МИНИСТЕРСТВО ОБЩЕГО И ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ РОСТОВСКОЙ ОБЛАСТИ
ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ РОСТОВСКОЙ ОБЛАСТИ
«РОСТОВСКИЙ-НА-ДОНУ КОЛЛЕДЖ РАДИОЭЛЕКТРОНИКИ, ИНФОРМАЦИОННЫХ И ПРОМЫШЛЕННЫХ ТЕХНОЛОГИЙ»
(ГБПОУ РО «РКРИПТ»)

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ РАБОТ ПО ДИСЦИПЛИНЕ

ОП.08 ОСНОВЫ ПРОЕКТИРОВАНИЯ БАЗ ДАННЫХ

Специальность:
09.02.07 Информационные системы и программирование

Квалификация выпускника:
разработчик веб и мультимедийных приложений

Форма обучения: очная

2023

СОГЛАСОВАНО
Начальник методического отдела

_____ Н.В. Вострякова

«26» апреля 2023 г.

УТВЕРЖДАЮ
Заместитель директора
по учебно-методической работе

_____ С.А.Будасова

«26» апреля 2023 г.

ОДОБРЕНО
Цикловой комиссией

_____

Пр. № 8 от «26» апреля 2023 г.
Председатель ЦК

_____ В. Н. Лохмин

Методические указания по выполнению практических (лабораторных) работ разработаны в соответствии с рабочей программой учебной дисциплины **ОП.08 Основы проектирования баз данных** специальности **09.02.07 Информационные системы и программирование**

Разработчик: Государственное бюджетное профессиональное образовательное учреждение Ростовской области «Ростовский-на-Дону колледж радиоэлектроники, информационных и промышленных технологий» (ГБПОУ РО «РКРИПТ»)

СОДЕРЖАНИЕ

Введение

Практические занятия по учебной дисциплине «ОП.08. Основы проектирования баз данных» составляют важную часть теоретической и профессиональной практической подготовки и направлены на подтверждение теоретических положений и формирование практических умений и практического опыта:

– проектировать реляционную базу данных;

– использовать язык запросов для программного извлечения сведений из баз данных

Лабораторные и практические занятия относятся к основным видам учебных занятий.

Выполнение студентами лабораторных и практических работ направлено:

– на обобщение, систематизацию, углубление, закрепление полученных теоретических знаний по конкретным темам дисциплин;

– формирование умений применять полученные знания на практике;

– реализацию единства интеллектуальной и практической деятельности;

– развитие интеллектуальных умений (аналитических, проектировочных, конструкторских и др.) у будущих специалистов;

– выработку при решении поставленных задач таких профессионально значимых качеств, как самостоятельность, ответственность, точность, творческая инициатива.

Ведущей дидактической целью лабораторных занятий является экспериментальное подтверждение и проверка существенных теоретических положений (законов, зависимостей).

Ведущей дидактической целью практических занятий является формирование практических умений – профессиональных (выполнять определенные действия, операции, необходимые в последующем в профессиональной деятельности) или учебных (решать задачи по математике, физике, химии, информатике и др.), необходимых в последующей учебной деятельности.

Содержанием лабораторных работ по дисциплине /профессиональному модулю являются экспериментальная проверка формул, методик расчета, установление и подтверждение закономерностей, ознакомление с методиками проведения экспериментов, установление свойств веществ, их качественных и количественных характеристик, наблюдение развития явлений, процессов и др. В ходе выполнения заданий у студентов формируются практические умения и навыки обращения с различными приборами, установками, лабораторным оборудованием, аппаратурой, которые могут составлять часть профессиональной практической подготовки, а также исследовательские умения (наблюдать, сравнивать, анализировать, устанавливать зависимости, делать выводы и обобщения, самостоятельно вести исследование, оформлять результаты).

Содержанием практических занятий по дисциплине /профессиональному модулю являются решение разного рода задач, в том числе профессиональных (анализ производственных ситуаций, решение ситуационных производственных задач, выполнение профессиональных функций в деловых играх и т.п.),

выполнение вычислений, расчетов, чертежей, работа с измерительными приборами, оборудованием, аппаратурой, работа с нормативными документами, инструктивными материалами, справочниками, составление проектной, плановой и другой технической и специальной документации и другое.

Содержание практических, лабораторных занятий охватывают весь круг профессиональных умений, на подготовку к которым ориентирована данная дисциплина/профессиональный модуль, которые в дальнейшем закрепляются и совершенствуются в процессе курсового проектирования, практикой по профилю специальности и преддипломной практикой.

Лабораторные занятия проводятся в специально оборудованных учебных лабораториях. Практическое занятие должно проводиться в учебных кабинетах или специально оборудованных помещениях (площадках). Продолжительность занятия – не менее 2-х академических часов. Необходимыми структурными элементами занятия, помимо самостоятельной деятельности студентов, являются инструктаж, проводимый преподавателем, а также организация обсуждения итогов выполнения работы.

Все студенты, связанные с работой в лаборатории, обязаны пройти инструктаж по безопасному выполнению работ, о чем расписываются в журнале инструктажа по технике безопасности.

Выполнению лабораторных и практических работ предшествует проверка знаний студентов, их теоретической готовности к выполнению задания.

Лабораторные и практические работы студенты выполняют под руководством преподавателя. При проведении лабораторных и практических занятий учебная группа может делиться на подгруппы численностью не менее 8 человек. Объем заданий для лабораторных и практических занятий спланирован с расчетом, чтобы за отведенное время они могли быть выполнены качественно большинством студентов.

Формы организации работы обучающихся на лабораторных работах и практических занятиях: фронтальная, групповая и индивидуальная.

При фронтальной форме организации занятий все студенты выполняют одновременно одну и ту же работу. При групповой форме организации занятий одна и та же работа выполняется бригадами по 2 - 5 человек. При индивидуальной форме организации занятий каждый студент выполняет индивидуальное задание.

Отчет по практической и лабораторной работе представляется в электронном виде в формате, предусмотренном шаблоном отчета по практической, лабораторной работе. Защита отчета проходит в форме доклада обучающегося по выполненной работе и ответов на вопросы преподавателя.

Оценки за выполнение лабораторных работ и практических занятий могут выставляться по пятибалльной системе или в форме зачета и учитываться как показатели текущей успеваемости студентов.

Критерии оценки лабораторных, практических работ.

Оценка «5» ставится, если учащийся выполняет работу в полном объеме с соблюдением необходимой последовательности проведения опытов и измерений; самостоятельно и рационально монтирует необходимое оборудование; все опыты проводит в условиях и режимах, обеспечивающих получение правильных результатов и выводов; соблюдает требования правил безопасности труда; в отчете пра-

вильно и аккуратно выполняет все записи, таблицы, рисунки, чертежи, графики, вычисления; правильно выполняет анализ погрешностей.

Оценка «4» ставится, если выполнены требования к оценке «5», но было допущено два - три недочета, не более одной негрубой ошибки и одного недочёта.

Оценка «3» ставится, если работа выполнена не полностью, но объем выполненной части таков, позволяет получить правильные результаты и выводы: если в ходе проведения опыта и измерений были допущены ошибки.

Оценка «2» ставится, если работа выполнена не полностью и объем выполненной части работы не позволяет сделать правильных выводов: если опыты, измерения, вычисления, наблюдения
производились неправильно.

ПРАКТИЧЕСКАЯ РАБОТА № 1.
Нормализация реляционной БД, освоение принципов проектирования БД

1. Цель работы

Получить практический опыт нормализации базы данных.

2. Время выполнения работы – 2часа.

3. Краткие теоретические сведения

Общая методика нормализации БД

Одной из проблем разработки БД является проблема группировки данных в БД. Существует много способов группировки элементов данных; одни из них лучше, другие — хуже, а третьи могут привести к сложным проблемам.

Большинство БД непрерывно изменяется. Добавляются новые элементы данных, между ними устанавливаются новые связи и определяются новые способы их использования. При изменении БД необходимо сохранять старое представление пользователя о данных для того, чтобы избежать необходимости переписывать старые программы заново. Однако возможны и такие изменения связей между данными, которые требуют модификации программ. Например, иногда оказывается необходимым расщепить запись или сегмент на две части, или же изменить ключ некоторых элементов. Изменения такого рода являются крайне нежелательными, т.к. могут привести к разрушению структуры БД. В то же время такие разрушения можно сделать маловероятными, если группировка элементов данных и их ключи первоначально были хорошо продуманы.

Рассматриваемая методика рациональной группировки элементов данных, называемая нормализацией БД, является наиболее распространенной основой разработки структур БД, минимизирующей вероятность их разрушения.

Нормализация — это формальный аппарат ограничений на формирование таблиц, устраняющий дублирование, обеспечивающий непротиворечивость хранимых данных и уменьшающий трудозатраты на ведение БД.

Процесс нормализации заключается в разложении (декомпозиции) исходных отношений БД на более простые. При этом на каждой ступени этого процесса схемы отношений приводятся к нормальным формам. Для каждой ступени нормализации имеются наборы ограничений, которым должны удовлетворять отношения БД. Процесс нормализации имеет своей целью устранение избыточности данных и заключается в последовательном приведении БД к третьей нормальной форме (ЗНФ).

Первая нормальная форма (1НФ) требует, чтобы каждое поле таблицы БД было неделимым и не содержало повторяющихся групп.

Неделимость поля означает, что содержащиеся в нем значения не должны делиться на более мелкие поля. Например, если в поле «Подразделение» содержится название факультета и название кафедры, требование неделимости не соблюдается и необходимо выделить название факультета или кафедры в отдельное

поле; поле, содержащее Фамилию, имя и отчество, следует разделить на три поля — отдельно для фамилии, имени и отчества и т.д.

Повторяющимися являются поля, содержащие одинаковые по смыслу значения, например, если требуется получить статистику продаж четырех товаров по месяцам, можно создать поля для хранения данных о продаже по каждому товару (Таблица 1).

Таблица1.

Статистика продаж
Год **Месяц** Товар 1 Товар 2 Товар 3 Товар 4

Однако такой подход не годится, если количество товаров заранее не известно. Повторяющиеся группы следует устранить, сохранив в таблице единственное поле «Товар» (Таблица 2.) В результате получим запись, содержащую информацию о статистике продаж по одному товару, но этот товар может быть любым: для 4 товаров будем иметь 4 записи, для 104 товаров — 104 записи и т д.

Таблица 2.

Статистика продаж
Год **Месяц** Товар

Рассмотрим пример приведения БД к 1НФ. Пусть необходимо автоматизировать процесс отпуска товаров со склада по накладной, примерный вид которой показан в Таблице 3.

Таблица 3. Накладная

	Накладная 123			
Дата	Покупатель	Адрес		
10.01.10	ООО «Кормилец»	г. Иркутск, ул. Трактовая, 20		
Отпущенный товар	Количество	Ед. измерения	Цена	Общая стоимость
Тушенка Сахар Макароны	10000 200 1000	Банки кг кг	700 500 300	7000000 100000 300000
Итого				7400000

Сведем имеющиеся данные в одну таблицу. Приводя ее к 1НФ, учтем, что

впоследствии будет необходимо производить анализ продаж по городам. Поэтому из поля «Адрес» (допускающего толкование как делимого поля) выделим часть данных (город) в отдельное поле «Город».

Известно, что каждый покупатель может закупить в один день различное количество товаров, однако, чтобы не создавать повторяющихся групп, фиксируем факт отпуска каждого товара в отдельной записи. В результате получим таблицу, показанную в Таблице 4.

Вторая нормальная форма (2НФ) требует, чтобы все поля таблицы зависели от первичного ключа, т.е. чтобы первичный ключ однозначно определял запись и не был избыточен. Поля, которые зависят только от части первичного ключа, должны быть выделены в отдельные таблицы.

Таблица 4.

Отпуск товаров со склада
Дата Покупатель Город Адрес Товар Ед_измерения Цена_за_ед_измерения Отпущено_ед Общая_стоимость №_накладной

Для приведения к 2НФ выделим поля, которые входят в первичный ключ. Дата накладной и номер накладной по отдельности не могут уникально определять запись, поскольку они будут одинаковы для всех записей, относящихся к одной и той же накладной (напомним, что одна накладная в Таблице 9.4 представляется несколькими записями). Поэтому введем в первичный ключ поле «Товар». При этом исходим из предположения, что по одной накладной может быть отпущено одно наименование конкретного товара, т.е. не может быть ситуации, когда отпуск одного и того же товара оформляется в накладной двумя строками, что повлекло бы за собой две одинаковые записи в таблице «Отпуск товаров со склада».

Таблица 5.

Отпуск товаров со склада
Дата **Покупатель** **Товар** **№_накладной** Город Адрес Ед_измерения Цена_за_ед_измерения Отпущено_ед

Общая_стоимость

В таблице 5. показана структура таблицы после выделения полей сцепленного первичного ключа (эти поля отчеркнуты от прочих полей линией и располагаются в верхней части структуры таблицы). Созданный нами первичный ключ является избыточным: поле «№ накладной» однозначно определяет дату и покупателя. Для данной накладной не может быть никакой иной даты и никакого иного покупателя. Поле «Товар» в комбинации с № накладной, напротив, однозначно идентифицирует запись. После устранения избыточности первичного ключа получаем Таблицу 6.

Таблица 6.

Отпуск товаров со склада
Товар **№_накладной** Дата Покупатель Город Адрес Ед_измерения Цена_за_ед_измерения Отпущено_ед Общая_стоимость

Первое требование 2НФ выполнено, чего не скажешь о втором, гласящем, что значения всех полей записи должны однозначно зависеть от совокупного значения первичного ключа и не должна иметь место ситуация, когда некоторые поля зависят от части первичного ключа. В Таблице 9.6 поля «Единица измерения», «Цена за единицу измерения» зависят от значения поля «Товар», входящего в первичный ключ. Поэтому выделяем эти поля в самостоятельную таблицу «Товары» и определяем связь: поскольку один товар может присутствовать во многих накладных, таблицы «Товары» и «Отпуск товаров со склада» находятся в связи 1:М (рис. 1).

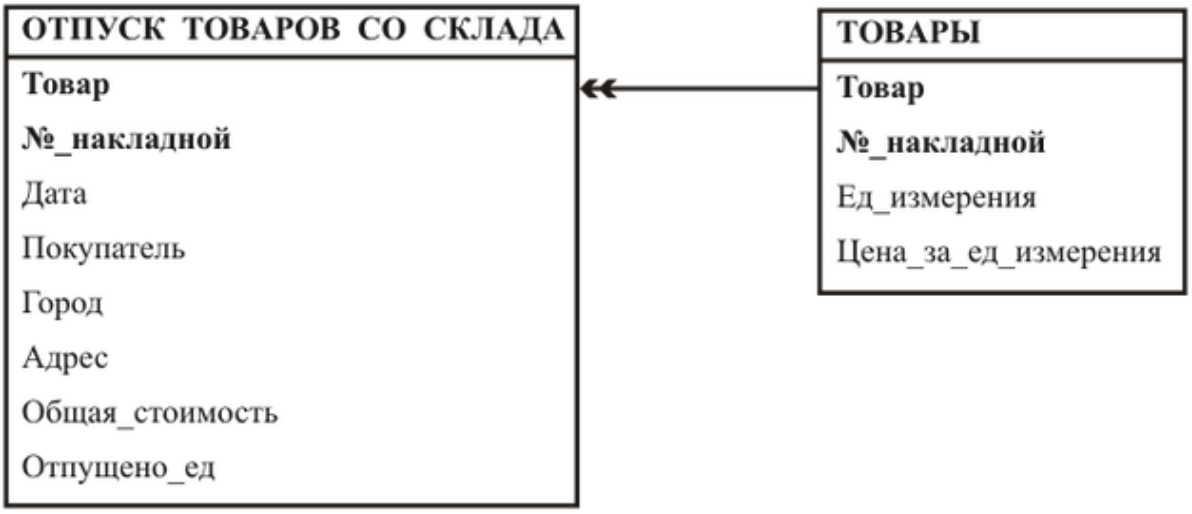


Рис. 1

Можно заметить, что теперь значение поля «Покупатель» никоим образом не зависит от пары значений «№ накладной», «Товар», а «зависит только от значения поля «№ накладной». Поэтому данное поле и зависящие от его значения поля «Город», «Адрес» выделяем в таблицу «Покупатели». Анализируя далее структуру таблицы «Отпуск товаров со склада», обнаруживаем, что одно из оставшихся полей — «Дата» — зависит только от значения поля «№ накладной». Поэтому выделяем дату и номер накладной в самостоятельную таблицу «Накладные» (рис. 2).

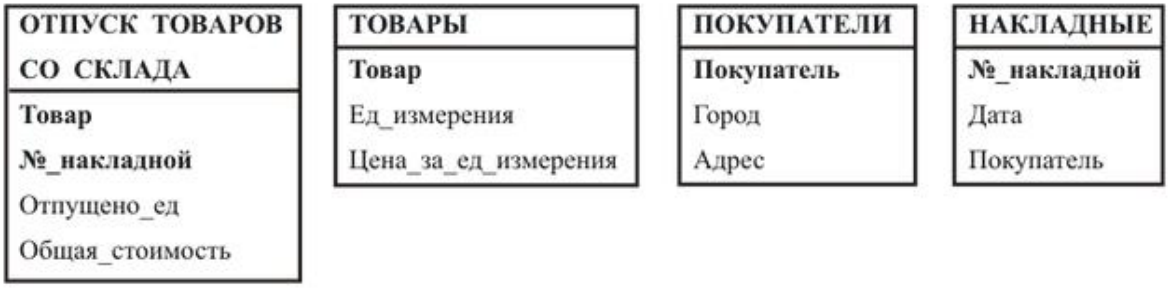


Рис. 2.

Установим связи между таблицами. Один покупатель может встречаться во многих накладных. Поэтому между таблицами «Покупатели» и «Накладные» имеется связь 1:М по полю «Покупатель». Одной накладной может соответствовать несколько товаров. Поэтому между таблицами «Накладные» и «Отпуск товаров со склада» имеется связь 1:М по полю «№ накладной» (рис. 3).

Обратите внимание, что в реляционной БД таблицы могут быть связаны только по одноименным полям, поэтому при выделении новых таблиц в них добавляют поля, по которым осуществляется связь с другими таблицами БД.

Рис. 3. БД во 2-й нормальной форме

Третья нормальная форма (ЗНФ) требует, чтобы в таблице не имелось транзитивных зависимостей между неключевыми полями, т.е. чтобы значение любого поля, не входящего в первичный ключ, не зависело от значения другого поля, также не входящего в первичный ключ.

В рассматриваемом примере можно увидеть, что в таблице «Отпуск товаров со склада» имеется зависимость значения поля «Общая стоимость» от значения поля количество т. е. «Отпущено_ед». По условию примера значение поля «Общая стоимость» может вычисляться как значение поля «Отпущено_ед», умноженное на значение поля «Цена за единицу измерения» из таблицы «Товары» (из записи с таким же значением поля «Товар»). Поэтому поле «Общая стоимость» из таблицы «Отпуск товаров со склада» удаляем.

Однако в случае, если «Цена за единицу измерения» зависит от количества отпущенного товара, что особенно распространено в наше рыночное время, сле-

дует создать отдельную таблицу, отображающую зависимость цены от количества отпущенного товара, после чего будет получена БД в 3НФ.

Замечание. В таблице «Покупатели» значение поля «Адрес» зависит от значения поля «Город», поскольку в разных городах могут оказаться улицы с одинаковыми названиями и, соответственно, дома с одинаковыми номерами (вспомним известный кинофильм «Ирония судьбы, или с легким паром»). Однако такой зависимостью можно пренебречь, поскольку поле «Адрес» в нашем случае носит чисто информационный характер и не должно входить в условия запросов самостоятельно. Вообще говоря, на практике не всегда возможно получить идеально нормализованную БД, да и не всегда это необходимо, т. е. бывает достаточно 2-й нормальной формы.

4. Перечень оборудования – персональный компьютер.

5. Порядок выполнения работы (Задания)

5.1. На основании приведенных данных в Таблице 7 необходимо получить БД в 1НФ.

Таблица 7

Ф.И.О	Должность	Оклад	Стаж	Надбавка	Кафедра	Дисциплина	Группа	Вид занятий
Волков И. С.	ассистент	1500	4	100	ГиМУ	СУБД	81	Практ.
						ИС	79	Практ.
Белкин А. М.	доцент	2000	8	300	ГиМУ	СУБД	81	Лекция
						ИТУ	81	Практ.
Зайцев С. С.	ассистент	1500	10	400	ГиМУ	ИС	79	Лекция
						ИТУ	81	Лекция
Медведев Е. В.	ассистент	1500	4	100	Э	Экономика	70	Лекция

5.2. Преобразовать схему БД в 1НФ ко 2НФ.

5.3. Преобразовать схему БД во 2НФ к 3НФ.

5.4. Результаты работы по всем этапам отобразите в отчете.

5.5. Приведение БД к более высоким нормальным формам.

6. Содержание отчета

6.1. Наименование работы

6.2. Цель работы

6.3. Перечень оборудования

6.4. Схема БД в 1НФ

6.5. Схема БД в 2НФ

6.6. Схема БД в 3НФ

6.7. Выводы
6.8. Ответы на контрольные вопросы

7. Контрольные вопросы

7.1. Объясните смысл терминов:
7.2. Нормализация.
- Избыточность данных.
- Аномалия обновления.
- Аномалия ввода.
- Атомарное значение.
- Нормальная форма Бойса-Кодда.

7.3. В каком случае БД находится в 1НФ?
7.4. В каком случае БД находится в 2НФ?
7.5. В каком случае БД находится в 3НФ?
7.6. Объясните, почему нежелательны таблицы, не подчиняющиеся второй или третьей нормальной форме.

8. Список литературы

8.1. Федорова Г.Н. Основы проектирования баз данных: Учебное пособие для СПО / Г.Н. Федорова. - М.: Академия, 2018.- 220с. (*Основное печатное издание ОПИ-1*)
8.2. Кумскова, И. А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.

ПРАКТИЧЕСКАЯ (ЛАБОРАТОРНАЯ) РАБОТА № 2
Преобразование реляционной БД в сущности и связи

1. Цель работы: изучение вопросов проектирования баз данных, принципов нормализации таблиц базы данных.

2. Время выполнения работы – 2 часа.

3. Краткие теоретические сведения

Одним из наиболее популярных средств формализованного представления предметной области является модель «сущность — связь», которая положена в основу значительного количества коммерческих CASE-продуктов, поддерживающих полный цикл разработки систем баз данных или отдельные его стадии.

Моделирование предметной области в этом случае базируется на использовании графических диаграмм, включающих сравнительно небольшое число компонентов и, самое важное, технологию построения таких диаграмм. Существует много версий ER-диаграмм, которые по-разному представляют связи, сущности и атрибуты и которые имеют различные ограничения и условия применимости.

Семантическую основу ER-модели составляют следующие предположения:

– та часть совокупности взаимосвязанных объектов реального мира, сведения о которых должны быть помещены в базу данных, может быть представлена как множество сущностей;

– каждая сущность обладает характеристическими свойствами (атрибутами), отличающими ее от других сущностей и позволяющими ее идентифицировать;

– сущности можно классифицировать по типам: каждый экземпляр сущности (представляющий некоторый объект) может быть отнесен к некоторому классу, или типу, сущностей, каждый экземпляр которого обладает общими для них свойствами и отличающим их от сущностей других классов;

– систематизация представления, основанная на классах, в общем случае предполагает иерархическую зависимость типов: сущность типа A является подтипом сущности B, если каждый экземпляр типа A является экземпляром сущности типа B;

– взаимосвязи объектов могут быть представлены как связи, или сущности, которые служат для фиксирования (представления) взаимозависимости двух или нескольких сущностей.

4. Перечень оборудования – персональный компьютер.

5. Порядок выполнения работы (Задания)

Задание № 1. Изучите процесс преобразования реляционной базы данных в сущности и связи:

ФИО	*Год рожд.*	*Должн.*	*Каф. №*
Иванов И.И.	1948	Зав.каф.	22
Сидоров С.С.	1953	Проф.	22
Гиацинтова Г.Г.	1945	Проф.	22
Цветкова С.С.	1960	Доцент	22
Козлов К.К.	1959	Доцент	22
Петров П.П.	1960	Ст.преп.	22
Лютикова Л.Л.	1977	Ассистент	22
Рыбин Р.Р.	1950	Зав.каф.	23
Китов К.К.	1944	Проф.	23
Раков В.В.	1958	Доцент	23
Соловьева С.С.	1958	Доцент	23
Воробьева В.В.	1959	Ст. преп.	23
Орлова О.О.	1966	Ассистент	23
Осетров С.С.	1976	Ассистент	23

Данные

1	Воробьева В.В.
2	Гиацинтова Г.Г.
3	Иванов И.И.
4	Китов К.К.
5	Козлов К.К.
6	Лютикова Л.Л.
7	Орлова О.О.
8	Осетров С.С.
9	Петров П.П.
10	Раков В.В.
11	Рыбин Р.Р.
12	Сидоров С.С.
13	Соловьева С.С.
14	Цветкова С.С.

1	1944
2	1945
3	1948
4	1950
5	1953
6	1958
7	1959
8	1960
9	1966
10	1976
11	1977

1	Ассистент
2	Доцент
3	Зав.каф.
4	Проф.
5	Ст. преп.

1	22
2	23

а

Связи

ФИО	Год рожд.	Должн.	Каф. №
3	3	3	1
12	5	4	1
2	2	4	1
14	8	2	1
5	7	2	1
9	8	2	1
6	11	1	1
11	4	3	2
4	1	4	2
10	6	2	2
13	6	2	2
1	7	5	2
7	9	1	2
8	10	1	2

Должн.	ФИО
1	6
1	7
1	8
2	14
2	5
2	10
2	13
3	3
3	11
4	12
4	2
4	4
5	9
5	1

Год рожд.	ФИО
1	4
2	2
3	3
4	11
5	12
6	10
6	13
7	5
7	1
8	14
8	9
9	7
10	8
11	6

Каф. №	ФИО
1	3
1	12
1	2
1	14
1	5
1	9
1	6
2	11
2	4
2	10
2	13
2	1
2	7

б

Задание № 2. Создать функциональную модель предметной области БД по варианту индивидуального задания.

Вариант 1. Деятельность пункта обмена валюты. В пункте обмена валют создана локальная информационная система, автоматизирующая процесс учета

сделок купли-продажи валюты. Информационная система обеспечивает ввод, хранение и поиск информации о сделках, совершенных в данном пункте обмена. Каждой сделке присваивается уникальный цифровой код.Информация о сделке содержит сведения о дате и времени сделки, суммах покупаемой и продаваемой валют, фамилии, имени, отчестве и номере паспорта клиента, а также о фамилии, инициалах и учетном номере личного дела кассира в отделе кадров. Система позволяет вычислять денежный оборот за один или несколько дней, а также осуществлять поиск информации о сделках по номеру паспорта клиента.

Вариант 2. Работа информационной системы коммерческого банка. Информационная система обеспечивает следующие виды работ: формирование уникального идентификационного номера клиента, счета клиента и кассира банка; формирование уникального номера ссуды клиенту в любом отделении банка (номер ссуды отличается от номера счета); формирование входных документов (приходный ордер, расходный ордер); формирование выходных документов (отчет управляющего отделением, отчет о состоянии ссуд по отделению, отчет кассира за текущий день); реализацию запросов (список клиентов, у которых остаток по счету превышает 100000 руб., в какие дни неде6ли сумма выданных денег превышает сумму принятых денег от клиентов).

Вариант 3. Работа информационной подсистемы деканата факультета университета. Информационная система обеспечивает формирование:

− входных документов (списки студентов по учебным группам и курсам, списки студентов,

находящихся в академическом отпуске, списки студентов обучающихся по индивидуальным планам, списки студентов участвующих в выполнении НИР);

− выходных документов (расписание занятий студентов по учебным группам на семестр, список студентов, слушающих заданный учебный курс, список учебных курсов, список студентов, не прошедших текущую аттестацию, списки отлично успевающих студентов, сведенья о трудоустройстве выпускников, сведенья о студентах, проживающих в общежитии университета и сведенья о студентах нуждающихся в общежитии).

Вариант 4. Деятельность переговорного пункта. Информационная система пункта обеспечивает:

− ввод данных об авансовом взносе клиента при предоставлении ему междугородних и международных переговоров;

− ввод данных о тарифах за услуги связи, с учетом особенностей заказа клиента (льготное время, международный звонок, IP-телефония, факс и пр.).

− формирование отчетов о продолжительности разговора клиента, о полной стоимости услуги, предоставленной клиенту, о количестве услуг, предоставленных всем клиентам за указанный период времени (день, неделю, месяц) с разбивкой по видам услуг (междугородние переговоры, международные переговоры, факс, доступ в Интернет и др.).

Вариант 5. Деятельность производственно-технического отдела фирмы. В производственно-техническом отделе гипотетической фирмы создана локальная информационная подсистема, автоматизирующая решение задач учета состояния и модернизации компьютерного парка и офисной техники.

Информационная подсистема обеспечивает:

− создание, корректировку и хранение данных о состоянии компьютерного парка и офисной техники с разбивкой по структурным подразделениям фирмы.

− создание, сохранение, корректировку и вывод на печать заявок на модернизацию компьютерного парка и офисной техники с разбивкой по структурным подразделениям фирмы.

− формирование отчетов о техническом состоянии и модернизации компьютерного парка и офисной техники фирмы за указанный период времени (месяц, квартал, полугодие и год).

Задание № 3. Создать концептуальную модель БД по варианту индивидуального задания.

Задание № 4. Определить первичные ключи реляционных таблиц. Задать внешние ключи для организации связей с соответствующими сущностями.

6. Содержание отчета

6.1. Наименование работы

6.2. Цель работы

6.3. Выполнение индивидуального задания.

7. Список литературы

7.1. Федорова Г.Н. Основы проектирования баз данных: Учебное пособие для СПО / Г.Н. Федорова. - М.: Академия, 2018.- 220с. (*Основное печатное издание ОПИ-1*)

7.2. Кумскова, И. А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.

ПРАКТИЧЕСКАЯ (ЛАБОРАТОРНАЯ) РАБОТА № 3.
Проектирование реляционной БД. Нормализация таблиц. Задание ключей. Создание основных объектов БД

1. Цель работы

Получить практический опыт проектирования базы данных с использованием CASE-средств.

2. Время выполнения работы – 2 часа.

3. Краткие теоретические сведения

Для анализа структуры данных и информационного моделирования существует множество программных продуктов. Один из них –Microsoft Office Visio, поддерживающий методологию IDEF1X (Integration DEFinition for Information Modeling). Он позволяет построить логическую модель данных, представляющую собой совокупность информационных объектов и связей между ними, а также физическую модель, непосредственно связанную с конкретной СУБД.

Схемы сущностей и отношений

С помощью шаблонов баз данных Visio можно смоделировать схему сущностей и отношений. Она состоит из следующих основных компонентов.

Сущность. Фигура сущности — это объект данных. В базе данных сущность обычно представлена таблицей. Каждая строка в таблице соответствует экземпляру сущности.

Атрибут. Фигура атрибута — это свойство сущности. Атрибуты могут определять экземпляр сущности (первичный ключ). Атрибуты также могут быть частью связанной таблицы, в которой общий атрибут используется для присоединения данных.

Отношение. Эти фигуры используются совместно для отображения структуры базы данных и отношений между таблицами. Диаграммы баз данных могут применяться для проектирования систем баз данных или демонстрации запроса к базе данных.

Создать новую или реконструировать существующую базу данных в модель можно с помощью шаблона "Схема модели базы данных" (рис.1), воспользоваться его набором элементов:

1. "Сущность-связь" - для баз на основе SQL92 и более ранних версий стандарта,
2. "Объектно-реляционная схема" - для баз на основе SQL99 и более поздних версий стандарта.

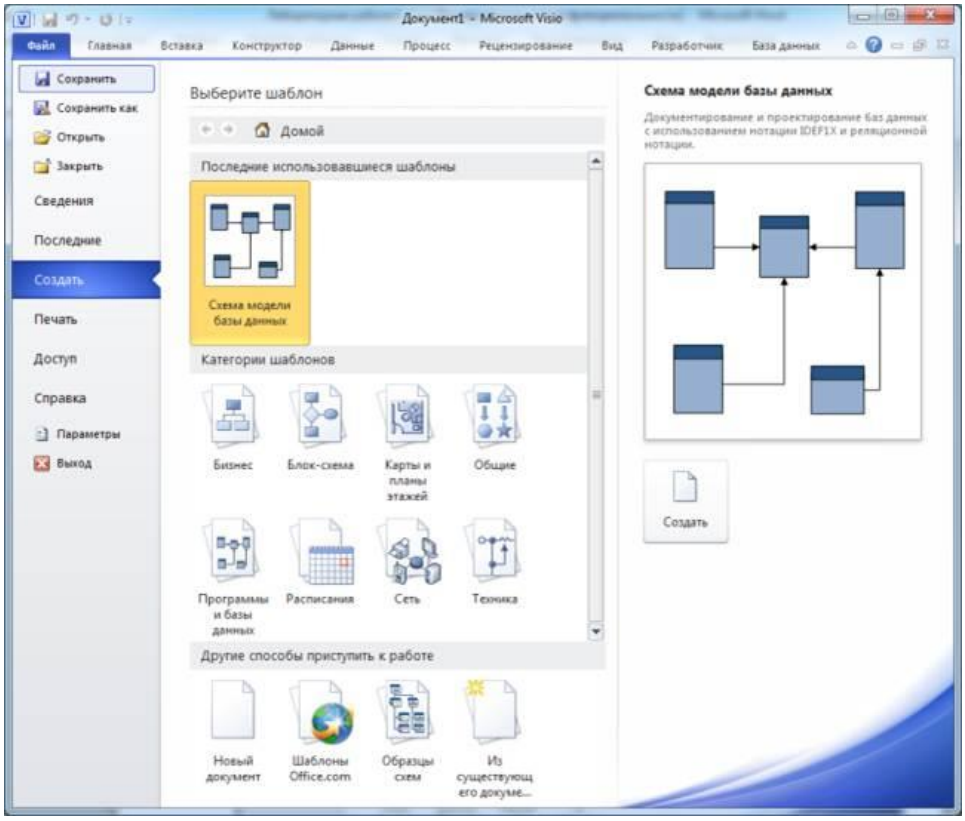


Рисунок 1 - Создание документов

Анализ предметной области

В качестве примера можно использовать список всех сущностей, информацию о которых, потребуется сохранять в базе данных, причем, для каждой сущности определен необходимый список атрибутов и список всех связей, которые потребуются, в формате, указанном в таблице 1.

Таблица 1

Номер	Сущность	Атрибуты
1	Меры	дата/время, адрес, базовая станция, попыток установки соединений, успешно установленных соединений, исполнитель
2	ДатаВремяИзмерения	дата/время измерения
3	Адрес	город, улица, дом
4	Станция	номер
5	Исполнитель	фамилия, имя, отчество, отдел, должность

Примечание 1: информацию о дате и времени всегда лучше выносить за пределы таблицы фактов (главной таблицы) по нескольким причинам: размерность числового ключа обычно меньше, чем размерность типа данных для даты/времени - экономится место в главной таблице; в этом случае удобно использовать дополнительные флаги, например, рабочие/нерабочие дни, сезонности и т.п.

Примечание 2: разбиение значения адреса на более мелкие удобно, если, например, потребуется поднять результаты измерений для определенного города

Номер	Первая	Вторая сущность	Тип связи (один-к-одному, один-

связи	сущность		ко-многим, многие-к-одному, многие-ко-многим)
1	Меры	ДатаВремяИзмерения	Многие-к-одному
2	Меры	Адрес	Многие-к-одному
3	Меры	Станция	Многие-к-одному
4	Меры	Исполнитель	Многие-к-одному

Создание новой схемы модели базы данных

Если существующая база данных, которую требуется использовать в качестве основы, отсутствует, можно начать с пустой модели базы данных и добавить собственные таблицы и связи.

Параметры документа базы данных

1. Откройте вкладку **Файл**.
2. Выберите команду **Создать** и пункт **Программное обеспечение и базы данных**, а затем дважды щелкните элемент **Схема модели базы данных**.
3. На вкладке **База данных** в группе **Управление** нажмите кнопку **Параметры отображения**.
4. В диалоговом окне **Параметры документа базы данных** выберите нужный набор символов и другие параметры таблицы и связи (рис.2), а затем нажмите кнопку **ОК**.

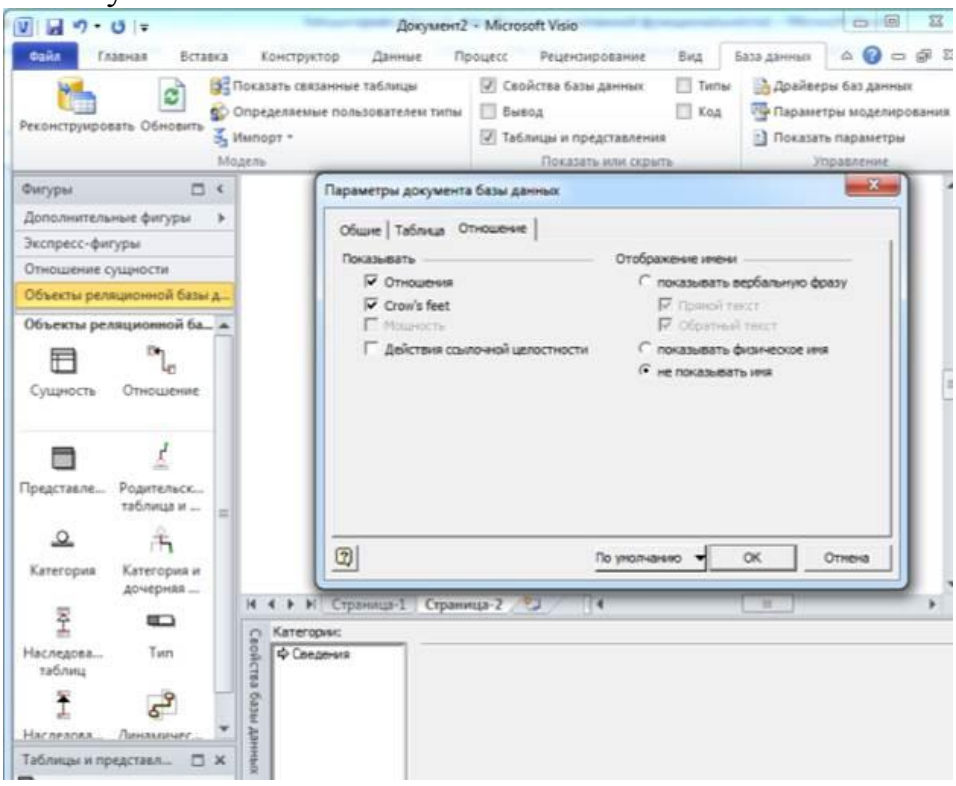


Рисунок 2 - Параметры документа базы данных

Создание таблиц

1. Установите удобный вам размер схемы (например, 100 %), а затем из контейнера **Объекты реляционной базы данных** перетащите на диаграмму элемент **Сущность**.
2. В списке **Категории** убедитесь, что выбрана строка **Определение** и введите следующие значения (табл.1):

Физическое имя– *Меры*,
Концептуальное имя– *Меры*,

3. Перейдите на строку **Столбцы**, убедитесь, что переключатель стоит в положении **Физический тип данных Microsoft Access**, введите имя и выберите тип данных, как показано на рисунке 3. Чтобы изменить тип данных для столбца, щелкните его поле **Тип данных**, а затем из списка выберите тип данных или введите тип данных в список. Поле **Заметки** заполнять не обязательно, значения в нем генерируются автоматически.
4. Установите флажок **Обязательное** для столбцов, которые не могут иметь значения NULL.
5. Установите флажок **PK** (первичный ключ) для столбцов, однозначно определяющих каждую строку таблицы базы данных.
6. В группе **Категории** выберите вариант **Индексы**, **Триггеры**, **Проверка** или **Дополнительные**, чтобы создать эти дополнительные элементы.
7. На вкладке **Конструктор** в группе **Темы** можно выбрать стиль оформления таблиц, например, **Цветов Яркая, эффект Простая тень** (рис.3)

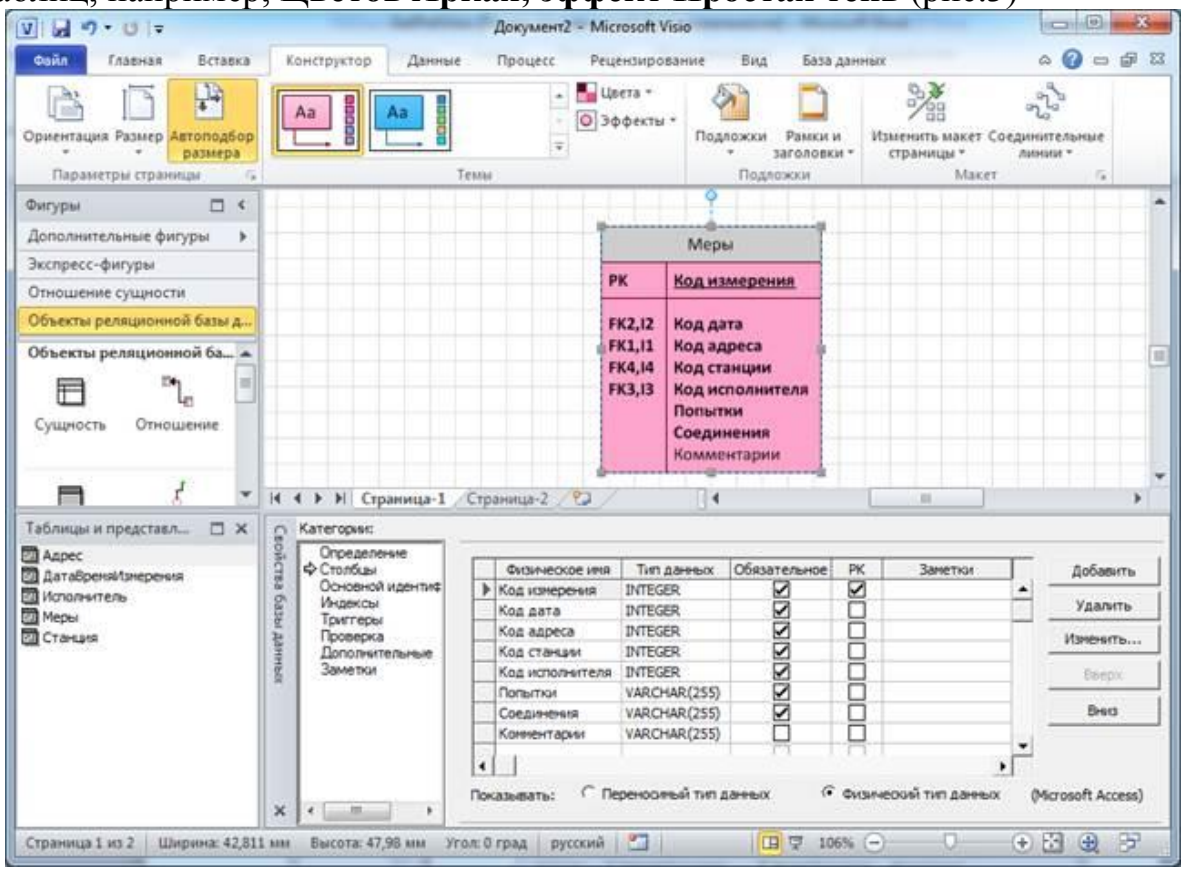
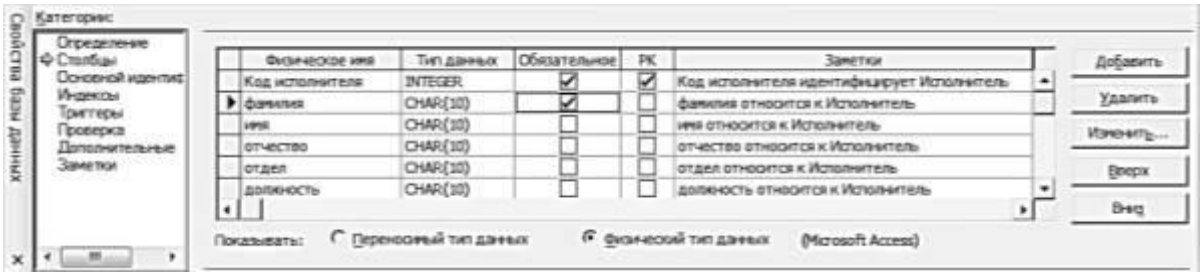


Рисунок 3 - Проектирование сущности Меры

Точно также создайте еще сущности **Адрес, Станция, Исполнитель и ДатаВремяИзмерения** со столбцами, как показано на рисунке 4.

а)

б)



в)

г)

Рисунок 4 - Проектирование сущностей: а) Адрес, б) Станция, в) Исполнитель, г) ДатаВремяИзмерения

Создание связей

После того, как создание всех объектов сущностей завершено, необходимо создать отношения между таблицами. Эта операция производится так:

1. Убедитесь, что в схеме отображены все таблицы.
2. Нажмите на кнопку **Соединительная линия** на вкладке **Главная** в группе **Сервис** (соседняя кнопка с указателем мыши не должна быть нажата).
3. Наведите указатель мыши на таблицу (родительскую) с первичным ключом (например, **ДатаВремяИзмерения**). Таблица будет выделена красным.
4. Перетащите таблицу (родительскую) с первичным ключом **ДатаВремяИзмерения** на таблицу (дочернюю) с внешним ключом **Меры**. Если все сделано правильно, то на схеме появится стрелка с красными квадратами на конце и вначале.
5. Если размерность связи не устраивает, то дважды щелкните связь. В окне **Свойства базы данных** в группе **Категории** выберите вариант **Прочее**. В группе **Мощность** выберите размерность, наиболее подходящую к связи. Для отношения «один-ко-многим» наилучшим вариантом будет **0 или более** или **1 или более** (Для отношения «один-к-одному» — **0 или 1** или **ровно 1).**

Создание связей в первый раз получается не всегда. Обратите внимание, что в Visio для столбцов, между которыми создаются отношения, должен совпадать тип данных и название (с учетом регистра). Иначе будет создано дополнительное поле с внешним ключом **FK 5 Код даты** (рис.5), а поле **FK 2 Код дата** останется

без соединения. В СУБД (например, Access) такое условие не является обязательным. В итоге схема может выглядеть так, как показано на рисунке 5.

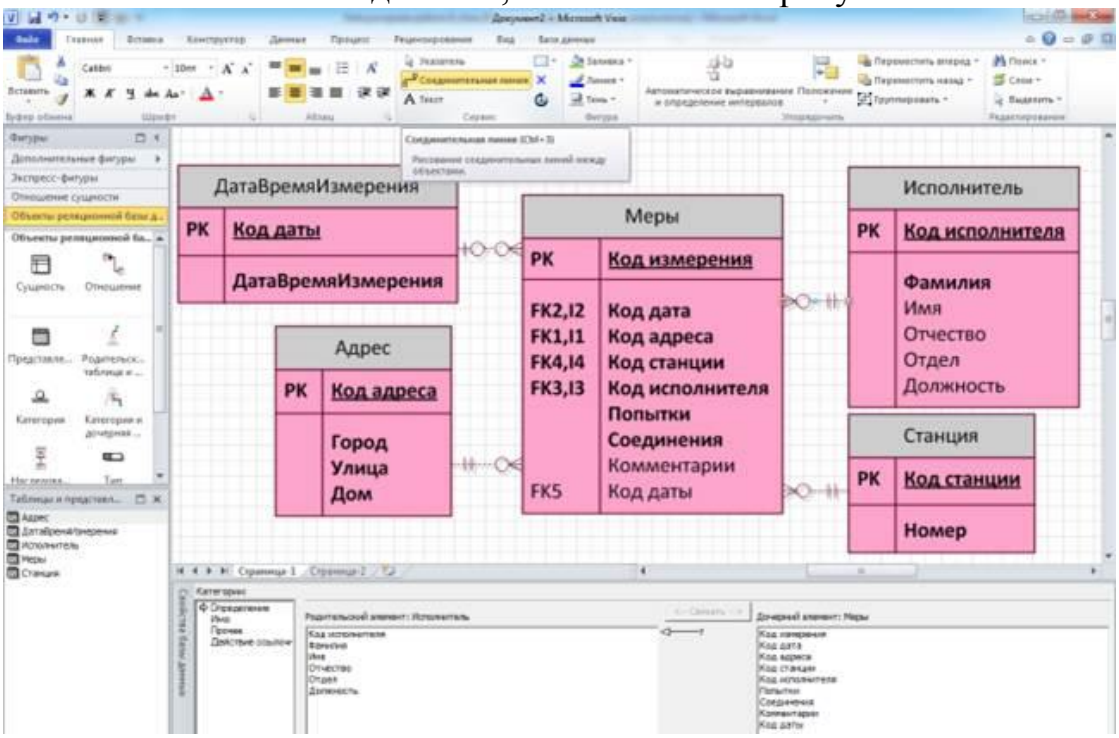


Рисунок 5- Схема Отношения между таблицами в Visio

Создание индексов

Использование индексов повышает производительность или скорость работы базы данных при выполнении запроса.

1. Дважды щелкните таблицу (**Меры)**, к которой требуется добавить индекс, а затем в окне **Свойства базы данных** в списке **Категории** выберите вариант **Основной идентификатор** и включите флажок **Создать индекс**.
2. Затем в окне **Свойства базы данных** в списке **Категории** выберите вариант **Индексы** и нажмите кнопку **Создать.**
3. В диалоговом окне **Создать индекс** введите имя для индекса (**Меры**), а затем нажмите кнопку **ОК**.
4. В списке **Тип индекса** выберите тип создаваемого индекса — **Только неуникальный**.
5. В списке **Доступные столбцы** выберите имя столбца **Код даты**, который требуется включить в этот индекс, а затем нажмите кнопку **Добавить .**
6. В списке **Индексированные столбцы** снимите флажок **Убывание**, чтобы создать индекс с возрастающим порядком сортировки (рис.6).

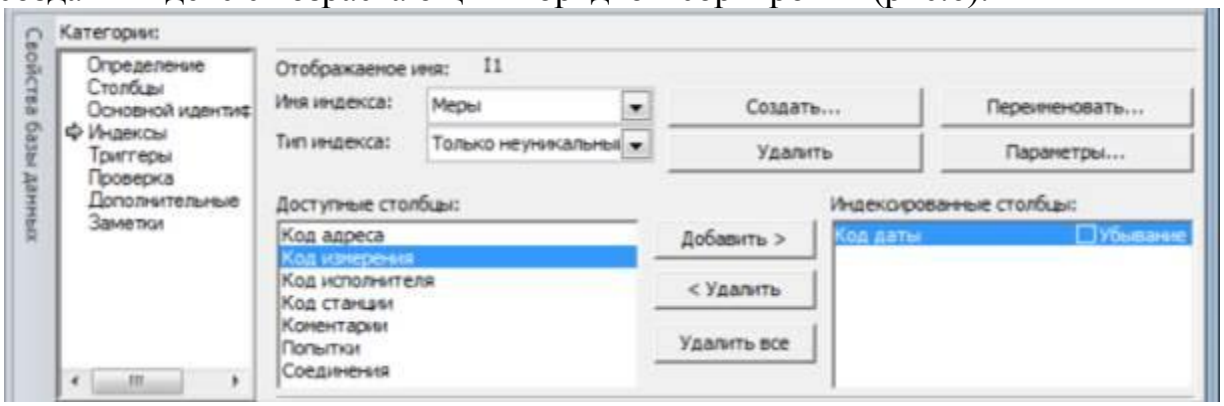


Рисунок 6 - Создание индексов

Создание проверки

Использование проверки позволяет контролировать работу базы данных при выполнении запроса.

Применение предложений проверки гарантирует, что данные, введенные в столбец, находятся в пределах конкретного диапазона значений. Например, можно создать предложение проверки того, что значения столбца «Возраст» превышают 65.

1. Дважды щелкните таблицу, чтобы открыть окно **Свойства базы данных**.
2. В группе **Категории** выберите вариант **Столбцы**, а затем выберите столбец **Соединение**, которому требуется добавить предложение проверки.
3. Нажмите кнопку **Изменить**.
4. На вкладке **Проверка** диалогового окна **Свойства столбца** введите ограничения, указанные на рисунке 7 .

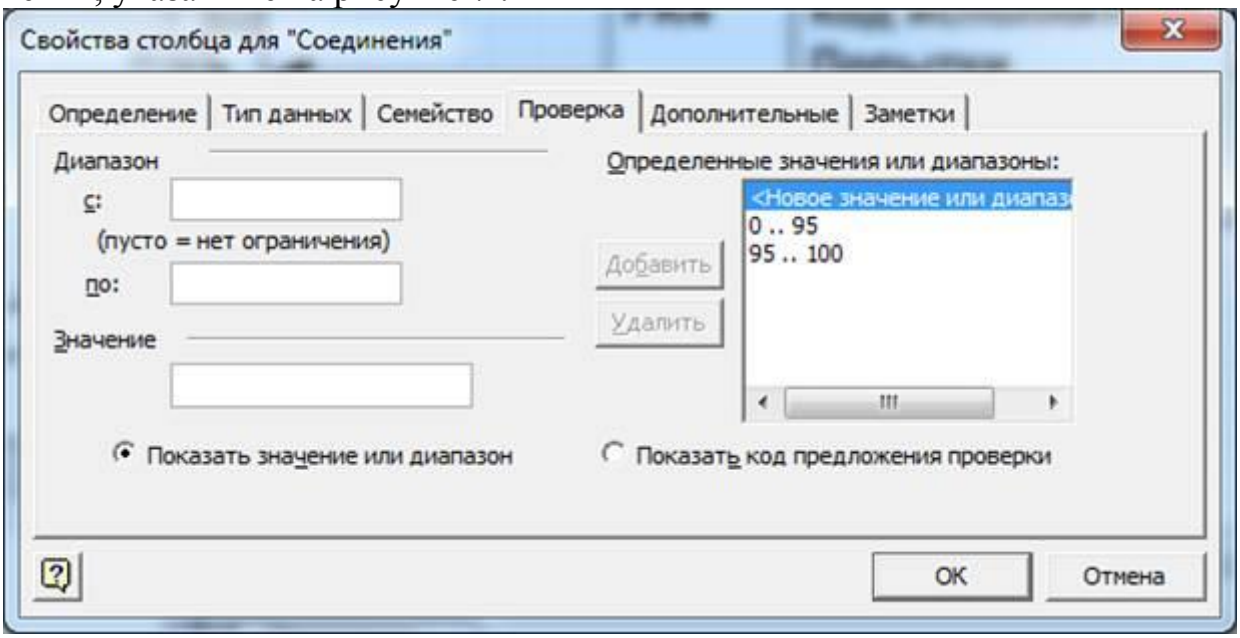


Рисунок 7 - Создание диапазонов проверки

Реконструирование схемы модели базы данных

Если существует база данных, которую нужно смоделировать для лучшего понимания ее структуры или применения в качестве основы создания новой модели, то для извлечения схемы или структуры базы данных можно воспользоваться мастером реконструирования. В качестве примера можно использовать базу данных созданную в Access 2010, по исходным данным, приведенным в таблице 1 (рис.8).

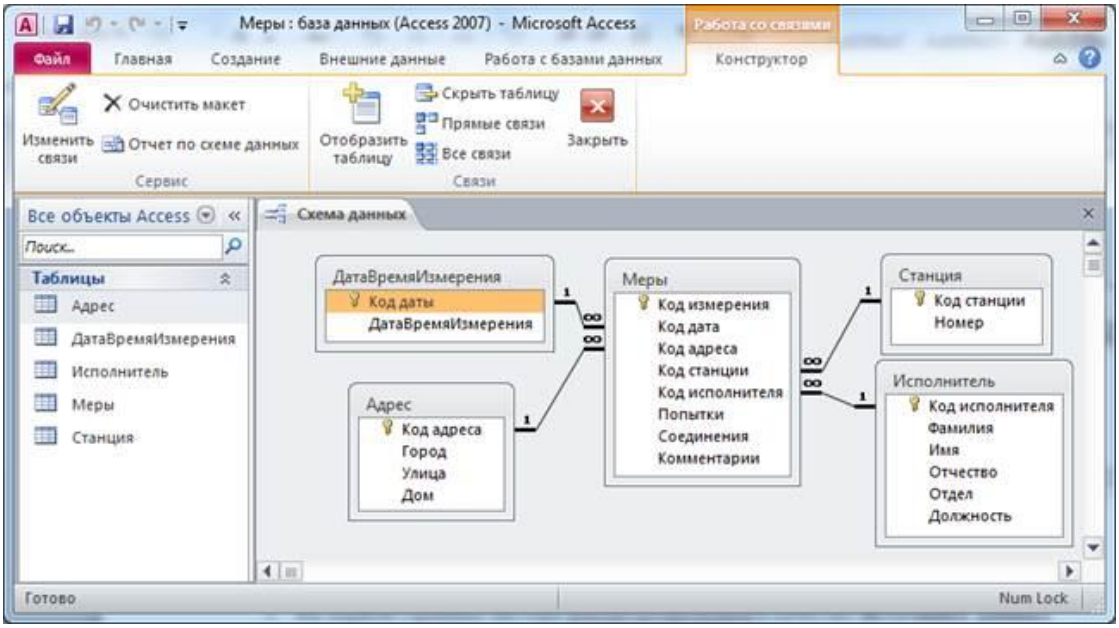


Рисунок 8 - Схема Отношения между таблицами в Access

1. Выберите команду **Создать** и пункт **Программное обеспечение и базы данных**, а затем дважды щелкните элемент **Схема модели базы данных**.
2. На вкладке **База данных** в группе **Модель** нажмите кнопку **Реконструирование**.
3. На первой странице мастера реконструирования в качестве **Источника данных** укажите **Microsoft Access Database** и нажмите кнопку **Далее**.
4. Укажите свой **Логин**, **Пароль** и нажмите **ОК**.
5. Укажите **Имя базы данных** (**Меры. accdb**) и нажмите **ОК**.
6. Типы объектов для реконструирования оставьте по умолчанию и нажмите кнопку **Далее**.
7. Выберите все таблицы и представления для реконструирования, нажав кнопку **Выделить все** и нажмите кнопку **Далее**.
8. Выберите параметр **Да, добавить фигуры на текущую страницу** и нажмите кнопку **Далее**, затем **Да** и **Готово**.

В результате будут извлечены выбранные сведения, и в окне «Вывод» будут отображены примечания о процессе извлечения, т.е. должна получиться схема модели базы данных аналогичная схеме, приведенной на рисунке 9.

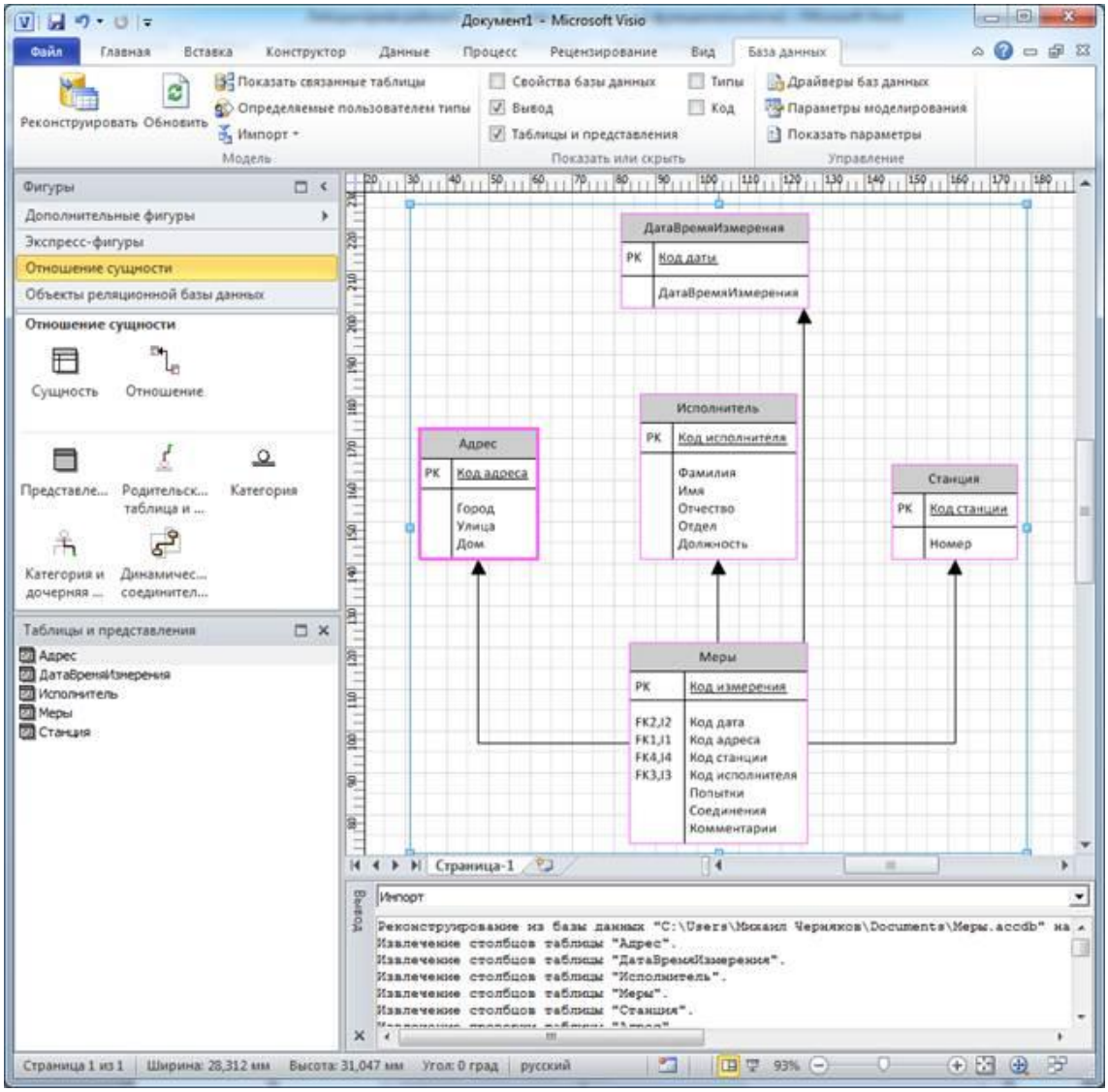


Рисунок 9 - Схема Отношения между таблицами реконструированная из Access в Visio

Импорт и уточнение существующей модели

Существующие модели VisioModeler или PLATINUM ERwin можно импортировать для создания схемы модели базы данных Visio. Эта возможность применяется только для файлов моделей VisioModeler версии 2.0 или более поздних (IMD) и для файлов моделей PLATINUM ERwin версий 2.6, 3.0 и 3.52 (ERX).

1. Откройте вкладку **Файл**.
2. Выберите команду **Создать** и пункт **Программное обеспечение и базы данных**, а затем дважды щелкните элемент **Схема модели базы данных**.
3. На вкладке **База данных** в группе **Модель** нажмите кнопку **Импорт** и выберите тип модели **ERX -файл Erwin** (рис.10).
4. Введите путь и имя файла для модели, которую требуется импортировать, или нажмите кнопку **Обзор**, чтобы найти файл модели, а затем нажмите кнопку **Открыть.**
5. В диалоговом окне **Импорт** нажмите кнопку **ОК**.

Выполняется импорт файла, ход которого отображается в окне «Вывод». Импортированные таблицы будут отображены в окне «Таблицы и представления».

6. В окне **Таблицы и представления** выберите таблицы для моделирования, а затем перетащите их на страницу документа (рис.10).

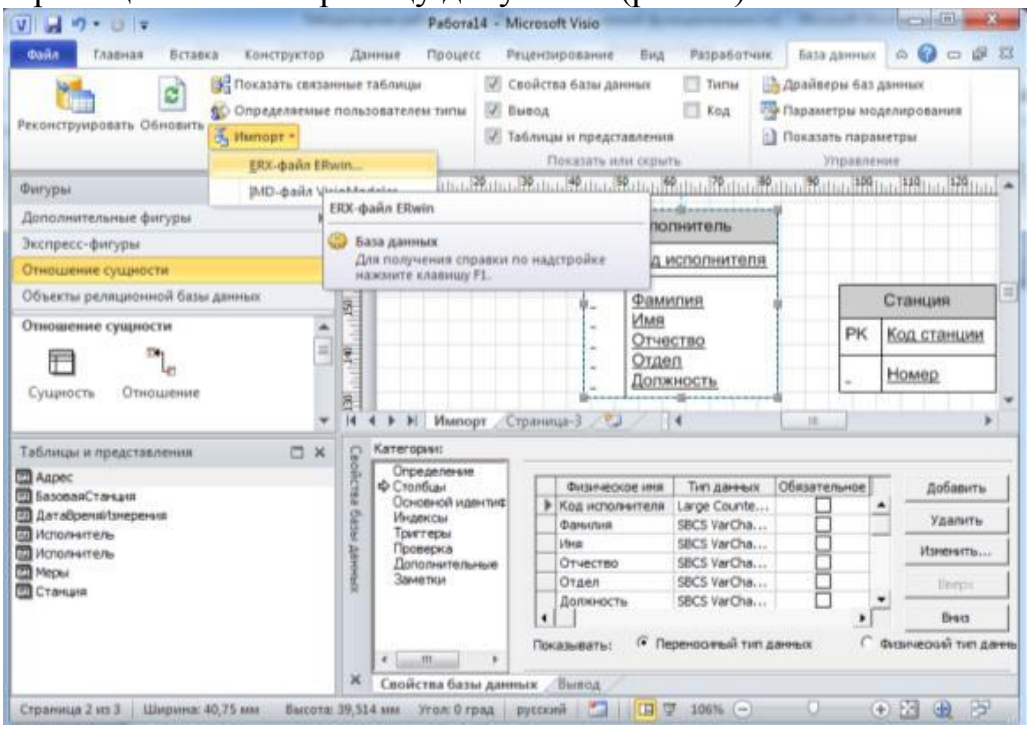


Рисунок 10 - Импорт моделей в Visio

4. Перечень оборудования – персональный компьютер.

5. Порядок выполнения работы (Задания)

5.1. В соответствии с вариантом задания выполнить проектирование базы данных.

6. Содержание отчета

1. Наименование работы.
2. Цель работы.
3. Выполнение индивидуального задания.

Варианты заданий

Вариант 1

Организация учета успеваемости в колледже

Вариант 2

Учет и выдача книг в библиотеке колледжа

Вариант 3

Отдел кадров компании

Вариант 4

Отдел поставок некоторого предприятия

Вариант 5

Пункт проката видеозаписей

Вариант 6

Стоматологическая клиника

Вариант 7

Кинотеатры (информация для зрителей)

Вариант 8

Ресторан (информация для посетителей).

Вариант 9

Информационная поддержка деятельности склада.

Вариант 10

Информационная поддержка деятельности адвокатской конторы.

Вариант 11

Информационная поддержка деятельности гостиницы.

Вариант 12

Информационное обеспечение фирмы по аренде автомобилей

Вариант 13

Информационная поддержка деятельности фитнес-клуба.

Вариант 14

Информационная поддержка деятельности аптечного склада.

Вариант 15

Электронный журнал посещаемости

7. Список литературы

7.1. Федорова Г.Н. Основы проектирования баз данных: Учебное пособие для СПО / Г.Н. Федорова. - М.: Академия, 2018.- 220с. (*Основное печатное издание ОПИ-1*)

7.2. Кумскова, И. А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.

ПРАКТИЧЕСКАЯ (ЛАБОРАТОРНАЯ) РАБОТА № 4.
Создание проекта БД. Создание БД. Редактирование и модификация таблиц. Создание ключевых полей. Задание индексов

1. **Цель работы:** получение навыков создания баз данных, создания и заполнения таблиц.
2. **Время выполнения работы** – 2 часа.
3. **Краткие теоретические сведения**

Одно из важнейших достоинств реляционных баз данных состоит в том, что вы можете хранить логически сгруппированные данные в разных таблицах и задавать связи между ними, объединяя их в единую базу. Для задания связи таблицы должны иметь поля с одинаковыми именами или хотя бы с одинаковыми форматами данных. Связь между таблицами устанавливает отношения между совпадающими значениями в этих полях. Такая организация данных позволяет уменьшить избыточность хранимых данных, упрощает их ввод и организацию запросов и отчетов. Поясним это на примере. Допустим, что в базе надо хранить данные о студентах (фамилия, изучаемая дисциплина) и преподавателях (фамилия, номер кафедры, ученая степень, преподаваемая дисциплина). Если хранить данные в одной таблице, то в строке с фамилией каждого студента, изучающего конкретную дисциплину, будут храниться все атрибуты преподавателя, читающего эту дисциплину. Это же огромная избыточность данных. А если хранить данные о студенте в одной таблице, о преподавателе - в другой и установить связь между полями «читаемая дисциплина -«изучаемая дисциплина» (фактически это одинаковые поля), то избыточность хранимых данных многократно уменьшится без ущерба для логической организации информации.

В Access можно задать три вида связей между таблицами: «один-ко-многим», «многие-ко-многим» и «один-к-одному».

Связь «один-ко-многим» является наиболее часто используемым типом связи между таблицами. В такой связи каждой записи в таблице А могут соответствовать несколько записей в таблице В (эти записи называют внешними ключами), а запись в таблице В не может иметь более одной соответствующей ей записи в таблице А.

При связи «многие-ко-многим» одной записи в таблице А могут соответствовать несколько записей в таблице В, а одной записи в таблице В - несколько записей в таблице А. Такая схема реализуется только с помощью третьей (связующей) таблицы, ключ которой состоит из, по крайней мере, двух полей, одно из которых является общим с таблицей А, а другое - общим с таблицей В.

При связи «один-к-одному» запись в таблице А может иметь не более одной связанной записи в таблице В и наоборот. Этот тип связи используют не очень часто, поскольку такие данные могут быть помещены в одну таблицу. Связь с отношением «один-к-одному» используют для разделения очень широких таблиц, для отделения части таблицы по соображениям защиты, а также для сохранения сведений, относящихся к подмножеству записей в главной таблице.

Целостность данных

Целостность данных означает систему правил, используемых в Access для поддержания связей между записями в связанных таблицах, а также обеспечивает

защиту от случайного удаления или изменения связанных данных. Установить целостность данных можно, если выполнены следующие условия:

связанное поле главной таблицы является ключевым полем или имеет уникальный индекс;

связанные поля имеют один тип данных (здесь существует исключение - поле счетчика может быть связано с числовым полем), если в последнем в свойстве Размер поля указано значение «Длинное целое»;

обе таблицы принадлежат одной базе данных Access, если таблицы являются связанными, то они должны быть таблицами Access; для установки целостности данных база данных, в которой находятся таблицы, должна быть открыта; для связанных таблиц из баз данных других форматов установить целостность данных невозможно.

Постановка задачи

Требуется разработать базу данных ДЕКАНАТ, содержащую четыре таблицы: СТУДЕНТЫ, ПРЕПОДАВАТЕЛИ, ДИСЦИПЛИНЫ, ОЦЕНКИ.

Создание структуры базы данных и установление связей между таблицами

1. Создайте базу данных **ДЕКАНАТ**, выполнив следующие действия:

загрузите Access, в появившемся окне выберите пункт НОВАЯ БАЗА ДАННЫХ, затем нажмите кнопку ОК;

в окне ФАЙЛ НОВОЙ БАЗЫ ДАННЫХ задайте имя файла и нажмите кнопку СОЗДАТЬ.

2. Создайте структуру таблицы **ПРЕПОДАВАТЕЛИ**. Для этого:

в окне базы данных выберите вкладку ТАБЛИЦЫ, а затем нажмите кнопку СОЗДАТЬ;

в окне НОВАЯ ТАБЛИЦА выберите пункт КОНСТРУКТОР и нажмите кнопку ОК;

в результате проделанных операций открывается окно таблицы в режиме конструктора, в котором следует определить поля таблицы.

Таблица ПРЕПОДАВАТЕЛИ должна содержать следующие поля:

Код преподавателя;

Фамилия;

Имя;

Отчество;

Дата рождения;

Должность;

Стаж;

Телефон.

При определении поля ДАТА РОЖДЕНИЯ используем маску для удобного ввода даты (т. е. в датах точки будут вводится автоматически). Для этого в Свойства полей на вкладке Общие установите курсор на поле маска, справа появится кнопка с тремя точками - нажмите на нее. В появившемся окне создания масок выбирайте КРАТКИЙ ФОРМАТ ДАТЫ.

В поле ДОЛЖНОСТЬ используем мастер подстановок для того, чтобы не вводить, а выбирать из списка нужную должность с использованием ввода должности, которой нет в списке. В режиме СОЗДАНИЯ ПОДСТАНОВОК выбираем

ФИКСИРОВАННЫЙ НАБОР ЗНАЧЕНИЙ, далее создаем 1-й столбец с должностями:

профессор;

доцент;

старший преподаватель;

ассистент.

Закончив создание списка в режиме конструктора на вкладке ПОДСТАНОВКА, посмотрите появившиеся изменения после работы мастера. Проверьте строку ОГРАНИЧИТЬСЯ СПИСКОМ, в котором должно стоять слово НЕТ.

В поле СТАЖ в общих свойствах поля установите УСЛОВИЕ НА ЗНАЧЕНИЕ > 0, СООБЩЕНИЕ ОБ ОШИБКЕ введите - стаж должен быть больше 0.

В поле ТЕЛЕФОН наберите маску для ввода 999-99-99, которая позволит не набирать тире в номере телефона при вводе в поле.

В качестве ключевого задайте поле КОД ПРЕПОДАВАТЕЛЯ.

Закройте таблицу ПРЕПОДАВАТЕЛИ в режиме конструктора.

Остальная часть этой таблицы также будет заполняться в режиме формы.

3. Таблица **СТУДЕНТЫ** должна содержать следующие поля:

Код студента;

Фамилия;

Имя;

Отчество;

Номер группы;

Адрес;

Телефон;

Дата рождения;

Медалист.

В поле МЕДАЛИСТ создайте ПОЛЕ СО СПИСКОМ без ввода новых значений, а также задайте ЗНАЧЕНИЕ ПО УМОЛЧАНИЮ "Нет" (кавычки обязательны).

4. Создайте структуру таблицы **ДИСЦИПЛИНЫ** аналогично п.1

Таблица ДИСЦИПЛИНЫ должна содержать следующие поля:

Код дисциплины;

Название дисциплины;

Код преподавателя;

Номер семестра;

Экзамен.

В качестве ключевого задайте поле КОД ДИСЦИПЛИНЫ.

Поле КОД ПРЕПОДАВАТЕЛЯ будет заполняться при помощи мастера подстановок из таблицы ПРЕПОДАВАТЕЛИ. Из доступных полей таблицы ПРЕПОДАВАТЕЛИ выберите: КОД ПРЕПОДАВАТЕЛЯ, ФАМИЛИЯ, ИМЯ, ОТЧЕСТВО, скройте ключевое поле. После работы мастера при заполнении поля КОД ПРЕПОДАВАТЕЛЯ таблицы будут отображаться ФИО преподавателя для выбора, но в таблице ДИСЦИПЛИНЫ поле КОД ПРЕПОДАВАТЕЛЯ будет оставаться числовым целым.

Поле ЭКЗАМЕН заполняйте при помощи поля со списком двух значений Экзамен или Зачет.

Закройте таблицу ДИСЦИПЛИНЫ.

Оставшаяся часть таблицы будет заполняться в режиме формы.

5. **Таблица ОЦЕНКИ** должна содержать следующие поля:

Код студента;

Код дисциплины;

Номер семестра;

Оценка.

Ключ будет составной: КОД СТУДЕНТА, КОД ДИСЦИПЛИНЫ, НОМЕР СЕМЕСТРА (в режиме конструктора выделите три поля и задайте ключ).

Разработайте схему данных, т.е. создайте связи между таблицами. Для этого:

Выполните команду СЕРВИС=> СХЕМА ДАННЫХ. На экране появится окно СХЕМА ДАННЫХ.

Выполните команду СВЯЗИ=>ДОБАВИТЬ ТАБЛИЦУ.

В появившемся окне будет выделено название одной таблицы. Нажмите кнопку ДОБАВИТЬ.

Переведите выделение на имя следующей таблицы и нажмите кнопку ДОБАВИТЬ. Аналогично добавьте оставшиеся две таблицы.

Закройте окно нажав кнопку ЗАКРЫТЬ.

Создайте связь между таблицами ДИСЦИПЛИНЫ и ОЦЕНКИ. Для этого подведите курсор мыши к полю КОД ДИСЦИПЛИНЫ в таблице ДИСЦИПЛИНЫ, нажмите левую клавишу мыши и, не отпуская ее, перетащите курсор на поле КОД ДИСЦИПЛИНЫ в таблице ОЦЕНКИ, а затем отпустите левую клавишу мыши. На экране откроется окно СВЯЗИ.

Щелкните по ячейке ОБЕСПЕЧЕНИЕ ЦЕЛОСТНОСТИ ДАННЫХ -в ней должна появиться галочка.

Щелкните по ячейкам КАСКАДНОЕ ОБНОВЛЕНИЕ СВЯЗАННЫХ ПОЛЕЙ и КАСКАДНОЕ УДАЛЕНИЕ СВЯЗАННЫХ ЗАПИСЕЙ.

Информация. Задание каскадного обновления связанных полей и каскадного удаления связанных записей позволит редактировать записи только в таблице ДИСЦИПЛИНЫ, а в таблице ОЦЕНКИ эти действия будут со связанными записями выполняться автоматически. Например, если вы удалите из таблицы ДИСЦИПЛИНЫ один предмет, то в таблице ОЦЕНКИ удалятся все строки, связанные с этим предметом.

Нажмите кнопку СОЗДАТЬ. Связь будет создана.

Аналогично создайте связи между полем КОД ПРЕПОДАВАТЕЛЯ в таблице ПРЕПОДАВАТЕЛИ и полем КОД ПРЕПОДАВАТЕЛЯ в таблице ДИСЦИПЛИНЫ, а также между полем КОД СТУДЕНТА в таблице СТУДЕНТЫ и полем КОД СТУДЕНТА в таблице ОЦЕНКИ.

Закройте окно схемы данных, ответив ДА на вопрос о сохранении макета.

Заполните поля таблиц для применения в других практических работах.

4. Перечень оборудования – 2 часа.

5. Порядок выполнения работы (Задания)

5.1. В соответствии с вариантом задания выполнить операции по созданию базы данных.

6. Содержание отчета

1. Наименование работы
2. Цель работы
3. Выполненные задания в соответствии с вариантом.

7. Список литературы

7.1. Федорова Г.Н. Основы проектирования баз данных: Учебное пособие для СПО / Г.Н. Федорова. - М.: Академия, 2018.- 220с. (*Основное печатное издание ОПИ-1*)

7.2. Кумскова, И. А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.

ПРАКТИЧЕСКАЯ (ЛАБОРАТОРНАЯ) РАБОТА № 5.
Редактирование, добавление и удаление записей в таблице. Применение логических условий к записям. Открытие, редактирование и пополнение табличного файла

1. Цель работы

Получить практический опыт организации запросов на редактирование, добавление и удаление записей в таблице базы данных.

2. Время выполнения работы

3. Краткие теоретические сведения

Объекты «Таблица» и «Запрос» в Microsoft Access являются основополагающими, т.к. часто запросы могут использоваться вместо таблиц. Например, форма или отчет могут основываться как на таблице, так и на запросе. Таблицы и запросы содержат множество полей и записей. Оба эти объекта могут предоставлять исходную информацию, необходимую для формы или отчета.

Типы запросов, создаваемых в Microsoft Access

1 **Запросы на выборку.**

Наиболее часто используемым запросом является запрос на выборку. Запрос на выборку возвращает данные из одной или нескольких таблиц, а также результаты, которые при желании пользователь может изменить (с некоторыми ограничениями). Также можно использовать запрос на выборку, чтобы сгруппировать записи для вычисления сумм, средних значений, пересчета и других действий

2 **Запросы с параметрами.**

Это запросы, при выполнении которых в диалоговом окне пользователю выдается приглашение ввести данные, например условие для возвращения записей или значение, которое должно содержаться в поле. Можно создать запрос, в результате которого выводится приглашение на ввод нескольких данных, например, двух дат. В результате будут возвращены все записи, находящиеся между указанными двумя датами. Также запросы с параметрами удобно использовать в качестве основы для форм и отчетов. Например, на основе запроса с параметрами можно создать месячный отчет о доходах. При выводе данного отчета, на экране появится приглашение ввести месяц, доходы которого интересуют пользователя. После ввода месяца на экране будет представлен требуемый отчет. Для получения дополнительных сведений о запросах с параметрами нажмите кнопку. Можно создать специальную форму или диалоговое окно, которое вместо диалогового окна запроса с параметрами будет выводить приглашение на ввод параметров запроса.

3 **Перекрестные запросы**

В перекрестном запросе отображаются результаты статистических расчетов (такие как суммы, количество записей и средние значения), выполненных по данным из одного поля таблицы. Эти результаты группируются по двум наборам данных, один из которых расположен в левом столбце таблицы, а второй - в верхней строке. Существует возможность вывести данные в перекрестной таблице без создания в базе данных отдельного запроса. Для этого следует использовать мастер сводных таблиц. В сводной таблице пользователь имеет возможность изме-

нять заголовки строк или столбцов, что позволяет анализировать данные различными способами.

4 **Запросы на изменение (запросы на создание таблицы, удаление, обновление, добавление записей)**

Запрос на изменение - это запрос, который за одну операцию вносит изменения в несколько записей. Существует четыре типа запросов на изменение: на удаление, на обновление и добавление записей, а также на создание таблицы.

- **Запрос на удаление.**

Удаляет группу записей из одной или нескольких таблиц. Например, запрос на удаление позволяет удалить записи о товарах, поставки которых прекращены или на которые нет заказов. С помощью запроса на удаление можно удалять только всю запись, а не отдельные поля внутри нее.

- **Запрос на обновление записей**

Вносит общие изменения в группу записей одной или нескольких таблиц.

Например, на 10 процентов поднимаются цены на все молочные продукты или на 5 процентов увеличивается зарплата сотрудников определенной категории. Запрос на обновление записей позволяет изменять данные в существующих таблицах.

- **Запрос на добавление.**

Добавляет группу записей из одной или нескольких таблиц в конец одной или нескольких таблиц. Например, появилось несколько новых клиентов, а также база данных, содержащая сведения о них. Чтобы не вводить все данные вручную, их можно добавить в таблицу «Клиенты». Запрос на добавление также полезен при выполнении следующих действий.

a) Добавление полей на основе условий отбора. Например, необходимо добавить имена и адреса клиентов с очень крупными заказами.

b) Добавление записей, если некоторые поля из одной таблицы не существуют в другой. Например, таблица «Клиенты» содержит 11 полей. Пусть требуется добавить записи из другой таблицы с полями, соответствующими 9 из 11 полям таблицы «Клиенты». Запрос на добавление добавит данные в совпадающие поля и пропустит остальные.

- **Запрос на создание таблицы**

Создает новую таблицу на основе всех или части данных из одной или нескольких таблиц. Запрос на создание таблицы полезен для выполнения следующих действий.

a) Создание таблицы для экспорта в другую базу данных Microsoft Access.

Например, требуется создать таблицу, содержащую несколько полей из таблицы «Сотрудники», а затем экспортировать эту таблицу в базу данных, используемую отделом кадров.

b) Создание отчетов, содержащих данные, соответствующие определенному моменту времени. Например, 15 мая 2011 года необходимо напечатать отчет об объеме продаж, сделанных в первом квартале, основанный на данных, содержащихся в базовой таблице на 9:00 A.M. 1 апреля 2011. Отчет, основанный на запросе или инструкции SQL, выбирает из таблиц самые последние данные (данные на 15 мая 2011), а не записи на указанный момент времени. Чтобы получить дан-

ные на 9:00 A.M. 1 апреля 2011, необходимо разработать запрос на создание таблицы, в котором требуемые записи отбираются в зависимости от указанного момента времени и помещаются в новую таблицу. Затем в качестве основы для отчета следует использовать эту таблицу, а не запрос.

c) Создание резервной копии таблицы.

d) Создание архивной таблицы, содержащей старые записи. Например, можно создать таблицу, сохраняющую все старые заказы, прежде чем удалить их из текущей таблицы «Заказы».

e) Повышение быстродействия форм и отчетов, базирующихся на многотабличных запросах или инструкциях SQL. Например, требуется вывести на печать несколько отчетов, базирующихся на запросе, включающем пять таблиц, в котором рассчитываются общие итоги. Чтобы ускорить процесс, разработайте запрос на создание таблицы, извлекающий необходимые записи и сохраняющий их в одной таблице. Затем на базе этой таблицы создайте отчет или укажите ее в инструкции SQL как источник записей для формы или отчета. Это позволит обойтись без повторных запусков запроса для каждого отчета. Однако следует помнить, что после выполнения запроса на создание таблицы, данные в этой таблице не изменяются.

5 Запросы SQL (запросы на объединение, запросы к серверу, управляющие запросы, подчиненные запросы). Запрос SQL - это запрос, создаваемый при помощи инструкций SQL (структурированный язык запросов). Примерами запросов SQL могут служить запросы на объединение, запросы к серверу, перекрестные и подчиненные запросы.

В режиме Конструктора запрос позволяет ввести условия отбора выводимых на экран полей и записей таблицы и установить порядок их отображения.

Перед тем, как изучить процесс создания запроса и работы с запросом, познакомимся с диалоговым окном «Запрос на выборку в режиме конструктора».

Панель, расположенная в верхней части окна, содержит список всех полей из таблицы. Нижняя половина окна запроса включает в себя область, называемую бланк запроса, он обладает следующими свойствами.

В первой строке определяется набор полей, включаемый в динамический набор.

Создадим запрос на выборку жестких дисков, имеющих емкость не менее 8 Гбайт при цене менее 150 условных единиц. Результирующая таблица должна содержать также адрес поставщика и номер его телефона.

1. Запустите программу Microsoft Access (Пуск - Программы - Microsoft Access).
2. В окне Microsoft Access включите переключатель Открыть базу данных, выберите ранее созданную базу Комплектующие и щелкните на кнопке ОК.
3. В окне Комплектующие: база данных откройте панель Запросы. Дважды щелкните на значке Создание запроса в режиме Конструктора, откроется бланк запроса по образцу. Одновременно с ним откроется диалоговое окно Добавление таблицы.
4. В окне Добавление таблицы выберите таблицу Поставщики и щелкните на кнопке Добавить. Закройте окно Добавление таблицы.

5. В списке полей таблицы Поставщики выберите поля, включаемые в результирующую таблицу: Компонент, Модель, Цена оптовая, Поставщик, Телефон. Выбор производите двойными щелчками на именах полей.

6. Задайте условие отбора для поля Компонент. В соответствующую строку введите: Жесткий диск. Из таблицы будут выбираться не все изделия, а только жесткие диски.

7. Задайте условие отбора для поля Цена оптовая. В соответствующую строку введите: < 150. Из таблицы будут выбираться только изделия, имеющие цену менее 150 условных единиц.

8. Надо задать условие отбора по основному потребительскому параметру — емкости жесткого диска. Однако в таблице Поставщики такого поля нет. С другой стороны, в ней есть поле Модель, которое однозначно определяет параметры изделия. Благодаря тому, что по полю Модель у нас установлена связь с таблицей Комплектующие, мы получаем возможность ввести в запрос поле Основной параметр, взяв его из другой таблицы.

Добавьте список полей таблицы Комплектующие в верхнюю часть бланка запроса по образцу. Для этого щелкните правой кнопкой мыши в верхней области бланка и в открывшемся контекстном меню выберите пункт Добавить таблицу — откроется уже знакомое нам окно Добавление таблицы. Выберите в нем таблицу Комплектующие.

9. Двойным щелчком на поле Основной параметр в списке полей таблицы Комплектующие введите это поле в бланк запроса по образцу.

10. В строке Условие отбора столбца Основной параметр введите условие >8 (емкость диска более восьми гигабайт).

11. Закройте бланк запроса по образцу. При закрытии запроса введите его имя — Выбор комплектующих.

12. В окне Комплектующие: база данных откройте только что созданный запрос и рассмотрите результирующую таблицу. Ее содержательность зависит от того, что было введено в таблицы Комплектующие и Поставщики при их наполнении в упражнении 1. Если ни одно изделие не соответствует условию отбора и получившаяся результирующая таблица не имеет данных, откройте базовые таблицы и наполните их модельными данными, позволяющими проверить работу запроса.

Формирование запросов для многотабличной базы данных

Типы запросов:

Запрос на Выборку - выбирает данные из взаимосвязанных таблиц и других запросов. Результатом является таблица, которая существует до закрытия запроса.

Перекрестный Запрос - предназначен для группирования данных и представления их в компактном виде, удобен для анализа. В соответствии с заданием начните формировать запросы.

Запрос на Создание Таблицы - основан на запросе на выборку, но результат сохраняется в таблице

Запросы на Обновление, Добавление, Удаление - Запросы ДЕЙСТВИЯ, в результате которых изменяются данные в таблице.

1. Создайте запрос на выборку, в котором на экран должен выводиться состав 151 группы. Для этого:

откройте вкладку ЗАПРОСЫ;

нажмите кнопку СОЗДАТЬ;

в появившемся окне выберите ПРОСТОЙ ЗАПРОС и нажмите кнопку ОК;

в появившемся окне в ячейке ТАБЛИЦЫ/ЗАПРОСЫ выберите из раскрывающегося списка таблицу СТУДЕНТЫ;

перенесите все поля из окна ДОСТУПНЫЕ ПОЛЯ в окно ВЫБРАННЫЕ ПОЛЯ;

нажмите кнопку ДАЛЕЕ; выводить надо все поля, поэтому еще раз нажмите кнопку ДАЛЕЕ;

в появившемся окне введите имя запроса ГРУППА;

нажмите кнопку ГОТОВО; на экране появится таблица с данными запроса, но вам надо, чтобы при выполнении запроса спрашивался номер группы (для реализации этого перейдите в режим конструктора);

в строке УСЛОВИЯ ОТБОРА для поля НОМЕР ГРУППЫ введите [Введите номер группы];

выполните запрос, выполнив команду ЗАПРОС=> ЗАПУСК;

на экране появится таблица с данными о студентах выбранной группы;

сохраните запрос и закройте таблицу запроса.

В строке УСЛОВИЯ ОТБОРА могут применятся операции сравнения и логические операции, интервалы (Between And), шаблон (Like), встроенные функции (например, DATE() текущая дата), выражения (DATE() - 10 десять дней назад).

Для просмотра запроса в режиме SQL необходимо воспользоваться пунктом меню ВИД - РЕЖИМ SQL.

2. Создайте запрос с параметрами, в котором выводятся оценки студентов заданной группы по заданной дисциплине. Для этого:

на вкладке ЗАПРОСЫ нажмите кнопку СОЗДАТЬ;

выберите ПРОСТОЙ ЗАПРОС и нажмите ОК;

выберите таблицу СТУДЕНТЫ и перенесите поля ФАМИЛИЯ, ИМЯ, ОТЧЕСТВО, НОМЕР ГРУППЫ в окно ВЫДЕЛЕННЫЕ ПОЛЯ;

Внимание! В дальнейшем под фразой В ТАБЛИЦЕ ….. ВЫБЕРИТЕ ПОЛЕ ….. будем понимать выбор таблицы, выбор поля и перенос его в окно ВЫДЕЛЕННЫЕ ПОЛЯ.

в таблице ДИСЦИПЛИНЫ выберите поле НАЗВАНИЕ ДИСЦИПЛИНЫ;

в таблице ОЦЕНКИ выберите поле ОЦЕНКИ, вы сформировали 6 полей запроса - они связаны между собой посредством схемы данных;

нажмите кнопку ДАЛЕЕ, затем в появившемся окне снова нажмите кнопку ДАЛЕЕ;

в появившемся окне введите имя запроса ОЦЕНКИ ГРУППЫ, затем щелкните по ячейке ИЗМЕНЕНИЕ СТРУКТУРЫ ЗАПРОСА (в ней должна появиться черная точка) - это позволит сразу перейти в режим конструктора;

нажмите кнопку ГОТОВО;

в строке УСЛОВИЯ ОТБОРА для поля НОМЕР ГРУППЫ введите фразу в квадратных скобках: [Введите номер группы];

в строке УСЛОВИЯ ОТБОРА для поля НАЗВАНИЕ ДИСЦИПЛИНЫ введите фразу: [Введите название дисциплины];

выполните запрос;

в первом появившемся диалоговом окне введите номер группы, затем нажмите ОК, во втором – название дисциплины и нажмите ОК; на экране появится таблица со списком группы и оценками по дисциплине;

сохраните запрос и закройте таблицу запроса.

3. Создайте перекрестный запрос о среднем балле в группах по дисциплинам. Но такой запрос строится на основе одной таблицы или одного запроса. Поэтому надо сначала сформировать запрос, в котором были бы поля НОМЕР ГРУППЫ, НАЗВАНИЕ ДИСЦИПЛИНЫ и ОЦЕНКИ. Для этого:

на вкладке ЗАПРОСЫ нажмите кнопку СОЗДАТЬ;

выберите ПРОСТОЙ ЗАПРОС и нажмите ОК;

выберите из таблицы СТУДЕНТЫ поле НОМЕР ГРУППЫ;

выберите из таблицы ДИСЦИПЛИНЫ поле НАЗВАНИЕ ДИСЦИПЛИНЫ;

выберите из таблицы ОЦЕНКИ поле ОЦЕНКИ;

нажмите кнопку ДАЛЕЕ, затем в появившемся окне снова нажмите кнопку ДАЛЕЕ;

в появившемся окне введите имя запроса ДИСЦИПЛИНЫ ОЦЕНКИ ГРУППЫ;

нажмите кнопку ГОТОВО;

сохраните запрос и закройте таблицу запроса.

Теперь можно создавать перекрестный запрос. Для этого:

на вкладке ЗАПРОСЫ нажмите кнопку СОЗДАТЬ;

выберите ПЕРЕКРЕСТНЫЙ ЗАПРОС и нажмите кнопку ОК;

щелкните по ячейке ЗАПРОСЫ, выберите ДИСЦИПЛИНЫ ОЦЕНКИ ГРУППЫ и нажмите кнопку ДАЛЕЕ;

для заголовков строк выберите поле НАЗВАНИЕ ДИСЦИПЛИНЫ и нажмите кнопку ДАЛЕЕ;

для заголовков столбцов выберите поле НОМЕР ГРУППЫ и нажмите кнопку ДАЛЕЕ;

выберите функцию AVG, т.е. среднее значение (она по умолчанию уже выделена) и нажмите кнопку ДАЛЕЕ;

введите название запроса СРЕДНИЕ ОЦЕНКИ и нажмите кнопку ГОТОВО; откроется таблица перекрестного запроса;

обратите внимание на то, что Access создает еще итоговое значение средних оценок по дисциплинам;

закройте таблицу запроса.

4. Создайте итоговый запрос ОТЛИЧНИКИ с выполнением вычислений над группами записей. Для этого:

на вкладке ЗАПРОСЫ нажмите СОЗДАТЬ;

выберите ПРОСТОЙ ЗАПРОС;

в таблице студенты выберите поля ФАМИЛИЯ, ИМЯ, ОТЧЕСТВО, НОМЕР ГРУППЫ, а в таблице ОЦЕНКИ выберите поле ОЦЕНКИ;

нажмите кнопку ДАЛЕЕ, затем в появившемся окне снова нажмите кнопку ДАЛЕЕ;

в появившемся окне введите имя запроса ОТЛИЧНИКИ;

щелкните по ячейке ИЗМЕНЕНИЕ СТРУКТУРЫ ЗАПРОСА;

нажмите кнопку ГОТОВО.

Информация. Для создания этого запроса нужно воспользоваться операцией группировки. Будем считать отличниками тех студентов, которые набрали за четыре экзамена 20 баллов. Операция группировки позволит просуммировать оценки студентов по всем экзаменационным дисциплинам.

Для выполнения групповых операций нажмите на панели инструментов клавишу или выполните команду ВИД=> ГРУППОВЫЕ ОПЕРАЦИИ;

в строке ГРУППОВЫЕ ОПЕРАЦИИ поля ОЦЕНКИ щелкните по ячейке групповые операции. Откройте раскрывающийся список и выберите функцию SUM;

в строке УСЛОВИЯ ОТБОРА поля ОЦЕНКИ введите 20;

выполните полученный запрос.

5. Создайте запрос с использованием вычисляемых полей из таблицы ПРЕПОДАВАТЕЛИ:

В запросах, в отличие от таблиц, над полями могут производиться вычисления. При этом могут использоваться как арифметические выражения, так и встроенные функции ACCESS. Вычисляемое поле, включенное в запрос позволяет получить новое поле с результатами вычислений только в ТАБЛИЦЕ ЗАПРОСА и не создает полей в таблицах БД. Сформировать выражение можно при помощи ПОСТРОИТЕЛЯ ВЫРАЖЕНИЙ, который запускается из контекстного меню, связанного со строкой УСЛОВИЕ ОТБОРА на бланке или при помощи соответствующей кнопки на панели инструментов. При составлении выражений имена полей заключаются в квадратные скобки, символьные константы - в кавычки, имена объектов БД отделяются от полей "!";

на вкладке ЗАПРОСЫ нажмите кнопку СОЗДАТЬ;

выберите ПРОСТОЙ ЗАПРОС;

в таблице ПРЕПОДАВАТЕЛИ выберите поля ФАМИЛИЯ, ИМЯ, ОТЧЕСТВО, КАФЕДРА, ДОЛЖНОСТЬ, ДАТА РОЖДЕНИЯ;

нажмите кнопку ДАЛЕЕ, затем в появившемся окне снова нажмите кнопку ДАЛЕЕ;

в появившемся окне введите имя запроса ВОЗРАСТ ПРЕПОДАВАТЕЛЕЙ;

нажмите кнопку ГОТОВО;

выберите вариант ИЗМЕНЕНИЕ СТРУКТУРЫ ЗАПРОСА;

в режиме конструктора нужно добавить поле, которого нет в таблице ПРЕПОДАВАТЕЛИ;

это вычисляемое поле ВОЗРАСТ, которое вычисляется следующим образом из текущей даты вычесть дату рождения преподавателя и полученную разность разделить на 366, так как разница дат получится в днях. Для этого в заголовке вычисляемого поля при помощи контекстного меню вызвать команду ПОСТРОИТЕЛЬ ВЫРАЖЕНИЙ и набрать следующее выражение: ВОЗРАСТ (DATE()-[ПРЕПОДАВАТЕЛИ] ![ДА-ТА_РОЖД])/366, где DATE() текущая дата;

Нажмите кнопку ГОТОВО.

Информация. При составлении выражений имена полей заключаются в квадратные скобки , символьные константы - в кавычки , имена объектов БД отделяются от полей "!" .

6. Создайте запрос действие на изменение зарплаты преподавателей в таблице ПРЕПОДАВАТЕЛИ. Для этого:

на вкладке ЗАПРОСЫ нажмите СОЗДАТЬ;

выберите ПРОСТОЙ ЗАПРОС;

в таблице ПРЕПОДАВАТЕЛИ выберите поле ЗАРПЛАТА;

нажмите кнопку ДАЛЕЕ, затем в появившемся окне снова нажмите кнопку ДАЛЕЕ;

в появившемся окне введите имя запроса ИЗМЕНЕНИЕ ЗАРПЛАТЫ;

щелкните по ячейке ИЗМЕНЕНИЕ СТРУКТУРЫ ЗАПРОСА;

нажмите кнопку ГОТОВО;

в строке УСЛОВИЯ ОТБОРА введите < 2000;

откройте пункт меню ЗАПРОС и выберите ОБНОВЛЕНИЕ;

в строке конструктора запроса ОБНОВЛЕНИЕ в поле ЗАРПЛАТА введите [ЗАРПЛАТА]* 1,1;

выполните запрос, подтвердив готовность на обновление данных;

закройте запрос, подтвердив его сохранение;

откройте форму ПРЕПОДАВАТЕЛИ;

просмотрите изменение зарплаты у преподавателей, получающих меньше 2000 р.;

закройте форму.

7. Создайте запрос на создание архива для отчисленных студентов и на удаление студента из таблицы СТУДЕНТЫ. Для этого:

на вкладке ЗАПРОСЫ нажмите СОЗДАТЬ;

выберите ПРОСТОЙ ЗАПРОС;

в таблице СТУДЕНТЫ выберите все поля;

выберите ПОДРОБНЫЙ ОТЧЕТ и нажмите кнопку ДАЛЕЕ;

в появившемся окне введите имя запроса СОЗДАНИЕ_АРХИВА;

щелкните по кнопке ИЗМЕНЕНИЕ СТРУКТУРЫ ЗАПРОСА;

нажмите кнопку ГОТОВО;

в строке УСЛОВИЯ ОТБОРА введите: в поле ФАМИЛИЯ - [ввод фамилии], в поле ИМЯ - [ввод имени], в поле ОТЧЕСТВО - [ввод отчества], в поле НОМЕР ГРУППЫ - [ввод группы];

откройте пункт меню ЗАПРОС и выберите команду СОЗДАНИЕ ТАБЛИЦЫ;

выполните запрос СОЗДАНИЕ АРХИВА;

для этого в режиме диалога введите данные о студенте помещаемом в архив, ГРУППА – номер группы;

в результате выполнения запроса на СОЗДАНИЕ АРХИВА должна появиться новая таблица АРХИВ;

посмотрите ее содержание, там должна быть запись о студенте;

создайте запрос на удаление записи из таблицы СТУДЕНТЫ, предварительно помещенной в АРХИВ;

на вкладке ЗАПРОСЫ нажмите кнопку СОЗДАТЬ;

выберите ПРОСТОЙ ЗАПРОС;

в таблице СТУДЕНТЫ выберите все поля и нажмите кнопку ДАЛЕЕ;

выберите подробный отчет, затем в появившемся окне снова нажмите кнопку ДАЛЕЕ;

в появившемся окне введите имя запроса ОТЧИСЛЕНИЕ СТУДЕНТА;

щелкните по кнопке ИЗМЕНЕНИЕ СТРУКТУРЫ ЗАПРОСА;

откройте пункт меню ЗАПРОС и выберите УДАЛЕНИЕ;

в строке УСЛОВИЯ ОТБОРА введите: в поле ФАМИЛИЯ - [ввод фамилии], в поле ИМЯ -[ввод имени], в поле ОТЧЕСТВО - [ввод отчества], в поле НОМЕР ГРУППЫ - [ввод группы];

закройте запрос;

выполните запрос ОТЧИСЛЕНИЕ СТУДЕНТА;

для этого введите фамилию, имя, отчество, номер группы;

откройте форму СТУДЕНТЫ и убедитесь в удалении записи о студенте;

закройте форму.

Самостоятельно создайте запрос на добавление в АРХИВ других записей таблицы СТУДЕНТ и назовите его ДОБАВЛЕНИЕ В АРХИВ и запрос на ВОССТАНОВЛЕНИЕ СТУДЕНТА из АРХИВА в таблицу СТУДЕНТ.

Для каждого из созданных запросов создайте форму (можно рекомендовать автоформу в столбец или ленточную автоформу), удобную для просмотра данных.

4. Перечень оборудования – персональный компьютер.

5. Порядок выполнения работы (Задания)

5.1. В соответствии с вариантом задания выполнить запросы к базе данных.

6. Содержание отчета

6.1. Наименование работы

6.2. Цель работы

6.3. Выполненные задания в соответствии с вариантом.

7. Список литературы

7.3. Федорова Г.Н. Основы проектирования баз данных: Учебное пособие для СПО / Г.Н. Федорова. - М.: Академия, 2018.- 220с. (*Основное печатное издание ОПИ-1*)

7.4. Кумскова, И. А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.

ПРАКТИЧЕСКАЯ (ЛАБОРАТОРНАЯ) РАБОТА № 6.
Проведение сортировки и фильтрации данных. Поиск данных по одному и нескольким полям. Поиск данных в таблице

1. Цель работы

Получить практический опыт проведения сортировки и поиска и фильтрации записей базы данных.

2. Время выполнения работы – 2 часа.

3. Краткие теоретические сведения

В таблице, содержащей сотни или тысячи записей, бывает довольно трудно найти нужную информацию. Microsoft Access представляет несколько средств поиска и просмотра информации:

- Поиск записей по тексту, содержащемуся в любом из полей.
- Расстановка записей по возрастанию или убыванию значений конкретного поля.
- Изменение количества отображаемых записей с помощью фильтра.

Поиск текста в записях

Поиск одной или нескольких записей с нужным текстом, удобно осуществлять средствами поиска. С помощью команды Найти из меню Правка или кнопки Найти с панели инструментов Таблица в режиме таблицы введите искомый текст и затем укажите, должна ли Access искать во всех полях или же только в текущем, определите положение текста внутри поля и установите, нужно ли при поиске учитывать регистр символов. Отыскав первую подходящую запись, Access выделит искомый текст. После этого Вы можете либо продолжить, либо отменить поиск.

Поиск текста в текущем поле:

1. Отобразите таблицу в Режиме таблицы.
2. Щелкните столбец, в котором Вы надеетесь найти нужный текст.
3. Щелкните кнопку Найти на панели инструментов Таблица в режиме таблицы.
4. Введите искомый текст. Если при поиске нужно учитывать регистр символов, щелкните флажок С учетом регистра, иначе – сбросьте флажок.
5. Для начала поиска щелкните кнопку Найти.
6. Для перехода к следующей записи, содержащий заданный текст, щелкните кнопку Найти далее.
7. Для закрытия диалогового окна и возврата в таблицу щелкните кнопку Закрыть.

Поиск текста во всех полях:

1. Отобразите таблицу в Режиме таблицы.
2. Щелкните кнопку Найти на панели инструментов Таблица в режиме таблицы.

3. Введите искомый текст.

4. Сбросьте флажок Только в текущем поле.

5. Для начала поиска щелкните кнопку Найти.

6. Для перехода к следующей записи, содержащий заданный текст, щелкните кнопку Найти далее.

7. Для закрытия диалогового окна и возврата в таблицу щелкните кнопку Закрыть.

Изменение порядка записей в таблице

Сортировкой можно изменить последовательность записей в таблице, расположив их по возрастанию или по убыванию значений выбранного поля. Сортировку можно задать по нескольким полям.

Расположение записей по возрастанию значений одного поля:

1 Установите курсор в любом месте столбца, по полям которого хотите отсортировать записи.

2. Щелкните кнопку Сортировка по возрастанию на панели инструментов Таблица в режиме таблицы.

Сортировка записей по нескольким полям

При изменении порядка записей в таблице иногда требуется задать сортировку более чем по одному полю. Такая операция называется комбинированной сортировкой. В этом случае поля должны следовать в таблице друг за другом. Access сама отсортирует их слева направо - лишь укажите нужные поля. Например, чтобы составить список клиентов и контактирующих с ними сотрудников, сначала отсортируйте записи по полю Клиент, а затем – по полю Сотрудник.

Сортировка записей по нескольким полям:

1. Отобразить таблицу в Режиме таблицы.

2. Щелкните заголовок первого столбца, на основе значений которого хотите отсортировать записи, и, не отпуская кнопку мыши, протащите указатель вправо по столбцам, чтобы задать и их поля.

3. Для сортировки записей по возрастанию или по убыванию щелкните кнопку Сортировка по возрастанию или Сортировка по убыванию на панели инструментов Таблица в режиме таблицы.

Поиск и замена текста в таблице

Иногда при обновлении данных таблицы необходимо внести одинаковые изменения сразу в несколько записей. Для поиска и замены фрагмента текста во всей таблице используется механизм замены. Access позволяет задавать либо каждую замену по отдельности, либо все сразу. Кроме того, Вы можете указать направление поиска, необходимо учитывать регистр символов, а также задать проверку на совпадение со всем полем или с его частью.

Поиск и замена текста:

1. Отобразить таблицу в Режиме таблицы.

2. Щелкните столбец, в котором Вы рассчитываете найти искомый текст.

3. Щелкните меню Правка и затем команду Заменить.

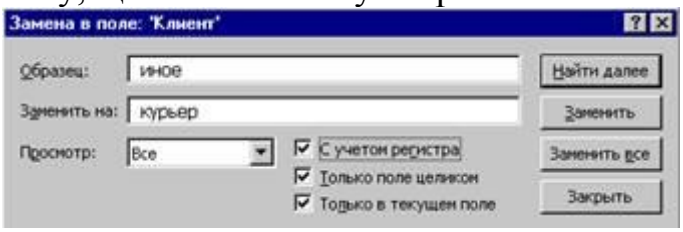
4. В поле Образец введите текст, подлежащий замене.

5. В поле Заменить на введите новый текст.

6. Для поиска первого упоминания текста щелкните Найти далее.

7. Для замены первого найденного фрагмента текста щелкните кнопку Заменить, для замены всех найденных фрагментов – кнопку Заменить все, а для перехода к следующему фрагменту – кнопку Найти далее.

8. Закончив работу, щелкните кнопку Закрыть.



Просмотр заданных записей с помощью фильтра

Фильтр позволяет отображать на экране не все записи таблицы, а только нужные. Access фильтрует записи по одному или нескольким полям, после чего отображает только те из них, которые удовлетворяют заданным условиям. Для придания таблице первоначального вида отключите фильтр.

Фильтрование таблицы по выделенному:

1. Отобразить таблицу в Режиме таблицы.

2. Установите курсор в поле, по которому хотите задать фильтрование.

3. Щелкните кнопку Фильтр по выделенному (1 кнопка) на панели инструментов Таблица в режиме таблицы. В нижней части окна таблицы появится сообщение о количестве записей, удовлетворяющих критериям Вашего фильтра. Буквы ФЛТР в строке состояния означают, что фильтр активен.

Отключение фильтра:

Щелкните кнопку Удалить фильтр (3 кнопка) на панели инструментов Таблица в режиме таблицы. Из строки состояния исчезнут буквы ФЛТР.

Сохранение фильтра

Если Вы предполагаете, что созданный Вами сложный фильтр или критерии фильтрования могут пригодиться снова, сохраните их в виде запроса. Хотя запросы имеют некоторые уникальные возможности, которые выходят за пределы возможностей фильтров, в данном случае Вы можете создать простой запрос, чтобы сохранить уже созданный фильтр.

Сохранение фильтра как запроса:

1. Отобразите отфильтрованную таблицу в Режиме таблицы.

2. Щелкните меню Записи, установите указатель мыши на команду Фильтр и в появившемся подменю щелкните команду Расширенный фильтр. В окне расширенного фильтра Вы увидите принятую схему условий отбора записей.

3. Щелкните меню Файл и затем – команду Сохранить как запрос.

4. Введите в текстовое поле Имя запроса имя сохраняемого запроса.

5. Для сохранения фильтра в виде запроса щелкните кнопку ОК. Имя сохраненного запроса появится во вкладке Запросы окна базы данных.

Создание сложных фильтров с помощью Конструктора фильтров

В Access для разработки сложных фильтров можно воспользоваться Конструктором. Когда Вы вписываете в одну вкладку несколько условий одновременно, то создаете И-фильтр. Он отображает только записи, удовлетворяющие всем заданным во вкладке критериям. Записи, удовлетворяющие одному из нескольких условий, просматривают с помощью ИЛИ-фильтра. При его применении отображаются все записи, соответствующие любому из условий, заданных во вкладке Найти или в любой из активных вкладок Или.

Создание И-фильтра:

1. Отобразите отфильтрованную таблицу в Режиме таблицы.

2. Щелкните кнопку Изменить фильтр (2 кнопка) на панели инструментов Таблица в режиме таблицы.

3. Если необходимо отказаться от установок предыдущего фильтра, щелкните кнопку Очистить бланк на панели инструментов Фильтр.

4. Щелкните пустую ячейку под заголовком поля, значения которого Вы хотите использовать для фильтрования.

5. Щелкните стрелку раскрывающегося списка и выберите значение поля, по которому будет проведено фильтрование. Чтобы фильтр работал правильно, текст, вводимый в ячейку бланка конструктора фильтра, заключите в кавычки.

6. Выполните аналогичные действия для каждого поля, задаваемого для фильтрования.

7. Щелкните кнопку Применение фильтра на панели инструментов Фильтр.

Создание ИЛИ-фильтра:

1. Отобразить таблицу в Режиме таблицы.

2. Щелкните кнопку Изменить фильтр на панели инструментов Таблица в режиме таблицы.

3. Если необходимо отказаться от установок предыдущего фильтра, щелкните кнопку Очистить бланк на панели инструментов Фильтр.

4. Щелкните пустую ячейку под заголовком поля, значения которого Вы хотите использовать для фильтрования.

5. Щелкните стрелку раскрывающегося списка и выберите значение поля.

6. Выполните аналогичные действия для каждого поля, заданного для фильтрования. Любое из них задает дополнительные критерии отбора записей.

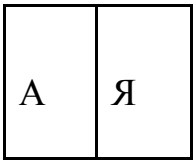
7. Для задания дополнительных условий фильтрования щелкните вкладку Или внизу окна Конструктора.

8. Щелкните кнопку Применение фильтра на панели инструментов Фильтр.

Сортировка данных (в режиме таблицы)

Сначала следует выделить поле, по которому надо произвести сортировку данных (т. е. надо выделить весь столбец нашей таблицы), для чего достаточно

щелкнуть на имени поля в заголовке таблицы. Затем нужно щелкнуть на одной из двух кнопок:



С помощью первой из этих кнопок производится сортировка текстовых данных по алфавиту, а числовых - по возрастанию, с помощью второй данные сортируются в обратном порядке.

Фильтрация данных (в режиме таблицы)

Иногда бывает необходимо увидеть на экране не все записи базы данных (строки таблицы), а лишь те из них, которые отвечают определенным условиям. Этот процесс называется фильтрацией или выборкой данных.

Чтобы осуществить выборку необходимых данных, следует:

1. Выполнить команду Записи, Фильтр, Изменить фильтр.
2. Щелкнуть на имени требуемого поля в списке и ввести условие (маску) выбора.
3. Выполнить команду Фильтр, Применить фильтр.

На экране появится таблица, которая будет содержать строки, данные которых отвечают указанному условию. Чтобы вновь увидеть всю таблицу целиком, следует выполнить команду Записи, Удалить фильтр.

В условиях помимо математических равенств и неравенств можно использовать знак «*», который заменяет собой любую последовательность символов, или знак «?», заменяющий один символ.

Ниже приведены варианты условий выборки данных из нашей базы и результаты выборки.

Условие: зарплата: >=200 (сотрудники, у которых зарплата больше или равна 200)

код	Фамилия	Имя	Телефон	Зарплата
1	Иванов	Егор		200
3	Ефимов	Евгений		210

Условие: имя: Н* (сотрудники, имена которых начинаются на букву Н)

код	Фамилия	Имя	Телефон	Зарплата
2	Семенов	Николай		160

4. Перечень оборудования – персональный компьютер.

5. Порядок выполнения работы (Задания)

5.1. В соответствии с вариантом задания выполнить сортировку, создать фильтры и осуществить поиск по одному и нескольким полям в базе данных.

6. Содержание отчета

6.1. Наименование работы

6.2. Цель работы

6.3. Выполненные задания в соответствии с вариантом.

7. Список литературы

7.1. Федорова Г.Н. Основы проектирования баз данных: Учебное пособие для СПО / Г.Н. Федорова. - М.: Академия, 2018.- 220с. (*Основное печатное издание ОПИ-1*)

7.2. Кумскова, И. А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.

ПРАКТИЧЕСКАЯ (ЛАБОРАТОРНАЯ) РАБОТА № 7.
Работа с переменными. Написание программного файла и работа с табличными файлами. Заполнение массива из табличного файла. Заполнение табличного файла из массива

1. Цель работы

2. Время выполнения работы – 2 часа.

3. Краткие теоретические сведения

Переменные бывают локальными и глобальными. Локальные - действуют на уровне функций, глобальные доступны из любого места программы. Существуют также и другие ограничения видимости переменных, например, классы. Для определения переменных в Access и ввода кода на Visual Basic существует специальная среда для работы с этими программами.

Для начала определим название нашей программы. Это будет приложение Microsoft Access «Новый калькулятор.mdb». Разработку его начнем с описания переменных, констант, которые потребуются при создании нашей базы данных.

appAccess – это объектная переменная, в которой мы будем создавать новую базу данных

appCM – это константа, которая управляет размером управляющих элементов в формах, отчетах. В Access элементы измеряются в твипах, а нам удобнее указывать размеры в сантиметрах, поэтому, и вводим эту переменную.

appFolder – эта переменная будет содержать главный каталог нашего приложения. Далее мы будем использовать ее для загрузки рисунков, программ, справки и других внешних элементов базы данных

```
Public appAccess As Access.Application 'Приложение программы
Public Const appCM As Integer = 567  'Размерность (1см=567 твип)
Public appFolder As String 'Папка приложения
```

Продолжим создание приложения с разработки всех функций, которые требуются для работы базы данных. Сначала программным способом, т.е. без мастеров, создадим базу данных, затем таблицы, запросы и т.д.

Создадим главную функцию, которая будет связывать в единое целое создание объектов: таблиц, запросов, форм и т.п.

```
Public Function funCreateApp() As Boolean
    On Error GoTo 999 'Назначаем переход по ошибке
    appAccess = New Access.Application 'Создаем приложение
    appFolder = funGetAppFolder '<1> Находим папку
    funCreateDatabase "Новый калькулятор.mdb" '<2> Создаем базу данных
    funCreateTable "Калькулятор" '<3> Создаем таблицу
    funCreateQueries() '<4> Создаем запросы
    funCreateModule() '<5> Импортируем модули
    funCreateForm "Мой калькулятор"   '<6> Создаем форму
    funCreateReport "Отчет об итогах" '<7> Создаем отчет
    funCreateMacro "AutoExec" '<8> Создаем макрос
    funImportModule("Справка.bas", "Справка") '<9> Создаем справку
    funCreateMenu "Калькулятор" '<10> Создаем меню
```

```
    funCloseDatabase "Новый калькулятор.mdb" '<2> Закрываем программу
    funCreateApp = True 'Возвращаем значение
    Exit Function 'Выходим из программы
999:
    MsgBox Err.Description 'Сообщаем об ошибке
    Err.Clear() 'Очищаем поток от ошибок
End Function
```

Профессионально выполненная программа всегда имеет внешние файлы, например basic-программы, рисунки *.bmp. Чтобы к ним обеспечить быстрый доступ, напишем функцию, которая будет определять папку, где установлена основная программа.

Применяется эта программа для назначения данных глобальной переменной appFolder

```
Public Function funGetAppFolder() As String
    Dim fs
    On Error GoTo 999  'Назначаем переход по ошибке
    fs = CreateObject("Scripting.FileSystemObject") 'Создаем файловую систему
    funGetAppFolder = fs.GetFile(CurrentDb.Name).ParentFolder 'Находим папку
    fs = Nothing 'Уничтожаем переменную
    Exit Function 'Выходим из программы
999:
    MsgBox Err.Description 'Сообщаем об ошибке
    Err.Clear() 'Очищаем поток от ошибок
End Function
```

Если мы открыли базу данных, то ее необходимо закрыть. Это хороший способ программирования, хотя при закрытии окна Access (appAccess.Quit) база данных закрывается автоматически.

В этом примере база данных «Новый калькулятор.mdb» (переменная strMDB) будет закрыта, все ее объекты сохранены, а потом произойдет сжатие данных.

```
Public Function funCloseDatabase(strMDB As String) As Boolean
    On Error GoTo 999  'Назначаем переход по ошибке
    funCloseDatabase = False 'Возвращаем результат при ошибке
    appAccess.CloseCurrentDatabase() 'Закрываем базу данных
    appAccess.Quit acQuitSaveAll 'Выходим с сохранением
    funCompactDatabase strMDB '<2> Сжимаем базу данных
    funCloseDatabase = True 'Возвращаем результат
    Exit Function 'Выходим из программы
999:
    MsgBox Err.Description 'Сообщаем об ошибке
    Err.Clear() 'Очищаем поток от ошибок
End Function
```

При работе с базой данных Microsoft Access постепенно будет расти размер базы данных.

Это связано с тем, что, удаляя элементы или записи «физически» из базы они не удаляются, уничтожаются только ссылки.

Для уменьшения размера базы данных, напишем программу для сжатия ее файла. В качестве имени файла, передаваемого в программу, введем переменную StrMDB.

```
Public Function funCompactDatabase(strMDB As String) As Boolean
    Dim tmpMDB As String, fs, sFullPath As String
```

```
    On Error GoTo 999  'Назначаем переход по ошибке
    funCompactDatabase = False 'Возвращаем результат при ошибке
    sFullPath = appFolder & "\" & strMDB 'Устанавливаем полное имя файла
    fs = CreateObject("Scripting.FileSystemObject") 'Создаем файловую систему
    tmpMDB = fs.GetTempName 'Получаем временный файл
    DBEngine.CompactDatabase(sFullPath, tmpMDB, dbLangCyrillic) 'Сжимаем базу данных
    fs.CopyFile(tmpMDB, sFullPath, True) 'Переписываем файл
    Kill tmpMDB 'Удаляем временный файл
    fs = Nothing 'Уничтожаем объект файловой системы
    funCompactDatabase = True 'Возвращаем результат
    Exit Function 'Выходим из программы
999:
    MsgBox Err.Description 'Сообщаем об ошибке
    Err.Clear() 'Очищаем поток от ошибок
End Function
```

4. Перечень оборудования – персональный компьютер.

5. Порядок выполнения работы (Задания)

5.1. В соответствии с вариантом задания выполнить

6. Содержание отчета

6.1. Наименование работы

6.2. Цель работы

6.3. Выполненные задания в соответствии с вариантом.

7. Список литературы

7.1. Федорова Г.Н. Основы проектирования баз данных: Учебное пособие для СПО / Г.Н. Федорова. - М.: Академия, 2018.- 220с. (*Основное печатное издание ОПИ-1*)

7.2. Кумскова, И. А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.

ПРАКТИЧЕСКАЯ (ЛАБОРАТОРНАЯ) РАБОТА № 8.
Добавление записей в табличный файл из двумерного массива. Работа с командами ввода-вывода. Использование функций для работы с массивами

1. Цель работы

2. Время выполнения работы – 2 часа.

3. Краткие теоретические сведения

Функции или Операторы ввода (InputBox) и вывода (MsgBox)

В VBA ввод и вывод информации (для взаимодействия с пользователем) можно осуществлять в диалоговых окнах. Диалоговое окно ввода значений реализуется встроенной функцией **InputBox.** В окне ввода, реализованное функцией InputBox, отображается поле для ввода значения переменной, в которое пользователь должен ввести определенное значение. Далее пользователь должен нажать кнопку ОК.

Функция **InputBox()** имеет следующий синтаксис:

Имя_Переменной = InputBox(Prompt, [Title], [Default], [XPos], [YPos], [HelpFile], [Context])

Где аргументы: Prompt или Сообщение - обязательный аргумент, который задает в диалоговом окне информационное сообщение. Все остальные аргументы являются необязательными. Title задает заголовок окна. На рис 1 приведен модуль, в котором применена функция InputBox.

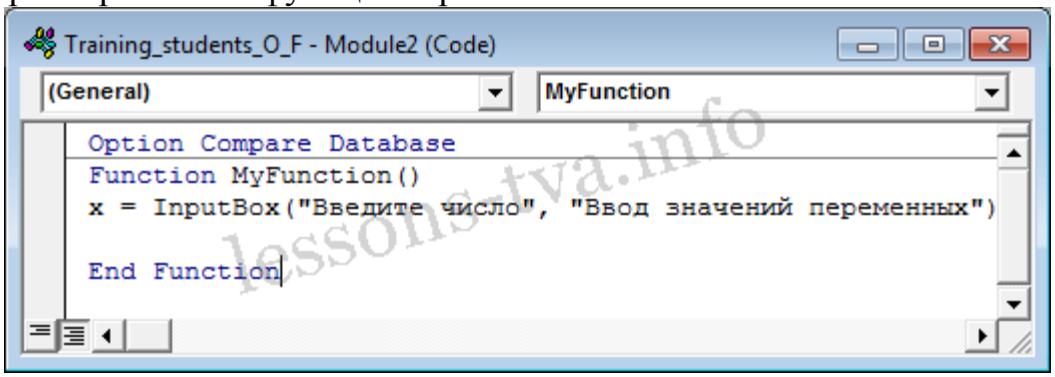


Рис. 1.

После выполнения модуля 2 появляется окно сообщения "Ввод значений переменных" (рис.2), в котором нужно ввести число и нажать кнопку ОК. В окне диалога (Рис.2), реализованном функцией InputBox (рис 1), отображаются: Заголовок окна - Ввод значений переменных; Сообщение - Введите число; Кнопки (по умолчанию) - ОК и Cancel; Поле предназначенное для ввода значений переменной.

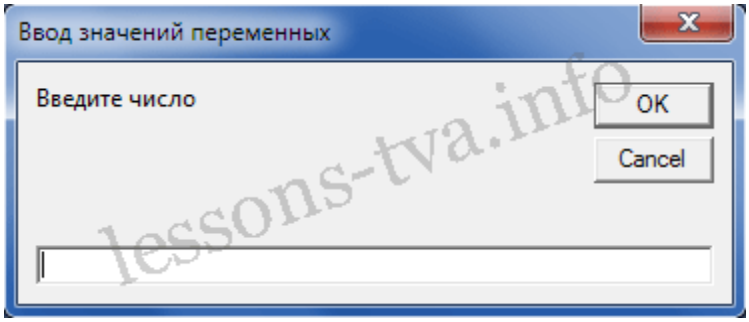


Рис. 2.

Для вывода информации применяются диалоговые окна сообщений, реализуемые оператором **MsgBox** или функцией **MsgBox()**. MsgBox может использоваться как оператор. Оператор MsgBox осуществляет вывод информации в диалоговом окне и устанавливает режим ожидания нажатия кнопки пользователем.

Оператор MsgBox имеет следующий синтаксис: MsgBox Prompt, [Buttons], [Title], [HelpFile], [Context]

Где аргументы: Prompt или Сообщение - обязательный аргумент, задающий в окне выводимое информационное сообщение. Все остальные аргументы являются необязательными. Buttons - Кнопки, которые можно использовать в диалоговом окне вывода сообщений. В окне сообщений могут применяться различные кнопки (ОК, Отмена и т.д.). Если не указывать, какие кнопки необходимо отображать в окне сообщений, то по умолчанию отображается кнопка ОК. Кроме того, в диалоговых окнах вывода сообщений можно использовать различные значки (vbQuestion - значок вопросительного знака, vbExclamation - значок восклицательного знака и т.д.).

Модуль, в котором MsgBox используется как оператор, приведен на рис. 3 (оператор MsgBox "3", vbOKCancel, "Вывод значений").

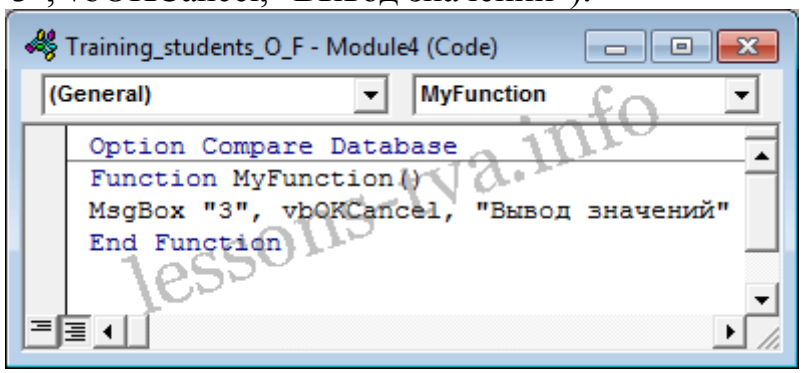


Рис. 3

При запуске модуля 4 на исполнение отображается окно сообщений "Вывод значений" (рис. 4), в котором необходимо нажать кнопку ОК. В окне (Рис.4), реализованном оператором MsgBox (Рис. 3), отображаются: Заголовок окна - Вывод значений; Сообщение - 3; Кнопки - ОК и Отмена.

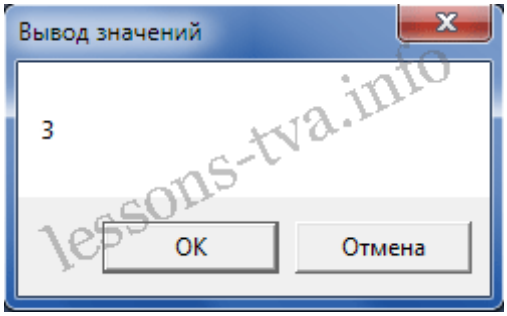


Рис. 4

Например, для вычисления функции типа $y = 5\ x^2 + 7\ x + 9$, можно использовать функцию InputBox и оператор MsgBox (рис. 5)

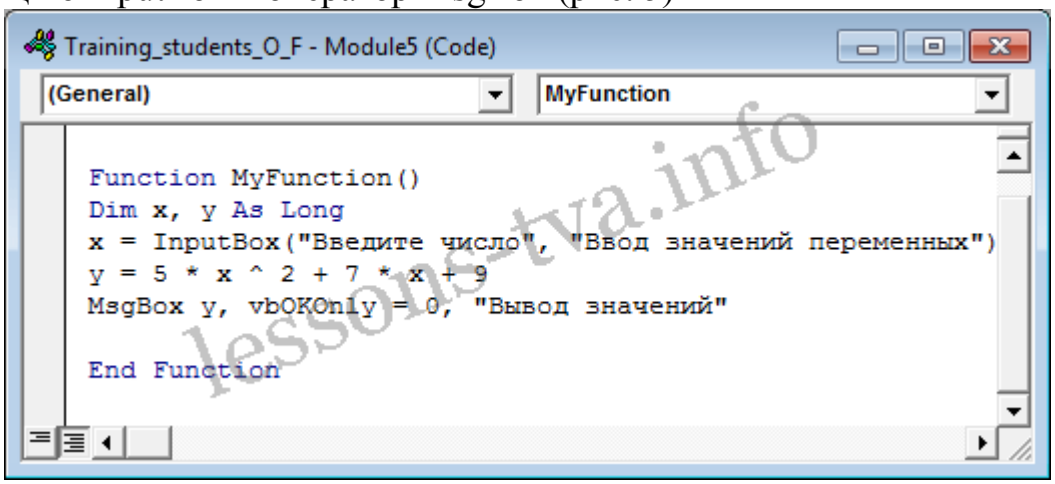


Рис. 5

После выполнения модуля 5 отображается окно ввода

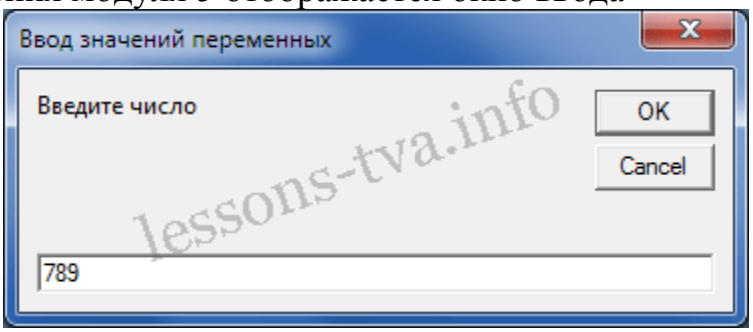


Рис. 6

После ввода числа, например 789, и щелчка на кнопке ОК, появляется окно сообщения, в котором отображается результат вычисления функции $y = 5\ x^2 + 7\ x + 9$.

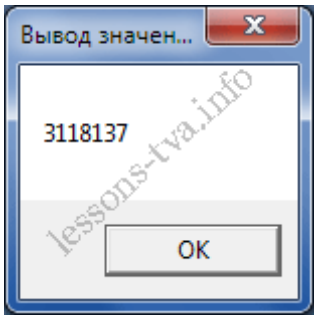


Рис. 7

MsgBox можно использовать в качестве функции. Функция MsgBox() имеет следующий синтаксис: MsgBox (Prompt, [Buttons], [Title], [HelpFile], [Context]). В этом случае в окне диалога используют несколько различных кнопок. При нажа-

тии кнопки в окне диалога функция MsgBox() возвращает значение типа Integer, которое зависит от того, какая из кнопок была нажата в диалоговом окне вывода сообщений.

Dictionary - этот объект можно использовать для создания массивов, даже для форм. Таким образом можно создать интерфейс, который будет открывать 2 одинаковые формы, что в Access нереально создать обычным способом.

```
'   Использование массива Dictionary для таблицы
Public Function funArrayDictionary() As String
Dim s As String, i  As Integer, dbs As Database, rst As Recordset
Dim myArray, myBooks 'Переменные для массива

    On Error GoTo 999 'Обработка ошибки

'1.Открытие таблицы
    Set dbs = CurrentDb 'Выбираем базу данных
    Set rst = dbs.OpenRecordset("SELECT * FROM [Мои книги]") 'Создаем запрос
    If (rst.RecordCount = 0) Then 'Проверяем таблицу
        rst.Close 'Закрываем запрос
        MsgBox "Нет данных" 'Сообщаем об этом
        Exit Function
    End If

'2. Заполнение запроса
    rst.MoveLast
    rst.MoveFirst

'3. Заполнение массива
    Set myArray = CreateObject("Scripting.Dictionary") 'Создаем массив
    myArray.RemoveAll 'Удаляем все
    For i = 0 To rst.RecordCount - 1
          myArray.Add CStr(rst!Ключ), CStr(rst!Книга) 'Добавляем значение в массив
          rst.MoveNext 'переходим на следующую запись
    Next i

'4. Проверка массива
    myBooks = myArray.Items        'Выбираем все книги
    For i = 0 To myArray.Count - 1 'Создаем цикл
        s = s & myBooks(i) & vbCrLf  'Создаем список книг
    Next
    funArrayDictionary = s 'Возвращаем список

'5. Конец примера
    myArray.RemoveAll 'Удаляем массив
    rst.Close
    Set dbs = Nothing '!Внимание. Посылаем ... переменную!
    Exit Function
999:
    MsgBox Err.Description
    Err.Clear
    rst.Close
End Function
```

4. Перечень оборудования – персональный компьютер.

5. Порядок выполнения работы (Задания)

5.1. В соответствии с вариантом задания используйте функции ввода-вывода, функции для работы с массивами.

6. Содержание отчета

6.1. Наименование работы

6.2. Цель работы

6.3. Выполненные задания в соответствии с вариантом.

7. Список литературы

7.1. Федорова Г.Н. Основы проектирования баз данных: Учебное пособие для СПО / Г.Н. Федорова. - М.: Академия, 2018.- 220с. (*Основное печатное издание ОПИ-1*)

7.2. Кумскова, И. А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.

ПРАКТИЧЕСКАЯ (ЛАБОРАТОРНАЯ) РАБОТА № 9.
Создание меню различных видов. Модификация и управление меню

8. Цель работы

9. Время выполнения работы – 2 часа.

10. Краткие теоретические сведения

11. Перечень оборудования – персональный компьютер.

12. Порядок выполнения работы (Задания)

5.2. В соответствии с вариантом задания выполнить ????

13. Содержание отчета

6.4. Наименование работы

6.5. Цель работы

6.6. Выполненные задания в соответствии с вариантом.

14. Список литературы

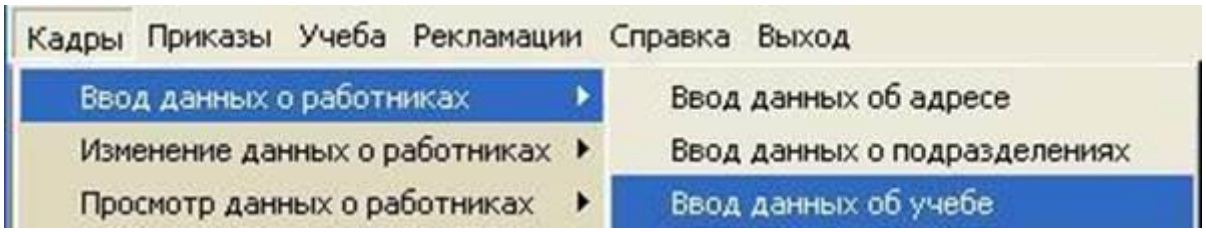
7.3. Федорова Г.Н. Основы проектирования баз данных: Учебное пособие для СПО / Г.Н. Федорова. - М.: Академия, 2018.- 220с. (*Основное печатное издание ОПИ-1*)

7.4. Кумскова, И. А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.

ПРАКТИЧЕСКАЯ (ЛАБОРАТОРНАЯ) РАБОТА № 10.
Создание рабочих и системных окон. Добавление элементов управления рабочим окном

1. Цель работы
2. Время выполнения работы – 2 часа.
3. Краткие теоретические сведения

Система меню – изображение на экране списка команд для выбора пользователем следующего действия системы путем указания выбранной опции средствами управления курсором. На рис. 11 представлено главное меню для учета кадров с действием первого пункта, вызывающего ниспадающее меню.



Система меню проектируется посредством создания макросов.

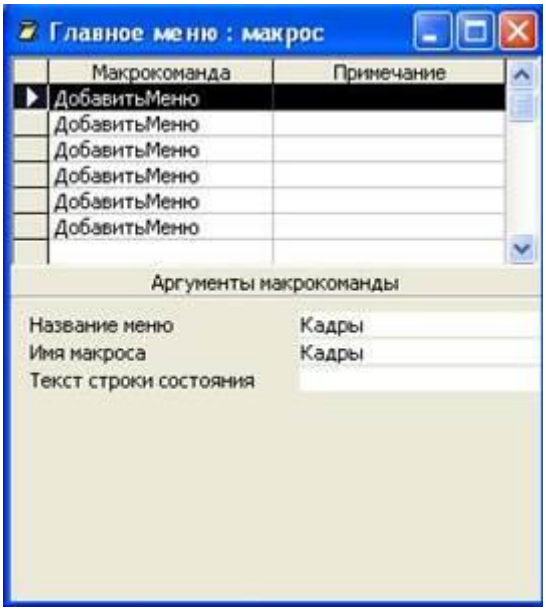
Макросом называют набор из одной или более команд, выполняющих определенные, часто используемые операции, например, открытие форм или печать отчетов. Макросы могут быть полезны для автоматизации часто выполняемых действий. Если записать эти действия в виде макроса, то они будут выполняться автоматически при запуске макроса. Например, при нажатии пользователем кнопки можно запустить макрос, который распечатает отчет или выведет на экран форму. Поэтому перед проектированием меню должны быть созданы объекты приложения (таблицы, запросы, формы, отчеты). Основным компонентом макроса является *макрокоманда*, которая самостоятельно или в комбинации с другими макрокомандами определяет выполняемые в макросе действия. Серия макрокоманд, из которых состоит макрос, выполняется каждый раз при его запуске.

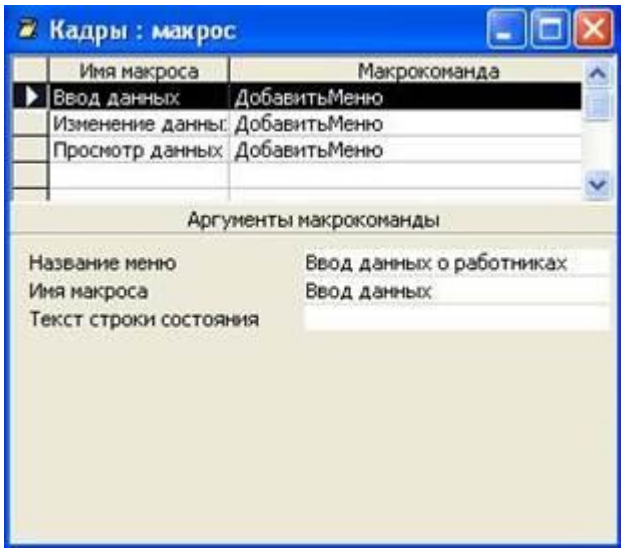
Для создания макроса необходимо открыть вкладку *Создание* и нажать кнопку *Макросы* в окне базы данных. Появится окно *конструктора макросов*.

В окне конструктора макросов создается список макрокоманд, которые требуется выполнить при запуске макроса. При первом открытии конструктора макросов будут отображены столбцы *Макрокоманда*, *Аргументы* и *Примечание*.

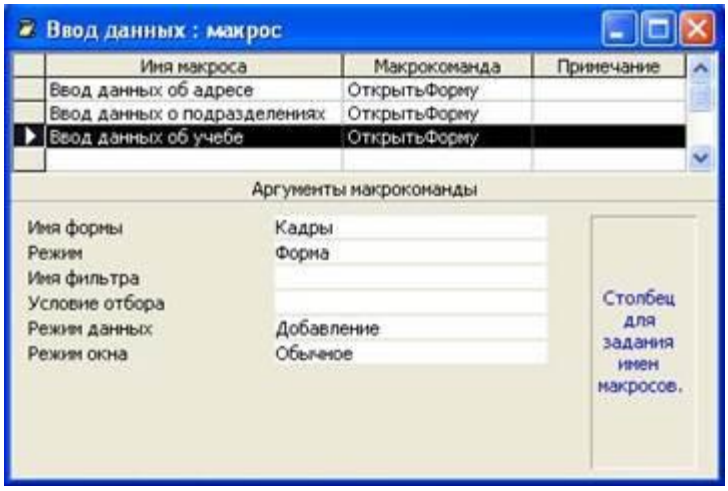
В столбце *Макрокоманда* перечисляются команды, подлежащие выполнению (например, открыть форму, распечатать отчет, выполнить запрос). Чтобы в ячейке столбца *Макрокоманда* задать нужную макрокоманду, ее следует выбрать из списка. Для построения исходного меню в ячейке *Макрокоманда* следует выбрать *Добавить меню.*

Образцы заполнения полей конструктора макросов для меню учета кадров представлены на рис. 12 а), б), в), г).





а) б)



в) г)

Примечание, содержит комментарии к командам. При выполнении макроса оно игнорируется, однако его заполнение делает текст макроса понятнее.

Выполнение каждой макрокоманды зависит от ее аргументов. Аргументы вводятся в специально отведенные для этого поля, расположенные в нижней части окна конструктора макроса. Аргументы могут вводиться с помощью клавиатуры, однако, по возможности, лучше выбирать их из списка, чтобы избежать некорректных значений. После заполнения окна конструктора макросов и аргумен-

тов макрокоманд, созданное пользовательское меню необходимо сохранить под именем *Главное меню*.

Далее следует открыть новое окно конструкторамакросов. Затем на вкладке *Конструктор* необходимонажать кнопку *Имена макросов*. В столбце *Имя макроса* ввести вручную название макроса. При создании макросов в полях *Имя макроса* следует повторять названия пунктов главного меню. Затем выбрать макрокоманду (например, открыть форму) и форму из окна аргументов макрокоманд, которую будет открывать созданный макрос – см. рис. 12 в) и г). Макросы создаются по числу макрокоманд в системе меню.

Для того чтобы, пользовательское меню отражалось вместо основного меню Access, нужно во всех формах и отчетах находясь в режиме *Конструктора* нажатьна кнопку *F4* либо вызвать контекстное меню правой кнопки мыши и нажать кнопку *Свойства*. Откроется окно свойств, где на вкладке *Другие* в пустом поле *Строка меню* нужно вручную набрать имя макроса пользовательского меню *Главное меню*.

Выполнение макроса может начинаться по команде пользователя, при вызове из другого макроса или процедуры обработки события, а также в ответ на событие в форме, отчете или элементе управления. Чтобы запустить макрос нужно нажать на кнопку *Восклицательный знак* либо на макросе *Главное меню* в области переходов > *Работа с базами данных* > *Надстройки* > *Команда меню.*

4. Перечень оборудования – персональный компьютер.

5. Порядок выполнения работы (Задания)

5.1. В соответствии с вариантом задания создать меню.

6. Содержание отчета

6.1. Наименование работы

6.2. Цель работы

6.3. Выполненные задания в соответствии с вариантом.

7. Список литературы

7.1. Федорова Г.Н. Основы проектирования баз данных: Учебное пособие для СПО / Г.Н. Федорова. - М.: Академия, 2018.- 220с. (*Основное печатное издание ОПИ-1*)

7.2. Кумскова, И. А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.

ПРАКТИЧЕСКАЯ (ЛАБОРАТОРНАЯ) РАБОТА № 11.
Создание файла проекта базы данных. Создание интерфейса входной формы. Использование исполняемого файла проекта БД, приемы создания и управления

1. Цель работы

2. Время выполнения работы – 2 часа.

3. Краткие теоретические сведения

Пусть имеется некоторая база данных, созданная в СУБД Microsoft Access. Файл базы данных имеет имя «db1.mdb«. Путь к файлу базы данных

E:\Programs\C_Sharp\WindowsFormsApplication1\db1.mdb

База данных имеет одну таблицу с именем «Tovar».

Необходимо осуществить подключение базы данных к Windows-приложению на языке C# средствами Microsoft Visual Studio 2010. Приложение должно быть реализовано как Windows Forms Application.

1. Создание приложения типа Windows Forms Application.

Запустить MS Visual Studio. **Создать приложение Windows Forms Application**.

2. Вызов мастера подключения.

Для доступа к файлу базы данных необходимо сделать его подключение к приложению. Это осуществляется путем вызова команды «Add New Data Source…» из меню «Data» (рис. 1) либо кликом на крайней левой кнопке с панели инструментов Data Source.

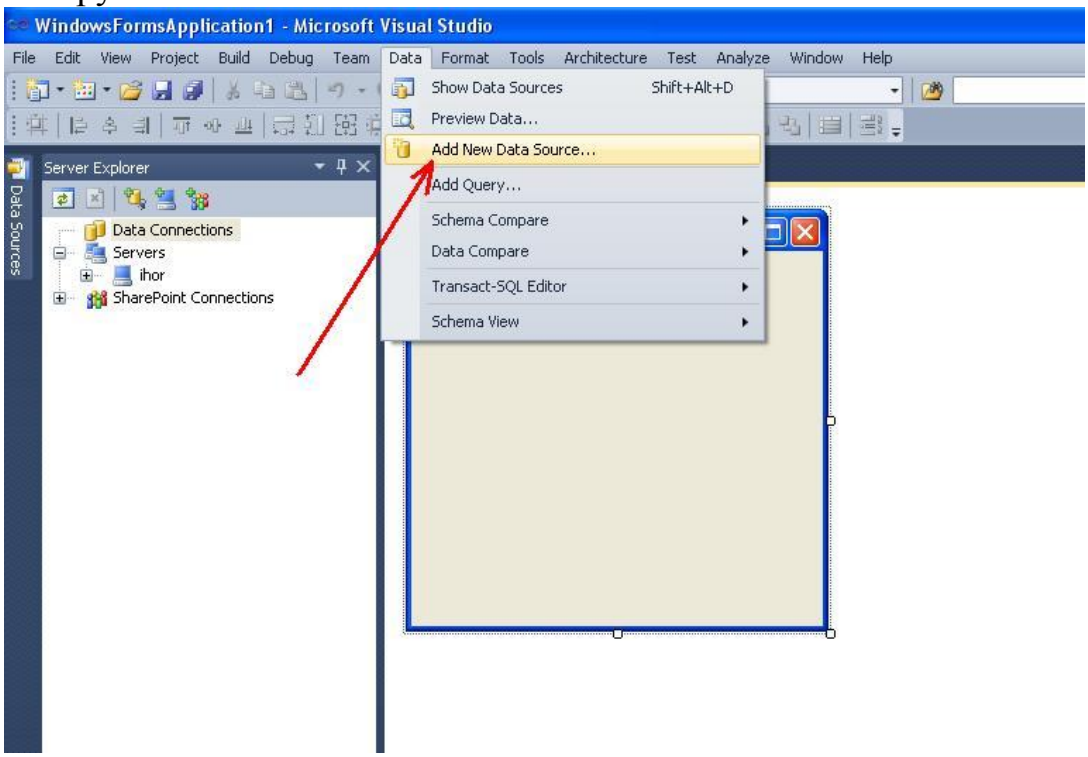


Рисунок. 1. Вызов мастера подключения к файлу базы данных

3. Выбор типа источника данных.

В результате откроется окно мастера для подключения к источнику данных которое изображено на рис. 2.

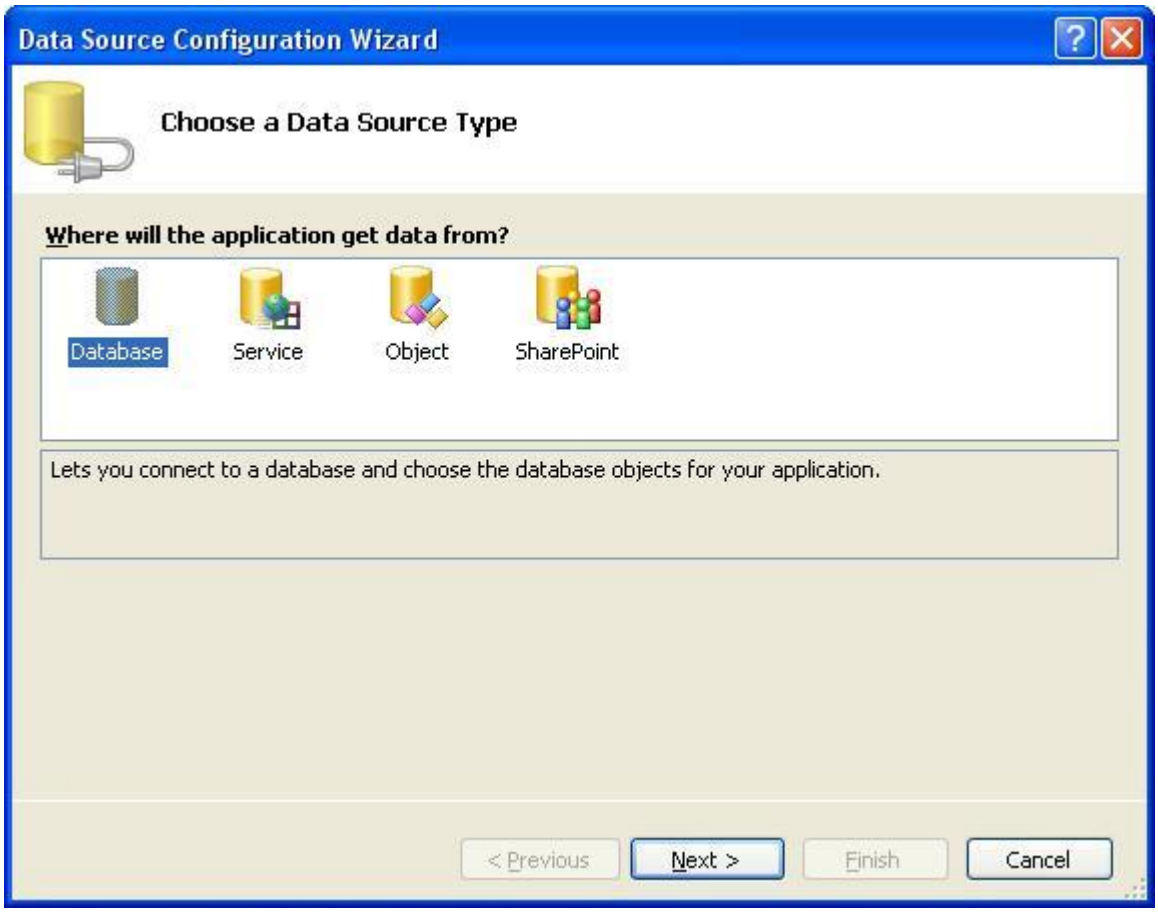


Рисунок. 2. Выбор типа подключения из которого приложение будет получать данные

В окне необходимо выбрать один из четырех возможных вариантов подключения к источнику данных. В MS Visual Studio существует четыре типа подключения к источникам данных:

- Database – подключение к базе данных и выбор объектов базы данных;
- Service – открывает диалоговое окно Add Service Reference позволяющее создать соединение с сервисом, который возвращает данные для вашей программы;
- Object – позволяет выбрать объекты нашего приложения, которые в дальнейшем могут быть использованы для создания элементов управления (controls) с привязкой к данным;
- Share Point – позволяет подключиться к сайту SharePoint и выбрать объекты для вашей программы.

В нашем случае выбираем элемент Database и продолжаем нажатием на кнопке Next.

4. Выбор модели подключения к базе данных.

Следующий шаг – выбор модели подключения к базе данных (рис. 3).

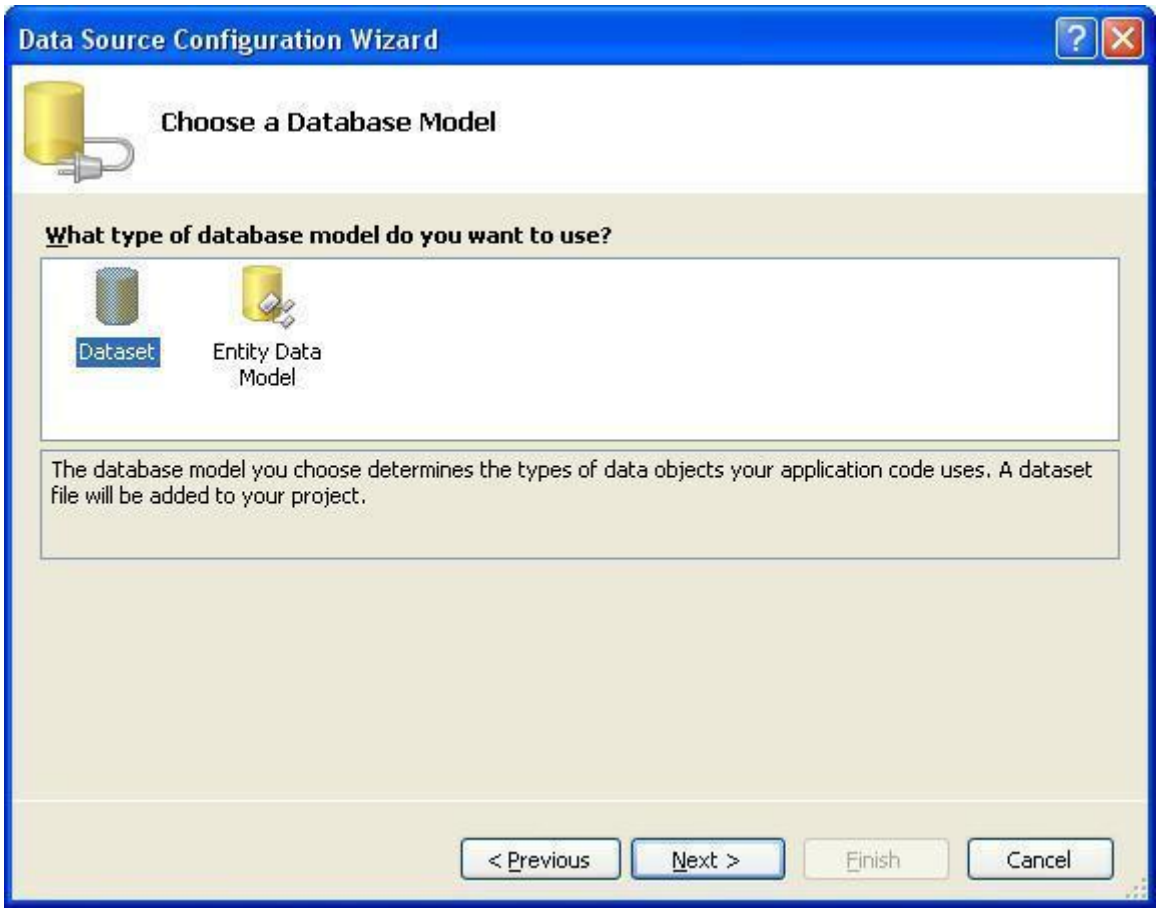


Рис. 3. Выбор модели подключения к базе данных

Система предлагает выбор одного из двух вариантов:

- модели данных на основе набора данных (Dataset);
- модели данных Entity, что означает, что система может сгенерировать модель данных из базы данных которой могут выступать сервера баз данных Microsoft SQL Server, Microsoftt SQL Server Compact 3.5 или Microsoft SQL Server Database File, либо создать пустую модель как отправную точку для визуального проектирования концептуальной модели с помощью панели инструментов.

В нашем случае выбираем тип модели данных DataSet.

5. Задание соединения с БД.

Следующим шагом мастера (рис. 4) есть выбор соединения данных которое должно использоваться приложением для соединения с базой данных.

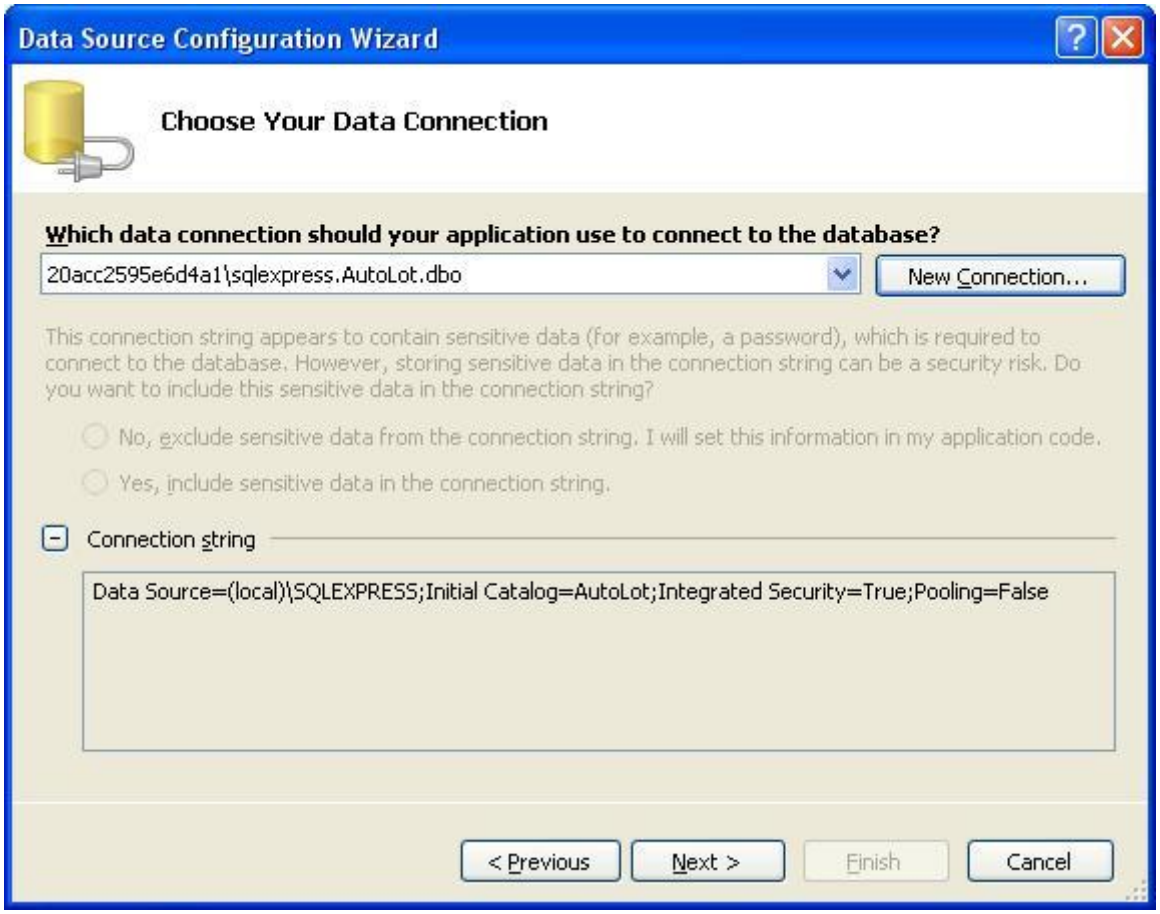


Рис. 4. Выбор соединения с базой данных

Для создания нового соединения необходимо выбрать кнопку «New Connection…«. В результате откроется окно «Add Connection» (рис. 5) в котором нужно добавить новое соединение Microsoft Access и выбрать маршрут к файлу базы данных.

В нашем случае поле «Data source» уже содержит нужный нам тип соединения «Microsoft Access Database File (OLE DB)«.

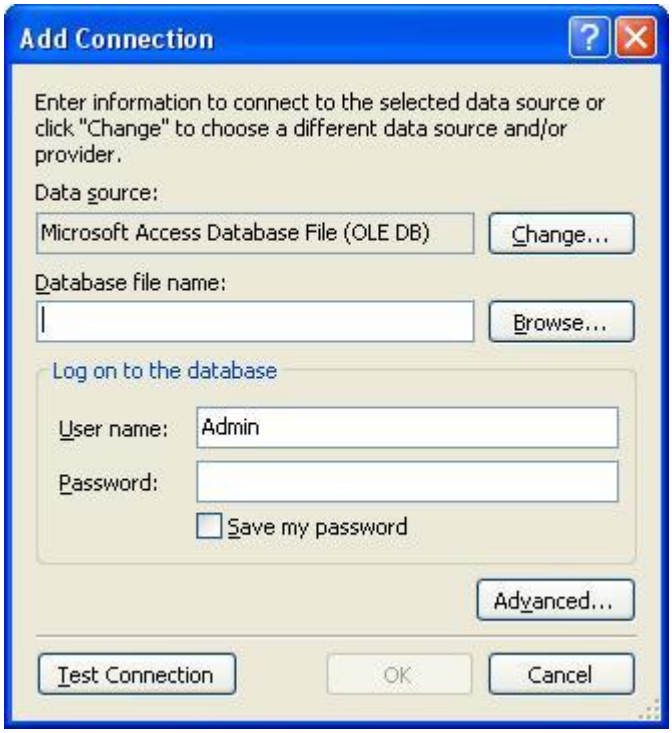


Рис. 5. Добавление нового соединения и выбор файла базы данных

Если нужно выбрать другую базу данных, то для этого используется кнопка «Change…«, которая открывает окно, изображенное на рисунке 6.

Рис. 6. Смена источника данных

В окне на рисунке 6 системой Microsoft Visual Studio будет предложено следующие виды источников данных:

- Microsoft Access Database File – база данных Microsoft Access;
- Microsoft ODBC Data Source – доступ к базе данных с помощью программного интерфейса ODBC (Open Database Connectivity);
- Microsoft SQL Server;
- Microsoft SQL Server Compact 3.5;
- Microsoft SQL Server Database File;
- Oracle Database – база данных Oracle.

Нажимаем кнопку «Browse…» и в открывшемся окне (рис. 7) «Add Connection» выбираем маршрут к файлу базы данных «db1.mdb«. Целесообразно

размещать файл базы данных в каталоге содержащим исполняемый модуль приложения.

Для проверки правильности установленного соединения можно воспользоваться кнопкой «Test Connection«.

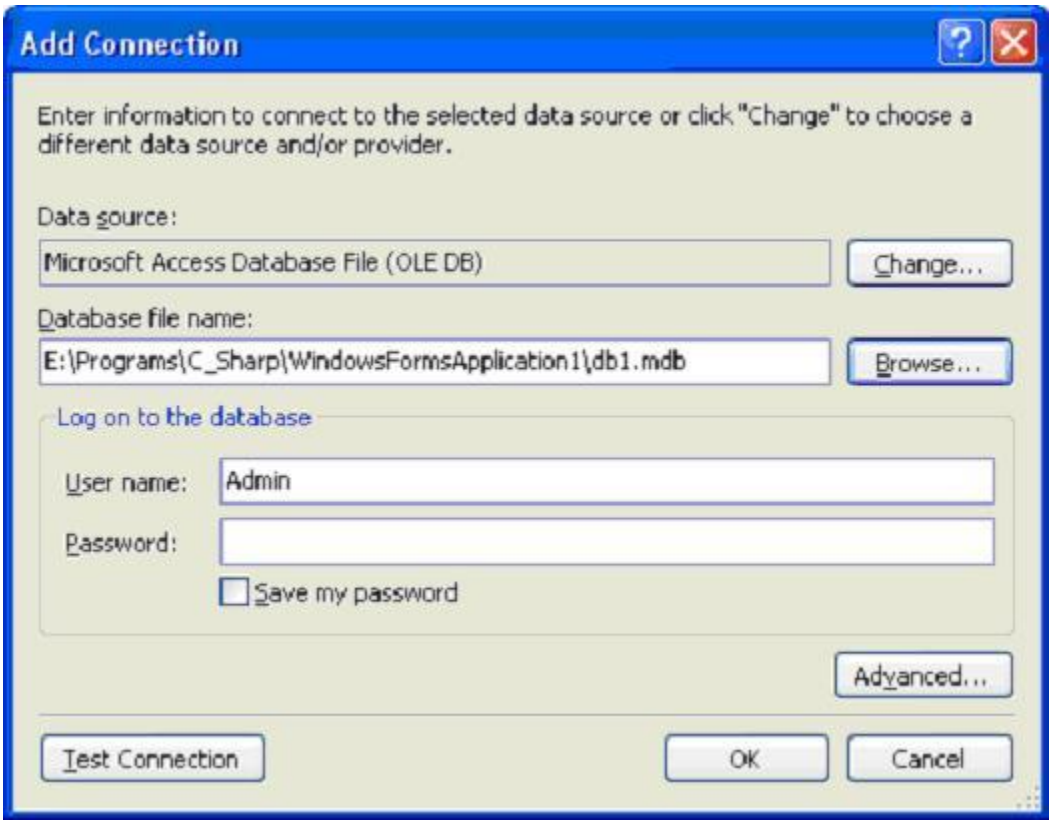


Рисунок 7. Окно «Add Connection» с выбранной базой данных «db1.mdb«

После нажатия на кнопке ОК система сгенерирует строку «Connection string» (рис. 8) который в дальнейшем будет использован для программного подключения к базе данных.

Кликаем на «Next» для продолжения работы мастера.

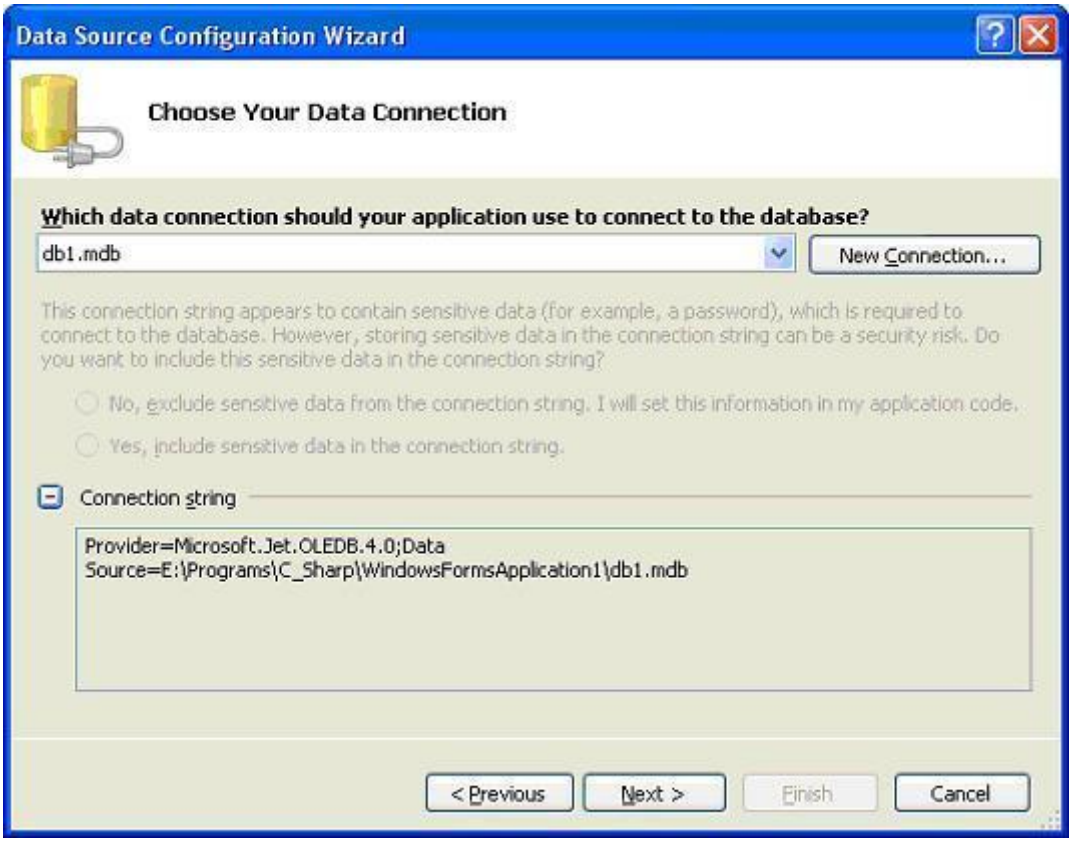


Рис. 8. Строка Connection string

После выбора Next система выдаст информационное окно следующего вида (рис. 9). Если выбрать «Да», то файл базы данных «db1.mdb» будет копироваться в выходной каталог приложения каждый раз при его запуске в среде MS Visual Studio. Как правило, это каталог, содержащий основные модули приложения. В нашем случае каталог

E:\Programs\C_Sharp\WindowsFormsApplication1\WindowsFormsApplication1

В этом каталоге размещаются все основные исходные модули проекта, например Program.cs (модуль, содержащий основную функцию WinMain()), Form1.cs (содержит исходный код обработки главной формы приложения) и другие.

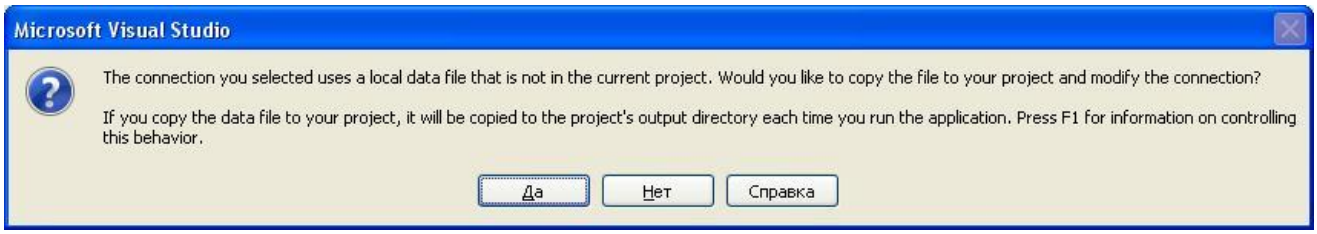


Рисунок 9. Окно добавления файла базы данных в проект

6. Формирование конфигурационного файла приложения.

После выбора кнопки «Next» мастера откроется следующее окно, в котором предлагается сохранить строку соединения в конфигурационный файл приложения (рис. 10).

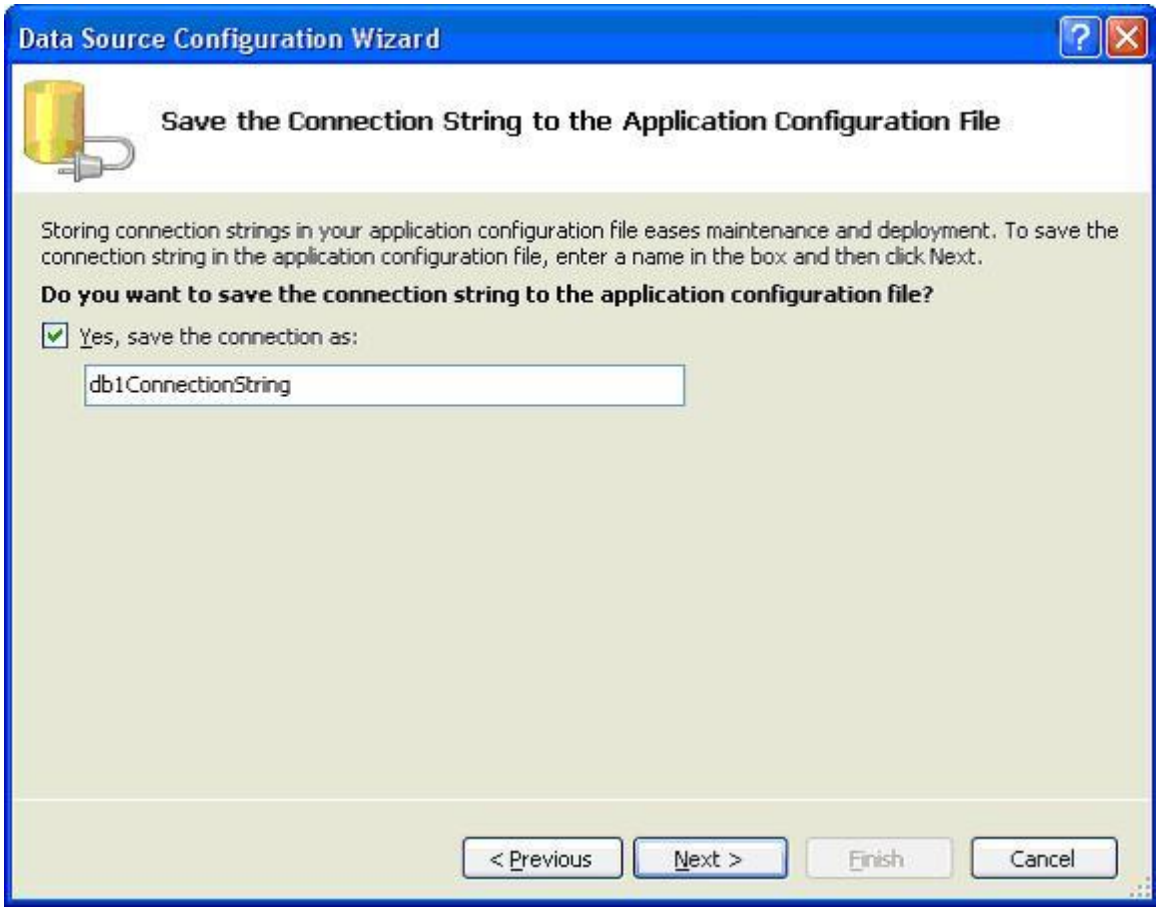


Рисунок 10. Предложение записи строки подключения к базе данных в конфигурационный файл приложения

Ничего не изменяем, оставляем все как есть (кликаем на Next).

⇑

7. Выбор объектов базы данных для использования в программе

Последнее окно мастера (рисунок 11) предлагает выбрать список объектов (таблиц, запросов, макросов, форм и т.д.), которые будут использоваться в наборе данных. Как правило выбираем все таблицы базы данных. В нашем примере база данных содержит всего одну таблицу с именем Tovar.

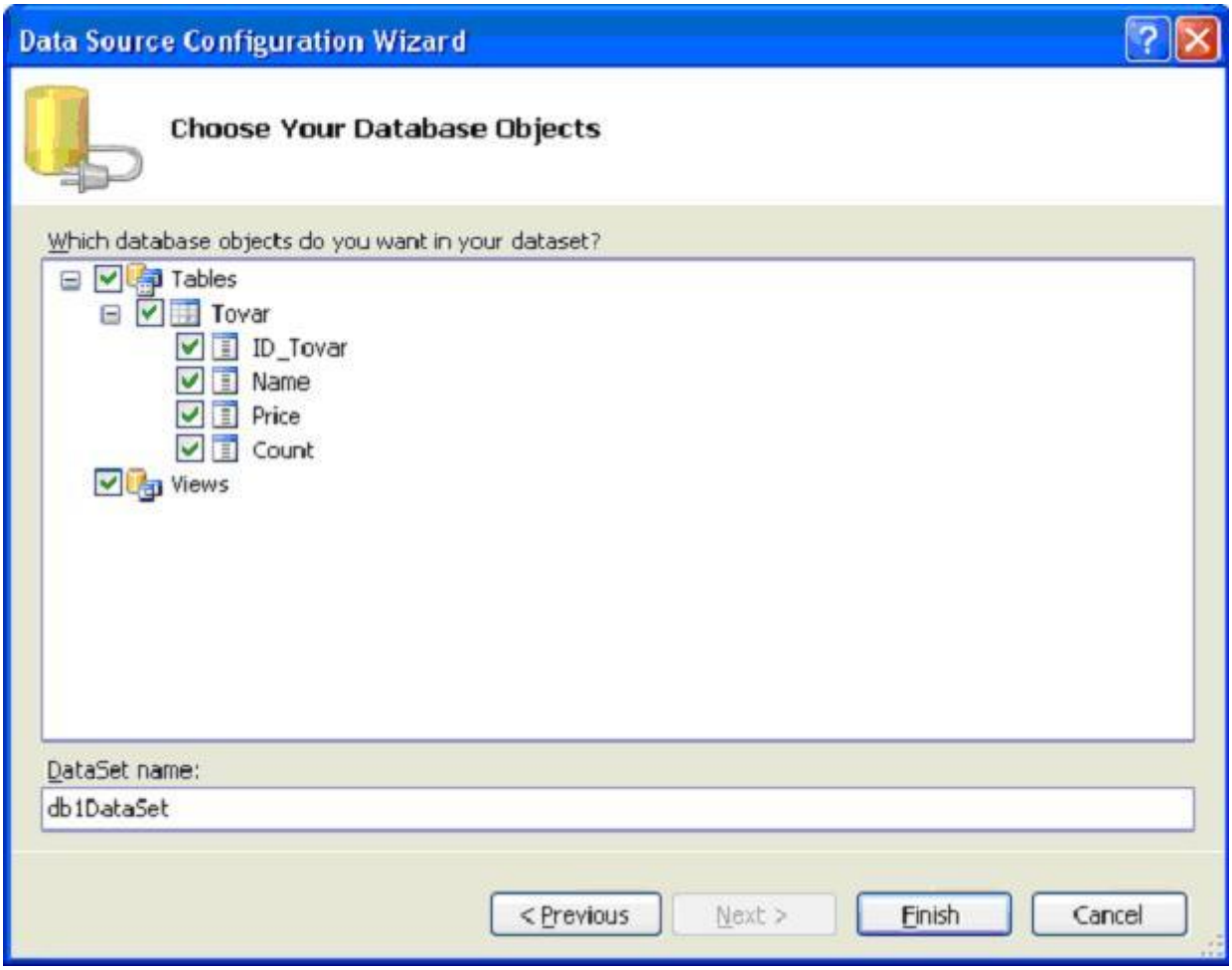


Рисунок 11. Выбор объектов базы данных, которые будут использоваться в данном наборе данных

После выбора кнопки «Finish» заканчиваем работу с мастером подключения. Теперь база данных подключена к приложению и будет автоматически подключаться при его запуске или при его проектировании в MS Visual Studio.

8. Что же изменилось в программе после выполнения мастера?

Если выбрать панель Data Source (рисунок 12), то можно увидеть, как подключен набор данных с именем db1Dataset в котором есть таблица с именем Tovar.

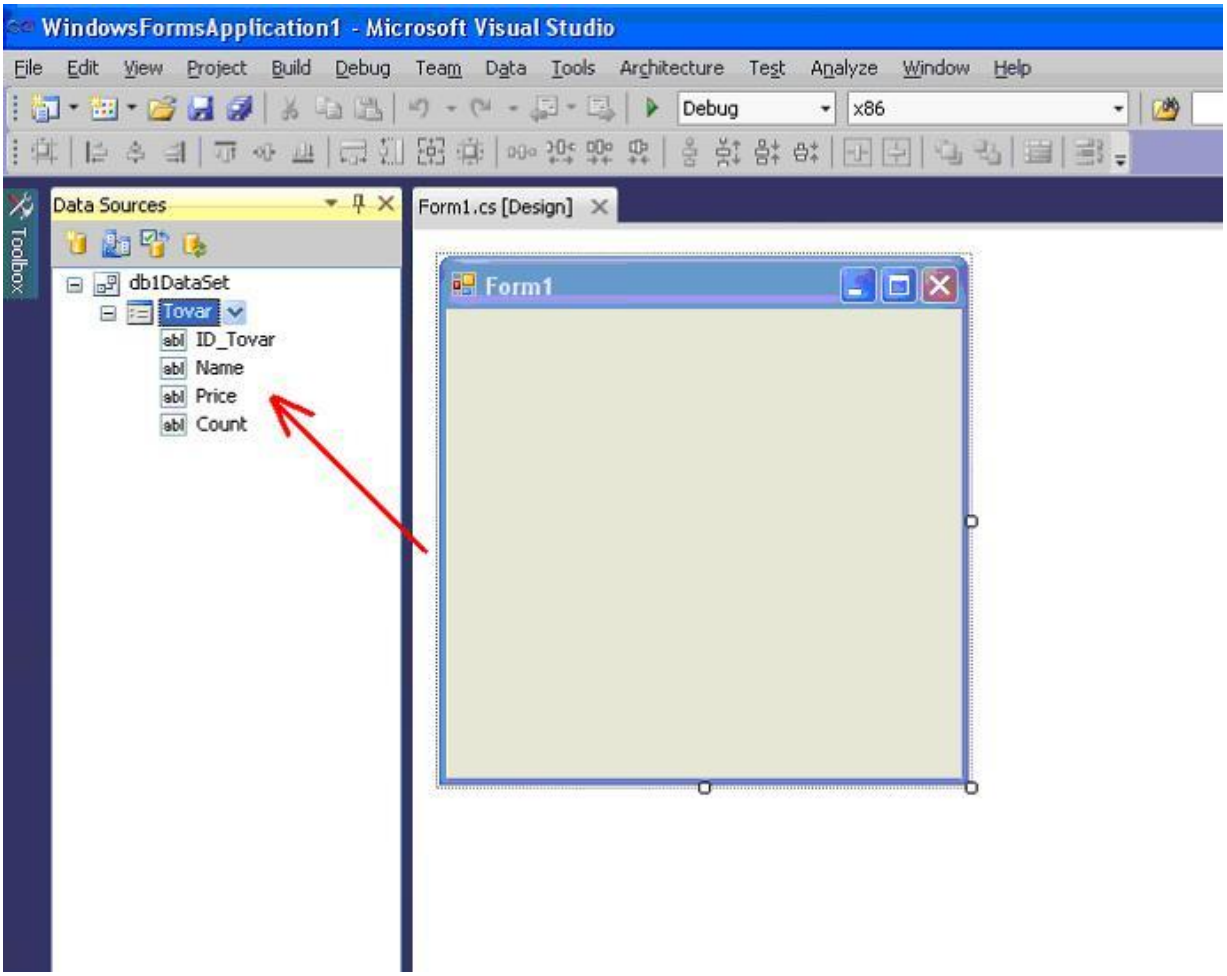


Рисунок 12. Окно DataSources содержит подключение к базе данных

Точно также можно увидеть изменентия в панели Server Explorer (рисунок 13), где появилась база данных «db1.mdb» с таблицей Tovar и ее полями. Приложение может подключать не только одну, но и несколько баз данных.

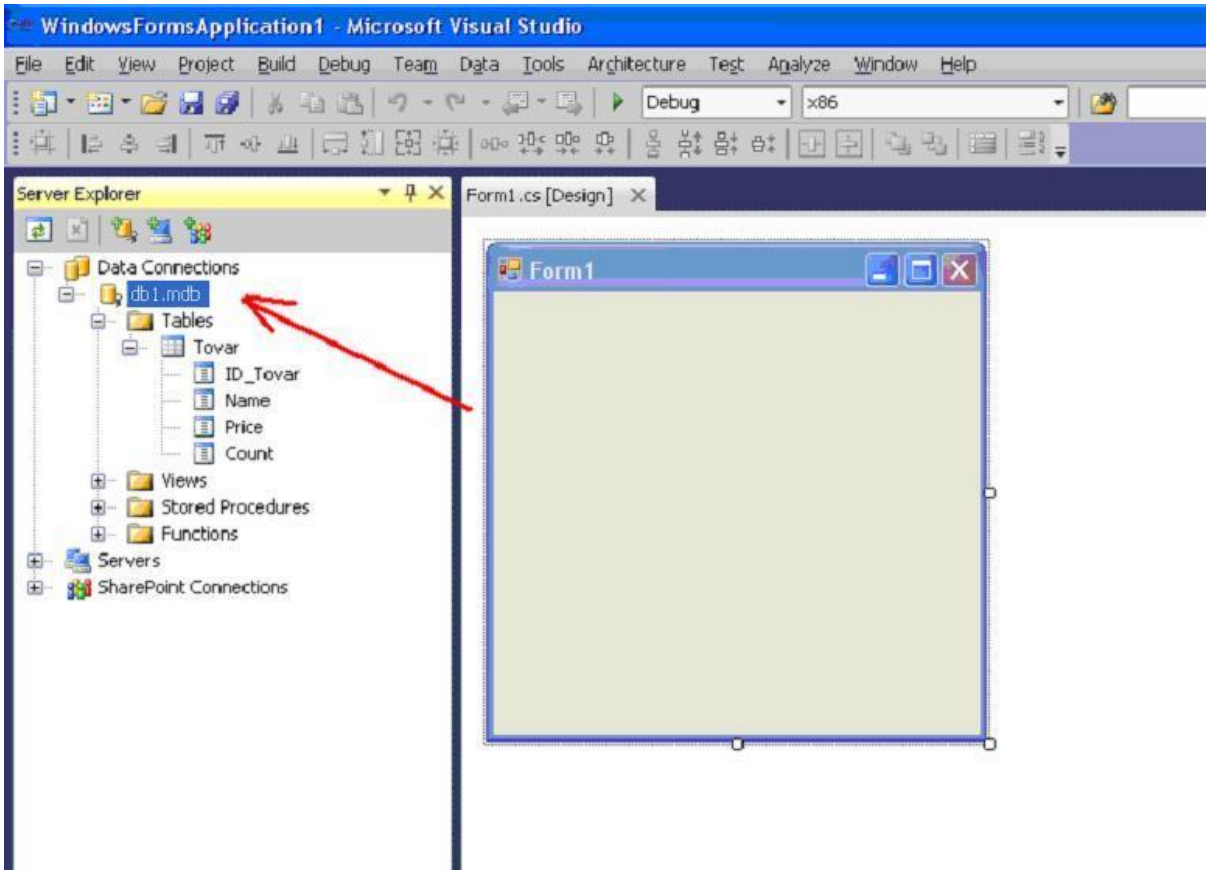


Рис. 13. Окно приложения с изменениями в панели Server Explorer

9. Подключение методов оперирования базой данных.

Для того, чтобы использовать методы, которые будут работать с базой данных MS Access (и не только MS Access), необходимо подключить пространство имен System.Data.OleDb.

Для этого в основной форме (Form1.cs) в Solution Explorer выбираем режим просмотра кода (View Code) из контекстного меню (рис. 14) и вначале файла добавляем следующую строку:

```
using System.Data.OleDb;
```

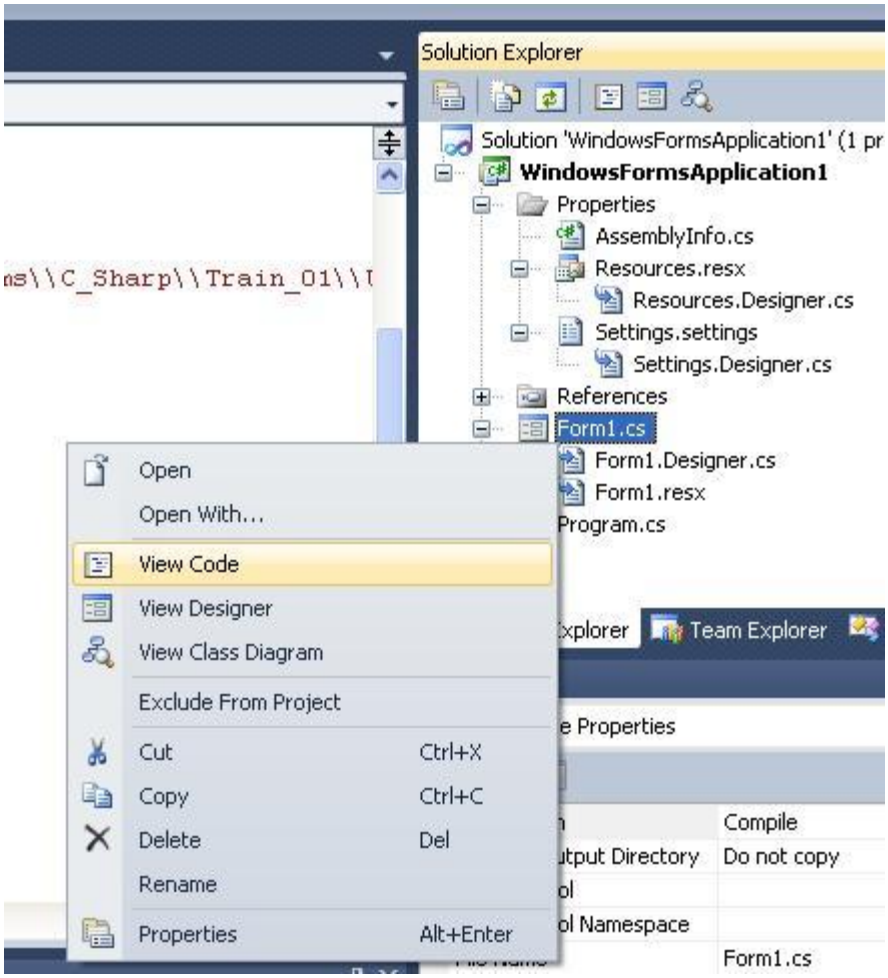


Рисунок 14. Вызов программного кода главной формы приложения (Form1.cs) с помощью Solution Explorer

Общий вид верхней части файла Form1.cs будет следующим:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.OleDb;
```

На этом этапе подключение к базе данных db1.mdb выполнено. Дальнейшими шагами есть создание программного кода для оперирования данными в базе данных.

4. Перечень оборудования – персональный компьютер.

5. Порядок выполнения работы (Задания)

5.1. В соответствии с вариантом задания выполнить подключение базы данных к Windows-приложению.

6. Содержание отчета

6.1. Наименование работы

6.2. Цель работы

6.3. Выполненные задания в соответствии с вариантом.

7. Список литературы

7.1. Федорова Г.Н. Основы проектирования баз данных: Учебное пособие для СПО / Г.Н. Федорова. - М.: Академия, 2018.- 220с. (*Основное печатное издание ОПИ-1*)

7.2. Кумскова, И. А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.

ПРАКТИЧЕСКАЯ (ЛАБОРАТОРНАЯ) РАБОТА № 12.
Создание формы. Управление внешним видом формы

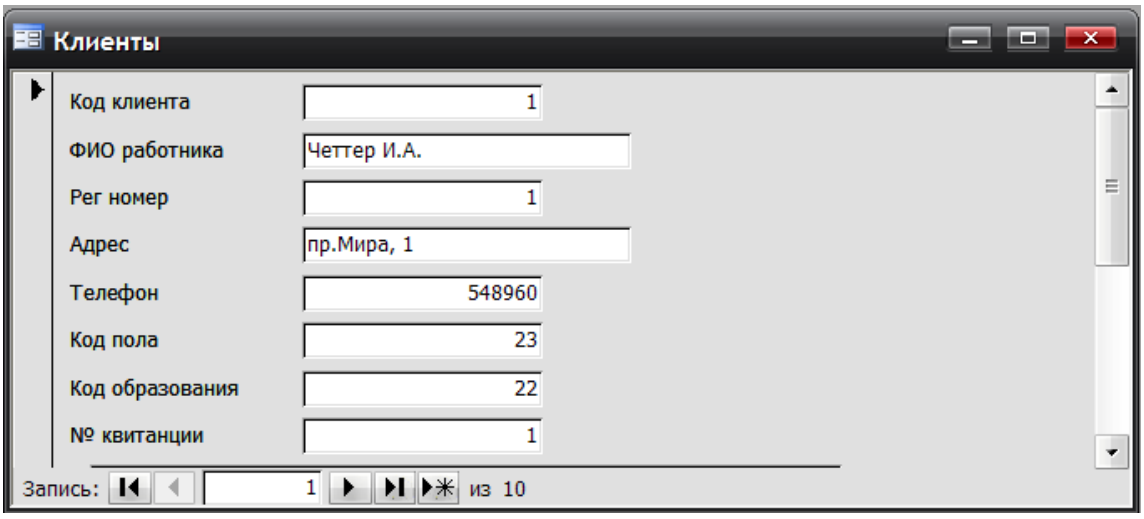
1. Цель работы
2. Время выполнения работы – 2 часа.
3. Краткие теоретические сведения

1. ***Создадим форму с помощью автоформы*** – **Клиенты**. Для этого будем использовать таблицу **Клиенты**.

1.1. В окне базы данных выберем вкладку Таблицы.

1.2. Выберем таблицу **Клиенты** и откроем в режиме таблица.

1.3. Нажмем кнопку раскрытия списка рядом с кнопкой **Новый объект** на панели инструментов и выберем элемент **Автоформа**.



2. ***Создадим форму при помощи мастер*** – **Работодатели**. Для этого будем использовать таблицу **Работодатели**.

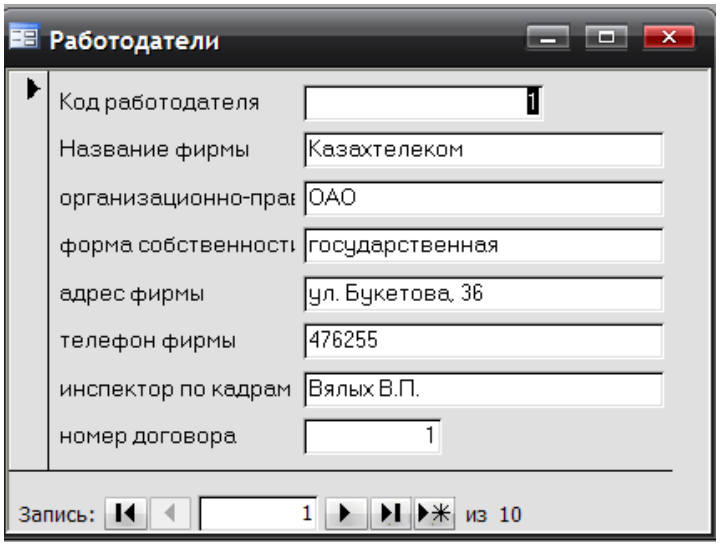
2.1. В окне базы данных выберем вкладку **Формы.**

2.2. Нажмем кнопку **Создать.**

2.3. В диалоговом окне **Новая форма** выберем пункт **Мастер форм**.

2.4. Выберем таблицу **Работодатели** и выберем поля, которые будут включаться в форму.

2.5. Выберем внешний вид форм – **В один столбец**, требуемый стиль – **Стандартный**.

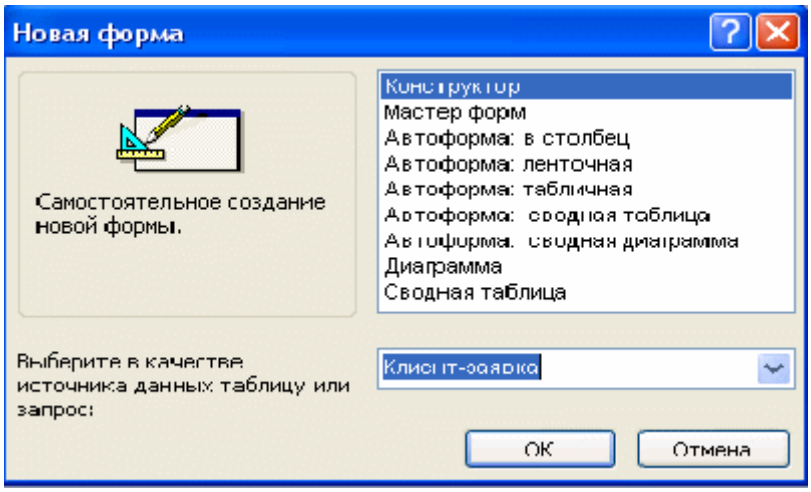


3. ***Создадим форму без помощи мастер*** – **Клиент-заявка**. Для этого будем использовать запрос **Клиент-заявка.**

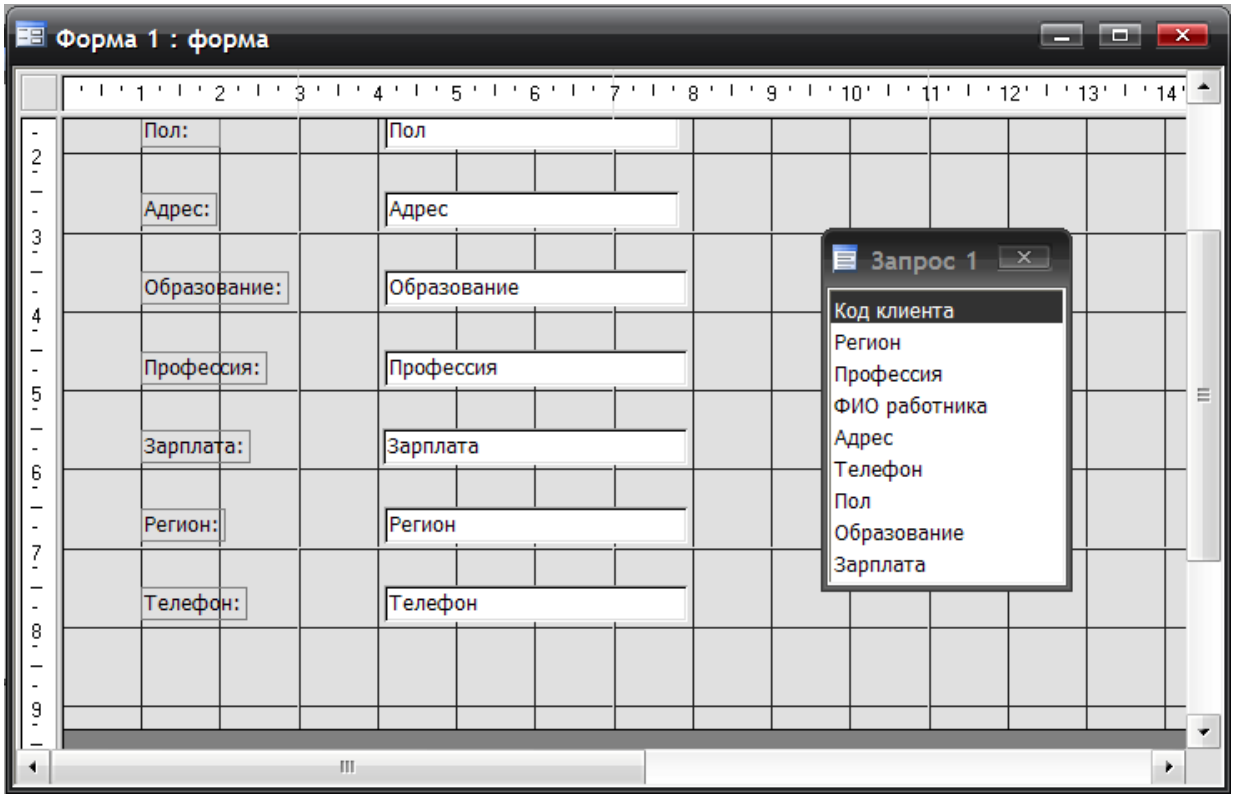
3.1. В окне базы данных выберем вкладку **Формы.**

3.2. Нажмем кнопку **Создать.**

3.3. В диалоговом окне **Новая форма** выберем пункт **Конструктор**.



3.4. В область данных переносим необходимые поля.



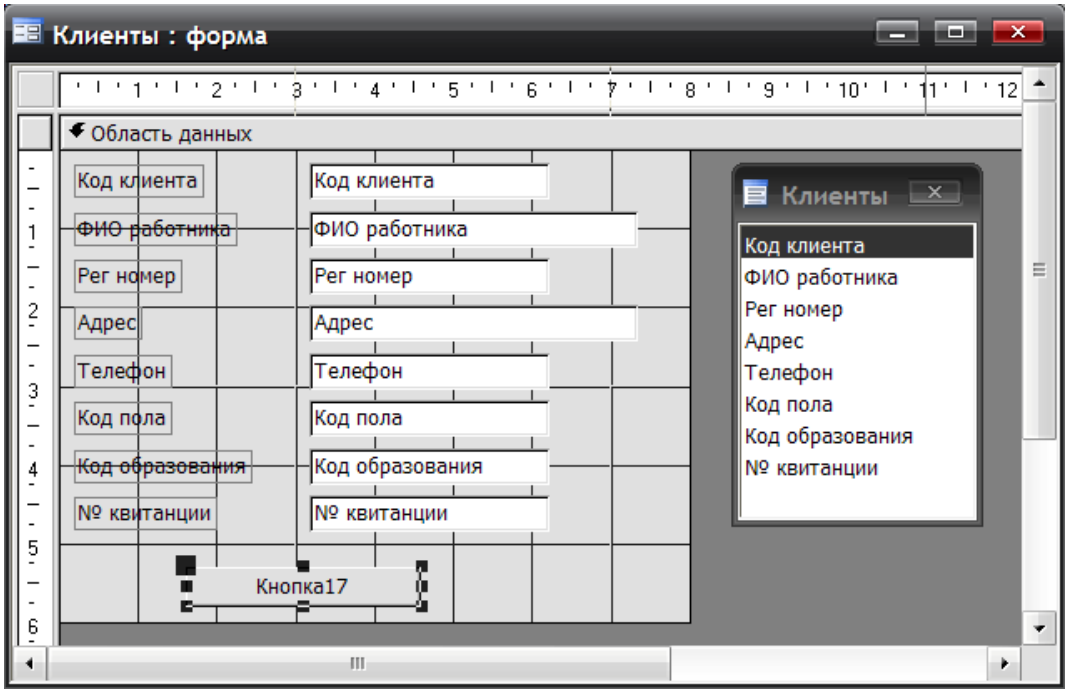
4. Добавим элемент управления в форму **Клиенты** – *кнопку просмотра таблицы.*

4.1. Откроем форму **Клиенты** в режиме конструктора.

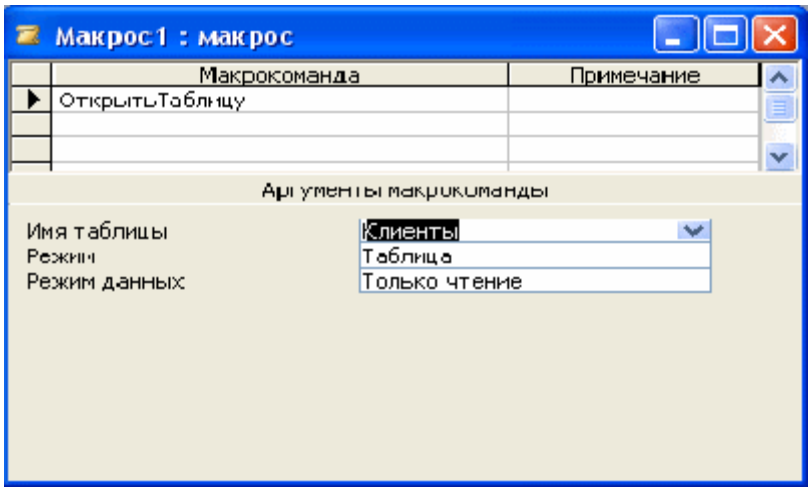
4.2. Щёлкнем мышкой по кнопке Панели инструментов на элементе **Кнопка.**

4.3. Пользуясь разметкой экрана формы, переместим курсор мыши туда, где необходимо разместить эту кнопку.

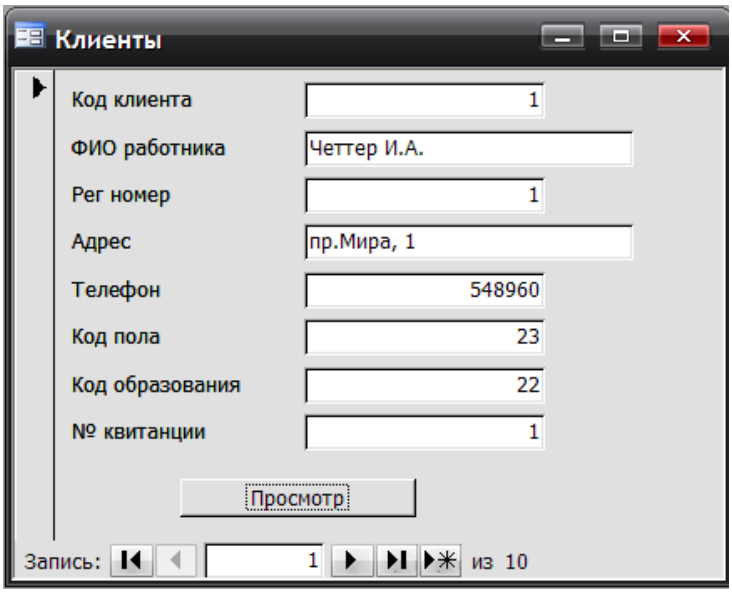
4.4. Нажмем на кнопку мыши, растянем рамку кнопки до нужного размера.



4.5. Щёлкнем мышкой по кнопке Панели инструментов на элементе **Свойства** и отредактируем поля. На нажатие кнопки привяжем макрос открытия таблицы **Клиенты**.



4.6. Сохраняем изменения формы и смотрим что получилось.



При нажатии на элемент кнопку **Просмотр**, выходит соответствующая таблица на экран.

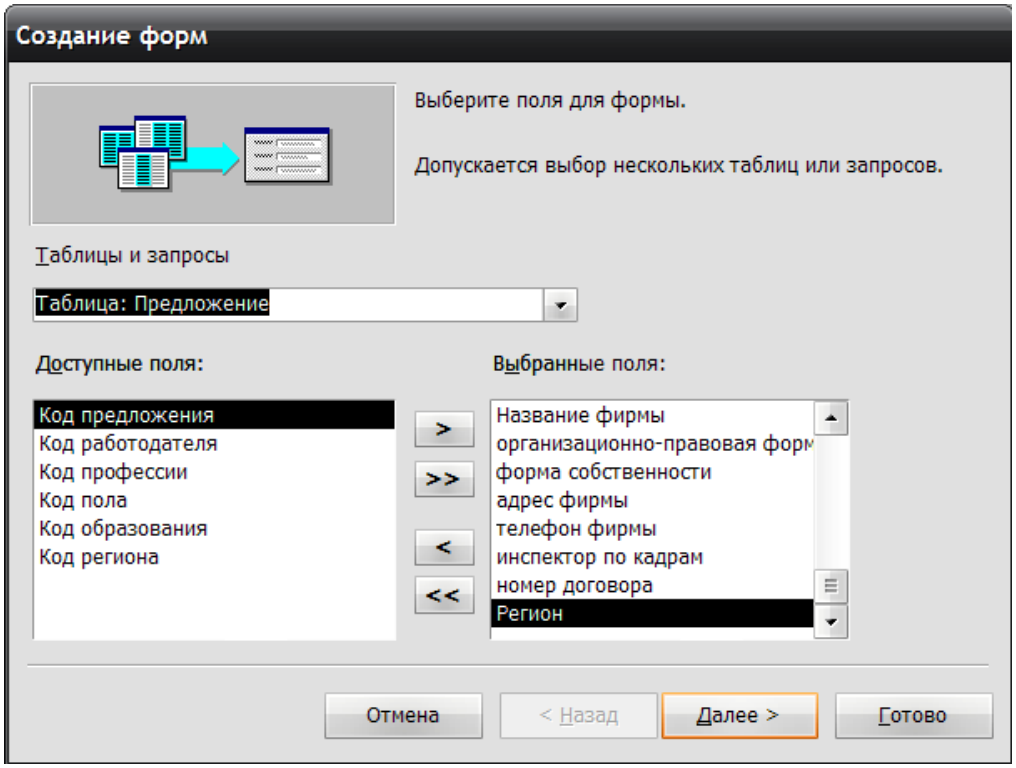
Код клиента	ФИО работника	Рег номер	Адрес	Телефон	Код пола	Код образования	№ квитанции
1	Четтер И.А.	1	пр.Мира, 1	548960	23	22	1
2	Хайдаев Е.С.	2	ул. Гагарина, 10	784897	22	23	2
3	Щановская Л.А.	3	ул. Лесная, 45	746982	23	22	3
4	Тимошенко Г.Д.	4	пр. Мира, 28	697545	23	22	4
5	Баймакишева М.	5	ул. Гагарина, 11	478213	23	23	5
6	Щелкунов А.Н.	6	ул. К.Лебнихта, 8	387190	22	23	6
7	Игнатьев Я.О.	7	ул.Я.Гашека, 6	401874	22	22	7
8	Климов А.Е.	8	пр. Мира, 18	495645	22	23	8
9	Сапрыкин А.А.	9	ул. К.Лебнихта, 1	356465	22	22	9
10	Шалашова Л.Н.	10	ул. Гагарина, 67	342875	23	23	10

5. ***Создадим многотабличную форму с помощью мастера*** **– Работодатели-предложение**. Для этого будем использовать таблицы **Работодатели, Предложение.**

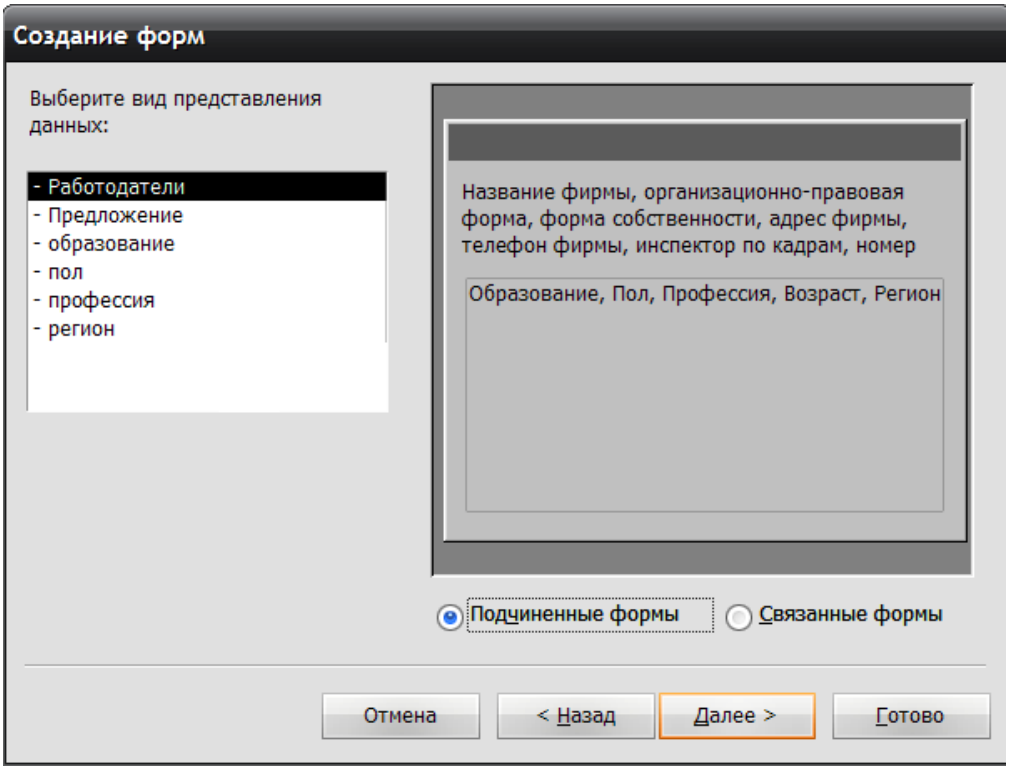
5.1. Нажмем кнопку **Создать.**

5.2. В диалоговом окне **Новая форма** выберем пункт **Мастер форм**.

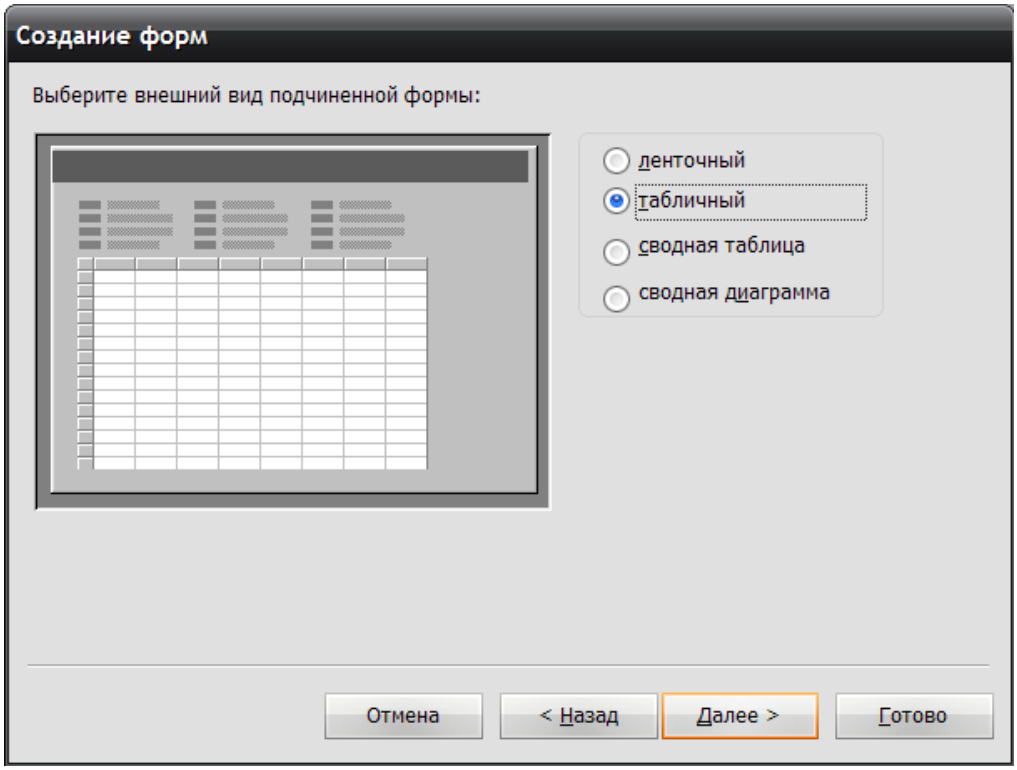
5.3. Выберем последовательно таблицы **Работодатели** и **Предложение** и выберем поля, которые будут включаться в форму.



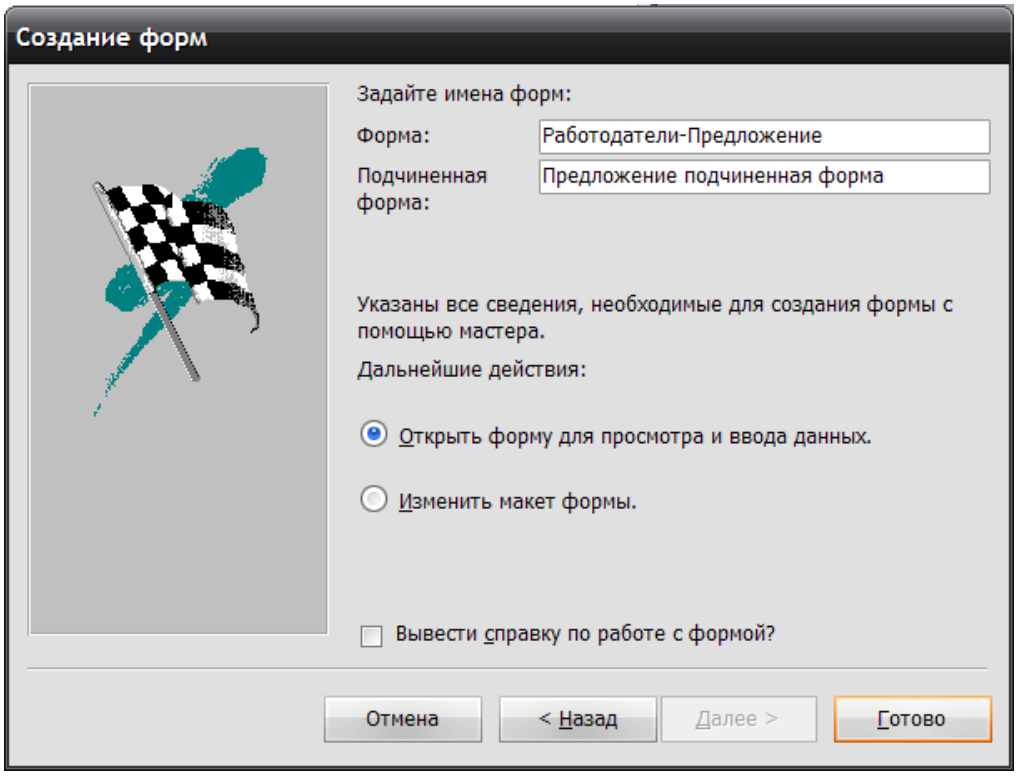
5.4. Выберем тип представления данных: **Подчиненная форма**.



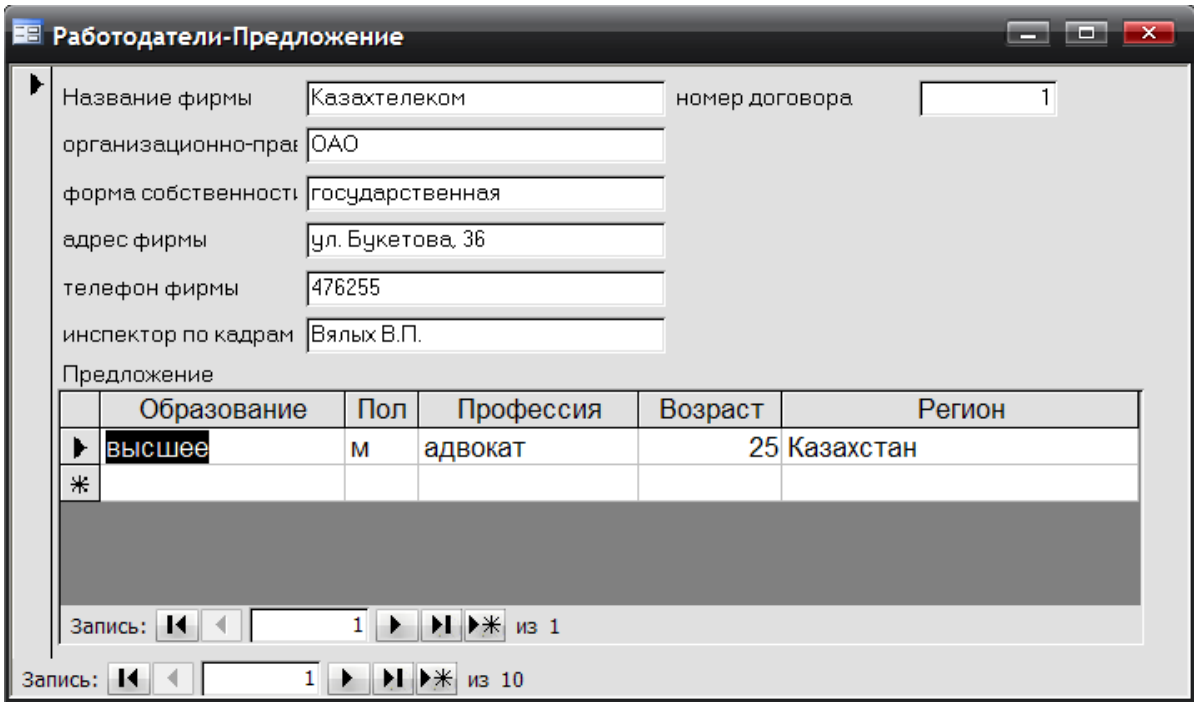
5.5. Выберем внешний вид подчиненной формы - **табличный.**



5.6. Сохраним форму под именем **Работодатели-предложение.**



Результат выполнения формы **Работодатели-предложение**.



4. Перечень оборудования – персональный компьютер.

5. Порядок выполнения работы (Задания)

5.1.В соответствии с вариантом задания выполнить построение форм.

6. Содержание отчета

6.1. Наименование работы

6.2. Цель работы

6.3. Выполненные задания в соответствии с вариантом.

7. Список литературы

7.1. Федорова Г.Н. Основы проектирования баз данных: Учебное пособие для СПО / Г.Н. Федорова. - М.: Академия, 2018.- 220с. (*Основное печатное издание ОПИ-1*)

7.2. Кумскова, И. А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.

Список используемой литературы

3.2.1. Печатные издания

1. Федорова Г.Н. Основы проектирования баз данных: Учебное пособие для СПО / Г.Н. Федорова. - М.: Академия, 2018.- 220с. (*Основное печатное издание ОПИ-1*)
2. Кумскова, И. А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.

3.2.2. Электронные издания (электронные ресурсы)

1. Голицына О.Л. Основы проектирования баз данных: Учебное пособие для СПО / О.Л. Голицына, Т.Л. Партыка, И.И. Полпов. - М.: Форум, 2019.- 416с. ЭБС Знаниум: https://znanium.com/
2. Илюшечкин В.М. Основы использования и проектирования баз данных: Учебник для СПО / В.М. Илюшечкин.- М.: Юрайт, 2019.- 214с. - ЭБС Юрайт: https://urait.ru/ebs/.
3. Стружкин Н.П. Базы данных. Проектирование. Практикум: Учебное пособие для СПО / Н.П. Стружкин, В.В. Годин.- М.: Юрайт, 2019.- 291с. ЭБС Юрайт: https://urait.ru/ebs/.
4. Агальцов, В.П. Базы данных. В 2-х кн. Кн. 2. Распределенные и удаленные базы данных [Электронный ресурс]: учебник / В.П. Агальцов. - М.: ФОРУМ: ИНФРА-М, 2018. - 271 с. - ЭБС «Znanium.com» -http://znanium.com/
5. Гагарина, Л.Г. Разработка и эксплуатация автоматизированных информационных систем [Электронный ресурс]: учебное пособие / Л.Г. Гагарина. - М.: ФОРУМ: Инфра-М, 2018. - 384 с. - ЭБС «Znanium.com» - http://znanium.com/

1. http://digital-edu.ru – справочник образовательных ресурсов «Портал цифрового образования».
2. http://fcior.edu.ru – Федеральный центр информационно-образовательных ресурсов (ФЦИОР).
3. http://school-collection.edu.ru – Единая коллекция цифровых образовательных ресурсов.
4. http://window.edu.ru – Единое окно доступа к образовательным ресурсам Российской Федерации.
5. http://www.intuit.ru – открытые Интернет-курсы «Интуит».

3.2.3. Дополнительные источники

1. Осипов Д.Л. Технологии проектирования баз данных. М: ДМК-Пресс, 2019 г. – 498 с..
2. Мартишин, С.А. Базы данных. Практическое применение СУБД SQL и NoSQL-типа для проектирования информационных систем [Электронный ресурс]: учебное пособие / С.А. Мартишин, В.Л. Симонов, М.В. Храпченко. - М.:

ФОРУМ: ИНФРА-М, 2018. - 368 с. - ЭБС «Znanium.com» http://znanium.com

3. Немцова, Т.И. Практикум по информатике. Компьютерная графика и web-дизайн [Электронный ресурс]: учебное пособие / Т.И. Немцова, Ю.В. Назарова; под ред. Л.Г. Гагариной. - М: ФОРУМ: ИНФРА-М, 2019. - 288 с.- ЭБС «Znanium.com» - http://znanium.com

4. Тараканов, О.В. Базы данных [Электронный ресурс]: учебник / Л.И. Шустова, О.В. Тараканов. - М.: ИНФРА-М, 2018. - 304 с.- ЭБС «Znanium.com» - http://znanium.com

5. Тарасов, С.В. СУБД для программиста. Базы данных изнутри [Электронный ресурс] / С.В. Тарасов. - М.: СОЛОН-ПРЕСС, 2015. - 320 с. - ЭБС «IPRbooks» - http://www.iprbookshop.ru.