МИНИСТЕРСТВО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ УНИВЕРСИТЕТ

VISUAL FOXPRO – СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

Методические указания к лабораторному практикуму

Самара 2004

Составитель: Логанова Л.В.

УДК 004.451.5 (075)

VISUAL FOXPRO – система управления базами данных: Методические указания к лабораторному практикуму/Самар.гос.аэрокосм.ун-т; Сост. Л.В.Логанова. Самара, 2004. 42 с.

Содержат краткие теоретические сведения и задания для лабораторных работ с СУБД Visual FoxPro 7.0 по курсу «Базы данных и экспертные системы».

Методические указания предназначены для студентов, обучающихся по направлению 511600 «Прикладная математика и физика».

Подготовлены на кафедре технической кибернетики.

Печатаются по решению редакционно-издательского совета Самарского государственного аэрокосмического университета

Рецензент канд.техн.наук, доц. Чигарина Е.И.

ЛАБОРАТОРНАЯ РАБОТА №1

Тема: Создание свободных таблиц и средства их ведения

Создание таблицы заключается в определении структуры таблицы и ее заполнении. Каждая таблица хранится в памяти на диске в виде файла с расширением .dbf. Имена таблицы и файла с расширением .dbf совпадают, поэтому количество символов в имени свободной таблицы не может превышать 10 символов. Если же таблица включена в базу данных , то можно использовать имя таблицы, содержащее до 128 символов.

Структура таблицы включает перечень имен полей таблицы (Name) с указанием типов (Type) и размерностей полей (Width – общая длина поля, Decimal – длина дробной части), указание ключевых полей (Index).

Имя поля должно начинаться с буквы или знака подчеркивания, включать буквы, цифры и знак подчеркивания. В предыдущих версиях FoxPro длина имени поля не превышала 10 символов. Длина имени поля таблицы, входящей в базу данных, может находиться в диапазоне от1 до 254 символов, причем использование пробелов в имени недопустимо.

Основные типы полей, используемые в Visual FoxPro 7.0, приведены в табл. №1.

Независимо от того, каким образом вводятся данные в таблицу (в режиме просмотра или при выполнении программы), достоверность ввода в каждое поле контролируется в соответствии с определенными условиями.

Вкладка Fields (Поля) окна Table designer (Конструктора таблиц) имеет три области:

Display (Отобразить), Field validation (Проверка достоверности полей), Map field type to classes (Отобразить тип поля в классы). Первая содержит следующие поля ввода: Format (Формат), которое позволяет установить формат отображения поля таблицы, Input mask (Маска ввода), позволяет определить маску при вводе данных в поле таблицы, Caption (Надпись), позволяет

Таблица 1

Тип поля	Содержимое
Character (Символьный)	Любые символы. Максимальный размер символьного поля – 254 символа
Numeric (Числовой)	Включает цифры от 0 до 9, знак (необязательно), дробную часть (необязательно)
Float (Вещественный)	Числа, имеющие дробную часть (представляются в формате с плавающей точкой), определенные с обычной точностью
Double (Двойной вещественный с двойной точностью)	Числа, имеющие дробную часть (представляются в формате с плавающей точкой), определенные с двойной точностью
Integer (Числовой, целочисленный)	Целые числа
Data (Дата)	Дата, определяющая день, месяц и год. Ввод в это поле контролируется Visual FoxPro автоматически
DataTime (Дата и время)	Дата и время. По умолчанию формат вводимого значения имеет вид мм/дд/гг чч:мм:СС
Logical (Логический)	Логические данные. Может принимать только .T. – истина, .F. – ложь
Memo (Текстовое поле произвольной длины)	Примечания, размер которых неограничен. Может содержать любую символьную информацию, размер которой ограничивается лишь объемом доступного дискового пространства. Фактически – это указатель на блок данных в файле с расширением fpt (файлы dbf и fpt одноименны). Для заполнения данного

Тип поля	Содержимое
	поля нужно дважды щелкнуть мышью по данному полю (или CTRL+Home), после чего в открывшемся окне набрать нужный текст примечаний. Закрыв окно редактирования (CTRL+W), значение поля будет сохранено, а слово memo в поле базы данных изменится на Memo.
General (Общее, двоичное поле произвольной длины)	Поле, используется для хранения ссылки на объект. Под объектом понимается любой текстовый файл, звуковой файл, диаграмма, иллюстрация или другой файл, созданный с помощью приложения, поддерживающего технологию OLE. Для заполнения данного поля нужно двойным щелком мыши на нем открыть окно редактирования. После чего выбрать пункт меню Edit(правка)-insert object(вставить объект) и в открытом выбранном приложении создать нужный объект (например, рисунок). Закрыв окно приложения и окно редактирования поля, поле gen будет заменено на Gen, значение будет сохранено
Currency (Денежный)	Поле для представления денежных сумм, используется для хранения чисел с четырьмя знаками после запятой
Memo binary	Двоичный текст произвольной длины. Содержит любые данные поля memo, которые можно использовать, не изменяя кодовую страницу
Character binary	Двоичный текст длиной до 254 символов. Содержит любые символьные данные, которые используются, без изменения кодовой страницы

определить другой заголовок для поля; вторая содержит следующие поля редактирования: Rule (Правило), в которое вводится логическое выражение, используемое при проверке значимости поля, Message (Сообщение), которое содержит сообщение о неправильном вводе значений, Default value (Значение по умолчанию), содержащее значение поля, которое будет вводиться во все новые записи по умолчанию; последняя содержит поля редактирования с определением библиотеки и ее класса, базового для объекта, представляющего поля таблицы определенного типа в экранной форме.

Вкладка Table(Таблица) позволяет определить свойства таблицы, осуществляя проверку достоверности ввода данных на уровне записей, выдачу сообщения в случае неправильного ввода в области Record Validation.

В области Triggers возможна установка трех триггеров – выражений логического типа, результат вычислений которых определяет реакцию системы на добавление, модификацию или удаление записи таблицы. Если значение результата вычисления выражения равно .F., соответствующие действия отменяются и генерируется сообщение об ошибке: «Trigger failed».

При работе с таблицей для перехода к нужной записи в Visual FoxPro существует множество способов. Чтобы перейти к некоторой записи таблицы, открытой в режиме просмотра, следует в меню выбрать Table-Go to Record. Далее выбрать: Top-переход к первой записи, Bottom-к последней записи, Next-к следующей от текущей, Previous-к предыдущей, Record #-к записи с указанным номером.

Locate – последовательный поиск первой записи, удовлетворяющей заданному условию. В диалоговом окне следует указать область поиска, выбрав следующие варианты:All-по всем записям, Next+число-поиск от текущей записи указанное число записей, Record+число-поиск от начала таблицы указанное число записей, Rest-поиск от текущей записи до конца таблицы; условие For (поиск записей, для которых выполняется указанное условие) или условие While (поиск записей до тех пор, пока выполняется введенное условие), определяемые с помощью конструктора выражений. Для продолжения поиска записей, удовлетворяющих условию, в командном окне следует ввести Continue.

Для индексирования таблицы по одному полю необходимо на вкладке Fields выбрать поле для индексирования, затем щелкнуть кнопкой мыши в колонке Index и выбрать порядок индексирования.

Задание на лабораторную работу

1. Создайте свой каталог, в котором вы будете хранить файлы базы данных, набрав команду mkdir с указанием каталога stud, номера группы, своего имени (например, mkdir c:\stud\626\andrei). Для задания постоянного каталога в процессе сеанса работы с FoxPro выполните команду: Set Default to <имя каталога>.

2. Создайте таблицу, воспользовавшись конструктором таблиц. Для чего в меню **Файл (File)** выберите **Создать таблицу(New table)** с именем Stud.dbf, в которой хранятся сведения о студентах. Таблица содержит следующие поля:
- номер зачетной книжки Name – Num, Type – Numeric, Width – 6 ,
- фамилия, имя, отчество Name – Fio, Type – Character, Width – 25,
- год поступления в университет Name – God, Type – Integer,
- дата рождения Name – Dr, Type – Data,
- вид обучения (платное или бесплатное) Name – Plata, Type – Logical ,
- средний балл при поступлении Name – Sr, Type – Float, Width – 4, Decimal-2,
- стоимость обучения в платной группе Name – Money, Type – Currency,
- портрет Name – Port, Type – General,
- адрес Name – Adr, Type – Memo.

3. В таблицу, хранящую сведения о студентах, добавить ограничения на уровне поля: номер зачетной книжки должен быть больше 1000, по умолчанию значение года рождения студентов – 1986.

4. Задать ограничение на уровне записи для таблицы, хранящей результаты сессии – сумма оценок по трем предметам не должна превышать 15. Запретить изменения в таблице сведений о студентах с помощью триггера.

5. Заполнить 4-5 записями созданную таблицу.

6. Добавить, отметить на удаление и удалить записи таблицы, используя соответствующие пункты меню Table для открытой таблицы, находящейся в режиме просмотра.

7. Осуществить последовательный поиск записей с информацией о студентах, дата рождения которых равна заданной – { / / }.

8. Изменить структуру таблицы, установив индексные выражения для созданной таблицы, изменив последовательность значений некоторых полей.

ЛАБОРАТОРНАЯ РАБОТА №2

Тема: Создание и ведение базы данных реляционного типа

Реляционная база данных – это совокупность взаимосвязанных таблиц, описывающих некоторую предметную область. Под предметной областью понимается часть реального мира, определяющая информационные потребности системы. Структура таблиц определяет эффективность программ, обрабатывающих эти таблицы и все приложение в целом. Реляционная модель баз данных основывается на математических принципах теории реляционных наборов. Для простоты манипулирования данными при создании таких наборов рекомендуется нормализовать данные.

Один файл базы данных с несколькими индексными или мультииндексными файлами при открытии помещается в область памяти, именуемую рабочей областью. Различают простой и мультииндексный файлы. Простой индексный файл имеет расширение .IDX и содержит по одному значению ключа для каждой записи. Мультииндексный файл имеет расширение .CDX и одновременно управляет несколькими индексами для каждой записи. Отдельные ключевые поля в таких файлах называются тегами. Мультииндексные файлы бывают структурные и неструктурные. Структурный имеет то же имя, что и таблица базы данных, и открывается вместе с соответствующей таблицей. Неструктурный файл нужно открывать специально, его имя отлично от имени таблицы, для которой проводится упорядочение записей.

При создании базы данных в Visual FoxPro поддерживаются связи один к одному и один ко многим. Связи типа многие ко многим преобразуются к связям многие к одному и один ко многим на более ранних этапах проектирования баз данных.

Для создания базы данных средствами Visual FoxPro необходимо выбрать в меню File(Файл) – New(создать) – Database-New(База данных-новый файл). Созданный поименованный файл будет иметь расширение .DBC. Далее в меню следует выбрать Database(База данных) – Add или New для добавления или создания таблицы в открытой базе данных. Для установления связи между таблицами необходимо предварительно проиндексировать каждую из них, причем поле, по которому осуществляется связь, должно быть одного типа и размерности в обеих таблицах.

В Visual FoxPro поддерживаются следующие типы индексов:

Первичный (Primary) применяется для связывания таблиц и определения условий целостности данных в базе данных. Этот индекс может быть только один и не может содержать повторяющихся значений.

Кандидат (Candidate) не допускает повторяющихся значений, но любая таблица может иметь более одного индекса этого типа.

Обычный (Regular) обеспечивает хранение значений индексного выражения для всех записей таблицы. Если несколько записей имеют одинаковые значения, то каждое из них хранится отдельно.

При указании индексных полей в процессе создания структуры таблицы кроме выбора типа индекса можно выбирать и тип упорядочения (по возрастанию или убыванию), определить вид индексного выражения.

При установлении связи одна из них является родительской, а другая дочерней. Для установления связи 1:1 типы индексов в каждой из связываемых таблиц должны быть типа «Cadidat» или «Primary» («Кандидат» или «Первичный» соответственно). В случае связи 1:n поле связи родительской таблицы является первичным ключом и тип индекса устанавливается «Cadidat» или «Primary». В дочерней таблице поле, по которому осуществляется связь, является внешним ключом и имеет тип Recular или Unique (Обычный или уникальный соответственно). Чтобы связать таблицы по индексному полю, мышь подводится к индексному полю родительской таблицы, удерживая нажатой левую клавишу переносится на соответствующее индексное поле дочерней таблицы, по которому устанавливается связь. Аналогично устанавливаются необходимые связи между всеми таблицами, входящими в базу данных. Отношения, созданные в Конструкторе баз данных, сохраняются и после окончания работы с системой и могут быть использованы для создания представлений и запросов.

Для установления связей между таблицами на время сеанса работы с FoxPro необходимо также проиндексировать связываемые таблицы по общим полям, затем следует выбрать пункты меню Window (Окно) - Data session (Сеанс данных). В открывшемся окне для установления связи 1:1 подвести курсор к родительской таблице, после чего щелкнуть мышью на кнопке Relations (Связь). Затем подвести курсор к дочерней таблице, после выбора которой в правой части окна отразится установленная связь. Для установления связи 1:n после установления связи 1:1 нажать кнопку «1 to many». В правой части окна будет показана связь между соответствующими таблицами. При просмотре связанных таблиц переход по записям осуществляется синхронно.

При удалении, замене или вводе данных в таблицы, между которыми установлены отношения, следует определить правила для обеспечения целостности данных на уровне базы данных, тогда все приложения будут выполняться правильно. Для определения правил для обеспечения целостности данных

базу данных следует упаковать, выполнив команду меню Database-Clean Up, а затем команду Database-Referential Integrity. Использование вкладок Rules for Updating, Rules for Deleting, Rules for Inserting открывшегося окна Referential Integrity Builder позволяет определить правила обеспечения целостности данных в связанных таблицах при замене, удалении и вставке данных. Для каждого из этих действий с помощью переключателей могут быть установлены следующие правила:

На вкладке Rules for Updating при изменении ключевого значения в родительской таблице

- Cascade – каскадно изменяются все связанные записи дочерней таблицы в соответствии с новым значением ключа;
- Restrict – запрещается внесение изменений, если в дочерней таблице имеются связанные записи;
- Ignore – разрешаются любые изменения без изменения записей дочерней таблицы.

На вкладке Rules for Deleting при удалении записи в родительской таблице

- Cascade – удаляются все связанные записи в дочерней таблице;
- Restrict – запрещается удаление записей, если в дочерней таблице имеются связанные записи;
- Ignore – разрешается удаление записей без изменения записей дочерней таблицы.

Rules for Inserting при добавлении новой или модификации существующей записи дочерней таблицы

- Restrict – запрещается добавление записей, если в родительской таблице нет соответствующего ключевого значения;
- Ignore – разрешается вставка записей.

Щелчок на кнопке ОК приведет к созданию хранимых процедур, используемых в качестве «триггеров».

Транкзации – это еще одно средство управления обновлением данных для сохранения целостности данных. Транкзацией называют последовательность операций, которые все изменения записывают в оперативную память или на диск. Окончательное обновление файлов с расширением .dbf, .fpt, .cdx, входящих в одну базу данных, происходит только при завершении этих операций. Команда BEGIN TRANSACTION инициирует транкзацию, которая продолжается до выполнения первой команды ROLLBACK или END TRANSACTION. Первая из них используется для выполнения возврата, т.е. аннулирования всех выполненных действий. Вторая завершает текущую транкзацию и сохраняет все выполненные изменения. Тран-

кзации могут быть вложенными (до 5 уровней). Для них выполняются следующие правила:

1) изменения, сделанные во всех вложенных транкзациях, фиксируются только последней командой END TRANSACTION;

2) команда END TRANSACTION завершает только свой уровень транкзации;

3) команда ROLLBACK отменяет изменения только своего уровня транкзации;

4) изменения, выполненные для одних и тех же данных, на более высоком уровне транкзации имеют приоритет над изменениями, выполненными на более низком уровне.

Задание на лабораторную работу

1. Создать базу данных Base.dbc, хранящую информацию о результатах сессий студентов одной группы. Для этого добавить в базу данных созданную ранее таблицу Stud.dbf, проиндексировать по полю Num с типом индекса «Кандидат».

2. Создать новую таблицу Session.dbf, хранящую результаты сессий, включающую поля: номер зачетной книжки, номер сессии и оценки по трем предметам.

3. Проиндексировать эту таблицу по номеру зачетной книжки, определив тип индекса как регулярный.

4. Связать две таблицы, установив связь 1:n.

5. Установить временную связь 1:n между таблицами Stud.dbf и Session.dbf. Просмотреть, как установленная связь отображается при движении по записям таблицы.

6. Учесть ограничения ссылочной целостности с помощью конструктора.

ЛАБОРАТОРНАЯ РАБОТА №3

Тема: Создание запросов к базе данных, использование результата

SQL (Structured Query Language – язык структурированных запросов) используется для анализа таблиц и баз данных через запросы (Query). Особенность языка состоит в том, что его команды ориентированы на работу не с отдельными записями, а с наборами записей как свободных, так и реляционно-связанных таблиц и баз данных. В командах SQL формулируется критерий и способы преобразования информации. При этом SQL сам открывает нужные таблицы и оставляет их открытыми, открывает и создает нужные индексные файлы и определяет наиболее эффективные последовательности операций для получения результата. С помощью SQL можно создавать и модифицировать таблицы, добавлять, изменять и удалять записи в таблицах, выполнять операции запросов и др.

Для создания запросов в Visual FoxPro имеются удобные инструментальные средства. Для открытия окна конструктора запросов необходимо выбрать следующие пункты меню:

File-New-Query-New-New file.

Окно конструктора состоит из области представления таблиц, отношений между ними и области для записи условий и выводимых в запросе полей.

Последняя, в свою очередь, включает 6 вкладок:

Fields, которая позволяет определить, какие поля таблиц будут включены в результат выполнения запроса, Join – определяет условия соединения таблиц, Filter – определяет условия выборки , Order by – позволяет упорядочить выборку по одному или нескольким полям/выражениям, включенным в результат запроса , Group by – выполняет группировку записей на основе значений одного или более полей, предлагает набор агрегатных функций – SUM(), AVG() и т.д. На этой вкладке расположена кнопка, добавляющая в выражение SQL предложение HAVING для определения условия отбора, которому должна удовлетворять группа, чтобы попасть в результат запроса, Miscellaneous – смешанные параметры.

Для добавления таблицы в область представления таблиц достаточно щелкнуть правой клавишей мыши в любом месте окна конструктора и выбрать в открывшемся меню Add table. В запрос могут входить как свободные таблицы, так и таблицы в составе базы данных. При открытой базе данных можно включить в запрос и представление (View).

Для построения сложных выражений, используемых как условие соеди-

нения таблиц или выборки на вкладках Join, Filter в ниспадающем меню, расположенным под надписью Logical, выбрать OR или AND.

В выражениях условий выборки могут быть использованы следующие предикаты: предикаты сравнения {=, <>,>,<,>=,<=}; предикат Between (namefields, <значение1>, <значение2>)-принимает значения между <значение1> и <значение2>, предикат вхождения в множество IN (множество); предикат сравнения с образцом LIKE и not LIKE, предикат сравнения с неопределенным значением IS null и IS NOT NULL.

При работе с конструктором запросов в основном меню появляется пункт Query, содержащий пункты, которые также позволяют выполнять действия по созданию запроса и определять направление вывода результата запроса (Query destination).

При выборе Browse запрос выводится на экран в виде таблицы, при закрытии она удаляется из памяти, при выборе Cursor результат выводится во временный файл, который можно просматривать и двигаться по записям в любом направлении, Table результат выводится в файл с расширением .DBF, GRAF результат будет представлен в виде графика, Screen результат отобразится в главном окне и может быть послан на принтер или в файл, Report и Label позволят отобразить результат в новой или существующей форме отчета или этикетки.

Задание на лабораторную работу

Создать следующие запросы к базе данных «Результаты сессии студентов», использовать различные направления вывода результатов запроса:

1. Вывести сведения о студентах, год поступления которых в институт равен заданному.
2. Вывести фамилии студентов и размер оплаты за обучение для студентов, обучающихся платно.
3. Вывести сведения о студентах, обучающихся платно и средний балл которых при поступлении меньше заданного.
4. Вывести фамилии студентов в алфавитном порядке с указанием даты рождения.
5. Вывести год и количество студентов, поступивших в заданном году.
6. Вывести фамилии студентов, сдавших сессию, указанного номера с оценками не ниже 4 по всем предметам.

ЛАБОРАТОРНАЯ РАБОТА №4

Тема: Создание и использование представлений SQL

Для избирательного доступа к составляющим базы данных используется достаточно мощное средство – представление SQL, которое объединяет гибкость запроса с возможностью модификации данных в представлении. Представления можно определить как обновляемые курсоры SQL. Локальное (Local) представление базируется на таблицах Visual FoxPro, а удаленное (remote) – на использовании ODBC.

Синтаксис команды создания представления SQL и добавления его к текущей базе данных имеет следующий вид:

CREATE SQL VIEW [<имя представления>] [REMOTE]
[CONNECTION<имя соединения> [SHARE]
/CONNECTION<имя источника данных>]
[AS <команда SELECT>]

REMOTE – определяет использование удаленных данных для построения представления SQL.

CONNECTION <имя соединения> - определяет имя соединения при открытом представлении для работы с удаленными данными. Без опции [SHARE] соединение будет уникальным.

CONNECTION<имя источника данных> - определяет существующий источник данных в ином, чем DBF, формате, с которым устанавливается соединение. Обращение к таким данным выполняется через интерфейс ODBC. Описание параметров соединения хранится в базе данных.

Использование в этой команде операции SELECT (SQL) позволяет выделить в представлении определенные поля и записи из одной или нескольких включенных в базу данных таблиц и модифицировать результат выборки, в отличие от запроса, который не допускает редактирования результата.

В Visual FoxPro для создания представлений можно воспользоваться конструктором представлений (View Designer). В отличие от конструктора запросов он имеет еще одну вкладку – Update Criteria. Вы можете решить, какие таблицы будут модифицироваться и как именно: Update, Delete, Insert. На вкладке Update criteria справа от списка Field Name расположены две группы переключателей, где указываются ключевые поля, а также поля, которые могут модифицироваться.

С использованием представлений можно создавать параметрические запросы. В этом случае запрос будет выполняться с разными исходными данны-

ми. Для этого при создании представления вызывается диалоговое окно View Parameters, выбором следующих пунктов меню Query- View Parameters, в котором указываются имена и типы параметров, используемых в части Select-Where. До показа информации из вида параметрам присваиваются конкретные значения, например из командного окна или же формы, затем запускается представление.

Данные, хранящиеся вне среды Visual FoxPro в форматах других программных продуктов (Microsoft SQL Server, Oracle и др.), или удаленная база данных Visual FoxPro называются удаленными данными, которые, хотя и хранятся вне используемой среды, также могут обрабатываться. При этом в качестве инструмента используют Remote View (Удаленное представление). Для доступа к удаленным данным используют такие средства, как ODBC (Open database Connectivity – открытые средства связи с базами данных) и Connections(Соединения).

Создание удаленного представления начинается с выполнения команды меню Database – New Remote View. Далее возможны варианты работы с мастером View Wizard или без него New View. При выборе второго варианта открывается диалоговое окно Select Connection or Data Source для выбора одного из методов доступа: использование источника данных (Available Data sources) или использование соединений (Connection). При использовании источника данных открывается список доступных источников, созданных с помощью средства ODBC Administrator. В результате выбора одного из них в списке Available Data sources запустится конструктор представлений View Designer, работа с ним рассмотрена выше. Несколько удаленных представлений можно объединить при создании локального представления, установив в диалоговом окне Add Table or View переключатель Select в положение View.

Во втором случае удаленный источник данных определяется заранее с помощью средства ODBC Administrator и хранится в базе данных Visual FoxPro. После выполнения команды Database- Connection открывается одноименное диалоговое окно с кнопкой NEW для создания нового соединения с использованием конструктора Connection Designer. Теперь созданное соединение можно выбрать в диалоговом окне Select Connection or Data Source и построить удаленное представление, при использовании которого Visual FoxPro через ODBC соединится с ODBC-драйвером сервера базы данных и выполнит команду Select, определенную представлением.

Задание на лабораторную работу

1. Создать просмотры для базы данных «Результаты сессии студентов».

1.1. Вывести сведения о студентах, сдавших указанную сессию на «отлично».

1.2. Вывести фамилии студентов, поступивших в институт в указанном году и обучающихся в платной группе.

1.3. Открыть созданные представления и модифицировать разрешенные для модификации данные.

2. Создать параметрические запросы.

2.1. Вывести фамилии студентов, поступивших в институт в указанном через параметр году.

2.2. Вывести фамилии студентов, сдавших на отлично указанный через параметр номер сессии.

3. Создать удаленное представление.

ЛАБОРАТОРНАЯ РАБОТА №5

Тема: Организация меню

Цель создания меню – обеспечить пользователю простой доступ ко всем компонентам приложения. Обычно содержит собственное меню, которое заменяет основное меню Visual FoxPro, и команды для выполнения конкретных задач. С помощью меню организуется одновременная работа с такими заранее созданными объектами, как базы данных, таблицы, представления, запросы, формы, отчеты, программы и т.д.

Меню любой сложности строится из меню двух типов: горизонтального и вертикального (всплывающего). Горизонтальное меню, примером которого является основное меню Visual FoxPro, состоит из нескольких горизонтально расположенных пунктов, которые называются Pad-пунктами (например, File, Edit и т.д.). Всплывающее меню – Popup – состоит из нескольких вертикально расположенных пунктов, которые называются BAR и появляются только при активизации соответствующего Pad-пункта. Popup – меню может использоваться как в составе меню более высокого уровня, так и самостоятельно.

Для создания меню средствами Visual FoxPro мы воспользуемся средствами визуального проектирования. В результате выполнения команды File-New-Menu или команды Create menu <имя меню> откроется диалоговое окно для выбора вариантов меню: Menu – линейное Bar-меню или Shortcut – всплывающее Popup-меню. При выборе Menu откроется окно конструктора меню, а в основное меню добавится пункт Menu.

При выборе Menu-Quick Menu создается меню, которое содержит все компоненты основного меню Visual FoxPro. Его можно модифицировать или использовать некоторые пункты в собственном меню.

Для создания собственного меню нужно выйти из Quick Menu и повторить команды по созданию меню, описанные выше. Конструирование пользовательского меню начинается с пунктов главного меню.

Окно конструктора меню включает следующие разделы:

Prompt – указывается имя пункта меню. По умолчанию первая буква в имени пункта определяет клавишу быстрого доступа. Для изменения клавиш быстрого доступа используйте символы «\<», расположенные слева от буквы текста команды меню, которая выбрана как клавиша быстрого доступа. Для вывода вместо имени пункта меню разделительной черты в имени пункта меню необходимо набрать два символа '\-'.

Result – результирующее действие при выборе пункта меню. Возможно 4 вида результирующих действий: Command-для выполнения единственной команды, записанной справа, Pad name или Bar# – для ввода имени команды системного меню Visual FoxPro, Submenu – для создания подменю, для определения пунктов которого следует нажать Create, Procedure – для соответствующего пункта меню выполняется процедура, для ввода которой необходимо нажать клавишу Edit.

Options – опции меню, с помощью которых можно создавать горячие клавиши, для чего в окне Prompt Options использовать поля ввода Key label и Key text, осуществлять динамическое управление меню, записав в поле ввода Skip For окна Prompt Options логическое выражение, в зависимости от значения которого меню оказывается заблокированным и недоступным для пользователя. В том же окне в поле ввода Message записывается сообщение, которое будет выведено в статусной строке при выборе данного пункта меню. Удаляются или вставляются выбранные пункты меню щелчком на кнопке Delete или Insert в области Item окна Menu Designer. Чтобы изменить порядок пунктов меню, достаточно выбрать пункт и перетянуть его вверх или вниз. Для предварительного просмотра созданного меню щелкните на кнопке Preview.

После сохранения меню создаются файлы описания меню с расширением .mnx, .mnt. В любой момент меню можно модифицировать командой MODIFY MENU <имя файла с расширением .mnx >или в меню выбрать пункты File-Open. Обычно работа с меню управляется главной программой приложения, содержащей команду DO <имя программного файла меню>. Для генерирования программного файла меню следует выполнить команду

Menu-Generate и ввести имя этого файла, который по умолчанию получает расширение .mpr. Для возврата в системное меню следует выполнить команду Set Sysmenu to Default.

Задание на лабораторную работу

1. Создать меню, содержащее помимо пунктов основного меню Visual FoxPro, добавив в него пункт Utilities, включающий два подпункта: Calendar (имя пункта системного меню, предназначенного для работы с календарем – *MST_Diary),* Calculator (_MST_Calcu).

2. Создать собственное меню:

File	
Open DataBase	Open DataBase <имя базы данных>
Open Table	Use <имя таблицы>
Close	Close all
Edit	
Append	Append Blank
Delete	Delete
Recall	Recall
Browse	Browse last
Query	Do <имя файла.qpr>
Exit	Set Sysmenu to default

В случае, если открытых таблиц нет, пункт меню Edit должен быть заблокирован и недоступен (использовать функцию Used, возвращающую .T., если таблица открыта).

Созданное меню сохранить, сгенерировать и запустить.

ЛАБОРАТОРНАЯ РАБОТА №6

Тема: Создание и модификация экранной формы

В Visual FoxPro формат отображения в виде формы называется экранной формой. Форма – это визуальный объект контейнерного типа, который может содержать в себе другие объекты (в том числе и другие объекты контейнерного типа). В Visual FoxPro представлен большой набор элементов управления, а главное, эти элементы обладают достаточным количеством событий и методов для создания полноценного пользовательского интерфейса, отвечающего современным требованиям. Экранные формы сохраняются в файлах с расширением .scx или .sct. Экранная форма –самодостаточный

программный класс и не нуждается в процедуре формирования программного кода.

Экранные формы создаются и модифицируются при помощи визуальных средств проектирования Visual FoxPro. С этой целью используется мастер и/или конструктор форм. Запустить мастер или конструктор форм можно при помощи соответствующих команд системного меню Visual FoxPro.

Можно сначала создать стандартную форму с использованием Wizard Form (Мастер форм), а затем модифицировать ее с помощью Form Designer (Конструктор форм).

Для запуска мастера достаточно выбрать следующие пункты меню File-New-Form-Wizard или Tools-Wizards-Form. Далее следует выбрать, будет ли это форма для одной таблицы (Form-Wizard) или для связанных таблиц (One to Many Form-Wizard). Отвечая на вопросы мастера, завершить конструирование формы.

Для создания формы с использованием Form Designer (конструктора форм) необходимо выполнить следующую команду меню File-New-Form-New file. Далее следует выбрать, будет ли это форма для одной таблицы (Form-Wizard) или для связанных таблиц (One to Many Form-Wizard). После этого откроется окно Form Designer.

При работе с конструктором форм прежде всего следует определить среду окружения, т.е. выбрать таблицы и установить отношения между ними. Окно Data Environment для определения среды окружения открывается командой View-Data Environment или командой Data Environment контекстного меню. В открывшееся окно добавить таблицу/таблицы, выбрав пункты меню Data Environment-Add или команду Add контекстного меню.

Для формы, как и для каждого ее элемента, можно открыть окно Properties, которое имеет пять вкладок: All – все свойства, методы, задаваемые формой, а также события, на которые данная форма способна реагировать; Data – свойства только данных формы; Layout – свойства, определяющие внешний вид формы; Methods – методы для каждого события, на которое реагирует форма; Other – информация о родительском классе, а также пользовательских свойствах и методах.

При конструировании формы используются следующие панели инструментов – Layout Toolbar (для выравнивания объектов в форме, некоторые аналогичные команды содержатся в меню Format), Form Controls Toolbar (для размещения в форме стандартных элементов управления), Color Palette Toolbar(предназначена для изменения цветовой палитры элементов), для вывода которых на экран следует выбрать соответствующие пункты меню View.

При установке курсора мыши на кнопках панелей инструментов высвечиваются подсказки. Панель инструментов Form Controls Toolbar содержит стандартные элементы управления, приведенные в табл. 2.

Таблица 2

Название кнопок	Назначение
Select Objects	Выбор других элементов управления
View Classes	Выбор класса
Label	Создание надписей
Text Box	Создание поля для ввода текста
Edit Box	Создание поля для редактирования Memo-полей
CommandButton	Создание командной кнопки
OptionGroup	Создание одного или нескольких переключателей
CheckBox	Создание флажка
Grid	Создание контейнера сетки
ComboBox	Создание поля с ниспадающим списком
ListBox	Создание списка
Sprinner	Создание счетчика
Line	Создание линии
Shape	Создание прямоугольника
Container	Организация контейнера
Image	Получение изображения
CommandGroup	Создание контейнера с несколькими командными кнопками
Timer	Создание невидимого таймера
PageFrame	Создание контейнера блока страниц
OLEBouldControl	Работа с содержимым поля General
OLEContainerControl	Работа с любым OLE-объектом
Separator	Создание разделителя между элементами
HyperLink	Создание используемого в Internet объекта
BuilderLock	Вызов построителя элемента управления
ButtonLock	Закрепление выбора кнопки панели

Для размещения любого элемента управления в форме предварительно установите режим выбора, щелкнув кнопкой мыши на кнопке Select Object, а затем на выбранном элементе мышью обведите область под объект в любом месте формы. При отпускании кнопки мыши объект отобразится в выделен-

ной области. Чтобы разместить несколько одинаковых объектов на форме, выберите кнопку Button Lock с изображением замка. При этом кнопка выбранного объекта останется зафиксированной до выбора другого объекта.

Задание на лабораторную работу

1. Создать форму с помощью мастера форм для таблицы, хранящей сведения о студентах, используя различные стили форм и кнопок.

2. Создать форму с помощью конструктора форм для вывода сведений из таблицы, содержащей командные кнопки для движения по записям (начало – go top, конец – go bottom, вперед – skip 1, назад – skip -1), удаления – delete, добавления – append blank, закрытия формы – release thisform.

3. Создать форму с помощью конструктора для вывода информации из базы данных, для двух взаимосвязанных таблиц.

ЛАБОРАТОРНАЯ РАБОТА №7

Тема: Создание набора форм FormSet

Для синхронной работы сразу с несколькими формами, как с одним объектом, создают набор форм FormSet. Весь набор обрабатывается одной командой. Например, одновременно на экране можно показать или убрать все формы, входящие в набор, в то время как для множества одиночных форм в данном случае пришлось бы организовывать цикл и ожидать появления каждой формы. Синхронная работа с формами, входящими в FormSet, обеспечивается одним и тем же сеансом данных (Data Session). Для создания группы экранных форм, находясь в конструкторе форм, выбрать следующие команды меню Form-Create Form Set. В результате будет создана группа экранных форм, включающая одну форму Form1. Для добавления в группу очередной формы выполнить меню Form-Add Form. Для осуществления доступа к форме в группе форм при работе приложения необходимо выполнить команду THISFORMSET.FORMS<номер формы>.метод/свойство или THISFORMSET.FORMS<имя формы>.метод/свойство.

Для выполнения лабораторной работы вам придется воспользоваться следующими методами: Hide – скрыть форму, Show – показать форму. Для вывода графических изображений – это методы для изображения линий (line) и прямоугольника (Box), очистить форму – Clear, выгрузить форму из памяти Release. Кроме стандартных свойств и методов формы пользователь может создать свои свойства и методы на уровне формы. Для этого вызывается команда New Property, New Method пункта меню Form.

Задание на лабораторную работу

1. Создать объект – группу экранных форм, состоящую из двух экранных форм. Первая форма содержит поля для ввода координат для вычерчивания линии или прямоугольника и кнопки с надписями **линия, прямоугольник, выход**. Вторая форма содержит кнопку **назад**. При запуске объекта на экране появляется первая форма. При нажатии на кнопки **прямоугольник, линия** форма становится невидимой, на экране появляется вторая форма, где рисуется прямоугольник или линия по точкам, координаты которых введены в первую форму. При нажатии на кнопку **назад** вторая форма исчезает, появляется первая. Кнопка **выход** освобождает память от обеих форм.

2. В форму по работе с базой данных «Результаты сессии» добавить новое свойство – ограничение на количество записей по результатам сессии для одного студента и добавить новый метод, возвращающий .T., если текущее количество сессий для данного студента меньше или равно значению нового свойства и .F. в противном случае. Этот новый метод использовать в объекте Grid, содержащем сведения о сессии. Если при добавлении в объект очередной записи ограничение не выполняется, вывести сообщение о невозможности введения новой записи о результате сессии для данного студента. Для вывода сообщения использовать команду MessageBox(текст сообщения).

ЛАБОРАТОРНАЯ РАБОТА №8

Тема: Обработка событий в Visual FoxPro

Модель события Visual FoxPro позволяет программисту разрабатывать немодальные приложения, что дает возможность, например, автоматически координировать множественные формы и одновременно выполнять множественные экземпляры форм. Окно или форма являются немодальными, если без предварительного их закрытия возможно переключение к другим форме или окну. Система обработки событий Visual FoxPro автоматически вызывает код события в ответ на действия пользователя, нужно только посредством определения класса объекта управления определить, что должно произойти после выбора данного объекта управления, а для последующей обработки не следует писать никакого дополнительного кода. В общем случае событие – это переход объекта из одного состояния в другое. К основным событиям можно отнести загрузку формы (load), удаление формы (Unload), инициализацию объекта (init), щелчок мыши (Click), активизацию (Got Focus), деактивизацию (Lost Focus) и другие. Событие RightClick происходит при нажатии на правую кнопку мыши. Перекрыв это событие, мож-

но создать всплывающее меню. Для этого на уровне формы, перекрыв событие Init или Load, описывается меню с помощью операторов вида:

Define popup <имя меню> in window <имя формы, в которой используется меню>

Define Bar <номер пункта меню> of <имя меню> prompt «название пункта меню»

On selection popup <имя меню> do <имя командного файла, запускаемого при выборе пунктов меню>.PRG

Следующий программный код позволяет описать меню, состоящее из пунктов A, B:

```
Define popup m in window form1
Define Bar 1 of m prompt «A»
Define Bar 2 of m prompt «B»
On selection popup m do g.PRG
```

Перекрыв событие нажатия правой кнопки мыши, например на уровне формы, вызывается созданное меню: Move popup <имя меню> TO <номер строки, номер столбца, где будет отображено меню>

ACTIVATE POPUP <имя меню>

Например, move popup m to 6,35

Activate popup m

Также необходимо создание командного файла, код которого позволит описать действия при выборе того или иного пункта меню.

Например, g.prg:

```
Procedure g
If bar()=1
 Messagebox(«текст1»)
Else
 Messagebox(«текст2»)
Endif
endproc
```

Событие Valid происходит на уровне ввода значения в поле базы данных. Это событие возвращает .T., если введенное значение удовлетворяет условию, в противном случае значение не вводится в базу данных. Например, код события может включать следующие команды:

```
 If this.value><значение типа поля ввода>
    Return .T.
 Else
    Retun.F.
```

Endif

Событие GotFocus возникает при активизации объекта, а LostFocus при деактивизации объекта. Перекрыв эти события на уровне поля, можно при попадании фокуса на поле выделить его цветом, набрав последовательность команд:

THIS.FORECOLOR=RGB(0,0,0)

THIS.BACKCOLOR= RGB(255,0,0)

При потере фокуса:

THIS.FORECOLOR=RGB(0,0,0)

THIS.BACKCOLOR= RGB(255,255,255)

Обработка события Message приводит к выводу сообщения в строке статуса.

Обработка события Programmatic Change происходит при изменении у объекта свойства Value программным путем, то есть когда в программе есть код Object.value=значение.

Задание на лабораторную работу

1. Создать всплывающее меню при работе на уровне поля, содержащее пункты: ввод значения по умолчанию и очищение поля.

2. Выполнить проверку значения вводимой даты поступления в институт до ввода в базу данных.

3. Вывести сообщение в строку статуса, поясняющее содержимое поля ввода при попадании фокуса на это поле.

4. Выделить поле особым цветом при попадании фокуса в это поле и вернуть прежний цвет при потере фокуса.

ЛАБОРАТОРНАЯ РАБОТА №9

Тема: Создание отчетов и этикеток

Отчеты используются для вывода информации, содержащейся в таблицах, на экран, в файл или принтер. Отчет может содержать различные итоговые данные, может быть красочно оформлен. Для создания отчетов в Visual FoxPro можно использовать Report Wizard (Мастер отчета) и Report Designer (Конструктор отчета).

Чтобы воспользоваться Мастером отчета, необходимо выполнить команду Tools-Wizard-report или File-New-report- Wizard. Далее выбрать один из вариантов : One-to-many report wizard (для создания отчета для связанных таблиц) или Report Wizard (для создания отчета по одной таблице). Работа с мастером достаточно проста и поэтому описываться не будет.

Запуск конструктора отчетов происходит в результате выполнения команды File-New-Report-New или введения команды CREATE REPORT <имя отчета> в командном окне.

При работе с отчетом в основное меню добавляется пункт Report для работы с конструктором. Для модификации существующего отчета используется команда Modify Report <имя отчета>. Для быстрого создания стандартного отчета достаточно выполнить команду Report-Quick Report. При этом все поля данных в отчете разместятся автоматически.

Окно конструктора отчета содержит следующие полосы, составляющие отчет:

- Title – заголовок отчета – обычно здесь же выводится текущая дата;
- Page Header – верхний заголовок каждой страницы;
- Group Header – верхний заголовок каждой группы данных;
- Detail – поля каждой записи таблицы;
- Group Footer – нижний заголовок каждой группы;
- Page Footer – нижний заголовок каждой страницы;
- Summary – итоговые данные.

Каждая полоса отчета имеет управляемую высоту, а также содержит элементы: текст, табличные поля, линии, прямоугольники и рисунки. Visual FoxPro позволяет перетаскивать элементы отчета из одной полосы в другую. Для формирования полос Title и Summary выберите команду меню Report-Title/Summary. В открывшемся окне установить флажки Title Band и Summary band. Конструирование отчета, как правило, включает в себя следующие этапы:

- Определение среды окружения;
- Размещение текста и полей;
- Размещение линий, прямоугольников и рисунков;
- Сохранение отчета.

Сохраненный файл имеет расширение .frx, .frm, .frt.

При модификации отчета, созданного мастером, среда окружения уже определена, а чтобы определить ее для нового отчета, выполните команду View-Data Environment.

В раскрывшемся окне выберите мышью в контекстном меню пункт Add. В диалоговом окне Add Table or View выберите базы данных, таблиц, представлений, используемых в отчете. Каждый открытый файл в окне Data Environment будет представлен списком полей. Для работы с элементами отчета активизируйте панель инструментов командой View-Report Controls Toolbar. Назначение кнопок панели инструментов определяется подсказкой,

которая появляется при установке курсора на кнопку. Поля таблицы можно разместить в полосе отчета, обозначенной словом Detail. Для этого достаточно выбрать и перетащить мышью поле открытой таблицы из окна Data Environment. Двойным щелчком мыши на поле в этой полосе открывается окно Report Expression для относительного размещения поля в отчете и условий его печати. Кроме того, можно вызвать построитель для поля, щелкнув кнопкой мыши справа от поля ввода Format. В открывшемся диалоговом окне можно выбрать и установить необходимые параметры для символьных, числовых полей и типа даты.

В конце отчета или в конце каждой группы можно вывести итоговые поля, значения которых являются результатом вычисления выражения. Для чего нужно выбрать кнопку Calculations окна Report Expression, а затем в окне Calculate Field – требуемую опцию.

Для группирования данных применяется команда Report-Data Grouping. С каждой группой данных могут выполняться следующие операции:

- Вычисления внутри каждой группы;
- Печать текста в верхних и нижних заголовках для выделения отдельных групп;
- Переход на новую страницу перед началом печати каждой группы;
- Печать каждой группы с новой страницы.

При формировании выражений, определяющих поля отчета, используются также переменные, изменяя значение которых можно изменить весь отчет, не разрабатывая его заново. Команда Report-Variables открывает одноименное диалоговое окно для определения имени переменной, ввода ее значения или начального значения. В списке Calculate можно выбрать операцию, которая должна быть выполнена над переменной. Для задания причины сброса переменной используется список Reset at. Если после печати отчета переменная больше не понадобится, то следует установить флажок Release after report. Для создания вычисляемых полей нужно щелкнуть по полю отчета правой кнопкой мыши, в окне конструктора выражения выбрать нужную функцию и построить требуемое арифметическое выражение для вычисляемого поля.

Для разметки страницы необходимо выполнить команду меню File-Page Setup и в открывшемся окне определить параметры разметки.

Для улучшения внешнего вида в отчет в качестве эмблемы можно поместить рисунок, а также различные рисунки для каждой записи.

Этикетка – это специальный вид отчета, распечатываемый на клейкой ленте. Чаще всего этикетки используются для печати адресов и сведений об

адресатах, которые наклеиваются впоследствии на конверты. При создании этикеток целесообразно сначала использовать Label Wizard, а затем Label Designer – для добавления рисунка графики и других элементов оформления.

Задание на лабораторную работу

1. С использованием мастеров отчет с группированием для таблицы, хранящей сведения о студентах; отчет для связанных таблиц.

2. С помощью конструктора отчетов создать отчет, включив в него все полосы.

2.1. Выполнить группировку студентов по году поступления и в алфавитном порядке внутри каждой группы.

2.2. Создать вычисляемые поля, подсчитав для каждого студента средний балл сдачи сессии, общее количество студентов в списке и количество студентов в каждой группе.

3. Создать этикетку для вывода сведений о студентах.

ЛАБОРАТОРНАЯ РАБОТА №10

Тема: Проект приложения Visual FoxPro

Проект объединяет базы данных, таблицы, основную программу, которая устанавливает среду и определяет интерфейс пользователя, экранные формы, панели инструментов, меню, запоминает расположение каждого компонента и обеспечивает управление ими. Кроме того, для обобщения информации в проекте рассматриваются запросы и отчеты.

Описание проекта представляет собой специальную таблицу (файл с расширением .pjx), которая содержит информацию о путях ко всем его компонентам.

Для создания проекта необходимо выполнить команду меню File-New-Project-New и в диалоговое окно Create ввести имя создаваемого проекта или выполнить команду Create project <имя проекта>. В открывшемся окне Project Manager (Диспетчер проекта) все его компоненты отображаются на шести вкладках:

Data – базы данных, свободные таблицы и запросы, хранимые процедуры;

Documents – формы, отчеты и этикетки;

Classes – классы и библиотеки классов;

Code – программные файлы (.prg) и библиотеки;

Other – меню, текстовые файлы, файлы клавишных макросов и переменных;

All – все включенные в приложение файлы.

Компоненты проекта располагаются на нескольких уровнях. Если слева от пункта имеется знак плюс, то в него включены пункты следующего уровня. При раскрытии уровня (щелчком мыши) знак плюс изменяется на минус. При создании нового или открытии уже существующего проекта в основное меню добавляется пункт Project, в раскрывающемся меню которого содержатся команды для работы с проектом. Диалоговое окно Project information открывается при выборе команды Project- Project info. Здесь можно ввести информацию о разработчике и каталоге, в котором находится проект, а также о том, нужно ли зашифровывать файл приложения и вводить отладочную информацию в собранное приложение, какой файл с расширением .ico использовать для обозначения минимизированного приложения. Вкладка Files окна Project information содержит список всех файлов, входящих в проект. В верхней части окна расположены кнопки для упорядочения файлов по типу, имени, дате создания. Для добавления компонентов в проект необходимо в окне Project Manager выбрать соответствующую вкладку и щелкнуть мышью на кнопке Add. Первый программный файл, добавленный в проект, является главным по умолчанию. Если необходимо назначить другой программный файл в качестве главного, следует его выбрать и выполнить команду Project-Set main. Такую же установку можно выполнить с помощью контекстного меню. Приведем пример главного программного файла

```
OPEN DATABASE stud
CLEAR
DO stud.mpr
READ EVENTS
CLOSE DATABASE
```

Для создания новых компонентов следует выбрать кнопку New. Кнопка Modify позволяет модифицировать выбранный компонент. Remove – позволяет удалить компонент из проекта или же с диска. В первом случае файлы только маркируются для удаления, а для их физического удаления проект необходимо упаковать, выполнив команду меню Project-Clean Up Project. Кнопка Browse – открывает выбранный компонент для просмотра. Синтаксис команд создания и модификации проекта в Visual FoxPro дополнен опцией NoprojectHook, определяющей, что при открытии проекта не будет создан объект ProjectHook:

Create/Modify project [<>] [in screen] [nowait] [save] [noshow] [NoprojectHook].

По умолчанию немедленно устанавливается связь открываемого проекта с объектом класса projectHook. В этом случае при манипуляциях пользователя с проектом генерируются события класса projectHook. При включении в код метода обработки событий данного класса дополнительных кодов или вызовов других методов возможности управления проектом возрастают.

Приложение представляет собой результат компиляции файла проекта и имеет расширение .app. Файл приложения можно создать как с помощью мастера, так и с помощью диспетчера проекта. В данной лабораторной работе мы рассмотрим второй способ, возможности которого значительно шире.

Для построения приложения щелкните на кнопке Build в окне Project Manager. Откроется диалоговое окно Build Options, в котором будут предложены следующие варианты построения:

Rebuild Project – обновление проекта;

Build Application – построение файла приложения с расширением .app;

Build Executable – создание исполняемого файла с расширением .exe;

Build COM DLL – создание файла динамической библиотеки с расширением .dll;

Ниже имеются четыре флажка:

Recompile All Files – обновление компонентов проекта, измененных после предыдущего построения приложения ;

Display Errors – отображение всех ошибок, обнаруженных при построении; но даже если этот флажок не установлен, результат диагностирования можно увидеть, выполнив команду меню Project-Errors;

Run after Build – устанавливается, если приложение необходимо запустить сразу после создания;

Regenerate Component Ids – регенерация компонентов; этот флажок доступен только при выборе Build Executable или Build COM DLL.

Запуск .app-файла выполняется командой Do <имя файла с расширением .app>.

Запуск .exe-файла приложения выполняется без входа в среду Visual FoxPro. Большой объем файла .exe не позволяет скопировать его на диск с целью переноса приложения на другой компьютер.

Для упаковки приложения и дополнительных файлов, а также для создания файла установки приложения используется мастер Setup Wizard. Предварительно необходимо все файлы, входящие в приложение, скопировать в один каталог. После чего выполнить команду Tools-Wizards-Setup для работы с мастером установки и далее выполнить действия, предписанные им.

Задание на лабораторную работу

1. Создать проект, добавив в него все ранее созданные компоненты: таблицы, запросы, меню, экранные формы, отчеты. В качестве главного выбрать меню, в которое добавить пункт для вывода отчета. Редактирование и просмотр данных выполнять с использованием собственных форм. Для вывода на экран отчета использовать команду: Report form <имя файла отчета>.
2. Создать приложение.
3. Создать файл установки приложения.

ЛАБОРАТОРНАЯ РАБОТА №11

Тема: Программирование в Visual FoxPro

Несмотря на наличие в Visual FoxPro большого количества мастеров, конструкторов и построителей, позволяющих создавать интерфейс пользователя визуально, часто невозможно обойтись без создания и использования дополнительных программ и функций.

Процесс создания программного файла начинается с запуска текстового редактора выполнением команды New-Program меню File. При этом в командном окне генерируется команда Modify command для создания программного файла с расширением .prg, который и представляет собой текстовый файл. Сохраненную в программном файле программу можно запустить на выполнение командой: Do <имя программного файла>. Скомпилированный программный файл имеет расширение .fxp.

Для более подробного ознакомления с языком программирования FoxPro можно обратиться к литературе, указанной в списке использованных источников, или же обратиться к Help.

Подробный формат лишь некоторых команд будет приведен ниже.

List [<диапазон записей>] [fields <список полей>] [for <вржl>] [while <вржl>] off

[to printer [prompt]/ to file <имя файла>] [noconcole] [nooptimize]

В результате выполнения этой команды на экран выводятся все записи, для которых логическое выражение, следующее за For или While, является истинным. OFF – отключает нумерацию записей. А для того , чтобы имена полей не выводились в качестве заголовков, перед выполнением команды List используют команду set heading off. Опции to printer или to file <имя файла> позволяют вывести информацию на принтер или в текстовый файл с указанным именем соответственно. Опция noconcole предотвращает параллельный вывод информации на экран при выводе на принтер или в файл.

Опция nooptimize отменяет Rushmore-технологию при выполнении команды List, что увеличивает скорость ее выполнения.

Синтаксис команды Display такой же, как и у List. Однако в случае подачи этой команды вывод информации на экран приостанавливается после каждой страницы, ожидая нажатия на любую клавишу. Без операндов эта команда выводит все поля только одной записи.

OPEN DATABASE <имя базы данных> - открытие базы данных.

USE <имя таблицы>- открытие таблицы.

Для организации последовательного поиска используются следующие команды :

LOCATE [границы] [WHILE<вржl>] [FOR <вржl>] – просматривает активную таблицу от начала до конца, пока не найдет первую запись, для которой логическое выражение, следующее за FOR, истинно. Совместно с LOCATE применяется команда CONTINUE.

SEEK (<вриж> [,<имя таблицы>][,<имя индексного файла>] [<номер рабочей области>]) – выполняет поиск записи с указанным <вриж> в индексном файле открытой таблицы, устанавливает указатель на найденную запись и возвращает значение .T. В отличие от SEEK команда FIND применяется в случае, если поиск записи выполняется только по значению символьного поля. FIND <искомая строка/подстрока символов>

SORT <границы> TO <имя таблицы> [ASCENDING/DESCENDING]

ON <поле> [/A][/C][/D] [,<поле> [/A][/C][/D]]

[WHILE <вржl>] [FOR < вржl >] [FIELDS <список полей>] – для создания таблицы, в которой записи отсортированы по указанным полям. [/A][/C][/D] – ключи для применения сортировки по возрастанию, независимо от строчных и прописных букв, по убыванию соответственно. По умолчанию данные сортируются по возрастанию.

Команда SET FILTER to <вржl> - выделяет в таблице группу записей, для которых логическое выражение истинно. Для снятия фильтра данная команда выполняется без операнда.

Команды управления и организации циклов организованы с использованием ключевых слов: IF.....ENDIF, DO WHILE .....ENDDO, FOR .....ENDFOR, DO CASE.....ENDCASE.

SCAN [границы] [FOR-условие] [WHILE-условие] [LOOP][EXIT]

<команды>

ENDSCAN-- команда для сканирования таблицы, причем для каждой записи, удовлетворяющей условиям, выполняются <команды>. LOOP-используется для передачи управления на начало цикла, не выполняя его полнос-

тью, EXIT-позволяет выйти за пределы незавершенного цикла, при этом управление передается следующей за ENDSCAN команде (аналогично ENDDO, ENDFOR).

Имеется большое количество арифметических, логических строковых функций, функций обработки даты и других.

Нельзя забывать, что Visual FoxPro – это платформа, поддерживающая гибридный язык программирования. Т.е. работая в данной среде, можно создавать программы, используя как модульное или процедурное программирование, так и объектно-ориентированное.

Объектно-ориентированная программа создается из заранее подготовленных программных модулей – объектов (Objects), которые являются совокупностью данных и функций – т.е. методов выполнения определенных действий, и которые можно изменять. Объектом может быть форма, элемент управления , таблица и т.д. Например, для получения диалогового окна достаточно выполнить команду: Messagebox («текст сообщения», <врж>, «заголовок окна»).

<врж> - суммарное значение кодов, которые определяют внешний вид окна.

Каждый объект имеет свойства (Properties), управляется событием, при возникновении которого выполняется связанный с ним программный код метода обработки этого события. Метод может быть и не привязан к событию и вызываться на исполнение в любой момент времени. Например, Show – отображает объект, Hide – скрывает объект. Синтаксис команд объектно – ориентированного программирования следующий:

- Для запуска метода обработки события для данного объекта выполнить команду <объект>.<событие>[LPARAMETERS<уникальный идентификатор элемента управления>]
- Для изменения свойства объекта <объект>.<свойство>=<значение>
- Для выполнения метода <объект>.<название метода>[(<параметры>)]
- Если объект является составной часть контейнера или родительского класса —

<Контейнер>.<объект>.<свойство>/<метод>

Задание на лабораторную работу

1. Создать форму для работы с базой данных, которая содержит поля базы данных и кнопки с именами:

- последовательный поиск;
- поиск в индексном файле;
- поиск в отсортированном файле;

- поиск с наложением фильтра;
- поиск с использованием команды Select;
- поиск с использованием сканирования.

2. Учесть ограничения ссылочной целостности с помощью транкзаций на операции добавления, изменения, удаления записей.

ЛАБОРАТОРНАЯ РАБОТА №12

Тема: Установление связей между таблицами базы данных, используя возможности языка FoxPro

При сложной взаимосвязи таблиц бывает удобно программным способом установить нужные связи, предварительно проиндексировав их по полю связи. Индексирование и создание индексного файла с расширением .idx для открытой таблицы выполняется командой

INDEX ON <вржI> TO <имя индексного файла> [FOR <вржI>] [DESCENDING] [UNIQUE] [ADDITIVE] [COMPACT]

В качестве <вржI> чаще всего используют имя поля , по которому упорядочивают таблицы, но индекс может быть и сложным. Опция DESCENDING – чаще всего используется для упорядочения записей по убыванию <вржI> (по умолчанию-по возрастанию). Опция UNIQUE позволяет игнорировать записи с повторяющимися значениям <вржI>. ADDITIVE – позволяет создавать индекс, не закрывая уже открытые индексные файлы. COMPACT – сохраняет индексируемые данные в строках фактической, а не фиксированной длины, а повторяющиеся элементы при этом записываются в одном экземпляре, что позволяет значительно сократить размер таблицы.

Мультииндексный файл создастся командой

COPY INDEXES <перечень файлов.idx>/ ALL [TO <имя неструктурированного файла .idx>].

Если опция TO <> не используется, то по умолчанию создается структурированный мультииндексный файл. Для большинства задач целесообразно использовать структурированный мультииндексный файл, т.к.не нужно тратить время на открытие индексов, а также выполнять реиндексацию. Для создания нового тега или дополнения файла с расширением .cdx новым тегом используют команду:

INDEX ON <вржI> TO TAG <имя тега> [OF <имя неструктурированного файла .cdx>] [FOR <вржI>] [DESCENDING] [UNIQUE] [ADDITIVE].

Из открытого мультииндексного файла указанные или все теги можно удалить командой:

DELETE TAG <имя тега1> [OF <имя.cdx1>][,... [<имя тегаN> [OF <имя.cdxN>]]]

Или DELETE TAG ALL [OF <имя.cdx>]. Открытие таблицы вместе с индексами выполняется командой:

USE <имя таблицы> IN <номер рабочей области> INDEX <список файлов .idx, .cdx> [AGAIN][NOUPDATE], причем опция AGAIN используется для открытия таблицы, но в другой рабочей области. NOUPDATE – позволяет открыть таблицу только в режиме чтения. Индексные файлы с расширением .idx при активной таблице могут быть открыты командой SET INDEX TO <список индексных файлов> [ADDITIVE].

Для определения индексного порядка используются следующие команды:

SET ORDER TO<имя файла.idx>/<номер файла.idx>/ TAG<имя тега> [DESCENDING] [ADDITIVE] - при изменении объекта поиска позволяет переназначить индекс.

Приоритет индекса или тега может быть определен также при открытии таблицы:

USE <имя таблицы> [ORDER <вржN>/<файл.idx>/ TAG <имя>
[OF <файл.cdx>] [ASCENDING/ DESCENDING]]

Безоперандная команда SET ORDER TO отключает все индексы от управления, оставляя их открытыми, что сокращает время, например, при редактировании таблицы.

Для установления связи 1:1 по указанному ключу с таблицей, размещенной в указанной рабочей области, применяется команда

SET RELATION TO key INTO <область1>[,..... [,keyN INTO <областьN>]][ADDITIVE]

Опция ADDITIVE используется для одновременной связи более двух таблиц. Предварительно связываемые таблицы должны быть проиндексированы, открыты в соответствующих рабочих областях вместе с индексными файлами. Область, в которой мы находимся в данный момент, называется активной рабочей областью, и можно работать с находящейся в ней таблицей, выполняя все допустимые команды системы.

Переход из одной рабочей области в другую осуществляется командой

SELECT <рабочая область/псевдоним>

Для обозначения первых десяти рабочих областей следует использовать буквы от A до J или номера 1-10. Если таблиц больше десяти, необходимо указывать только номера рабочих областей или псевдонимы открытых в них таблиц. Псевдонимом области по умолчанию является имя находящегося в

ней файла базы данных. В качестве псевдонима можно указать и любое другое слово в команде

USE <имя таблицы> IN <номер рабочей области> ALIAS <псевдоним>.

Для установления связи 1:N между двумя таблицами сначала устанавливается связь 1:1, а затем выполняется команда SET SKIP TO <список областей>, где указываются имена рабочих областей с таблицами, с которыми устанавливаются связи 1:N.

Задание на лабораторную работу

1. Создать форму, в которой кроме полей базы данных «Результаты сессии студентов» содержатся кнопки, позволяющие:

1.1. Установить связь 1:1 между таблицами сведений о студентах и результатами сессии и вывести сведения о студенте и результаты сдачи сессии по заданному номеру зачетной книжки.

1.2. Установить связь 1:N между таблицами сведений о студентах и результатами сессии по номеру зачетной книжки и вывести всю информацию о студентах, сдавших сессию указанного номера в срок.

1.3. Установить связь M:N между таблицами сведений о студентах и результатами сессии с использованием таблицы на связь и вывести сведения о студентах, сдавших сессии без троек (Таблица на связь хранит номера зачетных книжек и номера сессий).

1.4. Установить связь 1:N между таблицами сведений о студентах и результатами сессии по номеру зачетной книжки и связь N:1 между результатами сессии и таблицей с информацией о сессиях, содержащей информацию о номере сессии, номере семестра, виде семестра (осенний, зимний), номере курса.

ЛАБОРАТОРНАЯ РАБОТА №13

Тема: Классы и библиотеки классов

Класс объектов – это совокупность объектов с общими свойствами, присущими каждому экземпляру класса. Для создания экземпляров классов используется конструктор классов (Class Designer). Процесс создания классов в среде конструктора классов похож на аналогичный процесс создания формы в среде конструктора форм, но описание класса хранится в библиотеке классов (в файлах с расширением .vcx, .vct). Каждый из объектов, рассмотренных в предыдущих лабораторных работах, является экземпляром базового класса и наследует его характеристики. Базовые классы поставляются вместе с программным продуктом Visual FoxPro 7.0. Каждый базовый класс име-

ет свой набор методов, событий и свойств. Набор свойств класса содержит такие свойства, как Class, определяющее имя используемого класса, Base Class – имя базового класса, ClassLibrary – имя используемой библиотеки класса, ParentClass – имя родительского класса, поскольку на основе базового может быть создан новый класс, дочерний по отношению к тому, на основе которого он построен, и наследующий свойства и методы родительского класса. В отличие от базового класса, свойства и методы родительского класса можно изменять, добавлять в них новые, создавая, таким образом, пользовательский класс, который автоматически наследует все изменения родительского.

Пользовательский класс целесообразно создавать в том случае, когда новые функции действительно требуют создания отдельного класса, например для русификации проекта. Описание созданного класса помещается в библиотеку классов. В Visual FoxPro одинаково необходимо бывает использование как визуальных классов для создания одиночных и групповых элементов управления, контейнеров, экранных форм и панелей инструментов для организации пользовательского интерфейса, так и использование невизуальных классов для управления функциями и процессами в приложении. Примером такого объекта является Timer, применяемый для фиксации заданных промежутков времени с целью выполнения программой некоторых действий с заданной частотой или проверки показания системных часов для определения необходимости запуска программы или приложения.

В Visual FoxPro 7.0 имеется большой набор системных библиотек классов. На их основе создаются стандартные объекты с использованием мастера. Для просмотра информации об используемом классе, о библиотеке, в которую этот класс включен, следует выделить экземпляр класса и перейти на вкладку Other окна Properties. После чего по указанному пути следует отыскать указанную библиотеку и просмотреть ее или модифицировать. Таким образом, можно ознакомиться с любой другой библиотекой, классом, а также набором методов, разработанных высококвалифицированными программистами.

Для определения нового класса, создаваемого на базе родительского или базового, но обладающего другими значениями свойств, используют команду:

DEFINE <имя нового класса> AS <имя родительского или базового класса>

<определение значений свойств класса>

ENDFINE

Для создания нового класса необходимо выбрать следующие пункты меню File-New-Class или выполнить команду CREATE CLASS. В открывшемся диалоговом окне New Class определить следующие параметры:

Class Name – имя нового пользовательского класса;

Based On – имя класса, на основе которого создается новый класс;

Store In – имя существующей или новой библиотеки, в которой будет храниться новый класс. После чего откроется окно конструктора классов Class Designer. После сохранения созданного класса его можно добавить на панель инструментов Form Controls (пункт Add всплывающего меню). Пользовательскую библиотеку также можно добавить в число выбранных, используя вкладку Controls диалогового окна Options, выбрав опцию Visual class libraries, щелкнув на кнопке Add. Доступной она станет при использовании элемента управления View Classes панели инструментов Form Controls.

Component Gallery (Галерея компонентов) – мощное средство Visual FoxPro 7.0 для быстрой разработки приложений с использованием самых различных содержащихся в ней локальных и удаленных компонентов – библиотек и классов, готовых проектов, текстовых документов, ссылок на Web страницы, форм, отчетов и т.д. Для работы с Component Gallery следует выполнить следующую команду меню Tools- Component Gallery. В открывшемся окне в левой части представлен список стандартных каталогов Visual FoxPro, а в правой содержимое выбранного каталога. Окно Component Gallery используется по умолчанию и соответствует режиму Default. Кроме этого, в ниспадающем списке можно выбрать режим Class by Type – для вывода перечня классов, сгруппировав их по типу; режим Building Apps – для вывода перечня классов, используемых при построении приложения; режим Fast Form – для вывода перечня классов с заготовками для быстрого построения формы.

Для просмотра и модификации фундаментального класса в контекстном меню выбрать команду Modify или дважды щелкнуть на пиктограмме объекта. Откроется окно конструктора класса для редактирования свойств и методов класса, а также для модификации формы с помощью панели инструментов Form Controls. Для просмотра свойств и методов выбранного класса в окне Class Browser выберите в контекстном меню команду View in Browser. В левой части окна отобразится объект, а в правой – составляющие его объекты и соответствующие свойства и методы. После выполненных модификаций объект может быть перемещен или в форму, или в соответствующие разделы открытого проекта, или в основное окно Visual FoxPro.

Задание на лабораторную работу

1. Создать класс объектов как наследник от контейнера, в который поместить кнопки – вперед, назад, начало, конец. Добавить соответствующие коды на обработку событий нажатием кнопки. Этот объект вставить на уровне формы и показать ее работу.

2. Создать класс объектов - индикатор прогресса, состоящий из двух объектов – геометрической фигуры и текстовой метки, как наследник от контейнера. Сделайте индикатор переменного размера.

3. Добавить в форму контейнер из двух кнопок, позволяющих перемещать записи вперед и назад (фундаментальный класс – Simple Navigation Buttons), предварительно модифицировав объект, используя Галерею компонентов.

ЛАБОРАТОРНАЯ РАБОТА №14

Тема: Интегрирование в Internet приложений, созданных в Visual FoxPro 7.0

Одним из самых значительных направлений развития баз данных является взаимодействие Web технологиями, интегрирование в Интернет объектно-ориентированных приложений, в том числе и созданных Visual FoxPro 7.0.Для подключения базы данных и других компонентов приложения к страницам Web используются различные технологии, например ODBC или использование Visual FoxPro как OLE сервер. В частности, для Visual FoxPro из Web-броузера выбирается драйвер Visual FoxPro в качестве системного источника данных ODBC.

Данные Visual FoxPro становятся доступными для просмотра Web-броузером, отправки по электронной почте или размещения на Web-странице после преобразования в HTML-формат. Такое преобразование в Visual FoxPro выполняют следующие мастера:

- WWW Search Page Wizard – создает Web-страницу в формате HTML, которая запрашивает данные Visual FoxPro.
- Web Publishing Wizard – публикует данные Visual FoxPro на Web-странице.

Наиболее современной технологией внедрения приложения в Интернет является поддержка Active Document. Visual FoxPro 7.0 – это источник различных внедряемых документов, а для их просмотра в качестве броузера используется приложение Internet Explorer.

Для публикации содержимого таблицы на Web-странице нужно выпол-

нить команду Tools-Wizard- Web Publishing и, следуя указаниям мастера, завершить создание HTML-файла. HTML-файлы на основе экранных форм, запросов и отчетов Visual FoxPro могут быть созданы при работе с соответствующими конструкторами путем сохранения результата конструирования в HTML-файле.

HTML-файлы на основе экранных форм, отчетов и таблиц Visual FoxPro могут быть созданы с помощью Component Gallery путем использования специализированных классов из библиотеки фундаментальных классов, которые выполняют преобразование в HTML-формат.

Выполнив команду меню Tools – Component Gallery, для поиска специализированных классов последовательно открыть каталоги Visual FoxPro Catalog-Foundation Classes-Internet. В правой части окна галереи выбрать класс для преобразования компонентов в HTML-формат:

- Для таблицы _dbf2html-DBF – HTML;
- Для отчета _frx2html-FRX – HTML;
- Для экранной формы _scx2html-SCX – HTML.

Можно создать новую форму, выбрав класс SCX-HTML и перетащив его на форму. В окне построителя HTML-файла для формы в области ввода Source file задать имя файла формы источника, а в списке Output options выбрать один из вариантов дальнейшей работы с созданным файлом.

Добавив в форму командную кнопку, инициализирующую преобразование, введя следующий код:

```
ThisForm._csx2html1.nGenOutput=2
ThisForm._csx2html1.GenHTML()
```

В табл. 3 приведены возможные значения свойства nGenOutput класса _scx2html.

Таблица 3

0	Создается выходной файл
1	Создается выходной файл и отображается в редакторе FoxPro
2	Создается выходной файл и отображается в окне Internet-броузера
3	На экран выводится диалоговое окно Save As для ввода имени файла
4	Создается выходной файл и объект _oHTML
5	Создается объект _oHTML без сносок на файл-источник или выходной файл

Непосредственно преобразует файл в HTML-формат метод GenHTML(), который вызывается командой меню File – Save As HTML. После запуска формы на выполнение создается выходной файл и отображается в окне Internet-броузера.

Приложение Active Document создается в конструкторе проектов Visual FoxPro, имеет расширение .app и может выполнять все его функции. Главной особенностью приложения Active Document является то, что в проект добавляется библиотека, в которую включен класс, основанный на базовом классе _activdoc, поддерживающий свойства, события, методы для взаимодействия с Web-броузером. Именно этот класс и назначается в качестве главной программы приложения установкой Set main приложения Active Document. При запуске приложения активизируется специализированный OLE-объект активного документа, наследующий свойства, события и методы класса _activdoc. Возможности форм, используемых в приложении Active Document, значительно больше благодаря добавлению специальных свойств, событий и методов.

Задание на лабораторную работу

1.Создать HTML-файлы таблицы, отчета, запроса, экранной формы и просмотреть.

2. Создать HTML-файл экранной формы с помощью Component Gallery.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Базы данных: модели, разработка, реализация / Т.С. Карпова. – СПб.: Питер, 2002. – 304с.

2. Баженова И.Ю. Visual FoxPro 5.0. Объектно-ориентированные средства программирования – М.:Диалог-МИФИ, 1997.-320 с.

3. Каратыгин С.А., Тихонов А.Ф., Тихонова Л.Н. Visual FoxPro 6 – М.: ЗАО «Издательство БИНОМ», 1999.-784с.

4. Попов А.А. Программирование в среде СУБД FoxPro 2.0. Построение систем обработки данных.- М.: Радио и связь; Киев:ТОО «ВЕК», 1995.-352 с.

5. Мусина Т.В., Пушенко В.А. М91 Visual FoxPro 7.0. Учебный курс – К.: ВЕК+, Киев: BookStar, 2002. – 400 с.

Учебное издание

VISUAL FOXPRO – СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

Методические указания
к лабораторному практикуму

Составитель Логанова Лилия Владимировна

Редактор Т.К. Кретинина
Компьютерная верстка И.И. Спиридонова

Подписано в печать 29.03.04. Формат 60x84 1/16.
Бумага офсетная. Печать офсетная..
Усл.печ.л. 2,44. Усл.кр.-отт. 2,56. Уч.-изд. л. 2,75.
Тираж 60 экз. Заказ 42. Арт. С-3(Д1)/2004.

Самарский государственный аэрокосмический университет
имени академика С.П. Королева.
443086 Самара. Московское шоссе, 34

Отпечатано в УПЛ.
443056, Самара, пр. Масленникова, 37