Лабораторная работа №7

ПОСТРОЕНИЕ ГРАФИКОВ ПО ДАННЫМ БД. ПОСТРОЕНИЕ ОТЧЕТНЫХ ФОРМ.

Задание:

Лабораторная работа №7 состоит из двух основных частей:

Часть 1: изучение компонент для построения графиков по данным БД.
1.1) Изучение компонента TDBChart;
1.2) Создание графиков по информации из БД;
1.3) Разработка формы с реализацией построения графиков для БД кафе.

Часть 2: изучение компонент для построения отчетных форм.
2.1) изучение компонент с палитры QReport;
2.2) разработка варианта отчетной формы по информации из БД кафе.

1 Создание графиков. Компонент TDBChart

Компонент TDBChart предназначен для построения графиков по информации из БД.

Для того, чтобы создать график, необходимо поместить на форму компонент TDBChart. В форме будет создана заготовка формирования графиков. Затем необходимо щелкнуть по этой заготовке 2 раза будет произведен переход в редактор графика.

В среде этого редактора; можно установить свойства графика и его серий. Содержимое редактора графика представляет собой табулированный блокнот. Для нового графика первой всегда показывается закладка Chart и для страницы Chart - закладка Series (рисунок 1.1).

Каждая из закладок на странице Chart предназначена для установки параметров того или иного компонента графика.

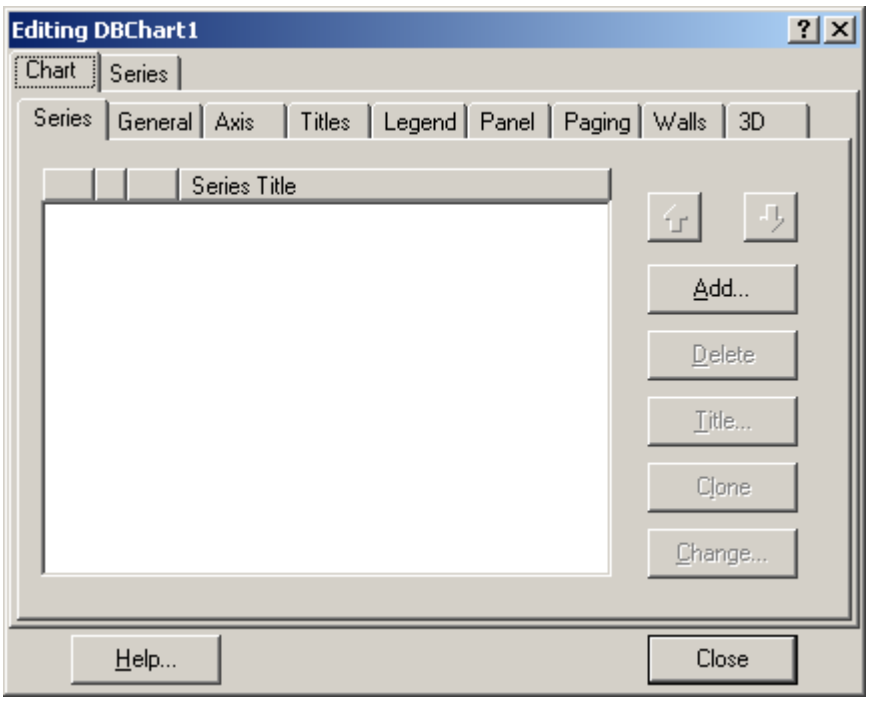


Рисунок 1.1 – Редактор графика

Series - содержит серии графика. Серией называется набор точек графика. На графике серии соответствует отдельная линия или ряд столбцов. Если в графике несколько серий, будет визуализировано несколько линий или рядов столбцов.

General - устанавливает общие параметры графика, такие как объемность графика, отступы от краев, возможность увеличения (Zoom) и др.

Axis - устанавливает свойства осей.

В области Show Axis определяется, для какой оси устанавливаются параметры - левой, правой, верхней или нижней. На странице, определяемой закладкой Scales, устанавливаются свойства масштаба значений по оси. Automatic устанавливает автоматическое масштабирование данных по оси - минимум и максимум вычисляются динамически, исходя из текущих значений серии. При отмене автоматического масштабирования можно установить автоматическое масштабирование минимального (Minimum) или максимального (Maximum) значения (отметка Auto). Для установки значения максимума и (или) минимума вручную следует нажать соответствующую кнопку Change. Шаг масштаба по оси выбирается автоматически, если в Desired Increment установлено значение 0. Установить фиксированное значение шага можно, нажав кнопку Change. Закладка Title позволяет установить текст заголовка по оси, угол расположения заголовка и шрифт, которым заголовок выводится. Закладка **Labels** задает параметры меток для оси. Закладка **Tiks** устанавливает параметры самой линии оси.

Titles - определяет заголовок графика, шрифт, выравнивание и др.

Legend - задает параметры легенды. Легенда - область графика, где приводится информация о графике и служит для пояснения графика.

Panel - определяет параметры панели, на которой располагается график.

Paging - устанавливает параметры многостраничного графика.

Walls - задает "стенку" графика.

1.1 Добавление серии в график и установка свойств серии в редакторе графика

На графике одновременно может располагаться несколько серий. В большинстве случаев их значения строятся по одинаковому закону и две и более серий одновременно показываются в графике для сравнения.

Чтобы добавить в график серию, следует на странице Chart, (закладка Series) нажать кнопку Add. После этого появится окно выбора типа серии (рисунок 1.2).

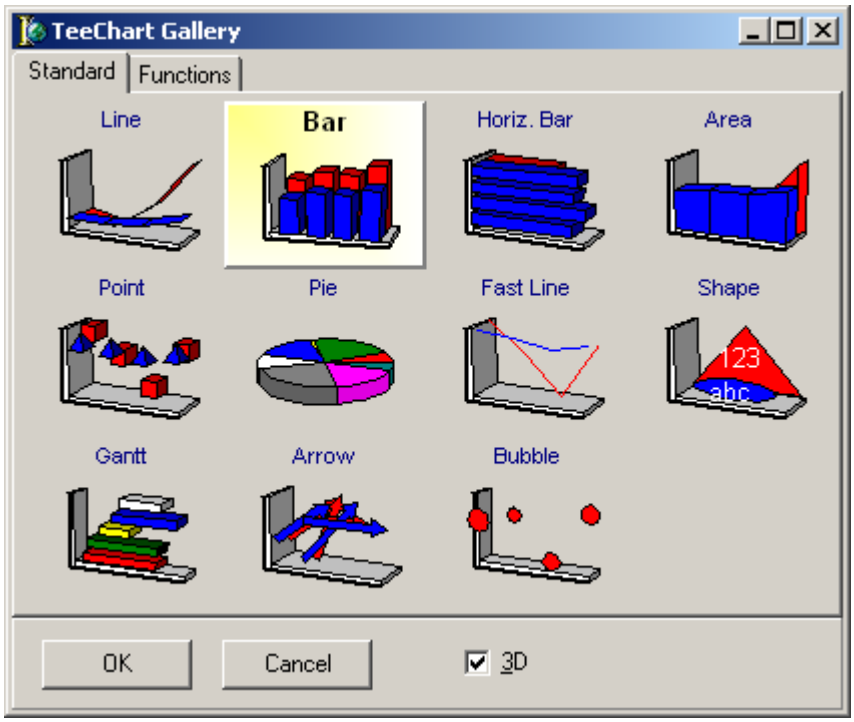


Рисунок 1.2 – Редактор графика – окно выбора типа серий

После выбора типа серии в график добавляется компонент, дочерний от базового типа TChartSeries - TLineSeries, TBarSeries, TPieSeries и т.д.).

Кнопка Add может использоваться для добавления других серий, кнопка Delete - для удаления текущей серии. После нажатия кнопки Title можно определить заголовок серии, кнопки Clone - создать новый экземпляр такой же серии в этом же графике, кнопки Change - изменить тип текущей серии.

Перейдем с закладки Chart на закладку Series. На этой странице представлен блокнот с закладками Format, General, Marks, DataSource.

Рассмотрим свойства серии, которые можно установить на страницах, соответствующих этим закладкам.

1.2 Выбор источника данных

Несомненно, главные свойства серии можно определить на странице DataSource. На ней определяется источник данных для серии. Выпадающий список ниже закладки позволяет определить тип источника данных для серии:

No Data - серии не назначается источник данных. Далее мы собираемся сделать, что программно. Кроме того, заготовленный шаблон серии может в разное время использоваться для показа данных из разных источников, которые мы также собираемся переключать программно во время выполнения.

Random Values - набор случайных чисел. Бывает полезен при формировании заготовки серии, источник данных которой мы собираемся установить позднее.

Function - функция (Copy, Average, Low, High, Divide, Multiply, Subtract, Add) - служит для построения графиков на основании данных в двух или более сериях.

DataSet - позволяет указать НД, значения полей (столбцов) которого будут использоваться для формирования точек серии. В качестве НД могут выступать компоненты TADOTable, TADOQuery и др.

Отметим, что не все типы серии требуют значений по осям Y или X. Для серий типа **Pie, Bar** можно указывать значения по **одной** из осей и значения меток Labels. В качестве меток могут использоваться символьные поля и поля типа даты и времени.

Если вернуться в редактор графика, на странице Series можно, помимо DateSource, увидеть закладки Format, General, Marks.

Их назначение:

Format - определяет свойства палитры, линий графика и т.д.

General - задает форматы данных.

Marks - устанавливает марки - значения в рамке над точками серии.

На странице Series (закладка Data Source) в качестве источника данных можно определить функцию. Функция обычно используется для показа отношений между другими сериями.

1.3 Добавление серии во время выполнения

Число серий, присутствующих в текущий момент в графике, во время выполнения можно определить при помощи метода компонента TDBChart

function SeriesCount;

Список серий графика содержится в его свойстве

property Series[Index:Longint];

где Index лежит в диапазоне 0..SeriesCount-1, поскольку отсчет серий идет с нуля.

Метод

procedure RefreshData;

обновляет данные в серии из наборов данных, которые служат их источником.

1.4 Работа с сериями. Компонент TChartSeries

Компонент TChartSeries является родительским типом для серий графика (для TLineSeries, TAreaSeries, TPointSeries, TBarSeries, THorizBarSeries, TPieSeries, TChartShape, TFastLineSeries, TArrowSeries, TGanttSeries, TBubbleSeries).

Свойства компонента TChartSeries

property Active - активизирует (показывает) серию в графике (значение true) и дезактивизирует (скрывает) серию (false). Например:

property DataSource - ссылается на компонент НД (TADOTable, TADOQuery и др.) или на другую серию, откуда берутся данные для показа в серии.

property HorizAxis - указывает, какая горизонтальная ось будет использована для серии. Значения: aTopAxis - верхняя горизонтальная ось; aBottomAxis - нижняя горизонтальная ось.

property Marks - описывает свойства марок серии. Т.е. значений в прямоугольниках, рисуемых для каждого значения серии. Свойства объекта Marks:

- property Arrow - задает свойства пера, рисующего марку. Свойства объекта Arrow:

- property Color - цвет линий;

- property Mode - способ рисования линий;

- property Style - стиль линий;

- property Visible - видимость линий;

- property Width - задает ширину линий;

- property ArrowLength - длина в пикселях линии, соединяющей марку с соответствующим изображением элемента серии. По умолчанию 16;

- property BackColor - определяет цвет фона марки.

- property Font - определяет шрифт, которым выводится информация внутри марки;

- **property ParentSeries** - содержит указатель на серию, к которой принадлежат марки;

- **property Style** - определяет содержимое марки.

Возможные значения свойства **Style**:

- **smsValue** - значения по осиУ (YValue), за исключением THorizBarSeries (XValue). - **smsPercent** - процентное значение, например "44%";

- **smsLabel** - показывает метку, ассоциированную с точкой графика, например "Сахарный песок" (при построении графика продаж по товарам); в том случае, если метки со значениями не ассоциированы, в марках выводятся сами значения;

- **smsLabelPercent** - показывает метку и процентное значение, например "Сахарный песок 44%".

- **smsLabelValue** - показывает метку и значение, например "Сахарный песок 9087". - **smsLegend** - показывает один из элементов легенды графика, список возможных значений доступен через свойство TextStyle;

- **sms Percent Total** - показывает процентное значение и общую сумму, от которой оно взято, например "44% от 20563".

- **smsLabelPercentTotal** - показывает метку, процентное число и общую сумму, например "Сахарный песок 44% от 20563";

- **smsXValue** - показывает значение по оси X (XValue), например "01.02.1997";

- **property Visible** -определяет, видимы ли (true) или нет (false) марки на графике.

- **property PercentFormat** - определяет формат показа процентных значений;

- **property SeriesColor** - определяет цвет, которым выводятся значения серии в графике.

- **property ShowInLegend** - определяет, показывать ли (true) легенду или нет (false) по умолчанию true;

- **property Title** - определяет заголовок серии; по умолчанию заголовок отсутствует, но он может быть назначен в редакторе графика (кнопка Title в окне Series).

- **property ValueFormat** - определяет формат показа значений серии; при прорисовке осей используется для форматирования меток, при прорисовке серии используется для форматирования значений, показываемых в марках;

- **property VertAxis** - определяет местоположение вертикальной оси - слева на графике (aLeftAxis) или справа (aRightAxis);

- **property XLabelsSource** - имя поля НД (или иного источника значений для серии), определяемого в свойстве DataSource. Содержимое этого поля служит для отображения значений по оси X. Поле должно быть типа, к которому применяется метод AsString. Если значение свойства опущено, значения по оси X не выводятся.

- **property Value[Index]** - обеспечивает доступ к элементу серии с индексом Index (значение в диапазоне 0… Count -1). Например:

- **property ValueSource** - указывает источник данных для формирования значений по оси X. В зависимости от того, каков источник данных для серии (свойство DataSource компонента TChartSeries), может содержать:

1) имя поля - числового типа, типа даты, времени, даты и времени; в этом случае свойство серии DataSource должно ссылаться на НД.

2) имя существующего TChart ValueList из другой серии; в этом случае свойство DataSource серии должно ссылаться на другую серию.

1.5 События компонента TChartSeries

Событие OnBeforeAdd - наступает перед добавлением точки в серию. В обработчике данного события может производиться анализ корректности добавляемых в серию точек. Наступает также при соединении серии с источником данных (TADOTable или TADOQuery).

Событие OnAfterAdd - происходит после добавления точки в серию.

Событие OnClearValues происходит при очистке серии от точек.

Событие OnClick - происходит при щелчке мышью на серии.

Событие OnGetMarkText -происходит при формировании марки для точки в серии. Обработчик может использоваться для изменения содержимого марки.

Задание 1. Построение графиков по БД кафе

- Будем строить графики по продажам типов блюд на форме Запросы (fmSQL).

- Перейдите на fmSQL.

- Нажмите правую кнопку на PageControl1 и выберете команду New Page. Появится новая закладка TabSheet. Для нее установите свойство Name в TabSheetDiagram, свойство Caption в Продажи по типам блюд.

-Разместите на TabSheetDiagram с закладки ADO компонент ADOQuery. Свойство Name укажите для него ADOQueryDiagram. Свойство Connection укажите в DMMain->ADOConnection1.

- В свойстве SQL для ADOQueryDiagram запишите запрос:

```
SELECT Тип, Sum(CCur(Количество*Цена*(1-Скидка))) AS Прибыль
```

FROM Продажи INNER JOIN (Блюда INNER JOIN Продано ON Блюда.Код_блюда = Продано.Код_блюда) ON Продажи.Номер_продажи = Продано.Номер_продажи
GROUP BY Блюда.Тип

- Свойство Active для ADOQueryDiagram установите в true.

-Разместите на TabSheetDiagram с закладки Data Controls компонент DBChart1. Свойство Name укажите DBChartDiagram.

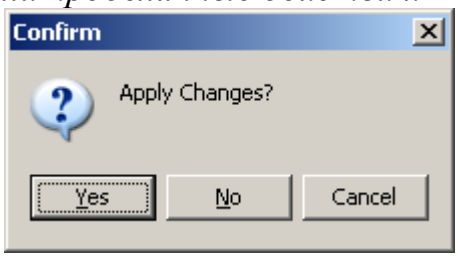
- Нажмите два раза левой кнопкой мыши на DBChartDiagram. Откроется редактор диаграмм.

- На закладке Series в редакторе DBChartDiagram нажмите Add. Выберете серию Bar и нажмите OK. Для нее укажите, нажав кнопку Title, вместо Series1 напишите SeriesBar.

- Для SeriesBar перейдите на закладку Series. На данной закладке перейдите на на закладку DataSource. На ней определяется источник данных для серии. Выпадающий список ниже закладки позволяет определить тип источника данных для серии. Укажите список DataSet. Выберете DataSet и из выпадающего списка выберете компонент ADOQueryDiagram.

- Укажите в Bar – поле Прибыль.

- Перейдите к закладке Chart. При этом будет запрошено применить ли проделанные действия. Нажмите в Aply Changes Yes.



Примечение:

*Для серий типа **Pie, Bar** можно указывать значения по **одной** из осей и значения меток Labels. В качестве меток могут использоваться символьные поля и поля типа даты и времени.*

- На закладке Series в редакторе DBChartDiagram нажмите Add. Выберете серию Pie и нажмите OK. Для нее укажите, нажав кнопку Title, вместо Series1 напишите SeriesPie.

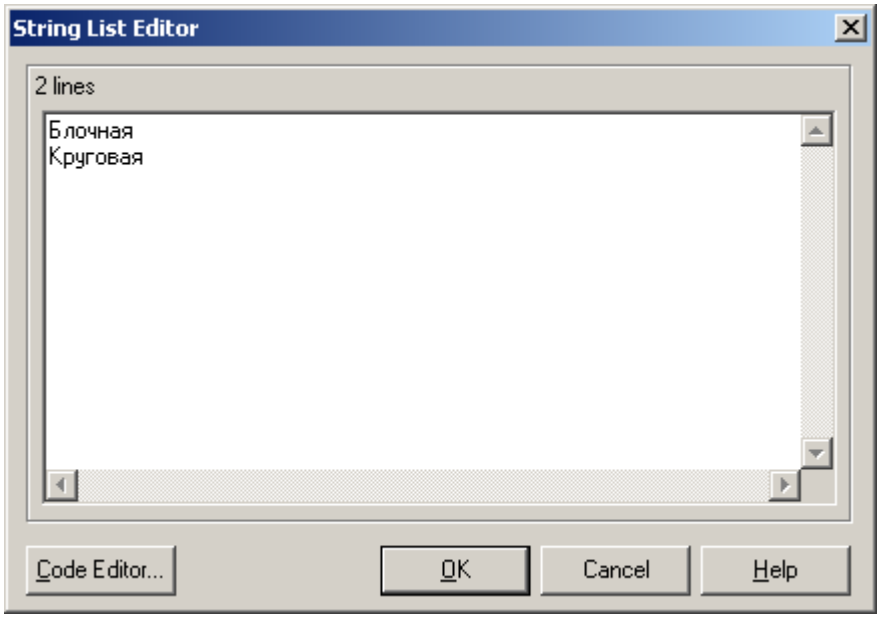
- Для SeriesBar перейдите на закладку Series. На данной закладке перейдите на на закладку DataSource. На ней определяется источник данных для серии. Выпадающий список ниже закладки позволяет определить тип источника данных для серии. Укажите список DataSet. Выберете DataSet и из выпадающего списка выберете компонент ADOQueryDiagram.

- Укажите в Pie – поле Прибыль.

- Перейдите к закладке Chart. При этом будет запрошено применить ли проделанные действия. Нажмите в Aply Changes Yes.

- Выберете закладку Titles . Укажите вместо TDBChart Продажи по типам.

Будем данные серии показывать в зависимости по нажатию радио-кнопки в группе радиокнопок. Для этого разместим на TabSheetDiagram компонент с палитры Standart RadioGroup. Укажите для него свойство Name в RadioGroupDiagram, свойство Caption укажите Тип диаграммы, свойство Align в alTop. В свойстве Items укажите:



Какая радиокнопка сейчас нажата содержит свойство ItemIndex. Его значения и будем отслеживать.

- Установите у RadioGroupDiagram свойство Columns в 2.

По нажатию первой кнопки будем показывать блочную диаграмму. Для этого в DBChartDiagram установите активной сформированную серию типа SeriesBar (оставив на ней галочку, отжав галочку на SeriesPie).

- Пропишите на событии OnClick у RadioGroupDiagram:

```
switch (RadioGroupDiagram->ItemIndex)
{

    case 0:
        DBChartDiagram->Series[0]->Active=true;
        DBChartDiagram->Series[0]->XLabelsSource="Тип";
        DBChartDiagram->Series[1]->Active=false;

        break;
   case 1:
        DBChartDiagram->Series[1]->Active=true;
```

```
        DBChartDiagram->Series[1]->XLabelsSource="Tun";
        DBChartDiagram->Series[0]->Active=false;
        break;

    }
```

- Будем на момент показа закладки для диаграмм TabSheetDiagram будем показывать блочную диаграмму.

- Для этого укажите в событии OnShow у TabSheetDiagram :

```
        RadioGroupDiagram->ItemIndex=0;
      RadioGroupDiagramClick(Sender);
```

- Укажите в событии OnShow у fmSQL:

```
        ADOQueryDiagram->Close();
        ADOQueryDiagram->Open();
```

- Установите у DBChartDiagram свойство Align в alClient.

- Укажите для SeriesBar в момент показа показывать и Типы и значения прибыли. Для этого нажмите два раза левой кнопкой мыши на DBChartDiagram.

- В редакторе выделите серию SeriesBar. Перейдите на закладку Series. В ней выберете Marks. В окне Style выберете Label and Value. Нажмите Close.

- Укажите для SeriesPie в момент показа показывать и Типы и значения прибыли в процентах. Для этого нажмите два раза левой кнопкой мыши на DBChartDiagram.

- В редакторе выделите серию SeriesPie. Перейдите на закладку Series. В ней выберете Marks. В окне Style выберете Label and Persent. Нажмите Close.

- Нажмите F9. Посмотрите результат формирования диаграмм по данным БД.

2 Компоненты для построения отчетов

Для подготовки отчетов можно использовать палитру QReport. На ней расположено множество элементов системы QuickReport.

2.1 Создание отчетов при помощи QReport

На странице палитры компонентов QReport расположено около двух десятков компонентов, применяемых для построения отчетов. "Главным" компонентом, является **TQuickRep**, определяющий поведение отчета в целом. Другие компоненты определяют составные части отчета. Перечислим основные:

- **TQRBand** - заготовка для расположения данных, заголовков, титула отчета и др.; отчет, в основном, строится из компонентов TQRBand, которые реализуют:
  - область заголовка отчета;
  - область заголовка страницы;
  - область заголовка группы;
  - область названий столбцов отчета;
  - область детальных данных, предназначенную для отображения данных самого нижнего уровня детализации;
  - область подвала группы;
  - область подвала страницы;
  - область подвала отчета.
- **TQRSubDetail** - определяет область, в которой располагаются данные подчиненной таблицы при реализации в отчете связи Master-Detail на основе существующей связи между ТБД;
- **TQRGroup** - применяется для группировок данных в отчете;
- **TQRLabel** - позволяет разместить в отчете статический текст;
- **TQRDBText** - позволяет разместить в отчете содержимое поля набора данных;
- **TQRExpr** - применяется для вывода значений, являющихся результатом вычисления выражений; алгоритм вычисления выражений строится при помощи редактора формул данного компонента;
- **TQRSysDate** - служит для вывода в отчете даты, времени, номера страницы, счетчика повторений какого-либо значения и т.д.;
- **TQRMemo** - служит для вывода в отчете содержимого полей комментариев;
- **TQRRichText** - служит для вывода в отчете содержимого полей форматированных комментариев;
- **TQRDBRichText** - служит для вывода в отчете содержимого полей форматированных комментариев, источником которых является поле набора данных;
- **TQRShape** - служит для вывода в отчете графических фигур, например, прямоугольников;
- **TQRDBImage** - служит для вывода в отчете графической информации, источником которой является поле набора данных;
- **TQRChart** - служит для встраивания в отчет графиков.

2.1.1 Компонент TQuickRep

Компонент TQuickRep определяет поведение и характеристики отчета в целом.

При размещении этого компонента в форме в ней появляется сетка отчета.

В дальнейшем в этой сетке располагаются составные части отчета, например, группы TQRBand и др.

Перечислим важнейшие свойства, методы и события компонента TQuickRep.

Свойство

property Bands состоит из множества логических значений (false/true), которые определяют включение в отчет отдельных видов составляющих:

- HasColumnHeader - заголовка столбцов отчета;
- HasDetail - детальной информации;
- HasPageFooter - подвала страницы;
- HasPageHeader - заголовка страницы;
- HasSummary - подвала отчета;
- HasTitle -заголовка отчета.

property DataSet – указывает на набор данных, на основе которого и создается отчет. Обычно для выдачи отчета используется один НД.

Если нужно вывести связанную информацию из нескольких таблиц БД, ее объединяют в одном НД при помощи оператора SELECT. В этом случае в качестве НД для отчета может использоваться компонент TADOQuery. Информацию из нескольких связанных НД можно включать в отчет, если эти наборы данных связаны в приложении отношением Master-Detail. В этом случае в качестве НД отчета указывается Master-набор, а ссылка на соответствующие Detail-наборы осуществляется в компонентах TQRSubDetail.

Если в отчет нужно включить информацию из несвязанных наборов данных, применяют композитный отчет, то есть отчет, составленный из группы других отчетов.

property Frame определяет параметры рамки отчета:

- Color - цвет линии рамки;
- DrawBottom - определяет, следует ли выводить линию снизу;
- DrawLeft- определяет, следует ли выводить линию слева;
- DrawRight- определяет, следует ли выводить линию справа;
- DrawTop- определяет, следует ли выводить линию сверху;
- Style - определяет стиль линии;
- Width - определяет ширину линии в пикселях.

property Page определяет параметры страницы.

property PrinterSettings определяет параметры принтера.

property PrintIfEmpty указывает (true), что следует печатать отчет даже в том случае, если он не содержит данных.

Методы:

procedure NewPage;
Выполняет переход на новую страницу. Может использоваться в обработчиках событий компонентов отчета BeforePrint или AfterPrint и не может - в обработчиках событий OnPrint, OnStartPage и OnEndPage.

procedure Print;
печатает отчет на принтере.

procedure PrinterSetup;
обеспечивает установки параметров принтера.

procedure Preview;
выводит отчет в окно предварительного просмотра.

Чтобы во время разработки отчета просмотреть в окне предварительного просмотра содержимое отчета в том виде, как он будет выводиться на печать, необходимо:
- выбрать отчет при помощи мыши;
- нажать правую кнопку мыши;
- во всплывающем меню выбрать элемент Preview.

Следует заметить, что при этом не будут видны некоторые данные, например, значения вычисляемых полей наборов данных. Они будут выводиться только во время выполнения.

События:
AfterPreview - наступает после закрытия окна предварительного просмотра отчета.
AfterPrint - наступает после вывода отчета на печать.
BeforePrint - наступает в момент генерации отчета, до выдачи окна предварительного просмотра отчета и до вывода отчета на печать.
OnEndPage -наступает в момент подготовки к генерации последней страницы отчета.
OnStartPage -наступает в момент подготовки к генерации первой страницы отчета.

2.1.2 Компонент TQRBand

Компоненты TQRBand являются основными составными частями отчета и используются для размещения в них статического текста и данных. Месторасположение компонента в отчете и его поведение определяются свойством **property BandType.**

Ниже перечислены возможные значения этого свойства:

- *rbTitle* - определяет компонент заголовка отчета. Информация, размещенная в компоненте TQRBand, располагается перед всеми другими частями отчета. Этот вид компонента TQRBand используется для вывода заголовочной информации отчета.

- *rbPageHeader* - определяет компонент заголовка страницы. Информация, размещенная в компоненте с этим значением свойства BandType, выводится всякий раз при печати новой страницы отчета прежде всех иных частей отчета (но после информации, размещенной в компоненте заголовка отчета - для первой страницы).

- *rbDetail* - компонент детальной информации. Выводится всякий раз при переходе на новую запись в НД отчета. Отчет печатается для всех записей НД, определяемого свойством отчета DataSet, начиная с первой записи и заканчивая последней. Позиционирование на первую запись и последовательный перебор записей в НД осуществляется компонентом TQuickRep автоматически.

- *rbPageFooter* - компонент подвала страницы. Выводится для каждой страницы отчета после всех иных данных на странице.

- *rbSummary* - компонент подвала отчета. Выводится на последней странице отчета после всей иной информации, но перед подвалом последней страницы отчета.

- *rbGroupHeader* - компонент заголовка группы. Применяется при группировках информации в отчете. Выводится всякий раз при выводе новой группы.

- *rbGroupFooter* - компонент подвала группы. Применяется при группировках информации в отчете. Выводится всякий раз при окончании вывода группы, после всех данных группы.

- *rbSubDetail* - компонент для выдачи детальной информации из подчиненного набора данных, при выводе в отчете информации из двух или более наборов данных, связанных в приложении при помощи механизма Master-Detail. Это значение присваивается компоненту автоматически, когда генерируется компонент TQRBand при размещении в форме компонента TQRSubDetail. Программа не должна устанавливать это значение в свойство BandType.

- *rbColumnHeader* - компонент для размещения заголовков столбцов. Размещается в отчете на каждой странице после заголовка страницы.

- *rbOverlay* - используется для совместимости с более ранними версиями отчетов.

Свойство

property Enabled указывает, печатается в отчете (true) или нет (false) информация, содержащаяся в компоненте TQRBand.

Свойство
property ForceNewPage указывает, должна ли информация в составе TQRBand всегда печататься с новой страницы (true) или нет (false).

Событие **BeforePrint -** наступает перед печатью информации, размещенной в области компонента TQRBand.

Компоненты TQuickRep и TQRBand являются минимально достаточными для создания простого отчета, не содержащего внутри себя группировок информации.

Компонент TQRBand, у которого в свойство BandType установлено значение rbColumnHeader, используется для представления заголовков столбцов. Заголовки столбцов определяются при помощи компонентов TQRLabel.

Компонент TQRBand, у которого в свойство BandType установлено значение rbPageHeader, используется для показа заголовка страницы. Он выводится для каждой новой страницы перед выводом другой информации.

Компонент TQRBand, у которого в свойство BandType установлено значение rbPageFooter, используется для показа подвала страницы. Он выводится для каждой страницы после вывода любой иной информации.

Информация в заголовке и подвале страницы может формироваться на основе статического текста (компоненты TQRLabel), значений полей (компоненты TQRDBText) и результатов вычисления выражений (компоненты TQRExpr).

Для компонента TQRBand для заголовка страницы (*rbPageHeader*) чтобы отчеркнуть линию вверху страницы необходимо установить в свойство компонента заголовка страницы Frame.DrawTop значение True, что обеспечивает вывод линии по верхнему краю области, занимаемой компонентом.

Аналогичным образом для компонента подвала страницы установив его свойство Frame.DrawBottom значение True, обеспечивается вывод линии по нижнему краю области, занимаемой компонентом.

Таким образом, вверху и внизу каждой страницы отчета выводятся линии.

2.1.3 Использование компонента TQRSysData для показа вспомогательной и системной информации

Компонент TQRSysData используется для показа вспомогательной и системной информации. Вид показываемой информации определяется свойством

property Data.

Ниже указаны возможные значения этого свойства:

- *qrsColumnNo* - номер текущей колонки отчета (для одноколоночного отчета всегда 1).
- *qrsDate* - текущая дата.
- *qrsDateTime* - текущие дата и время.
- *qrsDetailCount* - число записей в НД; при использовании нескольких НД - число записей в master-наборе. Для случая, когда НД представлен компонентом TQuery, эта возможность может быть недоступной, что связано с характером работы компонента TQuery, который возвращает столько записей, сколько необходимо для использования в текущий момент, а остальные предоставляет по мере надобности.
- *qrsDetailNo* - номер текущей записи в ИД. При наличии нескольких наборов - номер текущей записи в master-наборе.
- *qrsPageNumber* - номер текущей страницы отчета.
- *qrsPageCount* - общее число страниц отчета.
- *qrsReportTitle* - заголовок отчета.
- *qrsTime* - текущее время.

2.1.4 Группировки данных в отчете

Для группировки информации используется компонент TQRGroup. Его свойство Expression указывает выражение. В группу входят записи НД, удовлетворяющие условию выражения. При смене значения выражения происходит смена группы. Для каждой группы, если определены, выводятся заголовок группы и подвал группы. В качестве заголовка группы служит компонент TQRBand со значением свойства BandType, равным rbColumnHeader. В качестве подвала группы служит компонент TQRBand со значением свойства BandType, равным rbGroupFooter.

Свойство FooterBand компонента TQRGroup содержит ссылку на компонент подвала группы.

В заголовке группы, как правило, выводится выражение, по которому происходит группировка, и различные заголовки, если они нужны. В подвале группы обычно выводится агрегированная информация - суммарные, средние и т.п. значения по группе.

Поскольку свойство Expression не визуализирует значения выражения, необходимо разместить в группе компонент TQRExr и определить значение его свойства Expression.

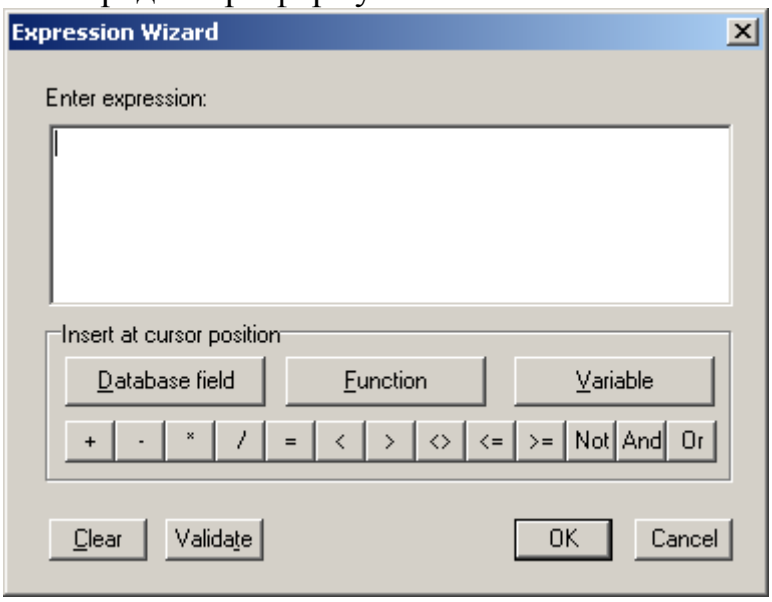
Часто внутри группы должны содержаться другие группы. В этом случае внутри одной группы выделяют другую группу посредством дополнительных компонентов TQRGroup.

2.1.5 Использование компонента TQRExpr для определения выражений

Выражение в отчетах формируется при помощи компонента TQRExpr.

Для того чтобы войти в редактор формул данного компонента, необходимо в инспекторе объектов выбрать свойство Expression и нажать кнопку в поле данных этого свойства.

Появится окно редактора формул:



Окно редактора формул компонента TQRExpr

Группа Function позволяет включить в выражение функцию, при этом выпадающий список Category определяет категорию функции (по умолчанию показываются все функции, значение ALL).

Группа Database Field позволяет включить в выражение поля набора данных, причем список Select Dataset определяет НД, а в списке Available Fields перечисляются поля текущего НД.

Группа Variable позволяет работать с различными переменными.

В группе без названия находятся кнопки, которые позволяют добавить константу, выражение, функцию или поле непосредственно в текст выражения.

2.1.6 Построение отчета на основе нескольких наборов данных, связанные в приложении как Master – Detail

Если необходимо выдавать отчет на основе более чем одной таблицы БД, можно поступить двумя способами:

1. В рамках компонента TADOQuery произвести соединение данных из нескольких таблиц БД в один НД, после чего определить в отчете нужные группировки;

2. Создать в приложении по одному НД на каждую таблицу БД, соединить эти наборы между собой связью Master-Detail (используя свойства MasterSource, MasterFields набора данных) и применить в отчете компонент (или несколько компонентов) TQRSubDetail для вывода информации из подчиненного (Detail) НД (или группы подчиненных НД); для вывода информации из основного (Master) НД, как и в обычных отчетах, применяется компонент TQRBand, у которого в свойстве BandType установлено значение rbDetail.

Компонент TQRSubDetail предназначен для показа информации в отчете из подчиненного НД. Его свойство

property DataSet указывает имя подчиненного НД, информация из которого будет выводиться в пространстве компонента TQRSubDetail. В остальном использование данного компонента аналогично использованию компонента TQRBand, у которого значение в свойство BandType установлено значение rbDetail.

Если необходимо определить заголовок и подвал для информации, группируемой в компоненте TQRSubDetail, следует воспользоваться свойством этого компонента.

property Bands - lанное свойство обладает двумя подсвойствами, указывающими на наличие или отсутствие заголовка и подвала.

2.1.7 Композитный отчет

Композитный (составной, сложный) отчет объединяет в себе несколько простых отчетов. При выдаче композитного отчета, входящие в его состав простые отчеты выводятся друг за другом.

Композитный отчет реализуется при помощи компонента TQRCompositeReport. В обработчике события OnAddReport ранее определенные простые отчеты добавляются в списковое свойство Report.

Задание 2. Разработка отчетной формы по информации из БД кафе

Создадим отчет, состоящий из информации по продажам блюд.

Будем формировать отчетную форму для выбранной пользователем продажи на форме продаж (fmSales).

- Перейдите на форму fmSales.

- Рядом с кнопкой поиска продаж (btnFindSale) разместите с закладки Additional кнопку BitBtn для отчета по продажам. Для нее укажите:

- свойство Name в btnReportSale;

- через свойство Glyph у btnReportSale в инспекторе объектов задайте соответствующую картинку для кнопки отчета по продаже;

- свойство Caption у btnReportSale задайте в Отчет.

Для формирования отчета создайте новую форму в проект.

- Нажмите в меню File/New/Form.

- Свойство Name у новой формы напишите в fmReport.

- Сохраните форму в проект. Для этого нажмите File/Save all и в диалоговом окне сохраните новую форму в той же папке, что и другие модули проекта. При сохранении укажите новому модулю имя Report.cpp.

- Перейдите на форму fmReport.

- С закладки QReport поместите компонент QuickRep1 – на нем будем формировать отчет. Свойство Name укажите у него QuickRepSale.

- Подключите на форме fmReport модуль данных. форму fmSQL.

Для этого, находясь на форме fmReport, выберете в меню File/Include Unit Hdr…(или нажмите Alt+F11). В диалоговом окне Use Unit выберете модуль DM.

- У QuickRepSale укажите свойство DataSet (набор данных) в DMMain->ADOT_Sales.

- Разместите на QuickRepSale с закладки QReport компонент QRBand1. Свойство BandType компонента QRBandl по умолчанию установлено в значение rbTitle, то есть компонент QRBand1 определяет заголовок отчета. Установим свойство BandType компонента QRBandl в rbPageHeader (заголовок страницы), чтобы данный заголовок был на каждой странице отчета, а не только в его заголовке.

- На QRBand1разместите с закладки QReport компонент QRLabel1 (метка). Задайте ей свойство Caption в Кафе.

- На QRBand1разместите с закладки QReport компонент QRLabel2 (метка). Задайте ей свойство Caption в г.Томск.

- На QRBand1разместите с закладки QReport компонент QRLabel3 (метка). Задайте ей свойство Caption в Спасибо за покупку.

- Задайте QRBand1 свойство Color в выбранный цвет, например, в clMoneyGreen.

- Чтобы текст наших меток не выделялся белым на выбранном цвете для QRBand1 задайте для меток QRLabel1, QRLabel2, QRLabel3 свойство Transpared в true.

- Задайте для меток QRLabel1, QRLabel2, QRLabel3 свойство Font (шрифт) в жирный шрифт fsBold (через Style) высотой 14 (через Size) пунктов цвета clNavy (через Color).

- Под QRBand1 разместите QRBand2. Свойство BandType компонента QRBand2 установим в rbDetail. Установим при этом у QRBand2 укажите

свойство Color в любой цвет (можно выбрать из стандартной палитры или добавить цвет самим), например clActiveBorder.

- Разместите в пространстве отчета, занимаемом компонентом QRBand2, компонент QRLabel4 (метка). Задайте ей свойство Caption в Чек на продажу номер:

- Разместите в пространстве отчета, занимаемом компонентом QRBand2 рядом с QRLabel4 компонент TQRDBText (статический текст из поля набора данных) с именем QRDBText1. Установим в свойство Dataset этого компонента в DMMain->ADOT_Sales, а в свойство DataField укажите из списка полей Номер_продажи.

- Разместите в пространстве отчета, занимаемом компонентом QRBand2 рядом с QRDBText1 компонент QRLabel5. Задайте ей свойство Caption в От:

- Разместите в пространстве отчета, занимаемом компонентом QRBand2 рядом с QRLabel5 компонент TQRDBText (статический текст из поля набора данных) с именем QRDBText2. Установим в свойство Dataset этого компонента в DMMain->ADOT_Sales, а в свойство DataField укажите из списка полей Дата.

- Разместите в пространстве отчета, занимаемом компонентом QRBand2 под компонетом QRLabel4 компонент QRLabel6. Задайте ей свойство Caption в Оформил сотрудник:

- Разместите в пространстве отчета, занимаемом компонентом QRBand2 рядом с QRLabel6 компонент TQRDBText (статический текст из поля набора данных) с именем QRDBText3. Установим в свойство Dataset этого компонента в DMMain->ADOT_Sales, а в свойство DataField укажите из списка полей Сотрудник.

- Задайте для QRLabel4, QRDBText1, QRLabel5, QRDBText2, QRLabel6, QRDBText3 свойство Transpared в true, свойство Font (шрифт) в жирный шрифт fsBold (через Style) высотой 10 (через Size).

- Добавьте в отчет компонент TQRSubDetail (имя QRSubDetail1).

- Установите свойство DataSet у QRSubDetail1 в DMMain->ADOT_DisheSale.

- Разместим в пространстве отчета, занимаемом компонентом QRSubDetail1 компоненты TQRDBText (статический текст из поля набора данных) рядом с которыми разметим соответствующие компоненты TQRLabel (метка):

- QRLabel7 (свойство Caption этого компонента в значение Блюдо) и QRDBText4 (свойство Dataset этого компонента в значение DMMain->ADOT_DisheSale, а свойство DataField в Блюдо);

- QRLabel8 (свойство Caption этого компонента в значение Количество) и QRDBText5 (свойство Dataset этого компонента в значение DMMain->ADOT_DisheSale, а свойство DataField в Количество);

- QRLabel9 (свойство Caption этого компонента в значение Цена) и QRDBText6 (свойство Dataset этого компонента в значение DMMain->ADOT_DisheSale, а свойство DataField в Цена);

- QRLabel10 (свойство Caption этого компонента в значение Скидка) и QRDBText7 (свойство Dataset этого компонента в значение DMMain->ADOT_DisheSale, а свойство DataField в Скидка);

- QRLabel11 (свойство Caption этого компонента в значение Сумма) и QRDBText8 (свойство Dataset этого компонента в значение DMMain->ADOT_DisheSale, а свойство DataField в Сумма);

- Для QRDBText8 установите в свойстве Font курсив fsItalic высотой 10 (Size) пунктов без изменения цвета.

- Добавьте в отчет компонент TQRBand (имя QRBand3).

- Свойство BandType компонента QRBand3 установим в rbGroupFooter.

- Разместите в пространстве отчета, занимаемом компонентом QRBand3, компонент QRLabel12. Установите свойство Caption этого компонента значение «Итого по продаже:» и установите в свойстве Font жирный шрифт fsBold высотой 10 (Size) без изменения цвета.

- Рядом с QRLabel12 разместите QRLabel13. Свойство Name для нее задайте в QRLabelItogo. У QRLabelItogo в свойстве Font укажите жирный шрифт fsBold высотой 12 (Size) без изменения цвета.

Перед обработчиками событий объявите переменную денежного типа для формирования итоговой цены для продажи. Для этого в модуле Report.cpp

после блока

```
__fastcall TfmReport::TfmReport(TComponent* Owner)
        : TForm(Owner)
{
}
```

Пропишите:

```
Currency s;
```

- В событии AfterPrint у QRSubDetail1 пропишите переменную, суммирующую по полю Сумма:

```
s += DMMain->ADOT_DisheSale->FieldByName("Сумма")->AsCurrency;
```

- В событии BeforePrint у QRBand2 пропишите:

```
s = 0;
```

- В событии OnPrint у QRLabelItogo пропишите:

```
Value = CurrToStr(s)+" руб.";
```

- Укажите у компонента QRSubDetail1 свойство FooterBand в QRBand3.
- Укажите у QRBand3 свойство LinkBand в QRSubDetail1.

Отчет сформирован.

Покажем сформированный отчет в режиме предварительно просмотра по нажатию кнопки Отчет на форме Продажи (fmSales).

- Перейдите на форму fmSales.
Подключите на форме fmSales форму fmReport.
Для этого, находясь на форме fmSales, выберете в меню File/Include Unit Hdr…(или нажмите Alt+F11). В диалоговом окне Use Unit выберете модуль Report.

-Перейдите на форму fmSales.
- В событии OnClick у btnReportSale пропишите:

```
int id = DMMain->ADOT_Sales->RecNo;
DMMain->ADOT_Sales->Filter="Номер_продажи="+DMMain->ADOT_Sales->FieldByName("Номер_продажи")->AsString;
DMMain->ADOT_Sales->Filtered=true;
fmReport->QuickRepSale->Preview();
DMMain->ADOT_Sales->Filtered=false;
DMMain->ADOT_Sales->RecNo = id;
```

- Нажмите F9.
- Посмотрите сформированный отчет по выбранной продаже.

По аналогии формируются и другие отчеты.

Задание 3. Оформление отчета к лабораторной работе

Оформить отчёт со следующим содержанием:

1. Титульный лист.
2. Цель работы.

3. Постановка задачи.
4. Краткая теория и ход выполнения заданий.
5. Описание результатов.

Приведите скриншоты полученной программы (с диаграммами и отчетной формой) и листинг программы.

6. Заключение (выводы).