

С. В. Тихов
А. И. Кушнеров

БАЗЫ ДАННЫХ

Практикум

Санкт-Петербург
2022

Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
**«Санкт-Петербургский государственный университет
промышленных технологий и дизайна»
Высшая школа технологии и энергетики**

**С. В. Тихов
А. И. Кушнеров**

БАЗЫ ДАННЫХ

Практикум

Утверждено Редакционно-издательским советом ВШТЭ СПбГУПТД

Санкт-Петербург
2022

УДК 681.3(075)
ББК 32.97я7
Т 462

Рецензент

кандидат технических наук, доцент Высшей школы технологии и энергетики
Санкт-Петербургского государственного университета промышленных технологий и дизайна
П. Е. Антонюк

Тихов, С. В.

Т 462 Базы данных: Практикум / С. В. Тихов, А. И. Кушнеров. — СПб.: ВШТЭ
СПбГУПТД, 2022. — 67 с.

Практикум соответствует программам и учебным планам дисциплины «Базы данных» для студентов очной формы обучения по направлению подготовки 01.03.02 «Прикладная математика и информатика». В практикуме представлены теоретические основы баз данных и практические задания для выполнения в компьютерных программах MS Access и MySQL Server.

УДК 681.3(075)
ББК 32.97я7

© ВШТЭ СПбГУПТД, 2022
© Тихов С. В., Кушнеров А. И., 2022

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	4
1. ОСНОВЫ БАЗЫ ДАННЫХ В MS ACCESS	5
2. ПРАКТИЧЕСКИЕ ЗАДАНИЯ ПО MS ACCESS	10
Практическое занятие № 1. Создание таблиц в режиме таблицы и с помощью конструктора	10
Практическое занятие № 2. Создание и изменение свойств полей	13
Практическое занятие № 3. Создание базы данных, состоящей из трех таблиц.....	22
Практическое занятие № 4. Создание и редактирование формы для ввода данных	26
Практическое занятие № 5. Обработка данных с помощью запросов	31
Практическое занятие № 6. Создание и редактирование отчетов	36
Практическое занятие № 7. Учет документов в базе данных.....	43
3. ОСНОВЫ БАЗЫ ДАННЫХ В MYSQL	51
3.1. Введение и теоретическая часть	51
3.2. Технические возможности СУБД MySQL.....	52
3.3. Типы данных. Обработка числовых данных	52
3.4. Описание СУБД MySQL.....	53
3.5. Реализация языка SQL в СУБД MySQL	55
БИБЛИОГРАФИЧЕСКИЙ СПИСОК.....	67

ВВЕДЕНИЕ

Актуальность дисциплины «Базы данных» обусловлена высоким спросом на разработку методов и программных средств для конструирования и использования прикладных информационных систем с учетом высоких нагрузок и при больших объемах данных. При этом все больше возрастает потребность в высококвалифицированных специалистах в области работы с базами данных: программистах, проектировщиках, системных администраторах и других смежных специалистах [4].

Практикум разработан в соответствии с программой курса «Базы данных» Федерального государственного образовательного стандарта для бакалавров по направлению подготовки 01.03.02 «Прикладная математика и информатика» с целью сформировать у студентов необходимые компетенции и навыки.

В первом разделе практикума представлены теоретические основы баз данных, во втором и третьем – практические задания для выполнения в компьютерных программах MS Access и MySQL Server соответственно.

1. ОСНОВЫ БАЗЫ ДАННЫХ В MS ACCESS

База данных Microsoft Access – это совокупность данных и объектов (т. е. форм, отчетов и т. п.), относящихся к определенной задаче и представляющих законченную систему. Базу данных Access составляют таблицы, запросы, формы, отчеты, страницы доступа, макросы и модули. Кроме того, приложение Access содержит некоторые другие объекты, в том числе связи, свойства базы данных и спецификации импорта и экспорта.

Таблица – объект базы данных, в котором данные хранятся в виде записей (строк) и полей (столбцов). Является основным структурным элементом системы управления реляционной базой данных.

Запрос – объект базы данных, позволяющий осуществлять поиск и вывод данных, хранящихся в таблицах, удовлетворяющих заданным условиям (в том числе из нескольких таблиц). С помощью запроса можно модифицировать и удалять записи таблиц, а также выполнять различные вычисления.

Форма – объект базы данных, являющийся элементом пользовательского интерфейса, предназначенный для просмотра, ввода и модификации данных в одной или более таблицах.

Отчет – объект базы данных, предназначенный для анализа и вывода на печать данных, организованных и отформатированных в соответствии с требованиями пользователя.

Макрос – макрокоманда или набор макрокоманд, используемый для автоматизации задач.

Модуль – объект базы данных, который позволяет создавать библиотеки подпрограмм и функций, используемых во всем приложении. Используя коды модулей, можно решать такие задачи, как обработка ошибок ввода, объявление и применение переменных, организация циклов и т. п. [3].

Проектирование базы данных (БД) состоит из двух основных фаз: логического и физического моделирования. Во время фазы логического моделирования конструктор собирает требования и разрабатывает модель, не зависящую от конкретной системы управления базами данных (СУБД). Во время фазы физического моделирования конструктор создает модель, оптимизированную для конкретного приложения СУБД; именно эта модель реализуется на практике.

В табл. 1 приведен список предусмотренных в приложении Microsoft Office Access типов данных, инструкции по их применению и сведения о размере места, необходимого для хранения данных каждого типа. При этом важно помнить, что максимальный размер файла базы данных составляет 2 гигабайта.

В теории реляционных баз данных таблица представляет собой изначально неупорядоченный набор записей. Единственный способ идентифицировать определенную запись в этой таблице – это указать набор атрибутов (свойств полей), который был бы уникальным для этой записи.

Таблица 1 – Типы полей Microsoft Office Access 2010

Тип данных	Применение	Размер
Короткий текст	Используется для буквенно-цифровых символов, включая текст, а также текст и числа, не применяемые в вычислениях.	До 255 символов
Длинный текст (Поле МЕМО)	Используется для текста, размер которого превышает 255 символов, или для текста, в котором используется форматирование. Свойство поля указывает, поддерживает ли поле форматированный текст.	До одного гигабайта, но в элементах управления отражаются только первые 64000 символов.
Числовой	Применяется для хранения числовых значений, которые используются в вычислениях (за исключением денежных сумм).	1, 2, 4, 8 или 12 байтов
Дата / время	Используется для хранения значений дат и времени.	8 байтов
Денежный	Используется для хранения денежных значений. Используется для предотвращения округлений во время вычислений.	8 байтов
Счетчик	Используется для формирования уникальных значений, которые могут применяться в качестве первичного ключа. Эти значения автоматически вставляются в поле при добавлении записи. Поля с типом данных «Счетчик» могут формироваться добавлением единицы, добавлением заданного значения или с помощью случайных чисел.	4 байта

Тип данных	Применение	Размер
Логический	Используется для логических значений: Да/Нет, Истина/Ложь или Вкл/Выкл.	1 бит (0,125 байта).
Поле объекта OLE	Используется для хранения объектов OLE из других программ Microsoft Windows.	До 1 гигабайта.
Гиперссылка	Используется для хранения гиперссылок (в том числе ссылок на объекты приложения Access, которые хранятся в базе данных).	До 1024 x 1024 x 1024 символов, для хранения которых требуется 2 гигабайта (2 байта на символ). Из них можно отображать до 65 535 символов в одном элементе управления.
Мастер подстановок	Используется для запуска мастера подстановок, позволяющего создавать поле, в котором в виде раскрывающегося списка отображаются значения из другой таблицы, запроса или списка значений (фактически мастер подстановок не является типом данных).	Если к полю подстановок присоединена таблица или запрос, то это размер присоединенного столбца. Если к полю подстановок не присоединен другой столбец (т. е. хранится список значений), то это размер текстового поля, используемого для хранения списка.
Вложение	Используется для хранения двоичных файлов (файлов, которые нельзя прочесть с помощью текстового редактора), таких как цифровые изображения (фотографии и другие изображения) или файлов, созданных с помощью других приложений Microsoft Office. Используя поле с типом данных «Вложение», можно вложить в одну запись более одного файла.	2 гигабайта для сжатых вложений. Для несжатых вложений приблизительно 700 КБ, в зависимости от степени сжатия вложений.
Вычисляемый	Для создания вычисляемого поля в таблице.	8 байтов.

Ключом называется набор атрибутов, однозначно определяющий запись. Существуют следующие виды ключей:

– Первичный ключ – представляет собой одно или несколько полей(столбцов), значения которых однозначно определяют каждую запись в таблице. Первичный ключ всегда должен иметь уникальный индекс. Первичный ключ используется для связывания таблицы с внешними ключами в других таблицах. Первичный ключ может быть естественным или искусственным. Ключ, состоящий из информационных полей таблицы (т. е. полей, содержащих полезную информацию об описываемых объектах) называется естественным ключом. Теоретически, естественный ключ всегда можно сформировать, в этом случае он носит название «интеллектуальный ключ». Искусственный ключ – это дополнительное служебное поле, единственное предназначение которого – служить первичным ключом. Значения этого поля не образуются на основе каких-либо других данных из БД, а генерируются искусственно. Как правило, суррогатный ключ – это просто числовое поле, в которое заносятся значения из возрастающей числовой последовательности.

– Внешний ключ – представляет собой одно или несколько полей (столбцов), содержащих ссылку на поле или поля первичного ключа в другой таблице. Внешний ключ определяет способ связи таблиц.

Ключи также делятся на два класса: простые и составные. Простой ключ состоит из одного атрибута, составной ключ состоит из нескольких атрибутов. Применение составных ключей усложняет объединение таблиц.

После создания таблицы для каждой темы в базе данных нужно предоставить приложению Microsoft Access средства, с помощью которых можно будет вновь объединять сведения при необходимости. Это делается путем помещения общих полей в связанные таблицы и определения связей между таблицами. После этого можно создавать запросы, формы и отчеты, одновременно отображающие сведения из нескольких таблиц.

Межтабличные связи могут объединять две и более сущности. Как правило, они соответствуют некоторому взаимодействию между сущностями и описывают связь, возникающую между ними. Во время логического проектирования связи между таблицами могут обладать собственными атрибутами. Такое отношение выделяется в отдельную сущность типа связь.

Отношения делятся на три основных типа, в зависимости от количества записей сущности, связанных с записью другой сущности:

Один – к – одному: каждой записи первой сущности соответствует только одна запись второй сущности, а каждой записи второй сущности – только одна запись первой сущности. Пример: автор, у которого в данный момент имеется лишь одна незавершенная книга.

Один – ко – многим: каждой записи первой сущности могут соответствовать несколько записей второй сущности, однако каждой записи второй сущности соответствует только одна запись первой сущности. Пример: издательство, выпустившее несколько книг.

Многие – ко – многим: каждой записи первой сущности могут соответствовать несколько записей второй сущности, а каждой записи второй сущности соответствуют несколько записей первой сущности. Пример: один автор может написать несколько книг, а у одной книги может быть несколько авторов. В реляционных базах данных этот тип отношений не реализуем, поэтому создается дополнительная сущность, ассоциирующая данную связь (ассоциация).

Связи между сущностями устанавливаются по равенству значений первичного и внешнего ключей. В Microsoft Access межтабличные связи можно создать непосредственно с помощью окна «**Схема данных**» или путем перетаскивания поля из области Список полей. В приложении Access межтабличные связи используются для того, чтобы продемонстрировать, как связать таблицы для использования их в объекте базы данных. Чтобы задать правила целостности данных для конкретной связи, при ее создании в Microsoft Office Access следует установить флажок **Обеспечение целостности данных**. Если данный флажок установлен, то любая попытка выполнить действие, нарушающее одно из перечисленных выше правил, приведет к выводу на экран предупреждения, а само действие будет отменено.

Чтобы преодолеть ограничения на удаление или изменение связанных записей, сохраняя при этом целостность данных, следует установить флажки **Каскадное обновление связанных полей** и **Каскадное удаление связанных полей**. Если установлен флажок Каскадное обновление связанных полей, то при изменении ключевого поля главной таблицы автоматически изменяются и соответствующие значения связанных записей. Если установлен флажок Каскадное удаление связанных полей, то при удалении записи в главной таблице удаляются и все связанные записи в подчиненной таблице.

После создания необходимых таблиц, полей и связей требуется еще раз просмотреть структуру базы данных и выявить возможные недочеты. Желательно это сделать на данном этапе, пока таблицы не заполнены данными. В Microsoft Access существуют инструменты, позволяющие улучшить структуру базы данных, а также получить полную информацию о ее объектах.

Мастер анализа таблиц может проанализировать структуру таблицы, предложить подходящие новые структуры и связи, а также разделить таблицу на новые связанные таблицы, если это имеет смысл.

Анализатор быстрого действия исследует всю базу данных, дает рекомендации по ее улучшению, а также осуществляет их.

Архивариус базы данных анализирует все параметры объектов базы данных, создавая при этом полный отчет о ее структуре [5, 6].

2. ПРАКТИЧЕСКИЕ ЗАДАНИЯ ПО MS ACCESS

Практическое занятие № 1

Создание таблиц в режиме таблицы и с помощью конструктора

Цель занятия: знакомство с СУБД Access и получение практических навыков по созданию таблиц.

Отрабатываемые вопросы

1. Ознакомление с «СУБД «Access».
2. Создание таблицы в режиме таблица.
3. Создание таблицы с помощью конструктора.

Организационно-методические указания

1. Создание таблицы в режиме «Таблица».

Создайте таблицу, содержащую сведения о сотрудниках предприятия. Включите в таблицу поля **Код Сотрудника** (ключевое поле), **Фамилия**, **Имя**, **Отчество**, **Дата Рождения** (табл. 2). Заполните таблицу сведениями о нескольких сотрудниках.

Таблица 2 – «Сотрудники»

Название поля	Тип данных
Код Сотрудника	Счетчик
Фамилия	Короткий текст
Имя	Короткий текст
Отчество	Короткий текст
Дата Рождения	Дата/Время

1.1. Запустите программную среду для создания баз данных Access. После запуска программы нажмите кнопку Пустая база данных рабочего стола (рис. 1). В диалоговом окне сохраните базу данных с именем **ОАО Рангоут** в папке, с тем же именем созданной ранее. Откроется новая база данных, содержащая одну строку и два столбца (столбцы в Access называются полями) (рис. 3). В основном окне среды баз данных Access на вкладке **Создание** выбрать команду **Таблица**.

1.2. Задайте названия полей таблицы. При вводе данных в Access полям присваиваются имена: **Поле1**, **Поле2** и так далее. Можно использовать предложенные имена или изменить их. Имена могут быть до 64 знаков длиной, а при вводе или изменении имени содержимое ячеек не изменяется. Кроме того, столбец может оказаться слишком узким, чтобы имя поля отображалось полностью. В этом случае **Кликните** заголовок столбца, наведите указатель мыши на правую границу столбца, чтобы он принял форму двусторонней стрелки, а затем перетащите границу столбца вправо.

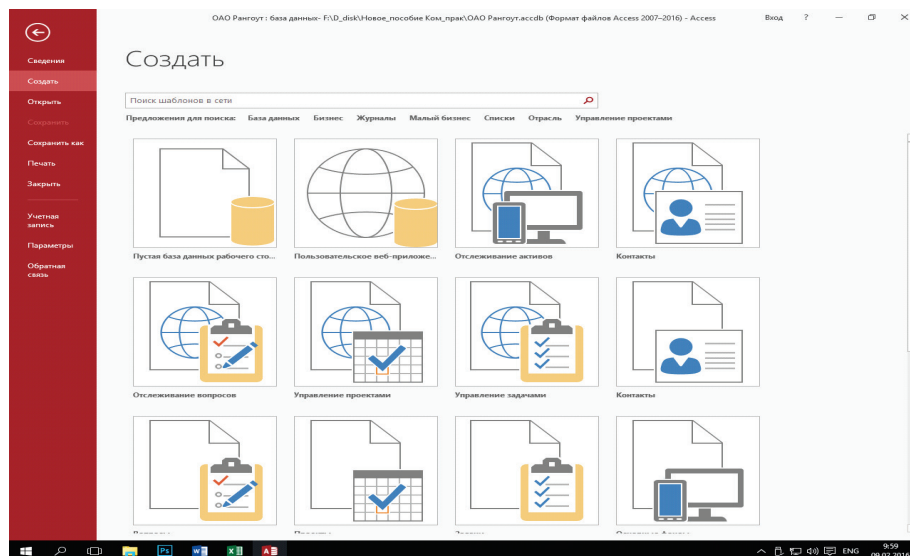


Рис. 1. Запуск Access

1.3. Определите тип полей в режиме **Таблица**. Чтобы задать тип данных, откройте вкладку **Поля** и выберите тип данных (рис. 1). Ключевое поле Access создаст автоматически. Задайте типы полей согласно табл. 1.

***Примечание.** Тип данных – это параметр, с помощью которого задается вид данных, которые можно ввести в каждый столбец таблицы. Например, если в одном столбце нужно хранить значения даты и времени, а в другом – показатели затрат, для первого столбца следует установить тип Дата/время, а для второго – Денежный. Не все доступные типы данных можно задать путем ввода данных в ячейку. Например, если нужно отобразить в столбце фотографии сотрудников, нельзя просто вставить фотографию в новое поле. Необходимо вручную задать тип данных Вложение.*

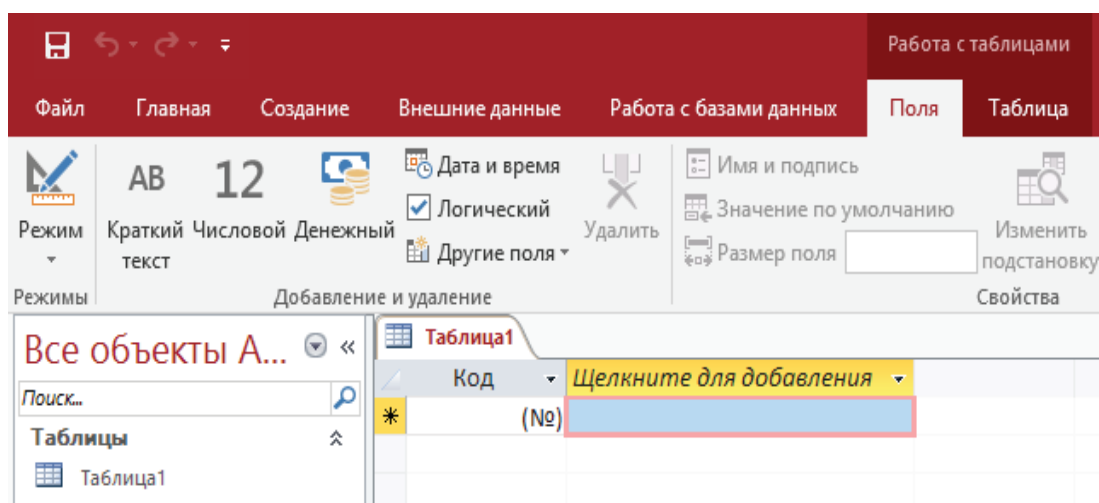


Рис. 2. Введение данных в режиме **Таблица**

1.4. Создайте таблицу, и заполните ее (рис. 2). Первое поле **Код Сотрудника** заполняется автоматически.

Таблица 3 – Заполнение таблицы «Сотрудники»

Фамилия	Имя	Отчество	Дата Рождения
Соколов	Александр	Васильевич	07.09.1975
Белых	Алексей	Иванович	23.03.1965
Мухин	Антон	Павлович	24.05.1961
Ильина	Анна	Васильевна	16.10.1983
Плужников	Дмитрий	Андреевич	15.05.1956
Андреева	Елена	Федоровна	05.01.1961
Бортникова	Алла	Александровна	23.12.1960
Иванова	Ирина	Петровна	30.06.1958
Трофимов	Дмитрий	Евгеньевич	09.10.1960
Борзов	Борис	Александрович	26.04.1967
Петрова	Ирина	Владимировна	05.12.1975

2. Создание таблицы с помощью конструктора.

Создайте таблицу «**Структура ОАО Рангоут**», в которой представлена информация о структурных подразделениях предприятия: название и сокращение (аббревиатура). Для создания таблицы воспользуйтесь режимом конструктора.

2.1. Выберите режим создания таблицы с помощью **Конструктора**, откроется окно конструктора. Выбор режима осуществляется кнопкой **Режим** в группе команд **Режимы** (вкладка Главная). В столбце **Имя** поля в первой строке введите название первого поля **Код Отдела**. Кликните на первой строке в столбце **Тип данных**. Появится указатель поля со списком. Кликните на указателе и в раскрывшемся списке выберите тип поля **Счетчик**. Кликните на первой строке (**Код отдела**). На вкладке **Конструктор** выберите группу команд **Сервис**, кликните по кнопке **Ключевое поле**, около имени **Код Отдела** появится символ ключа (данную операция Access выполняет автоматически).

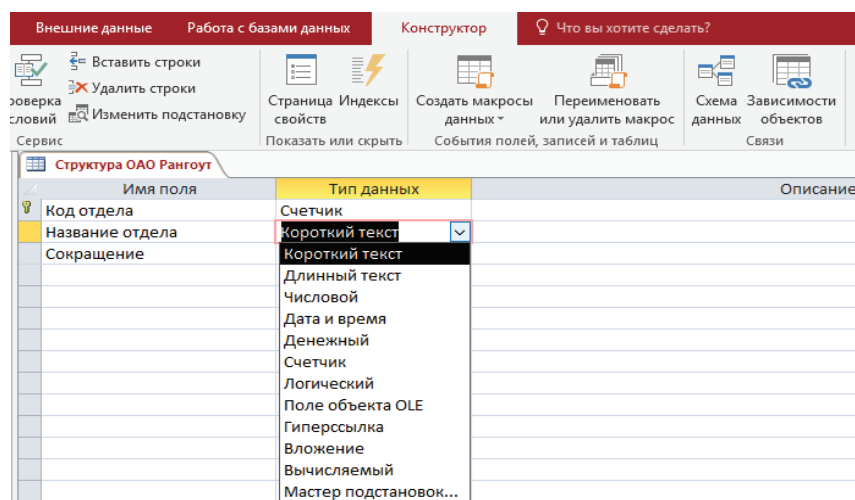


Рис. 3. Введение типов данных в таблицу в режиме конструктор

2.2. Введите названия и типы двух других полей (рис. 3):

Название отдела – тип данных короткий текст;

Сокращение – тип данных короткий текст.

2.3. Перейдите в режим **Таблица**. Программа предложит сохранить таблицу с некоторым именем. Введите имя таблицы: «**Структура ОАО Рангоут**». В режиме таблицы заполните столбцы данными (табл. 4).

Таблица 4 – Заполнение таблицы «Структура ОАО Рангоут»

Название Отдела	Сокращение
Отдел рекламы	ОР
Отдел снабжения	ОМТС
Дирекция	ДР
Бухгалтерия	БГ
Учебная часть	УЧ
Столовая	СТЛ
Технический отдел	ТХ

Практическое занятие № 2

Создание и изменение свойств полей

Цель занятия: научиться изменять свойства полей, создавать и заполнять новые поля. Закрепить навыки по созданию таблиц.

Отрабатываемые вопросы

1. Изменение свойств полей таблицы.
2. Создание нового поля с использованием подстановки значений из другой таблицы.
3. Создание нового поля с фиксированным набором значений.
4. Создание и заполнение поля с фотографией сотрудника.
5. Создание таблицы дополнительных сведений о сотруднике.

Организационно-методические указания

1. Изменение свойств полей таблицы.

1.1. При создании новой таблицы вы задавали только тип поля, а другие свойства полей определялись автоматически. Свойства полей вы можете просмотреть в режиме конструктора. Просмотрите свойства полей таблицы **Сотрудники** и измените некоторые свойства с целью ее улучшения.

1.2. Откройте таблицу **Сотрудники**. Для этого в **Области переходов** (рис. 4) выберите объект **Таблицы** и двойным щелчком мыши откройте таблицу **Сотрудники**.

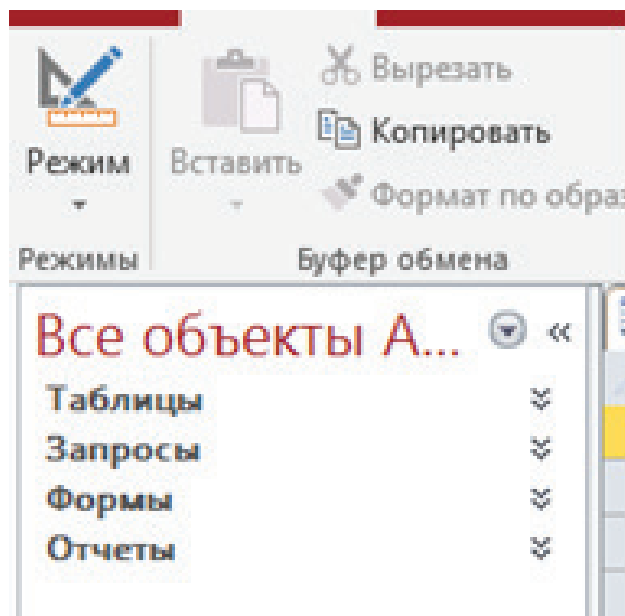


Рис. 4. Область переходов

1.3. Выберите режим **Конструктора**. В нижней части окна конструктора расположена информация о свойствах того поля таблицы, которое в данный момент выделено (рис. 2).

1.4. Выделите поле **Фамилия**. На вкладке **Общие** (свойства поля) рассмотрите строку со свойством **Размер поля**, в котором определяется максимальная допустимая длина текстовой строки этого поля. Вы увидите, что после создания таблицы здесь установлено значение 255 символов, но, конечно, не найдется фамилии такой длины. Измените значение на более реальное, например, 20.

1.5. Аналогичным образом измените размеры полей **Имя**, **Отчество**.

2. Создание нового поля с использованием подстановки значений из другой таблицы.

Создайте в таблице **Сотрудники** столбец **Название Отдела**, где для каждого сотрудника будет указываться название отдела, в котором он работает. Используйте **Столбец Подстановок**. **Столбец подстановок** используется в тех случаях, когда необходимо создать поле с повторяющимися значениями, чтобы все значения были введены одинаково. Тогда при заполнении таблицы значения полей можно будет заносить не с помощью ручного ввода, а путем выбора из списка. Для подстановки можно использовать столбец из ранее созданной таблицы.

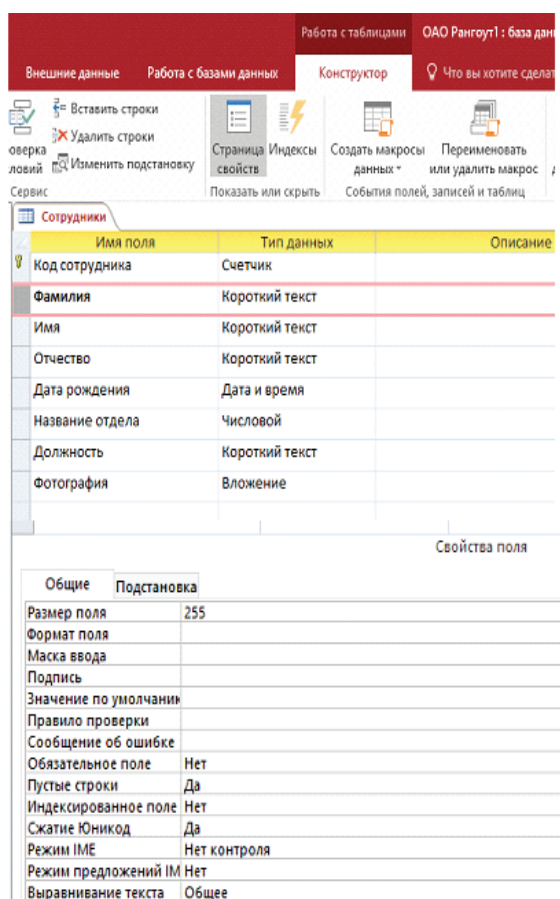


Рис. 5. Окно «Режим конструктора»

2.1. Откройте таблицу **Сотрудники**.

2.2. Выберите режим **Конструктора** (рис. 5).

2.3. В пустой строке **Кликните** в столбце **Тип данных**. (Имя поля вводить в этом случае необязательно). На вкладке **Конструктор** в группе команд **Сервис** выберите **Изменить подстановку**. Создайте новое поле, по следующей схеме.

Определение Источника данных. **Мастер** предлагает два варианта источника данных. Выберите первый: Объект «**столбец подстановки**» будет использовать значения из таблицы или запроса. Перейдите на следующий шаг, нажав кнопку **Далее** (рис. 6).

Выбор источника данных. В базе данных создана еще одна таблица «**Структура ОАО Рангоут**». Выберите ее в качестве источника данных.

2.4. Перейдите на следующий шаг, нажав кнопку **Далее**:

- выберите столбец подстановок;
- перенесите имя столбца **Название Отдела** из левого списка в правый;
- перейдите на следующий шаг, нажав кнопку **Далее**;
- четвертый и пятый шаги не требуют пояснений и исправлений;

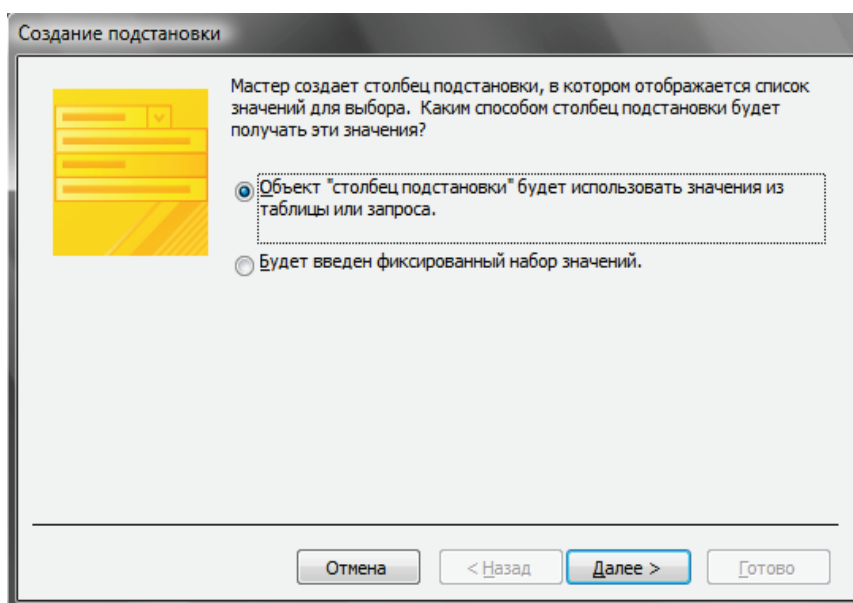


Рис. 6. Создание подстановок

– задайте подпись Название отдела и установите флажки, как показано на рис. 7.

2.5. Работа с мастером заканчивается щелчком на кнопке **Готово**. После завершения работы мастера просмотрите, как изменилась структура таблицы «Сотрудники». В ней появилось поле **Название отдела**.

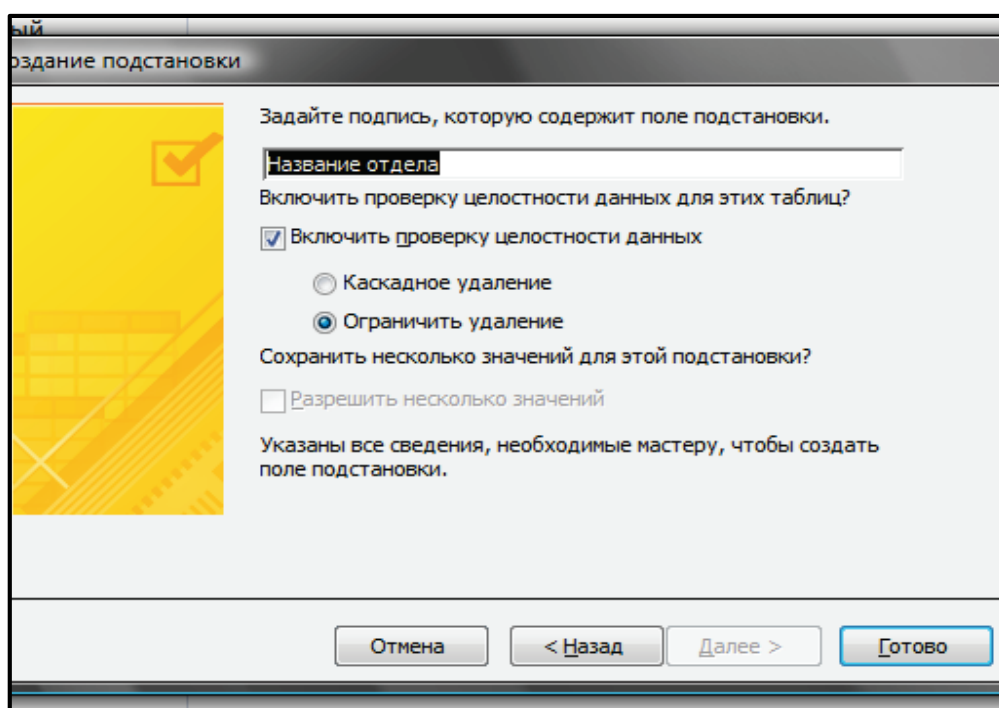


Рис. 7. Задание названия и включение флажков

2.6. Перейдите в режим таблицы и заполните новое поле, выбирая из списка для каждого сотрудника его место работы (табл. 5). После ввода данных закройте таблицу.

Таблица 5 – Заполнение поля «Название отдела»

Фамилия	Название Отдела
Соколов	Дирекция
Белых	Отдел снабжения
Мухин	Технический отдел
Ильина	Дирекция
Плужников	Дирекция
Андреева	Отдел снабжения
Бортникова	Отдел снабжения
Иванова	Бухгалтерия
Трофимов	Дирекция
Борзов	Отдел рекламы
Петрова	Учебная часть

3. Создание нового поля с фиксированным набором значений.

Как правило, на предприятии существует фиксированный список должностей.

Среда баз данных позволяет создать поле с фиксированным набором значений. В таблице «Сотрудники» создайте поле **Должность** с фиксированным набором значений, используя **Мастер подстановок**.

3.1. Откройте таблицу «Сотрудники» в режиме **Конструктора**.

3.2. В столбце **Тип данных** в пустой строке в списке выберите пункт **Мастер подстановок**.

3.3. На первом шаге работы установите переключатель «**Будет введен фиксированный набор значений**».

3.4. На втором шаге задайте число столбцов (1) и введите названия должностей: директор, замдиректора, секретарь, гл. бухгалтер, бухгалтер, менеджер, маркетолог, консультант, преподаватель, специалист (рис. 8).

3.5. На третьем шаге введите имя таблицы: **Должность**.

3.6. Завершите работу мастера щелчком на кнопке **Готово**.

Просмотрите, как изменилась структура таблицы. Кликните на вкладке **Свойство поля–Подстановка**. Вы увидите, что в строке **Источник строк** появился список должностей, записанных в кавычках и разделенных точкой с запятой.

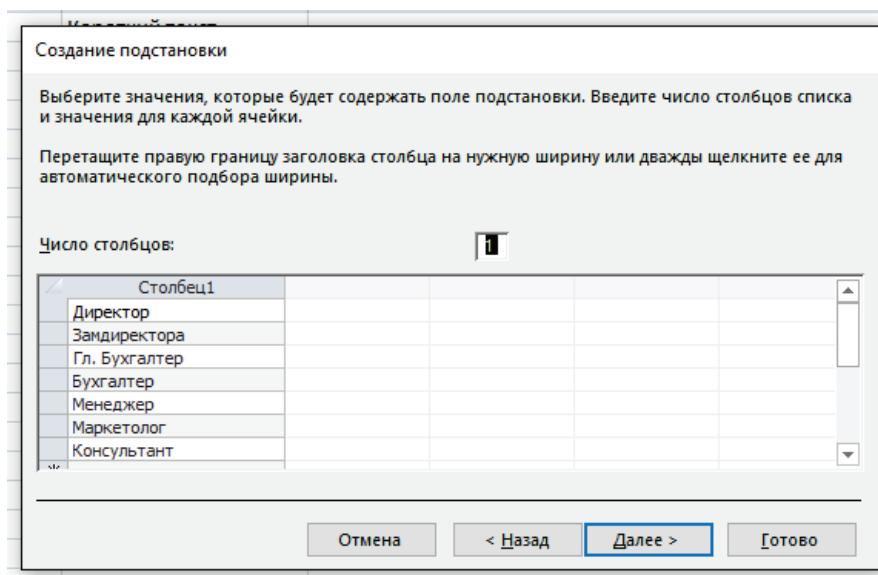


Рис. 8. Окно «Создание подстановки фиксированный набор значений»

3.7. Перейдите в режим **Таблицы** и заполните созданное поле (табл. 6).

Таблица 6 – Заполнение поля «Должность»

Фамилия	Должность
Соколов	Директор
Белых	Специалист
Мухин	Консультант
Ильина	Секретарь
Плужников	Зам. директора
Андреева	Менеджер
Бортникова	Менеджер
Иванова	Гл. бухгалтер
Трофимов	Менеджер
Борзов	Специалист
Петрова	Консультант

4. Создание и заполнение поля с фотографией сотрудника.

Традиционно на каждом предприятии в отделе кадров заводится листок по учету кадров. На этом листке обязательно помещается фотография сотрудника. В компьютерной базе данных по учету сведений о сотрудниках также можно вставить фотографию сотрудника. В этом случае фотография должна быть представлена не в бумажном виде, а в виде компьютерного файла графического формата **gif**, **jpg** или других графических форматов. В базе данных «**ОАО Рангоут**» создайте новое поле, в котором будет храниться фотография сотрудника.

4.1. В папке **ОАО Рангоут** создайте вложенную папку **Фотографии**.

4.2. Сохраните в папке **Фотографии** несколько графических файлов с фотографиями сотрудников. Графические файлы можно создать или сканированием бумажных фотографий, или сфотографировав сотрудника цифровым фотоаппаратом. Размер графического файла старайтесь сделать небольшим: 3х4 см, хотя, как вы увидите, это необязательно.

4.3. Откройте таблицу **Сотрудники** в режиме **Конструктора**. После чего выполните следующие действия:

- вставьте пустую строку после строки **Отчество**. Выделите строку **Дата Рождения**. На вкладке **Конструктор** в группе команд **Сервис** выберите пункт **Вставить Строки**;
- в пустую строку введите название поля **Фотографии**;
- в столбце **Тип поля** в пустой строке в списке выберите пункт **Вложение**. В свойствах поля на вкладке **Общие** в строке **Подпись** наберите **Фотографии** (рис. 9).

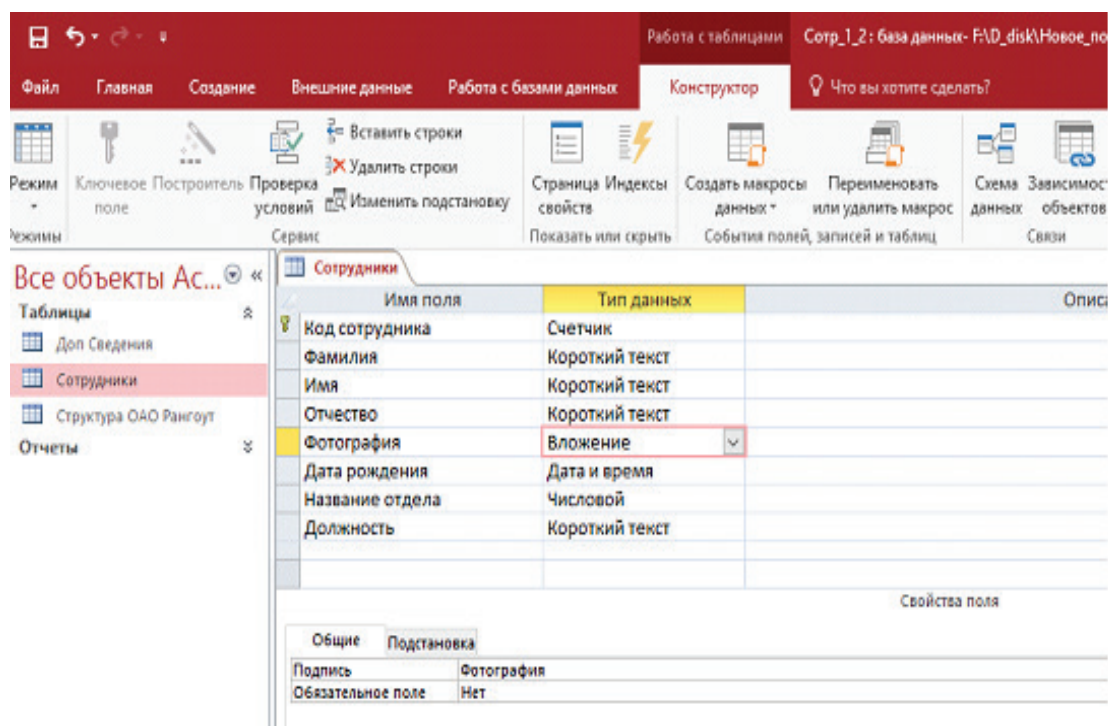
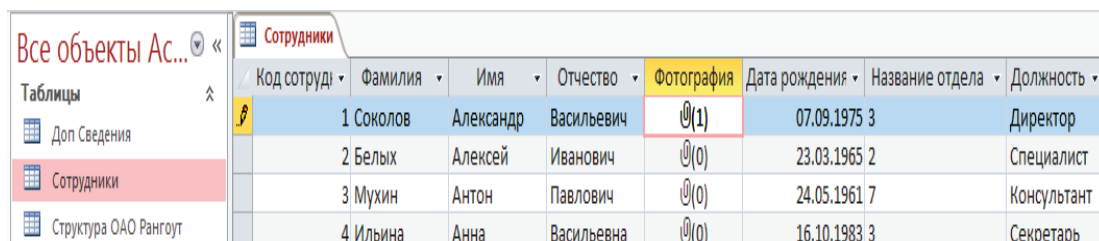


Рис. 9. Задание имени поля «Фотографии»

- перейдите в режим таблицы, в поле фотографии появится значок;
- для вставки фотографии сотрудника кликните правой кнопкой мыши в поле **Фотографии** в строке одного из сотрудников;
- в контекстном меню выберите пункт **Управление вложениями**, появится диалоговое окно **Вложения**;
- кликните по кнопке **Добавить**;
- выберите папку **Фотографии**, откройте в ней необходимый файл и в диалоговом окне **Вложения** нажмите **ОК**.

4.4. После вставки файла в поле **Фотографии** появится значок вложения с цифрой 1 в скобках (в данное поле вложена одна фотография (рис. 10)). Двойным щелчком на значке вы можете активизировать программу просмотра фотографии выбрав в контекстном меню команду **Открыть** (в дальнейшем, при создании форм, вы получите возможность просматривать фотографии прямо из базы данных).



Код сотруд	Фамилия	Имя	Отчество	Фотография	Дата рождения	Название отдела	Должность
1	Соколов	Александр	Васильевич	📎(1)	07.09.1975	3	Директор
2	Белых	Алексей	Иванович	📎(0)	23.03.1965	2	Специалист
3	Мухин	Антон	Павлович	📎(0)	24.05.1961	7	Консультант
4	Ильина	Анна	Васильевна	📎(0)	16.10.1983	3	Секретарь

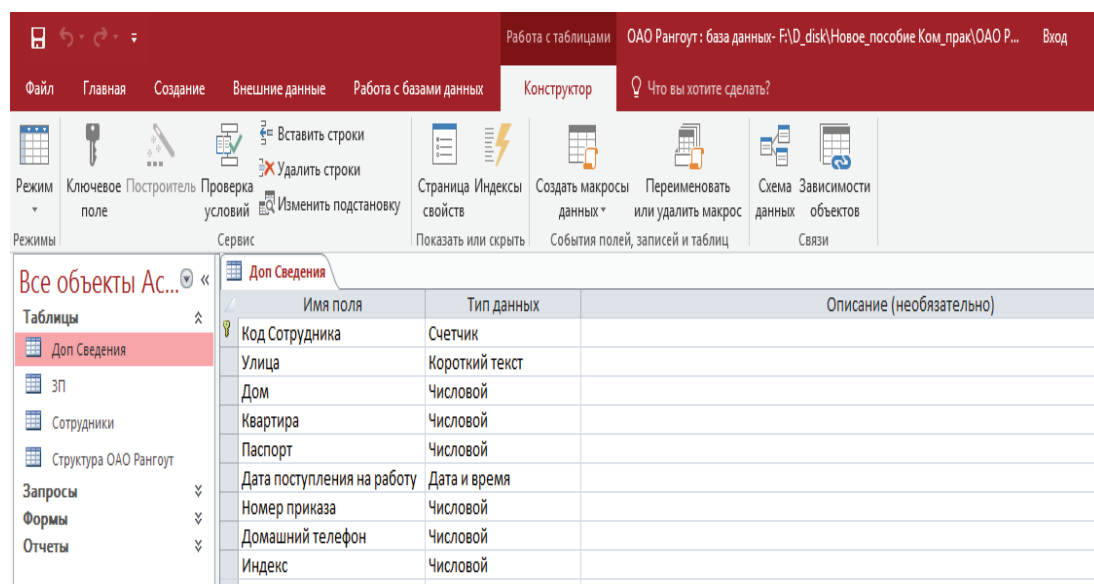
Рис. 10. Поле типа Вложение

5. Создание таблицы дополнительных сведений о сотруднике.

Создайте таблицу **Доп. Сведения**, в которой будут содержаться сведения о сотрудниках, необходимые для отдела кадров. Особенность такой таблицы состоит в том, что в ней будет столько же записей, сколько и в таблице **Сотрудники**. При этом каждая запись таблицы **Сотрудники** будет соответствовать только одной записи таблицы **Доп. Сведения**.

5.1. Запустите режим создания таблицы с помощью **Конструктора**.

5.2. Создайте структуру таблицы (рис. 11).



Имя поля	Тип данных	Описание (необязательно)
Код Сотрудника	Счетчик	
Улица	Короткий текст	
Дом	Числовой	
Квартира	Числовой	
Паспорт	Числовой	
Дата поступления на работу	Дата и время	
Номер приказа	Числовой	
Домашний телефон	Числовой	
Индекс	Числовой	

Рис. 11. Фрагмент структуры таблицы «Доп. Сведения»

5.3. Задайте ключевое поле (код сотрудника).

5.4. Закройте окно конструктора. При закрытии сохраните новую таблицу с именем **Доп. Сведения** (данные в таблицу пока не вносите).

5.5. В таблице **Доп. Сведения** создайте поля для дополнительных данных о сотрудниках: **паспорт, дата поступления на работу, номер приказа, домашний телефон, индекс и другие**. Новые поля располагайте в таблице в логической последовательности, а не в том порядке, как они перечислены выше. Например, индекс должен располагаться перед улицей, домашний телефон после адреса (рис. 12). Типы данных выберите самостоятельно.

Код Сотруд	Улица	Дом	Квартира	Паспорт	Дата постуг	Номер приказ	Домашний	Индекс	Щелкните для добавления
1	Ленина	55	1	103	201030404	03.12.2010	1	5343765	198589
2	Светланская	43	2	203040304	04.11.2012	2	5323654	198567	
3	Ивановская	32	3	453543534	15.05.2008	3	3423423	1298705	
4	Уборевича	9	4	423477234	19.03.2013	4	2523423	198505	
5	Крупской	70	5	654890899	11.03.2014	5	5894546	198450	
(№)		0		0	0		0	0	

Рис. 12. Поля таблицы «Доп. Сведения»

5.6. Для логически правильного расположения полей вам понадобится вставлять пустые строки между имеющимися строками. Для этого в режиме **Конструктора** выделите строку, перед которой будет располагаться вставленная строка, и на вкладке **Конструктор** в группе команд **Сервис** выберите пункт **Вставить строки**.

Практическое занятие № 3

Создание базы данных, состоящей из трех таблиц

Цель занятия: научиться создавать связи между таблицами. Закрепить навыки по работе с полями таблиц.

Отрабатываемые вопросы

1. Создание связи «один – ко – многим».
2. Создание связи «один – к – одному».

Организационно-методические указания

1. Создание связи «один – ко – многим».

В созданной базе данных **Сотрудники ОАО РАНГОУТ** у вас есть две таблицы: **Сотрудники** и **Структура ОАО Рангоут**. Установите связь «**Один – ко – многим**» между двумя таблицами. Эта связь означает, что в одном отделе может числиться много сотрудников, но ни один сотрудник не может числиться сразу в нескольких отделах. Установите **Обеспечение целостности данных**, которое означает, что все изменения в таблице **Структура ОАО Рангоут** отражаются и в таблице **Сотрудники**.

Примечание. Перед началом работы закрыть все активные таблицы. Для этого подведите курсор на вкладке команд к названию таблицы и, нажав правую клавишу мыши, выберите команду закрыть.

1.1. Кликните на кнопке **Схема данных** (вкладка **Работа с базами данных**, в группе команд **Отношения**). Открывшееся окно имеет рабочую область, в которую можно добавить необходимые таблицы и установить между ними связи. При использовании **Столбца подстановок** в рабочей области уже существует связь между таблицами, удалите ее. Для этого подведите курсор мыши к линии связи, соединяющей таблицы, нажмите правую клавишу, выбрав команду **Удалить**.

1.2. Кликните правой кнопкой мыши на свободном пространстве окна и с помощью контекстного меню добавьте две созданные таблицы. Таблицы появляются в окне в виде небольших окон. Заголовок окна соответствует названию таблицы, содержимое окна – названию полей (при открытии окна **Схема данных** в нем уже может находиться одна или обе созданные таблицы, в этом случае необходимо добавить только отсутствующие).

1.3. В таблице **Сотрудники** выделите поле **Название Отдела**.

1.4. Удерживая кнопку мыши, двигайте курсор к полю с названием **Код отдела** в другой таблице. Когда курсор мыши окажется внутри другого окна, он изменит свой вид. После этого отпустите кнопку мыши, и откроется диалоговое окно **Изменение связей** (рис. 13).

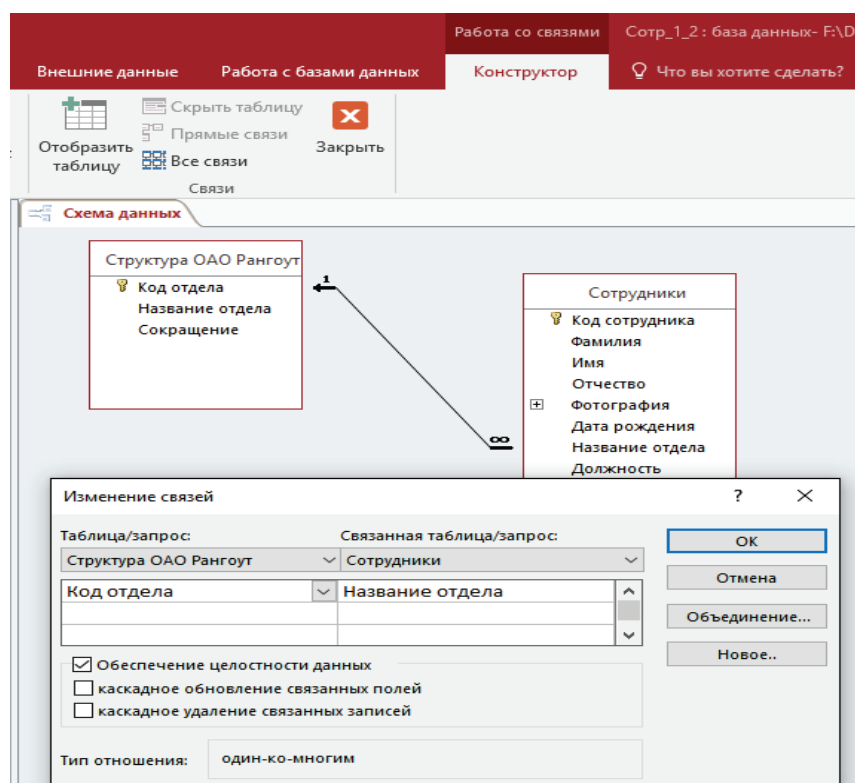


Рис. 13. Схема данных связь «один – ко – многим»

1.5. Кликните на кнопке **Объединение**. В дополнительном окне установите переключатель **3 – Объединение ВСЕХ записей из «Сотрудники» и только тех записей из «Структура ОАО РАНГОУТ», в которых связанные поля совпадают** (рис. 14).

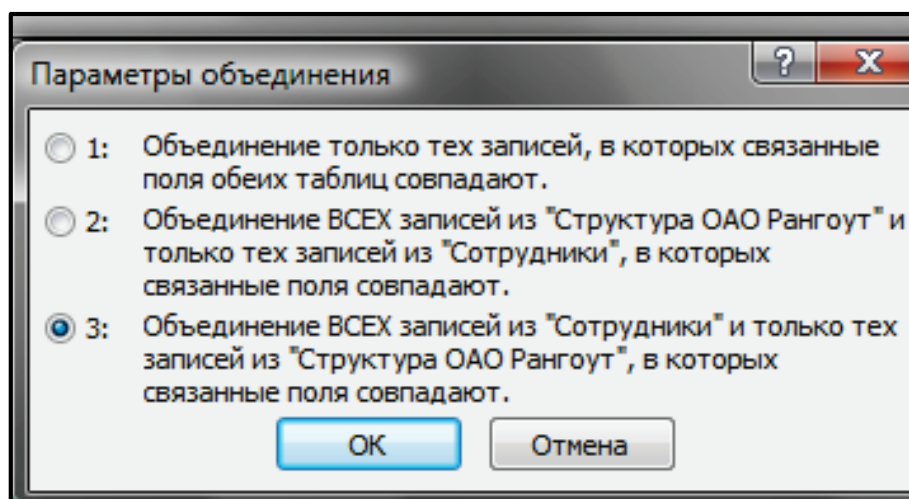


Рис. 14. Параметры объединения

1.6. Установите флажок **Обеспечение целостности данных** (рис. 13). Подтвердите свой выбор, нажав **ОК**. Просмотрите в окне **Схема данных** созданную связь. Связь имеет вид линии со стрелкой и надписями «1→». Эта связь называется «**Один – ко – многим**». Установленное обеспечение целостности данных означает, что, если вы измените название отдела, эти изменения отразятся и в таблице **Сотрудники**.

1.7. Откройте таблицу **Структура ОАО Рангоут** в режиме **Таблица**. Слева в таблице теперь расположен столбец со значками «+». Эти значки указывают на наличие связи «**Один – ко – многим**» и позволяют просмотреть запись из связанной таблицы.

1.8. Кликните по значку в какой-нибудь строке. Откроются строки из связанной таблицы с фамилиями сотрудников этого отдела.

2. Создание связи «Один – к – одному». Установите связь «**один – к – одному**» между таблицами **Доп. Сведения** и **Сотрудники**. Установите **Обеспечение целостности данных**. После установления связи заполните таблицу **Доп. Сведения** данными.

2.1. Откройте окно **Схема данных**.

2.2. Добавьте к схеме данных таблицу **Доп. Сведения**.

2.3. В таблице **Сотрудники** выделите поле **Код Сотрудника**.

2.4. Удерживая кнопку мыши, двигайте курсор к полю с таким же названием в таблице **Доп. Сведения**. Когда курсор мыши окажется внутри другого окна, он изменит свой вид. После этого отпустите кнопку мыши, и откроется диалоговое окно **Изменение связей**.

2.5. Установите флажок «**Обеспечение целостности данных**».

2.6. Кликните на кнопке «**Объединение**». В дополнительном окне установите переключатель **2**, подтвердите свой выбор, нажав **ОК**.

2.7. Просмотрите в окне **Схема данных** созданную связь. Связь имеет вид линии со стрелками и надписями «11». Эта связь называется «**Один – к – одному**» и означает буквально, что каждой записи в таблице **Сотрудники** будет соответствовать только одна, связанная с ней запись в таблице **Доп. Сведения**.

2.8. Закройте окно **Схема данных**.

2.9. Откройте таблицу **Сотрудники** в режиме **Таблица**. Слева в таблице теперь расположен столбец со знаками «+». Эти значки указывают на наличие связи «**Один – к – одному**» и позволяют просмотреть запись из связанной таблицы.

2.10. Кликните на значке «+». Откроется строка из связанной таблицы для ввода данных.

2.11. Введите дополнительные сведения для всех сотрудников (данные выбрать самостоятельно).

Задание для самостоятельной работы. Создание типа данных вычисляемый

1. Создайте таблицу «Зарботная плата» со следующими полями и типами данных (рис. 15).

Имя поля	Тип данных
Код сотрудника	Счетчик
Фамилия	Короткий текст
Оклад	Денежный
Премия	Вычисляемый
Налог	Вычисляемый
Итого	Вычисляемый

Рис. 15. Поля таблицы «Зарботная плата»

2. Установите связь «один – к – одному» между таблицами **Зарботная плата** и **Сотрудники**.

3. Введите в поле **Оклад** и сведения по каждому сотруднику (значения выбирайте самостоятельно).

4. В поле Премия в окне **Свойства поля** введите формулу вычисления премии «**Оклад*10 %**», как показано на рис. 16.

Общие	Подстановка
Выражение	[Оклад]*0,1
Тип результата	Действительное
Формат поля	Денежный
Точность	18
Шкала	0
Число десятичных знаков	2
Подпись	
Выравнивание текста	Общее

Рис. 16. Заполнение поля «Премия»

5. В полях «Премия», «Налог» и «Итого» введите соответственно формулы:

- [Оклад]*0,1;
- [Оклад]*0,13;
- [Оклад] + [Премия]-[Налог].

Практическое занятие № 4

Создание и редактирование формы для ввода данных

Цель занятия: научиться создавать и редактировать формы. Закрепить навыки по созданию связей между таблицами.

Отрабатываемые вопросы

1. Создание форм для ввода-вывода с помощью мастера форм.
2. Редактирование форм для ввода-вывода в режиме конструктора.

Организационно-методические указания

1. Создание формы для ввода данных. Выполняя предыдущие задания, вы научились вводить исходные данные путем заполнения построчно созданной таблицы. Однако среда баз данных позволяет вводить данные, предварительно создав Форму. Форма – это аналог карточки, в которой введены данные по одному конкретному объекту. Ввод данных непосредственно в таблицу не очень удобен, так как длина некоторых полей довольно большая и все столбцы одновременно не видны на экране. Другой недостаток заключается в том, что в таблице видны данные сразу по всем записям, а это не всегда желательно, особенно в тех случаях, когда необходимо соблюдать конфиденциальность. Форма – очень удобный способ заполнения новых записей, похожий на заполнение карточки. Создайте форму для ввода данных о сотрудниках. Включите в форму поля из двух связанных таблиц **Сотрудники** и **Доп. Сведения**.

1.1. На вкладке **Создание** в группе команд **Формы** выберите элемент управления – **Мастер форм**.

1.2. Запустите режим создания формы с помощью **Мастера** (рис. 17).

Выбор источника данных. В базе данных создана еще одна таблица **«Структура ОАО Рангоут»**. Выберите ее в качестве источника данных.

- Перейдите на следующий шаг, нажав кнопку **Далее**;
- выберите столбец подстановок;
- перенесите имя столбца **Название Отдела** из левого списка в правый;
- перейдите на следующий шаг, нажав кнопку **Далее**;
- четвертый и пятый шаги не требуют пояснений и исправлений.

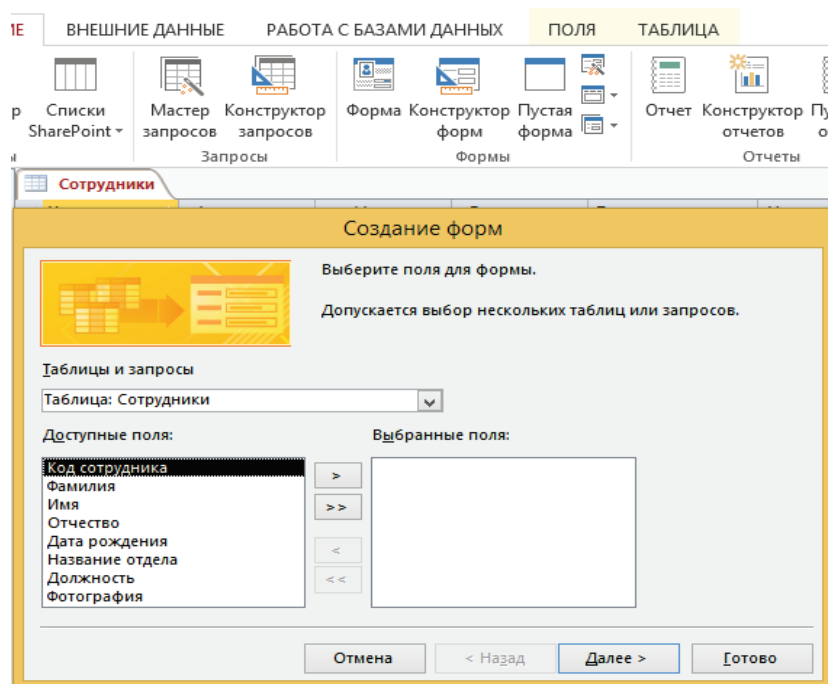


Рис. 17. Мастер форм

1.3. Создайте форму, следуя шагам мастера, для этого:

- выберите из таблицы поля для создания формы;
- в списке **Таблицы и запросы** выберите таблицу **Сотрудники**.

Перенесите из списка **Доступные поля** в список **Выбранные поля** все поля таблицы, щелкнув на кнопке **Добавить все «>>»** (рис. 18);

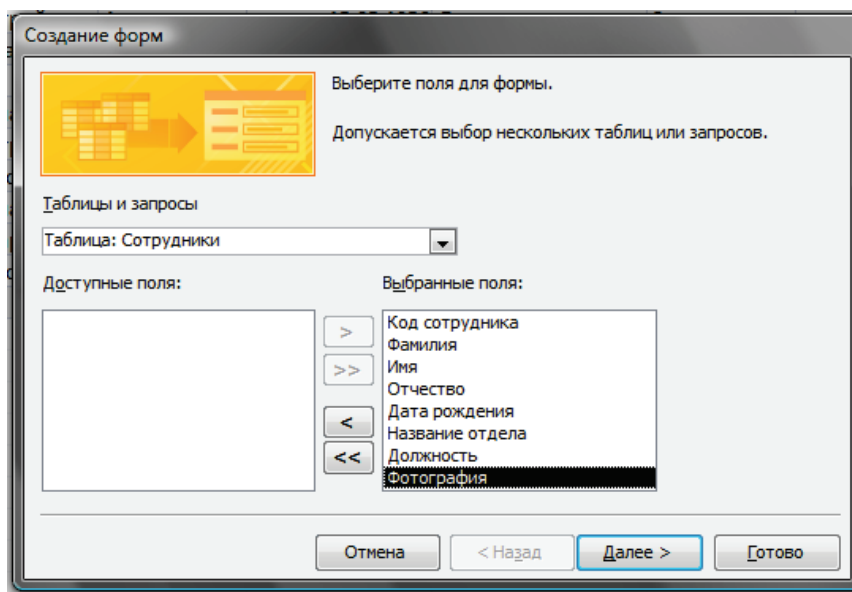


Рис. 18. Выбор полей формы

- выберите таблицу **Доп. Сведения**;
- перенесите из нее все поля (кроме первого поля **Код Сотрудника**, так как оно уже есть в списке);
- перейдите на следующий шаг, нажав кнопку **Далее**;
- выбор вида формы. Установите переключатель «В один столбец» – это наиболее удобный вид формы;
- перейдите на следующий шаг, нажав кнопку **Далее**;
- перейдите на следующий шаг, нажав кнопку **Далее**;
- введите имя формы **Сотрудники**;
- установите переключатель **Открыть форму** для просмотра и ввода данных;

– завершите работу с мастером щелчком на кнопке **Готово**.

1.4. Просмотрите имеющиеся записи, щелкая по кнопкам:

- переход к первой записи;
- переход на предыдущую запись;
- переход на следующую запись;
- переход к последней записи;
- переход на новую запись.

1.5. Введите еще несколько новых записей, пользуясь формой.

2. Редактирование формы. Процесс редактирования предполагает изменение вида некоторого объекта с целью его улучшения. При редактировании формы можно изменить названия полей для ввода, увеличить или уменьшить размер области ввода данных, изменить порядок расположения полей в карточке и многое другое. Для редактирования формы надо перейти в режим конструктора (рис. 19).

Рис. 19. Неотредактированная форма «Сотрудники»

Можно использовать конструктор и для создания форм, но это очень трудоемкая работа. Поэтому лучше поручить ее мастеру, а конструктор только для редактирования формы. На рис. 20 представлена форма, полученная после работы мастера. Отредактируйте полученную форму согласно рис. 21.

2.1. Откройте созданную форму **Сотрудники**.

2.2. Перейдите в режим **Конструктора**.

Примечание. В режиме конструктора рабочее поле представляет собой сетку с выделенными областями для расположения объектов: **Заголовок формы**, **Область данных**, **Примечание формы**. Если навести указатель мыши на границу области, ее можно увеличить или уменьшить. Основные объекты расположены в **Области данных** – это надписи полей и поля. Обратите внимание, что надпись и название поля могут не совпадать. Информация, расположенная в рамке надписи, не меняется при просмотре записей. Это аналог неизменяемой части карточки. Поле предназначено для ввода данных. В окне конструктора оно представляет собой белый прямоугольник с одной или несколькими строками. Первоначально в режиме Конструктора и для надписи, и для поля указано имя соответствующего поля. Вы можете отредактировать надпись. Название поля менять не рекомендуется, так как это может привести к ошибке.

2.3. Измените размеры рамки для фотографии, для этого:

- щелчком выделите рамку, по границе рамки появятся маркеры;
- подведите курсор к угловому маркеру так, чтобы он принял форму двусторонней стрелки;
- потяните границу за маркер и увеличьте ее; размер рамки контролируйте по сетке, в которой одна клетка соответствует 1 см.

Код сотрудника	Код сотрудника	№ приказа	№ приказа
Фамилия	Фамилия	Дата поступления на работу	Дата поступления на работу
Имя	Имя	Индекс	Индекс
Отчество	Отчество	Улица	Улица
Фотографии	Фотографии	Дом	Дом
		Квартира	Квартира
Дата рождения	Дата рождения	Домашний телефон	Домашний телефон
Название отдела	Название отдела	Мобильный телефон	Мобильный телефон
Должность	Должность		

Рисунок по умолчанию (отсутствует)
Мозаичное заполнение
Выравнивание рисунка
Масштабы рисунка
Ширина
Высота
От верхнего края
От левого края
Тип фона
Цвет фона
Тип границы
Ширина границы
Цвет границы
Оформление
Выравнивание текста
Стиль линий сетки
Стиль линий сетки
Стиль линий сетки
Стиль линий сетки
Цвет линий сетки
Ширина линий сетки
Ширина линий сетки
Ширина линий сетки
Ширина линий сетки

Рис. 20. Форма «Сотрудники»

2.4. Измените, свойства рамки так, чтобы фотография полностью помещалась внутри рамки независимо от ее исходного размера. Для этого:

- правой кнопкой мыши кликните внутри рамки;
- в контекстном меню выберите пункт **Свойства**;
- в открывшемся окне свойств рамки выберите вкладку **Макет**;
- кликните на строке **Масштабы рисунка** и пункт списка вписать в рамку или по размеру рамки (рис. 20).

2.5. Перейдите в режим формы и просмотрите все записи; убедитесь, что теперь все фотографии полностью размещаются внутри рамки.

2.6. Аналогично измените размеры рамок надписей.

2.7. Переместите рамки некоторых надписей в правую часть формы (рис. 5).

2.8. Измените шрифт названия надписей. Для этого:

- выделите щелчком надпись поля **Код Сотрудника**, при выделении по границе надписи появляются маркеры выделения;
- кликните внутри надписи, появится мигающий текстовый курсор, измените шрифт и его размеры;
- аналогичным образом измените вид других надписей.

2.9. Измените, если необходимо, размеры формы. Для этого:

- подведите курсор к нижней границе так, чтобы он принял форму двусторонней стрелки;
- захватите границу формы (не путать с границей окна) и потяните вниз;
- аналогичным образом измените горизонтальный размер формы.

Сотрудники			
Код сотрудника	1	Номер приказа	1
Фамилия	Соколов	Домашний телефон	5343765
Имя	Александр	Индекс	198589
Отчество	Васильевич	Улица	Ленина
Дата рождения	07.09.1975	Дом	1
Название отдела	Дирекция	Квартира	103
Должность	Директор	Паспорт	201030404
Фотография	03.12.2010		

Рис. 21. Результат редактирования формы в режиме конструктор

Практическое занятие № 5

Обработка данных с помощью запросов

Цель занятия: научиться создавать запросы.

Отрабатываемые вопросы

1. Создание запроса на выборку из двух таблиц с помощью мастера.
2. Создание вычисляемых полей в запросах.

Организационно-методические указания

1. Создание запроса на выборку из двух таблиц с помощью мастера.

Запрос – это операция, которая объединяет в себе основные режимы обработки данных: сортировку, фильтрацию, объединение данных из разных источников, преобразование данных. Под преобразованием данных понимается возможность создания вычисляемых полей, в которых по формулам на основании имеющейся информации получается новая. Создайте запрос **Исполнитель**, в котором представлены фамилии сотрудников и сокращенное название отдела, в котором они работают.

1.1. На вкладке **Создание** в группе **Другие** кликните по кнопке **Мастер запросов**.

1.2. Запустите режим создания запроса с помощью мастера (**Простой запрос**).

1.3. Создайте запрос, следуя шагам мастера:

Выбор из списка **Таблицы** и запросы таблицу **Структура ОАО РАНГОУТ**;

- перенесите в список выбранных полей (справа) поле Сокращение;
- выберите из списка Таблицы и Запросы таблицу **Сотрудники**;
- перенесите в список выбранных полей (справа) поле **Фамилия** (рис. 22);
- перейдите на следующий шаг, нажав кнопку **Далее**.

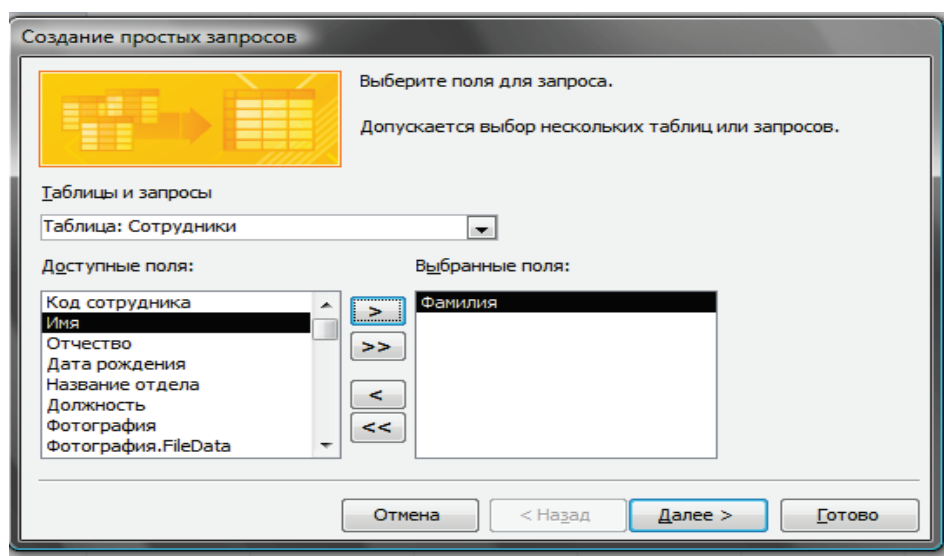


Рис. 22. Создание простых запросов

1.4. Введите имя запроса. Для этого:

- выберите тип запроса **Подробный**;
- введите имя запроса **Исполнитель**;
- завершите работу мастера, нажав кнопку Готово.

1.5. Просмотрите результаты запроса в режиме **Таблицы**.

1.6. Выберите режим **Конструктора**. Откроется бланк запроса, в верхней части бланка расположена схема связи таблиц, используемых в запросе. В нижней части расположена таблица описания полей запроса. В первой строке перечислены поля запроса. Во второй строке указано имя таблицы, из которой взято поле. В третьей строке можно задать сортировку полей (рис. 23).

Поле:	Имя таблицы:	Сортировка:	Вывод на экран:	Условие отбора:
Сокращение	Структура ОАО Ранг		☒	
Фамилия	Сотрудники		☒	
			☐	

Рис. 23. Бланк запроса

1.7. Установите в обоих полях сортировку по возрастанию. При наличии сортировки в нескольких полях, программа выполняет сначала первую, а потом вторую и т. д. Таким образом, запрос предоставляет возможность задать сортировку по нескольким полям.

1.8. Перейдите в режим просмотра **Таблицы**. Просмотрите, как изменилось расположение данных и не забудьте сохранить изменения.

2. Создание вычисляемых полей в запросах. Создайте запрос **Сотрудники – Запрос с вычисляемыми полями**, в которых по данным таблицы будут получены новые данные со следующими назначениями (табл. 7).

Таблица 7 – Назначение полей запроса

Имя поля запроса	Назначение
Код Сотрудника	Устанавливает связь получаемых в других полях данных с конкретным сотрудником по ключевому полю.
ФИО	Содержит фамилию, имя и отчество как одну строку.
Возраст	Вычисляет количество полных лет по дате рождения.
Месяц	Определяет номер месяца рождения по дате.
День	Определяет порядковый день месяца рождения по дате.

Вид бланка запроса представлен на рис. 24. Созданные в запросе поля **Месяц** и **День** позволяет по-другому провести сортировку сотрудников по месяцам и дням даты рождения и составить список, в котором сотрудники будут указаны в порядке дат рождения от начала года.

Рис. 24. Бланк запроса

2.1. На вкладке **Создание** кликните на кнопке **Конструктор запросов**.

2.2. Запустится режим создания запроса с помощью **Конструктора**. Откроется окно **Добавление таблицы**.

2.3. В окне **Добавление таблицы** выделите таблицу **Сотрудники** и кликните на кнопке **Добавить**. Откроется бланк запроса. В верхней части бланка представлен список полей таблицы **Сотрудники**.

2.4. В первом столбце бланка запроса введите имя поля **Код Сотрудника**, выбрав его из списка, который раскроется при щелчке на первой строке. Имя таблицы появится во второй строке автоматически.

2.5. В следующем столбце создайте поле **ФИО**, в котором фамилия, имя, отчество сотрудника будут представлены как единая текстовая строка. Для этого:

- наберите в столбце название **ФИО**;
- там же кликните правой кнопкой мыши, откроется окно **Построитель выражений** (рис. 25);
- в левом окне обзора раскройте папку **Таблицы** и в ней вложенную папку **Сотрудники**: поля таблицы Сотрудники будут представлены в среднем окне построителя запросов;
- введите формулу для вычисляемого поля **ФИО** согласно схеме (рис. 25);

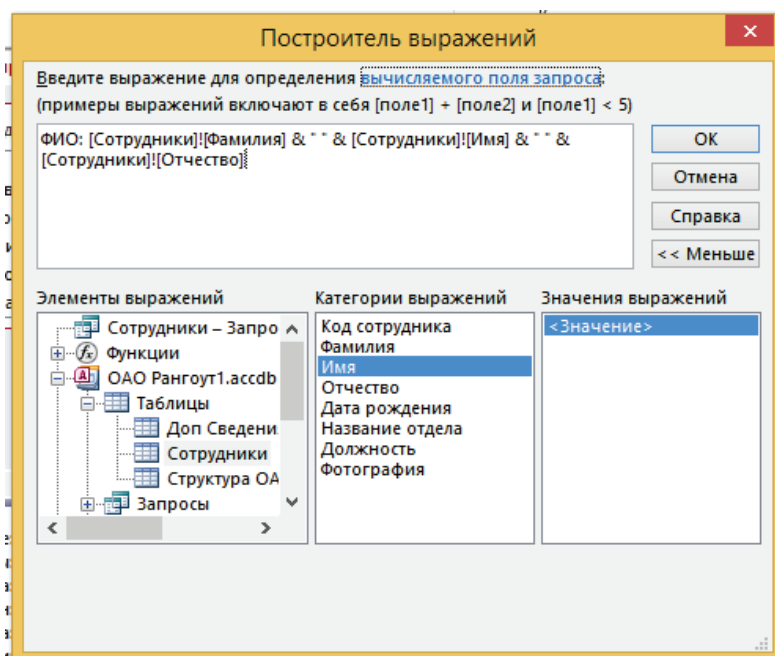


Рис. 25. Построитель выражений

- в столбце № 2 бланка запроса наберите **ФИО** ;
- дважды щелкните мышкой по полю Фамилия. Появится надпись: [Сотрудники].[Фамилия];
- наберите на клавиатуре &" "& (между кавычками сделать пробел);
- аналогично наберите всю схему;
- если при наборе появляется надпись «Выражение», удалите ее;
- завершите ввод формулы, нажав ОК;
- убедитесь, что формула появилась в бланке запроса. Так как ширина столбца не очень большая, то вся формула не будет видна. Либо увеличьте ширину столбца, либо просмотрите формулу, перемещая по ней курсор.

2.6. В третьем столбце постройте выражение для поля **Возраст**, в котором производится вычисление количества полных лет по дате рождения:
Возраст: Year(Now())-Year([Сотрудники]![Дата рождения]).

Примечание. Эта формула содержит встроенные функции Year(), которая вычисляет год по дате, и Now(), которая вычисляет текущую дату. Встроенные функции можно найти, открыв в построителе выражений в окне обзора папки Функции папку Встроенные функции. Возраст получается, как разность между годом, отсчитанным от текущей даты, и годом, отсчитанным от даты рождения.

2.7. В следующем столбце постройте выражение для поля **Месяц**, в котором производится вычисление по дате рождения порядкового номера месяца. В формуле используется встроенная функция **Month()**:

Месяц: Month([Сотрудники]![Дата Рождения]).

2.8. В следующем столбце постройте выражение для поля **День**, в котором производится вычисление по дате рождения порядкового дня месяца. В формуле используется встроенная функция **Day()**:

День: Day([Сотрудники]![Дата Рождения]).

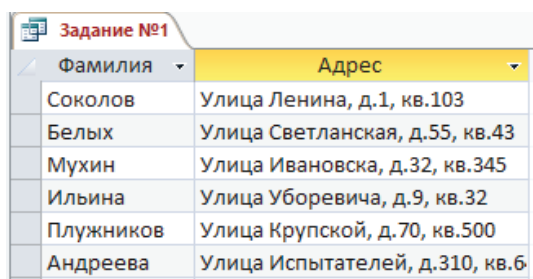
2.9. Задайте в бланке запроса сортировку по полю **Месяц**, затем по полю **День**.

2.10. Включите в бланк запроса поле **Дата рождения** из таблицы **Сотрудники**.

2.11. Перейдите в режим таблицы и просмотрите результаты работы запроса.

Задания для самостоятельной работы

Задание 1. Создайте в бланке запроса поле **Адрес**, в котором по названию улицы, номеру дома и квартиры формируется адрес в виде одной строки. На рис. 26 приведен результат выполненного запроса.



Фамилия	Адрес
Соколов	Улица Ленина, д.1, кв.103
Белых	Улица Светланская, д.55, кв.43
Мухин	Улица Ивановская, д.32, кв.345
Ильина	Улица Уборевича, д.9, кв.32
Плужников	Улица Крупской, д.70, кв.500
Андреева	Улица Испытателей, д.310, кв.6

Рис. 26. Запрос «Адрес»

Задание 2. Введите в бланк запроса условие, по которому отбираются все сотрудники в возрасте от 25 до 40 лет. Используйте для выполнения задания запрос Исполнитель. Наберите в поле Возраст необходимое условие с помощью

Построителя выражения, используя **Операторы сравнения** (>, <, <=, >=, =, <>) и **Логические** (**And**, **Not**, **Or**). Результат выполнения запроса приведен на рис. 27.

Код сотруд	Фамилия	Возраст
4	Ильина	33
12	Абрас	36
13	Борисов	29
(№)		

Рис. 27. Запрос «Задание № 2»

Практическое занятие № 6

Создание и редактирование отчетов

Цель занятия: научиться создавать и редактировать отчеты.

Отрабатываемые вопросы

1. Создание отчета с помощью мастера отчетов. Редактирование отчета в режиме конструктор.
2. Создание отчета «Дни Рождения».
3. Создание отчета «Представительский».

Организационно-методические указания

1. Создание отчета с помощью мастера отчетов. Редактирование отчета в режиме конструктор. Отчет – это средство для отображения данных при выводе на печать. Отчет формируется на основе созданных в базе данных таблиц и запросов. Создайте отчет **Сотрудники**, в котором формируется список сотрудников по отделам. Среда баз данных предоставляет возможность создать отчет при помощи **Конструктора** или **Мастера**. Наиболее удобно оформлять отчет, используя **Мастер**.

1.1. Во вкладке **Создание** в группе **Отчеты** выберите, **Мастер отчетов** и запустите его.

1.2. Выберите из таблицы **Структура ОАО РАНГОУТ** поле **Название отдела**, из таблицы **Сотрудники** – поля **Фамилия**, **Имя**, **Отчество**, **Должность** (рис. 28).

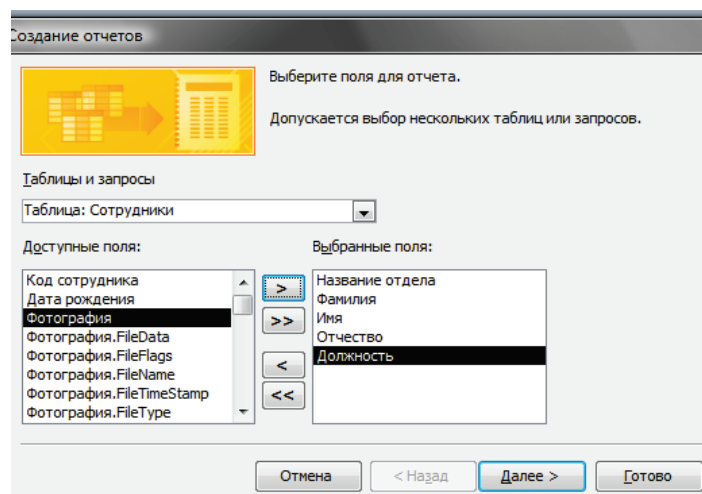


Рис. 28. Создание отчета с помощью мастера

1.3. Выберите группировку данных по названиям отделов. Группировка позволяет вывести названия отделов в виде оформленных подзаголовков (рис. 29).

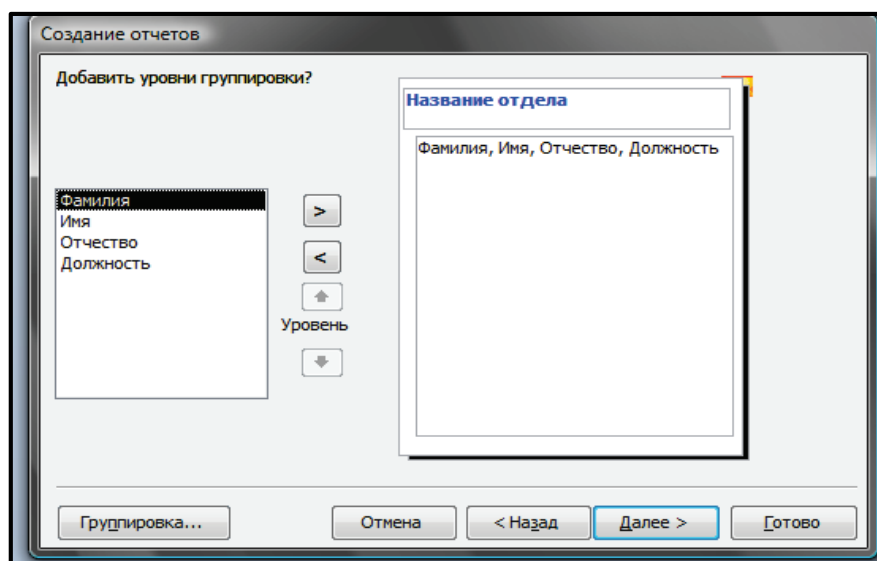


Рис. 29. Группировка данных

1.4. Задайте сортировку по полю **Фамилия** для того, чтобы фамилии в отчете были расположены в алфавитном порядке.

1.5. Выберите вид макета для отчета, например, **Ступенчатый**.

1.6. Введите заголовок отчета: **Список сотрудников**, и завершите работу мастера, щелкнув на кнопке **Готово**.

1.7. После завершения работы мастера включается **Режим предварительного просмотра отчета**. Фрагмент отчета представлен на

рис. 30. При просмотре можно заметить некоторые недочеты, которые требуют исправления, например, название таблицы и название столбцов расположены не по центру.

Сотрудники				
Название отдела	Фамилия	Имя	Отчество	Должность
Отдел рекламы	Борзов	Борис	Александрович	Специалист
Отдел снабжения	Андреева	Елена	Федоровна	Менеджер
	Белых	Алексей	Иванович	Специалист
	Бортникова	Алла	Александровна	Менеджер
Дирекция	Борисов	Борис	Борисович	Менеджер
	Ильина	Анна	Васильевна	Секретарь
	Плужников	Дмитрий	Андреевич	Замдиректора
	Трофимов	Дмитрий	Евгеньевич	Менеджер
Бухгалтерия				

Рис. 30. Фрагмент полученного отчета

1.8. Для исправления отчета перейдите в режим **Конструктора**. Отчет имеет несколько областей, в которых расположены объекты отчета: надписи, поля и другие элементы (табл. 8). Так же как и в форме, исправлять можно только надписи. Поля базы данных исправлять нельзя, так как это может привести к ошибке (рис. 31).

Таблица 8 – Характеристика областей отчета

Область отчета	Характеристика
Заголовок отчета	Информация, которая встречается в начале отчета только на первой странице
Верхний колонтитул	Информация, которая повторяется сверху на каждой странице (заголовки столбцов)
Заголовок группы	Поле, значения которого берутся в качестве заголовков в начале каждой группы
Область данных	Поля, из которых берутся основные данные для отчета
Нижний колонтитул	Информация, которая повторяется внизу на каждой странице (заголовки столбцов)

Сотрудники

По состоянию дел на: [=Now()]

Название отдела

Фамилия

Имя

Отчество

Должность

Название отдела

Фамилия

Имя

Отчество

Должность

=Page " & [Page] & " of " & [Pages]

Рис. 31. Вид областей отчета

Примечание. Чтобы узнать, с каким объектом вы работаете, наведите курсор на объект и кликните правой кнопкой. В появившемся контекстном меню выберите пункт **Свойства**. В титульной строке окна свойств будет написано название объекта.

1.9. Кликните мышкой поочередно на тех полях (эти поля будут выделяться маркерами) которые вы хотите откорректировать.

1.10. Кликните внутри надписи: появится текстовый курсор. Задайте для названий таблицы и столбцов форматирование по центру.

1.11. Во вкладке **Конструктор** выберите **Элементы управления**.

1.12. На панели элементов выделите кнопку **Надпись** и нарисуйте небольшую рамку для надписи в области заголовка. В рамке напишите «По состоянию дел на: <>».

1.13. Перенесите из области нижнего колонтитула в область заголовка объект с функцией **Now()**, который выводит в отчет текущую дату.

1.14. Установите начертание поля группы: полужирный, курсив. Фрагмент откорректированного отчета представлен на рис. 32.

Сотрудники

По состоянию дел на: 12 февраля 2016 г.

Название отдела	Фамилия	Имя	Отчество	Должность
Отдел рекламы	Борзов	Борис	Александрович	Специалист
Отдел снабжения	Андреева	Елена	Федоровна	Менеджер
	Белых	Алексей	Иванович	Специалист
	Бортникова	Алла	Александровна	Менеджер
Дирекция	Борисов	Борис	Борисович	Менеджер
	Ильина	Анна	Васильевна	Секретарь
	Плужников	Дмитрий	Андреевич	Замдиректора
	Трофимов	Дмитрий	Евгеньевич	Менеджер
Бухгалтерия	Иванова	Ирина	Петровна	гл. бухгалтер

Рис. 32. Фрагмент отчета «Сотрудники»

2. Создание Отчета «Дни Рождения». Создайте отчет **Дни Рождения**, в котором формируется список сотрудников и их дни рождения, расположенные в порядке следования в календарном году.

2.1. Запустите **Мастер отчетов**.

2.2. Включите в отчет поля из запроса **Исполнитель: ФИО, Возраст, Дата Рождения, Месяц, День**.

2.3. Задайте сортировку по полю **Месяц**, затем по полю **День**. Перейдите в режим **Конструктор** и откорректируйте надписи в верхнем колонтитуле:

- удалите из макета отчета надписи и поля **Месяц** и **День**;
- исправьте «**ФИО**» на «**Фамилия, Имя, Отчество**»;
- увеличьте размер рамки, отведенной под надпись **Дата Рождения**.

Уменьшите, если необходимо, размер рамок, отведенных под надпись и поле **ФИО**;

- расположите надписи и поля равномерно в пределах строки.

2.4. Фрагмент исправленного отчета представлен на рис. 33.

Дни рождения

Фамилия Имя Отчество	Возраст	Дата рождения
Андреева Елена Федоровна	55	05.01.1961
Борисов Борис Борисович	29	04.02.1987
Белых Алексей Иванович	51	23.03.1965
Борзов Борис Александрович	49	26.04.1967
Абрас Жихарка Семеновна	36	03.05.1980
Плужников Дмитрий Андреевич	60	15.05.1956
Мухин Антон Павлович	55	24.05.1961
Иванова Ирина Петровна	58	30.06.1958

Рис. 33. Фрагмент отчета «Дни рождения»

3. Создание отчета «Представительский». Создайте отчет **Представительский**, в котором создается единая форма нагрудной представительской карточки для сотрудников фирмы. Такая карточка называется «Бэйдж» от английского слова «badge» (знак). Отчет должен содержать эмблему предприятия, фамилию, имя, отчество сотрудника, название отдела и должность. Размер представительской нагрудной карточки 9x5,5. Размер контролируйте по сетке, в которой одна клетка соответствует 1 см.

3.1. Запустите **Мастер отчетов**.

3.2. Включите в отчет поля **Имя, Отчество, Фамилия, Должность** из таблицы **Сотрудники**, поле **Название Отдела** из таблицы **Структура ОАО РАНГОУТ**.

3.3. Выберите вид макета – «**В столбец**», стиль оформления – **Обычный**.

3.4. После создания отчета при помощи мастера перейдите в режим **Конструктора** для корректировки макета отчета.

3.5. Удалите объекты из области заголовка отчета и нижнего колонтитула. Для этого щелчком выделите объект и нажмите на клавишу **Delete**.

3.6. Удалите надписи полей из области данных.

3.7. Уменьшите до нуля высоту всех областей отчета, кроме области данных.

3.8. Измените размеры области данных до размеров нагрудной карточки 9x5,5 см. для этого наведите курсор на правую (или нижнюю) границу области до появления двусторонней стрелки, захватите и переместите границу.

3.9. Измените формат содержимого полей, согласно табл. 9.

Таблица 9 – Стили оформления полей

Название поля	Стиль поля
Имя	Шрифт Arial Сур, размер 14, полужирный, выравнивание по правому краю
Отчество	
Фамилия	
Название Отдела	Шрифт Bookman Old Style, размер 12, полужирный курсив, выравнивание по левому краю
Должность	Шрифт Bookman Old Style, размер 12, полужирный курсив, выравнивание по правому краю

3.10. Добавьте рисунок эмблемы. Для этого:

- откройте созданный ранее файл Эмблема (создайте его, или скачайте уже готовый из Интернет), скопируйте рисунок эмблемы;
- перейдите в окно конструктора отчета и вставьте из буфера обмена скопированный рисунок эмблемы;
- расположите в области данных рисунок эмблемы и поля.

3.11. Обведите все объекты рамкой, выбрав инструмент **Прямоугольник** на панели элементов.

3.12. Измените формат рамки: вид, толщину границы, цвет границы и внутреннего заполнения. Если рамка непрозрачная, поместите ее на задний план. **Макет** отчета представлен на рис. 34.

Примечание. Не забывайте просматривать результаты работы (режим Предварительный просмотр).

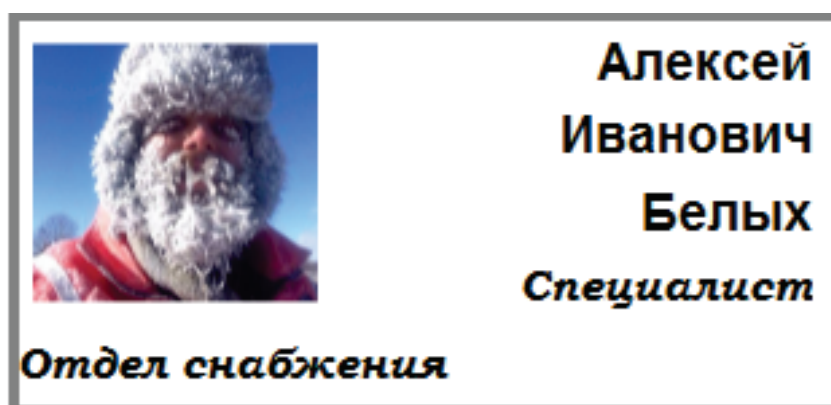


Рис. 34. Макет отчета «Представительский»

Практическое занятие № 7

Учет документов в базе данных

Цель занятия: научиться сопровождать базы данных. Закрепить знания, полученные на предыдущих занятиях.

Отрабатываемые вопросы

1. Создание базы данных документы ОАО Рангоут.
2. Создание поля типа гиперссылка.
3. Создание запроса на выборку для таблицы внутренних документов.
4. Учет входящих документов в базе данных «Документы ОАО Рангоут».

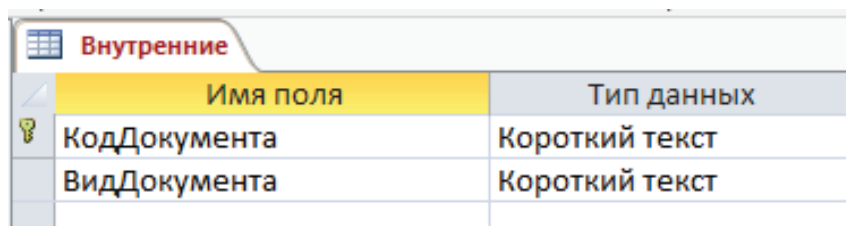
Организационно-методические указания

1. Создание базы данных «Документы ОАО РАНГОУТ». В среде баз данных можно организовать учет всех документов: внутренних, входящих, исходящих. Базы данных позволяют также вести на компьютере журналы учета сотрудников, направляемых в командировку или приезжающих в командировку, журнал учета телефонограмм. Создайте базу данных «Документы ОАО РАНГОУТ» и организуйте в ней учет внутренних документов. Для этого создайте таблицу **Внутренние**, которая содержит перечень всех видов внутренних документов, составляемых на предприятии, и их сокращенное обозначение.

1.1. Создайте новую базу данных с именем **Документы ОАО РАНГОУТ** и сохраните ее в папке **ОАО РАНГОУТ**.

1.2. Выберите объект **Таблицы**.

1.3. Создайте таблицу **Внутренние** со структурой, показанной на рис. 35.



Внутренние	
Имя поля	Тип данных
КодДокумента	Короткий текст
ВидДокумента	Короткий текст

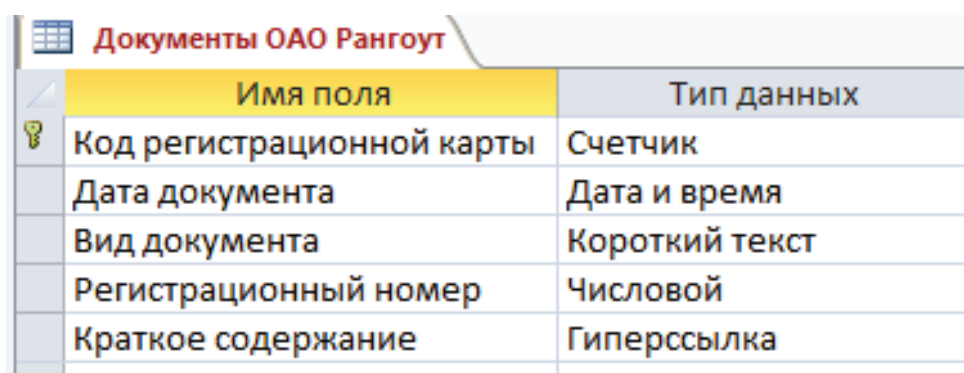
Рис. 35. Структура таблицы «Внутренние»

1.4. В режиме Таблицы введите данные, приведенные в таблице (табл. 10). Ключевое поле содержит условное сокращение названия документа. Во втором поле приведено полное название вида документа.

Таблица 10 – Данные для таблицы «Внутренние»

Код Документа	Вид Документа
АКТ	Акт
ПрЛС	Приказ ЛС (по личному составу)
Пр	Приказ ОД (по основной деятельности)
ПРТ	Протокол
СлЗ	Служебная записка
СПР	Справка

1.5. В режиме **Конструктора таблиц** создайте новую таблицу **Документы ОАО РАНГОУТ** со структурой, представленной на рис. 36.



Документы ОАО Рангоут	
Имя поля	Тип данных
Код регистрационной карты	Счетчик
Дата документа	Дата и время
Вид документа	Короткий текст
Регистрационный номер	Числовой
Краткое содержание	Гиперссылка

Рис. 36. Структура таблицы «Документы ОАО РАНГОУТ»

Пояснения к структуре. Поле **Код Регистр Карты** («код регистрационной карты») имеет тип данных **Счетчик** и является ключевым. При занесении в таблицу нового документа в этом поле автоматически появляется порядковый номер. Таким образом, это поле задает сплошную нумерацию всем внутренним документам в порядке занесения их в таблицу.

В поле **Дата Документа** заносится дата создания документа. Поскольку дата документа очень часто совпадает с днем его подписания и занесения в базу данных, удобно в свойствах поля ввести в строку **Значение** по умолчанию формулу вычисления даты: **= Date()**, а в строке **Формат** (свойства поля) выбрать «Краткий формат даты». Тогда при заполнении базы данных в этом поле будет автоматически появляться текущая дата. При необходимости ее можно исправить.

Поле **Вид Документа** является полем подстановки, связанным с созданной таблицей **Внутренние**. Создание «Столбца подстановок» рассматривалось в практической работе № 2, выполните это самостоятельно.

В поле Рег. Номер («регистрационный номер») предполагается заносить порядковые номера создания и внесения в базу данных отдельных видов документов. Таким образом, значения в этом поле могут повторяться. Например, может быть и приказ по основной деятельности № 1 и приказ по личному составу № 1.

В поле Краткое содержание предполагается заносить заголовок документа и ссылку на место его хранения. Тогда при просмотре базы данных вы можете сразу по ссылке открыть и просмотреть нужный документ.

1.6. Закройте все таблицы.

1.7. Откройте окно **Схема данных**, установите связь «Один – ко – многим» между таблицами **Внутренние** и **Документы ОАО РАНГОУТ**. Введение связи «Один – ко – многим» означает, что в базе данных может быть много документов одного и того же вида. Эта связь, как правило, создается автоматически при создании столбца подстановок, поэтому вам достаточно просмотреть в окне **Схема данных**, что связь организована правильно (рис. 37).

1.8. Установите обеспечение целостности данных (см. практическую работу № 3).

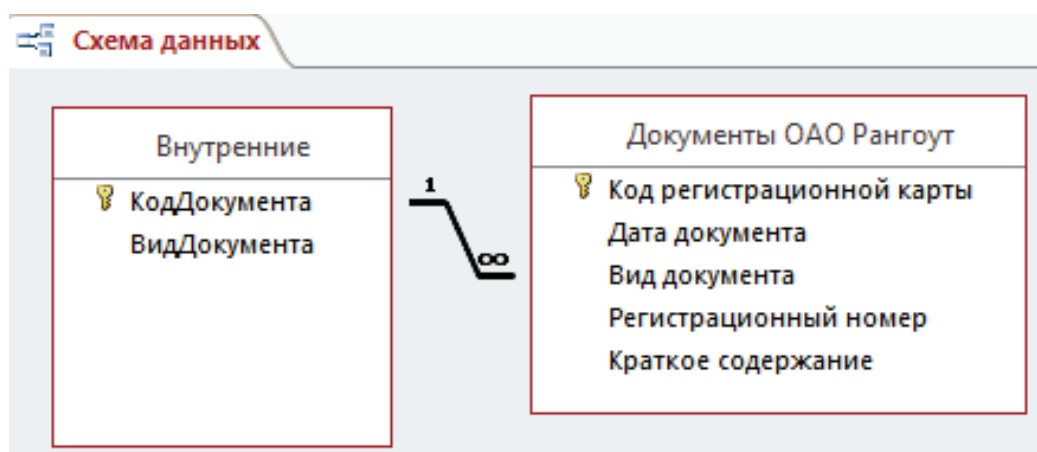


Рис. 37. Схема связи

2. Создание поля типа гиперссылка. Заполните таблицу **Документы ОАО РАНГОУТ** несколькими записями по разным видам документов. В поле типа **Гиперссылка** установите ссылку на соответствующий документ. Перед заполнением таблицы необходимо сохранить несколько файлов различных документов в папках. Это могут быть файлы, созданные и сохраненные вами при выполнении заданий по теме «**Текстовые процессоры**». Очень удобно, если файлы в папках имеют унифицированные названия. Например, файлы приказов по основной деятельности хранятся в папке **Мои документы/Внутренние/Основная деятельность/Приказы** и имеют имена **1-Пр2015, 2-Пр2015, 3-Пр2015** и т. д.; файлы приказов по личному составу

хранятся в папке **Мои документы/Внутренние/Личный состав/Приказы ЛС** и имеют имена **1-ПрЛС2015, 2-ПрЛС2015** и т. д. Такая нумерация указывает и на порядок создания приказов и на год. В конце года файлы можно заархивировать. Это в некотором смысле напоминает операцию передачи документов в архив.

2.1. Откройте таблицу **Внутренние**. Перед первым столбцом находится столбец, содержащий знак «+», который указывает на наличие связанной подчиненной таблицы;

2.2. Кликните на знаке «+» в строке **Приказ ОД**. Раскроется подчиненная таблица **Документы ОАО РАНГОУТ**.

2.3. Введите дату первого приказа, регистрационный номер (**1**) и краткое содержание. Обратите внимание, что при вводе текста краткого содержания, он выделяется синим цветом с подчеркиванием. Это традиционное выделение гиперссылки (рис. 38).

2.4. Кликните правой клавишей мышки на тексте в поле Краткое содержание.

Внутренние			
КодДокуме	ВидДокумента	Щелкните для добавления	
+ АКТ	Акт		
- Пр	Приказ ОД (по основной д		
Код регистр	Дата докум	Регистраци	Краткое содержание
1	12.02.2016	1	<u>Об утверждении структуры</u>
3	12.02.2016	2	<u>О порядке введения документации</u>
4	12.02.2016	3	<u>О графике работы в праздничные дни</u>
*	(№)		
- ПрЛс	Приказ ЛС (по личному со		
Код регистр	Дата докум	Регистраци	Краткое содержание
2	12.02.2016	1	<u>Об утверждении штатного расписания</u>
5	12.02.2016	2	<u>О приеме на работу</u>
6	12.02.2016	3	<u>О представлении отпуска за 2010г.</u>
*	(№)		
+ ПРТ	Протокол		
+ СлЗ	Служебная записка		
+ СПР	Справка		

Рис. 38. Данные для таблицы «Документы ОАО РАНГОУТ»

2.5. Выберите команды **Гиперссылка – Изменить гиперссылку**. Раскроется окно для ввода адреса **Гиперссылки**. В строке **Текст** будет записано краткое содержание документа (рис. 39).

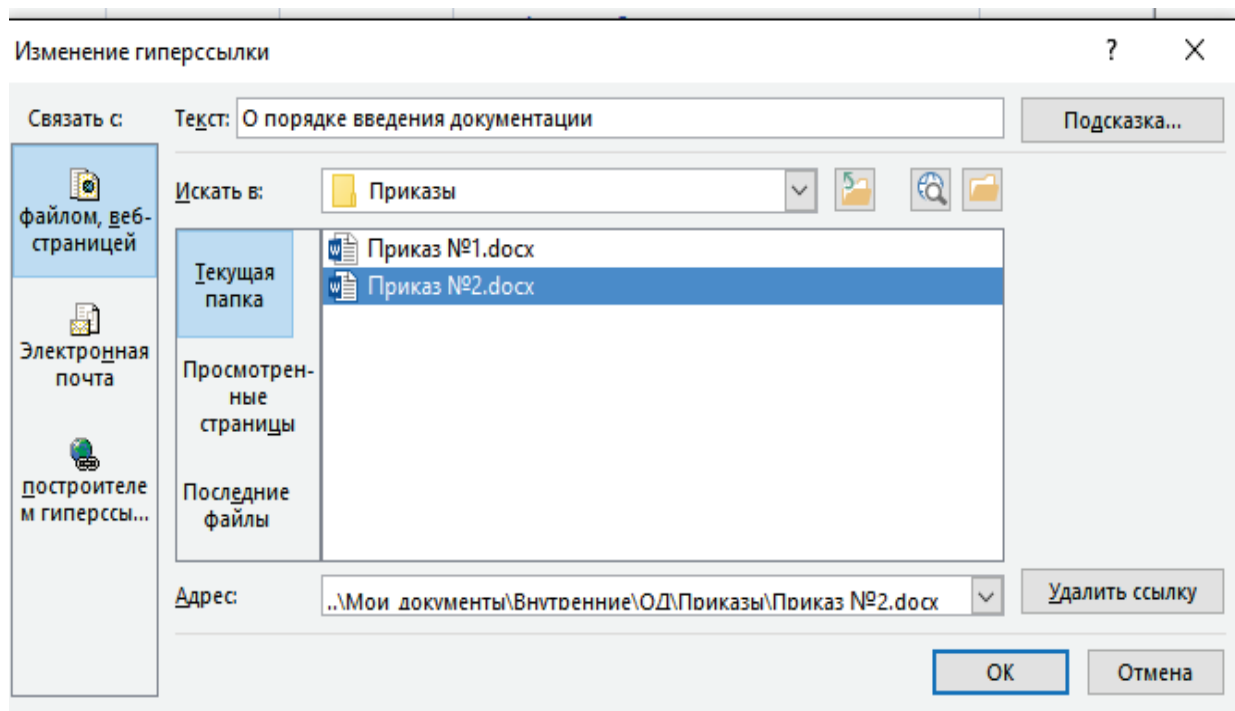


Рис. 39. Изменение гиперссылки

2.6. Введите имя файла документа. Для этого найдите нужное имя в списке последних файлов (как правило, составление документа предшествует записи его в базу данных, поэтому велика вероятность того, что он будет указан в этом списке) или кликните на кнопке **Файлы** в окне обзора выберите нужное имя файла.

2.7. Введите еще несколько созданных документов разного вида в базу данных (рис. 39). Обратите внимание, поле **Код Рег. Карты** заполняется автоматически в порядке ввода информации в базу данных. Регистрационный номер документов каждого вида вводится вручную. На рис. 5 можно увидеть, что приказы по основной деятельности и по личному составу имеют отдельную нумерацию, а учетные номера записей (поле **Код Рег. Карты**) появляются в порядке ввода их в таблицу.

3. Создание запроса на выборку для таблицы внутренних документов. Создайте запрос **Приказы ОД** на выборку из таблицы **Документы ОАО РАНГОУТ** приказов по основной деятельности.

3.1. Запустите **Мастер запросов**.

3.2. Для запроса выберите из таблицы **Документы ОАО РАНГОУТ** поля **Регистрационный Номер, Дата Документа, Краткое содержание, Вид документа**.

3.3. Задайте имя запроса **Приказы ОД**.

3.4. После работы **Мастера** просмотрите результаты запроса. Созданный в таком виде запрос отбирает все документы из таблицы, поэтому надо дополнительно задать условие отбора.

3.5. Перейдите в режим **Конструктора** и запишите в поле **Вид документа** условие отбора **Like "Пр"**, а также отключите флажок **Вывод на экран** (рис. 7).

3.6. Измените поле **Дата Документа** на вычисляемое поле **Дата Приказа** с функцией: **CStr ([Документы ОАО РАНГОУТ]![Дата Документа])** (рис. 40).

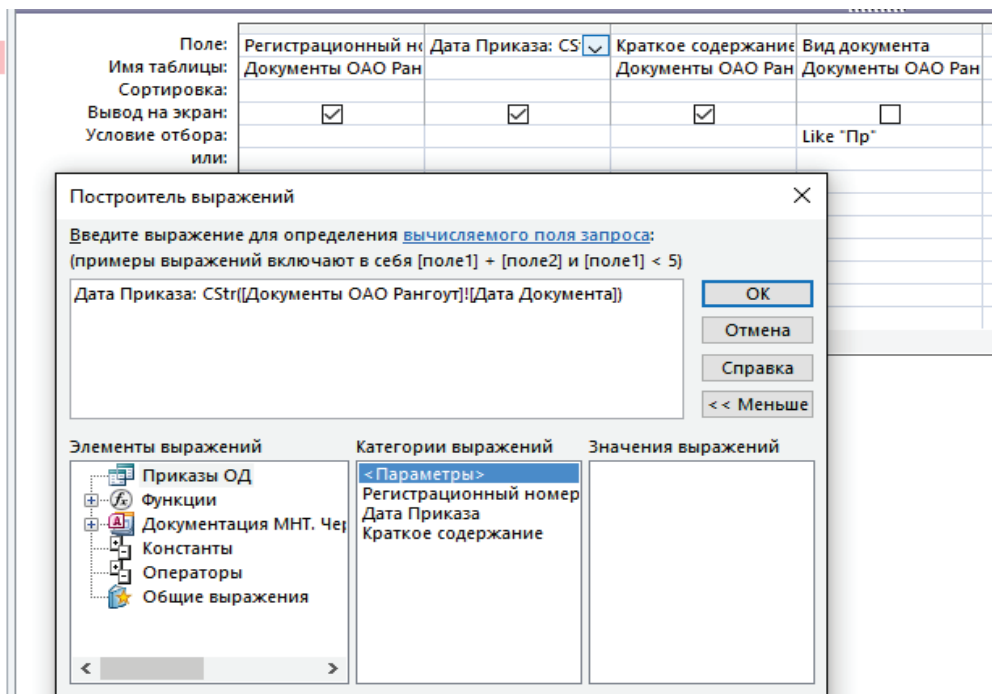


Рис. 40. Бланк запроса

Примечание. Функция **CStr()** преобразует аргумент любого типа, заключенный в скобках, в текстовый тип.

– Дата в текстовом виде применяется для организации слияния информации из базы данных с текстовым документом. Просмотрите итоги работы запроса. Теперь в таблице представлены только приказы по основной деятельности (рис. 41).

Приказы ОД		
Регистраци	Дата Приказа	Краткое содержание
1	12.02.2016	Об утверждении структуры
2	12.02.2016	О порядке введения документации
3	12.02.2016	О графике работы в праздничные дни

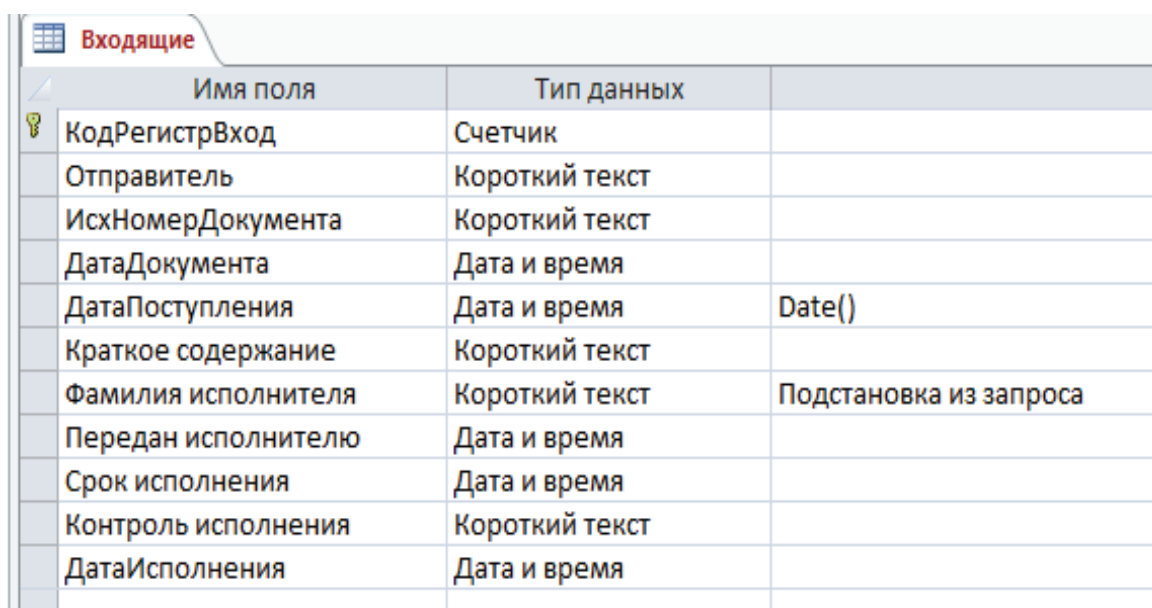
Рис. 41. Выполнение запроса «Приказы по ОД»

Задание для самостоятельной работы

Создайте запросы на выборку внутренних документов другого вида (приказы по личному составу).

4. Учет входящих документов в базе данных «Документы ОАО РАНГОУТ». Входящие документы являются в основном письмами, которые поступают в организацию по почте, то есть в бумажном виде, а не в виде электронного документа. Входящие документы подлежат регистрации в специальном журнале. Такой журнал можно создать в среде баз данных. Создайте таблицу **Входящие** для учета и регистрации входящих документов. На основе этой таблицы создайте: форму для ввода данных, запросы с различными критериями отбора данных, отчет для получения твердой копии сведений из таблицы.

4.1. Рассмотрите структуру таблицы **Входящие**. Она соответствует форме журнала регистрации входящих документов. В ней содержится поле **Фамилия исполнителя**, которое является полем подстановки из запроса **Исполнитель**, базы данных **ОАО Рангоут** (рис. 42).



Имя поля	Тип данных	
КодРегистрВход	Счетчик	
Отправитель	Короткий текст	
ИсхНомерДокумента	Короткий текст	
ДатаДокумента	Дата и время	
ДатаПоступления	Дата и время	Date()
Краткое содержание	Короткий текст	
Фамилия исполнителя	Короткий текст	Подстановка из запроса
Передан исполнителю	Дата и время	
Срок исполнения	Дата и время	
Контроль исполнения	Короткий текст	
ДатаИсполнения	Дата и время	

Рис. 42. Структура таблицы «Входящие»

4.2. Для формирования запроса **Исполнитель** надо установить связь базы данных **Документы ОАО РАНГОУТ** с базой данных **ОАО Рангоут**. Для этого:

- откройте базу данных **Документы ОАО РАНГОУТ**;
- выполните действия: на вкладке **Внешние данные** в группе **Импорт и связи** кликните по кнопке **Access**;

– в окне обзора откройте базу данных ОАО РАНГОУТ и выберите для вставки таблицы Сотрудники и Структура ЗАО ОАО РАНГОУТ. В области переходов базы данных Документы ОАО РАНГОУТ появятся эти таблицы.

4.3. Создайте запрос **Исполнитель**. Включите в запрос поля **Фамилия** из таблицы **Сотрудники** и **Код отдела** из таблицы **Структура ОАО РАНГОУТ**.

4.4. Создайте таблицу **Входящие** со структурой, приведенной на рисунке 9. В ключевом поле **Код Регистр Вход** («код регистрации входящих») будет автоматически устанавливаться номер документа в порядке их внесения в таблицу. В поле **Дата Поступления** введите значение по умолчанию: функцию **=Date()**, которая вычисляет текущую дату. При заполнении таблицы текущая дата всегда будет появляться в поле по умолчанию. При необходимости ее можно изменить. Поле **Фамилия** является полем подстановки из запроса **Исполнитель**.

4.5. Заполните таблицу несколькими записями.

Задания для самостоятельной работы

Задание 1. Создайте форму **Входящие** для ввода данных.

Задание 2. Создайте запрос на отбор входящих документов по критериям, которые приведены в таблице (табл. 11).

Таблица 11 – Критерии запроса

Поле	Условие отбора	Пояснение
<i>Дата Документа</i>	>Now()-30	Отбирает все документы за последний месяц
<i>Фамилия исполнителя</i>	= «Веселов»	Отбирает все документы, переданные для исполнения сотруднику Веселову

Задание 3. Составьте отчет либо по всей таблице, либо по запросу.

3. ОСНОВЫ БАЗЫ ДАННЫХ В MYSQL

Тема: «Организация SQL запросов к СУБД MYSQL».

Цель занятия: получить навыки работы с СУБД MySQL.

Отрабатываемые вопросы

1. Введение и теоретическая часть.
2. Реализация языка SQL в СУБД MySQL.
3. Создание удаленной базы данных MySQL на сервере.

3.1. Введение и теоретическая часть

В СУБД MySQL реализовано подмножество языка SQL, соответствующее спецификации ANSI SQL 92.

MySQL – это система управления базами данных.

База данных представляет собой структурированную совокупность данных. Эти данные могут быть любыми – от простого списка предстоящих покупок до перечня экспонатов картинной галереи или огромного количества информации в корпоративной сети. Для записи, выборки и обработки данных, хранящихся в компьютерной базе данных, необходима система управления базой данных, каковой и является СУБД MySQL. Поскольку компьютеры замечательно справляются с обработкой больших объемов данных, управление базами данных играет центральную роль в вычислениях. Реализовано такое управление может быть по-разному как в виде отдельных утилит, так и в виде кода, входящего в состав других приложений.

MySQL – это система управления реляционными базами данных.

В реляционной базе данных данные хранятся не все скопом, а в отдельных таблицах, благодаря чему достигается выигрыш в скорости и гибкости. Таблицы связываются между собой при помощи отношений, благодаря чему обеспечивается возможность объединять при выполнении запроса данные из нескольких таблиц. SQL в качестве части системы MySQL можно охарактеризовать как язык структурированных запросов плюс наиболее распространенный стандартный язык, используемый для доступа к базам данных.

Программное обеспечение MySQL – это программное обеспечение (ПО) с открытым кодом.

ПО с открытым кодом означает, что применять и модифицировать его может любой желающий. Такое ПО можно получать по Internet и использовать бесплатно. При этом каждый пользователь может изучить исходный код и изменить его в соответствии со своими потребностями. Использование программного обеспечения MySQL регламентируется лицензией GPL (GNU General Public License), <http://www.gnu.org/licenses/>, в которой указано, что

можно и чего нельзя делать с этим программным обеспечением в различных ситуациях [1].

MySQL является очень быстрым, надежным и легким в использовании. Если вам требуются именно эти качества, попробуйте поработать с данным сервером. MySQL обладает также рядом удобных возможностей, разработанных в тесном контакте с пользователями. Первоначально сервер MySQL разрабатывался для управления большими базами данных с целью обеспечить более высокую скорость работы по сравнению с существующими на тот момент аналогами. И вот уже в течение нескольких лет данный сервер успешно используется в условиях промышленной эксплуатации с высокими требованиями. Несмотря на то, что MySQL постоянно совершенствуется, он уже сегодня обеспечивает широкий спектр полезных функций. Благодаря своей доступности, скорости и безопасности MySQL очень хорошо подходит для доступа к базам данных по Internet [2].

3.2. Технические возможности СУБД MySQL

ПО MySQL является системой клиент-сервер, которая содержит многопоточный SQL-сервер, обеспечивающий поддержку различных вычислительных машин баз данных, а также несколько различных клиентских программ и библиотек, средства администрирования и широкий спектр программных интерфейсов (API).

3.3. Типы данных. Обработка числовых данных

В MySQL есть пять целых типов данных, каждый из которых может быть со знаком (по умолчанию) или без знака (при добавлении слова *UNSIGNED* после имени типа).

Таблица 12 – Характеристики числовых типов данных в MySQL

Имя типа	Кол-во бит	Диапазон со знаком	Диапазон без знака
TINYINT	8	-128..127	0..255
SMALLINT	16	-32768..32767	0..65535
MEDIUMINT	24	-8388608..8388607	0..16777215
INTEGER	32	-147483648..2147483647	0..4294967295
BIGINT	64	$-(2^{63})..(2^{63}-1)$	$0..(2^{64})$

Тип *INT* используется как псевдоним для *INTEGER*. Другими псевдонимами служат: *INT1*=*TINYINT*, *INT2*=*SMALLINT*, *INT3*=*MEDIUMINT*, *INT4*=*INT*, *INT8*=*BIGINT* и *MIDDLEINT*=*MEDIUMINT*.

Обработка строковых данных.

MySQL поддерживает следующие строковые типы (*M* обозначает максимальный отображаемый размер или PRECISION).

CHAR(M) – строка фиксированной длины всегда дополняемая справа пробелами. *M* может находиться в диапазоне от 1 до 255 символов.

VARCHAR(M) – строка переменной длины. Замыкающие пробелы удаляются базой данных при записи значения, что отличается от спецификации ANSI SQL. *M* может находиться в диапазоне от 1 до 255 символов.

ENUM('value1' , 'value2',...) – перечисление. Строковый объект, у которого может быть только одно значение, выбранное из заданного списка (или NULL). В перечислении может быть до 65 535 различных значений.

SET('value1' , 'value2' , ..) – множество. Строковый объект, у которого может быть несколько значений (или ни одного), каждое из которых должно выбираться из указанного списка. SET может иметь максимум 64 элемента.

Длина типов **CHAR** и **VARCHAR** ограничена 255 байтами. Все строковые типы поддерживают двоичные символы, включая **NULL** (для литеральных строк используйте метод *\$dbh->quote()*).

Поддерживаются также следующие псевдонимы:

BINARY(num) CHAR(num) BINARY

CHARVARYINGVARCHAR

LONG VARBINARY BLOB

LONG VARCHAR TEXT

VARBINARY(num) VARCHAR(num) BINARY

Основные команды.

Просмотреть список команд программы **mysql** можно, запустив ее с параметром **--help**:

```
shell>mysql --help
```

Выведенный этой командой на экран монитора список команд достаточно хорошо прокомментирован и поэтому здесь не приводится.

Запустить СУБД MySQL можно следующим образом:

```
mysql [OPTIONS] database
```

3.4. Описание СУБД MySQL

Клиентская часть СУБД MySQL названа **mysql**. Она обеспечивает интерфейс командной строки с СУБД MySQL и возможность не интерактивной пакетной обработки.

Опции, поддерживаемые программой **mysql**, представлены в табл. 13. Вы можете использовать или «короткий» одиночный символ или более подробную версию.

В интерактивном режиме **mysql** будет печатать результаты в таблице подобно примеру, приведенному ниже. Если не задан пароль или имя

пользователя, mysql попыбует зайти в систему на сервере базы данных с использованием вашего логина и НУЛЕВОГО (ПУСТОГО) пароля. Если ваш mysql логин отличается от вашего логина в unix, или если у вас пароль, то соединение с MySQL выполнено не будет.

Например:

```
bsd# mysql -u coobic
```

```
Welcome to the MySQL monitor. Commands end with ; or \g.
```

```
Your MySQL connection id is 4 to server version: 4.0.12-max
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

Таблица 13 – Опции MySQL

Опция	Описание
-\?, --help	Справка
-d, --debug=[options]	Вывести в протокол отладочную информацию. В общем виде <i>'d:t:o,filename'</i> .
-d, --debug-info	Вывести отладочную информацию при выходе из программы.
-e, --exec	Выполнить команду и выйти, неявная форма опции – <i>batch</i> .
-f, --force	Продолжить, даже если мы сталкиваемся с SQL ошибкой.
-h, --hostname=[hostname]	Задаёт имя сервера, с которым вы желаете соединиться.
-P, --port=[port]	Порт для соединения с сервером MySQL.
-p, --password=[password]	Пароль пользователя для соединения с сервером MySQL. Обратите внимание, что не должно быть пробела между -p и паролем.
-q, --quick	Быстрый (небуферизованный вывод), может замедлить сервер, если вывод приостановлен.
-s, --silent	Работать молча (подавить вывод)
-u, --user=[user]	Имя пользователя для соединения с сервером MySQL. Необязательно, если имя пользователя такое же, как ваш логин. По умолчанию именно ваш логин используется в качестве имени пользователя, что облегчает настройку.

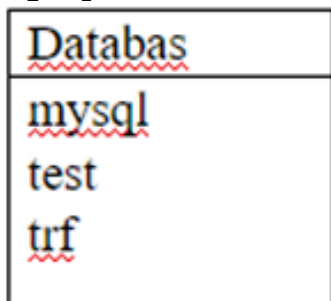
Опция	Описание
-v, --verbose	Подробный вывод. -v опция может быть удвоена или утроена для более подробного вывода. В программах русских авторов обычно именуется «уровнем болтливости программы».
-w, --wait	Если подключение терпит неудачу, то подождать и повторить попытку.
-B, --batch	Выполнить в пакетном режиме. Никаких запросов и никаких ошибок в STDOUT. Устанавливается автоматически при чтении из записи в канал (пайп). Результаты будут выведены в формате с разделением табуляцией. Одна строка результата соответствует одной строке вывода.
-I, --help	Справка, эквивалент -\?
-V, --version	Вывести информацию о версии пакета.

3.5. Реализация языка SQL в СУБД MySQL

3.5.1. Создание базы данных

Просмотреть список существующих баз данных в настоящее время на сервере можно при помощи команды **SHOW** (рис. 44):

```
mysql> SHOW DATABASES;
```



<u>Databas</u>
<u>mysql</u>
<u>test</u>
<u>trf</u>

```
3 rowsinset (0.00 sec)
```

База данных **mysql** просто необходима, так как в ней описываются пользовательские права доступа. База **test** часто применяется для экспериментов.

Впрочем, всех баз вы можете и не увидеть, если у вас нет привилегии **SHOW DATABASES**.

Если база данных **test** существует, попробуйте обратиться к ней:

```
mysql>USE test
Data base changed
```

В команде **USE**, как и **QUIT**, точка с запятой не нужна (конечно, данные команды тоже можно завершать точкой с запятой – никакого вреда от этого не будет). Команда **USE** отличается от остальных и кое-чем еще: она должна задаваться одной строкой.

Если администратор при выдаче разрешения создаст для вас базу, с ней можно сразу начинать работу. При отсутствии БД вам придется создать ее самостоятельно:

```
mysql> CREATE DATABASE perpetuum;
Query OK, 1 row affected (0.00 sec)
```

В Unix имеет значение регистр символов в именах баз данных (в отличие от ключевых слов SQL), так что в этой ОС вам всегда придется называть свою базу **perpetuum**, а не **Perpetuum** или еще как-нибудь. Это же правило распространяется и на имена таблиц (в Windows данное ограничение не действует, однако при обращении к базам и таблицам в пределах одного запроса, тем не менее, можно использовать только один регистр).

При создании базы данных она автоматически не выбирается; выбирать ее нужно отдельно. Сделать **perpetuum** текущей базой можно с помощью следующей команды:

```
mysql> USE perpetuum
Database changed
```

Создавать базу нужно только однажды, но выбирать ее приходится в каждом сеансе работы с **mysql**. Делать это можно с помощью команды **USE**, представленной выше. А можно выбирать базу и из командной строки при запуске **mysql**. Для этого достаточно лишь ввести ее имя после параметров соединения, которые нужно вводить в любом случае. Например:

```
shell>mysql -h host -u user -p perpetuum
Enterpassword: *****
```

Обратите внимание: в вышеприведенной команде **perpetuum** не является паролем. Ввести пароль в командной строке после параметра **-p** можно без пробела (например, **-pmypassword**, а не **-p mypassword**). Впрочем, пароль в командной строке все равно лучше не вводить, так как таким образом его могут и подсмотреть.

Удаление базы данных:

```
mysql> DROP DATABASE perpetuum;
```

3.5.2. Создание таблицы

При помощи команды **CREATE TABLE** определим структуру новой таблицы:

```
mysql> CREATE TABLE TABLE1 (  
-> NUMBER INTEGER PRIMARY KEY NOT NULL  
AUTO_INCREMENT,  
-> FAMALY VARCHAR(35),  
-> NAME VARCHAR(30),  
-> OTCHESTVO VARCHAR(35),  
-> DESCRIPTION BLOB,  
-> PHOTOFILE BLOB,  
-> EMAIL VARCHAR(40),  
-> ZVANIE VARCHAR(34),  
-> DOLGNOST VARCHAR(40),  
-> );
```

Query OK, 0 rows affected (0.06 sec)

Теперь, когда таблица создана, команда **SHOW TABLES** должна вывести следующее:

```
mysql> SHOW TABLES;
```

Tables_in_perpetuum
TABLE1

1 row in set (0.00 sec)

Проверить, правильно ли была создана таблица в соответствии с планом, можно при помощи команды **DESCRIBE**:

```
mysql> DESCRIBE pet;
```

Field	Type	Null	Key	Default	Extra
Name	varchar(20)	YES		NULL	
owner	varchar(20)	YES		NULL	
Species	varchar(20)	YES		NULL	
sex	char(1)	YES		NULL	
birth	date	YES		NULL	
death	date	YES		NULL	

Использовать команду **DESCRIBE** можно в любое время, например, если вы забудете имена столбцов или типы, к которым они относятся.

```
mysql> describe TABLE1;
```

Field	Type	Null	Key	Default	Extra
NUMBER	int(11)	auto_increment	PRI	NULL	
FAMALY	varchar(35)	YES		NULL	
NAME	varchar(30)	YES		NULL	
OTCHESTVO	varchar(35)	YES		NULL	
DESCRIPTION	blob	YES		NULL	
PHOTOFILE	blob	YES		NULL	
EMAIL	varchar(40)	YES		NULL	
ZVANIE	varchar(34)	YES		NULL	
DOLGHOST	varchar(40)	YES		NULL	

```
9 rows in set (0.02 sec)
```

Удаление таблицы:

```
mysql> DROP TABLE TABLE1;
```

Создание внешнего ключа:

```
CREATE DATABASE lab_1;
use lab_1;
create table kafedra(
sifr smallint(6) primary key,
name varchar(15) unique
);
create table prepodss(
ID int primary key auto_increment,
FIO varchar(50) unique,
dol varchar(50),
uch_step varchar(20),
kaf varchar(50),
foreign key(kaf) references kafedra(name)
);
```

3.5.3. Загрузка данных в таблицу

Создав таблицу, нужно позаботиться о ее заполнении. Для этого предназначены команды **LOAD DATA** и **INSERT**

```
mysql> INSERT INTO TABLE1 VALUES (NULL,
-> 'Kulikov', 'Denis', 'Alexandrovich',
-> 'coobic', NULL, 'denis@redcom.ru',
-> NULL, NULL);
```

Query OK, 1 row affected (0.03 sec)

3.5.4. Команда SELECT

Синтаксис:

```
SELECT [STRAIGHT_JOIN] [DISTINCT | ALL] select_expression,...
[FROM tables... [WHERE where_definition] [GROUP BY column,...]
[ORDER BY column [ASC | DESC], ...] HAVING full_where_definition
[LIMIT [offset,] rows] [PROCEDURE procedure_name]]
[INTO OUTFILE 'file_name'... ]
```

Здесь where_definition:

```
where_definition:
where_expr or where_expr [AND | OR] where_expr
```

where_expr имеет формат:

```
where_expr:
column_name [> | >= | = | <> | <= | <]
column_name_or_constant          or          column_name          LIKE
column_name_or_constant or
column_name IS NULL or column_name IS NOT NULL or
(where_definition)
```

3.5.5. Описание команды SELECT

Оператор **SELECT** является краеугольным камнем всего языка SQL. Он используется, чтобы выполнить запросы к базе данных. В MySQL версии меньше 3.21.x использование предложения **WHERE** очень ограничено. **HAVING** будет работать там, где предложение **WHERE** ничего не делает. В основном, использовать функции с **WHERE** нельзя, но можно использовать функции с **HAVING**. **HAVING** по существу есть **WHERE** применительно к результатам. Он используется главным образом для узкой области данных, возвращенных запросом. Вы должны иметь права *select* для использования **SELECT**.

Ниже приведена простая команда, запрашивающая у сервера информацию о его версии и текущей дате. Введите ее в командной строке `mysql>` и нажмите Enter:

```
mysql>SELECTVERSION() , CURRENT_DATE;
```

version()	current_date
4.0.12-max	2003-10-08

```
1rowinset (0.00 sec)
```

```
mysql>
```

Этот запрос иллюстрирует следующие особенности `mysql`:

Команда обычно состоит из SQL-выражения, за которым следует точка с запятой. Из этого правила есть и исключения – команды без точки с запятой. Одним из них является упомянутая выше команда **QUIT**, остальные мы рассмотрим позднее.

Когда пользователь вводит команду, **mysql** отправляет ее серверу для выполнения и выводит на экран сначала результаты, а затем – новую строку **mysql>**, что означает готовность к выполнению новых команд.

mysql выводит результаты работы запроса в виде таблицы (строк и столбцов). В первой строке этой таблицы содержатся заголовки столбцов, а в следующих строках – собственно результаты. Обычно заголовками столбцов становятся имена, полученные из таблиц базы. Если же извлекается не столбец таблицы, а значение выражения (как это происходит в приведенном выше примере), **mysql** дает столбцу имя запрашиваемого выражения.

mysql сообщает количество возвращаемых строк и время выполнения запроса, что позволяет в некоторой степени составить представление о производительности сервера. Эти значения обычно весьма впечатляют, так как представляют обычное (а не машинное время), кроме того, на них оказывает влияние загрузка сервера и скорость работы сети (для сокращения размеров листингов в остальных примерах этой главы строка "rows in set" удалена).

Для ввода ключевых слов можно использовать любой регистр символов. Приведенные ниже запросы абсолютно идентичны:

```
mysql>SELECT VERSION() , CURRENT_DATE;
mysql>select version() , current_date;
mysql>SeLeCt vErSiOn() , current_DATE;
```

А это – еще один запрос. В нем демонстрируется использование **mysql** в качестве несложного калькулятора:

```
mysql>SELECT SIN(PI()/4) , (4+1)*5;
```

SIN(PI()/4)	(4+1)*5
0.707107	25

Все команды, представленные выше, были относительно короткими и состояли из одной строки. В одну строку можно поместить и несколько команд. Но каждая из них должна заканчиваться точкой с запятой:

```
mysql>SELECT VERSION;раст() ; SELECT NOW() ;
```

VERSION()
4.0.12-max

1 row in set (0.00 sec)

NOW()
2003-10-08 23:22:38

1 row in set (0.00 sec)

Вот пример несложного выражения, занимающего несколько строк:

```
mysql>SELECT
->USER()
->,
->CURRENT_DATE;
```

user()	current_date
coobic@localhost	2003-10-08

1 row in set (0.00 sec)

Если вы решите отменить исполнение набираемой команды, наберите:

```
mysql>SELECT
->USER()
->\c
mysql>
```

Обратите внимание на метку: после ввода команды \c она снова принимает вид mysql>, показывая, что программа **mysql** перешла в режим ожидания указаний.

В табл. 14 приведены все возможные варианты вида метки командной строки и соответствующие им состояния **mysql**:

Таблица 14 – Метки командной строки

Метка	Значение
mysql>	Ожидание новой команды
->	Ожидание следующей строки многострочной команды
'>	Ожидание следующей строки, сбор строкового выражения, начинающегося с одиночной кавычки (' ')
">	Ожидание следующей строки, сбор строкового выражения, начинающегося с двойной кавычки (" ")

Обычно многострочные команды получаются случайно, когда хочешь создать обычную команду, но забываешь поставить завершающую точку с запятой. В таком случае **mysql** ожидает продолжения:

```
mysql>SELECT USER()
->
```

Если с вами произошло подобное (вы думаете, что завершили команду, но программа выдает только метку ->), то **mysql**, вероятнее всего, ждет точки с запятой. Не обратив внимание на метку командной строки, можно довольно долго ждать выполнения команды, не понимая в чем дело. А достаточно лишь поставить точку с запятой, завершив команду, которую **mysql** и выполнит:

```
mysql>SELECT USER()
-> ;
```

user()
coobic@localhost

1 row in set (0.00 sec)

Метки '>' и '>' используются при сборе строк. В MySQL строки можно заключать как в одинарные (' '), так и в двойные (' "') кавычки (можно, например, написать 'hello' или "goodbye"), к тому же mysql позволяет вводить строковые выражения, состоящие из нескольких строчек текста. Метка '>' или '>' обозначает, что вы ввели строку, открывающуюся символом кавычек " или '"', но еще не ввели завершающую строковое выражение закрывающую кавычку.

Это, конечно, нормально, если вы собираетесь создать большое строковое выражение из нескольких строчек. Но это не слишком частый случай. Гораздо чаще оказывается, что вы просто забыли поставить закрывающую кавычку. Например:

```
mysql>SELECT * FROM my_table WHERE name = "Smith AND
age < 30;
">
```

Если ввести такую команду **SELECT**, нажать Enter и подождать результатов, ничего не произойдет. Тут-то и нужно обратить внимание на метку командной строки, выглядящую вот так: ">". Это значит, что **mysql** ждет ввода завершающей части строки. (Теперь заметили ошибку в команде? В строке "Smith нет закрывающей кавычки.)

Что делать в этом случае? Проще всего было бы отменить команду. Однако теперь просто набрать \c нельзя, так как mysql примет эти символы за часть собираемой строки! Вместо этого нужно ввести закрывающие кавычки (тем самым дав mysql понять, что строка закончилась) и лишь затем набрать \c:

```
mysql>SELECT * FROM my_table WHERE name = "Smith AND
age < 30;
"> "\c
mysql>
```

Метка командной строки снова примет вид mysql>, показывая готовность **mysql** к выполнению команд.

Знать значение меток '>' и '>' необходимо, так как при вводе незавершенной строки все последующие строки будут игнорироваться mysql – включая строку с командой **QUIT!** Это может основательно сбить с толку, особенно, если не знать, что для отмены команды перед соответствующей последовательностью символов необходимо поставить закрывающую кавычку.

Команда Delete

Синтаксис команды **DELETE** полностью соответствует спецификации SQL92, поэтому в данных методических указаниях не приводится.

3.5.6. Особенности СУБД MySQL

Ядро MySQL само удаляет пробелы, находящиеся в конце строки. Другая характерная особенность состоит в том, что при сравнениях и сортировке регистр символов в колонках *CHAR* и *VARCHAR* не учитывается, если только не задать атрибут *BINARY*, например:

```
CREATE TABLE foo (A VARCHAR(10)BINARY);
```

В версиях MySQL после 3.23 можно осуществлять сравнение строк без учета регистра с помощью модификатора *BINARY*:

```
SELECT * FROM table WHERE BINARY column = "A"
```

Национальные символы обрабатываются в сравнениях соответственно системе кодировки, указанной во время компиляции, по умолчанию ISO-8859-1. Системы кодировки, не соответствующие ISO, в частности UTF-16, не поддерживаются.

Конкатенация строк производится с помощью функции SQL *CONCAT*(s1, s2, ...).

Обработка данных типа «дата»

MySQL поддерживает следующие типы даты и времени:

DATE

Дата в диапазоне с 0000-01-01 по 9999-12-31. MySQL выводит значения *DATE* в формате *YYYY-MM-DD*, но позволяет присваивать значения колонкам *DATE* в таких форматах:

YYMMDD

YYYYMMDD

YY..MM.DD

YYYY.MM.DD

Здесь «.» может быть любым нецифровым разделителем, а при задании года двумя цифрами предполагается год 20YY, если YY меньше 70.

DATETIME

Комбинация даты и времени. Поддерживаемый диапазон от 0000-01-01 00:00:00 до 9999-12-31 23:59:59. MySQL выводит значения *DATE-TIME* в формате *YYYY-MM-DD HH:MM:SS*, но допускает присвоение значений колонкам *DATETIME* с использованием показанных выше форматов для *DATE* с добавлением *HH:MM:SS*.

TIMESTAMP(M)

Временная метка. Диапазон от 1970-01-01 00:00:00 и до некоторого момента в 2032 или 2106 году, в зависимости от специфического для операционной системы типа *time_t*. MySQL выводит значения *TIMESTAMP* в форматах *YYYYMMDDHHMMSS*, *YYMMDDHHMMSS*, *YYYYMMDD* или *YYMMDD*, в соответствии со значением *M*, равным 14 (или же отсутствием подобного значения), 12, 8 или 6, но присваивать значения колонкам *TIMESTAMP* позволяет с помощью строк или чисел. Этот формат вывода не согласуется с руководством, поэтому его нужно проверять в каждой версии, т. к. он может измениться.

Колонки типа `TIMESTAMP` удобно использовать для регистрации времени операций `INSERT` или `UPDATE`, поскольку если для них не указывается значение, им присваивается время последней операции. Текущее время устанавливается и при попытке присвоить значение `NULL`.

TIME

Время в диапазоне от 838:59:59 до 838:59:59. MySQL выводит значения времени в формате `HH:MM:SS`. Присваивать значения колонкам `TIME` можно с помощью формата `[[[DAYS] [H]H:`

MM:]SS[:fraction] или `[[[[H]H]H]H]MM]SS [.fraction]`.

YEAR

Год. Допустимыми значениями являются 1901-2155 и 0000 в формате четырех знаков и 1970-2069 при использовании двух цифр (70-69). При вводе двузначные годы в диапазоне 00-69 воспринимаются как 2000-2069. (`YEAR` является новым типом данных в MySQL 3.22.)

Обработка данных типа LONG/BLOB

В MySQL поддерживаются следующие типы BLOB:

- 1) `TINYBLOB` / `TINYTEXT` максимальный размер 255 ($2^{**}8 - 1$)
- 1) `BLOB` / `TEXT` максимальный размер 65535 ($2^{**}16 - 1$)
- `MEDIUMBLOB` / `MEDIUMTEXT` максимальный размер 16777215 ($2^{**}24 - 1$)
- `LOB` / `LONGTEXT` максимальный размер 4294967295 ($2^{**}32 - 1$)

Во всех типах BLOB допускаются двоичные символы.

Максимальный размер значений параметров метода `bind_param()` ограничен только максимальным размером команды SQL. По умолчанию это 1 Мбайт, но можно его увеличить почти до 24 Мбайт, изменив в `mysqld` переменную `max_allowed_packet`.

Передавать `TYPE` или другие атрибуты методу `bind_param()` при связывании этих типов не нужно.

Имена таблиц и колонок

В MySQL имена таблиц и колонок должны быть не длиннее 64 символов.

В MySQL можно заключать имена таблиц и колонок в одиночные кавычки (использование стандартных двойных кавычек допустимо, если база данных запущена с ключом `--ansi-mode`).

Это необходимо делать, если имя содержит специальные символы или зарезервированное слово. Ограничения на имена таблиц вызваны тем, что таблицы хранятся в файлах и имена таблиц являются, в действительности, именами файлов. В частности, чувствительность имен таблиц к регистру определяется файловой системой, также недопустимы некоторые символы, например «.» и «/».

В именах колонок регистр не различается, но имена сохраняются без преобразования регистра.

В MySQL в именах могут использоваться символы национальных алфавитов (с установленным восьмым битом).

Автоматическая генерация ключей и последовательностей

Для всех целочисленных полей в MySQL можно установить атрибут *AUTO_INCREMENT*. Это значит, что если есть таблица:

```
CREATE TABLE a (  
id INTEGER AUTO_INCREMENT NOT NULL PRIMARY KEY,  
...)
```

И команда:

```
INSERT INTO a (id,...) VALUES (NULL...),
```

то автоматически генерируется уникальный ID (то же, если поле ID вообще не указывать в команде вставки). Сгенерированный ID позднее можно извлечь:

```
$sth->{mysql_insertid} (1 21_xx)
```

```
$sth->{insertid} (1 20_xx)
```

Либо, если вы используете *\$dbh->do*, а не *prepare/execute*, выполните команды:

```
$dbh->{mysql_insertid} (1 21_xx)
```

```
$dbh->do("SELECT LAST_INSERT_ID() " ); (1 20_xx)
```

MySQL не поддерживает непосредственно генераторы последовательностей, но они легко могут эмулироваться (детали можно найти в руководстве по MySQL). Например:

```
UPDATE seqSETid=last_insert_id(id+1)
```

3.5.7. Создание удаленной базы данных MySQL на сервере

Ознакомиться с операциями, производимыми с консоли СУБД и выполнить следующие действия:

- создать новую удаленную базу данных MySQL на сервере, названием которой будет is_№_в_журнале, пароль для доступа №_в_журнале;
- просмотреть доступные пользователю базы данных и версию СУБД;
- создать в созданной базе данных одну таблицу;
- отобразить структуру таблицы;
- добавить 5 записей в таблицу;
- просмотреть все записи в таблице;
- удалить одну запись в таблице;
- очистить таблицу;
- проверить различные варианты выборки (поиска) по значению одного из полей таблицы;
- таблица должна содержать минимум 5 строковых полей, 2 – цифровых, 1 – для хранения двоичных файлов (рисунков, фотографий);

- убедиться в работоспособности СУБД и продемонстрировать ее работу преподавателю;
- Составить отчет по работе.

Содержание отчета:

1. Цель работы.
2. SQL-скрипт, использованный для создания БД.
3. Структура созданной таблицы.
4. Пример наполнения таблицы.
5. Версия СУБД и доступные пользователю базы данных.

Контрольные вопросы

- Основные возможности СУБД MySQL и поддерживаемые платформы?
- Типы данных, с которыми работает СУБД MySQL?
- Команды, используемые для подключения к СУБД в консольном режиме?
- Правила работы в консольном режиме с СУБД MySQL?
- Особенности работы с СУБД MySQL?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Документация к серверу MySQL. [Электронный ресурс]. – URL: <http://www.mysql.org>
2. Карпова, И. П. Базы данных. Курс лекций и материалы для практических занятий [Текст]: учебное пособие / И. П. Карпова. СПб.: "Питер", 2013. – 240 с.
3. Коваленко, Т. А., Сирант, О. В. Работа с базами данных [Текст] / Т. А. Коваленко, О. В. Сирант. – М.: ИНТУИТ, 2016. – 150 с.
4. Петров, Г. А. Базы данных [Текст]: учебное пособие/ Г. А. Петров, С. В. Тихов, В. П. Яковлев; М-во образования и науки РФ, СПбГТУРП. – СПб.: СПбГТУРП, 2015. – 74 с.
5. Петров, Г. А., Стеклова, Г. А., Тихов, С. В. Практикум по информатике. Часть 1 [Текст] / Г. А. Петров, Г. А. Стеклова, С. В. Тихов. – СПб.: ВШТЭ СПбГУПТД, 2017. – 58 с.
6. Тихов, С. В., Кушнеров, А. И. Базы данных [Текст]: методические указания по выполнению курсовой работы /сост. С. В. Тихов, А. И. Кушнеров; М-во науки и высшего образования РФ, ВШТЭ СПбГУПТД. – СПб.: ВШТЭ СПбГУПТД, 2019 – 34 с.

Учебное издание

Тихов Сергей Владимирович
Кушнеров Александр Игоревич

Базы данных

Практикум

Редактор и корректор А. А. Чернышева
Техн. редактор Д. А. Романова

Темплан 2021 г., поз. 45

Подписано к печати 23.11.22.

Формат 60x84/16.

Бумага тип № 1.

Печать офсетная.

Печ.л. 4,3.

Уч.-изд. л. 4,3.

Тираж 30 экз. (1 завод)

Изд. № 45.

Цена «С». Заказ №

Ризограф Высшей школы технологии и энергетики СПбГУПТД,
198095, Санкт-Петербург, ул. Ивана Черных, 4.