox1.KeyValue := null;
  //имитируем выполнение события OnCloseUp в компоненте DBLookupComboBox1
  DBLookupComboBox1CloseUp(nil);
end;
```

Запустив приложение можно проверить, как будет работать фильтрация данных для графика. Пример работы фильтра продемонстрирован на рис 5.11. Фильтр можно применять к любым данным используемым для отображения на графике (так можно отобразить данные с учетом временного периода).

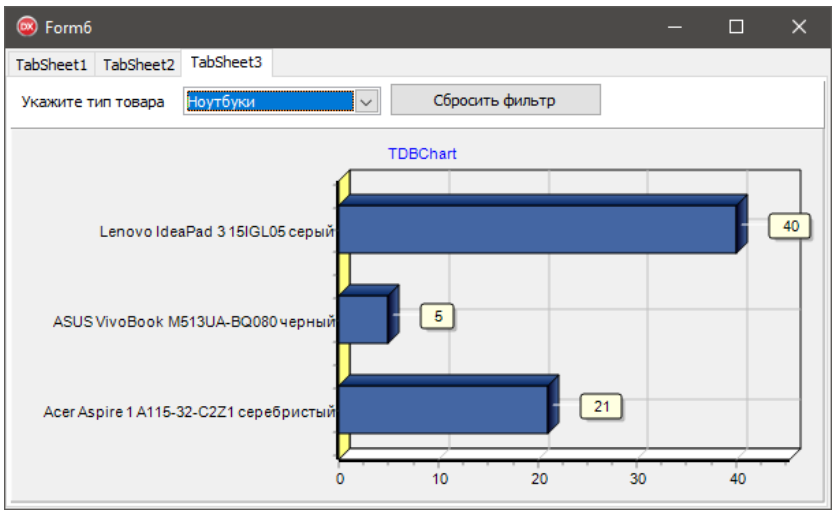


Рис. 5.11. График с выбором категории товаров

Далее необходимо настроить подписи к осям и задать название самому графику с использованием параметра **Title**. Пример настройки заголовка графика на рисунке 5.12, настройка заголовка вертикальной левой оси (**Left Axis**) на рис 5.13.

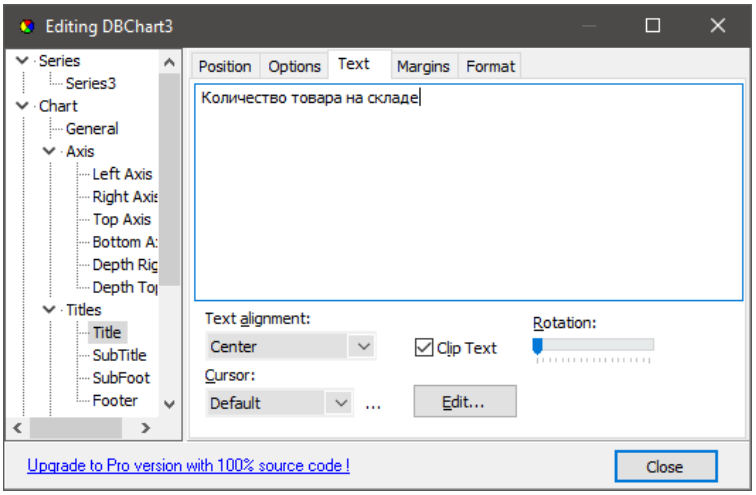


Рис. 5.12. Настройка заголовка графика

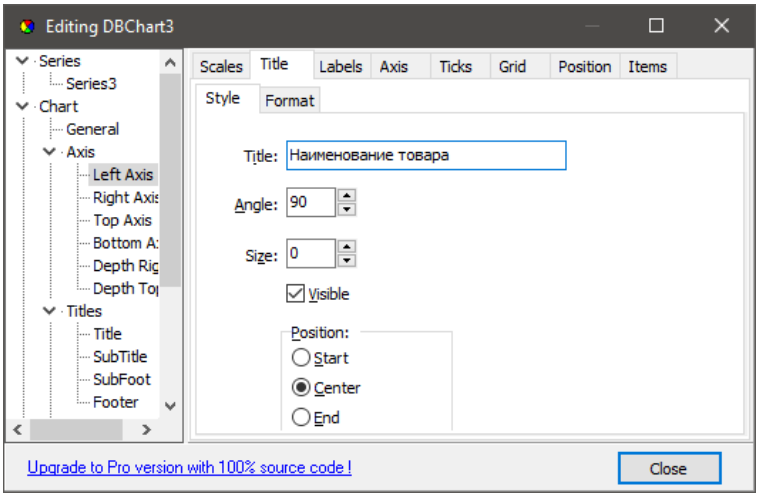


Рис. 5.13. Настройка заголовка левой оси

Подобным образом можно настроить надписи заголовков и других осей. При этом можно задавать позиционирование и угол поворота текста. Пример настроенного графика с активной фильтрацией по другому типу товара показан на рис 5.14.

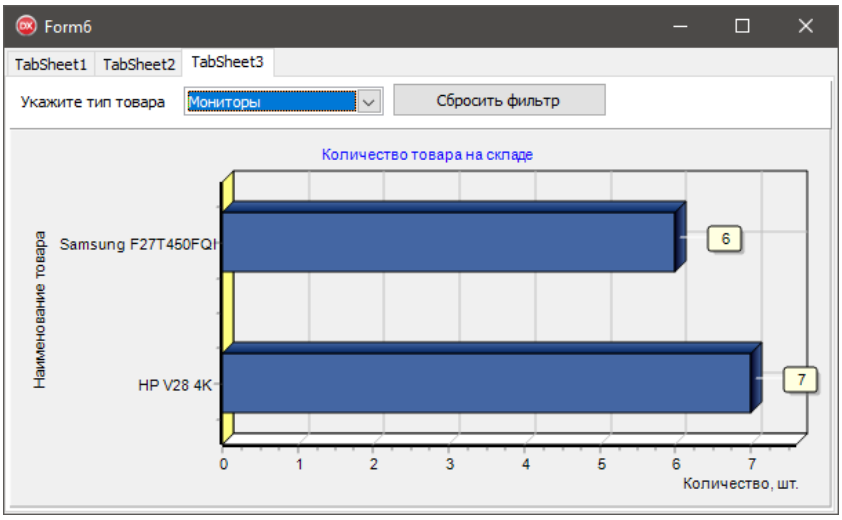


Рис. 5.14. График после настройки осей

Помимо графического представления данных из базы часто используются разнообразные отчеты (сводки информации).

5.2 Использование генератора отчетов

Генератор отчетов – это особый класс программ, востребованный как программистами, так и корпоративными пользователями. Подобных программ-генераторов создано немало, в рамках лабораторной работы ознакомимся с генератором отчетов **FastReport**. Он может быть установлен при помощи менеджера пакетов (**GetIt Pakage Manager**) доступного в меню **Tools**. Пример поиска генератора отчетов приведен на рис 5.15.

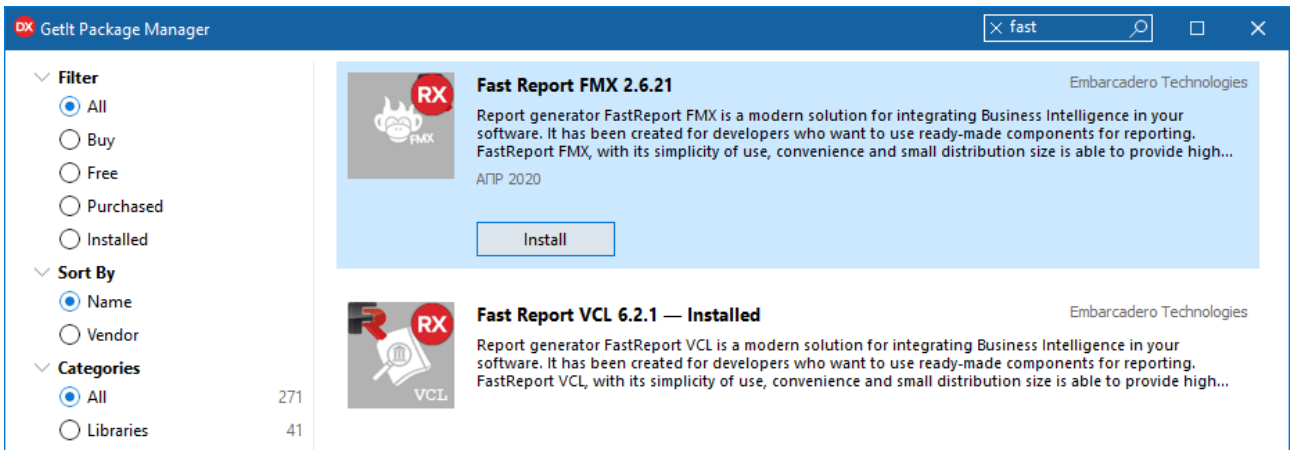


Рис. 5.15. Менеджер пакетов

Также можно скачать инсталлятор бесплатной версии с сайта https://www.fastreport.ru/ru/fastreport-embarcadero-edition/ .

FastReport представлен в виде набора компонентов и дизайнера, инструментарий которого включает огромное количество средств для визуального оформления, создания, настройки и изменения шаблонов отчетов.

Рассмотрим пример создания простого отчета **Список товаров** выводящего сведения из каталога товара (содержит перечень товаров с их стоимостью). Для этого разместите в **DataModule2** компонент **frxDBDataset.** В свойстве **DataSet** укажите **Q_v_goods_catalog**.

Разместим на **DataModule2** основной компонент **FastReport** – **frxReport**. Щелкните два раза на **frxReport**, откроется дизайнер отчетов, далее выберите пункт меню **File** затем **New report** (рис 5.16):

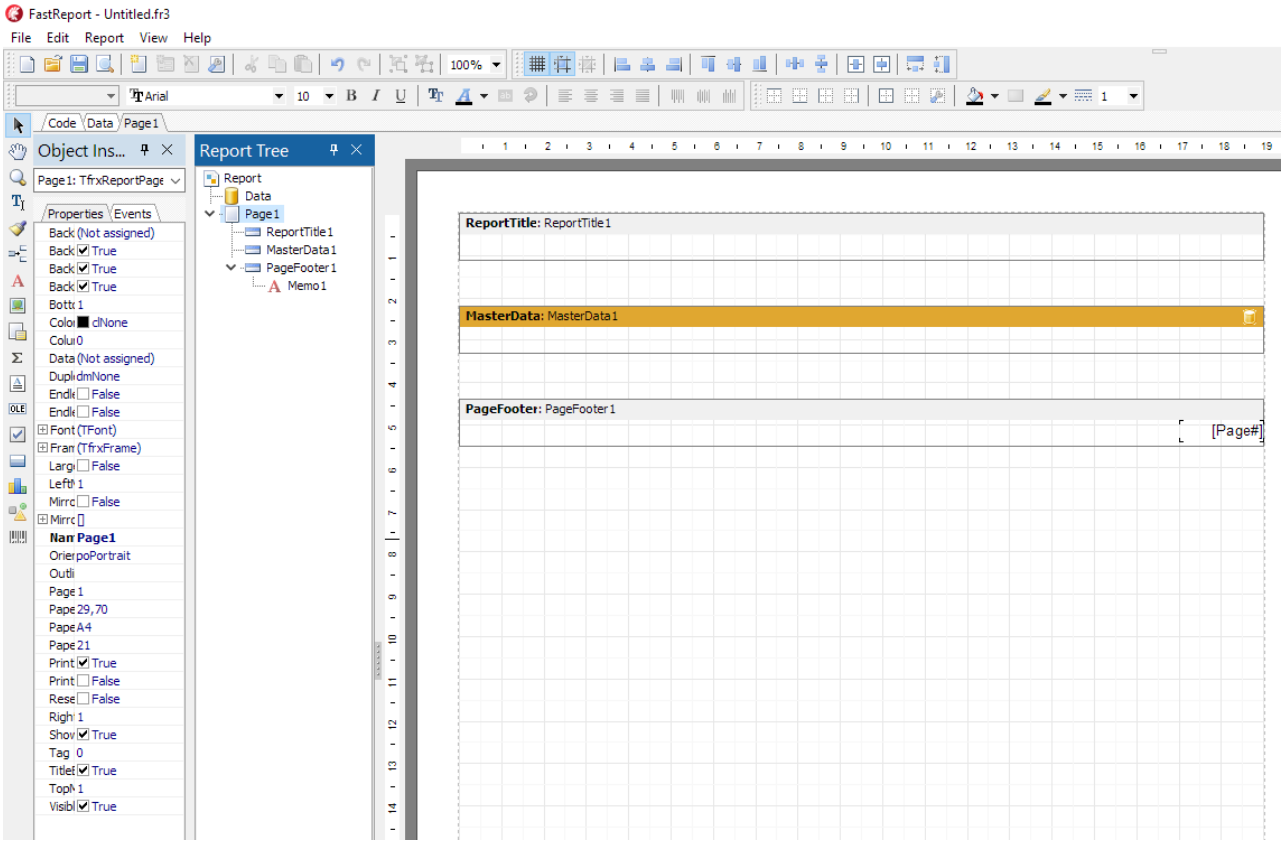


Рис. 5.16. Дизайнер запросов *frxReport*

Это было проделано, чтобы **FastReport** автоматически создал пустой шаблон с тремя бэндами: **Report Title**, **Master Data** и **Page Footer**. Чтобы данные запроса были видны в **FastReport**, необходимо разрешить их использование. Для этого выберем пункт меню **Report** затем **Data** и пометим наш набор данных галочкой (рис 5.17):

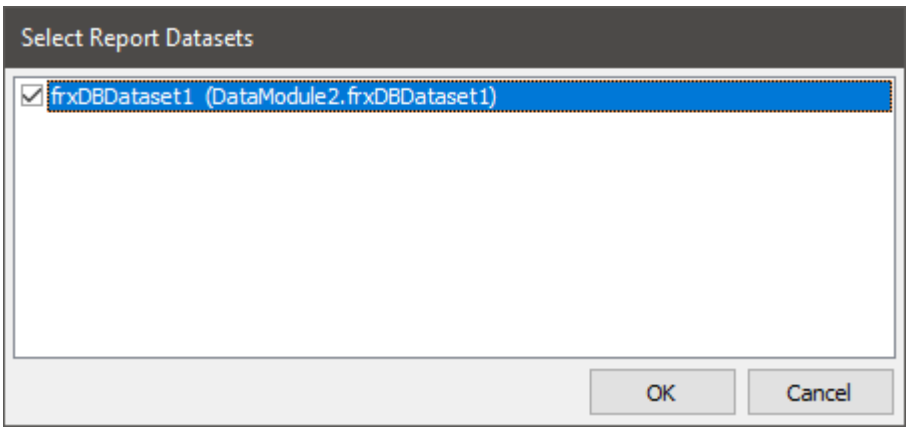


Рис. 5.17. Окно выбора набора данных для отчета

Приступим к созданию формы отчета. На бэнд **Report Title** поместим объект **Text** с текстом **Список товаров**. Бэнд **Master Data** подключим к нашему источнику данных. Это можно сделать тремя способами:

– сделать двойной щелчок на бэнде и выбрать НД;

– выбрать пункт **Edit...** из контекстного меню бэнда и указать НД;

– щелкнуть на свойстве **DataSet** в инспекторе объектов и указать НД.

Разместим на бэнде три объекта **Text Object**, которые будут отображать наименование товара, его тип и стоимость (рис 5.18):

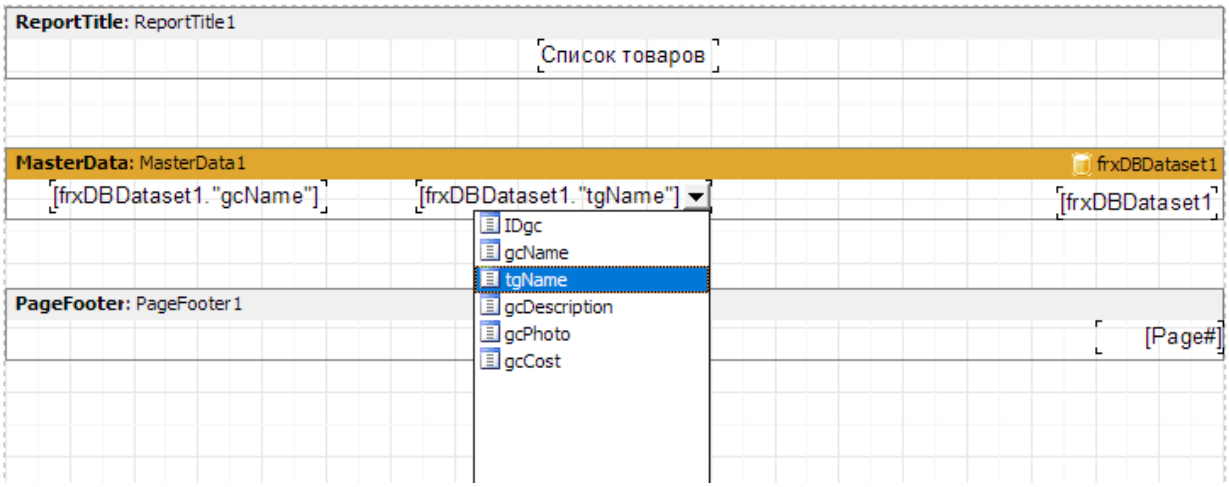


Рис. 5.18. Добавление полей в отчет

После запуска отчета на выполнение в меню **File** подпункт **Preview** или при помощи кнопки **F9** мы увидим сформированный отчет (рис 5.19):

Список товаров

Acer Aspire 1	Ноутбуки	25499
Lenovo IdeaPad 3	Ноутбуки	27955
ASUS VivoBook	Ноутбуки	49999
DEXP Mars E317	Системные блоки	41999
ASUS ROG Strix	Системные блоки	64990
Samsung F27T450FQI	Мониторы	15199
HP V28 4K	Мониторы	19399

Рис. 5.19. Фрагмент готового простейшего отчета

Далее сформируем отчет в форме **Этикетка** (это отчет, в котором данные о товаре выводятся построчно в небольшой области).

Осуществим создание этикеток к товарам с использованием данных из **Q_v_goods_catalog** (будем использовать уже имеющийся на **DataModule2** компонент **frxDBDataSet1)**. Разместим на **DataModule2** еще один компонент **frxReport (**имя по умолчанию **frxReport2**), запустим его дизайнер и создадим новый отчет.

Для формирования этикетки товаров необходимо вначале задать в свойствах страницы отчета отображение в виде двух колонок. Это можно сделать в окне настройки параметров страницы (рис 5.20), которая вызывается двойным щелчком мыши по пустой области страницы, либо через пункт **Edit** контекстного меню страницы.

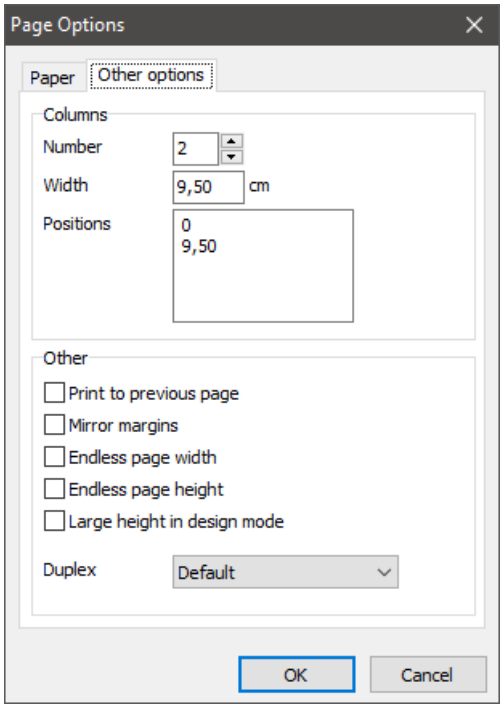
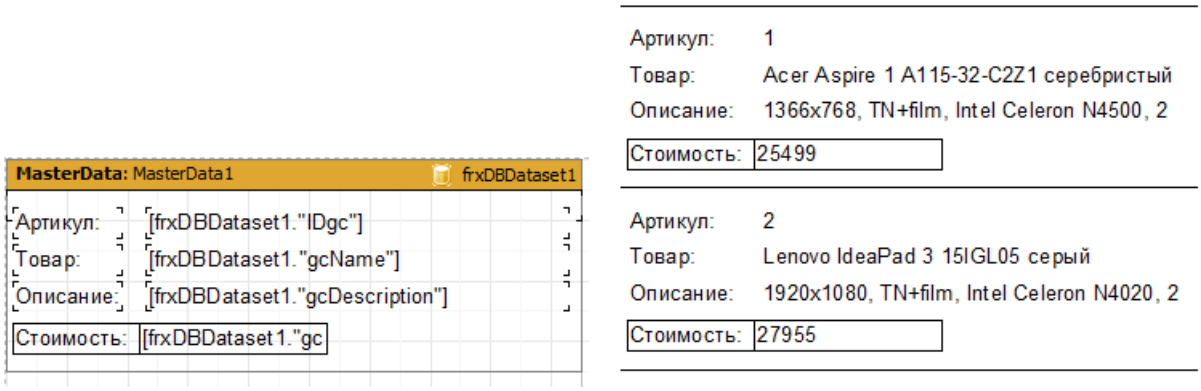


Рис. 5.20. Настройка параметров страницы в две колонки

Разместим элементы и настроим **MasterData** как показано на рис 5.21. Поскольку описание имеет довольно длинное содержание, то оно все не помещается при просмотре отчета.



а б

Рис. 5.21. Пример этикетки: *а* – макет; *б* – фрагмент готового отчета

Для того чтобы область описания содержала весь доступный текст необходимо задать свойству **StrechMode** объекта **Text**, отображающего поле **frxDBDataset1."gcDescription"**, значение **smActualHeight**.

У бэнда **MasterData1** включим растягивание путем установки свойства **Stretched** в значение **True**. Запустив отчет при предварительном просмотре, мы увидим следующие этикетки (рис. 5.22):

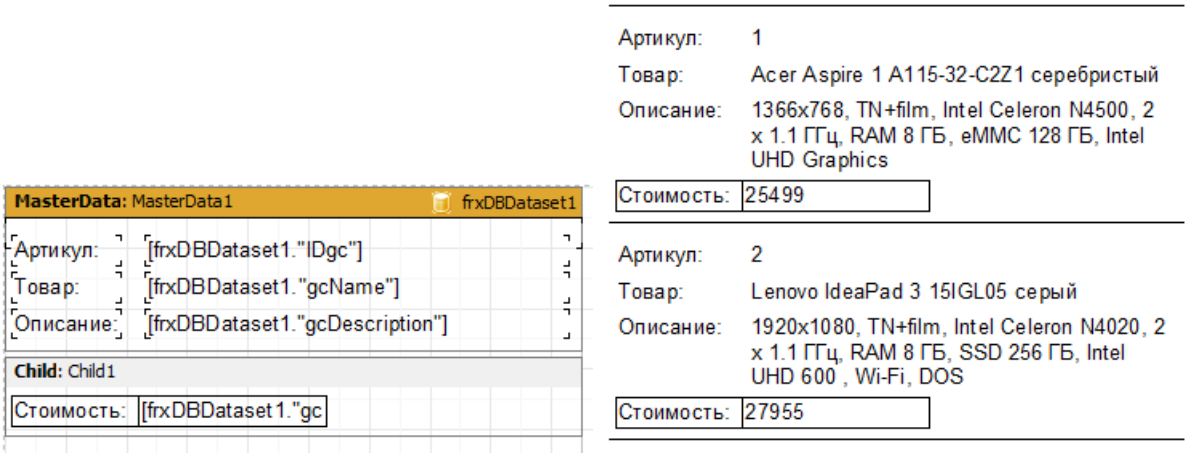
Рис. 5.22. Фрагмент готового отчета с растягиванием

Объект с данными поля **frxDBDataset1."gcDescription"** содержит длинный текст и поэтому он растянулся на несколько строк, а находящийся под ним объект, привязанный к полю также сместился ниже.

Это произошло потому, что в инспекторе объектов по умолчанию все объекты имеют включенное свойство **ShiftMode** (**ShiftMode = smAlways**). Такие объекты смещаются вниз, если над ними есть растяги-

ваемый объект, при этом высота, на которую смещается объект, зависит от того, сколько данных находиться в элементе.

Однако это неприемлемо, необходимо, чтобы объект с текстом **Стоимость** также смещался. Для этого в **FastReport** есть специальный тип бэнда – **Child**-бэнд, который привязывается к основному бэнду и выводится после него. Модифицируем наш отчет как показано рис. 5.23. Для того чтобы связать основной бэнд с дочерним, у бэнда **MasterData1** в инспекторе объектов установим свойство **Child** в значение **Child1**. Теперь каждый раз при печати основного бэнда будет выводиться и дочерний.



а б

Рис. 5.23. Пример отчета: *а* – макет; *б* – фрагмент готового отчета

Для того чтобы избежать переноса **Child**-бэнда на следующую страницу, т. е. его отрыва от основного бэнда, в инспекторе объектов у основного бэнда свойство **KeepChild** необходимо установить в значение **True**.

Мы рассматрели отчеты, в которых присутствовал только один датабэнд – **MasterData** (данные первого уровня). Это давало возможность печатать данные из одной таблицы БД. **FastReport** позволяет печатать отчеты, содержащие до шести уровней данных. Но в реальных приложениях редко приходится печатать отчеты с большой вложенностью данных, при этом, как правило, ограничиваются одним-тремя уровнями.

Создадим двухуровневый отчет, отражающий сведения об операциях. Он будет содержать данные из НД **Q_v_goods_operations** и **Q_v_operation_goods_list**: первый – это перечень операций, второй – список товаров, участвующих в операции. Для создания отчета разместим два компонента **frxDBDataSet** и один компонент **frxReport** (с именем **frxReport3**).

Теперь настроим компоненты. Один из компонентов **frxDBDataSet** назовем **op_master** (изменим свойства **Name** и **UserName**), а другой

op_detail. В свойстве **DataSet** укажем **Q_v_goods_operations** и **Q_v_operation_goods_list** для **op_master** и **op_detail** соответственно.

Поскольку при реализации лабораторных работ связь *Master-Detail* была реализована, то в текущий момент ничего не меняем в НД. При необходимости нужно организовывать связь *Master-Detail*.

Перейдем к созданию шаблона отчетов. В дизайнере отчета подключим источники данных и разместим на странице макета бэнды **MasterData** и **DetailData**, после чего настроим объекты **Text** для отображения данных из соответствующих полей (рис. 5.24). В **MasterData** будут отображаться данные полей: **op_master."oDateTime"** и **op_master."opType"**.

В **DetailData** будут отображаться данные полей: **op_detail."tgName"**, **op_detail."gcName"**, **op_detail."ocCount"**, **op_detail."gcCost"** и **op_detail."ogcFCost"**.

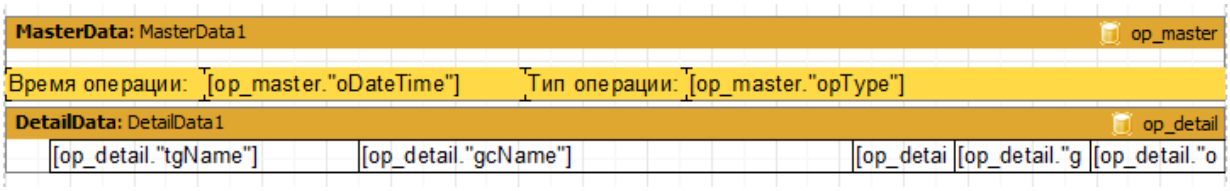
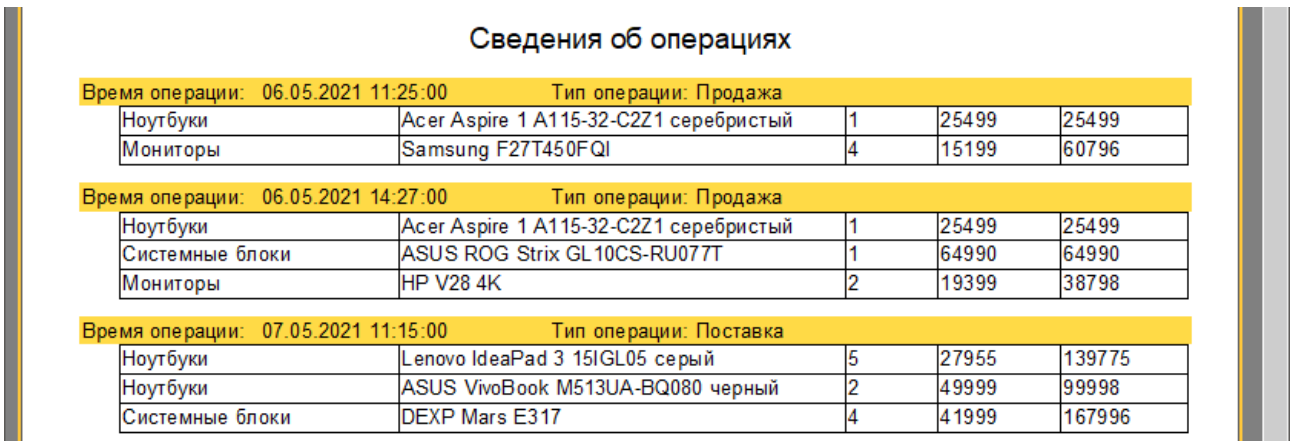


Рис. 5.24. Фрагмент макета отчета типа *Master–Detail*

Обратим внимание на то, что бэнд **MasterData** должен располагаться выше бэнда **DetailData**. В противном случае **FastReport** при запуске отчета сообщит об ошибке. Фрагмент готового *Master–Detail* отчета показан на рис 5.25:



Сведения об операциях

Время операции: 06.05.2021 11:25:00	Тип операции: Продажа			
Ноутбуки	Acer Aspire 1 A115-32-C2Z1 серебристый	1	25499	25499
Мониторы	Samsung F27T450FQI	4	15199	60796
Время операции: 06.05.2021 14:27:00	Тип операции: Продажа			
Ноутбуки	Acer Aspire 1 A115-32-C2Z1 серебристый	1	25499	25499
Системные блоки	ASUS ROG Strix GL10CS-RU077T	1	64990	64990
Мониторы	HP V28 4K	2	19399	38798
Время операции: 07.05.2021 11:15:00	Тип операции: Поставка			
Ноутбуки	Lenovo IdeaPad 3 15IGL05 серый	5	27955	139775
Ноутбуки	ASUS VivoBook M513UA-BQ080 черный	2	49999	99998
Системные блоки	DEXP Mars E317	4	41999	167996

Рис. 5.25. Фрагмент отчета типа *Master–Detail*

КОНТРОЛЬНЫЕ ВОПРОСЫ

Для чего применяется функция **CAST** в **SQL**?

Какие виды серий поддерживает компонент **DBChart**?

Продемонстрируйте отображение данных в виде графиков в разработанном приложении.

Опишите назначение источника данных для серии **Function**.

Для чего применяются генераторы отчетов?

Продемонстрируйте отображение данных в виде отчетов в разработанном приложении.