МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ВОЛОГОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра «Управляющие и вычислительные системы»

СУБД ACCESS
Инструментальные средства

Методические указания и задания к лабораторным работам по дисциплине «Базы данных»

Факультет электроэнергетический
Специальность 230101 – Вычислительные машины комплексы, системы и сети
Направление 230100 - Информатика и вычислительная техника

Вологда
2011

УДК 681.3.06

СУБД ACCESS. Инструментальные средства: методические указания и задания к лабораторным работам по дисциплине «Базы данных».- Вологда: ВоГТУ, 2011. - 24 с.

Приведены методические указания по разработке баз данных на основе СУБД ACCESS и применению инструментальных средств СУБД: таблиц, запросов, форм и отчетов. Предлагается цикл лабораторных работ для студентов дневной формы обучения.

Утверждено редакционно-издательским советом ВоГТУ

Составитель В.Н. Черняев, канд. техн. наук, доцент

Рецензент В.П. Ананьев, канд. техн. наук, доцент

ВВЕДЕНИЕ

Хранение и обработка колоссальных объемов данных, которые циркулируют в настоящее время, невозможны без применения компьютерной техники. Существует большой выбор приложений, поддерживающих хранение и обработку данных. Они называются *Системами Управления Базами Данных* (СУБД).

Эксплуатационными показателями качества базы данных являются следующие:

- минимальный объем компьютерной памяти (уникальность каждого элемента данных по всей базе);
- максимально быстрый доступ к любому элементу данных;
- удобство текущей эксплуатации базы данных.

Наилучшей по этим показателям структурой базы данных ныне признана **реляционная** (Relation – «Отношение», «Связь»). Данные в реляционной базе хранятся в системе *связанных* друг с другом **таблиц**.

В рамках данной дисциплины рассматривается СУБД ACCESS, как один из компонентов широко известного комплекса приложений **MICROSOFT OFFICE.** Это приложение обладает высокой мобильностью (устанавливается на компьютер в составе **OFFICE**) и вместе с тем позволяет создавать базы данных (БД), удовлетворяющие самым изысканным требованиям.

При построении любой базы данных первый и наиболее важный шаг – формальное описание той предметной области, данные которой, собственно, и предполагается хранить и обрабатывать.

Предметная область

В цикле лабораторных работ предлагается разработать базу данных для некоего лечебного учреждения **«ПРОФИЛАКТОРИЙ»**

Условия:

- Количество пациентов — не ограничено
- Время лечения — 20 дней
- Лечение – исключительно медикаментозное или процедурное
- По каждому пациенту фиксируется:
  - ФИО
  - Дата поступления

- Каждому пациенту назначается лечащий врач (терапевт)
- Каждый пациент может лечиться от нескольких заболеваний
  - по каждому заболеванию пациента хранится текущее состояние: (*улучшение, ухудшение, без изменений*);
  - для лечения каждого заболевания применяется сколько угодно доступных процедур.

В данном пособии предлагается цикл лабораторных работ по созданию БД для обработки информации о медицинской стороне деятельности профилактория, в соответствии с приведенным выше описанием. Инструментальные средства БД, приемы работы с ней и виды документации описываются в соответствующих разделах.

Лабораторная работа 1
ТАБЛИЦЫ И СХЕМА ДАННЫХ

Данные в реляционной базе хранятся во взаимосвязанных таблицах. Строки таблицы называются **записями**, а столбцы – **полями**. Очевидно, все записи имеют одну и ту же структуру (набор полей). Количество записей в таблице неограничено.

При разработке системы таблиц следует обеспечить первый из названных выше показателей качества базы (уникальность хранимых данных). Достигается это следующим образом: в той таблице, где элемент данных встречается первый и единственный раз, он должен быть автоматически увязан с уникальным числовым значением (кодом), стоящим в специальном поле той же записи. В любом месте базы, где требуется обращение к этому элементу данных, появляется не сам элемент, а его код. Таким образом, в базе дублируются не данные, а только внутренние коды.

Одно или несколько полей таблицы объявляются **ключевыми.** В таблице не может быть записей, совпадающих по ключевым полям.

Ключевые поля («ключи») автоматически оказываются **индексированными**. Таковыми же можно объявить и любые другие поля. При этом для каждого индексированного поля в ACCESS создается внутренняя **индексная таблица**, состоящая из двух полей: в первом в упорядоченном виде хранятся все имеющиеся значения индексированного поля, во втором – номера записей основной таблицы, где эти значения встречаются. Таким образом, в ин-

дексной таблице столько же записей, сколько и в основной, причем у каждой основной таблицы может быть несколько индексных. Естественно, это приводит к дополнительным затратам памяти, поэтому слишком «широкие» поля индексировать не стоит. Индексирование обеспечивает максимально быстрый поиск данных в базе. Видимо, не стоит индексировать и те поля, по которым не предполагается поиск данных в дальнейшем.

1.1. Конструктор таблиц

Наиболее гибким инструментом разработки таблиц является **конструктор таблиц**. При разработке таблицы необходимо:

- назначить имена для всех полей;
- назначить тип данных для всех полей;
- установить требуемые свойства полей;
- установить ключ (ключи)

Собственное имя присваивается таблице при ее сохранении (при закрытии конструктора).

Тип данных выбирается из списка (рис. 1). Поясним особенности некоторых из этих типов.

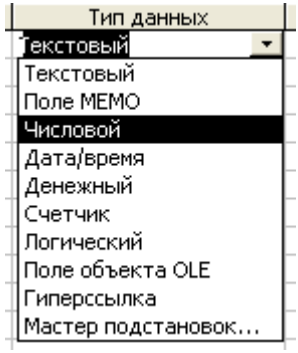

Рис. 1. Типы данных в Конструкторе

Если требуется хранить большие текстовые данные (до 65 килобайт), следует выбрать не тип *«Текстовый»*, а тип *«Поле MEMO»*.

Тип *«Счетчик»* автоматически генерирует уникальные числовые значения, нарастающие на 1 от записи к записи. Весьма удобен для ключевого поля.

Тип *«Поле объекта OLE»* позволяет хранить практически любой фрагмент информации, циркулирующей в современных компьютерах и компьютерных сетях – документ Word, таблицу Excel, другую базу ACCESS, рисунок после сканера и т.д.

Элемент данных типа *«Гиперссылка»* хранится вне данной базы (и вне компьютера, например, в Интернет). В таблице – только его адрес.

1.2. Свойства поля

Набор свойств зависит от назначенного типа данных. Свойства устанавливаются на вкладке «Общие», внизу экрана конструктора (рис. 2). Особых комментариев они не требуют. Не следует обходить вниманием свойство «Подпись». Если Подпись задана, то она будет появляться везде, где при ее отсутствии появлялось бы внутреннее имя поля (а оно может быть неудобочитаемым).

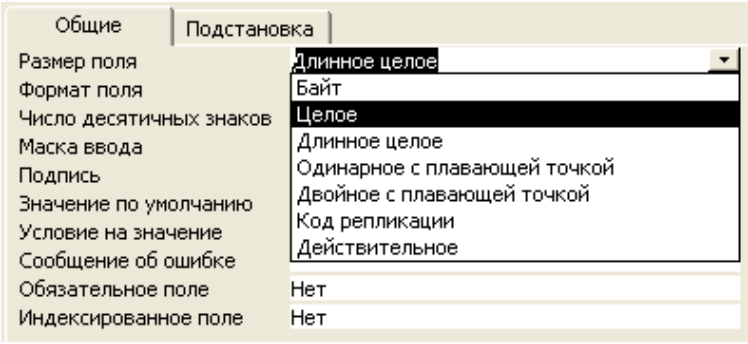

Рис. 2. Свойства поля

1.3. Подстановка

Прием «Подстановка» состоит в следующем.

Часто поля содержат повторяющиеся данные, причем набор допустимых значений не слишком велик. Во-первых, храниться должны не сами значения, а их коды (см. выше). Во-вторых, такие значения при установке в базу удобнее было бы выбирать из списка, а не вводить с клавиатуры. В-третьих, при вводе с клавиатуры часто появляются «незаметные» ошибки, с которыми данные внешне выглядят одинаково, а «внутри ACCESS» воспринимаются как различные. В таком случае удобно применить прием «Подстановка». Внешне он выглядит как выбор данных из списка.

Подстановку можно построить с помощью мастера (см. рис. 1) или «вручную». Остановимся на втором варианте.

Список подстановки может храниться как в самом макете таблицы, так и в отдельной таблице (ее часто называют «справочник»). Справочник применяют в случае, когда список подстановки

достаточно обширен или его предполагается оперативно корректировать.

При исполнении подстановки «вручную» справочник должен быть уже готов (и, возможно, заполнен). Он содержит минимум два поля: одно – типа «Счетчик», другое – те данные, которые требуется подставлять.

Подстановку настраивают в режиме конструктора таблиц, в разделе «Свойства поля», на вкладке «Подстановка»:

Тип элемента управления	**Поле со списком**
Тип источника строк	**Таблица или запрос**
Источник строк	*<ИМЯ СПРАВОЧНИКА>*
Присоединенный столбец	*<ТОТ, В КОТОРОМ КОД>*
Число столбцов	*<СКОЛЬКО В СПРАВОЧНИКЕ>*

После этого подстановка заработает, но в подстановочном списке будут видны *все поля* справочника. Чаще всего смотреть на коды бессмысленно. Избавиться от этого можно, если в свойстве «Ширина столбцов» установить видимую ширину полей в сантиметрах:

0 ; 3,5 см

Поле кода после этого сузится «до нуля».

Если список подстановки требуется разместить не в справочнике, а непосредственно в макете таблицы, параметры будут следующими:

Тип источника строк	**Список значений**
Источник строк	*<все значения через точку с запятой;>*

1.4. Схема данных

На заключительном этапе создания системы таблиц объявляются **связи** между этими таблицами.

В каждой связи участвуют две таблицы, причем связаны они по тем своим полям, которые совпадают не только по смыслу, но и по текущим значениям.

В данной БД применяется только основной тип связи – «один ко многим». Такая связь означает, что каждая запись одной («ведущей») таблицы может иметь несколько соответствующих записей в другой («ведомой»).

Связь может быть кратной (по нескольким парам полей).

При построении связей можно потребовать **обеспечение целостности данных** с возможностью **каскадного обновления** или **каскадного удаления** связанных данных. В предлагаемой БД целостность данных означает следующее: при удалении записи из таблицы «Пациенты» автоматически удаляются все связанные с нею записи из таблиц «Диагнозы» и «Назначения».

Пример схемы данных приведен на рис. 3 .

При построении следует «вынести» в поле схемы все таблицы, а затем, «перетаскивая» мышью изображения полей с одной таблицы на другую, по очереди установить все связи, задавая свойство обеспечения целостности.

Рис. 3. Схема данных

ЗАДАНИЕ к лабораторной работе 1

Создать систему таблиц базы банных, соответствующую описанию предметной области во ВВЕДЕНИИ. Назначить разумные типы данных и свойства полей. Структура таблиц приведена ниже. Ключевые поля выделены жирным.

ПАЦИЕНТЫ	ДИАГНОЗЫ	НАЗНАЧЕНИЯ
№пациента_П	**№пациента_Д**	**№пациента_Н**
ФИО	**№заболевания_Д**	**№заболевания_Н**
Дата_пост	Состояние	**№процедуры_Н**

ЗАБОЛЕВАНИЯ	ПРОЦЕДУРЫ	ВРАЧИ
№заболевания	**№процедуры**	**№врача**

Заболевание Процедура ФИО_врача

Построить схему данных.

Ввести в структуру данных новый элемент – «Лечащий врач» - с учетом следующих обстоятельств:

- все врачи – терапевты одинаковой квалификации
- врач закрепляется за конкретным пациентом и в состоянии организовать лечение любого заболевания.

В задании соответствующая таблица уже упомянута, но место этих данных в общей структуре не определено.

Лабораторная работа 2
ЭКРАННЫЕ ФОРМЫ

Формы – основной инструмент построения интерфейса между пользователем и БД. Они обеспечивают ввод, просмотр и редактирование данных, а также управление работой БД в целом.

Имеются три «уровня» разработки форм: **автоматический**, с помощью **мастера форм** или в режиме **конструктора**. Последний, как и в плане таблиц, имеет наибольшую гибкость, поэтому будем пользоваться именно им. Можно также сделать с помощью мастера *заготовку* формы, а потом *доработать* ее в режиме конструктора. Это избавит от большинства «нетворческих» шагов.

Известны несколько типов форм:

- табличная (выглядит примерно так же, как и «открытая» таблица);
- ленточная (по характеру та же таблица, но более «изысканная»);
- «в столбец» (примитивная форма, отображающая только одну текущую запись);
- обычная (наиболее универсальный и изощренный тип формы).

Создание формы конструктором сводится к последовательному размещению в ее экранном окне **элементов управления** и настройке их **свойств**. Окно формы состоит из трех областей: **об-**

ласти данных, заголовка и **примечания**. Любую из них можно «выключить», установив ее высоту равной нулю.

Понятие «Свойства» - одно из центральных. Остановимся на нем подробнее.

В окне свойств приводятся свойства или формы в целом, или одной из ее областей, или любого элемента управления и группы элементов – т.е. того объекта, который предварительно был помечен щелчком (щелчками) «мыши». Свойства сведены в пять категорий:

- **макет** (все, что относится к внешнему виду);
- **данные** (из какого источника БД объект получает данные, если они у него могут быть вообще);
- **события** (внешние воздействия на объект, на которые, возможно, потребуется особая реакция БД. Например, щелчок «мышкой» по *кнопке* в форме.);
- **другие** (те, которые не вошли в три предыдущие категории);
- **все** (полный перечень).

Самый простой способ открыть окно «Свойства» - через контекстное меню (щелчок правой кнопкой по объекту).

Элементы управления выносятся в форму из специальной **Панели элементов**. Ее можно включить из инструментального меню «ВИД».

Перечислим основные элементы управления.

- **Надпись** (текст в форме, который никогда не меняется);
- **Поле** (отображает данные из БД, текстовые или числовые, возможно, в специальных форматах типа *дата/время, денежный* и т.д.);
- **Выключатель, Переключатель, Флажок** (отображают данные типа *boolean*, т.е. «включено - выключено», при этом имеют разные образы на экране);
- **Группа** (обеспечивает «взаимодействие» элементов из предыдущего пункта, т.е. если «включается» один из них, то другие в группе «выключаются»);
- **Список** и **Поле со списком** (позволяют вводить данные в БД не с клавиатуры, а из фиксированного списка, встроенного непосредственно в форму);
- **Кнопка** (элемент, щелчок «мышкой» по которому вызывает специально запрограммированную реакцию БД);
- **Рамка объекта** (для отображения объектов типа OLE);

- **Набор вкладок** (отображает любые данные в любом составе, но «в несколько слоев». Видна только «верхняя вкладка». Любую вкладку можно вывести «на передний план» (и сделать активной) щелчком по ее ярлычку);
- **Подчиненная форма** (отображает данные из *источника*, с которым **связан** источник данных для основной формы, по типу «один ко многим»);
- **Рисунок** (статичный оформительский элемент);
- **Линия** (статичный оформительский элемент);
- **Прямоугольник** (статичный оформительский элемент).

При установке элементов из *Панели элементов* все они оказываются **свободными**, т.е. не связанными ни с какими данными из БД. Иногда это и требуется, например, для полей с *вычисляемыми* значениями (правила вычисления таких значений оговариваются особо). Впоследствии можно будет назначить свойство «Данные» для таких элементов.

Если к некоторым полям формы жестко привязаны данные полей из конкретной таблицы или запроса, целесообразно изначально объявить последние *источником данных* для этой формы (см. «Свойства»). При этом появится возможность выносить в форму поля не из *Панели элементов*, а из **СПИСКА ПОЛЕЙ**, который может быть открыт соответствующей командой. Поля окажутся **привязанными** к данным.

ЗАДАНИЯ к лабораторной работе 2

Задание 1. Построить **единую** форму для корректировки данных трех таблиц: «Заболевания», «Процедуры», «Врачи» (все таблицы типа «Справочник»). Структура формы приведена на рис. 4.

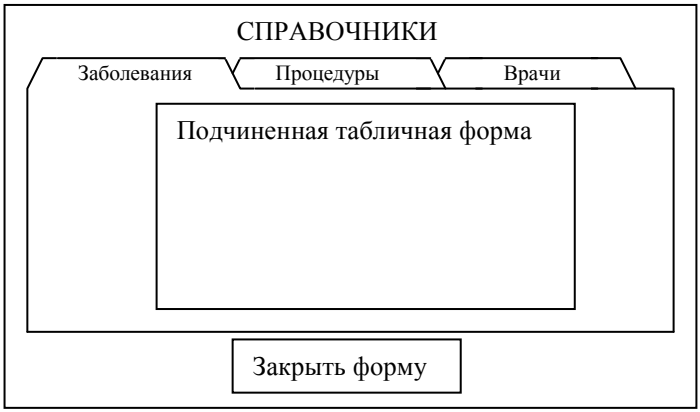

Рис. 4. Структура формы «Справочники»

1. Три таблицы уже существуют (возможно, пустые);
2. Для каждой из этих таблиц создать **табличную** форму;
3. Конструктором откорректировать макеты созданных форм в соответствии со следующими критериями:
   3.1. Все формы – одной ширины;
   3.2. Полосы прокрутки – только по вертикали;
   3.3. Отключить заголовок вместе с «шапками» таблиц;
   3.4. Исключить поле кодов (только данные);
   3.5. Выключить поле номера записи;
   3.6. Выключить область выделения;
   3.7. Выключить кнопки размеров окна, оконного меню и закрытия;
   3.8. Установить тип границы – «Отсутствует» или «Тонкая».
4. Создать Конструктором основную форму «Справочники» (источника данных у нее **нет**) и удалить «лишние» элементы макета, как в п.3.

4.1. Разместить в заголовке надпись «Справочники» и, возможно, стилизованный рисунок из файла.

4.2. Разместить в примечании кнопку «Закрыть форму», предварительно включив кнопку «Мастера» на панели элементов. Мастер кнопок последовательно задаст ряд вопросов, на которые следует обдуманно ответить.

4.3. В области данных разместить набор из трех вкладок. Изначально появятся только две вкладки. Третью следует *добавить* через меню «Вставка». Отрегулировать размеры ярлычков. Установить свойства «Подпись» для всех вкладок.

4.4. Последовательно на все три вкладки внедрить элементы «Подчиненная форма» и привязать к ним в качестве данных ранее созданные табличные формы.

4.5. При необходимости разместить в форме дополнительные элементы дизайна и провести окончательное форматирование, последовательно переключаясь в «Режим формы» и обратно в «Конструктор» и внося необходимые изменения.

Задание 2. Построить форму «Сведения о пациентах» для обработки данных из трех основных таблиц «Пациенты», «Диагнозы», «Назначения». Структура формы приведена на рис. 5.

Основная форма содержит подчиненную форму «Заболевание», а та, в свою очередь, подчиненную форму «Назначения». С каждой формой связан свой источник данных. Формы целесообразно разрабатывать, начиная с «внутренней» («Назначения»). Эта форма – табличная, а все остальные – обычные.

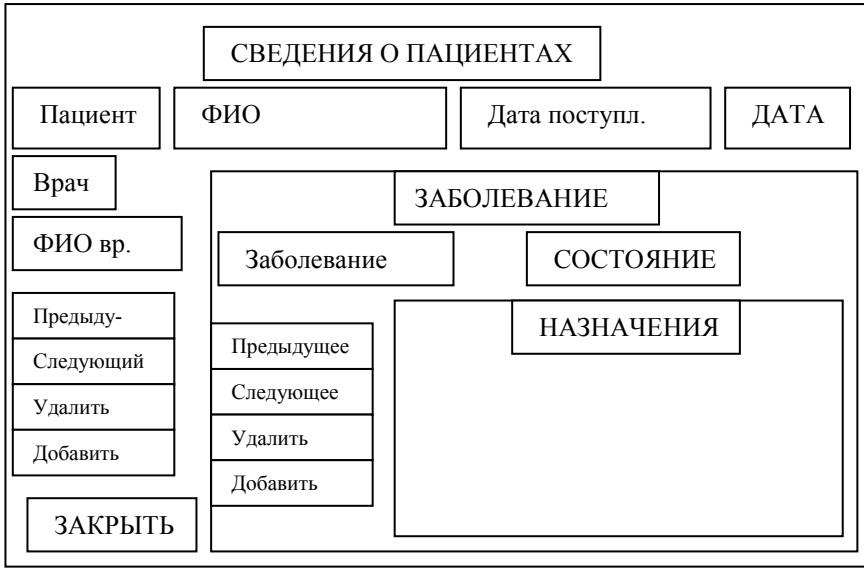

Рис. 5 . Структура формы «Сведения о пациентах»

В формы устанавливаются *Поля, Надписи* и *Управляющие Кнопки*.

Технология подготовки форм такая же, как и в задании 1.

Форма имеет следующие особенности:

- в каждый момент видны данные только по одному пациенту и только по одному его заболеванию. Переходы по записям – с помощью кнопок «Предыдущий» и «Следующий»;
- назначения для показанного заболевания видны все (таблица);
- «перелистывая» записи в форме «Заболевание», можно заметить, что доступны диагнозы только одного пациента, а не

все записи таблицы «Диагнозы», как можно было бы предположить. Таким образом, автоматически подключена внутренняя фильтрация данных. Аналогично, во внутренней форме можно увидеть назначения только для **одного** пациента, по **одному** из его заболеваний;

- вводятся только данные, коды же распространяются по таблицам в соответствии с организованными ранее подстановками и затребованным обеспечением целостности данных. Можно утверждать, что структура таблиц и схема данных выбраны такими, какие есть, только лишь потому, что в дальнейшем для их обслуживания предполагалось применить именно эту конструкцию формы.

Задание 3. Разработать ГЛАВНУЮ КНОПОЧНУЮ ФОРМУ, в которой разместить две кнопки ОТКРЫТИЯ построенных ранее форм, кнопку ЗАКРЫТИЯ БАЗЫ и любые уместные элементы дизайна. В дальнейшем в эту форму будут добавляться кнопки открытия других объектов БД.

Лабораторная работа 3
ЗАПРОСЫ

Запрос – объект БД, позволяющий автоматизировать процесс выборки нужных данных из БД, обновления и удаления данных, а также объединения данных из нескольких таблиц. Данные от запроса можно сохранить в виде таблицы. Запрос может служить источником данных для любой формы или отчета, точно так же, как и таблица. Кроме того, при необходимости запрос можно открыть в режиме таблицы.

Существуют следующие типы запросов:

- Запрос на выборку (из одной таблицы и многотабличный;
- Запрос на обновление (изменение существующих данных);
- Запрос на добавление (новых данных);
- Запрос на удаление (записей таблицы или записей каскада таблиц);
- Запрос на создание таблицы;
- Перекрестный запрос.

3.1. Запрос на выборку

Это базовый тип запроса, являющийся основой для построения всех других. Его можно создать как с помощью **Мастера**, так и **Конструктором**. Как и прежде, рассматривается последний вариант (заготовку можно создать мастером, а затем доработать ее конструктором). Экран Конструктора запросов выглядит согласно рис. 6 («бланк запроса»).

В верхнюю часть бланка выносятся таблицы, которые будут служить источником данных. Таблицы оказываются связанными в соответствии с существующей схемой данных.

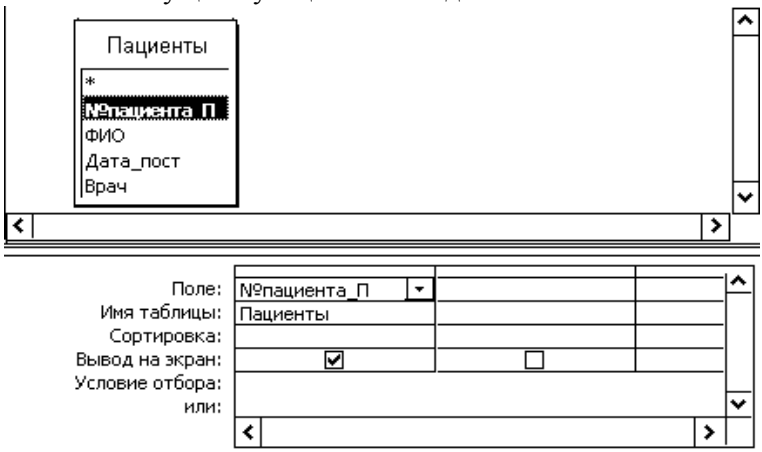

Рис. 6. Бланк простого запроса на выборку

В нижней части устанавливаются свойства полей запроса. Поля могут быть «перетащены» из таблиц верхней части. При этом автоматически заполняются графы «Поле» и «Имя таблицы». Графы «Сортировка» и «Вывод на экран» комментариев не требуют. В графе «Условие отбора» по правилам VB записывается условное выражение. В запрос будут включены только записи, удовлетворяющие этому условию (в т.ч. и по нескольким полям). Условие отбора может содержать объединение по «ИЛИ» (последние графы).

Некоторые примеры условий отбора:

>19 - отбираются все записи, значение по этому полю в которых больше 19;

Between 10 And 20 - значение поля лежит между 10 и 20 включительно;

In (1,2,15) - значение поля должно быть одним из трех перечисленных;

Like "Ц##*" Or "Э##*" - текст в поле должен начинаться с букв «Ц» или «Э», далее должны следовать строго две цифры, а потом – все что угодно.

Особый вид условия отбора – фраза в квадратных скобках. Например:

[Введите число :]

Такая конструкция называется **Параметром запроса**. При открытии запроса ACCESS потребует ввести реальное значение параметра, которое будет подставлено в условие отбора на место квадратных скобок.

Запросы могут содержать **вычисляемые** поля. У них в графу «Поле» устанавливается не имя поля, а **выражение,** записанное по правилам VB. Такие поля могут так же выводиться или не выводиться на экран, сортироваться или содержать условие отбора. Выражение можно формировать вручную или с помощью **построителя выражений**. Его же можно применять и для задания сложных условий отбора. Построитель запускается кнопкой «Построить». Вид окна построителя показан на рис. 7.

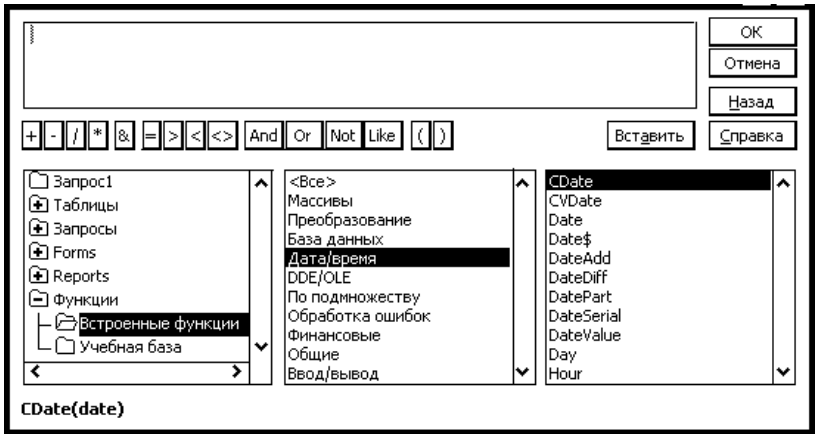

Рис. 7. Окно построителя выражений

Выражение формируется в верхней части окна, с применением имеющихся инструментов или клавиатуры. Каскад списков в нижней части окна позволяет применять стандартные функции VB или элементы пользовательских объектов (например, ссылаться на поля существующих таблиц или открытых форм). В частности, на рис. 7 в этом каскаде открыт доступ к функциям *дата/время.* Некоторые элементы выражения могут представлять собой параметры, как и в условиях отбора. Их значения всякий раз будут запрашиваться извне.

Иногда требуется получить не отдельные записи таблицы, а итоговые данные по группам записей. В этом случае строится **итоговый запрос.** Для этого в бланк запроса включается еще одна графа – «Групповая операция» (одноименной кнопкой на панели инструментов). Исходное значение в этой графе – **Группировка**. В этом случае запрос группирует записи в таблице, но выводит только первую запись из каждой группы, уникальную по всем полям запроса, и без итогов. Для получения итогов значение «Группировка» заменяется на одну из **итоговых функций:**

- **Sum** - сумма значений этого поля по всей группе;
- **Avg** - среднее арифметическое;
- **Min** - минимальное по группе;
- **Max** - максимальное по группе;
- **Count** - количество записей в группе;
- **StDev** - стандартное отклонение по группе;
- **Var** - дисперсия по группе;
- **First** - первая запись в группе;
- **Last** - последняя запись в группе.

Очевидно, некоторые из этих функций применимы только к числовым или денежным полям.

Особый тип итогового запроса – перекрестный. Он строится на базе двух и более таблиц. Одна таблица обеспечивает **заголовки строк**, вторая – **заголовки столбцов.** Требуемые **данные** могут быть взяты из любой из этих таблиц или из какой-то другой.

В качестве примера рассмотрим перекрестный запрос, дающий информацию о состоянии всех пациентов по всем их заболеваниям (рис. 8).

	ФИО	Вывих руки	Гипертония	Грипп	Расслоени
	Васнецов	без изменения	улучшение		
	Голованов		улучшение		
	Полуянов			улучшение	
	Пушкин			ухудшение	без изменени
▶	Семенов				без изменени

Рис. 8. Перекрестный запрос

В конструкторе этот запрос выглядит так, как на рис. 9.

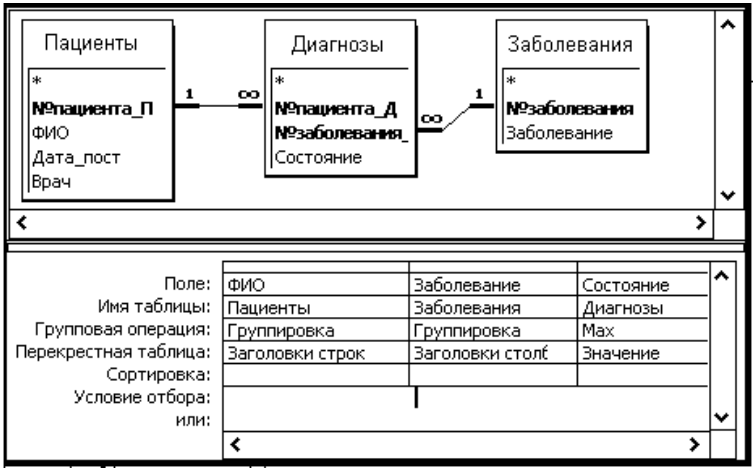

Рис. 9. Бланк перекрестного запроса в конструкторе

После установки в бланк всех требуемых таблиц дается команда меню **Запрос/Перекрестный**, после чего в бланке появляется еще одна строка – **Перекрестная таблица.** Поля запроса устанавливаются так, так описано выше. В строке «Перекрестная таблица» устанавливается соответственно «Заголовки строк», «Заголовки столбцов» или «Значение» (последнее – это те данные, которые, собственно, и требуются от запроса). Обратите внимание на строку «Групповая операция». В полях заголовков это строго «Группировка». В поле данных – итоговая функция, возвращающая неискаженное значение (Max, Min, Last, First, но не Count или Var).

Специальные запросы строятся на основе заранее созданных запросов на выборку после подачи специальной команды пере-

ключения вида запроса. Естественно, всегда сохраняется возможность и обратного переключения к запросу на выборку.

3.2. Запрос на удаление

Этот запрос удаляет из таблицы (таблиц) отобранные записи. Для преобразования запроса на выборку в запрос на удаление достаточно дать команду меню **Запрос/Удаление** в режиме конструктора.

3.3. Запрос на обновление

Запрос строится в режиме конструктора, после подачи команды меню **Запрос/Обновление**. При этом в бланке бывшего запроса на выборку появляется особая строка – **Обновление**. В этой строке по полям, требующим обновления, устанавливаются выражения любой сложности, которые и дают новые значения для этих полей (естественно, только в ранее отобранных записях).

3.4. Запрос на создание таблицы

Преобразовать существующий запрос в запрос на создание таблицы можно командой меню **Запрос/Создание таблицы.** После этого в диалоге придется ввести имя новой таблицы и ее размещение (в текущей БД или в какой-то другой). Естественно, новая таблица будет иметь структуру исходного запроса на выборку, вплоть до совпадения имен полей. В этой таблице будут и те поля, которые в запросе были вычисляемыми. Можно запретить включение в таблицу некоторых полей запроса, сняв с них флажок «Вывод на экран». Новая таблица будет содержать только записи, попавшие под условие отбора.

3.5. Запрос на добавление

Это наиболее сложный вид запроса на изменение. Смысл его в добавлении отобранных записей в существующую таблицу, имя которой должно быть введено после подачи команды меню **Запрос/Добавление** (см. предыдущий запрос). Открытие (запуск) этого запроса обходится без проблем, если «принимающая» таблица тождественна запросу по структуре записи, именам полей и

типам данных. Если это не так – в строке «Добавление» бланка следует указать имена полей, действующие в принимающей таблице.

ЗАДАНИЯ к лабораторной работе 3

Задание 1. Построить запрос на выборку «Осталось», показывающий, сколько дней осталось каждому пациенту до выписки из профилактория (срок пребывания – 20 дней). Базовая таблица – «Пациенты». Имеется вычисляемое поле:

20 – (Date() – [Пациенты]![Дата_пост])

Если запрос возвращает отрицательные значения значит из БД не удалены ранее выписавшиеся пациенты. Чтобы на этих местах появились нули, достаточно модифицировать вычисляемое поле:

IIf(20 – (Date() – [Пациенты]![Дата_пост])<0;0;
20 – (Date() – [Пациенты]![Дата_пост]))

Задание 2. Построить запрос на выборку «Выписать», показывающий пациентов с истекшим сроком пребывания в профилактории (20 дней), подлежащих выписке. Базовая таблица – «Пациенты». Установить условие отбора по полю «Дата_пост»:

(Date() – [Пациенты]![Дата_пост])>19

Убедиться в правильности работы запроса.

Задание 3. Построить запрос с параметром «По заболеванию», позволяющий:

- ввести с клавиатуры название заболевания;
- получить данные обо всех пациентах, страдающих этим заболеванием.

Базовые таблицы – две: «Пациенты» и «Заболевания»

Задание 4. Построить итоговый запрос «Квалификация», показывающий количество пациентов, закрепленных за каждым врачом, и сколько раз каждый врач зафиксировал состояние «ухудшение» у своих пациентов.

Задание 5. Построить запрос на добавление, пополняющий архив данных о выписанных пациентах, и связанный с ним запрос на удаление выписанных пациентов из основных таблиц.

Лабораторная работа 4
ОТЧЕТЫ

Отчет предназначен для вывода информации на бумагу. Отчет целесообразно строить так же, как и экранную форму – с помощью **конструктора отчетов**. Как и в случае формы, в области отчета следует установить необходимые элементы управления и отформатировать их.

Далее назовем только отличительные признаки отчета по отношению к форме.

Отчет может быть многостраничным. Поэтому в конструкторе определены не три, а пять областей:

- область данных;
- область заголовка;
- область примечания;
- верхний колонтитул;
- нижний колонтитул.

Заголовок и примечание (если они есть) встречаются в отчете по одному разу, на своих естественных местах. Колонтитулы (если они есть) устанавливаются в начале и в конце каждой страницы. В них принято размещать однотипную для всех страниц информацию (например, номер страницы, текущая дата и т.д.).

При компоновке отчета следует заботиться о его реальной ширине: она не должна превосходить ширину стандартной бумаги за минусом ширины правого и левого полей. Поля устанавливаются через инструментальное меню, команда *Файл / Параметры страницы*.

Размеры элементов управления на бумаге оказываются такими, какие зафиксированы в конструкторе. Исключение составляет только «Подчиненный отчет». Его высота в конструкторе устанавливается минимально необходимой для отображения одной строки данных в заданном формате. Высота после открытия отчета определяется динамически, в зависимости от количества информации в подчиненном отчете (на бумаге теряет смысл такой элемент, как «Полоса прокрутки»).

В отчетах не применяются некоторые элементы управления, широко распространенные в формах: «Кнопка», «Раскрывающийся список», «Поле со списком», «Флажок», «Переключатель», «Набор вкладок».

Наоборот, большое значение приобретают элементы дизайна (линии, прямоугольники, рисунки) и особый элемент – «Разрыв страницы». Он позволяет при необходимости перевести текст на следующую страницу, пока не кончилась текущая.

Часто применяются отчеты с параметрами. При открытии такого отчета на экране последовательно появятся приглашения для ввода текущих значений всех параметров, и только после этого отчет пойдет на принтер. Параметры внедряются в отчет следующим образом: устанавливают свободное поле, затем в его свойство «Данные» вносят любую фразу, которую хотелось бы видеть на экране в качестве приглашения.

Любой отчет может быть открыт на экране для предварительного просмотра.

ЗАДАНИЕ к лабораторной работе 4

Разработать отчет «Выписной лист» согласно рис. 10. Отчет содержит подчиненный табличный отчет «Заболевания» с полями «Заболевание» и «Состояние». Источником данных отчета должны быть запросы на выборку. Следует обеспечить возможность подключения любого из имеющихся запросов с помощью макроса и, таким образом, печатать выписные листы для различных групп пациентов одним и тем же отчетом (по фамилии, по дате поступления и т.д.).

Поле «Дата» содержит текущую дату (дату выписки). Следует также установить элемент «Разрыв страницы», чтобы на одной странице печатался только один выписной лист.

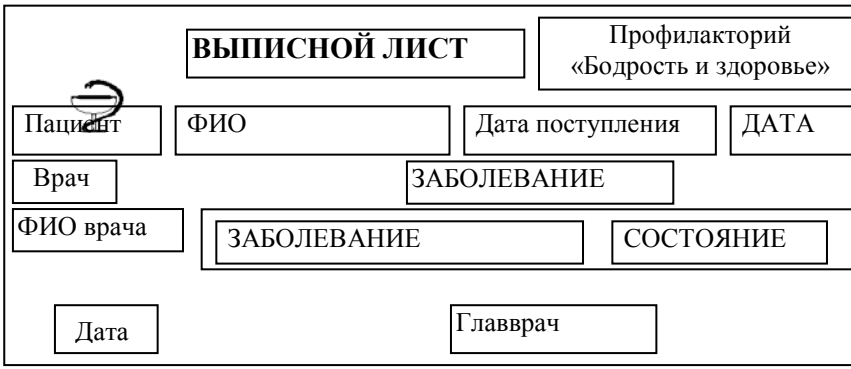

Рис. 10. Макет отчета

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

Основная литература

1.Базы данных: учебник для ВУЗов/ под ред. А.Д. Хомоненко. -СПб: Корона Принт, 2000. – 416 с.:ил.

2.Праг, К.Н. Секреты ACCESS для Win95/ К.Н. Праг,У.С. Амо, Дж. Д. Фокселл.-Киев: Диалектика,1996. – 526 с.:ил.

3.Харитонова, И. Программирование в ACCESS 2002/ И. Харитонова, Н. Вольман. -СПб.: Питер, 2002. –476 с.:ил.

Дополнительная литература

1.Голицына, О. Базы данных: учебное пособие / О.Л. Голицина, Н.В. Максимов, И.И. Попов.- М.: ФОРУМ, 2007. -400 с.:ил.

СОДЕРЖАНИЕ

Подписано в печать 20.06.2011. Усл. печ. л. . Тираж .
Печать офсетная. Бумага офисная. Заказ № _____

Отпечатано: РИО ВоГТУ, г. Вологда, ул. Ленина, 15