Создание сложных запросов

ГЛАВА

18

В этой главе...

В данной главе рассматривается работа со сложными запросами. До сих пор мы обсуждали главным образом сравнительно простые запросы на выборку, извлекающие данные из одной или нескольких таблиц на основе заданных критериев. Коротко были рассмотрены запросы на изменение, позволяющие модифицировать значения полей таблицы, а также добавлять и удалять записи. В настоящей главе будут рассмотрены обобщающие (итоговые) и перекрестные запросы, выполняющие сложные вычисления, и запросы на обновление, изменяющие структуру таблицы.

Дополнительная информация

В примерах главы используется база данных `Chapter18.accdb`. Если вы еще не скопировали ее на жесткий диск с компакт-диска, сделайте это сейчас. База данных `Chapter18.accdb` создана путем импорта данных из таблиц Oracle, поэтому имена полей и таблиц в ней состоят из букв верхнего регистра. Импорт данных рассматривается в главе 17.

Вычисляемые поля

В результирующий набор запроса можно выводить не только поля таблицы, но и поля, содержащие вычисляемые значения. Вычисляемый столбец (т.е. столбец, содержащий вычисляемые поля) можно создать многими способами, включая следующие.

- Конкатенация двух или большего количества полей текстового типа с помощью символов амперсанта (`&`).
- Вычисление числового значения с помощью математической формулы.
- Использование встроенных и пользовательских функций.

Создайте вычисляемый столбец `DiscountPrice` (Цена после скидки) на основе значений столбца `LIST_PRICE` (Цена в каталоге), хранящихся в таблице `BOOKS_EDITION`. Для этого выполните следующие действия.

1. **Откройте базу данных `Chapter18.accdb`.**
2. **Активизируйте вкладку Создание (Create). В разделе Другие (Other) щелкните на кнопке Мастер запросов (Query Wizard).**
3. **Выберите поля `ISBN` и `LIST_PRICE` таблицы `BOOKS_EDITION`. Щелкните на кнопке Далее (Next).**
4. **Установите переключатель подробный (Detail) и щелкните на кнопке Далее.**
5. **Установите переключатель Изменить макет запроса (Modify Query Layout) и щелкните на кнопке Готово (Finish).**
6. **В решетке запроса щелкните в новом поле и введите выражение `DiscountPrice:LIST_PRICE*0.75`[1].**

   В новом столбце будет выведена цена с учетом скидки (т.е. цена, уменьшенная на 25%).

На заметку

Вводить имя таблицы перед именем каждого поля не обязательно, поскольку в запросе используется только одна таблица. Однако для большей наглядности структуры запроса рекомендуется вводить имена таблиц, поэтому введите выражение `DiscountPrice:BOOKS_EDITION.LIST_PRICE*0.75`.

7. **На ленте щелкните на кнопке Выполнить (Run).**

   Результат выполнения запроса показан на рис. 18.1.

Рис. 18.1. Запрос вывел вычисляемый столбец `DiscountPrice`

[1] *Если в окне региональных параметров Windows в качестве десятичного разделителя установлена запятая, нужно ввести* `...*0,75`. — Примеч. ред.

В рабочую среду Access встроен построитель выражений, облегчающий создание сложных выражений для форм, отчетов и вычисляемых столбцов запросов. Чтобы применить построитель выражений, выполните следующие действия.

1. **Переключите запрос, созданный в предыдущем упражнении (см. рис. 18.1), в режим конструктора.**
2. **Откройте окно построителя выражений. Для этого щелкните правой кнопкой мыши в пустой верхней ячейке решетки справа от выражения `DiscountPrice`.**
3. **Выберите в контекстном меню команду Построить (Build).**

   Откроется диалоговое окно Построитель выражений (Expression Builder).
4. **На левой панели построителя разверните узел `Функции` (Functions). Щелкните в узле `Встроенные функции` (Built-In Functions).**
5. **На правой панели построителя найдите функцию `IIf` и дважды щелкните на ее имени. Функция появится на верхней панели.**

На заметку Встроенная условная функция `IIf` принимает три параметра: `IIf(`*условие*`,` *выражение1*`,` *выражение2*`)`. Если условие равно `True`, выполняется формула *выражение1*, в противном случае выполняется *выражение2*. Функция возвращает результат выполненного выражения.

6. **С помощью функций `IIf` и `IsNull` создайте выражение, показанное на рис. 18.2. Щелкните на кнопке OK. Выражение будет вставлено в ячейку решетки.**

Рис. 18.2. Использование построителя выражений для создания вычисляемого столбца[2]

На заметку Встроенная функция `IsNull` выясняет, содержит ли поле или переменная значение. Если поле пустое (т.е. содержит `Null`), функция `IsNull` возвращает значение `True`, в противном случае — `False`. Обычно в базах данных функция `IsNull` используется для устранения пустых полей. Если в любом месте выра-

[2] *Обратите внимание на то, что в окне построителя используются разделители, установленные в окне региональных стандартов русифицированной версии Windows. В качестве разделителя элементов списка используется точка с запятой, а в качестве десятичного разделителя — запятая. В кодах SQL и VBA они будут автоматически заменены запятой (разделитель элементов списка) и точкой (десятичный разделитель).* — Примеч. ред.

жения встречается значение Null, результат тоже равен Null, независимо от других значений, входящих в выражение. Для устранения этого эффекта нужно проверять входные значения выражений с помощью функции IsNull.

7. **На ленте выберите команду Вид⇨Режим SQL (View⇨SQL View).**

   Запрос будет выведен в режиме SQL. Оператор SQL должен иметь следующий вид.

```
SELECT BOOKS_EDITION.[ISBN],
   BOOKS_EDITION.[LIST_PRICE],
   [BOOKS_EDITION].[LIST_PRICE]*0.75 AS DiscountPrice,
   IIf(IsNull([PAGES]),[LIST_PRICE]*0.75,
            [LIST_PRICE]*0.75*([PAGES]/500)) AS Выр1
FROM BOOKS_EDITION;
```

   Если поле PAGES содержит значение Null, то использовать его в выражении нельзя. В этом случае нужно взять значение из таблицы BOOKS_EDITION. Чтобы визуально проконтролировать значения PAGES, переключите запрос в режим конструктора и добавьте в результирующий набор запроса столбец PAGES.

8. **Переключите запрос в режим таблицы.**

   Результат выполнения запроса показан на рис. 18.3.

Рис. 18.3. Результат выполнения запроса

Выражение, вставленное в вычисляемый столбец, может содержать поля не только одной, но и многих таблиц, в том числе связанных. Используя связанные таблицы, можно создать столбец, вычисляемый на основе значений, хранящихся в других базах данных.

Вычисление количества записей в таблице или запросе

Для выяснения общего количества записей в существующей таблице или результирующем наборе запроса используется встроенная функция `Count(*)`. Звездочка — это специальный параметр функции. Чтобы выяснить общее количество записей, хранящихся в таблице `BOOKS_EDITION`, выполните следующие действия.

1. **Создайте новый запрос на основе таблицы `BOOKS_EDITION`.**
2. **Переключите запрос в режим конструктора. Щелкните в ячейке Поле (Field) первого столбца.**
3. **Введите в ячейке выражение `Count(*)`. Переключите запрос в режим таблицы.**

   Запрос должен возвратить значение `17`, поскольку таблица `BOOKS_EDITION` содержит 17 записей (если, экспериментируя с таблицей, вы не изменили количество записей).

Функцию `Count` можно использовать для подсчета количества записей, удовлетворяющих заданному критерию. Предположим, что нужно подсчитать, сколько книг, упомянутых в таблице `BOOKS_EDITION`, имеют количество страниц (`PAGES`) больше 300. Для этого выполните следующие действия.

1. **Создайте новый запрос на основе таблицы `BOOKS_EDITION`.**
2. **В режиме конструктора щелкните в пустой ячейке Поле.**
3. **Введите в ячейку выражение `Count(*)`.**
4. **На верхней панели конструктора дважды щелкните на имени поля `PAGES`, чтобы добавить его в запрос.**
5. **В столбце `PAGES` снимите флажок Вывод на экран (Show).**

   В результирующий набор запроса должен быть выведен только столбец, содержащий функцию `Count(*)`, поскольку запрос выводит обобщенное значение, а не индивидуальные записи. Если попытаться вывести еще какой-либо столбец, Access возвратит сообщение об ошибке.
6. **В ячейку Условие отбора (Criteria) столбца `PAGES` введите выражение `>300`.**
7. **Вместо слова `Выражение1` введите фразу `Количество книг объемом более 300 страниц`.**
8. **Выполните запрос.**

   Запрос должен вернуть число `10`. Вручную подсчитайте в таблице `BOOKS_EDITION` количество книг, содержащих более 300 страниц, чтобы убедиться в том, что запрос выполнен правильно.

Вывод первых N записей

В рабочей среде Access можно создать запрос, находящий по заданному критерию и возвращающий первые N записей. Кроме того, запрос может возвращать заданный процент первых записей, например первые 5% записей.

Предположим, что в таблице BOOKS_EDITION нужно отсортировать книги по названиям и вывести первые 10 книг (названия книг приведены в столбце TITLE таблицы BOOKS_PUBLICATION). Для решения этой задачи выполните следующие действия.

1. **Создайте новый запрос на основе таблиц BOOKS_EDITION (Издание книг) и BOOKS_PUBLICATION (Публикации книг).**

   Эти две таблицы должны быть объединены на основе значений столбца PUBLICATION_ID (Идентификационный код публикации). Обратите внимание на использование термина "публикация" в английском языке и, соответственно, в именах столбцов. Публикацией считается одно название. Одной публикации могут соответствовать несколько книг, изданных разными предприятиями на разных носителях (например, в твердой обложке, в мягкой обложке, на компакт-диске).

2. **В режиме конструктора установите отношение между таблицами на основе значений PUBLICATION_ID. Для этого наведите указатель на имя поля PUBLICATION_ID в одной таблице и перетащите его на это же поле в другой таблице.**

3. **Щелкните правой кнопкой мыши на линии, обозначающей отношение.**

4. **Выберите в контекстном меню команду Параметры объединения (Join Properties).**

   Откроется диалоговое окно Параметры объединения (рис. 18.4).

Рис. 18.4. Проверка объединения двух таблиц

   На рис. 18.4 видно, что две таблицы объединены на основе общего поля PUBLICATION_ID. Щелкнув на кнопке Режим SQL (SQL View), выведите код SQL запроса, приведенный ниже. Полужирным шрифтом выделены параметры объединения.

```
SELECT FROM BOOKS_EDITION
 INNER JOIN BOOKS_PUBLICATION
 ON BOOKS_EDITION.PUBLICATION_ID =
       BOOKS_PUBLICATION.PUBLICATION_ID;
```

5. **Добавьте в запрос столбцы TITLE, ISBN, PRINT_DATE и FORMAT. Задайте сортировку записей по возрастанию на основе значений столбца TITLE.**

6. **Щелкните на кнопке Режим таблицы (Table View), чтобы увидеть результирующий набор запроса. Как видите, запрос возвращает 17 записей.**

7. **Переключите запрос в режим конструктора.**

8. **Задайте вывод первых 10 записей. Для этого наведите указатель на раскрывающийся список Возврат (Return), расположенный в разделе Настройка запроса (Query Setup) ленты. Через секунду появится всплывающая подсказка, сооб-**

щающая о назначении раскрывающегося списка. Дважды щелкните на списке и вместо слова Все (All) введите значение `10`.

9. **Если щелкнуть на стрелочке, будет выведен список значений, включая значения в процентах. Можете выбрать один из элементов списка или вручную ввести нужное значение. Выполните запрос.**

   Запрос возвратит первые 10 записей из 17.

Если в раскрывающемся списке Возврат выбрать значение `5`, запрос все равно возвратит 10 записей. Это объясняется тем, что запрос возвращает первые 5 уникальных записей плюс все дублированные.

Как запрос сохраняет выбранные поля

При открытии запроса в режиме конструктора обратите внимание на то, что со времени последнего сохранения запроса его конструкция в некоторых случаях автоматически изменяется. Это объясняется тем, что при сохранении запроса Access переупорядочивает (иногда даже удаляет) поля на основе следующих правил.

- Если флажок Вывод на экран (Show) данного поля снят, а условие отбора задано, Access перемещает поле в последний правый столбец решетки.
- Если флажок Вывод на экран снят и поле не имеет условия отбора или директивы сортировки, Access удаляет поле из решетки запроса.
- После создания обобщенного выражения с оператором `Sum` в обобщающем запросе Access заменяет его выражением с функцией `Sum`.

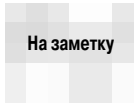
Следовательно, после сохранения и повторного открытия запроса его конструкция может выглядеть немного иначе, чем перед выполнением этих операций. В следующих разделах этот вопрос рассматривается подробнее.

Сокрытие полей

В некоторых случаях нужно отменить вывод одного или нескольких полей в результирующий набор запроса. Ранее в главе рассматривался запрос, возвращающий количество книг объемом более 300 страниц. Запрос возвращает одно число (количество книг), а не данные по каждой книге. Поэтому запрос должен вывести только столбец, содержащий обобщенное значение, остальные столбцы должны быть скрыты.

Чтобы скрыть или удалить столбец из результирующего набора, нужно снять флажок Вывод на экран, принадлежащий данному столбцу в решетке запроса (рис. 18.5).

Если с помощью флажка Вывод на экран скрыть столбец, не используемый в критериях сортировки или фильтрации, Access автоматически удалит его при сохранении запроса. После этого удаленный столбец можно будет использовать в запросе, только вновь добавив его в решетку путем перетаскивания с верхней панели.



Если запрос используется в отчете или форме, он должен вернуть все столбцы, связанные с элементами управления, включая скрытые столбцы.

Рис. 18.5. Чтобы скрыть столбец PAGES*, нужно снять его флажок* Вывод на экран

Переименование полей запроса

Переименование полей результирующего набора запроса используется для того, чтобы их назначение было более очевидным. Новое имя отображается при выводе результирующего набора запроса в режиме таблицы. В предыдущих разделах вы уже встречались с переименованием выражения, лежащего в основе вычисляемого столбца. Переименование обычных полей выполняется аналогичным образом. Поле получает новое имя только в результирующем наборе запроса, но не в таблице.

Рассмотрим запрос, возвращающий первые 10 записей. Этот запрос вы создали, выполняя упражнение ранее в главе (см. рис. 18.4). Откройте его в режиме конструктора. В решетке перед именем столбца PRINT_DATE введите фразу **Дата публикации:**. Тогда в результирующем наборе запроса эта фраза будет выведена в заголовке столбца вместо мало информативного имени PRINT_DATE. Код SQL запроса имеет следующий вид (полужирным шрифтом выделена директива переименования).

```
SELECT TOP 5
   BOOKS_PUBLICATION.TITLE,
   BOOKS_EDITION.ISBN,
   BOOKS_EDITION.PRINT_DATE AS [Дата публикации],
   BOOKS_EDITION.FORMAT
 FROM BOOKS_EDITION
 INNER JOIN BOOKS_PUBLICATION
 ON BOOKS_EDITION.PUBLICATION_ID =
      BOOKS_PUBLICATION.PUBLICATION_ID
 ORDER BY BOOKS_PUBLICATION.TITLE;
```

В коде SQL переименование выполняется директивой AS, размещенной после имени столбца таблицы.

На заметку Заголовок, заданный для поля таблицы в режиме конструктора, используется в запросе в качестве имени поля таблицы.

После переименования поля в качестве заголовка столбца, расположенного в результирующем наборе запроса, используется только новое имя. Это же имя используется и в любом элементе управления формы или отчета, источником для которого служит запрос. В каждой новой форме или отчете, которые созданы на основе запроса, используется новое имя поля.

На заметку Изменение имени поля в запросе не затрагивает имя поля в нижележащей таблице, однако во всех последующих процессах, использующих запрос, применяется измененное имя поля.

Имя выражения или поля запроса (т.е. новое имя, после которого стоит двоеточие) можно использовать в других выражениях, входящих в этот же запрос. Предположим, что запрос содержит вычисляемый столбец `FullName`, в котором имя и фамилия сотрудника объединяются с помощью встроенной функции. Тогда имя выражения `FullName` можно использовать в других ячейках решетки.

На заметку Имя выражения нельзя использовать в критериях, ссылающихся на поля, используемые выражением. Иными словами, в запросе не должно быть цикличного использования имен.

Сокрытие и вывод столбца в режиме конструктора

Как уже говорилось, скрыть столбец запроса можно в режиме конструктора, сняв флажок Вывод на экран. Кроме того, скрыть столбец можно также в режиме таблицы путем перетаскивания правой границы столбца. Если перетащить ее влево до совпадения с левой границей, то ширина столбца станет равной нулю и он будет невидим.

Свойства запроса

Существует несколько способов установки свойств запроса при его создании. Откройте запрос в режиме конструктора, активизируйте на ленте вкладку Конструктор (Design) и щелкните на кнопке Страница свойств (Properties), расположенной в разделе Показать и скрыть (Show and Hide). Или установите курсор в решетку запроса, щелкните правой кнопкой мыши в пустой ячейке решетки и выберите в контекстном меню команду Свойства (Properties). В обоих случаях будет открыто окно свойств запроса.

Набор свойств на уровне запроса зависит от его типа и от таблиц и полей, используемых в запросе. В табл. 18.1 приведено описание свойств. Крестиками отмечены свойства, которыми можно манипулировать в запросе данного типа.

Таблица 18.1. Свойства запросов

Свойство	Описание	Тип запроса					
		На выборку	Перекрестный	На обновление	На удаление	На создание таблицы	На добавление
Описание (Description)	Текст, описывающий таблицу или запрос	X	X	X	X	X	X
Режим по умолчанию (Default View)	Позволяет задать режим, в котором будет запускаться запрос по умолчанию. Допустимые значения: Режим таблицы (Datasheet), Сводная таблица (Pivot Table) и Сводная диаграмма (Pivot Chart)	X	X				
Вывод всех полей (Output All Fields)	Результирующий набор запроса содержит все поля	X				X	X
Набор значений (Top Values)	Количество (или процентная часть от общего количества) выводимых записей	X				X	X
Уникальные значения (Unique Values)	Выводить только уникальные значения полей в результирующей таблице	X				X	X
Уникальные записи (Unique Records)	Выводить только уникальные записи в результирующей таблице	X		X	X	X	X
При запуске предоставляются права (Run permissions)	Позволяет задать права владельца для любого пользователя	X	X	X	X	X	X
База данных-источник (Source Database)	Имя внешней базы данных для всех таблиц в запросе	X	X	X	X	X	X

Продолжение табл. 18.1

Свойство	Описание	Тип запроса					
		На выборку	Перекрестный	На обновление	На удаление	На создание таблицы	На добавление
Строка подключения-источник (Source Connect Str)	Имя приложения, использованного для подключения к внешней базе данных	X	X	X	X	X	X
Блокировка записей (Record Locks)	Блокировка записей во время выполнения запроса (обычно запроса на изменение)	X	X	X	X	X	X
Тип набора записей (Recordset Type)	Определяет возможность редактирования таблиц, использованных в запросе. Возможные значения: Динамический набор (Dynaset), Динамический набор (несогл.) (Dynaset (inconsistent updates)) или Статический набор (Snapshot)	X	X				
Время ожидания ODBC (ODBC Timeout)	Количество секунд ожидания перед выводом сообщения об ошибке открытия базы данных	X	X	X	X	X	X
Фильтр (Filter)	Имя фильтра, автоматически загружаемого вместе с запросом	X					
Порядок сортировки (Order By)	Способ сортировки записей	X					
Максимальное число записей (Max Records)	Максимальное количество записей, возвращаемых базой данных ODBC	X					
Ориентация (Orientation)	Позволяет задать отображение полей в результатах запроса слева направо или справа налево	X	X	X	X	X	X
Имя подтаблицы (Subdatasheet Name)	Идентифицирует подчиненный запрос	X	X	X		X	X

Окончание табл. 18.1

Свойство	Описание	Тип запроса					
		На выборку	Перекрестный	На обновление	На удаление	На создание таблицы	На добавление
Подчиненные поля (Link Child Fields)	Имя поля (имена полей) в подчиненном запросе	X	X	X		X	X
Основные поля (Link Master Fields)	Имя поля (имена полей) в главной таблице	X	X	X		X	X
Высота подтаблицы (Subdatasheet Height)	Максимальная высота подтаблицы	X	X	X		X	X
Развернутая подтаблица (Subdatasheet Expanded)	Определяет, появляются ли подтаблицы в развернутом виде при просмотре данных в режиме таблицы	X	X	X		X	X
Заголовки столбцов (Column Headings)	В этой строке устанавливаются заголовки столбцов		X				
Использовать транзакцию (Use Transaction)	Определяет, выполняется ли запрос на изменение как одна транзакция			X	X	X	X
Останов при ошибке (Fail on Error)	Останавливает операцию при возникновении ошибки			X	X		
Таблица-получатель (Destination Table)	Имя таблицы, в которую будет помещен результат запроса					X	X
База данных- получатель (Destination DB)	Имя базы данных, в которую будет помещен результат запроса					X	X
Строка подключения-получатель (Dest Connect Str)	Строка подключения базы данных, содержащая параметры соединения					X	X

Обобщающие запросы

Во многих случаях информацию в таблицах нужно найти на основе обобщений заданного столбца или столбцов. Например, иногда нужно выяснить общее количество клиентов, являющихся одновременно покупателями и продавцами, или общее количество денег, потраченных всеми покупателями автомобилей на протяжении последнего года. Рабочая среда Access содержит инструменты решения этих задач без помощи кода.

Обобщенные значения вычисляются с помощью обобщающих функций, позволяющих найти заданное значение на основе содержимого столбца. Например, с помощью обобщающих функций можно вычислить среднюю цену автомобиля заданного типа, максимальную или минимальную цену автомобиля, общее количество записей, в которых клиент является покупателем или продавцом (или тем и другим одновременно).

Дополнительная информация

Ранее в главе уже рассматривалось использование функции `Count(*)` для подсчета записей. Функция `Count(*)` является обобщающей.

Для создания обобщающих запросов (другое их название — *итоговые*) используется специализированный мастер. Кроме того, обобщенные (итоговые) значения можно создавать в режиме конструктора с помощью кнопки Итоги (Totals), на которой изображен символ суммирования.

Мастер обобщающих запросов

Одно из окон мастера обобщающих запросов предоставляет возможность выбрать тип обобщения, например суммирование, подсчет, поиск минимального или максимального значений и т.д. В детализирующем отчете выводится каждая запись, а в обобщающем — обобщенное значение. Откройте запрос, показанный на рис. 18.3, в режиме конструктора. Во вкладке Создание (Create) щелкните на кнопке Мастер запросов (Query Wizard). Добавьте в новый запрос все поля запроса, показанного на рис. 18.3. Установите флажок итоговый (Total) и щелкните на кнопке Итоги (Totals). Будет открыто окно, в котором можно задать параметры обобщения (рис. 18.6).

Рис. 18.6. Задание параметров обобщающего запроса

Что такое обобщающая функция?

Термин "обобщающая" означает, что информация собирается из многих мест и концентрируется в одном значении. Обобщающая (aggregate) функция извлекает из таблицы много записей, выполняет над ними математические вычисления и возвращает обобщенное значение. Результатом ее работы может быть усредненное значение, количество записей, максимальное значение и т.д.

Обобщающие запросы в режиме конструктора

Чтобы создать запрос, выполняющий обобщающие вычисления, создайте запрос на выборку и щелкните на кнопке Итоги, расположенной во вкладке Конструктор (Design).

В решетке запроса появится новая строка Групповая операция (Group By). Выберите поля, по значениям которых будет выполняться обобщение, и скройте поля, которые не должны выводиться в результирующий набор.

Создайте запрос на выборку на основе таблицы BOOKS_EDITION. Добавьте в него поля FORMAT (Формат книги) и PAGES (Количество страниц). Щелкните на кнопке Итоги. В строке Групповая операция столбца PAGES выберите значение Sum. Запрос сгруппирует книги по форматам (твердая обложка, мягкая обложка и т.д.) и для каждой группы вычислит общее количество страниц. Код SQL запроса приведен ниже.

```
SELECT BOOKS_EDITION.FORMAT,
       Sum(BOOKS_EDITION.PAGES) AS [Сумма страниц]
  FROM BOOKS_EDITION
  GROUP BY BOOKS_EDITION.FORMAT;
```

Раскрывающийся список ячейки Групповая операция содержит двенадцать элементов, принадлежащих четырем категориям (табл. 18.2). В категорию обобщающих функций входит девять элементов списка, в другие категории — по одному элементу.

Таблица 18.2. Категории групповых операций

Категория	Количество элементов списка	Описание
Группировка (Group By)	1	Группировка записей, имеющих одинаковые значения в данном столбце; для каждой группы запрос вычисляет одно обобщенное значение, следовательно, количество записей в результирующем наборе будет равно количеству групп (т.е. количеству разных значений группирующего столбца)
Обобщающие функции	9	Выполнение обобщающих математических операций над значениями полей, принадлежащих данной группе
Выражение (Expression)	1	Вычисление произвольного обобщенного значения для каждой группы
Условие (Where)	1	Фильтрация записей перед группировкой и вычислением обобщающих значений

Категория группировки

Столбец, в ячейке которого расположено значение Группировка, используется для группирования записей. Например, если значение Группировка расположено в столбце FORMAT, то каждая группа будет состоять из записей, в которых поле FORMAT имеет одно и то же значение.

Категория выражения

Значение Выражение, расположенное в ячейке Групповая операция, сообщает Access о том, что ячейка Поле содержит произвольное обобщающее выражение, вычисляемое для каждой группы.

Категория условия

Значение Условие, расположенное в ячейке Групповая операция, сообщает Access о том, что в столбце решетки задан критерий, ограничивающий количество извлекаемых и обрабатываемых записей. Фактически критерий является фильтром записей. Записи проверяются на соответствие критерию до выполнения любых групповых операций.

Категория обобщающих функций

Эта категория состоит из девяти обобщающих функций: Sum, Avg, Min, Max, Count, StDev, Var, First, Last. Каждая функция (табл. 18.3) выполняет операции над значениями столбца, принадлежащими одной группе, и выводит вычисленное обобщенное значение в результирующий набор запроса. Таким образом, в каждой группе обобщающая функция получает много значений, а возвращает одно. Каждое возвращенное значение принадлежит одной группе.

Таблица 18.3. Обобщающие функции

Функция	Возвращаемое значение	Поддерживаемые типы столбцов
Count	Количество записей, удовлетворяющих всем заданным критериям и не содержащих значения Null в данном столбце	Счетчик, Числовой, Денежный, Дата/время, Логический, Текстовый, Поле MEMO, Поле объекта OLE
Sum	Общая сумма полей	Счетчик, Числовой, Денежный, Дата/время, Логический
Avg	Среднее арифметическое значений полей	Счетчик, Числовой, Денежный, Дата/время, Логический
Max	Максимальное значение в группе	Счетчик, Числовой, Денежный, Дата/время, Логический, Текстовый
Min	Минимальное значение в группе	Счетчик, Числовой, Денежный, Дата/время, Логический, Текстовый
StDev	Стандартное (среднеквадратичное) отклонение значений	Счетчик, Числовой, Денежный, Дата/время, Логический
Var	Дисперсия значений	Счетчик, Числовой, Денежный, Дата/время, Логический
First	Значение поля первой записи	Счетчик, Числовой, Денежный, Дата/время, Логический, Текстовый, Поле MEMO, Поле объекта OLE
Last	Значение поля последней записи	Счетчик, Числовой, Денежный, Дата/время, Логический, Текстовый, Поле MEMO, Поле объекта OLE

Например, с помощью обобщающих функций можно вычислить и вывести в результирующий набор запроса максимальное, минимальное и среднее количество страниц для каждого значения FORMAT таблицы BOOKS_EDITION.

Категории Группировка, Выражение и Условие можно использовать для столбца любого типа, например, для столбцов Поле MEMO, Текстовый, Логический. Однако некоторые обобщающие функции можно использовать не со всеми типами столбцов. Например, нельзя суммировать текстовые данные или найти максимальное значение объектов OLE.

Глобальные обобщения

Глобальная обобщающая функция использует данные всех записей, возвращая одну запись. Пример использования глобальной обобщающей функции Count(*) рассмотрен ранее в главе.

Частные обобщения

Вычислять обобщенные значения можно не только для всех записей, но и для записей, входящих в группу. Предположим, что запрос объединяет три таблицы, как показано на рис. 18.7.

Рис. 18.7. Запрос подсчитывает количество значений FORMAT в каждой группе

Запрос возвращает частные обобщения групп, т.е. количество форматов книг для каждого автора и названия. Группы состоят из записей, в которых совпадают фамилия автора и название книги. На рис. 18.7 первая группа содержит четыре записи, а остальные — по одной. Код SQL запроса, показанного на рис. 18.7, имеет следующий вид.

```
SELECT BOOKS_AUTHOR.NAME,
       BOOKS_PUBLICATION.TITLE,
       Count(BOOKS_EDITION.FORMAT) AS [Количество форматов]
FROM (BOOKS_AUTHOR INNER JOIN BOOKS_PUBLICATION
 ON BOOKS_AUTHOR.AUTHOR_ID=BOOKS_PUBLICATION.AUTHOR_ID)
 INNER JOIN BOOKS_EDITION ON
 BOOKS_PUBLICATION.PUBLICATION_ID=
            BOOKS_EDITION.PUBLICATION_ID
 GROUP BY BOOKS_AUTHOR.NAME, BOOKS_PUBLICATION.TITLE;
```

Можно задать другой уровень обобщения и группировки. Предположим, группа должна состоять из записей, содержащих информацию об одном авторе. Тогда для каждой группы (т.е. для каждого автора) можно отдельно подсчитать количество названий и форматов книг.

Рис. 18.8. Запрос подсчитывает количество названий и форматов для каждого автора

Группирование выполняется по значениям одного поля, а не двух, как в предыдущем запросе. Код SQL запроса, показанного на рис. 18.8, имеет следующий вид.

```
SELECT BOOKS_AUTHOR.NAME,
    Count(BOOKS_PUBLICATION.TITLE) AS [Количество названий],
    Count(BOOKS_EDITION.FORMAT) AS [Количество форматов]
 FROM (BOOKS_AUTHOR INNER JOIN BOOKS_PUBLICATION
    ON BOOKS_AUTHOR.AUTHOR_ID = BOOKS_PUBLICATION.AUTHOR_ID)
 INNER JOIN BOOKS_EDITION
 ON BOOKS_PUBLICATION.PUBLICATION_ID =
 BOOKS_EDITION.PUBLICATION_ID
GROUP BY BOOKS_AUTHOR.NAME;
```

Фильтрация обобщаемых записей

Кроме группирования записей, в решетке запроса можно задать критерии, ограничивающие количество обрабатываемых или выводимых записей. Критерий может быть установлен для полей следующих типов:

- используемое для группирования;
- используемое для обобщения;
- не используемое ни для группирования, ни для обобщения.

Рассмотрим запрос, конструкция которого показана на рис. 18.9.

Фильтрация записей выполняется на основе значений полей NAME, LIST_PRICE и PAGES. Ниже приведен код SQL запроса.

```
SELECT BOOKS_AUTHOR.NAME,
       BOOKS_EDITION.FORMAT,
       Sum(BOOKS_EDITION.LIST_PRICE) AS [Сумма стоимостей],
       BOOKS_EDITION.PAGES
FROM (BOOKS_AUTHOR INNER JOIN BOOKS_PUBLICATION
  ON BOOKS_AUTHOR.AUTHOR_ID = BOOKS_PUBLICATION.AUTHOR_ID)
  INNER JOIN BOOKS_EDITION
  ON BOOKS_PUBLICATION.PUBLICATION_ID =
```

Рис. 18.9. Фильтрация на основе группирующих и обобщаемых полей

```
                  BOOKS_EDITION.PUBLICATION_ID
GROUP BY BOOKS_AUTHOR.NAME,
         BOOKS_EDITION.FORMAT,
         BOOKS_EDITION.PAGES
HAVING (((BOOKS_AUTHOR.NAME)>"A")
   AND ((Sum(BOOKS_EDITION.LIST_PRICE))>10)
   AND ((BOOKS_EDITION.PAGES)>200));
```

Существенный недостаток этого запроса состоит в том, что фильтрация задана в директиве `HAVING`, расположенной после группирующей директивы `GROUP BY`. Фильтры, применяемые не к группам, а записям, следует размещать в директиве `WHERE`. Она выполняется перед директивами `ORDER BY` и `GROUP BY`. Обобщать данные, не возвращаемые запросом, не имеет смысла. Если запрос возвращает 10 записей, это не создаст проблем, однако, если перед группированием нужно отфильтровать миллион исходных записей, оставив только 1000, нерациональное размещение фильтра существенно ухудшит быстродействие запроса.

Перекрестные запросы

Перекрестные запросы — гибкий инструмент анализа информации. В рабочей среде Access можно создавать запросы и отчеты, в которых данные классифицируются, смешиваются, сравниваются и обобщаются по заданным правилам. Перекрестные запросы полезны для обобщения и поиска закономерностей в данных. Чем-то они похожи на электронные таблицы. В простейшем перекрестном запросе значения полей сравниваются друг с другом. Сравнение может выполняться для данных одной таблицы.

Предположим, что нужно создать перекрестный запрос для сравнения разных параметров книг. На рис. 18.10 показан результирующий набор перекрестного запроса, возвращающего количество названий книг каждого автора для разных форматов (`Hardcover` — твердая обложка, `Paperback` — мягкая обложка, `Audio Cassette` — компакт-диск).

Рис. 18.10. Простой перекрестный запрос

В первые годы появления и развития реляционных баз данных для создания перекрестных и других аналитических запросов нужно было создавать промежуточные таблицы. В настоящее время средства SQL позволяют создавать аналитические запросы без помощи промежуточных таблиц. Сейчас их можно создавать, используя такие команды, как CUBE, ROLLUP и GROUPING SETS.

На заметку Команды CUBE, ROLLUP и GROUPING SETS входят в стандарт ANSI языка SQL.

Драйвер Access обрабатывает перекрестные запросы немного иначе, чем обычные. Ниже приведен код SQL запроса, показанного на рис. 18.10. В коде используются команды TRANSFORM и PIVOT.

```
TRANSFORM Count(Books.TITLE) AS [Count-TITLE]
SELECT Books.FORMAT,
       Count(Books.TITLE) AS [Итоговое значение TITLE]
FROM Books
GROUP BY Books.FORMAT
PIVOT Books.NAME;
```

В этом коде столбец NAME (Имя и фамилия автора) используется в команде PIVOT в качестве группирующего поля перекрестного запроса. Запрос создает группы на основе значений столбца NAME. В "обычной части" запроса записи группируются по форматам книг, а функция Count, "как обычно", подсчитывает количество книг данного формата. Команда TRANSFORM передает количество книг каждого формата команде PIVOT, которая создает перекрестный результирующий набор.

Обратите внимание на то, что объект Books, выбранный при создании перекрестного запроса, является не таблицей, а запросом. Запрос Books объединяет три таблицы.

```
SELECT BOOKS_AUTHOR.NAME,
       BOOKS_PUBLICATION.TITLE,
       BOOKS_EDITION.FORMAT

FROM
   (BOOKS_AUTHOR INNER JOIN BOOKS_PUBLICATION
    ON BOOKS_AUTHOR.AUTHOR_ID =
              BOOKS_PUBLICATION.AUTHOR_ID)
    INNER JOIN BOOKS_EDITION
    ON BOOKS_PUBLICATION.PUBLICATION_ID =
               BOOKS_EDITION.PUBLICATION_ID;
```

Чтобы создать перекрестный запрос, показанный на рис. 18.10, выполните следующие действия.

1. **В режиме конструктора создайте простой запрос на основе объединения таблиц BOOKS_AUTHOR, BOOKS_EDITION и BOOKS_PUBLICATION.**

На заметку Простой запрос можно создать, написав его код SQL вручную, либо с помощью решетки в окне конструктора. Второй способ намного легче и быстрее.

   Конструкция запроса показана на рис. 18.11. Выбраны три таблицы. Для объединения таблиц перетащите общие имена полей из первой таблицы во вторую, а из второй — в третью.

Рис. 18.11. Конструкция запроса `Books`*, на основе которого вы создадите перекрестный запрос*

2. **Переименуйте созданный запрос, присвоив ему имя `Books`.**
3. **Запустите мастер запросов и выберите перекрестный запрос. В качестве основы создаваемого перекрестного запроса выберите запрос `Books`.**
4. **В следующем окне мастера задайте заголовки строк, как показано на рис. 18.10. Для этого переместите поле `FORMAT` в список выбранных полей.**
5. **В следующем окне выберите для заголовков столбцов поле `NAME`. В следующем окне выберите обобщающую функцию Число (Count). Щелкните на кнопках Далее (Next) и Готово (Finish).**

   Результирующий набор запроса должен выглядеть так же, как на рис. 18.10. Запрос подчитал количество книг каждого формата для всех авторов и для каждого автора отдельно.

Перекрестные запросы и лежащие в их основе запросы на объединение в реальных задачах намного сложнее, чем в рассмотренном выше примере. Сложность запроса зависит от количества уровней группирования и обобщения. Если уровней много как по вертикали, так и по горизонтали, то результирующий набор запроса неудобен для восприятия вследствие огромного количества строк и столбцов. Если перекрестный запрос возвращает более нескольких десятков строк и столбцов, работать с ним практически невозможно.

Записи, не имеющие подчиненных, и повторяющиеся записи

Кроме простого и перекрестного запросов, с помощью мастера можно создавать запросы еще двух типов, представленных элементами списка в первом окне мастера.

- **Повторяющиеся записи** (Find Duplicates). Запрос этого типа выводит дублированные записи одной таблицы. Дублированными считаются записи, содержащие одинаковые значения в заданных полях.
- **Записи без подчиненных** (Find Unmatched). Этот запрос выводит все записи, не имеющие ассоциированных записей â связанной таблице (например, продажи без клиентов). Такие записи обнаруживаются с помощью внешнего объединения двух таблиц.

Вывод повторяющихся записей

При создании запроса типа Повторяющиеся записи мастер спрашивает, какие поля таблицы нужно использовать для поиска дубликатов, а затем предлагает задать другие поля, которые нужно вывести в результирующий набор запроса. После ответа на эти вопросы мастер создает и выводит запрос.

Запрос Повторяющиеся записи часто используется для выявления дублированных значений ключей. Такая задача возникает, когда таблица Access создается путем копирования или импорта существующей заполненной таблицы. Если при попытке создать уникальный ключ Access возвращает сообщение об ошибке, это значит, что столбец ключа содержит значения `Null` или дублированные значения. Найти дублированные значения можно с помощью запроса типа Повторяющиеся записи.

Рассмотрим следующий запрос, объединяющий три таблицы.

```
SELECT BOOKS_AUTHOR.NAME, BOOKS_PUBLICATION.TITLE,
BOOKS_EDITION.FORMAT, BOOKS_EDITION.LIST_PRICE
FROM (BOOKS_AUTHOR INNER JOIN
BOOKS_PUBLICATION
ON BOOKS_AUTHOR.AUTHOR_ID = BOOKS_PUBLICATION.AUTHOR_ID)
INNER JOIN BOOKS_EDITION
ON BOOKS_PUBLICATION.PUBLICATION_ID =
BOOKS_EDITION.PUBLICATION_ID;
```

Сохраните запрос, а затем создайте на его основе новый запрос, выводящий только столбец `NAME`. Присвойте новому запросу имя `Дублированные_записи`. Результирующий набор запроса (рис. 18.12) содержит в столбце `NAME` много повторяющихся значений.

Создайте запрос, находящий дублированные записи. Для этого выполните следующие действия.

1. **Запустите мастер запросов и выберите в первом окне элемент Повторяющиеся записи.**
2. **Выберите запрос `Дублированные_записи`, затем поле `NAME` и выполните запрос.**

   Результат показан на рис. 18.13. Каждый автор упоминается один раз, а в следующем столбце приведено количество повторяющихся записей.

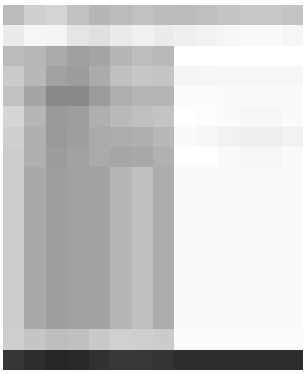


Рис. 18.12. Запрос возвращает много дублированных записей

Рис. 18.13. Запрос нашел повторяющиеся записи и вывел их количество

Вывод записей, не имеющих подчиненных записей

Запрос типа Записи без подчиненных позволяет выявить записи, не имеющие ассоциированных записей в связанной таблице, т.е. *висящие* записи.

Висящими называются записи, принадлежащие таблице на стороне "многие" отношения "один ко многим", а также записи на стороне "один" отношения "один ко многим" или "один к одному", не имеющие ассоциированных записей в связанной таблице.

При создании запроса этого типа мастер попросит задать имена двух таблиц и поля, связывающего таблицы. Затем мастер попросит задать поля, которые нужно вывести в результирующий набор, и создаст запрос.

Обычно запросы типа Записи без подчиненных используются для выявления записей, нарушающих ссылочную целостность отношения. В некоторых случаях таблицы Access создаются путем копирования или импорта существующих заполненных таблиц. Если после импорта попытаться создать отношение между таблицами и задать ссылочную целостность, драйвер Access выведет сообщение об ошибке. Для выявления записей, не позволяющих задать ссылочную целостность, можно применить запрос типа Записи без подчиненных.

Ниже приведен код SQL запроса типа Записи без подчиненных, выявляющего в таблице BOOKS_EDITION поля, не имеющие ассоциированных полей в таблице BOOKS_PUBLICATION.

```
SELECT BOOKS_EDITION.ISBN
FROM BOOKS_EDITION LEFT JOIN BOOKS_PUBLICATION
ON BOOKS_EDITION.[PUBLICATION_ID] =
     BOOKS_PUBLICATION.[PUBLICATION_ID]
WHERE (((BOOKS_PUBLICATION.PUBLICATION_ID) Is Null));
```

Как видите, запрос типа Записи без подчиненных — это всего лишь внешнее объединение (см. главу 4).

Запросы SQL

Ниже перечислены четыре типа запросов, которые нельзя создать с помощью решетки в режиме конструктора. Их можно создать, введя код запроса в режиме SQL.

- **Запрос на объединение** сравнивает поля нескольких таблиц или запросов с одним набором записей.
- **Запрос к серверу** передает команды SQL непосредственно соединению ODBC, а через него — базе данных, совместимой с драйвером ODBC.
- **Запрос определения данных** (другое название — *управляющий запрос*) создает или изменяет таблицы, создает индексы и выполняет другие операции, изменяющие структуру базы данных.

На заметку

Для создания запроса любого из этих трех типов щелкните правой кнопкой мыши в окне конструктора запросов и выберите в контекстном меню команду Запрос SQL (SQL Specific). Откроется меню, содержащее типы запросов. Кроме того, эти три элемента контекстного меню представлены кнопками в разделе Тип запроса (Query Type) вкладки Конструктор (Design), когда запрос открыт в режиме конструктора.

- **Вложенный запрос** используется для определения поля или критерия для поля в другом запросе.

Создание запросов на объединение

Создание запроса на объединение начинается в режиме конструктора с выбора двух таблиц или запросов. Затем в режиме SQL нужно добавить код объединения.

1. **Создайте новый пустой запрос в режиме конструктора. Добавьте в запрос таблицу `BOOKS_AUTHOR`.**
2. **Добавьте оба поля (`AUTHOR_ID` и `NAME`) таблицы `BOOKS_AUTHOR` в решетку запроса.**
3. **Щелкните правой кнопкой мыши на верхней панели конструктора и выберите в контекстном меню команду Запрос SQL⇨Объединение (SQL Specific⇨ Union).**

   Откроется окно текстового редактора SQL, в котором нужно создать объединение двух таблиц. Объединять можно также таблицу с этой же таблицей. Такая операция тоже выполняется в редакторе SQL.

4. **Введите в окне редактора следующий код SQL.**

```
SELECT * FROM BOOKS_AUTHOR
UNION
SELECT * FROM BOOKS_AUTHOR
```

Результирующий набор показан на рис. 18.14.

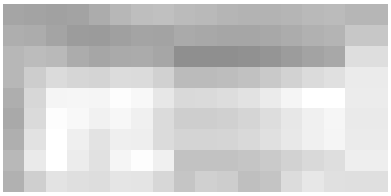


Рис. 18.14. Запрос на объединение дважды извлек записи из одной и той же таблицы

Во всех запросах на объединение должны соблюдаться следующие правила.

- Количество полей на обеих сторонах объединения должно быть одинаковым.
- Типы данных всех полей должны совпадать.
- Поля должны быть распложены в одинаковой последовательности.
- Поля, отсутствующие в одном из операторов `SELECT` и присутствующие в другом, можно заменить значениями `NULL`, как в следующем примере.

```
SELECT AUTHOR_ID, NAME, NULL FROM BOOKS_AUTHOR
UNION
SELECT PUBLICATION_ID, FORMAT, ISBN FROM BOOKS_EDITION
```

Результирующий набор показан на рис. 18.15.

Рассмотренные выше два запроса имеют существенный недостаток, особенно первый запрос. Таблица `BOOKS_AUTHOR` читается дважды, хотя используется только одна копия записей. Это объясняется тем, что дублированные записи предварительно удалены из нее.

Рис. 18.15. Запрос на объединение двух разных таблиц

При использовании команды `UNION` в операторе `SELECT` она копирует только записи, не дублированные при объединении таблиц. Существование дубликатов определяется содержимым всех полей, выбранных командой `UNION`. Если две записи имеют одинаковое содержимое во всех выбранных полях, они считаются дубликатами, и выводится только одна запись. Если

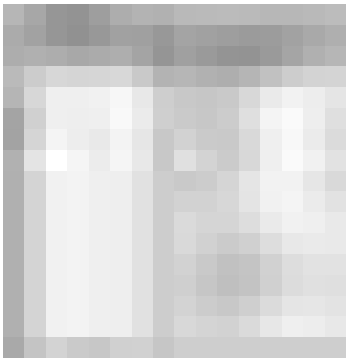


Рис. 18.16. Запрос на объединение возвратил все записи, включая дублированные

другие поля, имеющие разные значения, не используются запросом UNION, они (поля) не применяются для выявления дубликатов. Чтобы увидеть все записи объединения двух таблиц, нужно применить ключевое слово ALL после команды UNION. Тогда команда объединения будет иметь вид UNION ALL SELECT.

Для вывода всех записей запрос нужно изменить следующим образом.

```
SELECT * FROM BOOKS_AUTHOR
UNION ALL
SELECT * FROM BOOKS_AUTHOR
ORDER BY AUTHOR_ID;
```

Результат показан на рис. 18.16.

Кроме того, в запрос добавлена директива ORDER BY, благодаря которой на рис. 18.16 записи с одним и тем же автором расположены рядом друг с другом.

На заметку Директива сортировки ORDER BY применяется к результату объединения, а не отдельно к первому или второму запросу SELECT. Иными словами, сортировка выполняется после объединения.

Создание запросов к серверу

Запрос к серверу передает команды SQL непосредственно на серверную базу данных, например Oracle, Microsoft SQL Server и т.д. К серверной базе данных обычно подключено много клиентских приложений Access. Команды передаются с использованием синтаксиса серверной базы данных. Следовательно, чтобы создать запрос к серверной базе данных, нужно ознакомиться с ее документацией.

Запросы к серверу могут извлекать записи, изменять данные и выполнять процедуры и триггеры, хранящиеся на сервере. Кроме того, они могут создавать на сервере новые таблицы на уровне базы данных (в отличие от локальных таблиц на уровне клиентской базы данных).

После создания запроса к серверу нужно специфицировать параметры соединения с базой данных. Строку соединения можно вручную ввести в свойство ODBCConnectStr запроса. Можно также выбрать команду Построить (Build) и задать параметры сервера, с которым нужно установить соединение. Если не задать строку соединения, Access выведет приглашение ввести параметры соединения при попытке выполнить запрос к серверу.

Создание запросов на определение данных

В рабочей среде русифицированной версии Access запросы на определение данных называются *управляющими*. В реальных приложениях они используются редко, поскольку все, что можно сделать с помощью управляющих запросов, можно также сделать (причем легче) с помощью инструментов графического интерфейса Access. Управляющие запросы используются для создания или изменения объектов базы данных. Для этого в них применяются такие операторы SQL:

- CREATE TABLE (создание таблицы);
- ALTER TABLE (изменение таблицы);
- DROP TABLE (удаление таблицы);

- CRATE INDEX (создание индекса);
- DROP INDEX (удаление индекса).

Показанный ниже код управляющего запроса, введенный в окно SQL, создает локальную таблицу Access и присваивает ей имя TelephoneList (Список телефонов).

```
CREATE TABLE TelephoneList
( [TeleID] integer, [FullName] text,
[Address1] text, [Address2] text,
[Address3] text, [Country] text,
[Phone 1] text, [Phone 2] text,
[FaxPhn 1] text, [Notes] memo,
CONSTRAINT [Index1] PRIMARY KEY ([TeleID]) );
```

Если активизировать этот запрос в базе данных Access, в ней будет создана таблица TelephoneList. Ниже приведен код управляющего запроса, создающего индекс существующей таблицы TelephoneList на основе столбцов Country (Страна) и FullName (Фамилия). Индексу присваивается имя CountryName.

```
CREATE INDEX CountryName
On TelephoneList ([Country], [FullName]);
```

На заметку В каждом управляющем запросе может присутствовать только один оператор SQL.

Вложенные запросы SQL в запросах Access

Средства Access позволяют создавать операторы SQL в запросах Access на изменение и выборку. Оператор SQL можно вставить в ячейку Поле (Field) для определения нового поля или в ячейку Условие отбора (Criteria) для применения к полю заданного условия. С помощью вложенных запросов можно решать следующие задачи.

- Поиск значений в родительском запросе на основе их равенства или неравенства значениям, возвращаемым вложенным запросом. Для этого во вложенном запросе используются ключевые слова ANY, IN, ALL.
- Проверка существования значений в результирующем наборе вложенного запроса с помощью ключевых слов EXISTS и NOT EXISTS.
- Создание вложенных запросов следующих уровней, т.е. вложенных во вложенный запрос.

Вложенный оператор SQL можно размещать в ячейках Поле и Условие отбора в решетке конструктора родительского запроса. Оператор SQL, размещенный в ячейке Поле, создает новый столбец в результирующем наборе родительского запроса, а размещенный в ячейке Условие отбора — задает критерий, ограничивающий (фильтрующий) возвращаемые записи.

Запросы на изменение

Запрос на изменение не просто извлекает группы записей и выводит их в результирующий набор. Слово “изменение” означает, что запрос изменяет нечто в базе данных. Запрос на изменение можно рассматривать как запрос на выборку, выполняющий некоторые дополнительные операции над таблицами базы данных.

Типы запросов на изменение

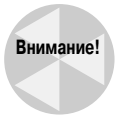
При создании любого запроса рабочая среда Access автоматически начинает его проектирование как запроса на выборку. Затем разработчик задает тип запроса в окне мастера или с помощью инструментов режима конструктора.

Запрос на изменение нельзя создать с помощью мастера запросов. Чтобы создать запрос на изменение, активизируйте режим конструктора, щелкните на верхней панели правой кнопкой мыши и выберите тип запроса. Ниже перечислены четыре команды контекстного меню, определяющие тип запроса.

- **Создание таблицы** (Make Table Query). Запрос выбирает записи из существующих таблиц базы данных и сохраняет их в новой таблице. Предположим, что нужно создать таблицу журнала и скопировать в нее все неиспользуемые записи, чтобы можно было удалить их из рабочих таблиц базы данных. Например, запись считается неиспользуемой, если клиент ничего не купил и не продал на протяжении двух лет. Для решения этой задачи нужно создать запрос типа Создание таблицы, копирующий неиспользуемые записи в новую таблицу и удаляющий их из рабочих таблиц.
- **Обновление** (Update Query). Запрос обновляет данные (т.е. изменяет значения полей) в существующих таблицах.
- **Добавление** (Append Query). Запрос добавляет новые записи в существующую таблицу. Предположим, что один из клиентов, о котором не было ничего слышно на протяжении четырех лет, хочет сделать покупку. Прежнюю информацию о нем нужно переместить в рабочий файл из файла резервной копии. Для решения этой задачи можно применить запрос типа Добавление, который добавит записи из таблиц резервной копии в рабочие таблицы.
- **Удаление** (Delete Query). Запрос удаляет из таблицы записи, удовлетворяющие заданному критерию.

Узнать запрос на изменение на панели навигации можно по специальной пиктограмме, на которой изображен восклицательный знак.

В отличие от запросов на выборку, выводящих данные в заданном виде, запросы на изменение выполняют операции над записями, хранящимися в нижележащих таблицах. Это могут быть операции копирования данных из других таблиц, изменения содержимого полей существующей таблицы, удаления записей.



Поскольку запрос на изменение, в отличие от запроса на выборку, может повредить или уничтожить данные, соблюдайте следующие правила предосторожности: всегда создавайте резервную копию таблицы перед выполнением запроса на изменение; всегда просматривайте предварительный результат запроса на изменение (с помощью кнопки Режим таблицы (Table View)) перед его выполнением. Учитывайте, что щелчок на кнопке Режим таблицы выводит не результат работы запроса, а выбранные им данные.

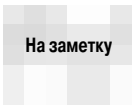
Просмотр результата запроса на изменение

Запросы на изменение могут разрушить данные, поэтому использовать их надо осторожно. Просмотрите данные, которые обрабатываются запросом (щелкнув на кнопке Режим

таблицы), перед запуском запроса, а после его выполнения проверьте, правильно ли внесены изменения. Перед созданием и выполнением запроса на изменение проанализируйте весь процесс и продумайте стратегию проверки изменений, поскольку отменить изменения таблицы будет невозможно.

Просмотр результатов запроса перед обновлением и удалением

Прежде чем запустить запрос на изменение, переключите его в режим таблицы, чтобы увидеть набор данных, которые будут обработаны запросом. При обновлении записей с помощью запроса на изменение операции выполняются над нижележащей таблицей, используемой запросом. Перед фактическим выполнением операций обновления или удаления просмотрите изменяемые данные, щелкнув на кнопке Режим таблицы.



Если запрос на изменение модифицирует поля, используемые для выбора записей, просмотрите нижележащую таблицу или запустите запрос на выборку, чтобы увидеть изменения. Предположим, что запрос удалил часть записей, удовлетворяющих заданному критерию. Тогда результирующий набор на основе этого же критерия будет пустым, поскольку в таблице уже нет удовлетворяющих его записей. Удалив критерий, можно просмотреть оставшиеся записи и проверить, все ли ненужные записи удалены.

Вывод результирующей таблицы запроса на добавление записи или создание таблицы

В отличие от запросов на обновление и удаление, запросы на создание таблицы и добавление записей копируют результирующие записи в другую таблицу. После задания полей и критерия в окне конструктора запросы на создание таблицы и добавление записей копируют заданные поля и записи в другую таблицу. При выполнении запроса его результат сохраняется в другой таблице (не в текущей).

Критерии выбора

В критериях выбора можно использовать любые выражения, состоящие из полей, функций и операторов, для задания любых ограничивающих условий, которые нужно разместить в запросе. Критерий выбора служит фильтром, сообщающим Access о том, какие записи нужно найти в нижележащих таблицах или вывести в результирующий набор. Запрос на изменение не создает результирующий набор, поэтому критерии выбора используются в нем для задания набора записей, обрабатываемых запросом.

После щелчка на кнопке Режим таблицы выводится результирующий набор только заданных критериев и полей до изменения, а не фактическая таблица, содержащая новые или добавленные записи. Чтобы просмотреть окончательный результат работы запроса на создание таблицы или добавление записей, откройте новую таблицу и просмотрите ее содержимое. Если запрос на изменение не планируется использовать в дальнейшем, удалите его.

Результат запроса на изменение нельзя отменить

Запрос на изменение копирует или модифицирует данные, хранящиеся в нижележащих таблицах. После выполнения запроса на изменение результат его работы сохраняется в таблицах. Поэтому перед работой с запросами на изменение создайте запрос на выборку, чтобы убедиться в том, что используемые критерии выбора записей работают правильно.

Запрос на изменение может повредить данные, поэтому перед его запуском всегда создавайте резервную копию нижележащих таблиц. Если запрос на изменение больше не будет использоваться, удалите его.

Создание запросов на изменение

Запрос на изменение создается почти так же, как и запрос на выборку. Нужно задать используемые поля и критерии выбора. Кроме того, нужно сообщить Access необходимый тип запроса: на создание таблицы, добавление, обновление или удаление.

Запросы на обновление

Каждую запись таблицы можно обновить индивидуально с помощью формы. Еще один способ: можно спроектировать запрос на выборку, создающий результирующий набор обновленной таблицы, а затем заменить исходную таблицу результирующим набором. Однако, если записей больше тысячи, любой из этих способов требует слишком много времени.

Лучше всего решить эту задачу с помощью запроса на обновление, который вносит все изменения в одной операции. При этом не только экономится время, но и уменьшается количество ошибок во время ввода данных в форму вручную.

Чтобы создать простой запрос на обновление, выполните следующие действия.

1. **В режиме конструктора создайте новый запрос на выборку и добавьте в него все столбцы таблицы `BOOKS_AUTHOR`.**
2. **Щелкните правой кнопкой мыши на верхней панели запроса и выберите в контекстном меню команду Тип запроса⇨Обновление (Query Type⇨Update Query).**

   Можете также щелкнуть на кнопке Тип запроса: обновление, расположенной в разделе Тип запроса. В любом случае будет открыто окно кода SQL. Введите оператор

   ```
   UPDATE BOOKS_AUTHOR SET BOOKS_AUTHOR.[NAME]="Unknown";
   ```

Если выполнить запрос, все фамилии авторов, хранящиеся в таблице `BOOKS_AUTHORS`, будут заменены словом `Unknown`.

Запрос на обновление фактически выполняется оператором `UPDATE`, который обновляет одну запись или многие записи таблицы.

Запросы на создание таблиц

Запрос этого типа создает новую таблицу на основе критерия выбора. Чтобы создать запрос этого типа, выполните следующие действия.

1. **В режиме конструктора создайте запрос на выборку, выводящий все записи таблицы `BOOKS_AUTHOR`.**
2. **Щелкните правой кнопкой мыши на верхней панели конструктора и выберите команду Тип запроса⇨Создание таблицы (Query Type⇨Make-Table Query).**

Можете также щелкнуть на кнопке Тип запроса: создание таблицы, расположенной в разделе Тип запроса. В любом случае будет открыто диалоговое окно Создание таблицы. Задайте в нем имя новой таблицы и щелкните на кнопке OK. В открывшемся окне SQL введите код

```
SELECT BOOKS_AUTHOR.AUTHOR_ID
BOOKS_AUTHOR.NAME INTO [New_Authors]
FROM BOOKS_AUTHOR;
```

На заметку Запрос на создание таблицы выбирает записи из таблиц базы данных и вставляет их в новую таблицу. Эти операции можно также выполнить с помощью операторов `INSERT INTO...SELECT`.

При создании многих запросов типа Создание таблицы удобнее пользоваться кнопкой Тип запроса: создание таблицы, чем контекстным меню. Драйвер Access присваивает запросу очередное имя, например `Запрос10`. Если при выполнении запроса в базе данных существует таблица с именем, заданным в запросе для новой таблицы, запрос переопределит таблицу, выведя приглашение подтвердить замену. Можете в ответ на приглашение задать новое имя. Изменить имя новой таблицы в запросе (естественно, до его запуска) можно, отредактировав свойство `Таблица-получатель` (Destination Table).

Запросы на добавление

Запрос типа Добавление создает в существующей таблице новые записи (т.е. добавляет записи), используя данные существующих таблиц. Таблица, в которую добавляются записи, должна существовать до запуска запроса на добавление. Она может находиться в текущей или в другой базе данных Access.

Чаще всего в запросах на добавление используется некоторый критерий выбора. Учитывайте, что для добавления записей в таблицу, хранящуюся в другой базе данных Access, запрос на добавление — не всегда самый эффективный инструмент. Например, если нужно добавить все поля и все записи из существующей таблицы в новую таблицу, то лучше воспользоваться командами Копировать (Copy) и Вставить (Paste).

На заметку Записи можно добавлять как в закрытую, так и в открытую таблицу. Закрывать таблицу перед добавлением записей нет необходимости. Однако Access не обновляет автоматически внешний вид открытой таблицы, в которую добавлены новые записи. Чтобы обновить внешний вид таблицы, нажмите комбинацию клавиш <Shift+F9>. Вы увидите новые, добавленные записи.

Создавая запрос на добавление, придерживайтесь следующих правил.

- Если таблица, в которую добавляются записи, имеет первичный ключ, добавляемые записи не могут иметь значения `Null` или дублированные значения в полях первичного ключа. В противном случае драйвер Access не добавит записи, причем не выведет при этом никакого сообщения.
- Для добавления записей в таблицу, определенную в другой базе данных Access, нужно указать имя и маршрут файла другой базы данных.
- Если в ячейке Поле решетки запроса находится символ `*`, то использовать в запросе отдельные столбцы этой же таблицы невозможно. Драйвер Access решит, что вы пытаетесь добавить содержимое полей дважды в одну и ту же запись, и не выполнит команду добавления.
- При добавлении записей с полем типа `Счетчик` не включайте в записи это поле, если принимающая таблица использует его в качестве первичного ключа. Кроме того, если

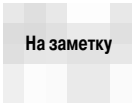
записи добавляются в пустую таблицу и нужно, чтобы в ней были новые значения Счетчик на основе некоторого критерия, не используйте поля типа Счетчик.

Для создания запроса на добавление выполните следующие действия.

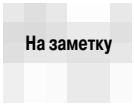
1. **Создайте запрос на выборку, выводящий все записи и поля таблицы BOOKS_AUTHOR.**
2. **Щелкните на кнопке Тип запроса: добавление (Query Type⇨Append Query), расположенной в разделе Тип запроса.**

   В открывшемся диалоговом окне Добавление задайте имя таблицы, в которую нужно добавить записи, и щелкните на кнопке OK. Введите в окне SQL код

```
INSERT INTO authors ( AUTHOR_ID, NAME )
SELECT BOOKS_AUTHOR.AUTHOR_ID, BOOKS_AUTHOR.NAME
FROM BOOKS_AUTHOR;
```



Запрос на добавление выполняет оператор INSERT, который вставляет результирующий набор оператора SELECT в таблицу. Оператор SELECT можно создать в режиме конструктора.



Запрос на добавление копирует только поля, которые имеют одинаковые имена в обеих таблицах. Предположим, что одна таблица состоит из шести полей, а другая — из девяти, причем в другой таблице только пять из шести полей имеют имена, совпадающие с именами полей первой таблицы. Тогда при добавлении записей из первой таблицы во вторую будет скопировано содержимое только пяти полей, а остальные поля останутся пустыми.

Если в запросе на добавление используются символ * и отдельное поле той же таблицы, для которой задан символ *, то критерий для отдельного поля не должен быть размещен в ячейке Добавление (Append To). В противном случае Access возвратит сообщение об ошибке. Это объясняется тем, что поле критерия уже используется в запросе (символ * задает выбор всех полей).

Запросы на удаление

Запрос типа Удаление (Delete Query) по сравнению с другими типами запросов на обновление самый опасный, поскольку восстановить удаленные им записи невозможно.

Как и другие запросы на обновление, запрос на удаление обрабатывает группу записей на основе критерия выбора.

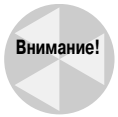
Запрос типа Удаление может удалять записи из многих таблиц, однако при выполнении этой операции нужно учитывать следующее.

- Удаление записей не должно нарушать ссылочную целостность отношений между таблицами.
- Для объединения таблиц должен быть установлен флажок обеспечение целостности данных (Enforce referential integrity).
- Для отношений “один к одному” и “один ко многим” должен быть установлен флажок каскадное удаление связанных записей (Cascade delete related records).

Если отношение в запросе не определено, а флажок каскадное удаление связанных записей снят, то запрос удалит записи только на стороне “многие”. Чтобы после этого удалить записи на стороне “один”, нужно удалить таблицу “многие” из запроса.

Рассмотренный выше метод неэффективный и громоздкий. Поэтому для удаления связанных записей убедитесь в том, что отношение определено, а флажок каскадное удаление

связанных записей установлен. Тогда связанные записи на обеих сторонах отношения можно удалять, запустив запрос один раз.



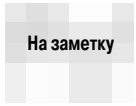
Восстановить записи после их удаления невозможно, поэтому всегда создавайте резервные копии таблиц перед выполнением запросов на удаление.

Чтобы создать запрос на удаление, выполните следующие действия.

1. **Создайте запрос на выборку, выводящий все записи и поля таблицы `BOOKS_AUTHOR`.**
2. **Щелкните правой кнопкой мыши на верхней панели конструктора и выберите в контекстном меню команду Тип запроса: удаление (Query Type⇨Delete Query).**

   Можете также щелкнуть на кнопке Тип запроса: удаление, расположенной на ленте в разделе Тип запроса. В окне SQL появится следующий код.

```
DELETE BOOKS_AUTHOR.AUTHOR_ID, BOOKS_AUTHOR.NAME
FROM BOOKS_AUTHOR;
```



Запрос на удаление выполняет команду `DELETE` для многих записей на основе заданного критерия.

Не забывайте, что восстановить записи, удаленные запросом, невозможно. Поэтому перед запуском запросов на удаление всегда создавайте резервные копии таблиц.

Сохранение запроса на обновление

Запрос на обновление сохраняется так же, как и запрос любого другого типа. В режиме конструктора его можно сохранить, щелкнув на кнопке Сохранить (Save) панели инструментов быстрого доступа или выбрав команду Office⇨Сохранить. Если сохранение выполняется в первый раз, Access попросит подтвердить имя запроса, предлагаемое по умолчанию, или ввести другое имя.

Кроме того, запрос сохраняется при закрытии окна конструктора.

Выполнение запроса на изменение

После сохранения запроса на изменение его можно выполнить, дважды щелкнув на его имени на панели навигации. Драйвер Access выведет окно, сообщающее о количестве изменяемых записей, и попросит подтвердить изменение.

Устранение неполадок в запросах на изменение

При работе с запросами на изменение могут возникнуть некоторые проблемы. Во время выполнения запроса иногда появляются сообщения разных типов, например, о том, что записи утеряны вследствие нарушения правил для ключевых полей, или о том, что записи заблокированы. В следующих разделах рассматриваются некоторые из этих проблем.

Ошибки типов данных при добавлении или обновлении

Если запрос попытается ввести в поле значение, не соответствующее типу поля, Access преобразует его в значение `Null`, ничего не сообщая об этом пользователю. Следовательно, Access добавит записи, но с пустыми полями.

Нарушения в ключевых полях

При попытке добавить записи в другую базу данных, в которой определен первичный ключ, Access не добавит те записи, в которых значения поля первичного ключа дублируют существующие значения.

Драйвер Access не позволит обновить запись и значение первичного ключа, если значение уже существует. Изменить значение первичного ключа на другое можно при следующих условиях.

- Новое значение первичного ключа не существует в таблице.
- Изменяемое поле не связано с полями других таблиц.

Драйвер Access не позволяет удалять записи на стороне "один" отношения "один ко многим", если предварительно не удалены связанные записи на стороне "многие".

Драйвер Access не позволяет добавлять или обновлять поля, если новые значения будут дублировать уникальные индексные поля. Индексным является поле, свойству `Индексированное поле` (Index) которого присвоено значение `Да (Совпадения не допускаются)` (Yes (No Duplicates)).

Блокировка полей на уровне записей в многопользовательской среде

Драйвер Access не выполнит запрос на изменение записей, заблокированных другими пользователями. При выполнении запроса на обновление или добавление можно заставить Access продолжить изменение всех других, не заблокированных значений, однако тогда вы не сможете выяснить, какие записи остались не измененными.

Текстовые поля

При добавлении или обновлении полей текстового типа, новая длина которых меньше исходной длины, Access отсекает символы, не поместившиеся в новое поле. Не забывайте, что окно сообщений при этом не выводится.

Резюме

Запросы — это сердцевина каждого приложения баз данных. Они выполняют главную работу по сбору и обработке информации, необходимой пользователям. Без запросов вам пришлось бы писать длинный код VBA для решения каждой задачи, решить которую можно, несколько раз щелкнув мышью на панели конструктора запросов.

Вряд ли вы когда-либо будете использовать все методы, рассмотренные в данной главе. Средства Access не только мощные, но и весьма разнообразные. Однако, чем больше средств проектирования запросов вы будете знать, тем легче вам будет найти оптимальное решение. Не забывайте, что с помощью кода SQL можно сделать все, что можно сделать с помощью конструктора запросов, однако в большинстве случаев конструктор предоставляет более легкий и быстрый способ решения задачи.