|  |
| --- |
|  |

|  |
| --- |
| **Лабораторная работа  «SQL-запросы»** |
| *Лабораторный практикум по базам данных* |
|  |
| 1. Построение простых запросов 2. Построение сложных запросов    * Запросы на выборку    * Запросы с параметрами    * Запросы на изменение |
|  |
| **ЧувГУ, каф. Вычислительной техники** |
| **23.02.2021** |
|  |

# Лабораторная работа «Создание SQL-запросов»

Суть любой базы данных – ответы на вопросы пользователя относительно данных, хранящихся в таблицах или других структурах. Такие вопросы формируются на специальном языке в виде запроса, обрабатываются СУБД и в качестве ответа предоставляют записи, которые соответствуют поставленному вопросу. Эти записи отображаются в виде промежуточной таблицы, которую называют *набором записей (Record Set)* . Запросы существуют в двух формах**:** как *инструкция SQL* или как *сохраненный именованный запрос.*

Запросы в Access можно разделить на три основные группы.

1. **Запросы на выборку.** Извлекают информацию из одной или нескольких таблиц, а также из других запросов, включая групповые и перекрестные запросы.
2. **Запросы на изменение.** Применяются для изменения данных в таблицах.
3. **Запросы к серверу.** Используются для передачи команд прямо на сервер SQL. Главной особенностью этого запроса является то, что он должен использовать синтаксис языка SQL сервера базы данных (для Microsoft SQL Server это — Transact-SQL).

В данной лабораторной работе рассматриваются, прежде всего, первых два вида запросов, для которых в приведенной ниже таблице обобщены области их применения в Access.

Таблица 1.1. Где в Access используются запросы

|  |  |
| --- | --- |
| **Запросы на выборку** | **Запросы на изменение** |
| В свойстве *Источник записей* формы | Макрокоманда OpenQuery |
| В свойстве *Источник записей* отчета | Инструкция VBA DoCmd OpenQuery |
| В качестве фильтра в форме | Запросы из контейнера базы данных |
| Страницы доступа к данным | Источники для запросов ADO Connection.Execute |
| Выборка записей для других запросов |  |

Справочную информацию по конструкциям языка Jet SQL можно найти в файле C:\Program Files\Common Files\Microsoft Shared\OFFICE11\1049\JetSQL40.CHM или через общую справку **🕮** Создание БД и объектов\Объекты БД\Работа с запросами/Справка по языку MS Jet SQL.

## Что надо знать

**Построение простых запросов**

1. Создание простого запроса с помощью Мастера запросов
2. Создание запроса с помощью Конструктора (**🕮** Разработка запросов**)**
3. Использование выражений в запросах (**🕮** Работа с выражениями)
   1. Выражения в Access
      1. Арифметические операторы
      2. Операторы присваивания и сравнения
      3. Логические операторы
      4. Операторы слияния строковых значений (& и +)
      5. Операторы идентификации «**!**» и «**.**»
      6. Операторы сравнения с образцом (Between, In, Like, Is Null)
      7. Идентификаторы объектов
      8. Константы (числовые, строковые, даты и времени)
      9. Функции
   2. Создание выражений с помощью построителя выражений (**🕮** Работа с выражениями **/**Общие сведения о построителе выражений )
4. Создание запросов с различным видом соединением
   * 1. Внутреннее соединение по нескольким полям
     2. Внешнее соединение (*Left Join и Right Join*)
     3. Рекурсивное соединение (**🕮**)
5. Создание запроса из фильтра
6. Отображение в запросе только максимальных или минимальных значений (**🕮** Предикаты ALL, DISTINCT, DISTINCTROW, TOP)
7. Изменение данных в результирующем множестве запроса
8. Печать результатов запроса

**Построение сложных запросов**

1. **Запросы на выборку**. Выборка данных из нескольких таблиц и запросов (Создание БД и объектов\Объекты БД\Работа с запросами**)**
   1. Вывод только уникальных записей (**🕮**)
   2. Использование условий отбора для поиска определенных записей (**🕮)**
   3. Создание вычисляемого поля в запросе (**🕮)**
   4. Итоговые записи в запросе (**🕮)**
   5. Запросы на автоподстановку значений (**🕮)**
2. **Запросы с параметрами** (**🕮**Создание БД и объектов\Объекты БД\Работа с запросами\Создание запросов\ Работа с запросами с параметрами)
   1. Использование одного параметра
   2. Использование нескольких параметров
   3. Использование параметров с подстановочными знаками
   4. Использование [типа данных](mk:@MSITStore:C:\Program%20Files\Microsoft%20Office\Office10\1049\acmain10.chm::/html/achowCreateParameterQueryThatPromptsCriteriaUsesSeS.htm##) для запросов с параметрами
   5. Создание пользовательского диалогового окна, запрашивающего ввод условий отбора запроса с параметрами (*выполняется в лабораторной работе «Разработка форм»*)
3. **Запросы на изменение** (**🕮**Создание БД и объектов\Объекты БД\Работа с запросами\Создание запросов\ Работа с запросами на изменение \ Запросы на изменение)
   1. На удаление записи (**🕮** [Инструкция DELETE](mk:@MSITStore:C:\Program%20Files\Common%20Files\Microsoft%20Shared\Office11\1049\jetsql40.chm::/dasqldelete.htm) )
   2. На обновление записи. (**🕮**  [Инструкция UPDATE](mk:@MSITStore:C:\Program%20Files\Common%20Files\Microsoft%20Shared\Office11\1049\jetsql40.chm::/dasqlupdate.htm))
   3. На добавление записей. (**🕮**  [Инструкция INSERT INTO](mk:@MSITStore:C:\Program%20Files\Common%20Files\Microsoft%20Shared\Office11\1049\jetsql40.chm::/dasqlinsertinto.htm))
   4. На создание таблицы.
      1. Создание таблицы на основе другой таблицы с помощью запроса (**🕮** справка)
      2. Создание запроса на создание таблицы (**🕮** справка)
      3. Выполнение запроса на создание таблицы
   5. Включение или отключение вывода запросов на подтверждение действий
4. **Анализ данных** с помощью запросов (**🕮** Создание БД и объектов \Объекты БД \ Работа с запросами \Создание запросов \ Работа с запросами с параметрами)
   1. Перекрестные запросы (**🕮** Перекрестные запросыСоздание запроса на выборку или перекрестного запроса)
      1. Использование фиксированных заголовков столбцов в перекрестных запросах
      2. Изменение уровня детализации в запросе
   2. Выполнение вычислений в запросе (**🕮** Вычисления в запросе)
      1. Вычисления по всем записям
      2. Вычисления по группе записей
      3. Обработка пустых значений (Null) в расчетах
   3. Отбор повторяющихся записей и записей без подчиненных
5. Подчиненный запрос. (**🕮)**
   1. Подчиненный запрос. Для подчиненного запроса пользователь должен ввести инструкцию SQL в строку **Поле** или **Условие отбора** в бланке запроса.
   2. Подчиненный запрос SQL
6. Создание запросов SQL (**🕮**Создание БД и объектов \Объекты БД \Работа с запросами \Работа с запросами SQL \Создание запросов SQL)
   1. Запрос на объединение
   2. Запрос к серверу
   3. Управляющий запрос (**🕮**Примеры управляющих запросов, Инструкция CREATE TABLE**)**
   4. Создание динамических запросов
      1. Использование метода RunSQL объекта DoCmd с 231
7. Настройка свойств полей запроса и свойств самого запроса (**🕮**Создание БД и объектов \Объекты БД \Работа с запросами \Настройка запросов )
8. Передача результатов запроса в другие приложения (**🕮** Работа с данными \ Работа с внешними данными)
   1. Включение таблиц и запросов в составные документы Microsoft Word (Сервис | Свяи с Office | Слияние с MS Word) (доп. задание)
   2. Экспорт данных (выбрать требуемый запрос и далее выполнить команду ***Экспорт***, выбрать тип файла, имя файла и сохранить)

Изображение выглядит как текст

Автоматически созданное описание

## Основное задание

1. Научитесь создавать простые запросы, используя *Мастер запросов*. При создании запросов используйте несколько таблиц или запросов. Создайте в *Мастере запросов* итоговые запросы.
2. Изучите работу *Конструктора запросов*. Просмотрите в режиме *Конструктора* запросы, созданные *Мастером запросов*
3. Создайте запросы с выражениями, в которых используются все виды операторов.
4. Создайте запрос с предикатом TOP
5. Создайте запросы с различным видом соединения
   1. с внутренним соединением
   2. с внешним соединением
   3. с self-соединением (самообъединение)
   4. с оператором UNION
6. Создайте запрос из фильтра. Создайте расширенный фильтр и сохраните его в виде запроса
7. Создайте запросы на изменение

**СОВЕТ***.* Чтобы создать запрос на изменение, проще всего разработать обычный запрос на выборку, удостовериться в его функциональности, после чего изменить его тип. Чтобы проверить, какое действие выполнит запрос на изменение, вместо команды Запуск нужно выбрать отображение запроса в режиме таблицы. Для этогощелкните на стрелке кнопки Вид и выберите из меню опцию Режим таблицы.

* 1. Создайте запрос на создание таблицы (CREATE TABLE)
  2. Создайте запрос на добавление записей ( INSERT INTO)

**Примечание.** Этот тип запросов можно использовать также для того, чтобы быстро клонировать записи таблицы. Откройте запрос на выборку в режиме Конструктора и выберите команду меню Запрос ⇨ Добавление. Выберите целевую таблицу в раскрывающемся списке Имя таблицы и щелкните на кнопке ОК.

* 1. Создайте запрос на обновление записей ( UPDATE SET)
  2. Создайте запрос на удаление записей ( DELETE)

1. Для каждого созданного запроса составьте его краткое описание. Для этого щелкните правой кнопкой по запросу и из контекстного меню выберите пункт *Свойства*.
2. Для хранения информации о запросах создайте дополнительную таблицу ***т\_РеестрЗапросов*** с заданной структурой:

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Имя поля** | **Тип данных** | **Описание** |
| 1 | QueryID | Текстовый | Имя запроса |
| 2 | QueryTitle | Текстовый | Описание запроса |
| 3 | QueryDesc | Поле Memo | Инструкция для пользователей |
| 4 | QueryTableDate | Дата/Время | Дата добавления запроса в таблицу |
| 5 | QueryExpose | Логический | Предоставить доступ пользователям? |

1. Создайте запрос ***з\_ОбновитьРеестрЗапросов***, используя информацию, хранящуюся в системных таблицах (см. приложение к лабораторной работе). По данному запросу информация из системных таблиц о существующих в проекте запросах будет добавлена в таблицу ***т\_ВсеЗапросы.*** Во время работы над проектом запускайте запрос по мере необходимости, обновляя таблицу новыми запросами, появившимися в системе.
2. После выполнения запроса ***з\_ОбновитьРеестрЗапросов,*** заполните пустые поля в таблице ***т\_ВсеЗапросы.*** Использование этой таблицы будет рассмотрено нижев лабо-раторной работе «Создание форм»
3. Организовать передачу результатов запроса в другие приложения
   * 1. Слияние в MS Office Word
     2. Публикация в MS Office Word
     3. Анализ в MS Office Excel

## Содержание отчета

Все созданные запросы записать в таблицу и пронумеровать.



***Описание запросов***

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Имя запроса** | **Текст запроса** | **Запрос на SQL** |
| 1 | зп\_АктерСпект | Для данного спектакля выдать списки всех актеров, принимающих в нем участие  *(одиночный)* | SELECT Спектакль.Название, Роль.Роль, Служащий.ФИО  FROM Спектакль INNER JOIN (Служащий INNER JOIN (Актер INNER JOIN Роль ON Актер.[#Актер] =Роль.[#Служ]) ON Служащий.[Код служащего]=Актер.[#Актер]) ON Спектакль.[#Сп]=Роль.[#Сп]  WHERE Спектакль.Название =[Введите название спектакля]  ORDER BY Служащий.ФИО; |
| 2 | з\_СпектСезона | Для всех актеров выдать списки спектаклей, в которых они принимают участие в данном сезоне  (*множественный*) | SELECT Служащий.ФИО, [Спектакли текущего сезона].Название, Роль.Роль  FROM ((Служащий INNER JOIN Актер ON Служащий.[Код служащего] = Актер.[#Актер]) INNER JOIN Роль ON Актер.[#Актер] = Роль.[#Актер]) INNER JOIN [Спектакли текущего сезона] ON Роль.[#Сп] = [Спектакли текущего сезона].[#Сп]  ORDER BY Служащий.ФИО; |
| 3 | з\_БилетыЗаказа | Для каждой строки заказа выдать список всех заказанных билетов  *(множественный*) | SELECT Клиент.[Кто заказывает], [Заказ билетов].[Дата заказа], Спектакль.Название, [Строка заказа].[Категория билета], [Строка заказа].[Количество билетов]  FROM Спектакль INNER JOIN (Репертуар INNER JOIN (Клиент INNER JOIN ([Заказ билетов] INNER JOIN [Строка заказа] ON [Заказ билетов].[Номер заказа] = [Строка заказа].[Номер заказа]) ON Клиент.Шифр\_клиента = [Заказ билетов].Клиент) ON Репертуар.[#Р] = [Строка заказа].[Строка репертуара]) ON Спектакль.[#Сп] = Репертуар.[#Сп]; |

**Примечание.** В тексте запроса важно отметить, с какого объекта начинается его выполнение и в какой последовательности надлежит переходить от одних объектов к другим для удовлетворения запроса. *Одиночный запрос* выполняется для определенного экземпляра исходного объекта. *Множественный* запрос выполняется многократно для всех экземпляров исходного объекта или некоторого его подмножества.

В приведенной ниже таблице отразить соответствие между разработанными запросами и видами запросов.



***Виды запросов***

|  |  |  |
| --- | --- | --- |
| **№** | **Вид запроса** | **Шифр (ы)** |
| 1 | Запросы с выражениями, использующие   1. операторы строковых значений 2. все операторы сравнения с образцом 3. операторы обработки дат 4. операторы условных выражений 5. статистические функции по подмножеству 6. комплексные текстовые константы и константы даты | + |
| 2 | Использование логических операторов, включающих в себя комбинацию «И», «ИЛИ» | + |
| 3 | Запрос с вычисляемым полем | + |
| 4 | Запрос с предикатом TOP | + |
| 5 | Запрос с внутренним соединением по нескольким полям | + |
| 6 | Запрос с внутренним неэквивалентным соединением | + |
| 7 | Запрос с использованием внешнего соединения (LEFT, RIGHT) | + |
| 8 | Запрос с self-соединением (самообъединение) | + |
| 9 | Подзапросы   * вложенный подзапрос * коррелированный подзапрос |  |
| 10 | Запрос с оператором UNION | + |
| 12 | Запрос с предикатом EXISTS | + |
| 13 | Запрос с сохраненным именованным запросом как источником исходных данных |  |
| 14 | Перекрестный запрос | + |
| 15 | Запросы с параметрами |  |
| 16 | Запросы на группирование данных   * Вычислений итоговых значений по всем записям * Вычислений итоговых значений по группам записей |  |
| 17 | Запросы на удаление записи | + |
| 18 | Запросы на обновление записи | + |
| 19 | Запросы на добавление записей | + |
| 20 | Запросы на создание таблицы | + |

## 

## Контрольные вопросы

1. Назовите две основные формы существования запроса.
2. Как отобразить в запросе только максимальные или минимальные значения?
3. Как отобразить в запросе только заданный процент записей от их общего числа ?
4. Как отобразить в запросе только уникальные записи ?
5. Пусть вы хотите одним способом проводить вычисления для одних записей и другим способом – для прочих. Существует ли способ выполнения в запросе вычислений по подобному условию?
6. Как различать значения **Null** и пустые строки с помощью выражения?
7. Как обрабатываются пустые поля в запросе?
8. Какие подстановочные знаки можно использовать в ACCESS?
9. Дайте характеристику основным типам объединения: внутреннее, внешнее, по несовпадению.
10. Что такое автоподстановка значений и как ее использовать в запросах?
11. Какие запросы в ACCESS называются *запросами SQL*?
12. Как создать таблицу на основе другой таблицы с помощью запроса?
13. Как отобразить в запросе в режиме таблицы только определенное число или определенную долю записей, содержащих максимальные или минимальные значения поля?
14. Как установить формат отображения данных для поля в запросе?
15. Приведите примеры выражений , используемых в запросах для обработки дат, текстовых значений, выполнения арифметических операций и т.д.
16. Как установить новое имя поля и дать имя вычисляемому полю в бланке запроса?
17. Что такое запрос на автоподстановку значений? Где можно использовать такие запросы?
18. Перечислите правила формирования условий в нескольких ячейках **Условие отбора** с помощью операторов **And** или **Or**
19. Как используются **з**апросы на создание таблицы?
20. Как в запросе создать полное или декартово объединение?

## Дополнительное задание

1. Слияние с документами MS Word (*Сервис|Свяи с Office | Слияние с MS Word*)
2. Использование типа данных для параметров в перекрестных запросах, запросах для диаграммы, запросах для логических полей

## ПРИЛОЖЕНИЕ

**Содержание**

* [Описание системных таблиц MS Access](#_Описание_системных_таблиц_MS Access)
* [Перекрестные запросы и их использование](#_Перекрестные_запросы_и_их использов)
* [Подчиненные запросы SQL](#_Подчиненные_запросы_SQL)

**Новиков Сергей**

## Описание системных таблиц MS Access

В этой статье хочу рассказать о системных таблицах MS Access - для чего они нужны, что в них содержиться и какую (порой весьма интересную) информацию можно из них извлечь. Поскольку, к сожалению, я нигде не смог найти описаний этих таблиц приведенная ниже информация была получена мной исключительно в процессе экспериментов над ними. Так что не исключено, что есть какие-то неточности или ошибочные выводы. Ну и, естественно, ни в коем разе не претендую на полноту изложенной информации.

MS Access имеет семь ситемных таблиц: **MSysACEs, MSysCmdbars, MSysModules, MSysModules2, MSysObjects, MSysQueries и MSysRelationships.**

*Таблица* ***MSysObjects.***

На мой взгляд - самая "важная" таблица. Содержит информацию об объектах базы данных.

Эта таблица имеет следующие поля:

|  |  |
| --- | --- |
| **Имя поля** | **Тип поля** |
| **Connect** | Поле МЕМО |
| **Database** | Поле МЕМО |
| **DateCreate** | Дата/время |
| **DateUpdate** | Дата/время |
| **Flags** | Числовой |
| **ForeignName** | Текстовый (255) |
| **Id** | Числовой (Long Int) |
| **Lv** | Поле объекта OLE |
| **LvExtra** | Поле объекта OLE |
| **LvModule** | Поле объекта OLE |
| **LvProp** | Поле объекта OLE |
| **Name** | Текстовый (255) |
| **Owner** | Двоичный (255) |
| **ParentId** | Числовой (Long Int) Prim.key |
| **RmtInfoLong** | Поле объекта OLE |
| **RmtInfoShort** | Двоичный (255) |
| **Type** | Числовой (Integer) |

**Id -** ключевое поле. Содержит уникальный идентификатор для каждого объекта БД.

**ParentId -** значение *Id* "родительского" объекта. Например, если объект "*Forms*" имеет Id = -2147483648, то все формы будут иметь "-2147483648" в поле *ParentId*. Стоит заметить, что у всех запросов и таблиц БД один и тот же "родитель" - "*Tables*".

**Type** - тип объекта БД. Вот значения этого поля для некоторых типов объектов:

|  |  |
| --- | --- |
| **Тип объекта БД** | **Значение поля Type** |
| "Родные" таблицы (в том числе и системные) | **1** |
| Глобальные "семейства" объектов (Relationships, Databases, Tables, Modules, SysRel, Scripts, Forms, Reports) | **3** |
| Запросы (в том числе и те, которые на самом деле являются SQL-выражениями в источниках данных форм, отчетов, элементов управления и т.д.) | **5** |
| Прилинкованные таблицы | **6** |
| Формы | **-32768** |
| Отчеты | **-32764** |
| Макросы | **-32766** |
| Модули (в том числе модули классов) | **-32761** |

**Name** - имя объекта. Для запросов, которые на самом деле являются SQL-выражениями источников записей, это имя стоится по определенным правилам:

* Источник записей формы (префикс ~sql\_f): **~sql\_f***ИмяФромы*Для источника записей формы с именем *Form1* это имя будет **~sql\_fForm1**
* Источник записей отчета (префикс ~sql\_r): **~sql\_r***ИмяОтчета*Для источника записей отчета с именем *Report1* это имя будет **~sql\_rReport1**
* Источник строк элемента управления (например списка) формы (префикс ~sql\_c): **~sql\_c***ИмяФормы***~sql\_c***ИмяЭлементаУправления*  
  Для элемента управления с именем *MyListBox1* в форме с именем *Form1* это имя будет **~sql\_cForm1~sql\_cMyListBox1**
* Источник строк элемента управления (например списка) отчета (префикс ~sql\_d): **~sql\_d***ИмяОтчета***~sql\_d***ИмяЭлементаУправления*  
  Для элемента управления с именем *MyListBox1* в отчете с именем *Report1* это имя будет **~sql\_dReport1~sql\_dMyListBox1**

**ForeignName** - для прилинкованных таблиц в этом поле содержиться "реальное" имя таблицы. Например, если таблица с имем *Table1* прилинкована с именем *MyTable1*, то в поле Name будет *MyTable1*, а в поле *ForeignName* - *Table1*. А для прилинкованного документа *Excel* в этом поле будет имя листа - например *Лист1$*.

**Database -** для прилинкованных таблиц в этом поле содержиться полный путь и имя файла, откуда была прилинкована таблица. Для прилинкованных Access-ных таблиц это будет, например, *D:\DBases\MyBase.mdb*, а для Excel-еских - *C:\MyDocuments\Raspisanie.xls*.

**Connect -** для прилинкованныз НЕ-Access-ных таблиц в этом поле содержиться значение свойства Connect. Например, для Excel это будет *Excel 5.0;HDR=NO;IMEX=2;*

**DateCreate -** дата и время создания объекта.

**DateUpdate -** дата и время обновления объекта.

Для чего нужны остальные поля этой таблицы - не знаю.

А теперь посмотрим, какие данные мы можем получить из этой таблицы:

|  |  |
| --- | --- |
| **Что можем получить** | **SQL** |
| Список "родных" таблиц | **SELECT** MSysObjects.Id, MSysObjects.Name **FROM** MSysObjects **WHERE** (((MSysObjects.Type)=1) AND ((Left([Name],4))<>"MSys" And (Left([Name],4))<>"USys")); |
| Список присоединенных таблиц | **SELECT** MSysObjects.Id, MSysObjects.Name **FROM** MSysObjects **WHERE** (((MSysObjects.Type)=6) AND ((Left([Name],4))<>"MSys" And (Left([Name],4))<>"USys")); |
| Список "родных" и присоединенных таблиц | **SELECT** MSysObjects.Id, MSysObjects.Name, MSysObjects.Type **FROM** MSysObjects **WHERE** (((MSysObjects.Type)=1 Or (MSysObjects.Type)=6) AND ((Left([Name],4))<>"MSys" And (Left([Name],4))<>"USys")); |
| Список запросов | **SELECT** MSysObjects.Id, MSysObjects.Name **FROM** MSysObjects WHERE (((Left([Name],4))<>"~sq\_") AND ((MSysObjects.Type)=5)); |
| Список форм | **SELECT** MSysObjects.Id, MSysObjects.Name **FROM** MSysObjects **WHERE** (((MSysObjects.Type)=-32768)); |
| Список отчетов | **SELECT** MSysObjects.Id, MSysObjects.Name **FROM** MSysObjects **WHERE** (((MSysObjects.Type)=-32764)); |
| Список макросов | **SELECT** MSysObjects.Id, MSysObjects.Name **FROM** MSysObjects **WHERE** (((MSysObjects.Type)=-32766)); |
| Список модулей | **SELECT** MSysObjects.Id, MSysObjects.Name **FROM** MSysObjects **WHERE** (((MSysObjects.Type)=-32761)); |
| Список форм, имеюших выражение SQL в источнике | **SELECT** MSysObjects\_1.Id AS [Id Форм], Right([MSysObjects]Name],Len([MSysObjects]![Name])-5) AS [Имя формы], MSysObjects.Id AS [Id Запроса] **FROM** MSysObjects, MSysObjects AS MSysObjects\_1 **WHERE** (((Right([MSysObjects]![Name],Len([MSysObjects]![Name])-5))= [MSysObjects\_1]![Name]) AND ((Left([MSysObjects]![Name],5))="~sq\_f") AND ((MSysObjects.Type)=5) AND ((MSysObjects\_1.Type)=-32768)); |
| Список отчетов, имеюших выражение SQL в источнике | **SELECT** MSysObjects\_1.Id AS [Id Отчета], Right([MSysObjects]![Name],Len([MSysObjects]![Name])-5) AS [Имя отчета], MSysObjects.Id AS [Id Запроса] **FROM** MSysObjects, MSysObjects AS MSysObjects\_1 **WHERE** (((Right([MSysObjects]![Name], Len([MSysObjects]![Name])-5))=[MSysObjects\_1]![Name]) AND ((Left([MSysObjects]![Name],5))="~sq\_r") AND ((MSysObjects.Type)=5) AND ((MSysObjects\_1.Type)=-32764)); |
| Список элементов управления форм, имеюших выражение SQL в источнике | **SELECT** MSysObjects\_1.Id, Mid([MSysObjects]![Name],6,InStr(2,[MSysObjects]![Name],"~sq\_c")-6) AS [Имя формы], Mid([MSysObjects]![Name],InStr(2,[MSysObjects]![Name],"~sq\_c")+5) AS [Имя элемента] **FROM** MSysObjects AS MSysObjects\_1, MSysObjects **WHERE** (((Mid([MSysObjects]![Name],6,InStr(2,[MSysObjects]![Name], "~sq\_c")-6))=[MSysObjects\_1]![Name]) AND ((Left([MSysObjects]![Name],5))="~sq\_c") AND ((MSysObjects.Type)=5) AND ((MSysObjects\_1.Type)=-32768)); |
| Список элементов управления отчетов, имеюших выражение SQL в источнике | **SELECT** MSysObjects\_1.Id, Mid([MSysObjects]![Name],6,InStr(2,[MSysObjects]![Name],"~sq\_d")-6) AS [Имя отчета], Mid([MSysObjects]![Name],InStr(2,[MSysObjects]![Name],"~sq\_d")+5) AS [Имя элемента] **FROM** MSysObjects, MSysObjects AS MSysObjects\_1 **WHERE** (((Mid([MSysObjects]![Name],6,InStr(2,[MSysObjects]![Name], "~sq\_d")-6))=[MSysObjects\_1]![Name]) AND ((Left([MSysObjects]![Name],5))="~sq\_d") AND ((MSysObjects.Type)=5) AND ((MSysObjects\_1.Type)=-32764)); |

Подчиненный запрос представляет собой инструкцию SQL SELECT, вложенную в запрос на выборку или запрос на изменение. Чтобы определить новое поле, данную инструкцию можно ввести в строку **Поле** в бланке запроса. Чтобы указать для данного поля условие отбора, введите инструкцию в строку **Условие отбора**. Подчиненные запросы используются для следующего:

* проверка наличия результатов подчиненного запроса (используются зарезервированные слова EXISTS или NOT EXISTS);
* поиск значений в основном запросе, которые равны, превышают или меньше значений, возвращаемых подчиненным запросом (используются зарезервированные слова ANY, IN или ALL);
* создания подчиненных запросов внутри подчиненных запросов (вложенные запросы)

[![Изображение выглядит как текст

Автоматически созданное описание](#_ПРИЛОЖЕНИЕ)

## Перекрестные запросы и их использование

В перекрестном запросе отображаются результаты статистических расчетов (такие как суммы, количество записей и средние значения), выполненных по данным из одного поля таблицы. Эти результаты группируются по двум наборам данных, один из которых расположен в левом столбце таблицы, а второй - в верхней строке.

### TRANSFORM *статФункция*

### *инструкция*

### PIVOT *поле [*IN *(значение\_1[, значение\_2[, ...]])]*

Ниже перечислены аргументы инструкции TRANSFORM:

### Элемент Описание

### *статФункция* Статистическая функция SQL, обрабатывающая указанные данные.

### *инструкция* Инструкция SELECT.

### *поле* Поле или выражение, которое содержит заголовки столбцов для результирующего набора.

### *значение*\_1, *значение*\_2 Фиксированные значения, используемые при создании заголовков столбцов.

### Дополнительные сведения

Когда данные сгруппированы с помощью перекрестного запроса, можно выбирать значения из заданных столбцов или выражений как заголовки столбцов. Это позволяет просматривать данные в более компактной форме, чем при работе с запросом на выборку.

Инструкция TRANSFORM является необязательной, однако, если она присутствует, то должна быть первой инструкцией в строке SQL. Она должна находиться впереди инструкции SELECT. Инструкция SELECT указывает поля, содержащие заголовки строк, и предложения GROUP BY, которое задает группировку по строкам. При желании можно включить и другие предложения, например, WHERE, для описания дополнительных условий отбора и сортировки. Кроме того, можно использовать подчиненные запросы как предикаты в перекрестном запросе, особенно в предложении WHERE.

### Пример

1. Запрос: **«**Запрос для сводной таблицы**»**

SELECT Репертуар.Дата, Спектакль.Жанр, Спектакль.Название, Выручка.[Продано билетов], Билет.Категория, Билет.Цена

FROM Спектакль INNER JOIN (Репертуар INNER JOIN (Билет INNER JOIN Выручка ON Билет.[#Бил] = Выручка.[#Бил]) ON Репертуар.[#Р] = Выручка.[#Р]) ON Спектакль.[#Сп] = Репертуар.[#Сп];

1. Перекрестный запрос

TRANSFORM Sum([Запрос для сводной таблицы].[Продано билетов]) AS [Sum-Продано билетов]

SELECT [Запрос для сводной таблицы].Название, [Запрос для сводной таблицы].Жанр, Sum([Запрос для сводной таблицы].[Продано билетов]) AS [Всего продано]

FROM [Запрос для сводной таблицы]

GROUP BY [Запрос для сводной таблицы].Название, [Запрос для сводной таблицы].Жанр

PIVOT [Запрос для сводной таблицы].Категория;

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Название** | **Жанр** | **Всего продано** | **Амфитеатр** | **Балкон** | **Ложа** | **Партер** |
| Дон Кихот | Балет | 322 | 60 | 73 | 74 | 115 |
| Жизель | Балет | 190 | 36 | 24 | 47 | 83 |
| Князь Игорь | Опера | 175 | 15 | 87 | 56 | 17 |
| Марица | Оперетта | 562 | 149 | 104 | 145 | 164 |
| Риголетто | Опера | 275 | 57 | 77 | 55 | 86 |
| Свадьба в Малиновке | Оперетта | 438 | 74 | 62 | 119 | 183 |
| Сильва | Оперетта | 788 | 213 | 170 | 300 | 105 |
| Травиата | Опера | 224 | 56 | 82 | 12 | 74 |

### Инструкция TRANSFORM, примеры

Использованиепредложения SQL TRANSFORM для создания перекрестного запроса, демонстрирующего число заказов, принятых каждым сотрудником в каждом квартале 1994 г. Для выполнения данной процедуры требуется функция SQLTRANSFORMOutput***.***

Sub TransformX1()

Dim dbs As Database

Dim strSQL As String

Dim qdfTRANSFORM As QueryDef

strSQL = "PARAMETERS prmYear SHORT; TRANSFORM " \_

& "Count(КодЗаказа) " \_

& "SELECT Имя & "" "" & Фамилия AS " \_

& "ФИО FROM Сотрудники INNER JOIN Заказы " \_

& "ON Сотрудники.КодСотрудника = " \_

& "Заказы.КодСотрудника WHERE DatePart " \_

& "(""yyyy"", ДатаРазмещения) = [prmYear] "

strSQL = strSQL & "GROUP BY Имя & " \_

& """ "" & Фамилия " \_

& "ORDER BY Имя & "" "" & Фамилия " \_

& "PIVOT DatePart(""q"", ДатаРазмещения)"

' Укажите в следующей строке путь к базе данных "Борей"

' на вашем компьютере.

Set dbs = OpenDatabase("Борей.mdb")

Set qdfTRANSFORM = dbs.CreateQueryDef \_

("", strSQL)

SQLTRANSFORMOutput qdfTRANSFORM, 1994

dbs.Close

End Sub

В следующем примере предложение SQL TRANSFORM используется для создания более сложного перекрестного запроса, демонстрирующего общую сумму заказов, принятых каждым сотрудником в каждом квартале 1994 г. Для выполнения данной процедуры требуется функция SQLTRANSFORMOutput.

Sub TransformX2()

Dim dbs As Database

Dim strSQL As String

Dim qdfTRANSFORM As QueryDef

strSQL = "PARAMETERS prmYear SHORT; TRANSFORM " \_

& "Sum(Всего) SELECT Имя & "" """ \_

& "& Фамилия AS ФИО " \_

& "FROM Сотрудники INNER JOIN " \_

& "(Заказы INNER JOIN [Сумма заказов] " \_

& "ON Заказы.КодЗаказа = " \_

& "[Сумма заказов].КодЗаказа) " \_

& "ON Сотрудники.КодСотрудника = " \_

& "Заказы.КодСотрудника WHERE DatePart" \_

& "(""yyyy"", ДатаРазмещения) = [prmYear] "

strSQL = strSQL & "GROUP BY Имя & "" """ \_

& "& Фамилия " \_

& "ORDER BY Имя & "" "" & Фамилия " \_

& "PIVOT DatePart(""q"",ДатаРазмещения)"

' Укажите в следующей строке путь к базе данных "Борей"

' на вашем компьютере.

Set dbs = OpenDatabase("Борей.mdb")

SQLTRANSFORMOutput qdfTRANSFORM, 1994

Set qdfTRANSFORM = dbs.CreateQueryDef \_

("", strSQL)

dbs.Close

End Sub

Function SQLTRANSFORMOutput(qdfTemp As QueryDef, \_

intYear As Integer)

Dim rstTRANSFORM As Recordset

Dim fldLoop As Field

Dim booFirst As Boolean

qdfTemp.PARAMETERS!prmYear = intYear

Set rstTRANSFORM = qdfTemp.OpenRecordset()

Debug.Print qdfTemp.SQL

Debug.Print

Debug.Print , , "Квартал"

With rstTRANSFORM

booFirst = True

For Each fldLoop In .Fields

If booFirst = True Then

Debug.Print fldLoop.Name

Debug.Print , ;

booFirst = False

Else

Debug.Print , fldLoop.Name;

End If

Next fldLoop

Debug.Print

Do While Not .EOF

booFirst = True

For Each fldLoop In .Fields

If booFirst = True Then

Debug.Print fldLoop

Debug.Print , ;

booFirst = False

Else

Debug.Print , fldLoop;

End If

Next fldLoop

Debug.Print

.MoveNext

Loop

End With

End Function

Значения, возвращенные аргументом поле, используются в качестве заголовков столбцов в результирующем наборе запроса. Например, продажи по месяцам в перекрестном запросе образуют результирующую таблицу из 12 столбцов. Аргумент поле можно ограничить, чтобы создать заголовки из фиксированных значений (значение\_1, значение\_2 ), перечисленных в необязательном предложении IN. Кроме того, фиксированные заголовки, которым не соответствуют реальные данные, можно использовать для создания дополнительных столбцов.

### *Совет.* Существует возможность вывести данные в перекрестной таблице без создания в базе данных отдельного запроса. Для этого следует использовать мастер сводных таблиц. В сводной таблице пользователь имеет возможность изменять заголовкистрок или столбцов, что позволяет анализировать данные различными способами.

[Изображение выглядит как текст

Автоматически созданное описание](#_Основное_задание)

## Подчиненные запросы SQL

Подчиненным запросом называют инструкцию SELECT, вложенную в инструкцию SELECT, SELECT...INTO, INSERT...INTO, DELETE или UPDATE или в другой подчиненный запрос.

**Синтаксис**

Подчиненный запрос создается одним из трех способов:

1. *сравнение* [ANY | ALL | SOME] (*инструкцияSQL*)
2. *выражение* [NOT] IN (*инструкцияSQL*)
3. [NOT] EXISTS (*инструкцияSQL*)

Ниже перечислены аргументы подчиненного запроса:

|  |  |
| --- | --- |
| **Элемент** | **Описание** |
| *сравнение* | Выражение и оператор сравнения, который сравнивает выражение с результатами подчиненного запроса. |
| *выражение* | Выражение, для которого проводится поиск в результирующем наборе записей подчиненного запроса. |
| *инструкцияSQL* | Инструкция SELECT, которая соответствует формату и всем правилам, принятым для инструкций SELECT. Она должна быть заключена в круглые скобки. |

**Дополнительные сведения**

Подчиненный запрос можно использовать вместо выражения в списке полей инструкции SELECT или в предложениях WHERE и HAVING. Инструкция SELECT используется в подчиненном запросе для задания набора конкретных значений, вычисляемых в выражениях предложений WHERE или HAVING.

Предикаты ANY или SOME, являющиеся синонимами, используются для отбора записей в главном запросе, которые удовлетворяют сравнению со всеми записями, отобранными в подчиненном запросе. В следующем примере отбираются все товары, цена которых больше, чем цена любого товара, проданного со скидкой в 25 процентов или более:

SELECT \* FROM Товары

WHERE Цена > ANY

(SELECT Цена FROM Заказано

WHERE Скидка >= .25);

Предикат ALL используется для отбора в главном запросе только тех записей, которые удовлетворяют сравнению со всеми записями, отобранными в подчиненном запросе. Если в предыдущем примере предикат ANY заменить предикатом ALL, результат запроса будет включать только те товары, чья цена больше, чем цена всех товаров, проданных со скидкой 25 или более. Это условие является значительно более жестким.

Предикат IN используется для отбора в главном запросе только тех записей, которые содержат значения, совпадающие с одним из отобранных подчиненным запросом. Следующий пример возвращает все товары, проданные со скидкой, большей или равной 25 процентам:

SELECT \* FROM Товары

WHERE КодТовара IN

(SELECT КодТовара FROM Заказано

WHERE Скидка >= .25.

И наоборот, предикат NOT IN используется для отбора в главном запросе только тех записей, которые содержат значения, не совпадающие ни с одним из отобранных подчиненным запросом.

Предикат EXISTS (с необязательным зарезервированным словом NOT) используется в логическом выражении для определения того, должен ли подчиненный запрос возвращать какие-либо записи.

В подчиненном запросе можно использовать псевдонимы таблиц для ссылки на таблицы, перечисленные в предложении FROM, расположенном вне подчиненного запроса. В следующем примере отбираются фамилии и имена сотрудников, чья зарплата равна или больше средней зарплаты сотрудников, имеющих ту же должность. В данном примере таблица «Сотрудники» получает псевдоним «Т1»:

SELECT Фамилия,

Имя, Должность, Оклад

FROM Сотрудники AS T1

WHERE Оклад >=

(SELECT Avg(Оклад)

FROM Сотрудники

WHERE T1.Должность = Сотрудники.Должность) Order by Должность;

В последнем примере зарезервированное слово не является обязательным.

Некоторые подчиненные запросы можно использовать в [перекрестных запросах](JavaScript:hhobj_4.Click()) как предикаты (в предложении WHERE). Подчиненные запросы, используемые для вывода результатов (в списке SELECT), нельзя использовать в перекрестных запросах.

**Подчиненные запросы SQL, пример**

Создание списка имен и сведений о каждом клиенте, разместившем заказ во втором квартале 1995 г.

В данном примере вызывается процедура EnumFields, текст которой приведен в примере для инструкции SELECT.

Sub SubQueryX()

    Dim dbs As Database, rst As Recordset

    ' Укажите в следующей строке путь к базе данных "Борей"

    ' на вашем компьютере.

    Set dbs = OpenDatabase("Борей.mdb")

    ' Создание списка имен и сведений о каждом клиенте,

    ' разместившем заказ во втором квартале 1995 г.

    Set rst = dbs.OpenRecordset("SELECT Фамилия," \_

        & " Название, Должность, Телефон" \_

        & " FROM Клиенты" \_

        & " WHERE КодКлиента" \_

        & " IN (SELECT КодКлиента FROM Заказы" \_

        & " WHERE ДатаРазмещения Between #04/1/95#" \_

        & " And #07/1/95#);")

        ' Заполняет набор записей.

    rst.MoveLast

    ' Вызывает процедуру EnumFields для печати содержимого

    ' объекта Recordset. Передает объект Recordset и

    ' требуемую ширину полей.

    EnumFields rst, 25

    dbs.Close

End Sub

[Изображение выглядит как текст

Автоматически созданное описание](#_ПРИЛОЖЕНИЕ)