Міністерство освіти і науки України

Рівненський державний гуманітарний університет

Факультет математики та інформатики

Кафедра інформатики та прикладної математики

Курсова робота на тему:

#### Засоби виводу інформації на принтер в об’єктно-орієнтованому середовищі програмування Delphi

Виконав:

студент IV курсу

факультету математики та інформатики

групи МІ-41 Карпович О. І.

Науковий керівник:

викл. Гаврюсєв С. М.

Рівне 2004

Зміст

Вступ.

§1. Аспекти використання стандартних компонентів ООС програмування Delphi для створення звітної документації.

* 1. опис компонентів – QReport, PrintDialog та PrintSetupDialog;
  2. створення звітів за допомогою Quick Reports;
  3. інше програмне забезпечення для побудови звітів
  4. §2. Приклади створення звітів

###### Висновок.

Список використаної літератури.

**Вступ**

За сучасного стрімкого розвитку інформаційних технологій часто виникає потреба у створенні нестандартниж можливостей для покращення дизайну і якості программного забезпечення. Метою роботи є огляд можливостей компонентів об’єктно-орієнтованого середовища програмування Delphi при виводі інформації на принтер. Зокрема огляд можливостей генератора звітів Quick Report. Він один з найбільш використовуваних генераторів, поряд з ним використовуються ще такі генератори як: Cristal Report та Fast Report. За допомогою закладки QReport в середовищі програмування Delphi здійснюється друк. Закладка QReport дозволяє працювати з базами даних. Крім QReport розглянемо можливості таких компонентів як Print Dialog та Print Setup Dialog.

**1.1. Опис основних компонентів Quick Reports.**

На сторінці *QReport* палітри компонентів розташовується більш двадцяти компонентів, застосовуваних для побудови звітів.

Найважливішим є компонент *QuickRep ,* що визначає побудову звіту в цілому. За допомогою інших компонентів створюються складові частини звіту.



Компонент *QRBand* являє собою спеціальну панель для розташування даних, заголовків, назви звіту та ін. Даний компонент є одним з основних. З його допомогою реалізуються такі області звіту:



область заголовка звіту;

область заголовка сторінки;

область заголовка групи;

область назв стовпців звіту;

область детальних даних;

область підгрупи;

область підсторінки;

область підзвіту.

Компонент *QRStringBand* має те ж призначення, що і *QRBand.* Відмінність полягає в тому, що даний компонент має вбудований список рядків *Items*. Вміст цього списку стає видимим у режимі друку або попереднього перегляду, якщо на компонент *QRStringBand* поміщений компонент *QRExpr.* Для кожного рядка в списку *Items* виводиться окрема смуга *QRStringBand.*

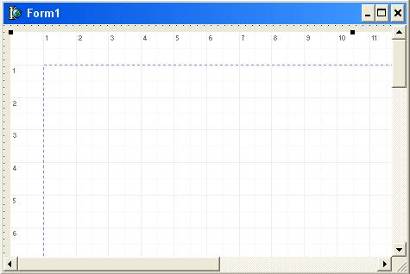


Інші компоненти, представлені на сторінці Палітри компонентів, приведені в табл.

Група компонентів *QRXXXXFilter* являє собою набір фільтруючих компонентів, використовуваних для перетворення звіту в текст, HTML-сторінку і т.п. при друці звіту.

Компоненти для створення звітів

|  |  |  |
| --- | --- | --- |
| Компонент | Призначення | |
| *QRSubDetail* | Являє собою область, у якій розташовуються дані підлеглої таблиці при реалізації в звіті зв'язку між таблицями БД | |
| *QRChildBand* | Виконує функції дочірньої смуги, що прив'язується до батьківської смуги і служить для її розширення. Будь-яка смуга може бути визначена як батьківська шляхом установки її властивості HasChild у True | |
| *QRGroup* | Застосовується для групування даних у звіті | |
| *QRLabel* | Дає можливість розмістити в звіті довільний текстовий рядок | |
| *QRDBText* | Використовується для висновку в звіті вмісту полючи набору даних | |
| *QRExpr* | Застосовується для висновку значень, що є результатом обчислення виражень. Формули для обчислення виражень задаються за допомогою спеціального редактора формул даного компонента | |
| *QRSysData* | Служить для відображення в звіті системних значень, таких як дата, час, номери сторінок і т.п. | |
| *QRMemo* | Дозволяє вставити в звіт багаторядковий текст | |
| *QRExprMemo* | Використовується для створення багаторядкових полів, що обчислюються | |
| QRRichText | | Дає можливість додати в звіт текст у форматі RTF | |
| *QRShape* | | Застосовується для відображення в звіті найпростіших геометричних фігур (квадрат, коло і т.д.) | |
| *QRImage* | | Служить для висновку в звіті графічної інформації, джерелом якої є поле набору даних | |
| *QRPreview* | | Використовується для створення нестандартних вікон попереднього перегляду. Стандартне вікно перегляду викликається за допомогою методу Preview компонента *QuickRep* | |
| *QRChart* | | Служить для додавання в звіт графіків | |



**Рис. 1. Форма додатка з розміщеної на ній порожньою сіткою звіту**

Розглянемо більш докладно компонент *QuickRep.* При розміщенні цього компонента на формі в ній з'являється сітка звіту (див. мал. 1). Надалі в цій сітці розташовуються різні складові частини звіту, наприклад, смуги *QRBand,* напису *QRLabel* і т.д. Нижче перераховані основні властивості компонента *QuickRep.*

Властивість *Bands (тип TQuickRepBands)* є об'єктом, що містить набір логічних властивостей, що після установки їх у значення *True* включають у звіт наступні елементи:

*HasColumnHeader* - заголовки стовпців;

*HasDetail* - детальну інформацію;

*HasPageFooter* - підсторінки;

*HasPageHeader* - заголовок сторінки;

*HasSumtvary* - підзвіту;

*HasTitle* - заголовок звіту.

Властивість *DataSet (тип TDataSet)* безпосередньо вказує на набір даних *(Table. Query і ін.)*, на основі якого створюється звіт. Джерело даних *DataSource* при цьому не використовується.

Властивість *Frame (тип TQRFrame)* визначає параметри рамки звіту, такі як наявність ліній (ліворуч, праворуч, зверху, знизу), колір ліній, стиль ліній (суцільна, пунктирна і т.п.), товщина лінії в пікселях.

Властивість *Options (тип TQuickReportQptions)* містить безліч наступних логічних значень:

1. *HasFirstHeader* - дозволяє друкувати заголовок першої сторінки;
2. *HasLastFooter* - дозволяє друкувати останню підсторінку;
3. *Concession* - дозволяє стискати звіт при виводу його в метафайл.

Властивість *Page (тип TQRFage)* визначає параметри сторінки. Усі ці параметри доступні у вікні *Report Setting* (див. мал. 2).

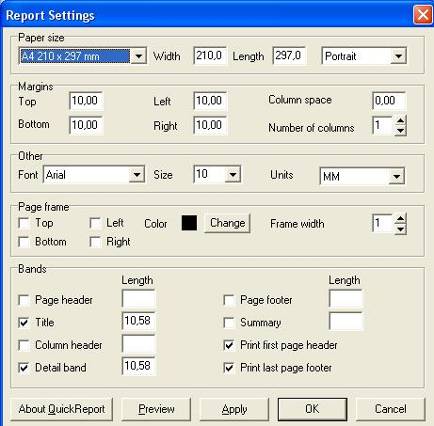
Властивість *Print If Empty (тип Boolean)*, встановлене в *True*, дозволяє друкувати звіт у тому випадку, якщо він не містить даних.

Властивість *ReportTitle (тип String)* визначає ім'я звіту (але не заголовок!). Дана властивість використовується для ідентифікації звіту в завданні на мережевий друк і може застосовуватися для вибору одного з декількох доступних звітів.

Властивість *ShowProgress (тип Boolean)*, встановлене в *True*, дозволяє показувати індикатор процесу друку звіту.

Властивість *SnapToGrid (тип Boolean)* прив'язує розташовувані в звіті компонента до сітки, якщо має значення *True*.

Властивість *Units (тип TQRUnits)* визначає одиниці виміру відстаней у звіті: дюйми, міліметри, пікселі, внутрішні одиниці компонента *QuickRep* (рівні 0,1 мм), символи тексту.



**Рис. 2 Вікно встановлення параметрів звіту**

Властивість *Zoom (тип Integer)* дозволяє встановити масштаб відображення звіту (у відсотках від його розмірів на листі папера) на етапі розробки. Може приймати значення в діапазоні від 1 до 300. Значення даної властивості не враховується при друці звіту або в режимі його попереднього перегляду.

Існує можливість встановлювати багато властивостей звіту за допомогою спеціального редактора властивостей, для виклику якого необхідно вибрати пункт *Report Setting* у локальному меню компонента *QuickRep* (див. мал. 2).

У цьому вікні представлені наступні групи параметрів.

Група *Paper size* задає характеристики сторінки: формат (наприклад, *А4 210x297 mm),* ширина *(Width),* довжина *(Length)* і напрямок друку *(Portrait* або *Landscape).*

Група *Margin* вказує поля звіту: зверху *(Тор),* знизу *(Bottom),* ліворуч *(Left),* праворуч *(Right),* а також кількість стовпчиків *(Number of columns)* і відстань між ними *(Column Space).*

Група *Other* дозволяє задати шрифт *(Font),* його розмір *(Size)* і використовувані одиниці виміру *(Units).*

Група *Page frame* встановлює властивості рамки:, наявність лінії зверху *(Тор),* знизу *(Bottom),* ліворуч *(Left),* праворуч *(Right),* колір ліній *(Color)* і їхню товщину *(Width).*

Група *Bands* визначає наявність різних смуг: заголовок сторінки *(Page header),* заголовок звіту *(Title),* заголовок стовпчиків *(Column header),* смуга для детальної інформації *(Detail band),* підсторінки *(Page footer),* підзвіту *(Summary).* Після вибору типу і висоти смуги вона з'являється в звіті, якщо натиснуто кнопку *Apply* або вікно закрите кнопкою *ОК.* Елементи *Print first page header* і *Print last page footer* керують, відповідно, друком заголовку на першій сторінці звіту і підзвіту на його останній сторінці.

У якості основних можна виділити наступні методи компонента *QuickRep.*

Метод *NewColumn* реалізує вивід інформації в наступному стовпчику звіту, а якщо визначено один стовпчик, то на наступній сторінці.

Метод *NewPage* забезпечує вивід інформації на наступній сторінці звіту.

Метод *Prepare* готує звіт для виводу у файл. Вивід звіту у файл виробляється в наступному порядку. Спочатку звіт підготовляється шляхом виклику методу *Prepare*, потім зберігається у файлі методом *Save* об'єкта *QRPrinter*, після чого цей об'єкт знищується і властивості *QRPrinter* присвоюється значення *NIL*. Це робиться, як показано нижче.

**QuickRep1.Prepare;**

**QuickRep1.QRPrinter.Save('Report.QRP');**

**QuickRep1.QRPrinter.Free;**

**QuickRep1. QRPrinter := NIL;**

Метод *Print* друкує звіт на принтері.

Метод *PrintBackground* дозволяє роздрукувати звіт у фоновому режимі. Після завершення друку викликається оброблювач події *OnAfterPrint*.

Метод *Preview* виводить стандартне вікно попереднього перегляду (див. мал. 3).

Метод *PrinterSetup* викликає стандартне вікно встановлення параметрів принтера.

*Компонент ORBand.* У даному розділі варто також згадати про найважливіші властивості компонента *QRBand,* тому що дані компоненти є основними складовими частинами звіту. Отже, розглянемо кілька специфічних властивостей компонента *QRBand.*

Властивість *AlignToBottom (тип Boolean)*, що має значення True, вказує на те, що смуга буде розташовуватися безпосередньо над підсторінкою, у той час як звичайно вона розташовується праворуч або знизу від попередньої смуги.

Властивість *BandType (тип TQRBandType)* визначає призначення смуги і може приймати одне з таких значень:

1. *rbTitle* - друкується на першій сторінці звіту і, як правило, містить його назву;
2. *rbPageHeader* - друкується вгорі кожної сторінки і містить її заголовок;
3. *rbDetail* - містить інформацію з набору даних і автоматично повторюється для кожного запису, починаючи з першої і закінчуючи останньою;
4. *rbPageFooter* - зберігає підсторінку і виводиться внизу кожної сторінки:
5. *rbSummary* - являє собою підзвіт і виводиться в його кінці;
6. *rbSubDetail* - містить детальну інформацію з підлеглого набору даних у випадку зв'язку між кількома таблицями БД;
7. *rbColuimHeader* - являє собою заголовки стовпців і розміщається на кожній сторінці звіту після заголовка сторінки;
8. *rbGroupFooter* - зберігає підгрупу, застосовується при угрупованнях інформації в звіті і виводиться наприкінці кожної групи;
9. *rbGroupHeader* - містить заголовок групи і виводиться на початку кожної групи;
10. *rbOverlay* - використовується для зворотної сумісності з попередніми версіями *QuickReport.* He рекомендується використовувати даний тип без особливих на те основ;
11. *rbChild* - резервується для використання компонентів *QRChildBand.* Даний тип не слід установлювати вручну.

Властивість *Enabled (тип Boolean)* дозволяє друкувати смуги, якщо має значення *True*.

Властивість *ForceNewColumn (тип Boolean)*, встановлене в *True*, вказує на те, що смуга буде друкуватися в наступному стовпчику.

Властивість *ForceNewPage (тип Boolean)*, встановлене в *True*, указує, що смуга буде друкуватися на новій сторінці.

Властивість *HasChild (тип Boolean)* визначає, чи буде дана смуга мати дочірню смугу. Якщо цій властивості привласнити значення *True*, то в звіті буде автоматично створена дочірня смуга.

# При виводі інформації на принтер в середовищі Delphi використовуються ще компоненти *PrintDialog* та *PrintSetupDialog.*

# Компонент *PrintSetupDialog* призначений для на лаштування параметрів принтера, не має особливих властивостей, тому що ці параметри суттєво відрізняються для різних видів принтерів. На основі цього компонента можна створювати свої власні компоненти для окремих принтерів.



Компонент *PrintDialog* відображає стандартне вікно друку Windows. В ньому мона задати різні параметри друку, які визначаються наступними властивостями:



|  |  |
| --- | --- |
| Властивість | Значення |
| Collate | Маркер. Розібрати по копіях |
| Copies | Число копій |
| FromPage | Номер сторінки з якої починається друк |
| MaxPage | Максимальне число сторінок, яке може бути надруковане |
| MinPage | Мінімальне число сторінок, яке може бути надруковане |
| Options | Другорядні параметри налаштування |
| PrintRange | Вигляд сторінок документу, що друкується. Можливі значення: prAllPages(всі сторінки); prSelection(сторінки вибраного фрагменту); prPageNums(сторінки з діапазону FromPage/ToPage) |
| PrintToFile | Має значення True, якщо вивід має здійснюватись не на принтер, а у файл |
| ToPage | Номер сторінки на якому закінчується друк |

**1.2. Створення звітів за допомогою Quick Reports.**

Створення найпростішого звіту не представляє особливих проблем. Для цього достатнім є використання тільки компонентів *QuickRep* і *QRBand.*

*Створення найпростішого звіту.* Розглянемо побудову простого звіту на невеликому прикладі.

*Приклад.* Нехай є таблиця students.db, у якій зберігаються дані про студентів інституту. У цій таблиці представлені такі поля: Номер студентського квитка *(Nom\_stud)*, Прізвище, ім'я, по батькові *(Fio)*. Група *(Grup)*. Потрібно створити найпростіший звіт, що містить заголовок і зведення з зазначеної таблиці БД.

Спочатку на форму поміщається компонент *Table*, що зв'язується з таблицею students.db. Таблиця відкривається, тобто властивості *Active* цього компонента привласнюється значення *True*. На форму поміщається компонент *QuickRep.* і його властивості *DataSet* встановлюється значення *Tablel*. У звіт додається компонент *QRBand.* При цьому його властивість *BandType* по замовчуванню має необхідне значення, а саме - *rbTitle*, тобто цей компонент буде містити заголовок звіту. Зверху на смугу заголовка поміщається компонент *QRLabel,* властивості *Caption* якого задається значення, що визначає заголовок звіту (наприклад, "Список студентів"). Крім того, при бажанні можна змінити шрифт, яким буде виводитися заголовок звіту. Це робиться за допомогою зміни значення властивості *Font*. Після цього в звіт додається ще один компонент *QRBand.* і його властивості *BandType* встановлюється значення *rbDetail*. У цій смузі буде відображатися вміст таблиці students.db. Потім на дану смугу поміщаються три компоненти *QRDBText,* що зв'язуються з полями таблиці БД - *Norn stud*. *Fio*, *Grup*. Для цього властивості *DataSet* кожного компонента *QRDBText* задається значення *Tablel*, а властивості *DataField* присвоюється ім'я відповідного поля.

Переглянути отриманий звіт, можна, клацнувши по ньому правою кнопкою миші і вибравши пункт *Preview* в відкритому контекстному меню. Зразковий вид вікна попереднього перегляду даного звіту показаний на мал. 5.

Якщо необхідно, щоб вікно попереднього перегляду відкривалося при створенні форми, досить створити такий оброблювач події *OnCreate* даної форми:

**procedure TForml.FormCreate(Sender: TObject);**

**begin**

**QuickRepl.Preview;**

**end;**

Для того щоб при закритті вікна попереднього перегляду закривалася і форма, на якій розташований звіт, можна використовувати такий оброблювач події *OnAf terPreview* компонента *QuickRep:*

**procedure TForml.QuickRep1AfterPreview (Sender: TObject);**

**begin**

**Forml.Close;**

**end;**

У результаті створення двох приведених оброблювачів подій може бути отримане найпростіший додаток, що забезпечує можливість перегляду і друку сформованого звіту.

*Включення виражень у звіт.* Для того щоб включити в звіт дані, одержувані в результаті обчислення деякого виразу, використовується компонент *QRExpr* . Вираз формуються за допомогою спеціального редактора, що викликається у вікні Інспектора об'єктів кнопкою у поле даних властивостей *Expression* вищевказаного компонента (див. мал. 6).



Вираз вводиться і редагується в поле *Enter expression* даного редактора. Вираз може складатися з імен полів, змінних, числових значень, функцій перетворення, арифметичних і логічних операцій, а також операторів порівняння. Імена полів додаються у вираження за допомогою допоміжного вікна, що викликається натисканням кнопки *Database field,* функції - за допомогою вікна, що відкривається при натисканні кнопки *Function,* а змінні - натисканням кнопки *Variable.* Також за допомогою відповідних кнопок додаються логічні, арифметичні оператори й оператори порівняння.

Розглянемо приклад складання виразу з наступним виводом результату його виконання в звіті. Нехай є таблиця shop.db, що містить найменування товарів, що є в магазині (поле *Tovar*), і його вартість (поле *Price*). Припустимо, що ціна товарів виражена в доларах США. Потрібно представити ціну в гривнях, з огляду на те, що курс дорівнює 5,35 гривень за один долар США. Крім цього, необхідно, щоб у звіті відображалася грошова величина, у якій виражена ціна товару. Таким чином, потрібно перетворити чисельне значення вартості товару до строкового типу, після чого до отриманого рядка додати під рядок “грн.”. Реалізувати це можна в такий спосіб. Після того як у звіт доданий компонент *QRExpr,* варто відкрити вікно редактора формул.

Спочатку в редакторі необхідно натиснути кнопку *Function.* У лівому списку вікна, що відкрився при цьому, потрібно вибратикатегорію *Other*, а потім у правому списку - функцію STR, що перетворить числове значення в строкове. Після цього варто натиснути кнопку *Continue,* щоб перейти до введення параметрів функції. У результаті відкриється спеціальне вікно введення параметрів (див. мал. 7).

Параметр можна ввести безпосередньо в поле одно строкового редактора або натиснути кнопку , розташовану праворуч від поля введення. Після натискання даної кнопки на екрані з'явиться початкове вікно редактора формул. Тому що, відповідно до поставленої задачі, необхідно буде перетворювати в рядок значення поля таблиці БД, то для задання його як параметра даної функції варто натиснути кнопку *Database field* і вибрати поле *Price* у списку полів таблиці *Tablel*. Тепер потрібно натиснути кнопку *ОК,* щоб вставити поле, потім *ОК,* щоб закрити вікно редактора формул, і ще раз *ОК,* щоб завершити введення параметра. У поле *Enter expression* буде сформована частина формули, що має вид:



STR( Tablel. Price)

Перш ніж перетворити значення поля *Price* до строкового виду, його необхідно помножити на 5.35, щоб представити ціну не в доларах, а в гривнях. З цією метою курсор встановлюється в позицію безпосередньо після імені *Price*, потім на панелі *Insert at cursor position* натискається кнопка *«\*»* і вручну вводиться значення 5.35.

Після цього залишилося тільки додати вказівку грошової одиниці. Для цього курсор переводиться в кінцеву позицію, натискається кнопка *«+»* і вручну вводиться підрядок “ грн. ”. У результаті сформована формула здобуває такий вид:

STR(Tablel.Price\*5.35)+ ' грн.'

У вікні попереднього перегляду звіту буде відображатися інформація приблизно на кшталт показане на мал. 8.

У цьому звіті відображається найменування товару, потім вміст поля *Price* у первісному вигляді і, нарешті, перелічена за допомогою компонента *QRExpr* вартість товару. Ціна виводиться двічі, щоб проілюструвати різницю між початковим і перетвореним значенням.

*Відображення системної інформації.* Для відображення допоміжної і системної інформації використовується компонент *QRSysData*. Вид відображуваної інформації визначається значенням властивості *Data* даного компонента. Ця властивість може приймати одне з таких значень:



1. *qrsColumnNo* - номер поточного стовпчика звіту:
2. *qrsDate* - поточна дата;
3. *qrsDataTinie* - поточна дата і час;

*qrsDetaiICount* - кількість записів у наборі даних; при використанні декількох наборів даних - кількість записів у головному наборі. Якщо набір даних представлений компонентом *Query,* ця можливість може бути недоступна, що зв'язано з характером роботи даного компонента, що повертає стільки записів, скільки необхідно для використання в даний момент часу;

1. *qrsDetailNo* - номер поточного запису в наборі даних;
2. *qrsPageNumber* - номер поточної сторінки звіту;
3. *qrsPageCount* - загальна кількість сторінок звіту;
4. *qrsReportTitle* - заголовок звіту;
5. *qrsTime* - поточний час.

Компонент *QRSysData,* як правило, розміщається в заголовку або підсторінці звіту.

*Групування даних у звіті.* Для групування даних використовується компонент *QRGroup .* Умова групування зберігається у властивості *Expression*, інакше кажучи, у групу будуть входити записи набору даних, що задовольняють встановленій умові. Роль такої умови виконує ім'я поля набору даних, на основі якого буде здійснюватися групування записів. Для кожної групи виводиться її заголовок. Як заголовок групи використовується компонент *QRGroup,* а як під компонент *QRBand* зі значенням властивості *BandType*, рівним *rbGroupFooter*. При цьому властивість *FooterBand* компонента *QRGroup* повинне обов'язково містити посилання на компонент підгрупи. Як правило, у заголовку групи виводиться вираження, що групує, а в підзаголовку - різна інформація, така як, наприклад, сумарне або середнє значення по групі в цілому.



Важливою умовою є те, що угрупування буде здійснюватися тільки на основі поточного індексу. Іншими словами, необхідно, щоб як поточний індекс був використаний індекс, побудований за полем (або полями, у випадку створення вкладених груп даних), що використовується для групування записів у звіті.

Розглянемо групування даних у звіті на прикладі. Тут же буде проілюстровано, як можна застосувати деякі інші компоненти.

*Приклад.* Нехай, є таблиця, у якій зберігається інформація про співробітників фірми. Нехай дана таблиця має такі поля: Табельний номер *(Tab\_nom)*, Ім'я співробітника *(Name)*, Відділ *(Otdel)*, Посада *(Position)*, Оклад *(Salary)*. Необхідно скласти звіт, що відображає дані з цієї таблиці. При цьому потрібно здійснити групування всіх записів за номерами відділів, у яких працюють співробітники. Також необхідно відображати назви стовпців звіту й у підсторінці виводити номер сторінки і поточну дату. Наприкінці звіту повинна бути виведена інформація про загальну кількість співробітників підприємства.

На першому кроці на форму міститься компонент *Table1* і його властивості *IndexFieldNames* або *indexName* установлюється відповідне значення, що містить посилання на індекс, побудований за полем *Otdel*.

Після того як на форму поміщений компонент *QuickRep1* і його властивість *DataSet* задана як *Tablel*, у звіті розміщаються такі смуги:

1. смуга заголовка звіту - компонентів *QRBandl*, у якого властивість *BandType* має значення *rbTitle*;
2. смуга заголовків колонок - компонентів *QRBand2* із значенням властивості *BandType*, рівним *rbCol*- *ColumnHeader*;
3. смуга групи - компонентів *QRGroupl*, його властивість *FooterBand* одержує значення *QRBand4* (посилання на підзвіт - див. нижче), а властивості *Expression* присвоюється значення *Tablel*.*Otdel* (встановлює поле, на основі якого буде вироблятися групування даних);
4. смуга детальної інформації - компонентів *QRBand3* для якого властивість *BandType* встановлена в *rbDetail*;
5. смуга підгрупи - компонентів *QRBand*. властивості *BandType* якого задане значення *rbGroupFooter*;
6. смуга підзвіту - компонентів *QRBand*, властивість *BandType* має значення *rbSuiranary*;
7. смуга підсторінки - компонентів *QRBand6,* зі значенням властивості *BandType*, рівним *rbPageFooter*

На смугах звіту розміщаються компоненти *QRLabel*, що містять різні пояснювальні записи, у тому числі заголовок звіту, заголовки стовпців і т.д.

На смугу групи *QRGroup1* міститься компонент *QRExpr, у* якого властивості *Expression* необхідно присвоїти значення *Tablel.Otdel*, що дозволяє відображати значення поля *Otdel* для поточної групи даних.

У смузі детальної інформації розміщаються компоненти *QRDBText*.що посилаються на відповідні поля набору даних, а саме на поля *Tab nom. Name*. *Position* і *Salary*.

У підгрупі додається компонент *QRExpr.* Його властивості *Expression* присвоюється значення *SUM (Tablel. Salary)* для підсумовування зарплати всіх співробітників відділу.

У підзвіті також включається компонент *QRExpr,* властивості *Expression* якого потрібно задати значення *COUNT*, що дозволяє порахувати загальне число виведених у звіті записів або, іншими словами, загальна кількість співробітників підприємства.

На смузі підсторінки необхідно розмістити два компоненти *QRSysData.* Властивості *Data* одного з них привласнити значення *qrsDate*, а іншого - *qrsPageNumber*. Таким чином, у підсторінці буде відображатися поточна дата і номер поточної сторінки звіту.

На мал. 9 показаний зразковий вид вікна попереднього перегляду сформованого звіту.

В звіті може застосовуватися і більш складне групування даних, оскільки кожна група може, у свою чергу, мати вкладену групу.

*Побудова звіту для зв'язаних наборів даних.* У тих випадках, коли потрібно створити звіт на основі даних, одержуваних з декількох таблиць БД, можна здійснити двома способами.

1. За допомогою компонента *Query* об'єднати дані з декількох таблиць в один набір даних, після чого будувати звіт звичайним чином, аналогічно тому, як було показано вище.
2. Створити кілька наборів даних (по одному на кожну таблицю) і з'єднати ці набори між собою зв'язком майстер-детальний, використовуючи їх властивості *MasterSource* і *Master* *Fields*. Далі для відображення інформації зі зв'язаних наборів даних у звіті застосовуються компонент *QRBand* (для головної таблиці) і компонент *QRSubDetail* (для підлеглої таблиці). Саме особливості застосування цього способу і будуть розглянуті нижче.

Компонент *QRSubDetail*  спеціально призначений для включення в звіт інформації з детального набору даних. Властивість *DataSet* цього компонента повинна містити посилання на детальний набір даних.



Нехай у БД бухгалтерії підприємства маються дві таблиці. Перша таблиця firm.db містить інформацію про співробітників і має такі поля: Табельний номер *(Tab\_nom)*, Ім'я співробітника *(Name)*, Відділ *(Otdel)*, Посада *(Dolgn)*, Оклад *(Salary)*. В другій таблиці zarplata.db зберігається помісячна інформація про виплату зарплати кожному співробітникові з початку поточного року і включає такі поля: Порядковий номер запису *(Ncmer)*, Табельний номер *(Tab\_nom)*, Місяць *(Mes)*, Кількість відпрацьованих днів *(Dni)*, Виплачена зарплата *(Zarpl)*.

Таблиці firm.db і zarplata.db знаходяться у відношенні один-до-одного, тобто одному співробітникові можуть відповідати кілька фактів виплати зарплати.

Для створення звіту, насамперед, потрібно розмістити на формі два компоненти *Table1* і *ТаЫе2*, зв'язавши їх, відповідно, з таблицями firm.db і zarplata.db. Крім того на формі міститься компонент *DataSource,* зв'язаний з компонентом *Tabl1*. Після цього між наборами даних встановлюється зв'язок по одноіменних полях *Tab\_nom*.

Після того як зв'язок був встановлений, можна перейти до проектування звіту. На форму додається компонент *QuickRep.* Властивості *DataSet* даного компонента присвоюється значення *Table1*. Потім у звіті розміщується смуга заголовка, тобто компонент *QRBand1*, властивість *BandType* якого рівна *rbTitle*. На цю смугу поміщається компонент *QRLabel, у* властивість *Caption* якого записується рядок, що позначає заголовок даного звіту. Потім додається ще одна смуга *QRBand2* і її властивості *BandType* присвоюється значення *rbDetail*. На цій смузі містяться чотири елементи *QRDBText,* що зв'язуються з полями *Tab\_nom*, *Name*, *Dolgn* і *Otdel* таблиці firm.db.

На наступному кроці в звіті розміщується компонент *QRSubDetail.* Даний компонент зв'язується з підлеглою таблицею: його властивості *DataSet* присвоюється значення *Таbl2*. На цій смузі містяться три компоненти *QRDBText,* що зв'язуються з полями *Mes*, *Dni* і *Zarpl* таблиці zarplata.db. Нарешті, у поле компонента *QRBand2* розміщаються заголовки стовпців, що відповідають відображуваним полям підлеглої таблиці

У результаті з підлеглої таблиці виводиться помісячна інформація про виплату зарплати для кожного співробітника, дані про яке утримуються в головній таблиці.

*Композитний звіт.* Композитним (складним, складеним) називається звіт, що поєднує в собі кілька простих звітів, що випливають при друці один за іншим.

Для створення композитного звіту використовується компонент *QRCompositeReport .* Об'єднання звітів здійснюється в оброблювачі події *OnAddReport* шляхом додавання назв відповідних звітів у список властивості *Report* даного компонента. Це робиться приблизно в такий спосіб:



**Procedure TForm3.QRCompositeReportlAddReports (Sender: TObj ect);**

**begin**

**QRCompositeReport1.Reports.Add**

**(Forml.QuickRep1);**

**QRCompositeReport1.Reports.Add**

**(Form2.QuickRep1);**

**end;**

У приведеному прикладі композитний звіт складається з двох звітів: *QuickRep1*, що був раніше визначений у формі *Form1*, і *QuickRep1*, визначений у формі *Form2*. Назви звітів збігаються, тому що на дві різні форми був поміщений новий компонент звіту, що за замовчуванням одержує ім'я *QuickRep1*. Природно, розроблювач може за бажанням змінити ім'я звіту. Друк і попередній перегляд складеного звіту виробляються так само, як і для простих звітів, наприклад, за допомогою методу *Preview*:

**1.3. Інше програмне забезпечення для побудови звітів.**

FastReport

Необхідність написання власного компонента для побудови звітів виникла в 1997 році, під час розробки системи обліку заробітної плати. Специфікою такої системи є велика кількість форм звітності, необхідність легкого настроювання їхнього зовнішнього вигляду. Існуючі доступні системи побудови звітів не мали потрібні характеристики, тому було прийняте рішення про розробку власного генератора звітів. Концепція була запозичена з 1С-Бухгалтерії 6.0 для Windows - основним елементом звіту є довільним образом обрамлений прямокутник із багаторядковим текстом усередині; у тексті можуть утримуватися змінні - рядка в квадратних дужках. Ця концепція виявилася досить життєздатною - незважаючи на те, що в найпершій версії генератора застосовувався тільки один бэнд - рядок даних, це дозволяло будувати багаторівневі звіти.

Пізніше, у 1998 році, генератор звітів оформляється у виді компонента (спочатку це був просто набір модулів, що підключаються,). З цього моменту він одержує назву "FastReport" і починає стрімко нарощувати функціональність. На сьогоднішній день FastReport - це цілком візуальний генератор звітів, тобто більшість звітів можна побудувати, користуючись тільки мишею.

Crystal Report

Популярний пакет генератора звітів Crystal Report призначений для створення звітів презентабельного виду різного типу: табличні запити до баз даних, створення діаграм, OLAP дані.

**Висновок**

Курсова робота була розроблена на основі об’єктно-орієнтованого середовища програмування Delphi. В ході виконання курсової роботи були розглянуті основні компоненти які призначені для виводу інформації на принтер в середовищі програмування. Було створено приклади програм які використовували різні компоненти закладки QReport. Зокрема було створено додаток, який дозволяє працювати із графічним звітом, звітом створеним на основі баз даних, та звітом, який друкує текстову інформацію з поля Memo.

**Список використаної літератури:**

**Друковані джерела:**

1. Delphi 7. Для профессионалов, Кенту М., Издательство Питер. 2004, Книга, 1101 с.
2. Программирование в среде Delphi 7. Глушаков С. В., Клевцов А. Л. Изд. 2-е дополненное м переработанное. –Харьков, 2003, 528с.
3. Программирование в Delphi 6. Архангельский А. Я.

**Інтернет ресурси:**

1. www.delphimastak.ru
2. www.delphimaster.ru
3. www.delphi.narod.ru