

МГТУ им Н.Э. Баумана

Мельников В.Ю.

**Методические указания
к лабораторным работам
по курсу «Базы данных»
на OpenOffice.org Base**

2010

Оглавление

ВВЕДЕНИЕ	4
СОЗДАНИЕ БАЗЫ ДАННЫХ В «OPENOFFICE».....	4
Создание базы данных	4
Создание таблиц.....	4
Таблицы учебной базы данных	6
Создание связей	6
Создание рекурсивных связей.....	7
Ввод данных в таблицы	8
Дефрагментизация таблиц	9
СОЗДАНИЕ ЗАПРОСОВ В «OPENOFFICE».....	10
Создание простых запросов	10
Вычисление значений выражений	12
Работа с NULL значениями.....	14
Исключение повторяющихся записей	14
Запрос данных из связанных таблиц	15
Контрольные задачи на простые запросы	16
Правое и левое соединение таблиц	16
Черная кошка в темной комнате	19
Если таблицы не соединять или «все комбинации».....	20
Статистические функции	21
Запросы непосредственно на SQL	22
Контрольные задачи на запросы на SQL	26
ЗАПРОСЫ НА ИЗМЕНЕНИЕ ДАННЫХ	27
Запрос SELECT ... INTO	27
Запрос INSERT INTO.....	27
Запрос DELETE	28
Запрос UPDATE	28
Запрос DROP TABLE.....	29
Запрос CREATE TABLE.....	29
Запрос ALTER TABLE	29
Запрос CREATE VIEW	30
Запрос DROP VIEW	30
Пакеты запросов на изменение.....	30
Пакеты запросов, формируемые SELECT запросами.....	31
ХРАНИМЫЕ ПРОЦЕДУРЫ И ФУНКЦИИ. DATEADD, TIMEADD	32
СОЗДАНИЕ ФОРМ С ВЫБОРОМ ЗНАЧЕНИЙ ИЗ СПИСКА.....	34
Создание форм с помощью мастера.	34
Ввод данных с помощью формы.	36
Формы с расположением элементов типа столбцы или блоки.	36
Редактирование формы с помощью конструктора.	37
Создание выпадающего списка	38
Создание выпадающего списка в таблице.....	39
Создание независимых субформ	39
ДОСТУП К ЭЛЕМЕНТАМ ФОРМ ИЗ ПРОГРАММ НА BASIC.....	40
Библиотеки макросов.....	40
Вызов функции по нажатию кнопки	41
Настройка вызова редактора макросов по кнопке.....	41
Получение значений, введенных в поля формы	42
Изменение значений, в полях формы	42
Получение данных из текущей записи формы	42
Изменение данных текущей записи формы	42

Представление зависимых таблиц на одной форме	43
Представление независимых таблиц на одной форме	43
Обновление форм после изменения данных в базе.	44
Обновление содержимого выпадающего списка.	45
Изменение запроса, связанного с формой	45
Открытие формы по нажатию кнопки.....	47
Выполнение SQL запросов из макросов.....	48
Выполнение запросов на изменение данных	48
Универсальная форма для запросов на изменение данных	48
Выполнение параметрических запросов	49
Выполнение запросов на выборку данных	49
Загрузка данных в массив и вывод списка	49
Загрузка файла в базу данных и выгрузка файла из базы	50
Приложение 1 - Встроенные функции и процедуры HSQLDB.....	53
Функции для работы с числами	53
Функции для работы со строками.....	53
Функции для работы с датой/временем.....	54
Функции информации о подключении к базе данных	54
Прочие функции.....	54
Литература	55

Введение

Данное руководство разработано на основе опыта автора в разработке информационных систем, с учетом трехлетнего опыта проведения лабораторных работ на OpenOffice в МГТУ им Н.Э. Баумана.

Упор делается на практические примеры с пояснениями, по аналогии с которыми можно создавать собственные запросы, формы и макросы.

Создание базы данных в «OpenOffice».

В рамках данного курса мы будем работать со встроенной СУБД «HSQL».

Создание базы данных

- Запустить программу «OpenOffice.org Base»
 - Выбирать переключатель «Создать новую базу данных» и нажать кнопку «Готово»
 - Выбрать папку и ввести имя файла базы данных. Нажать кнопку «Сохранить».
- Открывается главное окно программы.

Создание таблиц

- В левом поле главного окна выбрать элемент «Таблицы»
 - В поле «Задачи» выбрать элемент «Создать таблицу в режиме дизайна»
- Открывается конструктор таблиц. В него надо ввести описание создаваемой таблицы

Каждая строка таблички в верхней части окна конструктора содержит описание одной колонки (поля) создаваемой таблицы.

- В колонку «Название поля» вводим наименование колонки таблицы.
- В колонку «Тип поля» вводим тип данных для этой колонки.
 - INTEGER– целое число (4 байта).
 - VARCHAR– строка (ограниченной длины)
 - DOUBLE – вещественной число с двойной точностью (8 байт). Подходит для научных вычислений, но для подсчета денег, используйте NUMERIC, иначе ваш баланс не сойдется на 1-2 копейки. В финансовых расчетах свои особенности округления при вычислениях.

- NUMERIC – число с заданным числом знаков до и после запятой
- DATE – Дата – без учета времени
- TIME – Время – без учета даты
- TIMESTAMP – Дата + время
- LONGVARBINARY – Двоичные данные произвольной длины. Содержимое файла.
- В колонку «Описание» можно ввести примечание. Эта строка больше нигде не участвует.
- Обязательно, для одного из полей в левой колонке (серого цвета) из контекстного меню следует выбрать пункт «Первичный ключ». Значения в соответствующей колонке таблицы не должны повторяться. При попытке добавления записи с повторяющимся значением в ключевом поле будет выдана ошибка. Ключевое поле используется для быстрого поиска записей при выполнении запросов.

В некоторых случаях используется составной ключ. Для этого, надо выделить несколько полей (щелчками левой кнопки мыши в серой колонке при нажатой клавише «Ctrl») и из контекстного меню выбрать пункт «Первичный ключ». Комбинации значений в соответствующих полях не должны повторяться. Опыт показывает, что запросы, использующие составной ключ хуже оптимизируются СУБД и в некоторых случаях ускоряются в несколько раз после добавления колонки простого ключа.

После задания ключевого поля таблицу рекомендуется сохранить командой «Сохранить» из меню «Файл» или используя комбинацию клавиш «Ctrl+S». При первом сохранении будет запрошено имя таблицы. Внимание! СУБД HSQL держит все таблицы в оперативной памяти. Для сохранения изменений на диск надо в главном окне программы из меню «Файл» дать команду «Сохранить» или нажать соответствующую кнопку на панели инструментов. То же относится и к другим объектам базы данных. Периодически сохраняйтесь во избежание потерь данных при аварийном завершении программы.

Для ускорения поиска записей по другим колонкам, можно создать дополнительные индексы. Для этого из меню «Сервис» следует выбрать пункт «Проектирование индекса», нажать кнопку «Новый индекс», ввести имя нового индекса и выбрать из выпадающего списка имя поля. Индексы многократно ускоряют выполнение запросов. По мере необходимости, для ускорения запросов добавляется индекс за индексом, пока почти все колонки не окажутся проиндексированы. К сожалению, за использование индексов приходится расплачиваться увеличением оперативной памяти. Еще одно замечание. Индексы не ускоряют поиск по части строки.

Для выделенного поля в нижней части окна конструктора выводятся дополнительные свойства:

- Можно задать значение по умолчанию, которое будет автоматически заноситься в поле при добавлении новой записи.
- Можно включить режим «Обязательное». При попытке сохранения записи, в котором это поле не задано, выдается ошибка. Ключевое поле всегда обязательное.
- Можно задать формат представления числа. Для этого следует нажать кнопку «...» и выбрать в открывшемся диалоге один из готовых форматов или настроить собственный.
- Для целочисленного поля [INTEGER] Можно включить режим «Автозначение». Для каждой добавляемой в таблицу записи в это поле автоматически будет заноситься новое число на 1 больше предыдущего. Даже удаление всех записей из таблицы не сбрасывает счетчик.
- Для текстового [VARCHAR] поля можно задать длину поля. Внимание! При задании связей между таблицами важно, чтобы для связываемых полей совпадал и тип поля и длина поля. К сожалению, это выясняется только при вводе данных в таблицы.

Таблицы учебной базы данных

Создайте следующие таблицы (примечания насчет связей пока игнорируйте)

Таблица	Поле	тип	Примечания
Предмет	Код	Целое	Авто – значение, первичный ключ
	Наименование	Текст	
Преподаватель	Код	Целое	Авто – значение, первичный ключ
	ФИО	Текст	
Оценка	Код	Целое	Первичный ключ, 5-отлично, 0-неаттестован
	Наименование	Текст	
Факультет	Шифр	Текст(5)	Первичный ключ
	Наименование	Текст	
Группа	Шифр	Текст(10)	Первичный ключ
	Факультет	Текст(5)	Связь с Факультет. Шифр
	Курс	Целое	
Студент	№ зачетки	Текст(10)	Первичный ключ
	Группа	Текст(10)	Связь с Группа. Шифр
	ФИО	Текст	
	Дата рождения	Дата	Формат DD.MM.YYYY
Ведомость	Код	Целое	Авто – значение, первичный ключ
	№	Целое	Регистрационный номер ведомости
	Дата	Дата	Формат DD.MM.YYYY
	Группа	Текст	Связь с Группа. Шифр
	Предмет	Целое	Связь с Предмет. Код
	Преподаватель	Целое	Связь с Преподаватель. Код
Результат	Студент	Текст(10)	Первичный ключ, Связь с Студент.№зачетки
	Ведомость	Целое	Первичный ключ, Связь с Ведомость.код
	Оценка	Целое	Связь с Оценка.Код
Сообщество	Код	Целое	Авто – значение, первичный ключ
	Наименование	Текст	
	Группа сообществ	Целое	Связь с Сообщество.Код
Член сообщества	Сообщество	Целое	Первичный ключ, Связь с Сообщество.Код
	Студент	Текст(10)	Первичный ключ, Связь с Студент.№зачетки

В случае ошибки, выберите в главном окне нужную таблицу и из меню «Правка» дайте команду «Правка». Снова откроется окно конструктора таблиц, в котором можно внести изменения.

Чтобы переименовать таблицу используйте команду «Переименовать» и из меню «Правка».

Создание связей

После создания таблиц следует создать связи между соответствующими полями таблиц. Например, значение в поле «Группа» из таблицы «Студент» должно быть равно одному из значений в поле «Шифр» из таблицы «Группа».

Наличие связей дает следующие преимущества:

- Оператор не сможет ввести неправильное значение в связанное поле.
- Оператор не сможет удалить записи, на которые ссылаются другие записи.
- В конструкторах и мастерах система автоматически выполняет за нас некоторые действия, основываясь на информации о связях.

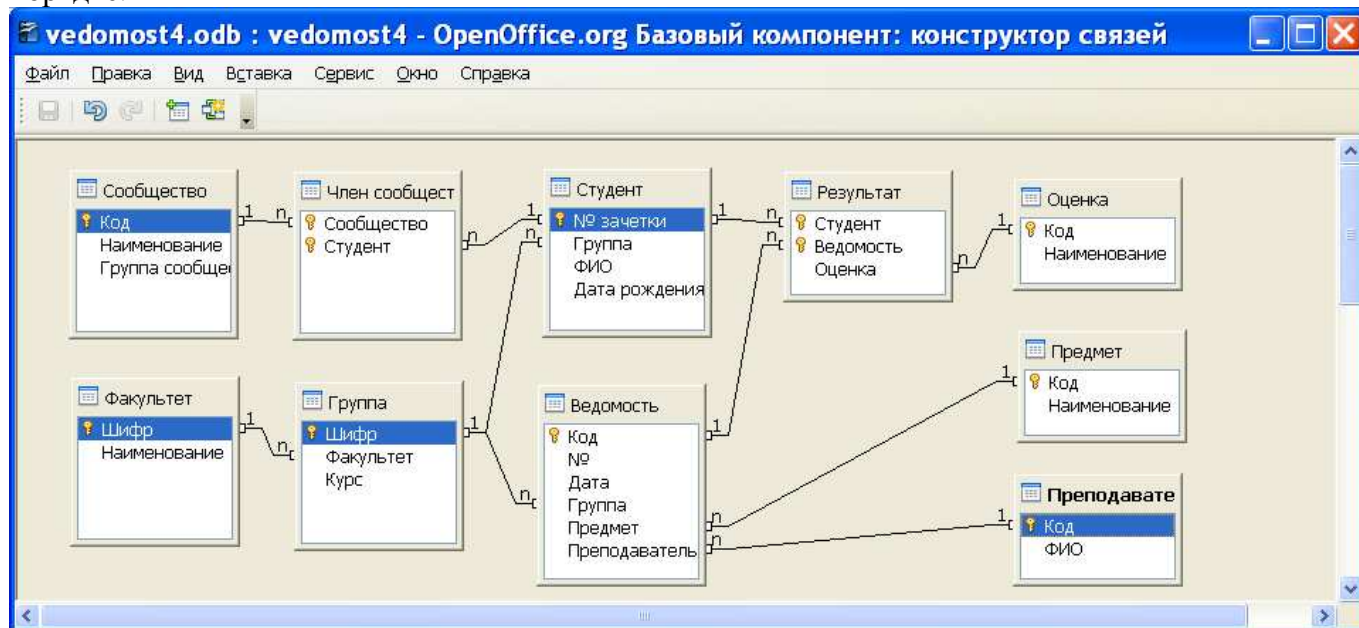
Для просмотра и создания связей, в главном окне выберите из меню «Сервис» команду «Связи». При первом открытии система автоматически предлагает добавить таблицы в окно

конструктора связей. В дальнейшем пользуйтесь командой «Добавить таблицу» из меню «Вставка». Таблицы удобно добавлять двойным щелчком мыши.

Чтобы добавить связь, подведите курсор мыши к полю одной из связываемых таблиц, нажмите левую кнопку мыши, и не отпуская ее, переместите указатель к связываемому полю другой таблицы. Если выдается сообщение об ошибке, попробуйте соединить эти поля в обратном порядке. Если и так не получается, значит, вы что-то напутали с типами или длиной полей, или в таблицу уже введены данные, которые не удовлетворяют создаваемой связи. Разбирайтесь по тексту сообщения.

Напротив связанных полей появились символы «1» и «n» это значит, что одной записи первой таблицы может соответствовать несколько записей второй таблицы.

Добавьте связи согласно приведенному ниже рисунку. Сначала проводите связи от не ключевых полей к ключевым полям. Затем можно добавить связи между ключевыми полями. При необходимости удалите неправильные связи (используя контекстное меню) и создайте их в другом порядке.



Для дополнительной настройки связи выполните на этой связи двойной щелчок левой кнопкой мыши. Открывается диалоговое окно «Связи».

Если вы хотите, например, чтобы вместе с записью о студенте удалились все его оценки, поставьте отметку «Удаление каскад». Если в свою очередь, связь между группой и студентом тоже имеет параметр каскадного удаления, то вместе с группой удалятся все студенты этой группы и их оценки.

Если параметр удаления выбран «Без действия», то СУБД не даст удалить запись, связанную этой связью с другими записями.

К сожалению, связь «Сообщество.Группа сообществ» - «Сообщество.код» нельзя добавить через интерфейс конструктора связей. Сохраните изменения и закройте конструктор связей.

Кстати, самое время, нажать кнопку «Сохранить» в главном окне «OpenOffice.org Base».

Создание рекурсивных связей

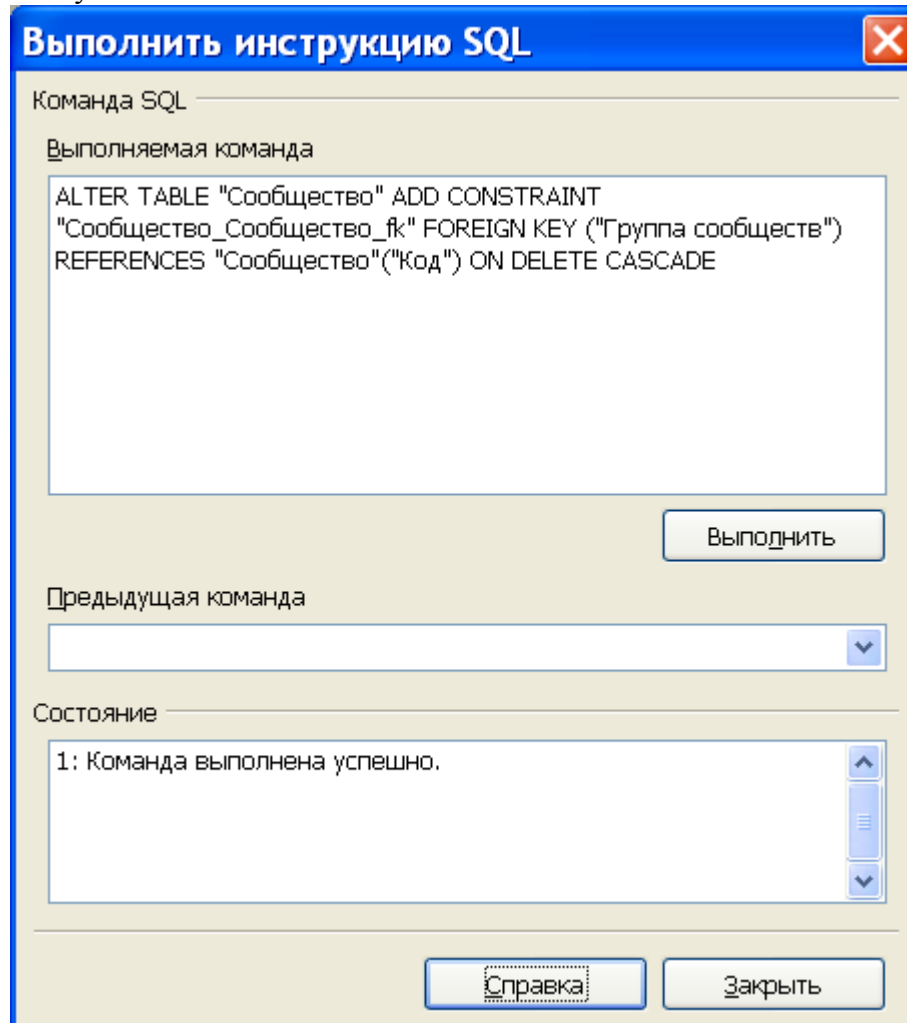
Связи, между полями одной таблицы называют рекурсивными.

Для добавления рекурсивной связи из меню «Сервис» главного окна выберите команду «SQL...». Открывается окно «Выполнить инструкцию SQL». В поле «Выполняемая команда» Введите следующий текст:

```
ALTER TABLE "Сообщество" ADD CONSTRAINT  
"Сообщество_Сообщество_fk" FOREIGN KEY ( "Группа сообществ" ) REFERENCES  
"Сообщество" ( "Код" ) ON DELETE CASCADE
```

Все имена таблиц и колонок должны быть заключены в кавычки и записаны точно так, как они были названы при создании. С символами в том же регистре и с тем же количеством пробелов. Имя связи создаваемой связи «Сообщество_Сообщество_fk» можно выбрать и более короткое.

Нажать кнопку «Выполнить».



Если команда выполнена успешно, нажимаем кнопку «заккрыть».

Иначе просматриваем текст сообщения (до конца) (не забудьте про полосу прокрутки).

Типичная ошибка: при создании таблицы или поля была сделана ошибка, а в выражении слово написано правильно. Откройте программу «Блокнот», скопируйте туда команду, закройте окно «Выполнить инструкцию SQL». Чтобы уточнить имя таблицы, выделите ее и из меню «Правка» дайте команду «Переименовать». Имя таблицы скопируйте в блокнот и нажмите кнопку «отмена». Чтобы уточнить имя поля, выберите нужную таблицу и из меню «Правка» дайте команду «Правка». Имя поля скопируйте в блокнот и закройте окно конструктора таблиц. Снова выполните команду. Не закрывайте блокнот. Может потребоваться еще одна итерация.

Откройте конструктор связей, чтобы убедиться, что связь появилась. Связь проходит за таблицей, поэтому плохо видна.

Кстати, удалить эту связь и изменить ее свойства можно и в конструкторе связей.

Ввод данных в таблицы

Откройте двойным щелчком мыши на редактирование таблицу «Факультет».

Последняя строка таблицы служит для создания новой записи. Щелчком мыши выбираем поле, и вводим данные. Для перехода в следующее поле удобно использовать клавишу «Tab».

Введенные данные сохраняются в таблице при переходе на новую строку или при нажатии кнопки «Сохранить текущую запись» на панели инструментов. При сохранении записи ошибка возникает в следующих случаях:

- Ключевое поле не заполнено, или введено уже существующее значение. (Для информации: в Microsoft Access и SQL server по умолчанию игнорируется регистр символов, так строки «АК» и «ак» будут считаться одинаковыми.)
- В связанное поле введено значение, которого нет в связанной таблице. Так мы не сможем сохранить в поле «Факультет» таблицы «Группа» значение «АК», пока не добавим соответствующую запись в таблицу «Факультет».

Чтобы отказаться от сохранения записи нажмите клавишу «Esc».

Чтобы скопировать запись, нажмите в нужной строке в левом (сером столбце) левую кнопку мыши, не отпуская ее, переместите курсор на следующую строку и отпустите. Будет выдано сообщение «Ошибка. Продолжить копирование?». Нажмите «Да». Не забудьте изменить ключевое поле перед сохранением записи.

Чтобы выделить группу расположенных подряд записей, выделяем первую из них (щелчком мыши в левом поле), затем при нажатой клавише «Shift» выделяем последнюю запись группы. Чтобы выделить или снять выделение с нескольких записей вразброс, выделяйте записи при нажатой клавише «Ctrl».

Чтобы удалить выделенные записи нажмите клавишу «Delete» и подтвердите удаление.

На панели инструментов имеются так же команды поиска и сортировки записей.

Сортируется только визуальное представление таблицы базы данных. Записи в базе могут лежать как угодно. Команда «Удалить фильтр/сортировку» отключает сортировку записей.

Команда «Быстрый фильтр» скрывает все записи, у которых значение в выделенном поле не равно выделенному значению. Это удобно при редактировании большой таблицы. Команда «Фильтр по умолчанию» позволяет задать более сложные условия отбора записей.

Вот и все небогатые возможности встроенного редактора записей таблицы.

Крайне неудобно также, вводить код связанной записи (например, код преподавателя) вместо выбора из списка.

К счастью, имеется мощный механизм форм для ввода данных, который мы рассмотрим далее.

Дефрагментизация таблиц

Обратите внимание, что, несмотря на удаление даже большого количества записей, размер файла не уменьшается. Значит, удаленные записи только помечаются, как удаленные, но остаются в файле. Кроме того, в базе данных хранится журнал событий и другая служебная информация.

Чтобы удалить всю «лишнюю» информацию из файла базы данных. Закройте все окна, кроме главного, сохраните базу данных в файл (на главном окне).

Из меню «Сервис» главного окна выберите команду «SQL...». В поле «Выполняемая команда» введите текст:

`CHECKPOINT DEFRAG`

И нажмите кнопку «Выполнить».

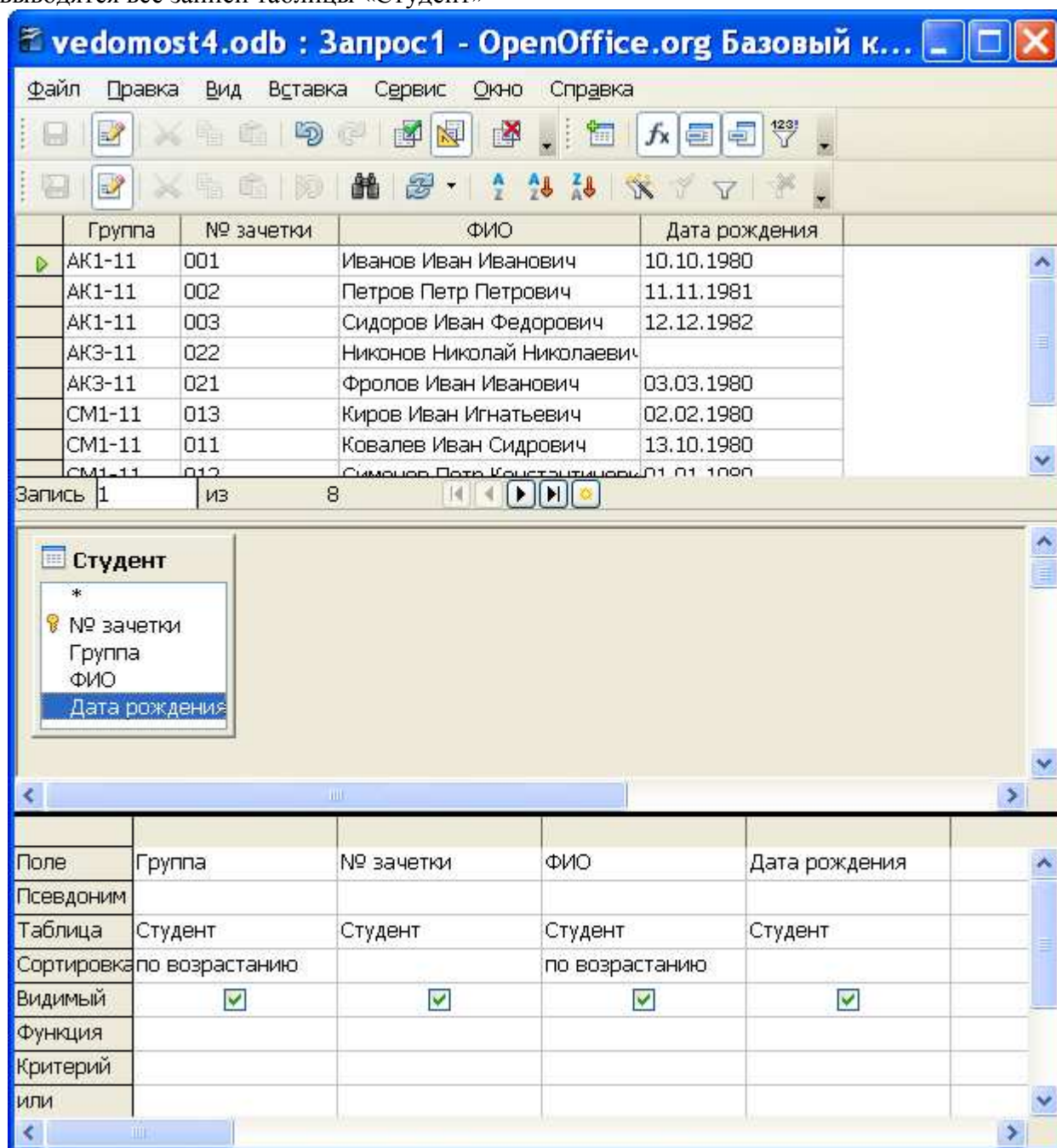
Создание запросов в «OpenOffice».

Создание простых запросов

- В левом поле главного окна выбрать элемент «Запросы»
- В поле «Задачи» выбрать элемент «Создать запрос в режиме дизайна»
- Выбрать в открывшемся окне таблицу «Студент», из которой будем выбирать данные
- Нажать кнопки «Добавить», «Заккрыть».

Открывается конструктор таблиц. В верхнем поле отображаются таблицы, из которых будет осуществляться выборка данных и связи между ними. В нижнем поле настраиваются выводимые колонки таблиц.

- Двойным щелчком на нужных полях добавляем колонки в нижнее поле.
- Для выполнения запроса нажать кнопку «Выполнить запрос» на панели инструментов. В верхней части конструктора запросов проявляется область вывода результатов, в которую выводятся все записи таблицы «Студент»



Чтобы отсортировать фамилии студентов по алфавиту надо подвести указатель мыши к полю «Сортировка» в колонке «ФИО», дважды щелкнуть левой кнопкой мыши и выбрать из раскрывшегося списка «по возрастанию». Разумеется, надо заново выполнить запрос.

Если добавить сортировку по группам получим список, отсортированный по группам, причем в пределах группы студенты отсортированы по ФИО. Всегда при сортировке левая колонка имеет приоритет.

Чтобы поменять порядок колонок, нужно нажать левую кнопку мыши в заголовке (серой строке) таблицы колонок, не отпуская ее, переместить указатель мыши в область заголовка другой колонки и отпустить кнопку мыши.

Если нас интересуют студенты только одной группы в поле «Критерий» колонки «Группа» вводим шифр нужной группы. Выполните запрос. Обратите внимание, что конструктор запросов автоматически заключил заданную строку в апострофы 'AK1-11'.

Чтобы отобразить студентов групп, начинающихся на подстроку «АК» следует задать в поле критерий для колонки «группа» строку `like 'AK*'`

На месте символа «*» может стоять 0 или более любых символов.

Чтобы найти студентов с именем «Иван» воспользуемся следующим выражением `like '* Иван *'` Обратите внимание на пробелы между «*» и именем. Если этого не сделать под условие попадут все «Ивановы» и «Ивановичи».

Удалите все условия.

Зададим условие на дату рождения `>= 11.11.1981`. Выполните запрос. Обратите внимание, что конструктор запросов автоматически добавил справа и слева от даты символ «#».

Добавим условие в поле «ИЛИ». Выполните запрос.

vedomost4.odb : Запрос1 - OpenOffice.org Базовый к...

Файл Правка Вид Вставка Сервис Окно Справка

Группа	№ зачетки	ФИО	Дата рождения
AK1-11	002	Петров Петр Петрович	11.11.1981
AK1-11	003	Сидоров Иван Федорович	12.12.1982
AK3-11	022	Никонов Николай Николаевич	
AK3-11	021	Фролов Иван Иванович	03.03.1980

Запись 1 из 4

Студент

- *
 - № зачетки
 - Группа
 - ФИО
 - Дата рождения

Поле	Группа	№ зачетки	ФИО	Дата рождения
Псевдоним				
Таблица	Студент	Студент	Студент	Студент
Сортировка	по возрастанию		по возрастанию	
Видимый	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Функция				
Критерий	'AK1-11'			>= #11.11.1981#
или	'AK3-11'			

Получили студентов из группы 'AK1-11' с датой рождения >= #11.11.1981# а так же всех студентов группы 'AK3-11'. Как понять, что получится в результате запроса?

Нажмите на панели инструментов кнопку «включить / выключить вид дизайна».

Две нижних области заменяются текстовым полем с запросом на языке SQL.

```
SELECT "Группа", "№зачетки", "ФИО", "Дата рождения" FROM "Студент" AS
"Студент" WHERE ( "Группа" = 'AK1-11' AND "Дата рождения" >= {D '1981-
11-11' } OR "Группа" = 'AK3-11' ) ORDER BY "Группа" ASC, "ФИО" ASC
```

Когда вы нажимаете кнопку «выполнить» конструктор запросов формирует такую строку, и передает ее драйверу СУБД. Драйвер передает ее на сервер СУБД, (если надо, по сети).

Результаты запроса передаются обратно через драйвер СУБД конструктору, который отображает их. Именно SQL является стандартным языком для всех СУБД. Таким образом, вы можете легко перейти на другую СУБД другого производителя. К сожалению, некоторые конструкции нечетко прописаны в стандарте, в результате существуют диалекты языка SQL. Так, что при смене СУБД обычно требуется тестирование и внесение некоторых изменений.

Но вернемся к нашему запросу. Во первых, можно заметить, что для критериев отбора, у которых не задана операция отношения, используется отношение «равно». Во вторых, условия отбора, заданные в разных колонках соединены логической операцией «AND». В третьих, условия в разных строках условия соединены логической операцией «OR». Отбираются только записи, для которых выполняется условие, заданное в секции «WHERE».

Если нужно, отобразить студентов групп 'AK1-11' и 'AK3-11' с датой рождения больше заданной, поправьте условие

```
SELECT "группа", "№зачетки", "ФИО", "Дата рождения" FROM "студент" AS
"студент" WHERE ( "Дата рождения" >= {D '1980-04-03' } AND ("группа" =
'AK1-11' OR "группа" = 'AK3-11' ) ) ORDER BY "группа" ASC, "ФИО" ASC
```

Если снова нажать на кнопку «включить / выключить вид дизайна». Конструктор попытается перевести заданное условие в табличную форму. Но бывают случаи некорректного преобразования.

Удалите все условия на группу.

Отберем теперь всех студентов с датой рождения в заданном диапазоне. Нам нужно получить условие

```
"Дата рождения">={D '1980-01-01' } AND "Дата рождения"<{D '1981-01-01' }
```

Чтобы сформировать это условие в конструкторе, добавим еще одно поле «Дата рождения» и зададим в ней второе условие.

Поле	группа	№зачетки	ФИО	Дата рождения	Дата рождения
Псевдоним					
Таблица	студент	студент	студент	студент	студент
Сортировка	по возрастанию		по возрастанию		
Видимый	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Функция					
Критерий				>= #01.01.1980#	< #01.01.1981#

Чтобы в результатах запроса не появилась вторая колонка колонку «Дата рождения» снимите отметку «Видимый» в дублирующей колонке.

Вычисление значений выражений

Удалите все условия.

К значениям полей можно применять различные функции, например, если мы хотим получить год рождения каждого студента, надо ввести вместо имени поля выражение:

YEAR("Дата рождения"). Обратите внимание, что имя поля заключается в кавычки. Не забудьте поставить отметку «Видимый»

vedomost4.odt : Запрос1 - OpenOffice.org Базовый компонент...

Файл Правка Вид Вставка Сервис Окно Справка

	Группа	№ зачетки	ФИО	Дата рождения	Год рождения
▶	AK1-11	001	Иванов Иван Иванович	10.10.1980	1980
	AK1-11	002	Петров Петр Петрович	11.11.1981	1981
	AK1-11	003	Сидоров Иван Федорович	12.12.1982	1982
	AK3-11	022	Никонов Николай Николаевич		
	AK3-11	021	Фролов Иван Иванович	03.03.1980	1980
	CM1-11	013	Киров Иван Игнатьевич	02.02.1980	1980
	CM1-11	011	Ковалев Иван Сидорович	13.10.1980	1980
	CM1-11	012	Симонов Петр Константинович	01.01.1980	1980

Запись 1 из 8

Поле	Группа	№ зачетки	ФИО	Дата рождения	YEAR("Дата рождения"
Псевдоним					Год рождения
Таблица	Студент	Студент	Студент	Студент	
Сортировка	по возрастанию		по возрастанию		
Видимый	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Функция					
Критерий					
Имя					

Чтобы в результатах запроса в заголовке нашей новой колонки выводилось не заданное выражение, а осмысленная строка, введите нужный текст в поле «псевдоним» нашей колонки. Кстати, при использовании результатов одного запроса в другом псевдонимы для вычисляемых колонок надо задавать обязательно.

Месяц можно посчитать с помощью функции `MONTH("Дата рождения")`

Можно задавать и более сложные выражения. Выведем возраст наших студентов на данный момент времени. Это можно сделать двумя способами:

`YEAR(NOW()) - YEAR("Дата рождения")` – берем год от текущей даты и вычитаем из него год от даты рождения.

`DATEDIFF('year', "Дата рождения", NOW())` – Воспользуемся специальной функцией. Эта же функция позволяет вычислять длительность интервала в днях:

`DATEDIFF('day', "Дата рождения", NOW())`

Полный список функций приведен в приложении «Встроенные функции и хранимые процедуры Hsqldb».

При формировании выпадающего списка часто требуется склеить значения из нескольких колонок таблицы. Например, если имеются однофамильцы, придется добавить к ФИО номер зачетки. Это делается с помощью оператора склеивания строк «|»:

`"ФИО" || ' / ' || "№зачетки" || ' / '`

vedomost3.odt : Запрос1 - OpenOffice.org Базовый компонент: построи...

Файл Правка Вид Вставка Сервис Окно Справка

	группа	№зачетки	ФИО	Дата рождения	Возраст	"ФИО" '/' "№зачетки" '/'
▶	AK1-11	001	Иванов Иван Иванович	10.10.1980	29	Иванов Иван Иванович/001/
	AK1-11	002	Петров Петр Петрович	11.11.1981	28	Петров Петр Петрович/002/
	AK1-11	003	Сидоров Иван Федорович	12.12.1982	27	Сидоров Иван Федорович/003/
	AK3-11	022	Никонов Николай Николаевич	04.04.1980	29	Никонов Николай Николаевич/022/
	AK3-11	021	Фролов Иван Иванович	03.03.1980	29	Фролов Иван Иванович/021/
	CM1-11	013	Киров Иван Игнатьевич	02.02.1980	29	Киров Иван Игнатьевич/013/
	CM1-11	011	Ковалев Иван Сидорович	13.10.1980	29	Ковалев Иван Сидорович/011/
	CM1-11	012	Симонов Петр Константинович	01.01.1980	29	Симонов Петр Константинович/012/

Запись 1 из 8

Поле	№зачет	ФИО	Дата рождения	YEAR(NOW()) - YEAR("Дата рождения")	"ФИО" '/' "№зачетки" '/'
Псевдоним				Возраст	
Таблица	студент	студент	студент		
Сортировка		по возр			
Видимый	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Функция					

Итак, запомните правила:

- Строковые константы заключаются в апострофы 'Иванов'
- Имена таблиц и колонок заключаются в кавычки "ФИО" Внимательно следите за пробелами и регистром символов.
- Даты заключаются между символами «#»: #11.11.1981#
- Числовые константы записываются, без каких либо знаков: 12345

Работа с NULL значениями

Откройте таблицу «студент» и сотрите у одного из студентов дату рождения. Такие значения называются «NULL значения». Сохраните запись.

Выполните наш запрос. Результатом выражения оказалась пустая строка.

Попробуйте задать условие на дату. Запись с «NULL» не удовлетворяет ни условию «<=#01.01.1980#» ни «>=#01.01.1980#», ни, как не странно «<> #01.01.1980#».

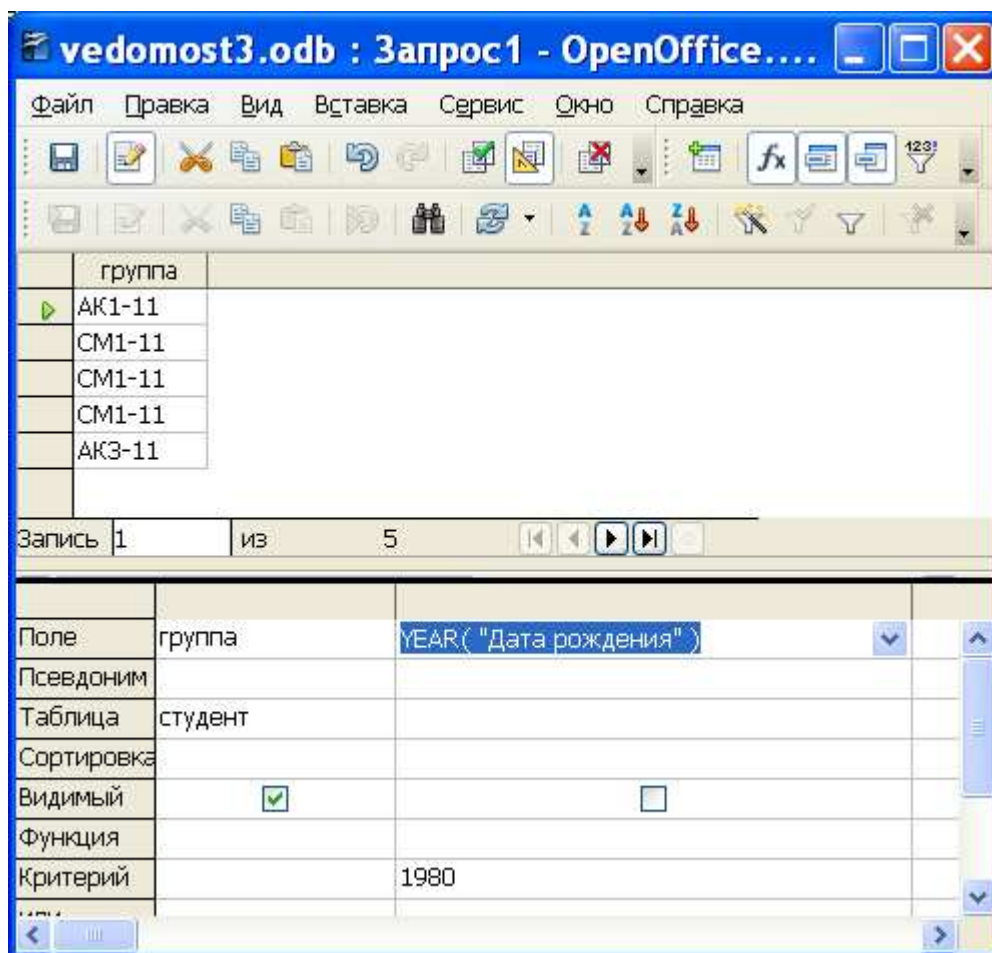
Для проверки на NULL значение используйте условия «IS EMPTY» и «NOT IS EMPTY»

Исключение повторяющихся записей

Составим список групп, в которых имеются студенты 1980 года рождения, причем, нас не интересуют ФИО студентов, только факт наличия студентов в группе.

Создайте новый запрос в режиме дизайна. Добавьте таблицу «Студент». Добавьте колонки «Группа» и «Дата рождения». Во второй колонке организуйте вычисление выражения «YEAR("Дата рождения")», задайте критерий отбора 1980, снимите отметку «Видимый».

После выполнения запроса получим:



Почему группа «СМ1-11» попала в список 3 раза? Дело в том, что в этой группе три студента 1980 года рождения. Чтобы избавиться от повторов? Нажмите на панели инструментов кнопку «Однозначные значения» (крайняя справа). Если кнопка утоплена, исключаются одинаковые строки в выборке.

Выполните запрос. Теперь повторов нет, а группы оказались отсортированы по алфавиту.

Запрос данных из связанных таблиц

Пусть нам нужно вывести ФИО преподавателей и наименования предметов по всем экзаменам заданной группы.

Особенностью этого запроса является то, что фамилия преподавателя, наименование предмета и шифр группы лежат в разных таблицах.

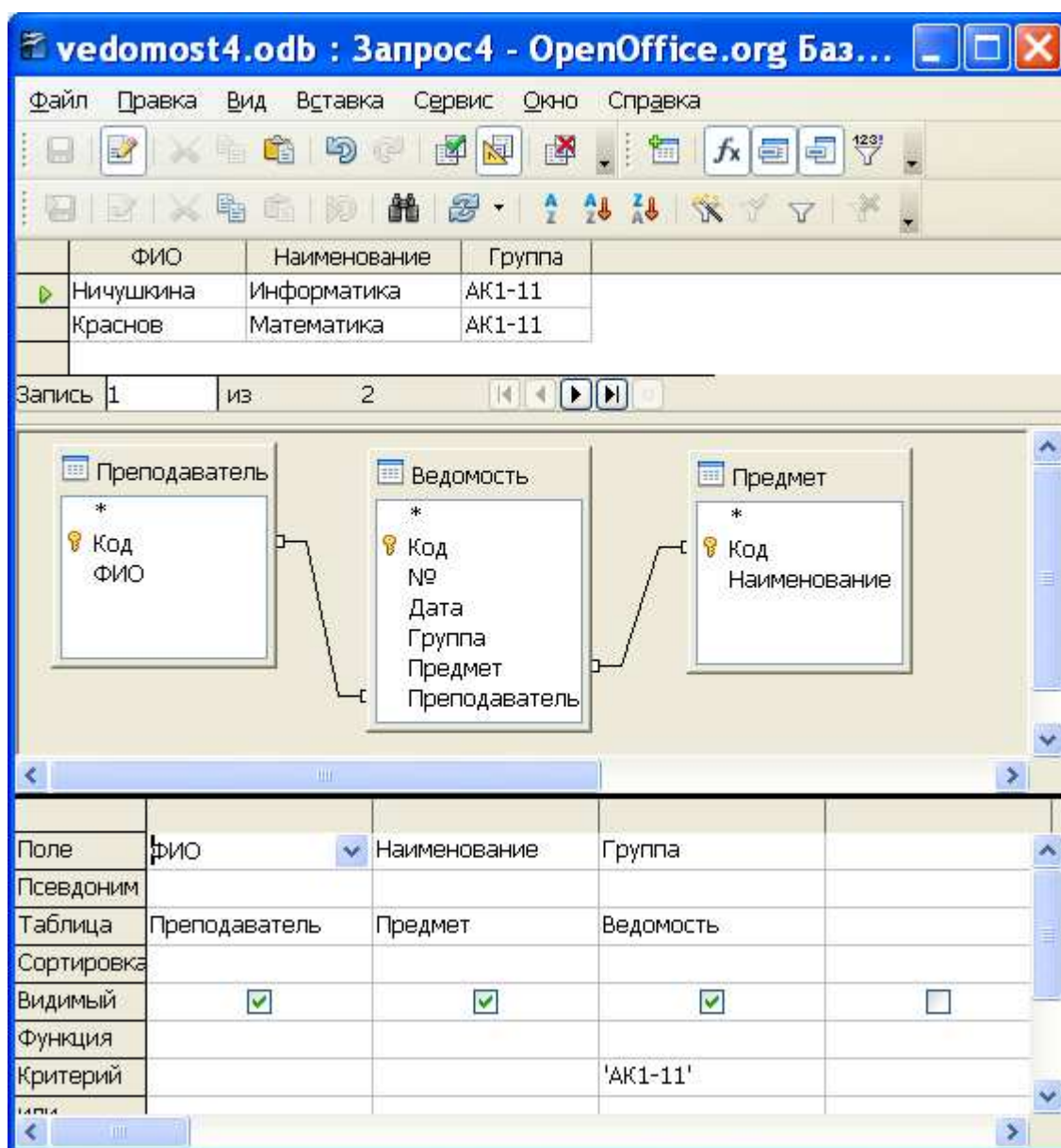
Создайте новый запрос в режиме дизайна. Добавьте таблицы «Преподаватель», «Ведомость» и «Предмет». Выбираем таблицу «Ведомость», поскольку в ней есть шифр группы, но важнее всего, что именно поля этой таблицы ссылаются на записи в таблицах предмет и преподаватель.

Обратите внимание, что конструктор запросов автоматически добавил связи между таблицами, пользуясь связями, которые мы создали в конструкторе связей. Если добавленные таблицы окажутся не связанными, значит, вы не добавили связь между таблицами, или таблицы не связаны непосредственно, тогда надо добавить таблицу, посредством которой связаны наши таблицы, хотя из этой таблицы нам и не нужно ни одного поля.

Но вернемся к нашему запросу. Добавьте колонки:

- «ФИО» из таблицы «Преподаватель»
- «Наименование» из таблицы «Предмет»
- «Группа» из таблицы «Ведомость»

Задайте шифр группы в поле «Критерий» колонки «Группа» и выполните запрос.



Не забудьте предварительно ввести данные в таблицы.

Контрольные задачи на простые запросы

1. Составить список групп для заданного факультета.
2. Определить список предметов (в алфавитном порядке), принимавшихся заданным преподавателем.
3. Вывести номера ведомостей и даты экзаменов, в весеннем семестре заданного года.
4. Создать список групп, сдававших экзамены заданному преподавателю в заданном месяце.
5. Составить список преподавателей (в алфавитном порядке), принимавших экзамены в текущем году.

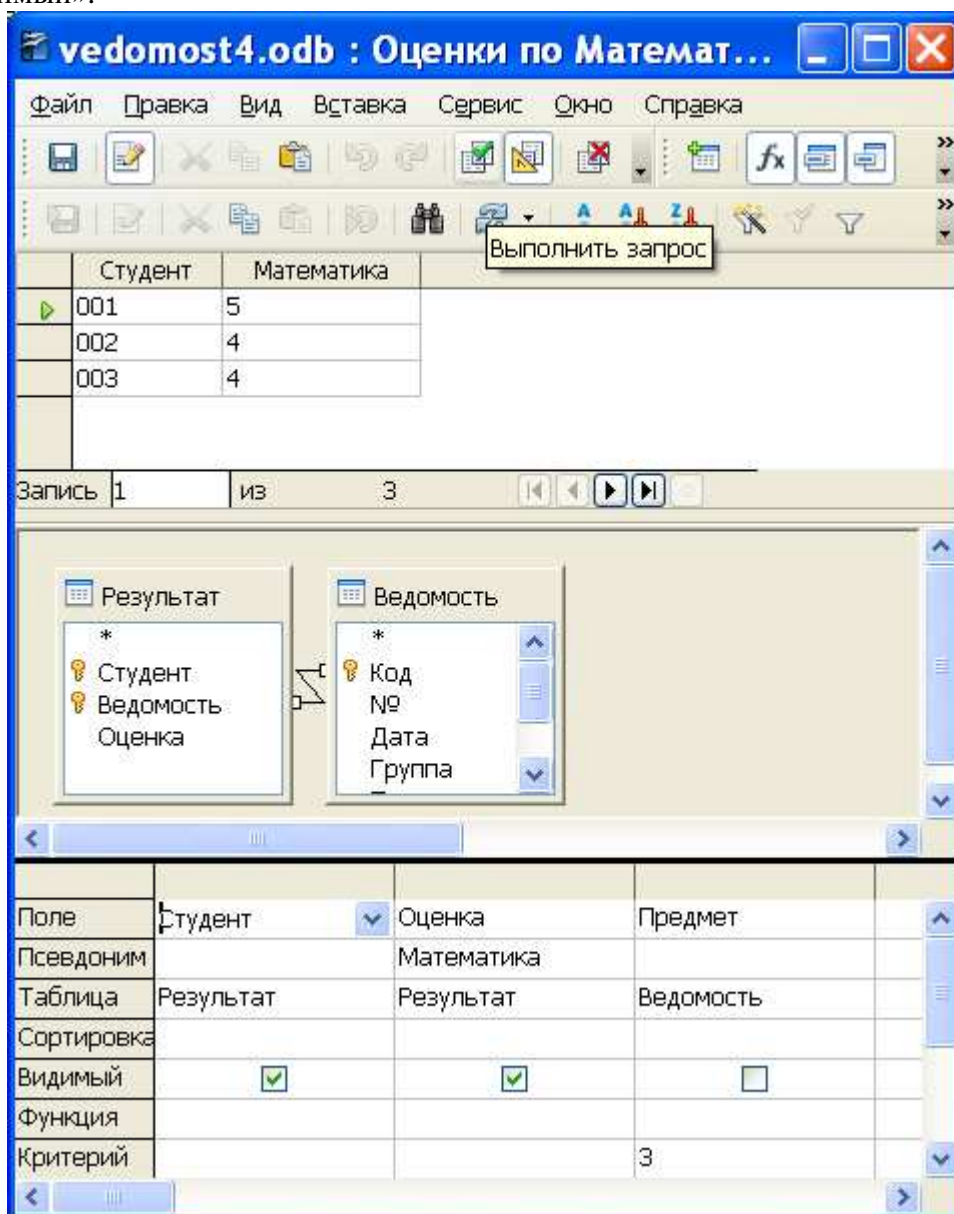
Правое и левое соединение таблиц

Пусть нам надо вывести оценки **всех** студентов по математике, но не для всех студентов есть записи в таблице результат.

К сожалению, в конструкторе «OpenOffice.org 3.0» в один запрос такую задачу решить не удалось. Подготовим вспомогательный запрос, формирующий список оценок по математике.

Создаем новый запрос в режиме дизайна. Добавляем таблицы «Результат» и «Вedomость». Добавляем колонки «Студент» и «Оценка» из таблицы «Результат». Добавляем колонку «Предмет» из таблицы «Вedomость». Колонке «Оценка» даем псевдоним «Математика». Для колонки «Предмет» задаем критерий отбора равный коду математики (Посмотрите в таблице

«Предмет»). Значения в колонке «Предмет» все одинаковые их мы и так знаем, поэтому, снимем отметку «Видимый».



Сохраним этот запрос под именем «Оценки по Математике».

Создаем новый запрос в режиме дизайна. Добавляем таблицу «Студент». Затем выбираем кнопку «Запросы» и добавляем наш вспомогательный запрос «Оценки по Математике». Часто сложный запрос удобно разбить на несколько простых вспомогательных запросов, а затем использовать их в главном запросе. Обычно, время выполнения от этого не меняется. Перед выполнением, собирается общий запрос и передается СУБД, которая его оптимизирует.

Внимание! Таблицы надо связать. Для этого надо подвести указатель мыши к полю «№ зачетки» из таблицы «Студент», нажать левую кнопку мыши и, не отпуская ее переместить указатель мыши на поле «Студент» из вспомогательного запроса, затем отпустить кнопку мыши.

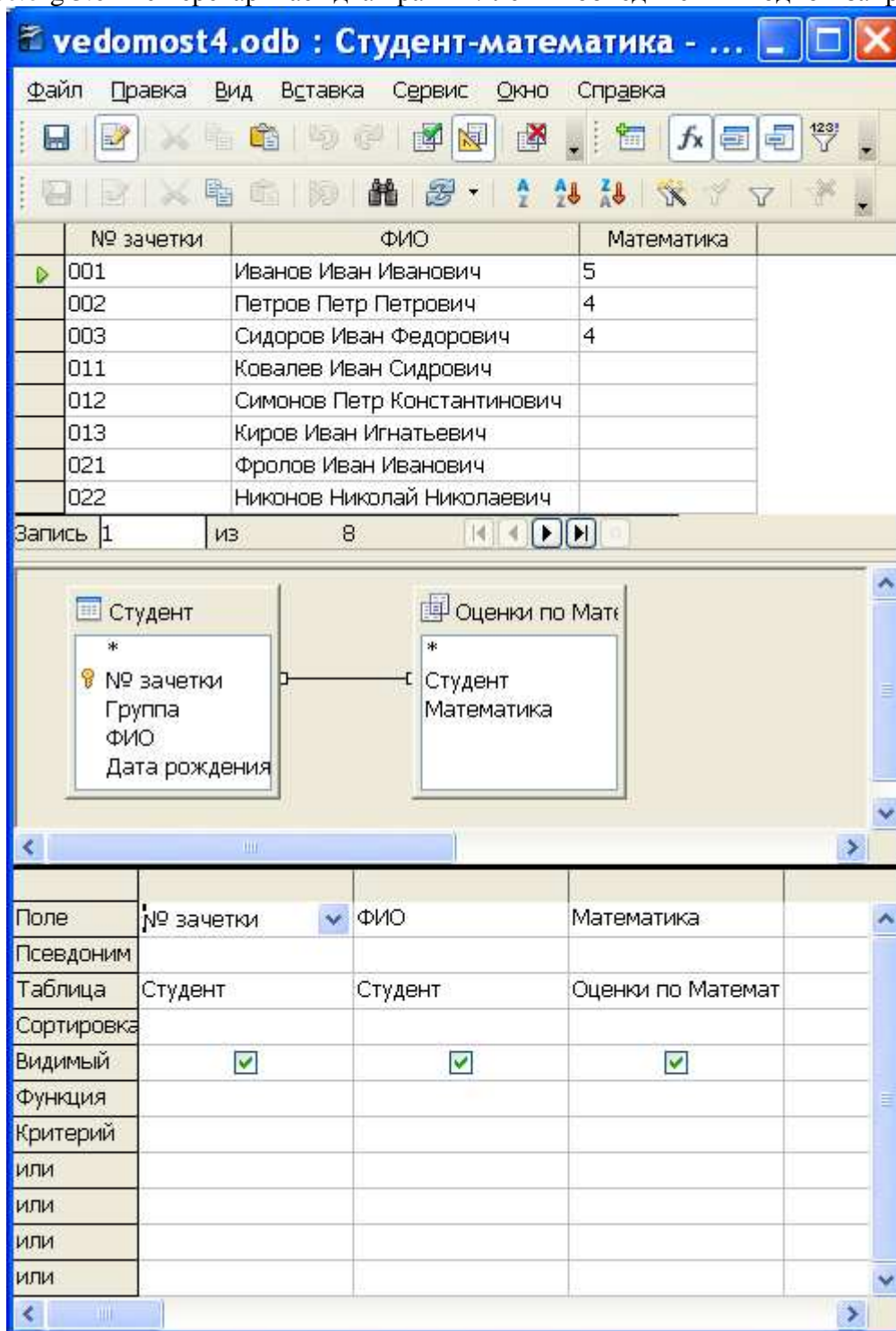
Добавляем колонки «№ зачетки» и «ФИО» из таблицы «Студент». И колонку «Информатика» из вспомогательного запроса.

Если сейчас выполнить запрос, получим фамилии студентов, сдававших математику и их оценки. Дело в том, что тип добавляемой связи по умолчанию «Внутреннее объединение». При таком соединении выводятся только записи, в котором значения в заданных полях совпадают. Чтобы получить список **всех** студентов, выполните двойной щелчок мыши на связи между таблицами и выберите из выпадающего списка «тип» «Правое объединение», нажмите «ОК».

Выполните запрос. Теперь получим полный список студентов и оценки по математике для тех студентов, которые ее сдавали. У остальных в колонке «Математика» пусто. Если не получилось, попробуйте «Левое объединение». Какое объединение выбрать, зависит не от расположения таблиц в среднем окне, а от порядка, в котором их добавляли. И еще одно

замечание, если подсказка в нижней части диалога «Свойства связи» не соответствует результату, не верь глазам своим.

Одним запросом эту задачу не удалось решить не только потому, что конструктор запросов в «OpenOffice.org 3.0» не переваривает два правых / левых объединения в одном запросе.



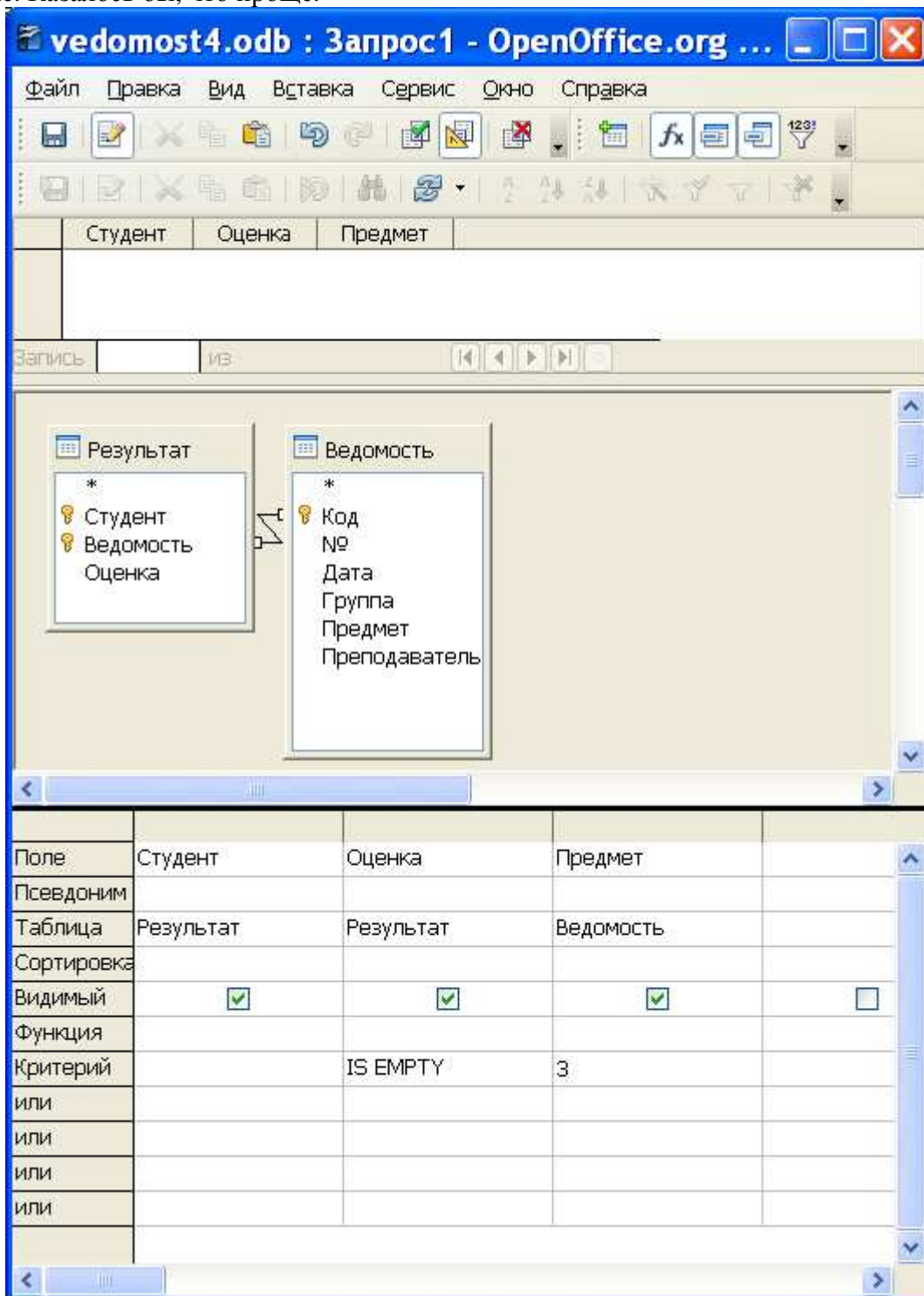
Сохраните полученный запрос под именем «студент - математика» и снимите с него копию (Ctrl+C, Ctrl+V в главном окне программы). Поэкспериментируем с ним.

Попробуйте задать условие на оценку =5 в колонке «Информатика». Любое условие в присоединенной таблице отсечет заодно и все пустые ячейки. Напоминаю что, NULL значение не удовлетворяет ни одной логической операции. Чтобы в отчете остались студенты, которые не сдавали математику, можно задать «IS EMPTY» в поле «или». Но все равно, из списка из списка пропадут студенты, получившие другие оценки. Это не ошибка программы, а результат заданного нами критерия отбора. Не нравится, смело создавайте вспомогательные запросы. Обычно СУБД их нормально оптимизирует. Кстати, иногда запрос с правым или левым объединением выполняются даже быстрее, чем запросы с внутренним объединением. Иногда СУБД выбирает неправильную последовательность выполнения запроса. Использование правого объединения

заставляет СУБД выполнять запрос в другом порядке, в результате, запрос выполняется в несколько раз быстрее.

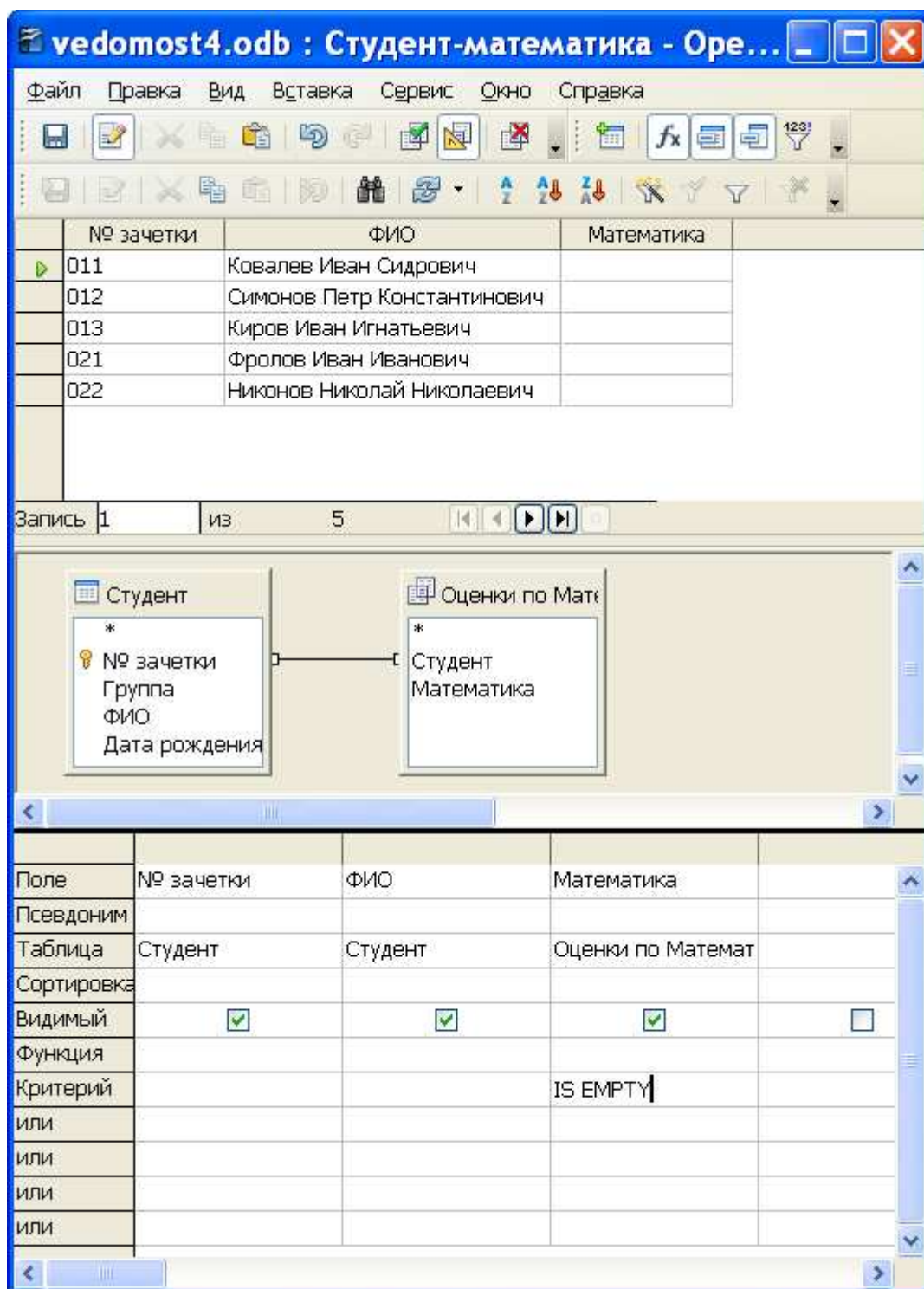
Черная кошка в темной комнате

Теперь предположим, что нам нужно найти студентов, которые **не** сдавали экзамен по математике. Казалось бы, что проще.



Но результатом такого запроса будет пустой список. Дело в том, что для студентов, не сдававших экзамен, не просто значение в поле «Оценка» пусто, а вообще нет записи. Трудно найти черную кошку в темной комнате, особенно если ее там нет.

Для решения нашей задачи надо взять полный список студентов из таблицы «Студент» и исключить из него студентов, которые имеют оценки. Берем наш запрос «студент - математика» и задаем для колонки «Математика» критерий «IS EMPTY»



Если таблицы не соединять или «все комбинации»

Наиболее частая ошибка при построении запросов – не задано соединение таблиц, в результате получаем все комбинации записей заданных таблиц или вспомогательных запросов. Если в двух таблицах всего по 100 записей получим множество из 10000 записей. Часто не удастся дождаться окончания выполнения такого запроса. Сохраняйте сложные запросы и всю базу перед выполнением запроса.

Однако это неприятное свойство запроса комбинировать записи можно использовать.

Предположим, у нас имеется пачка подотчетных бланков с последовательными номерами от 1111001 до 111500. В базе данных хранятся записи с номерами выданных бланков. Надо найти номера бланков, которые должны были остаться в пачке. Задача сложнее предыдущей, ведь нет таблицы с номерами всех бланков.

Для решения этой задачи создадим вспомогательную таблицу «Цифры» с единственным полем «N» типа «Integer» и занесем в эту таблицу всего 10 чисел от 0 до 9. Создадим запрос и

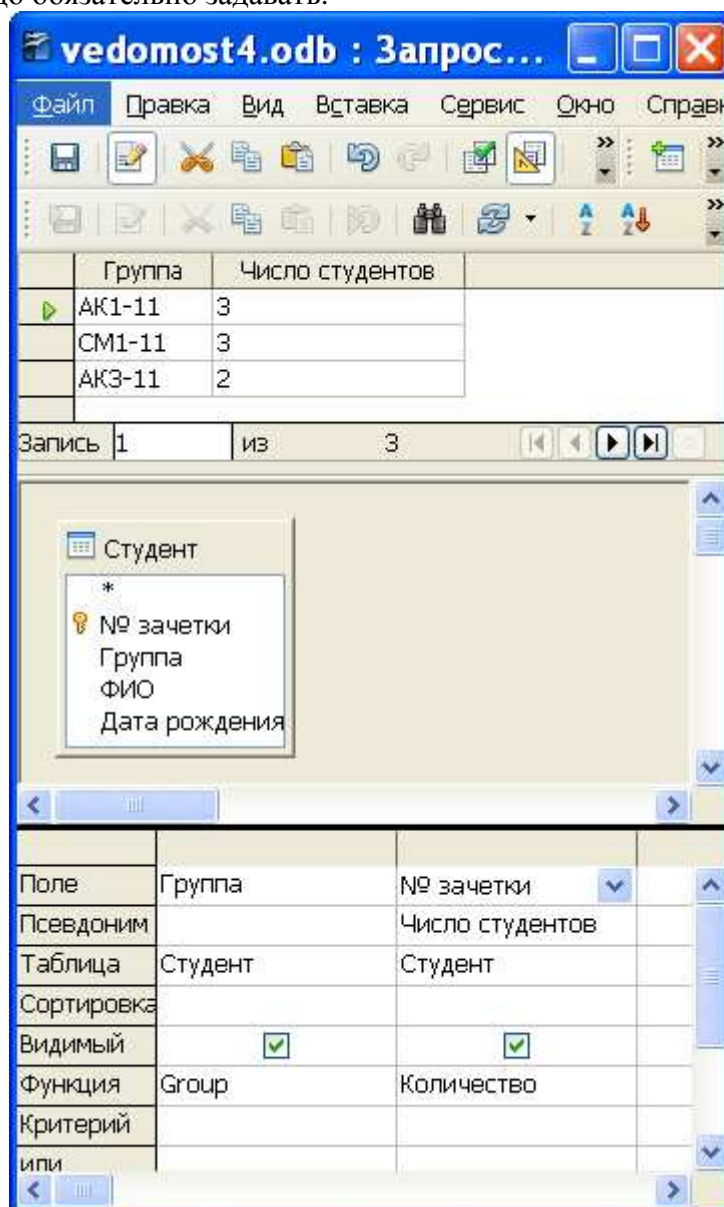
добавим одну и ту же таблицу «Цифры» три раза. Конструктор назначает таблицам псевдонимы «Цифры», «Цифры_1», «Цифры_2». Поскольку, связи между таблицами не заданы, результат будет формироваться на основе всех возможных комбинаций записей трех таблиц ($10 \times 10 \times 10 = 1000$ комбинаций). Единственную колонку запроса формируем на основе выражения:
`"Цифры"."N" * 100 + "Цифры_1"."N" * 10 + "Цифры_2"."N"`

Результатом запроса будет множество чисел от 0 до 999. Чтобы получить числа в диапазоне от 1111001 до 111500 добавим в выражение + 1111001 и зададим критерий отбора <111500

Чтобы получить номера неиспользованных бланков, соединяем правым соединением полученный подзапрос и данные об использованных бланках. Задаем в критерии отбора на 2 колонку IS EMPTY и получаем номера не использованных бланков.

Статистические функции

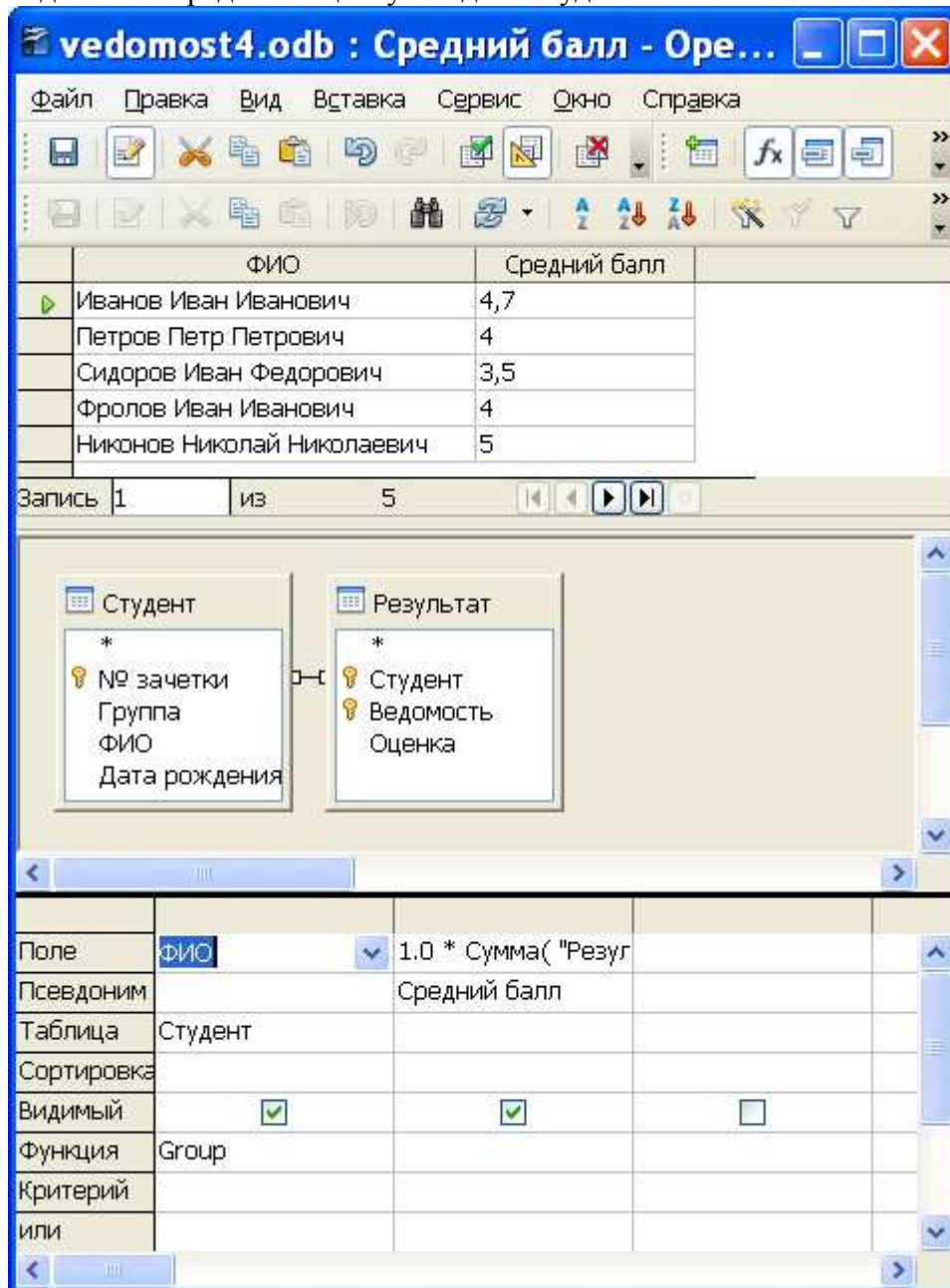
Для начала подсчитаем число студентов в каждой группе. Для этого делаем запрос с двумя колонками: «Группа» и «№ зачетки». В ячейке «Функция» для колонки «Группа» выбираем из списка «Group». Для колонки «№ зачетки» выбираем «Количество». Для улучшения внешнего вида результата запроса задаем псевдоним второй колонки «Число студентов». Кстати, чтобы этот запрос можно было использовать, в других запросах, как подзапрос, псевдонимы всех вычисляемых колонок надо обязательно задавать.



Запрос сортирует записи по полю «Группа» и превращает каждую группу записей с одинаковым значением шифра группы в одну запись, при этом подсчитывает количество не пустых значений в колонке «№ зачетки». Внимание, не любую колонку можно использовать при подсчете количества. Если бы для подсчета количества студентов мы использовали не номер

зачетки, а дату рождения, которая задана не у всех, мы получили бы меньший результат. Лучше всего использовать ключевое поле.

Если ни для одного поля не задана функция группировки результатом будет единственная строка. Так, если удалить колонку «Группа», запрос подсчитает общее число студентов в таблице. Теперь подсчитаем среднюю оценку каждого студента.



При использовании статистической функции хотя бы в одной колонке вы обязаны задать статистическую функцию для каждой колонки. Если мы хотим вывести для справки номер зачетки, мы должны использовать статистическую функцию, например, «Максимум».

Примечание: Функция AVG в русифицированной версии не работает, но нетрудно использовать формулу «1.0 * Сумма("Результат"."Оценка") / Количество("Результат"."Оценка")»

Обратите внимание, на множитель 1.0, он нужен, чтобы расчеты производились не с целыми, а с вещественными числами. Иначе, вместо 4.7 получим 4.

Запросы непосредственно на SQL

Не всегда для решения задач хватает возможностей конструктора запросов. В этом случае следует нажать кнопку «включить / выключить вид дизайна» и больше, в режим конструктора запросов не переключаться. Конструктор перестроит ваш запрос, как ему угодно. Открывать такой запрос следует командой «Редактировать в режиме SQL». Кроме того, SQL запросы, являются основным способом обращения к базе данных из программ на языках «С», «Pascal», «Java»...

Синтаксис запроса на выборку данных в общем виде следующий:

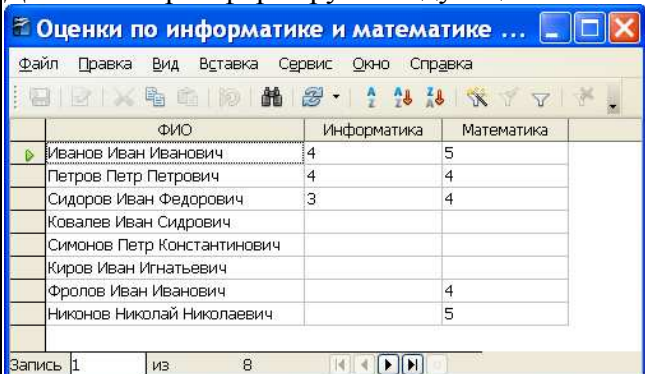
```

SELECT {selectExpression | * } [, ...]
FROM tableList
[WHERE Expression]
[GROUP BY Expression [, ...]]
[HAVING Expression]
[{ UNION [ALL | DISTINCT] | {MINUS [DISTINCT] | EXCEPT [DISTINCT] } |
INTERSECT [DISTINCT] } selectStatement]
[ORDER BY orderExpression [, ...]]
[LIMIT <limit> [OFFSET <offset>]];

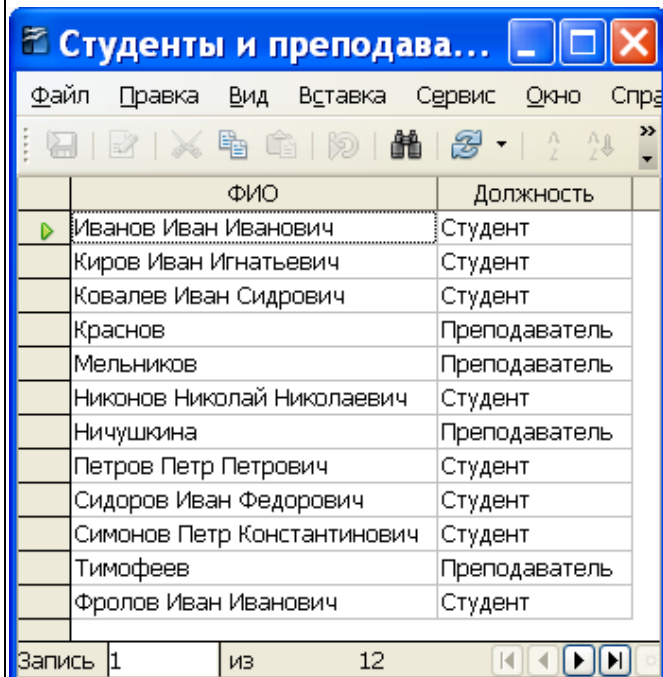
```

Рассмотрим основные формы запросов:

Запрос	Пояснения
SELECT * FROM "Студент"	Символ «*» означает, что выдаются все поля таблицы «Студент»
SELECT "Группа", "ФИО", "Дата рождения" FROM "Студент"	Выбираются только перечисленные поля
SELECT "Группа", "ФИО", "№ зачетки", "Дата рождения" FROM "Студент" ORDER BY "Группа", "ФИО"	Выбираемые записи сортируются сначала по шифру группы, затем по ФИО (по возрастанию).
SELECT "Группа", "ФИО", "№ зачетки", "Дата рождения" FROM "Студент" ORDER BY "Дата рождения" DESC	Для сортировки по убыванию следует после имени колонки поставить «DESC»
SELECT "Группа", "ФИО", "№ зачетки", "Дата рождения", YEAR(NOW()) - YEAR("Дата рождения") AS "Возраст" FROM "Студент" ORDER BY "Возраст"	Кроме колонок можно задавать выражения. При использовании результатов одного запроса в другом псевдонимы для вычисляемых колонок надо задавать обязательно. Псевдонимы можно использовать при сортировке и в условиях отбора.
SELECT "Группа", "ФИО", "№ зачетки", "Дата рождения", YEAR("Дата рождения") as "Год рождения", YEAR(NOW()) - YEAR("Дата рождения") AS "Возраст" FROM "Студент"	К сожалению, HSQL не позволяет использовать псевдонимы в выражениях вычисляемых колонок. Приходится повторять выражения.
SELECT "Группа", "ФИО", "№ зачетки", "Дата рождения", YEAR(NOW()) - YEAR("Дата рождения") AS "Возраст" FROM "Студент" WHERE "Возраст">20 AND "Группа"='AK1-11' ORDER BY "Возраст"	Условие отбора записей указывается после ключевого слова WHERE. Выбираются только записи, для которых результатом заданного логического выражения является TRUE.
SELECT "Группа", "ФИО", "№ зачетки", P."Наименование" AS "Предмет" FROM "Студент" AS S, "Результат" AS R, "Ведомость" AS V, "Предмет" AS P WHERE R."Студент" = S."№ зачетки" AND R."Ведомость" = V."Код" AND V."Предмет" =P."Код"	Чтобы выбрать записи из нескольких таблиц, надо перечислить их через запятую после «FROM». Чтобы меньше писать, используйте псевдонимы таблиц. Удобнее всего латинские буквы в верхнем регистре (их можно использовать без кавычек). Но учтите, что если задали псевдоним таблицы, использовать имя таблицы в выражениях уже нельзя. Не забывайте задавать связи между таблицами, иначе получите все комбинации записей. В данном примере связи задаются условиями: R."Студент" = S."№ зачетки" R."Ведомость" = V."Код" V."Предмет" =P."Код"
SELECT "Группа", "ФИО", AVG("Оценка"+0.0) FROM "Студент" AS "S", "Результат" AS "R" WHERE "R"."Студент" = "S"."№ зачетки" GROUP BY "Группа", "ФИО"	При вычислении статистических функций, колонки, по которым осуществляется группировка, перечисляются через запятую после ключевых слов «GROUP BY»

SELECT min ("Группа"), min ("ФИО"), AVG("Оценка"+0.0) FROM "Студент" AS "S", "Результат" AS "R" WHERE "R"."Студент" = "S"."№ зачетки" GROUP BY "№ зачетки"	Все колонки, перечисленные в SELECT, должны быть либо перечислены после «GROUP BY», либо быть статистическими функциями. В нашем примере каждый студент, связан только с одной группой, да и группировать правильнее по номеру зачетки. Чтобы нам вывести в этом случае шифр группы воспользуемся, например, функцией min()
SELECT min("Группа"), min("ФИО"), AVG("Оценка"+0.0) AS "Средний балл" FROM "Студент" AS "S", "Результат" AS "R" WHERE "R"."Студент" = "S"."№ зачетки" GROUP BY "№ зачетки" HAVING AVG("Оценка"+0.0)>4.0	Чтобы отобразить записи по итогам подсчета статистических функций, надо задать условие после ключевого слова «HAVING». К сожалению, в HSQL и здесь псевдонимом воспользоваться не удастся. Приходится повторять выражения.
SELECT "Группа", "ФИО", AVG("Оценка"+0.0) FROM "Студент" AS "S", "Результат" AS "R" WHERE "Группа"='AK1-11' AND "R"."Студент" = "S"."№ зачетки" GROUP BY "Группа", "ФИО" HAVING AVG("Оценка"+0.0)>4.0	Не следует ставить после «HAVING» условия отбора, которые можно задать после WHERE. Зачем загружать в память данные, и выполнять над ними статистические функции, если позже они будут все равно отброшены.
SELECT MAX("Средний балл") AS M FROM (SELECT AVG("Оценка" + 0.0) AS "Средний балл" FROM "Результат" GROUP BY "Студент")	Вместо таблицы можно указать в скобках вспомогательный запрос. Приведенный пример возвращает максимальный средний балл из средних баллов студентов.
SELECT "Группа", "ФИО", AVG("Оценка"+0.0) FROM "Студент" AS "S", "Результат" AS "R" WHERE "Группа"='AK1-11' AND "R"."Студент" = "S"."№ зачетки" GROUP BY "Группа", "ФИО" HAVING AVG("Оценка"+0.0) > (SELECT AVG("Оценка" + 0.0) AS "Средний балл" FROM "Результат")	Запрос, возвращающий единственное значение можно использовать в выражениях. Приведенный пример отбирает студентов со средним баллом выше среднего.
SELECT S."ФИО", R0."Информатика", R3."Математика" FROM "Студент" AS S LEFT OUTER JOIN (SELECT "Студент", "Оценка" AS "Информатика" FROM "Результат" AS "R", "Ведомость" AS "V" WHERE "R"."Ведомость"="V"."Код" AND "Предмет"=0) AS R0 ON R0."Студент" = "S"."№ зачетки" LEFT OUTER JOIN (SELECT "Студент", "Оценка" AS "Математика" FROM "Результат" AS "R", "Ведомость" AS "V" WHERE "R"."Ведомость"="V"."Код" AND "Предмет"=3) AS R3 ON R3."Студент" = "S"."№ зачетки"	Чтобы вывести ВСЕХ студентов и их оценки (если они есть) воспользуемся левым соединением таблицы «Студент» и подзапроса. « LEFT » означает, что выбираются все записи из таблицы или подзапроса заданного слева от данного ключевого слова. Условие соединения задаются после ключевого слова « ON » Данный запрос формирует следующий отчет 


```
SELECT "ФИО", 'Студент' AS "Должность"
FROM "Студент"
UNION ALL
SELECT "ФИО", 'Преподаватель' AS "Должность"
FROM "Преподаватель"
ORDER BY "ФИО"
```



ФИО	Должность
Иванов Иван Иванович	Студент
Киров Иван Игнатьевич	Студент
Ковалев Иван Сидрович	Студент
Краснов	Преподаватель
Мельников	Преподаватель
Никонов Николай Николаевич	Студент
Ничушкина	Преподаватель
Петров Петр Петрович	Студент
Сидоров Иван Федорович	Студент
Симонов Петр Константинович	Студент
Тимофеев	Преподаватель
Фролов Иван Иванович	Студент

Бывают случаи, когда надо объединить множества записей из нескольких таблиц или запросов. В этом случае следует воспользоваться конструкцией «**UNION ALL**». Не все СУБД поддерживают данную конструкцию, поэтому включим режим «Выполнить непосредственно команду SQL» (правая кнопка на панели инструментов). После последнего запроса можно поставить «**ORDER BY**» и перечислить поля, по которым будут сортироваться записи объединенной выборки. Чтобы исключить повторяющиеся записи следует использовать вместо «**UNION ALL**» просто «**UNION**». Аналогично можно организовать вычитание множеств записей (**MINUS** или **EXCEPT**) и пересечение множеств записей (**INTERSECT**).

```
SELECT S."ФИО" FROM
"Студент" AS S
LEFT OUTER JOIN
(SELECT "Студент", "Оценка" AS
"Информатика" FROM "Результат" AS "R",
"Ведомость" AS "V" WHERE
"R"."Ведомость"="V"."Код" AND "Предмет"=0)
AS R0
ON R0."Студент" = "S"."№ зачетки"
WHERE R0."Информатика" IS NULL
```

```
SELECT "№ зачетки" AS N FROM "Студент"
EXCEPT
SELECT "Студент" AS N FROM "Результат" AS
"R", "Ведомость" AS "V" WHERE
"R"."Ведомость"="V"."Код" AND "Предмет"=0
```

Список студентов, которые не сдавали экзамен по информатике можно получить двумя способами:

1. Используя левое соединение вывести список всех студентов и их оценки, и отобрать из них записи с NULL значением оценки.
2. Используя конструкцию «**EXCEPT**» исключить из множества всех студентов множество студентов, сдававших информатику (не забудьте включить режим «Выполнить непосредственно SQL»).

При выборе способа учитывайте, что «**EXCEPT**» поддерживают еще меньше СУБД, чем «**UNION**».

```
SELECT TOP 5
"ФИО", AVG("Оценка" + 0.0) AS "Средний балл"
FROM "Студент" AS S, "Результат" AS R
WHERE R."Студент" = "S"."№ зачетки"
GROUP BY "ФИО"
ORDER BY "Средний балл" DESC
```

Еще один запрос в режиме «Выполнить непосредственно команду SQL». Чтобы отобрать 5 лучших студентов, сортируем полный список студентов в порядке убывания среднего балла и отбираем 5 первых записей.

SELECT LIMIT 20 5 "Группа", "ФИО", "№ зачетки", "Дата рождения" FROM "Студент" ORDER BY "Группа", "ФИО"	При выводе на WEB страницу часто используется постраничный вывод найденных записей. В этом случае используйте конструкцию «LIMIT». Обязательно используйте сортировку! СУБД возвращает несортированные записи в разном порядке от запроса к запросу. Приведенный пример возвращает 5 записей, начиная с 20 записи полного списка (считая с 0). Этот запрос тоже выполняется в режиме «Выполнить непосредственно команду SQL».
---	--

Контрольные задачи на запросы на SQL

- Составить список групп для факультета «АК», в которых более 20 студентов.
- Определить список предметов, принимавшихся преподавателем Ивановым М.М.
- Определить средний возраст студентов по факультетам.
- Вывести средние баллы по группам в виде таблицы с колонками «Группа», «Информатика», «Математика»

Запросы на изменение данных

ВНИМАНИЕ конструктор запросов не годится для запросов на изменение базы данных.

Из меню «Сервис» главного окна выберите команду «SQL...». Открывается окно «Выполнить инструкцию SQL». В поле «Выполняемая команда» Введите команду и нажмите кнопку «Выполнить». Подтверждение выполнения команды выводится в поле «Состояние».

К сожалению, окно «Выполнить инструкцию SQL» модальное, то есть, пока его не закроете, доступа к главному окну программы не получите. А значит, не имя поля посмотреть ни в изменении таблицы убедиться. Рекомендую писать запросы в любом текстовом редакторе, затем открыть окно «Выполнить инструкцию SQL», скопировать в него (через буфер обмена) инструкцию и выполнить ее. Затем закрыть окно. Можно так же сделать собственную форму и связать с ней простенькую программу для выполнения инструкций SQL (смотри раздел «Универсальная форма для запросов на изменение данных» главы «Выполнение SQL запросов из макросов»).

Рассмотрим наиболее часто употребляемые запросы на изменение.

Запрос **SELECT ... INTO**

```
SELECT {selectExpression | * } [, ...]  
INTO [CACHED | TEMP | TEXT][2] newTable  
FROM tableList  
[WHERE Expression]  
[GROUP BY Expression [, ...]]  
[HAVING Expression]  
[{ UNION [ALL | DISTINCT] | {MINUS [DISTINCT] | EXCEPT [DISTINCT] } |  
INTERSECT [DISTINCT] } selectStatement]  
[ORDER BY orderExpression [, ...]]  
[LIMIT <limit> [OFFSET <offset>]];
```

Прежде чем делать серьезные изменения в таблице рекомендуется сделать копию таблицы с помощью команды типа **SELECT * INTO T1 FROM "Студент"**;

Эта команда создает новую таблицу на основе заданной выборки. Разумеется, можно использовать все рассмотренные выше возможности, которые дает команда **SELECT**. Поля новой таблицы будут того же типа, что и в исходной, первичный ключ, индексы и связи в новой таблице не создаются.

Еще один прием – скопировать часть таблицы, выполнить в копии несколько последовательных изменений, а затем перенести их в основную таблицу. Часто это проще и быстрее, чем писать и выполнять сложный запрос. А некоторые СУБД, вообще не позволяют изменять таблицу, участвующую в подзапросе отбора изменяемых записей.

Имя для временной таблицы рекомендуется давать короткое и состоящее только из заглавных латинских букв и цифр. Тогда в запросах на SQL его не придется заключать в кавычки.

Запрос **INSERT INTO**

```
INSERT INTO table [( column [,...])]  
{ VALUES(Expression [,...]) | SelectStatement};
```

Первая форма этого запроса позволяет добавить в существующую таблицу одну запись:

```
INSERT INTO T1 ("№ зачетки", "Группа", "ФИО", "Дата рождения") VALUES  
( '12345', 'AK1-11', 'Иванов Иван Иванович', '1980-10-10');
```

Поля таблицы не указанные в запросе заполняются NULL или значением по умолчанию (если оно было задано в свойствах поля).

Поле со свойством «автозначение» заполняется очередным значением счетчика. Прочие ключевые и обязательные поля должны быть указаны, иначе выдается сообщение, а запись не добавляется.

Вторая форма запроса «**INSERT INTO**» позволяет добавить в существующую таблицу группу записей, сформированных с помощью «**SELECT**». Например,

```
INSERT INTO "Студент" ("№ зачетки", "Группа", "ФИО") select "№ зачетки",  
"Группа", "ФИО" from T1;
```

Добавляет в таблицу «Студент» записи, сформированные на основе записей из таблицы T1.

Проследите, чтобы значения, уже имеющиеся в ключевых полях таблицы не встречались в добавляемых записях, иначе ни одна запись добавлена не будет.

Следующий запрос, добавит только записи, которых нет в таблице «Студент»:

```
INSERT INTO "Студент" ("№ зачетки", "Группа", "ФИО")
SELECT T."№ зачетки", T."Группа", T."ФИО"
FROM "T1" AS T
LEFT OUTER JOIN "Студент" AS S ON S."№ зачетки"=T."№ зачетки"
WHERE S."№ зачетки" IS NULL;
```

Здесь запрос «SELECT» соединяет таблицы «Студент» и «T1» по полю «№ зачетки» левым соединением. При таком соединении, у записей с «№ зачетки», которых нет в таблице «Студент», «S."№ зачетки"» будет равно NULL. Затем, условие «S."№ зачетки" IS NULL» отбирает такие записи.

Обычно, сначала в конструкторе запросов создается запрос на выборку данных, проверяются записи, которые он возвращает, после чего нажатием кнопки «Вкл/выкл вид дизайна» переходим в режим SQL, копируем SQL выражение, в окно «Выполнить инструкцию SQL», добавляем «INSERT INTO ...» и выполняем полученный запрос.

Запрос DELETE

```
DELETE FROM table [WHERE Expression];
```

Удаляет записи из заданной таблицы, отобранные согласно заданному условию.

Например,

```
DELETE FROM T1 WHERE "Группа"='AK1-11';
```

Удаляет из таблицы «T1» студентов группы «AK1-11».

Если условие не задать, удаляются все записи.

Если записи из других таблиц ссылаются, на удаляемые записи эти записи так же будут удалены (если задана связь с каскадным удалением) или будет выдана ошибка.

Следующий запрос, удалит из таблицы «Студент» только записи, на которые нет ссылок из таблицы «Результат»:

```
DELETE FROM T1
where "№ зачетки" NOT IN (SELECT R."Студент" AS N FROM "Результат" AS R);
```

Удаляются записи с «№ зачетки», которых нет в выборке, которую возвращает SELECT

Этот запрос можно записать иначе:

```
DELETE FROM T1
where NOT EXISTS (SELECT R."Студент" FROM "Результат" AS R WHERE
R."Студент"="№ зачетки");
```

Удаляются записи для которых SELECT возвращает пустую выборку.

Обратите внимание, на использование псевдонимов таблиц и колонок. Лучше перестраховаться, чем удалить не те записи.

Запрос UPDATE

```
UPDATE table SET column = Expression [, ...] [WHERE Expression];
```

Позволяет изменить одно или несколько полей в записях, отобранных согласно заданному условию. Например,

```
UPDATE T1 SET "Дата рождения"='1980-10-10' WHERE "Группа"='AK1-11';
```

Заносит в поле «Дата рождения» значение «1980-10-10» для студентов группы «AK1-11».

```
UPDATE T1 SET "Группа"='AK1-11', "ФИО"='Тест' WHERE "№ зачетки"='12345';
```

Заносит в поля «Группа» и «ФИО» значения «AK1-11» и «Тест» для студента с номером зачетки «12345».

Имеется возможность использовать выражения на основе полей изменяемой записи.

Например,

```
UPDATE T1 SET "ФИО"="ФИО" || '*' || "Группа" WHERE "№ зачетки"='12345';
```

Добавляет к ФИО студента с заданным номером зачетки символ «*» и шифр группы.

```
UPDATE "Преподаватель" SET "ФИО"=UPPER(SUBSTR("ФИО",1,1)) ||
LOWER(SUBSTR("ФИО",2));
```

Объединяет первый символ фамилии преподавателя в верхнем регистре с остальной частью ФИО в нижнем регистре и заносит полученную строку в поле «ФИО».

При необходимости, можно использовать значения, возвращаемые SQL запросом.

```
UPDATE T1 SET "ФИО"=(SELECT MAX(S."ФИО") AS M FROM T1 AS S) WHERE "№
зачетки"='12345' ;
```

Заносит в поле "ФИО" записи с "№ зачетки"='12345' максимальное ФИО из таблицы «Т1»
Внимание, запрос должен возвращать единственное значение.

Запрос DROP TABLE

```
DROP TABLE <table> [IF EXISTS] [RESTRICT | CASCADE];
```

Удаляет заданную таблицу. Например,
DROP TABLE T1

Запрос CREATE TABLE

```
CREATE [MEMORY | CACHED | [GLOBAL] TEMPORARY | TEMP [2] | TEXT[2]] TABLE <name>
( <columnDefinition> [, ...] [, <constraintDefinition>...] )
[ON COMMIT {DELETE | PRESERVE} ROWS];
```

Здесь columnDefinition:

```
columnname Datatype [(columnSize[,precision])]
[{DEFAULT <defaultValue> |
GENERATED BY DEFAULT AS IDENTITY
(START WITH <n>[, INCREMENT BY <m>])}] |
[[NOT] NULL] [IDENTITY] [PRIMARY KEY]
```

constraintDefinition:

```
[CONSTRAINT <name>]
UNIQUE ( <column> [,<column>...] ) |
PRIMARY KEY ( <column> [,<column>...] ) |
FOREIGN KEY ( <column> [,<column>...] )
REFERENCES <refTable> ( <column> [,<column>...])
[ON {DELETE | UPDATE}
{CASCADE | SET DEFAULT | SET NULL}][2] |
CHECK(<search condition>)[2]
```

Создает новую таблицу. Например,

```
CREATE TABLE XX ("Код" INTEGER PRIMARY KEY, "Name" VARCHAR(30),
"Преподаватель" INTEGER)
```

Создает таблицу с именем XX и колонками:

- «Код» – Целое, первичный ключ
- «Name» – Строка длиной до 30 символов
- «Преподаватель» – Целое

Типы данных можно посмотреть в конструкторе таблиц.

Позже можно будет добавить или удалить колонки и связи, используя запрос «ALTER TABLE».

Запрос ALTER TABLE

```
ALTER TABLE <tablename> ADD [COLUMN] <columnname> Datatype
[(columnSize[,precision])] [{DEFAULT <defaultValue> |
GENERATED BY DEFAULT AS IDENTITY (START WITH <n>[, INCREMENT BY <m>])}] |
[[NOT] NULL] [IDENTITY] [PRIMARY KEY]
[BEFORE <existingcolumn>];
```

Позволяет изменить существующую таблицу. Например,

```
ALTER TABLE XX ADD COLUMN "Дата регистрации" DATE DEFAULT NOW;
```

Добавляет в таблицу «XX» Колонку «Дата регистрации» типа «Дата». По умолчанию, в это поле будет заноситься текущая дата. Не забудьте дать команду обновить таблицы в главном окне.

```
ALTER TABLE XX DROP COLUMN "Дата регистрации";
```

Удаляет из таблицы «XX» Колонку «Дата регистрации»

```
ALTER TABLE <tablename> ALTER COLUMN <columnname> RENAME TO <newname>
ALTER TABLE <tablename> ALTER COLUMN <columnname> SET DEFAULT <defaultvalue>;
```

Изменяют свойства поля (колонки) таблицы

```
ALTER TABLE <tablename> ADD [CONSTRAINT <constraintname>]
PRIMARY KEY (<column list>);
```

Позволяет создать первичный ключ. Например,

```
ALTER TABLE XX ADD PRIMARY KEY ("Код", "Дата регистрации");
```

Создает первичный ключ на пару колонок "Код" и "Дата регистрации"

```
ALTER TABLE <tablename>
  ADD [CONSTRAINT <constraintname>] FOREIGN KEY (<column list>)
  REFERENCES <exptablename> (<column list>)
  [ON {DELETE | UPDATE} {CASCADE | SET DEFAULT | SET NULL}];
```

Позволяет создать связь. Например,

```
ALTER TABLE XX ADD CONSTRAINT "XX_Преподаватель_FK" FOREIGN KEY
("Преподаватель") REFERENCES "Преподаватель"("Код") ON DELETE CASCADE;
```

Создает связь с именем «XX_Преподаватель_FK» между полем «Преподаватель» и полем «Код» из таблицы «Преподаватель». Слова «ON DELETE CASCADE» означают, что при удалении преподавателя будут удаляться ссылающиеся на него записи из таблицы «XX». Имя связи может потребоваться, для последующего удаления связи.

```
ALTER TABLE XX DROP CONSTRAINT "XX_Преподаватель_FK";
```

Удаляет зависимость с именем «XX_Преподаватель_FK».

Запрос CREATE VIEW

```
CREATE VIEW <viewname>[(<viewcolumn>,...) AS SELECT ... FROM ... [WHERE Expression]
[ORDER BY orderExpression [, ...]]
[LIMIT <limit> [OFFSET <offset>]];
```

Создает представление. Представление можно использовать во всех запросах вместо таблиц. Например,

```
CREATE VIEW "Средний балл" AS SELECT "Студент"."№ зачетки", 1.0 *
SUM("Результат"."Оценка" ) / COUNT( "Результат"."Оценка" ) AS "Средний балл",
MAX("Студент"."ФИО") AS "ФИО" FROM "Результат" AS "Результат", "Студент" AS
"Студент" WHERE "Результат"."Студент" = "Студент"."№ зачетки" GROUP BY
"Студент"."№ зачетки"
```

Создает представление с именем «Средний балл» с колонками «№ зачетки», «Средний балл», «ФИО» на основе всех оценок студентов по всем предметам. Не забудьте дать команду обновить таблицы в главном окне.

Такой запрос обычно используется в составе пакета запросов. Обычно запрос создается в конструкторе запросов. А затем сформированный им SQL подставляется в выражение CREATE VIEW. Можно так же использовать команду «создать, как представление» из контекстного меню для любого запроса в главном окне.

Запрос DROP VIEW

Удаляет представление с заданным именем. Например,

```
DROP VIEW "Средний балл"
```

Пакеты запросов на изменение

В любую базу данных время от времени требуется вносить изменения. Растут аппетиты пользователей, приходит новый начальник и требует новых отчетов, появляются новая информация, которую надо хранить в системе. Наконец, просто приходит полный однофамилец, уже имеющегося сотрудника.

При этом обычно требуется не только создавать новые таблицы, но и менять существующие, да еще при этом нельзя останавливать работу системы.

Правильный подход в этой ситуации:

1. Доработать базу данных и программу, с ней взаимодействующую на модельной базе.
2. Потребовать специальный сервер для создания стенда и скопировать на него действующую в данный момент базу данных.
3. Написать последовательность SQL запросов, вносящих все необходимые изменения в базу данных.
4. Выполнить полученный пакет команда за командой, убедиться, что все команды работают так, как задумано.
5. Оттестировать работу программы с полученной базой данных
6. Еще раз перенести рабочую базу на сервер, прогнать все изменения пакетом. Оценить время выполнения пакета на реальном сервере. Допустимо ли приостанавливать

7. Сохранить реальную базу и только потом выполнять пакет запросов.

Бывают случаи, когда сложно написать запрос на изменение или добавление не очень большого множества записей. Или запрос на изменение выполняется долго, а СУБД на все это время блокирует доступ к изменяемой таблице и не дает работать другим пользователям.

Например, запишем в таблицу «Студент» наименования предметов, по которым они получили «5».

```
ALTER TABLE "Студент" ADD COLUMN BEST VARCHAR(255) DEFAULT ' ';
```

```
SELECT 'UPDATE "Студент" SET BEST=BEST||'' ||  
      "Предмет"."Наименование" || '' WHERE "№ зачетки"= '' ||  
      "Студент"."№ зачетки" || '' ;
```

```
"Результат" AS "Результат",
"Студент" AS "Студент",
"Ведомость" AS "Ведомость",
"Предмет" AS "Предмет"
```

```
"Результат"."Студент" = "Студент"."№ зачетки" AND
"Результат"."Ведомость" = "Ведомость"."Код" AND
"Ведомость"."Предмет" = "Предмет"."Код" AND
"Результат"."Оценка" = 5
```

Фрагменты результирующего SQL выражения (помечены красным) заключаются в апострофы и склеиваются с данными из таблиц с помощью «| |»

Номер зачетки должен быть строкой, и поэтому перед и после ее значения добавлены удвоенные апострофы «' ' ' | | "Студент". "№ зачетки" | | ' ' '»

Если ничего не поняли, посмотрите, что должно получиться в результате этого запроса. Это будет множество строк, представляющих собой SQL запросы:

```
UPDATE "Студент" SET BEST=BEST | ' Математика ' WHERE "№ зачетки"= '001';
UPDATE "Студент" SET BEST=BEST | ' Физика ' WHERE "№ зачетки"= '001';
UPDATE "Студент" SET BEST=BEST | ' Математика ' WHERE "№ зачетки"= '022';
```

Первый запрос добавит студенту с номером зачетки 001 в поле BEST к пустой строке, содержащейся в этой записи текст «Математика»

Второй запрос добавит студенту с номером зачетки 002 в поле BEST к строке «Математика», содержащейся в этой записи текст «Физика»

Третий запрос добавит студенту с номером зачетки 022 в поле BEST к пустой строке, содержащейся в этой записи текст «Математика»

В результате выполнения этих запросов у студента с номером зачетки 001 в колонке BEST будет «Математика Физика» а у студента с номером зачетки 022 «Математика»

К сожалению, в OpenOffice из конструктора запросов полученные строки, удастся копировать только по одной. Выходом является использование формы, которая будет описана далее в разделе «Вывод результатов запроса в текстовое окно».

31

Хранимые процедуры и функции. DATEADD, TIMEADD

А что делать, если встроенных процедур и функций недостаточно? В частности отсутствует встроенная функция, прибавляющая к заданной дате некоторое число дней, месяцев или лет.

В этом случае, придется писать программу.

Первый подход заключается в получении с помощью запросов данных, которые обрабатываются программой, и результаты с помощью других запросов заносятся в базу. Подробнее этот способ мы рассмотрим в разделе «Выполнение SQL запросов из макросов».

Второй подход заключается в написании процедур и функций, которые можно вызывать непосредственно из SQL запросов.

Большинство СУБД имеют встроенный язык (с различным синтаксисом) на котором можно написать хранимые в базе данных функции и процедуры. В HSQL вместо хранимых функций можно вызывать функции библиотек, написанных на языке JAVA.

Неплохой выбор, учитывая, что HSQL сам написан на JAVA, к сожалению, подключение библиотек не слишком удобно.

Рассмотрим подключение библиотеки на примере «hsqldbDates.jar»

(<http://www.ArielConstenlaHaile.com.ar/ooo/>), реализующей функции DATEADD и TIMEADD

Функция `ar.com.arielconstenlahaile.dbextras.funcsql.Fechas.agregarFecha(interval, amount, dt)` добавляет к дате, «dt» целое число «amount» лет, месяцев или дней, в зависимости от параметра «interval», который может принимать следующие значения: 'yy', 'year', 'mm', 'month', 'dd', 'day'.

Функция `ar.com.arielconstenlahaile.dbextras.funcsql.Fechas.agregarFechaHora(interval, amount, dt)` добавляет к дате / времени «dt», целое число «amount» лет, месяцев, дней, часов, минут, секунд, или миллисекунд. Параметр «interval» может принимать следующие значения: 'yy', 'year', 'mm', 'month', 'dd', 'day', 'mi', 'minute', 'ss', 'second', 'ms', 'millisecond'

Необходимо выполнить следующие действия:

1. Разрешите отображение скрытых папок. (Сервис/Свойства папки/Вид/Показывать скрытые файлы и папки)
2. Скопировать файл «hsqldbDates.jar» на жесткий диск (например в папку "C:\Program Files\OpenOffice.org 3"). **ВНИМАНИЕ!** Выбранный путь может содержать только латинские буквы, цифры и пробелы. Желательно хранить библиотеку вместе с файлами OpenOffice, поскольку, после удаления этой библиотеки все java-компоненты OpenOffice перестанут работать. В любом случае сделайте копию с конфигурационного файла "C:\Documents and Settings\Студент\Application Data\OpenOffice.org\3\user\config\javasettings_Windows_x86.xml" прежде чем выполнять следующие действия.
3. В главном окне программы из меню Сервис вызвать команду Параметры, выбрать из группы OpenOffice.org элемент Java. Нажать кнопку [Путь класса] затем кнопку [Добавить архив] найти нужный файл. Нажать кнопки [Открыть] [OK] [OK] и выйти из OpenOffice (все приложения).
4. Снова открыть нашу базу в OpenOffice.org Base
5. Разрешить вызов функций из этой библиотеки для этого: В главном окне программы из меню Сервис → Макросы → Управление макросами → OpenOffice.Org.Basic. Выделить элемент «Мои макросы → Standard» и нажать кнопку «Создать», затем «ОК». Откроется редактор basic.
6. Стереть имеющийся код и вставить следующую программу:

```
'-----
sub main
enableMethods( "addDateProc" , "ar.com.arielconstenlahaile.dbextras.funcsql.Fechas.*" )
end sub
'-----


Sub enableMethods(sName,sClass)
'Процедура enableMethods() - разрешает выполнение заданных процедур
' sName - имя правила
' sClass - класс, функции которого разрешено выполнять
oConfig = getConfigurationAccess( "org.openoffice.Office.DataAccess/DriverSettings" )
oHsqlDriver = oConfig.getByName( "com.sun.star.sdbcx.comp.hsqldb.Driver" )
oJavaMethods = oHsqlDriver.getByName( "PermittedJavaMethods" )
oJavaMethods.insertByName(sName,sClass)
oConfig.commitChanges( )
end sub
```



```

'-----
Function getConfigurationAccess(sNodePath$)
oConfigurationProvider =
    createUNOService("com.sun.star.configuration.ConfigurationProvider")
Dim oParametros(1) As New com.sun.star.beans.PropertyValue
oParametros(0).Name = "nodepath"
oParametros(0).Value = sNodePath
oParametros(1).Name = "EnableAsync"
oParametros(1).Value = FALSE
sConfiguration = "com.sun.star.configuration.ConfigurationUpdateAccess"
oConfigurationAccess =
    oConfigurationProvider.CreateInstanceWithArguments(sConfiguration,
    oParametros())
getConfigurationAccess() = oConfigurationAccess
End Function
'-----

```

7. Эту программу следует выполнить один раз, нажав кнопку  на панели инструментов. При повторном вызове она выдаст ошибку - элемент addDateProc уже добавлен.
8. Полное наименование функций длинновато, поэтому создаем псевдонимы функций: Из меню «Сервис» главного окна выберите команду «SQL...», и выполните команды:

```

CREATE ALIAS DATEADD FOR
    "ar.com.arielconstenlahaile.dbextras.funcsql.Fechas.agregarFecha";
CREATE ALIAS TIMEADD FOR
    "ar.com.arielconstenlahaile.dbextras.funcsql.Fechas.agregarFechaHora";

```

Обратите, что псевдонимы заданы латинскими буквами в верхнем регистре (их можно использовать без кавычек)

Теперь функции DATEADD и TIMEADD можно использовать в SQL запросах
Например, покажем наших студентов старше на год

```

SELECT "ФИО", DATEADD('YY', -1, "ДАТА РОЖДЕНИЯ") AS X FROM
"СТУДЕНТ" AS "СТУДЕНТ"

```

Если вы используете «конструктор запросов», выберите в нем колонки, задайте условия, а затем, отключите вид «дизайн» и включите режим «Выполнять непосредственно команду SQL».

Другой пример, сдвинем дату рождения всех студентов на месяц

```

UPDATE "СТУДЕНТ" SET "ДАТА РОЖДЕНИЯ"=DATEADD('MM', 1, "ДАТА
РОЖДЕНИЯ")

```

Создание форм с выбором значений из списка

Создание форм с помощью мастера.

Создадим форму для редактирования списка групп и списков студентов этих групп.

- В левом поле главного окна выбрать элемент «Формы»
- В поле «Задачи» выбрать элемент «Использовать мастер для создания формы»
- Из выпадающего списка выбираем таблицу «Группа»

- С помощью кнопки «>>>» добавить все поля таблицы «Группа» в правое поле.
- Нажать кнопку «Дальше»
- Чтобы на одной форме выводились и список групп и список студентов выбранной группы следует поставить последовательно отметки в полях «Добавить субформу», «Субформа, основанная на существующей связи».
- Выбрать таблицу «Студент».
- Нажать кнопку «Дальше»
- С помощью кнопки «>>>» добавить все поля таблицы «Студент» в правое поле.
- В списке студентов на форме всегда будут студенты только одной, выбранной группы. Поэтому поле «Группа» рекомендуется исключить с помощью кнопки «<<».

Мастер форм

Шаги

1. Выбор поля
2. Установка субформы
- 3. Добавить поля субформы**
4. Получить объединенные поля
5. Расположить элементы управления
6. Установка источника данных
7. Применить стили
8. Задать имя

Выберите поля вашей субформы

Таблицы или запросы
Таблица: Студент

Существующие поля
Группа

Поля в форме
№ зачетки
ФИО
Дата рождения

Двоичные поля всегда будут перечислены и могут быть выбраны из списка. Они будут отображены как изображения, если это возможно.

Справка < Назад Далее > Готово Отмена

- Также рекомендуется исключать поля с включенным автозаполнением, чтобы не путать пользователя.
- Нажать кнопку «Далее»
- Оставим тип расположения элементов «Лист данных»
- Остальные шаги пропустим. Нажать кнопку «Готово».
- Мастер автоматически создает форму и открывает ее для ввода данных. Имя созданной формы совпадает с именем главной таблицы.

vedomost4.odb : Группа - OpenOffice.org Базовы...

Файл Правка Вид Вставка Формат Таблица Сервис Окно Справка

Базовый Arial 10 Ж К Ч

Шифр	Факультет	Курс
AK1-11	AK	1
AK3-11	AK	2
CM1-11	CM	1

Запись 1 из 3

№ зачетки	ФИО	Дата рождения
001	Иванов Иван Иванович	10.10.1980
002	Петров Петр Петрович	11.11.1981
003	Сидоров Иван Федорович	12.12.1982

Запись 1 из 3

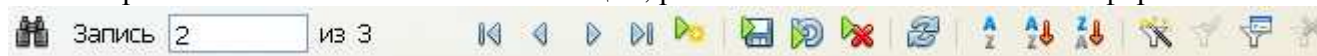
Страница 1 / 1 Обычный ВСТ СТАНД *






При выделении группы в верхней таблице, в нижней таблице высвечивается список студентов этой группы. При добавлении записи в нижнюю таблицу, в поле группа автоматически заносится шифр выбранной группы. Ну и, разумеется, автоматически заполняются поля с включенным режимом «автозначение»

Ввод данных с помощью формы.

Форма открывается в режиме ввода данных двойным щелчком мыши.

Обратите внимание на панель навигации, расположенной в нижней части формы.



Кнопки «», «», «», «» обеспечивают перемещение на первую, последнюю, предыдущую и следующую записи соответственно. Кнопка «*» перемещает в позицию добавление новой записи. Кнопка «Найти запись» позволяет найти запись по заданному условию.

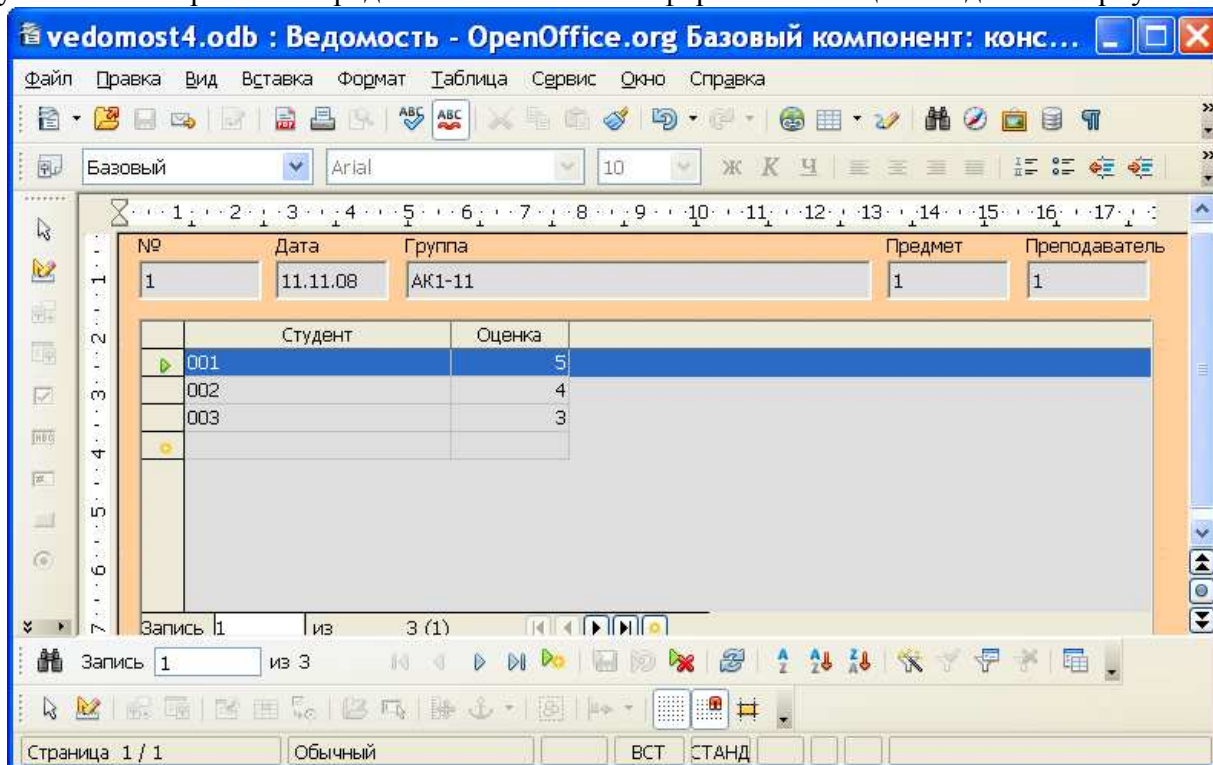
Введенные пользователем данные сохраняются в таблицу при попытке перехода к другой записи и при нажатии на кнопку «сохранить запись» на панели навигации.





Когда, как в нашем случае, на форме имеются несколько субформ, панель навигации управляет формой, в которой находится фокус ввода. Чтобы перейти к управлению другой субформой щелкните мышью по любому элементу ввода нужной субформы или главной формы.


При расположении элементов формы в виде «Лист данных», ее действие менее эффективно, зато более наглядно. В прочих типах расположения элементов помните, что ее действие ее действие точно такое же.

Формы с расположением элементов типа столбцы или блоки.

Создадим форму для ввода оценок студентов. На основе таблиц «Ведомость» и «Результат». Выберите тип представления головной формы «Столбцы – подписи сверху».



В этой форме представления данные из текущей записи таблицы представляются в отдельных полях формы. Несмотря на особую форму представления, на форме редактируется вся таблица «Ведомость». Для выбора записи, соответствующей нужной ведомости, используйте панель навигации. Но помните, что одна и та же панель управляет обоими субформами. Щелкните мышью по одному из полей субформы «группа». И используя кнопки «», «», «», «» выберите нужную группу.

В нижней субформе выводятся оценки, связанные с выбранной ведомостью. Чтобы добавить новую ведомость, нажмите кнопку «*» на панели навигации, введите информацию о ведомости и нажмите кнопку «Сохранить запись» на панели навигации. Только после сохранения


ведомости вы сможете добавлять оценки студентов. Обратите внимание, что надо вводить не ФИО преподавателя и наименование предмета, а коды соответствующих им записей.

Редактирование формы с помощью конструктора.


Созданная нами форма имеет несколько недостатков:

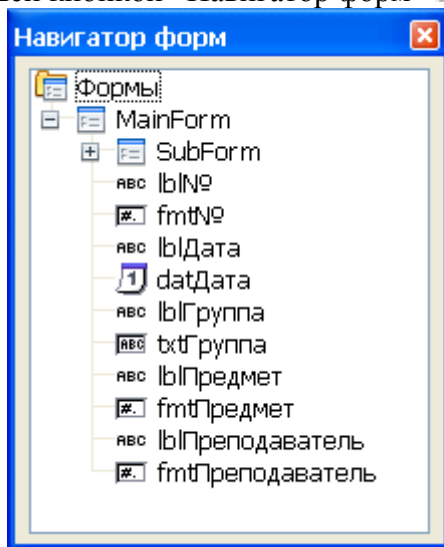
- Одни поля и колонки слишком широкие, а другие слишком узкие.
- Неудачный порядок и расположение элементов, и последовательность колонок.
- Неудачный цвет, шрифт, и форма представления данных.
- Особенно неудобно вводить коды связанных записей, да и в шифре группы легко сделать ошибку.

Чтобы изменить расположение, ширину элементов формы, оформление и многое другое надо закрыть форму и снова открыть ее с помощью команды «Изменить» из контекстного меню. Чтобы далее перейти в режим ввода данных можно, не закрывая окно дизайнера формы нажать

кнопку «Режим разработки» . Повторное нажатие этой кнопки снова переводит форму в режим разработки.

Чтобы передвинуть одно из полей формы щелкните по нему левой кнопкой мыши и не отпуская ее перетащите элемент на новое место. Обратите внимание, что надпись перетаскивается вместе с полем данных. Дело в том, что они объединены в группу. Чтобы разгруппировать элементы, выделите группу, и из контекстного меню выберите последовательно «Группировка», «Разгруппировать». Теперь можно изменять элементы по отдельности. В частности уменьшить ширину полей «№», «Группа» и увеличить поля «Предмет» и «Преподаватель». Впрочем, не спешите разгруппировывать все элементы.

Воспользуемся полезнейшей кнопкой «Навигатор форм» .



Выделите на навигаторе элемент «fnt№», и меняйте в окне конструктора формы ширину элемента для ввода номера ведомости не разбивая группу (кто попробовал дать команду «Разгруппировать» заметил, как медленно вызывается контекстное меню в конструкторе). Мало того, можно выделить (удерживая нажатой клавишу «Ctrl») несколько элементов и перетащить их все вместе на другое место.

Выберите из контекстного меню элемента «datData» команду «Свойства». Перейдите на закладку «Общие» и используя полосу прокрутки найдите свойство «Раскрываемый». Задайте ему значение «Да» и закройте диалог «Свойства». Теперь вы сможете выбирать дату на календаре, а не только вводить с клавиатуры.

Можно расширить колонки в таблице, для этого, подведите указатель мыши к границе колонок (в шапке таблицы), и перетащите с помощью мыши ее вправо.

Настала пора организовать ввод преподавателя путем выбора значений из выпадающего списка. Но сначала, как описано выше, поля «группа», «предмет», «преподаватель», выделите в поля «fntГруппа», «fntПредмет», «fntПреподаватель» и удалите их нажатием клавиши «Delete».

Создание выпадающего списка

Организуем ввод данных в поле «предмет» путем выбора наименования предмета из раскрывающегося списка.

Убедитесь, что на панели инструментов «Элементы управления» утоплена кнопка «Мастер» (нижняя кнопка на левой панели).

На панели инструментов «Элементы управления», расположенной слева от формы нажмите кнопку «Список» (обратите внимание, не «поле со списком», а именно «список»).

Теперь подведите курсор мыши к тому месту, где хотите создать выпадающий список, нажмите левую кнопку мыши, затем, не отпуская кнопку, растяните пунктирную рамку до нужных размеров. Отпустите кнопку мыши.

Высвечивается окно мастера списка.

Выберите таблицу, из которой будем выбирать данные («Предмет»). Нажмите «далее».

Выберите отображаемое поле «Наименование». Нажмите кнопку «далее».

В левом списке выберите поле «Предмет», а в правом «Код». Тем самым мы сообщаем, какие поля таблиц «Ведомость» и «Предмет» соответствуют друг другу.

Нажмите кнопку «Готово».

Убедимся, что все сделали правильно.

Нажмите кнопку «Режим разработки». Форма переходит в режим ввода данных и в полях появляются данные из первой записи (если она есть).

Нажмите кнопку в правой части поля «Предмет». Должен появиться список наименований предметов из таблицы предмет. Если список пуст, прежде всего, убедитесь, что в таблице «Предмет» есть записи.

Выберите один из предметов. Нажмите на панели навигации кнопку «Сохранить запись».

Для возврата в режим конструктора форм, снова нажмите кнопку «Режим разработки».

Если хотите, оформление вашего списка было тем же, что назначает мастер при создании формы, выделите наш список, дайте из контекстного меню команду «Элемент управления» (или команду «свойства» из контекстного меню навигатора форм). На закладке «Общие» нажмите кнопку «...». Напротив слов «Цвет фона» и введите в поля «Красный», «зеленый» и «синий» число 221. Свойство «Обрамление» задайте «Трехмерный вид». Для удобства обращения к этому полю из вашей программы (в дальнейшем) желательно задать «имя», например, «ListПредмет».

Создайте по аналогии поля для ввода группы и преподавателя.

№	Дата	Группа	Предмет	Преподаватель
1	11.11.2008	АК1-11	Базы данных	Тимофеев

Студент	Оценка
001	5
002	4
003	3

Запись 1 из 3

Имеется альтернативный способ создания выпадающего списка путем смены типа простого поля (команда «Заменить на ...» из контекстного меню навигатора формы). Но во первых, он сложнее (придется вручную заполнять поля на закладке «Данные»). А во вторых, «OpenOffice.org Base» версия «3.0» стал иногда аварийно завершаться при выполнении операции смены типа.

Создание выпадающего списка в таблице

1. Открываем форму на редактирование
2. Щелчком мыши на заголовке колонки выбираем колонку.
3. Командой «Заменить на ► список» из контекстного меню превращаем его в список (обратите внимание, не поле со списком, а именно список)
4. Даем команду «Столбец» из контекстного меню. На закладке «данные» в поле «тип содержимого списка» выбираем «Sql». В поле «содержимое списка» вводим запрос, который возвращает две колонки: представляемый текст и код записи: «SELECT "ФИО" , "№ зачетки" FROM "Студент" AS "Студент" ORDER BY "ФИО" ASC». Можно воспользоваться конструктором запросов, нажав кнопку «...». Связываемое поле оставляем «1» – это номер колонки запроса (считая с 0), из которой берутся данные для связывания с соответствующим полем редактируемой записи.
5. Сохраняем и закрываем форму.

Создание независимых субформ

Часто надо на одной форме представить данные из нескольких независимых таблиц или некую справочную информацию, получаемую с помощью запросов.

Добавим (в учебных целях) на форму «Ведомость» субформу для редактирования списка предметов. Обычно так не делают, но если надо будет представить на форме результаты запросов, то последовательность действий будет та же.

Вызываем навигатор форм.

Выделяем элемент «Формы» и из контекстного меню выбираем команды «Создать», «Форма».

Выделяем вновь созданную форму и из контекстного меню вызываем команду «Свойства»

На закладке «Общие» задаем имя формы, например, «formПредмет»

На закладке «Данные» задаем в поле «Содержимое» таблицу «Предмет».

Закрываем диалог «Свойства формы»

Убедитесь, что в навигаторе формы выделена наша новая форма «formПредмет».

На панели инструментов «Элементы управления» (слева) нажимаем кнопку «Дополнительные элементы управления» (третья снизу).

На этой панели нажимаем кнопку «Таблица»

Теперь подведите курсор мыши к тому месту, где хотите представить список предметов, нажмите левую кнопку мыши, затем, не отпуская кнопку, растяните пунктирную рамку до нужных размеров. Отпустите кнопку мыши.

Открывается окно «мастер элемента управления».

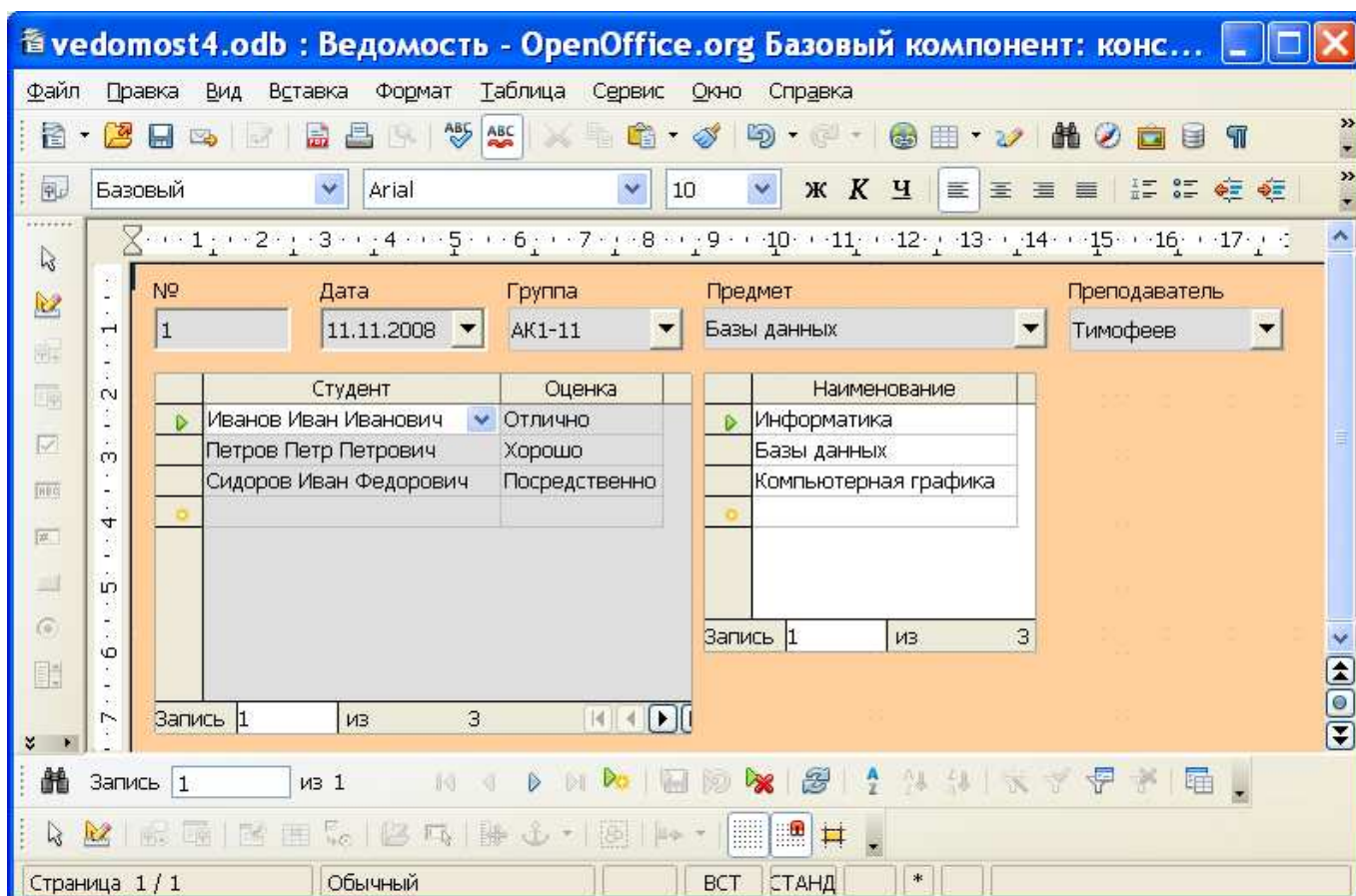
Выберите представляемые колонки (в нашем случае только «Наименование») и нажмите кнопку «Готово».

Отожмите кнопку «таблица» на панели «Дополнительные элементы управления» и закройте эту панель.

Расширьте, при необходимости колонки и таблица готова.

Убедимся, что все сделали правильно.

Нажмите кнопку «Режим разработки». Форма переходит в режим ввода данных, и в нашей новой таблице появляются данные из таблицы «Предмет».



Попробуйте добавить новый предмет. Не забудьте нажать кнопку «сохранить запись».

Внимание! Новый предмет не появится автоматически в выпадающем списке «Предмет» главной формы.

Щелкните на этом поле мышью и нажмите кнопку «Обновить» на панели навигации.

Обновятся все поля. Жаль, что форма при этом переходит на первую запись.

Есть способ обновить только поле выпадающего списка, и даже автоматически, но для этого надо сначала освоить работу с макросами.

Доступ к элементам форм из программ на basic

Библиотеки макросов

Функциональность конкретной задачи пользователя, а так же недостающую функциональность «OpenOffice», можно реализовать путем написания макросов - собственных функций на встроенном языке «OpenOffice.org Basic».

Прежде всего, надо установить средний уровень безопасности, для этого выбираем из меню «Сервис» команду «Параметры», раскрываем группу «OpenOffice.org», выбираем элемент «Безопасность». Нажимаем кнопку «Безопасность макросов», выбираем элемент «средний», сохраняем изменения «ОК», «ОК». После этого необходимо закрыть ВСЕ приложения «OpenOffice», включая Writer, только после этого понижение уровня безопасности подействует.

Имеется два места для хранения макросов, оба они имеют свои преимущества и недостатки:

- Макросы можно хранить в одной из форм (следовательно, в документе). Естественное место обработчиков различных событий формы (нажатие кнопки, переход на другую запись...). К сожалению, такие макросы доступны, только пока открыта форма. Изменить их можно, только открыв форму на редактирование. При закрытии формы закрывается редактируемый модуль.
- Начиная с версии 3.3 макросы можно хранить только в документе (в форме хранить нельзя).
- Макросы можно хранить в общих модулях, в папке "C:\Documents and Settings\UserName\Application Data\OpenOffice.org2\user\basic\Standard*.xba". Эти макросы доступны из всех документов, макросы из этих модулей удобнее всего отлаживать, но при переносе на другой компьютер документа надо не забыть перенести и используемые модули.

Для работы с макросами надо из меню «Сервис» выбрать последовательно команды «Макросы», «Управление макросами», «OpenOffice.org Бейсик». Открывается диалог «макросы», с которого можно:

- Выполнить макрос (кнопка «Выполнить»)
 - Вызвать макрос на редактирование (кнопка «Редактировать»)
 - Назначить запуск макроса (кнопка «Назначить») по команде меню, кнопке панели инструментов, событию жизненного цикла документа.
 - Открыть диалог «управление макросами» (кнопка «Управление») на котором можно:
 - Создать новый модуль (кнопка «Новый модуль»)
 - Удалить модуль (кнопка «Удалить»)
 - Перетащить (с помощью мыши) модуль в другую библиотеку.
- Если макрос хранится в форме, следует предварительно открыть эту форму на редактирование.

Вызов функции по нажатию кнопки

Переходим на закладку «Формы». Создаем новую форму в режиме дизайна. Уменьшаем размер окна, и нажимаем кнопки «Закрыть» (в правом верхнем углу окна), «Сохранить», вводим имя формы «Форма 1», «Сохранить».

Снова вызываем нашу форму на редактирование командой «Правка» из контекстного меню.

На панели инструментов «Элементы управления» выбираем элемент «Кнопка». В области формы нажимаем левую кнопку мыши и растягиваем рамку до размеров будущей кнопки и отпускаем кнопку мыши. Двойным щелчком мыши на кнопке вызываем диалог «Свойства». На закладке «общие» можно задать текст, картинку на кнопке и другие свойства кнопки.

Теперь подготовим макрос. Из меню «Сервис» формы последовательно выбираем команды «Макросы», «Управление макросами», «OpenOffice.org Бейсик». Открывается диалог «макросы».

Раскрываем элемент, соответствующий нашей форме, нажимаем кнопку «Создать», «ОК».

Пишем макрос.

```
'-----  
Sub OnBtnMsg(oEvent)  
msgbox("****")' выдать на экран сообщение ***  
End Sub  
'-----
```

И сохраняем модуль (Файл → Сохранить)

Переключаемся снова на нашу форму. На закладке «События» задаются реакции на разные события связанные с кнопкой. Находим строку «Нажатие клавиши мыши» и нажимаем последовательно кнопки «...», «Макрос». В левом окне раскрываем элементы «Private:object» «Standart», выбираем «Module1» в правом окне выбираем наш макрос «OnBtnMsg», [OK], [OK].

Закрываем окно свойств, сохраняем форму (Файл → Сохранить)

Переключаемся в режим работы с формой, для этого нажимаем на панели инструментов «Элементы управления» кнопку «режим разработки».

Теперь, если нажать нашу кнопку, должно появиться окошко с текстом «***»

Чтобы снова вернуться в режим редактирования формы, снова нажимаем кнопку «режим разработки».

Настройка вызова редактора макросов по кнопке

Чтобы не выполнять впредь вышеописанную процедуру вызова редактора макросов, настроим вызов диалога «Макрос OpenOffice.org Basic» по кнопке на стандартной панели инструментов главного окна.

Для этого переключаемся на главное окно «OpenOffice», выбираем из меню «Сервис» главного команду «Настройка». Выбираем закладку «Панели инструментов». В поле «сохранить в» выбираем наш документ. В поле «Панель инструментов» выбираем «Стандартная». В поле «Команды» выбираем команду, после которой будем добавлять новую кнопку. Нажимаем кнопку «Добавить», выбираем категорию «Бэйсик», команду «OpenOffice.org Basic», нажимаем кнопку «Добавить».

Чтобы назначить значок для кнопки, выбираем добавленную нами команду «OpenOffice.org Basic», Нажимаем кнопку «Изменить», выбираем «Выбрать значок». Выбираем подходящий значок, и нажимаем «ОК».

Чтобы сохранить настройки, нажимаем кнопку «ОК». Сохраняем документ.

Получение значений, введенных в поля формы

Вызываем форму «Форма 1» на редактирование, добавляем на нее еще одну кнопку и одно текстовое поле. Открываем окно «свойства» двойным щелчком мыши на текстовом поле и задаем на закладке «Общие» имя поля «TextBox1».

Вызываем на редактирование модуль, связанный с формой («Сервис» формы последовательно выбираем команды «Макросы», «Управление макросами», «OpenOffice.org Бейсик».).

```
'-----  
Sub OnBtnGetText(oEvent)  
oForm = oEvent.Source.getModel().getParent() 'Получаем объект формы  
sVar=oForm.getByName("TextBox1").text 'Получаем значение, введенное в форму  
msgbox(sVar) 'выдать на экран значение, введенное в поле  
End Sub  
'-----
```

Назначаем на событие нажатия кнопки макрос «OnBtnGetText».

Переключаемся в режим работы с формой.

При нажатии кнопки макрос должен вывести текст, введенный в текстовое поле.

Изменение значений, в полях формы

Создаем еще одно текстовое поле «TextBox2» для изменения текста в этом поле добавим в макрос «OnBtnGetText» строку:

```
oForm.getByName("TextBox2").text=sVar
```

Получение данных из текущей записи формы

Создаем новую форму в режиме мастера на основе одной из таблиц. Добавляем кнопку. Назначаем на событие нажатия кнопки макрос «OnBtnGetTextR».

```
'-----  
oForm = oEvent.Source.getModel().getParent()  
if oForm.isAfterLast() or oForm.isBeforeFirst() then  
    msgbox("Сначала выберите запись")  
    exit sub  
end if  
sCol1=oForm.getString(oForm.findColumn("студент")) 'Значение типа String  
iCol2=oForm.getInt(oForm.findColumn("оценка")) 'Значение типа Integer  
msgbox(sCol1+":"+iCol2) 'выдать на экран значение, введенное в поле записи  
End Sub  
'-----
```

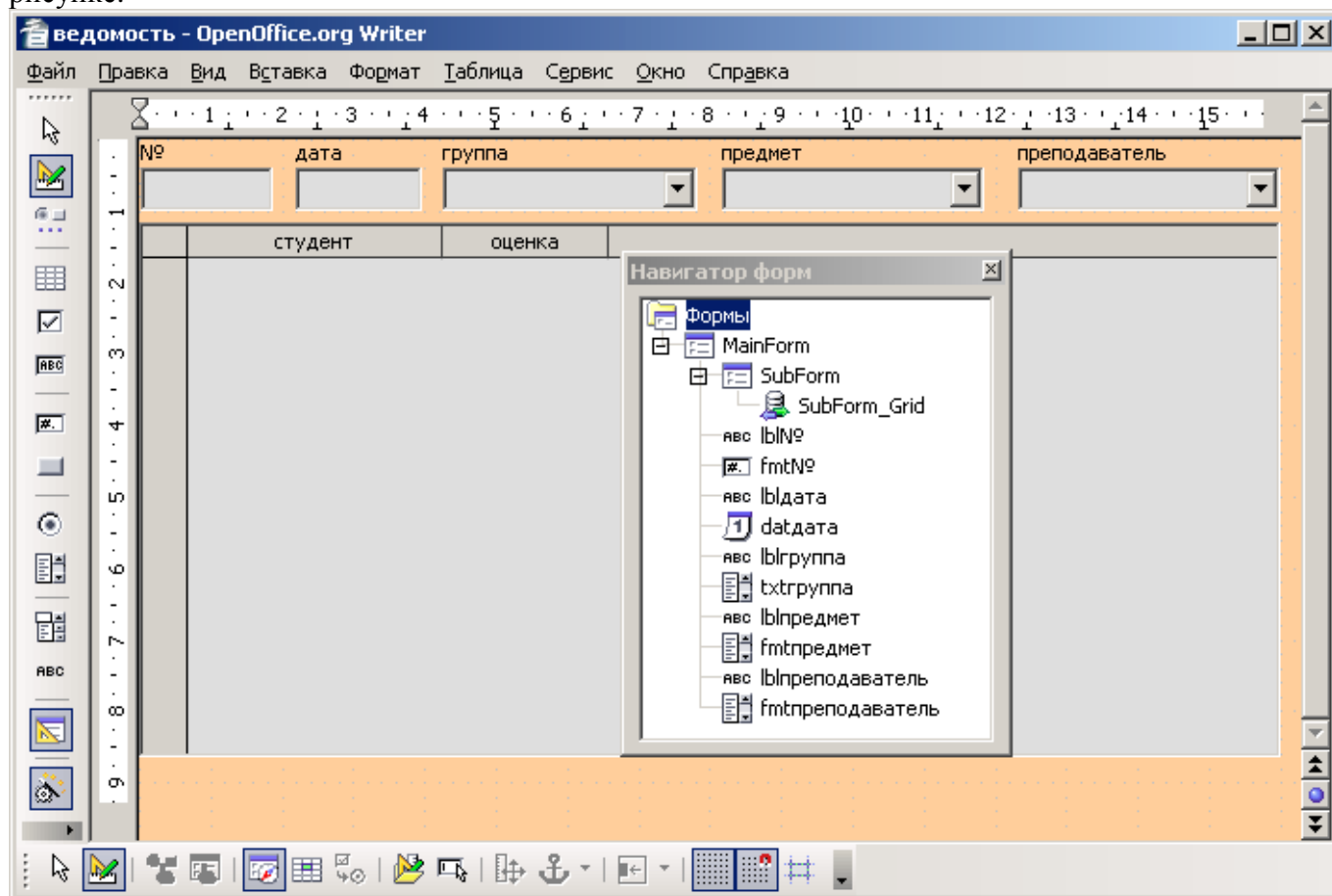
Обратите внимание, что для получения данных разных типов следует использовать разные функции «getString, getInt, getDouble, getDate»

Изменение данных текущей записи формы

```
'-----  
Sub OnBtnPlus(oEvent)  
oForm = oEvent.Source.getModel().getParent()  
if oForm.isAfterLast() or oForm.isBeforeFirst() then  
    msgbox("Сначала выберите запись")  
    exit sub  
end if  
lCol = oForm.findColumn("оценка") 'индекс колонки  
iVal=oForm.getInt(lCol) 'считываем значение  
oForm.updateInt(lCol,iVal+1) 'записываем новое значение  
oForm.updateRow() 'сохраняем измененную запись в базу данных  
End Sub  
'-----
```

Представление зависимых таблиц на одной форме

Форму, на которой представлены данные из двух связанных таблиц, можно создать с помощью мастера создания формы. Формируется система объектов представленная на следующем рисунке:



Обратите внимание на то, что одним термином «форма» в «OpenOffice.org Base» называется два разных объекта:

- форма – карточка, которую мы видим (в данном примере «Ведомость»)
- форма – источник данных для представления в элементах управления («MainForm» и «SubForm»).

Приходится ориентироваться по контексту.

Представление независимых таблиц на одной форме

Создаем, с помощью мастера создания формы, форму для представления данных из одной из таблиц или одного запроса.

Вызываем эту форму на редактирование.

Нажимаем кнопку «навигатор форм» на панели инструментов «Дизайн формы».

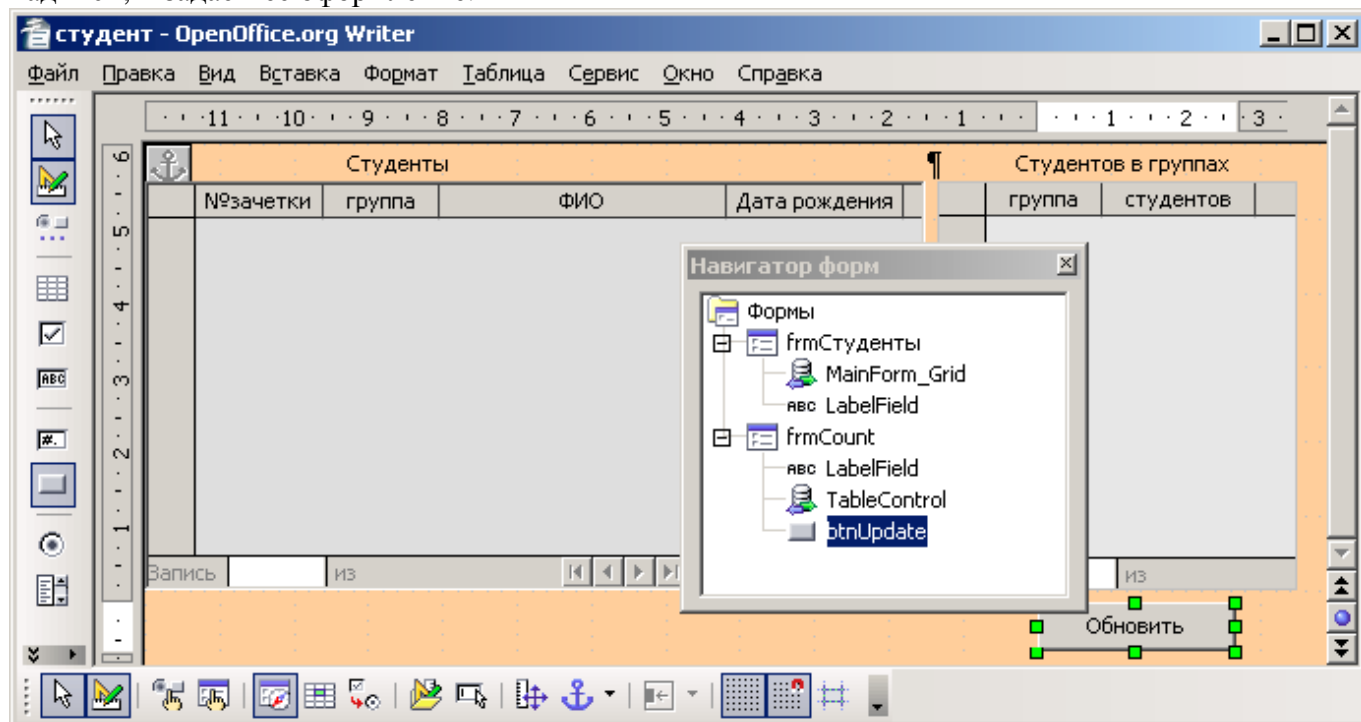
Выделяем элемент «Формы» и из контекстного меню выбираем команды «Создать», «Форму»

Переименуем формы для удобства работы с ними в дальнейшем. Для этого выделяем первую форму и из контекстного меню выбираем команду «Свойства». На закладке «Общие» в поле «Имя» задаем новое имя формы (например, frmСтуденты). Имя второй формы можно изменить, не закрывая окно свойств формы, достаточно выделить в навигаторе вторую форму и ввести имя (например, frmCount).

Теперь добавляем на форму «frmCount» визуальный элемент «таблица». Для этого выделяем в навигаторе нужную форму, нажатием на панели инструментов «Элементы управления» кнопки «Дополнительные элементы управления» вызываем одноименную панель инструментов, на ней нажимаем кнопку «Таблица». Нажимаем левую кнопку мыши на месте левого верхнего угла новой таблицы, и, не отпуская кнопку мыши, растягиваем пунктирную рамку до нужных размеров. После отпускания клавиши мыши появляется окно «мастер элемента управления таблицей», с помощью которого выбираем нужную таблицу или предварительно

созданный запрос, затем выбираем представляемые ее поля. Чтобы созданную таблицу можно было свободно перемещать по форме, выделяем нашу таблицу и выбираем из контекстного меню команду «Положение и размер». На закладке «Положение и размер» выбираем привязку к странице и нажимаем кнопку «ОК».

Для создания подписей к таблицам следует нажать на панели инструментов «Элементы управления» кнопку «Метка», и растянуть рамку на месте будущей надписи. Двойным щелчком на созданном элементе вызываем панель «Свойства элемента» и на закладке «Общие» вводим текст надписи, и задаем ее оформление.



Элементы управления, рекомендуется создавать структурно под той формой, к которой они относятся. При необходимости можно в навигаторе перетащить мышью элемент управления под другую форму.

Для того, чтобы количество студентов выводилось, не в виде «20,00», а как целое число, выделяем соответствующую колонку и из контекстного меню выбираем команду «Столбец». На закладке «Общие» в поле «точность» задаем 0 знаков после запятой.

Обновление форм после изменения данных в базе.

После изменения данных в одной таблице формы часто требуется обновлять данные, в других элементах формы. В нашем примере после изменения списка студентов требуется обновить количество студентов в группах, формируемое с помощью запроса.

Добавляем кнопку «Обновить» под форму «frmCount».

Пишем новый макрос:

```
Sub OnBtnUpdate(oEvent)
oForm = oEvent.Source.getModel().getParent()
oForm.reload()
End Sub
```

Впрочем, в данном конкретном случае правильнее связать процедуру обновления формы «frmCount» не с кнопкой, а с событием «После сохранения» формы «frmСтуденты».

Если источник события лежит под другой формой, или требуется обновить несколько форм следует использовать следующий код:

```
Sub OnBtnUpdate1(oEvent)
'oForm = oEvent.Source.getModel().getParent() 'код для событий от элементов формы
oForm=oEvent.Source 'источником события является непосредственно форма
oForms=oForm.getParent()
oForms.GetByName("frmCount").reload()
```

End Sub

Обратите внимание, что событие приходит непосредственно от формы, поэтому не требуется вызовов «getModel().getParent()»

Обновление содержимого выпадающего списка.

При написании макросов часто приходится переключаться из режима разработки в режим работы с формой и обратно при этом случается, что «пропадает» содержимое выпадающего списка. В этом случае поможет следующая программа:

```
Sub OnInit(oEvent)
oForm = oEvent.Source 'источником события является непосредственно форма
oForm.getByName("ComboBox1").refresh() 'Обновляем элемент ComboBox1
End Sub
```

Этот макрос следует связать с событием «При загрузке» в свойствах формы.

Изменение запроса, связанного с формой

До сих пор мы создавали с помощью конструктора запросы с конкретными значениями в поле «критерий» и там же их выполняли. В принципе имеется возможность делать запросы с параметрами, но при работе с формой имеется лучший способ. Можно изменять SQL выражение, с помощью которого выбираются данные для представления на форме.

Организуем вывод на форме информации о студентах группы, выбранной из выпадающего списка.

The screenshot shows a form with a dropdown menu at the top containing the text 'АКЗ-11'. Below the menu is a table with the following data:

	№ зачетки	ФИО	Дата рождения	Группа
▶	021	Фролов И	03.03.79	АКЗ-11
	022	Никонов		АКЗ-11
☀				

Сначала создаем в конструкторе запрос, выводящий информацию о студентах одной из групп, и сохраняем запрос. Рекомендуется, так же скопировать в буфер обмена текст полученного SQL запроса

Создаем, с помощью мастера форму на основе полученного запроса. Открываем ее в режиме редактирования. Из контекстного меню даем команду «Свойства формы» и на 2 закладке выбираем тип источника данных «Команда SQL». В поле «Содержимое» заносим текст SQL запроса для первого открытия формы. С помощью макроса мы будем менять текст этого запроса.

Добавим на форму элемент типа «поле со списком», используя левую панель инструментов. (Предварительно убедитесь, что утоплена кнопка «Мастер»). Отвечая на вопросы мастера, организуем вывод в выпадающий список шифров групп из таблицы «Группа».

Создаем модуль с приведенным ниже текстом программы.

```
Function replace(str as String,sFrom as String,sTo as String) as String
'Заменяет в строке str все подстроки sFrom на sTo и возвращает полученную строку
Dim s as String
Dim i as Integer
s=str
i=1
Do While true
    if i>len(s) then
        Exit Do
    End If
    i=InStr(i,s,sFrom)
    if i=0 then' подстрока sFrom не найдена
        Exit Do' выход из цикла
    End If
    s=mid(s,1,i-1) + sTo + mid(s,i+len(sFrom))
    i=i+1
End Function
```

```

Loop
replace=s 'чтобы функция вернула строку s
End Function
'-----
Sub OnChangeGroup(oEvent)
'меняет текст SQL запроса в соответствии с текстом, заданным в поле «ComboBox»
oForm = oEvent.Source.getModel().getParent() 'Получаем объект формы
sGrp=oForm.getByName("ComboBox").text 'Получаем значение, введенное в поле
s="SELECT "№ зачетки", "ФИО", "Дата рождения", "Группа" FROM "Студент" AS S
WHERE "Группа" = '?1'"
s=replace(s,"?1",sGrp) ' подставляем на место «?1» шифр группы
oForm.command=s ' меняем текст запроса в форме
oForm.reload()' загружаем данные в соответствии с запросом s
End Sub
'-----

```

Для элемента управления «Поле со списком», из контекстного меню даем команду «Элемент управления» и на 3 закладке связываем событие «Состояние изменено» с процедурой «OnChangeGroup».

Можно разместить на форме несколько полей и по нажатию кнопки получать выборку с учетом введенных данных.

The screenshot shows a form with two dropdown menus: 'Группа' (Group) with 'AK1-11' selected, and 'Дата рождения с' (Date of birth from) with '01.01.1980' selected. There is also a 'по' (to) dropdown with '31.12.1980' selected and an 'Искать' (Search) button. Below the form is a table with the following data:

	№ зачетки	ФИО	Дата рождения	Группа
▶	002	Петров Петр Петрович	11.11.80	AK1-11
✱				

В приведенной ниже программе функция «dateToStr» возвращает значение типа дата для подстановки в SQL выражение. С использованием функции «addCondition» формируется строка условий отбора записей, причем, если параметр не задан на форме, то и условие на него в SQL выражение не добавляется.

```

'-----
function addCondition(s as String, exp as String, v as String) as String
'если v<>"" подставляет в выражение exp значение v на место символа ?
'и добавляет в строку s вместе с ключевым словом WHERE или AND
if v<>"" then
    if s="" then
        s=s+" WHERE "+replace(exp,"?",v)'первое условие
    else
        s=s+" AND "+replace(exp,"?",v)
    end if
endif
addCondition=s 'чтобы функция вернула строку s
End Function
'-----
function dateToStr(v as Variant) as String
'если дата v задана возвращает строку для подстановки в SQL - {D '1980-01-01' }
'иначе возвращает пустую строку
s=""
if not isempty(v) then
    s=Str(v)
    s="{D ' "+mid(s,2,4)+"-"+mid(s,6,2)+"-"+mid(s,8,2)+"'}"
endif
dateToStr=s 'чтобы функция вернула строку s
End Function
'-----
Sub OnBtnRun(oEvent)
'меняет текст SQL запроса в соответствии со значениями,
'введенными в поля «ComboBox», «Date1», «Date2»
oForm = oEvent.Source.getModel().getParent() 'Получаем объект формы
'считываем значения из полей
sGrp=oForm.getByName("ComboBox").text 'Получаем значение
lDate1=oForm.getByName("Date1").date

```

```

lDate2=oForm.getByName("Date2").date
s="SELECT " & "№ зачетки" & ", " & "ФИО" & ", " & "Дата рождения" & ", " & "Группа" & " FROM " & "Студент" & " AS S "
c=" " & 'строка условий
c=addCondition(c, " " & "Группа" & " = '?' " & ", sGrp)
c=addCondition(c, " " & "Дата рождения" & " >= ? " & ", dateToStr(lDate1))
c=addCondition(c, " " & "Дата рождения" & " <= ? " & ", dateToStr(lDate2))
' получится WHERE "Группа" = 'AK1-11' AND "Дата рождения" >= {D '1980-01-01'}
AND "Дата рождения" <= {D '1980-12-31'}
s=s+c
oForm.command=s 'меняем текст запроса в форме
oForm.reload()' загружаем данные в соответствии с запросом s
End Sub
' -----

```

Открытие формы по нажатию кнопки

Создадим еще одну форму и сохраним ее под именем «Форма 2»

Вызываем форму «Форма 1» на редактирование, добавляем на нее еще одну кнопку.

Вызываем на редактирование модуль, связанный с формой.

Дописываем функцию – обработчик события.

```

' -----
Sub OnBtnOpenFrom(oEvent)
'открывает форму с именем "Форма2"
Dim args(1) As New com.sun.star.beans.PropertyValue
Dim container as variant
Dim oCon
oCon = oEvent.Source.Model.Parent.ActiveConnection
container = oCon.Parent.DatabaseDocument.FormDocuments
args(0).Name = "ActiveConnection"
args(0).Value = oCon
args(1).Name = "OpenMode"
args(1).Value = "open"
container.loadComponentFromURL("Форма2", "_blank", 0, args())
End Sub
' -----

```

Открываем окно «свойства» кнопки, задаем в качестве обработчика события «Нажатие клавиши мыши» макрос «OnBtnOpenFrom»

Выполнение SQL запросов из макросов

Выполнение запросов на изменение данных

Создадим с помощью мастера форму на основе таблиц «ведомость» и «результат» (смотри главу «Создание форм с выбором значений из списка»), добавим на нее кнопку «Добавить студентов группы», добавляем обработчик события нажатия этой кнопки – процедуру «OnBtnAddGroupStud»

```
'-----  
Sub OnBtnAddGroupStud(oEvent)  
'добавляет в таблицу "результат" записи,  
'соответствующие всем студентам текущей группы  
oForm = oEvent.Source.getModel().getParent()  
oForm.updateRow() 'сохраняем изменения пользователя в базу данных  
oCon=oForm.ActiveConnection 'объект для доступа к базе данных  
idVed=oForm.getInt(oForm.findColumn("код")) 'считываем значение (код ведомости)  
sGroup=oForm.getString(oForm.findColumn("группа")) 'считываем (шифр группы)  
sSQL="INSERT INTO "результат" SELECT "№зачетки" as "студент", "+idVed+" as  
"ведомость", null as "оценка" FROM "студент" WHERE "группа"=" "+sGroup+" "  
oStatement=oCon.createStatement() 'объект для выполнения запросов  
oStatement.executeQuery(sSQL) 'выполняем запрос  
oForm.GetByName("SubForm").reload() 'обновляем подчиненную форму  
End Sub  
'-----
```

Обратите внимание, что колонка шифр группы – текстового типа, поэтому формируем строку запроса так, чтобы шифр группы оказался, заключен в апострофы:

"[группа]='"+sGroup+""

Вообще в запросах на HSQL имена таблиц и колонок записываются в кавычках, но и признак строки в «Basic» тоже кавычки, поэтому кавычки удваиваются. Поле «группа» имеет строковый тип, поэтому значение шифра группы заключаем в апострофы.

Еще одно замечание: Выражение «oCon=oForm.ActiveConnection» правильно работает, только если форма связана с таблицей базы данных. В нашем случае эту операцию автоматически сделал мастер форм. При создании формы средствами конструктора форм, придется делать эту операцию самим.

Универсальная форма для запросов на изменение данных

Стандартный интерфейс конструктора запросов (даже в режиме SQL) ориентирован на получение выборки, данных. Запросы на изменение данных можно выполнять, используя команду «SQL» из меню «Сервис» главного окна. Но интерфейс данной команды имеет ряд недостатков, поэтому создадим собственную форму для выполнения запросов, обеспечивающую сохранение команд в таблице.

Создаем таблицу «commandsSQL», с колонками «Код» (целое, первичный ключ) и «strSQL» (Текст 1024 символа). Создаем форму, используя, мастер создания форм. Открываем форму в режиме изменения, задаем тип текста для текстового поля «Многострочный», добавляем кнопку «Выполнить», задаем обработчик события нажатия этой кнопки – процедуру «OnBtnExec»

```
'-----  
Sub OnBtnExec(oEvent)  
'Выполняет запрос, заданный в поле txtSQL  
'соответствующие всем студентам текущей группы  
oForm = oEvent.Source.getModel().getParent()  
oCon=oForm.ActiveConnection 'объект для доступа к базе данных  
sSQL=oForm.getByName("txtstrSQL").text  
oStatement=oCon.createStatement() 'объект для выполнения запросов  
oStatement.executeQuery(sSQL) 'выполняем запрос  
oCon.getTables().refresh() 'Оповещаем OpenOffice об изменении базы  
End Sub  
'-----
```

Внимание! Если ваш запрос создает или удаляет таблицу, после его выполнения необходимо переключиться на главное окно, в левом выбрать режим «Таблицы», затем выполнить

из меню «вид» команду «обновить таблицы». Еще одно замечание. Прежде, чем выполнять запрос, сохраните запись, после выполнения запроса это сделать не удастся. Как с этим бороться, пока не знаю.

Выполнение параметрических запросов

Когда одну и ту же SQL выражение надо выполнить много раз с разными параметрами, желательно вместо «CreateStatement()» использовать «PrepareStatement()».

При этом в SQL выражении на месте изменяемых параметров ставятся символы «?», а значения параметров задаются непосредственно перед выполнением запроса с помощью функций «SetString, SetInt, SetDouble»

```
'-----  
Sub OnBtnRnd(oEvent)  
'добавляет в таблицу M1 случайные числа  
'Перед вызовом этой функции необходимо удалить все данные из таблицы  
Dim i as Integer  
Dim j as Integer  
Dim m as Double  
oForm = oEvent.Source.getModel().getParent()  
oCon=oForm.ActiveConnection 'объект для доступа к базе данных  
sSQL="insert into "M1" ("i","j","M") values (?,?,?)"  
oStatement = oCon.PrepareStatement(sSQL)  
Randomize  
for i=1 to 3  
    for j=1 to 3  
        oStatement.SetInt(1,i)  
        oStatement.SetInt(2,j)  
        oStatement.SetDouble(3,20.0*Rnd-10.0)  
        oStatement.ExecuteUpdate()  
    next j  
next i  
End Sub  
'-----
```

Выполнение запросов на выборку данных

Функция «executeQuery» возвращает объект «выборка». Для перехода на первую и последующие записи выборки следует вызывать функцию «next()». Эта функция заодно возвращает «false» если записей больше нет. Внимание, даже если в выборке ровно одна запись, все равно надо вызвать функцию «next()».

Данные из текущей записи можно получить с помощью функций «getString, getInt, getDouble». В качестве параметра передается индекс колонки (считая с 1).

```
'-----  
Sub OnBtnShowSelRes(oEvent)  
'Выводит в MessageBox список ФИО студентов с зачетками  
oForm = oEvent.Source.getModel().getParent()  
oCon=oForm.ActiveConnection 'объект для доступа к базе данных  
strSQL="SELECT "ФИО","№зачетки" from "студент"  
oStatement=oCon.CreateStatement() 'объект для выполнения запросов  
oResult=oStatement.executeQuery(strSQL) 'выполняем запрос  
s="Номера зачетов:"+Chr(10)  
Do While oResult.next()  
    s=s+oResult.getString(1)+" - "+oResult.getString(2)+Chr(10)  
Loop  
MsgBox(s)  
End Sub  
'-----
```

Запросы на выборку данных так же можно делать с использованием «PrepareStatement»

Вывод результатов запроса в текстовое окно

Как говорилось ранее, для формирования пакета запросов на изменения с помощью SELECT запроса, конструктор запросов плохо подходит.

Создайте форму в режиме дизайна и разместите на ней 2 текстовых поля (с именами «query» и «TextBox» и типом текста «Многострочный») и кнопку «Выполнить». Свяжите с кнопкой следующую программу:

```

'-----
Sub OnBtnShowTextRes(oEvent)
'Выводит в MessageBox список ФИО студентов с зачетками
oForm = oEvent.Source.getModel().getParent()
oCon=oForm.ActiveConnection 'объект для доступа к базе данных
sSQL=oForm.getByName("query").text
oStatement=oCon.createStatement() 'объект для выполнения запросов
oResult=oStatement.executeQuery(sSQL) 'выполняем запрос
s=""
Do While oResult.next()
    s=s+oResult.getString(1)+Chr(10)
Loop
oForm.getByName("TextBox").text=s 'выводим строку в текстовое поле
End Sub
'-----

```

Загрузка данных в массив и вывод списка

Не все можно сделать SQL запросом. При наличии сложных вычислений с малым объемом данных можно использовать массивы и обсчитывать данные в памяти.

```

'-----
Function queryArray(oCon as Object,strSQL as String)
'возвращает массив, состоящий из элементов выборки,
'полученной в результате запроса strSQL
Dim a() As String 'Объявляем массив
Dim n as Integer 'Счетчик
n=0
oStatement=oCon.createStatement() 'объект для выполнения запросов
oResult=oStatement.executeQuery(strSQL) 'выполняем запрос
Do While oResult.next()
    n=n+1
    ReDim Preserve a(1 to n)'увеличиваем размер массива
    a(n)=oResult.getString(1)'заносим в новую ячейку значение
Loop
queryArray=a
End Function
'-----
Sub OnBtnArray(oEvent)
'загружает список групп в массив и выводит его в MessageBox
oForm = oEvent.Source.getModel().getParent()
oCon=oForm.ActiveConnection 'объект для доступа к базе данных
'составим массивы строк и колонок
aGrp()=queryArray(oCon,"SELECT DISTINCT " & "Группа" & " from " & "Ведомость" & "")
oList=oForm.getByName("ListBox") 'список – элемент формы
oList.StringItemList=aGrp 'выводим массив строк в список
End Sub
'-----

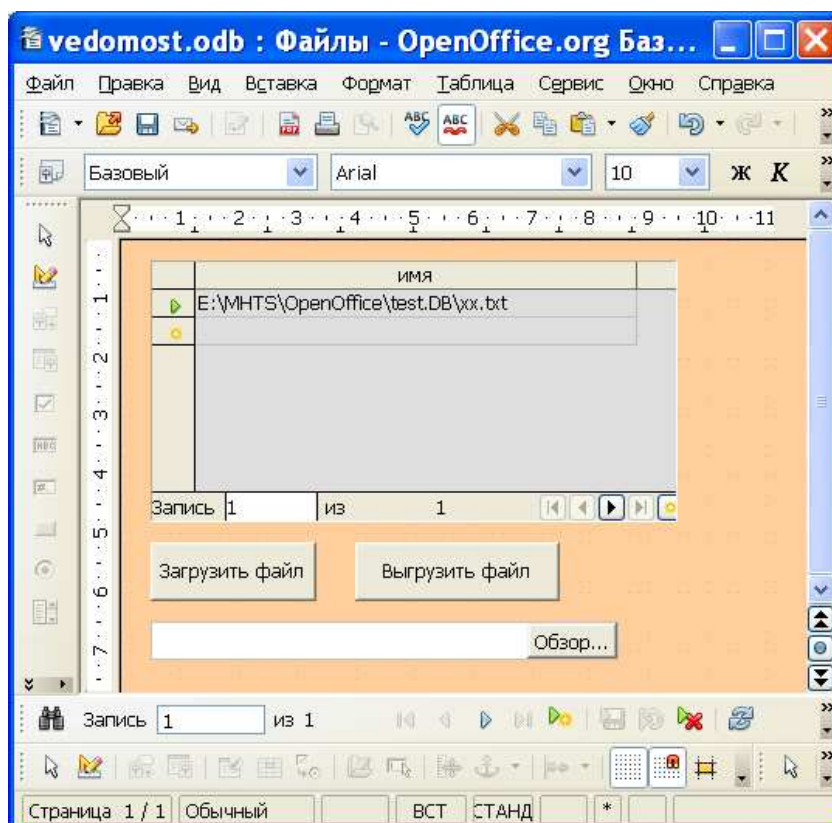
```

Загрузка файла в базу данных и выгрузка файла из базы

Создать таблицу «Файл» с полями:

Код	Целое[INTEGER]	Авто – значение, первичный ключ
Имя	Текст[VARCHAR]	
Данные	Картинка[LONGVARBINARY]	

Создать форму «Файлы» с помощью мастера. Добавить поле типа «Выбор файла» с помощью панели инструментов «Дополнительные элементы управления». Добавить кнопки «Загрузить файл» и «Выгрузить файл» и связать их с приведенными ниже функциями «putFile» и «getFile»



```

Sub putFile(oEvent)
'Загружает в базу файл с именем, заданным в элементе FileSelection
oForm = oEvent.Source.getModel().getParent()
sFileName=oForm.getByName("FileSelection").text
If NOT FileExists(sFileName) Then 'проверим наличие файла
    MsgBox("Не найден заданный файл")
    Exit Sub
End If
oCon=oForm.ActiveConnection 'объект для доступа к базе данных
oSimpleFileAccess = createUnoService("com.sun.star.ucb.SimpleFileAccess")
oStream = oSimpleFileAccess.openFileRead(sFileName)
lLen = oStream.getLength() 'Длина файла
sSQL = "insert into ""Файл""(""Имя"", ""Данные"") values (?, ?)"
oStatement = oCon.prepareStatement(sSQL)
oStatement.SetString( 1, sFileName)
oStatement.setBinaryStream(2, oStream, oStream.getLength())
oStatement.executeUpdate()
oStream.closeInput()
oForm.reload()'обновляем список файлов в форме
End Sub
'-----
Sub getFile(oEvent)
'Сохраняет в файл с именем, заданным в элементе FileSelection
'бинарные данные из текущей записи
Dim oData() 'Объявляем массив
oForm = oEvent.Source.getModel().getParent()
oForm.updateRow() 'сохраняем изменения текущей записи в базу данных
oCon=oForm.ActiveConnection 'объект для доступа к базе данных
idFile=oForm.getInt(oForm.findColumn("Код")) 'из текущей записи
sFile=oForm.getByName("FileSelection").text 'из элемента FileSelection

sSQL = "SELECT ""Данные"" FROM ""Файл"" WHERE ""Код""="&idFile
oStatement=oCon.createStatement() 'объект для выполнения запросов
oResult = oStatement.executeQuery(sSQL) 'выполняем запрос
oResult.next()' считываем запись
oStream = oResult.getBinaryStream(1)

oSimpleFileAccess = createUnoService("com.sun.star.ucb.SimpleFileAccess")
oSimpleFileAccess.writeFile(sFile, oStream)

```

End Sub

'-----

Приложение 1 - Встроенные функции и процедуры Hsqldb

Функции для работы с числами

ABS(d) – Возвращает значение типа Double, равное абсолютному значению d.
ACOS(d) - Возвращает значение типа Double, содержащее аркосинус d.
ASIN(d) - Возвращает значение типа Double, содержащее арксинус d.
ATAN(d) - Возвращает значение типа Double, содержащее арктангенс d.
ATAN2(a,b) - Возвращает значение типа Double, содержащее арктангенс a/b.
BITAND(a,b) – Возвращает побитовое AND чисел a и b
BITOR(a,b) – Возвращает побитовое OR чисел a и b
CEILING(d) – возвращает наименьшее целое, которое не меньше d (округление в меньшую сторону)
COS(d) Возвращает значение типа Double, содержащее косинус d.
COT(d) - Возвращает значение типа Double, содержащее котангенс d.
DEGREES(d) – переводит радианы в градусы
EXP(d) – Возвращает $e(2.718...)$ в степени d
FLOOR(d) – Возвращает наибольшее целое, которое не больше d (округление в большую сторону)
LOG(d) – Возвращает натуральный логарифм (base e)
LOG10(d) – Возвращает десятичный логарифм (base 10)
MOD(a,b) – Возвращает остаток от деления a/b
PI() - Возвращает π (3.1415...)
POWER(a,b) – Возвращает a в степени b
RADIANS(d) – переводит градусы в радианы
RAND() возвращает случайное число больше или равно 0.0 и меньше 1.0
ROUND(a,b) – Округляет a до и знаков после запятой
ROUNDMAGIC(d) solves rounding problems such as 3.11-3.1-0.01
SIGN(d) – Возвращает -1(если d<0), 0(если d==0) , +1(если d>0)
SIN(d) – Возвращает синус d
SQRT(d) – Возвращает значение типа Double, содержащее квадратный корень указанного числа.
TAN(A) - Возвращает значение типа Double, содержащее тангенс a.
TRUNCATE(a,b) – Округляет число a, путем отбрасывания десятичных знаков, после запятой (после b)

Функции для работы со строками

ASCII(s) – возвращает ASCII код левого символа строки s
BIT_LENGTH(str) - возвращает длину строки s в битах
CHAR(c) – возвращает символ, соответствующий ASCII – коду c
CHAR_LENGTH(str) - возвращает длину строки s в символах
CONCAT(str1,str2) – Добавляет строку str2 в конец строки str1
DIFFERENCE(s1,s2) - returns the difference between the sound of s1 and s2
HEXTORAW(s1) - returns translated string
INSERT(s,start,len,s2)
returns a string where len number of characters beginning at start has been replaced by s2
LCASE(s) – преобразует символы строки s к нижнему регистру. Возвращает полученную строку.
LEFT(s,count) – возвращает count первых символов строки s - requires double quoting - use
LENGTH(s) – возвращает длину строки s в символах
LOCATE(search,s,[start]) – возвращает позицию(считая с 1) первого вхождения подстроки search в строку s, начиная с start, или 0, если подстрока не найдена
LTRIM(s) – Удаляет все пробелы в начале строки s. Возвращает полученную строку.
OCTET_LENGTH(str) Возвращает длину строки в байтах (удвоенное число символов).
RAWTOHEX(s1) – Возвращает строку, содержащую коды символов строки s1, в шестнадцатеричной системе счисления
REPEAT(s,count) – возвращает строки s, повторенную count раз

REPLACE(s, old , new) – заменяет все вхождения подстроки old в строку s на new. Возвращает полученную строку.

RIGHT(s,count) – возвращает count последних символов строки s

RTRIM(s) - Удаляет все пробелы в конце строки s. Возвращает полученную строку.

SOUNDEX(s) returns a four character code representing the sound of s

SPACE(count) – возвращает строку, содержащую count пробелов

SUBSTR(s,start[,len]) – то-же, что SUBSTRING()

SUBSTRING(s,start[,len]) – Возвращает подстроку, строки s, начиная с символа с индексом start (счет с 1), и длиной len символов.

UCASE(s) – преобразует символы строки s к верхнему регистру. Возвращает полученную строку.

LOWER(s) – преобразует символы строки s к нижнему регистру. Возвращает полученную строку.

UPPER(s) – преобразует символы строки s к верхнему регистру. Возвращает полученную строку.

Функции для работы с датой/временем

CURDATE() – возвращает текущую дату
returns the current date

CURTIME()– возвращает текущее время

DATEDIFF(type, datetime1, datetime2) – возвращает число единиц времени, между datetime1 и datetime2. Единица времени задается параметром type: 'ms'='millisecond', 'ss'='second', 'mi'='minute', 'hh'='hour', 'dd'='day', 'mm'='month', 'yy' = 'year'. И короткая и длинная форма может быть использована.

DAYNAME(date) – возвращает наименование дня недели (английское)

DAYOFMONTH(date) – возвращает день месяца (1-31) для даты date

DAYOFWEEK(date) – возвращает номер дня недели (1-воскресенье)

DAYOFYEAR(date) – возвращает день года (1-366) для даты date

HOUR(time) – возвращает час (0-23)

MINUTE(time) – возвращает минуту (0-59)

MONTH(date) – возвращает номер месяца (1-12)

MONTHNAME(date) – возвращает наименование(английское) месяца (1-12)

NOW() - возвращает текущую дату и время

QUARTER(date) – возвращает номер квартала(1-4) для даты date

SECOND(time) – возвращает секунду(0-59)

WEEK(date) – Возвращает номер недели(1-53) года

YEAR(date) – возвращает год для даты date

CURRENT_DATE– возвращает текущую дату

CURRENT_TIME– возвращает текущее время

CURRENT_TIMESTAMP - возвращает текущую дату и время

Функции информации о подключении к базе данных

DATABASE() – возвращает имя базы данных, к которой подключен пользователь

USER() - возвращает имя пользователя, подключившегося к базе данных

CURRENT_USER - возвращает имя пользователя, подключившегося к базе данных

IDENTITY() – возвращает значение поля типа счетчик (автозначение) из последней добавленной записи в данном сеансе работы.

Прочие функции

IFNULL(exp,value) – Если результатом вычисления выражения «exp» является NULL, возвращает «value», иначе возвращает результат «exp».

COALESCE(expr1,expr2,expr3,...) – Если значение «expr1» не NULL, возвращает его, иначе, вычисляет выражение «expr2», и если результат не NULL, возвращает его. И так далее.

NULLIF(v1,v2) – Если «v1» равно «v2» - возвращает NULL, иначе возвращает «v1»

CASEWHEN(exp,v1,v2) – Если результат выражения «exp» истина, возвращает «v1» , иначе возвращает «v2»

CONVERT(exp,type) – преобразует значение выражения «exp» к типу «type». Например,
CONVERT(' 2010-12-31 ' ,DATE) , CONVERT(YEAR("d1 ") | ' -01-01 ' ,DATE)

CAST(exp AS type) – преобразует значение выражения «exp» к типу «type»

CASE v1 WHEN v2 THEN v3 END – Когда «v1» равно «v2», возвращает «v3» иначе NULL

CASE v1 WHEN v2 THEN v3 ELSE v4 END – Когда «v1» равно «v2», возвращает «v3» иначе «v4»

CASE WHEN expr1 THEN v1[WHEN expr2 THEN v2] [ELSE v4] END – Когда значение выражения «expr1» - TRUE возвращает «v1», иначе если значение выражения «expr2» - TRUE возвращает «v2» (таких блоков может быть несколько или не быть вообще), иначе возвращает «v4» (или NULL, если блок ELSE опущен)

EXTRACT ({ YEAR | MONTH | DAY | HOUR | MINUTE | SECOND } FROM dt) возвращает год или месяц или день, или час или минуту или секунду для значения даты/времени «dt»

POSITION(substring IN s) Возвращает позицию подстроки «substring» в строке «s» или 0 (если не такой подстроки нет)

SUBSTRING(s FROM start [FOR len]) Возвращает подстроку, строки s, начиная с символа с индексом start (счет с 1), и длиной len символов.

TRIM(LENDING ... FROM ...)[2]

TRIM(LEADING FROM s) Удаляет все пробелы в начале строки s и возвращает полученную строку.

TRIM(TRAILING FROM s) Удаляет все пробелы в конце строки s и возвращает полученную строку.

TRIM(BOTH FROM s) Удаляет все пробелы в начале и конце строки s и возвращает полученную строку.

Литература

1. Справочная система OpenOffice
2. Руководство по Hsqldb <http://www.hsqldb.org/doc/guide/index.html>
3. Using Macros With OOo Base By Andrew Pitonyak
<http://www.pitonyak.org/database/AndrewBase.pdf>