

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Саратовский государственный технический университет  
имени Гагарина Ю.А.»

Филиал федерального государственного бюджетного образовательного  
учреждения высшего образования  
«Саратовский государственный технический университет  
имени Гагарина Ю.А.» в г. Петровске



УТВЕРЖДАЮ  
Директор филиала СГТУ  
имени Гагарина Ю.А. в г.Петровске  
Е.А.Бесшапошникова  
«30» июня 2025 г.

## МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ РАБОТ

по дисциплине

ОП.08 «Основы проектирования баз данных»

направление подготовки

09.02.07 «Информационные системы и программирование»

Методические указания рассмотрены  
на заседании предметной (цикловой) комиссии  
общепрофессиональных дисциплин и  
профессиональных модулей  
«16» июня 2025 года, протокол №13

Председатель ПЦК /Ю.А.Табарова/

Петровск 2025

### **Пояснительная записка**

Методические указания по выполнению практических работ подготовлены на основе рабочей программы учебной дисциплины ОП.08 «Основы проектирования баз данных», разработанной на основе ФГОС СПО по специальности 09.02.07 «Информационные системы и программирование» и соответствующих общих (ОК) компетенций:

ОК 01. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам.

ОК 02. Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности.

ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста.

ОК 09. Пользоваться профессиональной документацией на государственном и иностранном языках

В результате освоения дисциплины обучающийся должен **знать**:

- основы теории баз данных;
- модели данных;
- особенности реляционной модели и проектирование баз данных;
- изобразительные средства, используемые в ER- моделировании;
- основы реляционной алгебры;
- принципы проектирования баз данных;
- обеспечение непротиворечивости и целостности данных;
- средства проектирования структур баз данных;
- язык запросов SQL

В результате освоения дисциплины обучающийся должен **уметь**:

- проектировать реляционную базу данных;
- использовать язык запросов для программного извлечения сведений из баз данных

Содержание практических занятий определено рабочей программой и тематическим планированием, соответствует теоретическому материалу изучаемых разделов учебной дисциплины.

Объем практических занятий по дисциплине определяется учебным планом по данной специальности.

Продолжительность практического занятия – 2 академических часа. Перед проведением практического занятия преподавателем организуется инструктаж, а по его окончании – обсуждение итогов.

Комплект методических указаний по выполнению практических работ по дисциплине ОП.08 «Основы проектирования баз данных» содержит 40 практических занятий.

**Перечень практических работ  
по дисциплине ОП.08 «Основы проектирования баз данных»**

**ПРАКТИЧЕСКАЯ РАБОТА № 1**

Тема: Модели данных

**ПРАКТИЧЕСКАЯ РАБОТА № 2**

Тема: Операции реляционной алгебры.

**ПРАКТИЧЕСКАЯ РАБОТА № 3**

Тема: Проектирование баз данных. Сбор сведений и системный анализ предметной области

**ПРАКТИЧЕСКАЯ РАБОТА № 4**

Тема: Проектирование баз данных. Сбор сведений и системный анализ предметной области

**ПРАКТИЧЕСКАЯ РАБОТА № 5**

Тема: Разработка концептуальной ER-модели предметной области

**ПРАКТИЧЕСКАЯ РАБОТА № 6**

Тема: Разработка концептуальной ER-модели предметной области.

**ПРАКТИЧЕСКАЯ РАБОТА № 7**

Тема: Создание логической реляционной модели базы данных

**ПРАКТИЧЕСКАЯ РАБОТА № 8**

Тема: Создание логической реляционной модели базы данных

**ПРАКТИЧЕСКАЯ РАБОТА № 9**

Тема: Нормализация баз данных

**ПРАКТИЧЕСКАЯ РАБОТА № 10**

Тема: Нормализация баз данных

**ПРАКТИЧЕСКАЯ РАБОТА № 11**

Тема: Преобразование реляционной базы данных в сущности и связи

**ПРАКТИЧЕСКАЯ РАБОТА № 12**

Тема: Преобразование реляционной базы данных в сущности и связи

**ПРАКТИЧЕСКАЯ РАБОТА № 13**

Тема: Задание ключей. Создание основных объектов БД

**ПРАКТИЧЕСКАЯ РАБОТА № 14**

Тема: Создание проекта БД. Создание БД. Редактирование и модификация таблиц

**ПРАКТИЧЕСКАЯ РАБОТА № 15**

Тема: Создание проекта БД. Создание БД. Редактирование и модификация таблиц

**ПРАКТИЧЕСКАЯ РАБОТА № 16**

Тема: Создание ключевых полей. Задание индексов. Установление и удаление связей между таблицами

**ПРАКТИЧЕСКАЯ РАБОТА № 17**

Тема: Редактирование, добавление и удаление записей в таблице. Применение логических условий к записям. Открытие, редактирование и пополнение табличного файла

## **ПРАКТИЧЕСКАЯ РАБОТА № 18**

Тема: Проведение сортировки и фильтрации данных. Поиск данных по одному и нескольким полям. Поиск данных в таблице.

## **ПРАКТИЧЕСКАЯ РАБОТА № 19**

Тема: Создание запросов

## **ПРАКТИЧЕСКАЯ РАБОТА № 20**

Тема: Создание формы. Управление внешним видом формы

## **ПРАКТИЧЕСКАЯ РАБОТА № 21**

Тема: Создание кнопочной формы

## **ПРАКТИЧЕСКАЯ РАБОТА № 22**

Тема: Создание меню различных видов. Модификация и управление меню. Макросы и модули

## **ПРАКТИЧЕСКАЯ РАБОТА № 23**

Тема: Создание меню различных видов. Модификация и управление меню. Макросы и модули

## **ПРАКТИЧЕСКАЯ РАБОТА № 24**

Тема: Обращение к объектам БД с помощью встроенного языка программирования VBA

## **ПРАКТИЧЕСКАЯ РАБОТА № 25**

Тема: Обращение к объектам БД с помощью встроенного языка программирования VBA

## **ПРАКТИЧЕСКАЯ РАБОТА № 26**

Тема: Создание интерфейса входной формы

## **ПРАКТИЧЕСКАЯ РАБОТА № 27**

Тема: Создание файла проекта базы данных. Использование исполняемого файла проекта БД, приемы создания и управления

## **ПРАКТИЧЕСКАЯ РАБОТА № 28**

Тема: Создание файла проекта базы данных. Использование исполняемого файла проекта БД, приемы создания и управления

## **ПРАКТИЧЕСКАЯ РАБОТА № 29**

Тема: Модификация содержимого БД. Добавление, удаление и обновление данных

## **ПРАКТИЧЕСКАЯ РАБОТА № 30**

Тема: Создание простых запросов к базе данных

## **ПРАКТИЧЕСКАЯ РАБОТА № 31**

Тема: Создание простых запросов к базе данных.

## **ПРАКТИЧЕСКАЯ РАБОТА № 32**

Тема: Создание сложных запросов к базе данных

## **ПРАКТИЧЕСКАЯ РАБОТА № 33**

Тема: Создание сложных запросов к базе данных

## **ПРАКТИЧЕСКАЯ РАБОТА № 34**

Тема: Символы подстановки и регулярные выражения (LIKE)

## **ПРАКТИЧЕСКАЯ РАБОТА № 35**

Тема: Вычисляемые поля

### **ПРАКТИЧЕСКАЯ РАБОТА № 36**

Тема: Вычисляемые поля

### **ПРАКТИЧЕСКАЯ РАБОТА № 37**

Тема: Проведение сортировки и фильтрации данных. Поиск данных по одному и нескольким полям. Поиск данных в таблице. Группировка данных (GROUP BY)

### **ПРАКТИЧЕСКАЯ РАБОТА № 38**

Тема: Проведение сортировки и фильтрации данных. Поиск данных по одному и нескольким полям. Поиск данных в таблице. Группировка данных (GROUP BY)

### **ПРАКТИЧЕСКАЯ РАБОТА № 39**

Тема: Разработка базы данных на языке SQL по индивидуальным заданиям

### **ПРАКТИЧЕСКАЯ РАБОТА № 40**

Тема: Разработка базы данных на языке SQL по индивидуальным заданиям

## **ИНСТРУКЦИИ ДЛЯ ОБУЧАЮЩИХСЯ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ РАБОТ**

Прежде чем приступить к выполнению заданий, внимательно прочитайте данные рекомендации. Практические работы включают в себя задания следующих видов.

### **1. Работа за компьютером**

В ходе выполнения практических работ студент должен:

- выполнять требования по охране труда
- соблюдать инструкцию по правилам и мерам безопасности в кабинете информационных технологий
- строго выполнять весь объем работы, указанный в задании
- соблюдать требования эксплуатации компьютерной техники (правила включения и выключения)
- предоставить отчет о проделанной работе по окончании выполненной работы, который должен содержать:

1. Название работы.
2. Цель работы.
3. Задание и его решение.
4. Вывод о проделанной работе.

Текст отчета по практической работе должен быть набран на компьютере шрифтом Times New Roman размером 14 пт. (при оформлении текста используется текстовый редактор MS Word). Шрифт, используемый в иллюстративном материале (таблицы и рисунки), рекомендуется уменьшить до 12 пт. Межстрочный интервал в основном тексте - полуторный. В иллюстративном материале межстрочный интервал рекомендуется сделать одинарным. Поля страницы должны быть: левое поле - 30 мм; правое поле – 15 мм; верхнее и нижнее поле - 20 мм.

Каждый абзац должен начинаться с красной строки. Отступ абзаца – 1,25 см от левой границы текста.

Студент должен выполнить практическую работу самостоятельно (или в группе, если это предусмотрено заданием). Практическая работа выполняется согласно заданию и методическим рекомендациям. После выполнения практической работы обучающийся самостоятельно себя контролирует путем ответов на вопросы. Результат работы представляется преподавателю в виде файла (файлов) в личном каталоге, защищается обучающимися.

По ходу выполнения работы при возникновении вопросов обучающийся может получить консультацию у преподавателя или самостоятельно воспользоваться лекционным материалом, рекомендуемой литературой.

### **2. Ответ на поставленные вопросы (с аргументацией)**

Прочитайте вопрос и вникните в него.

Для удобства подчеркните ту, фразу, которая, по вашему мнению, является главной. Это поможет вам быстрее сориентироваться при ответе на вопрос.

Если вы считаете, что можете ответить на вопрос без помощи лекции и дополнительной литературы – приступайте. Если же вопрос заставляет вас

сомневаться, откройте лекционную тетрадь (учебник или дополнительную литературу), прочитайте необходимый пункт, вникните в содержание и после этого приступайте за работу.

**ГЛАВНОЕ!** Не переписывайте отрывки лекции в рабочую тетрадь! Четко отвечайте на ПОСТАВЛЕННЫЙ вопрос!

Не забудьте привести аргументацию (обоснование) вашей позиции, если вопрос предполагает личностное отношение к проблеме.

### **3. Заполнение таблиц и схем**

Прочитайте название таблицы или схемы.

Исходя из названия, вы поймете цель предстоящей работы.

Воспользуйтесь материалами лекций или другими источниками, чтобы заполнить таблицу (схему).

Используйте цветные графические материалы для выделения строк, столбцов или элементов схем.

Особое внимание обращайтесь на четкость при отборе материала: делайте записи кратко и четко!

**4. Поиск информации в сети** — использование web-браузеров, баз данных, пользование информационно-поисковыми и информационно-справочными системами, автоматизированными библиотечными системами, электронными журналами. Поиск и обработка информации включает подготовку фрагмента практического занятия.

## ПРАКТИЧЕСКАЯ РАБОТА № 1

### Тема: Модели данных

**Цель работы:** изучить модели представления данных.

**Оборудование:** ПК, интернет, инструкции по выполнению работы.

**Справочный материал:**

*Иерархическая модель* содержит модель, представляющую собой упорядоченные наборы деревьев. Данная модель данных построена по принципу иерархии типов объектов (один тип объекта - главный, другие – подчиненные. Главный и подчиненные объекты связаны по типу "один ко многим". Для каждого подчиненного типа объекта имеется лишь один исходный тип объекта. Основной недостаток данной модели – достаточно длительный поиск необходимой информации.

*Сетевая модель.* В данной модели любой объект может быть как главным, так и подчиненным. Каждый объект имеет возможность участвовать в любом числе взаимодействий. Другими словами, любая информационная единица может иметь множество предков и множество потомков. В моделях подобного рода связи заложены внутри описаний объектов. Достоинством является гибкость модели, т.е. имеется возможность повышения быстродействия системы. Недостатком является нагрузка на информационные ресурсы.

*Реляционная модель данных.* Свое название получила от английского термина relation, что означает «отношение». При соблюдении определенных условий отношение можно представить в виде двумерной привычной для человека таблицы.

При табличной организации данных отсутствует иерархия элементов. Строки и столбцы могут быть просмотрены в любом порядке, поэтому высока гибкость выбора любого подмножества элементов в строках и столбцах. Любая таблица в реляционной базе состоит из строк, которые называют записями, и столбцов, которые называют полями. На пересечении строк и столбцов находятся конкретные значения данных. Для каждого поля определяется множество его значений.

В реляционной модели данных применяются разделы реляционной алгебры, откуда и была заимствована соответствующая терминология. В реляционной алгебре поименованный столбец отношения называется атрибутом, а множество всех возможных значений конкретного атрибута – доменом. Строки таблицы со значениями разных атрибутов называют кортежами. Атрибут, значение которого однозначно идентифицирует кортежи, называется ключевым (или просто ключом). Так ключевое поле – это такое поле, значения которого в данной таблице не повторяются. В отличие от иерархической и сетевой моделей данных в реляционной отсутствует понятие группового отношения. Для отражения ассоциаций между кортежами разных отношений используется дублирование их ключей. Сложный ключ выбирается в тех случаях, когда ни одно поле таблицы однозначно не определяет запись.

Записи в таблице хранятся упорядоченными по ключу. Ключ может быть



простым, состоящим из одного поля, и сложным, состоящим из нескольких полей. Сложный ключ выбирается в тех случаях, когда ни одно поле таблицы однозначно не определяет запись.

Кроме первичного ключа в таблице могут быть вторичные ключи, называемые еще внешними ключами, или индексами. Индекс – это поле или совокупность полей, чьи значения имеются в нескольких таблицах и которое является первичным ключом в одной из них. Значения индекса могут повторяться в некоторой таблице. Индекс обеспечивает логическую последовательность записей в таблице, а также прямой доступ к записи.

### Содержание работы:

**Задание 1.** Построить иерархическую модель базы данных для предметной области Университет. В состав университета входят факультеты, которые, в свою очередь, включают в себя кафедры и готовят студентов по направлениям. Студенты зачислены в группы, соответствующие тому или иному направлению подготовки. Любой преподаватель является сотрудником одной кафедры.

Каждая вершина дерева соответствует одному типу объектов предметной области, который может характеризоваться произвольным количеством свойств (атрибутов).



**Задание 2.** Построить реляционную модель данных рейсов самолета, содержащую номер рейса, пункты отправления и прибытия, время вылета и время прибытия, тип самолета, стоимость полета.

Домен пунктов отправления (назначения) – множество названий населенных пунктов, а домен номеров рейса – множество целых положительных чисел.

Таблица. Рейс

Номер рейса	Дни недели	Пункт отправления	Время вылета	Пункт назначения	Время прибытия	Тип самолета	Стоимость билета
138	2_4_7	Баку	21.12	Москва	0.52	ИЛ-86	115.00
57	3_6	Ереван	7.20	Киев	9.25	ТУ-154	92.00
1234	2_6	Казань	22.40	Баку	23.50	ТУ-134	73.50
242	1 по 7	Киев	14.10	Москва	16.15	ТУ-154	57.00

86	2_3_5	Минск	10.50	Сочи	13.06	ИЛ-86	78.50
137	1_3_6	Москва	15.17	Баку	18.44	ИЛ-86	115.00
241	1 по 7	Москва	9.05	Киев	11.05	ТУ-154	57.00
577	1_3_5	Рига	21.53	Таллин	22.57	АН-24	21.50
78	3_6	Сочи	18.25	Баку	20.12	ТУ-134	44.00
578	2_4_6	Таллин	6.30	Рига	7.37	АН-24	21.50

Смысл доменов состоит в следующем. Если значения двух атрибутов берутся из одного и того же домена, то, вероятно, имеют смысл сравнения, использующие эти два атрибута (например, для организации транзитного рейса можно дать запрос «Выдать рейсы, в которых время вылета из Москвы в Сочи больше времени прибытия из Архангельска в Москву»). Если же значения двух атрибутов берутся из различных доменов, то их сравнение, вероятно, лишено смысла: стоит ли сравнивать номер рейса со стоимостью билета?

Степень отношения «Рейс» равна 8. Кардинальное число или мощность отношения – это число его кортежей. Мощность отношения «Рейс» равна 10. Кардинальное число отношения изменяется во времени в отличие от его степени.

**Задание 3.** Построить иерархическую и сетевую модель данных многоуровневой файловой системы. Указать корневой тип, подтипы, узлы.

**Задание 4.** Построить реляционную модель данных. Указать атрибуты, кортежи, степень отношения, кардинальное число, домен.

а) Студенты ВУЗа;

б) Книги.

## ПРАКТИЧЕСКАЯ РАБОТА № 2

### Тема: Операции реляционной алгебры.

**Цель работы:** научиться применять на практике операции реляционной алгебры.

**Оборудование:** ПК, интернет, инструкции по выполнению работы.

#### Справочный материал:

Результатом пересечения отношений A и B будет отношение с тем же заголовком, что и у отношений A и B, и телом, состоящим из кортежей, принадлежащих одновременно обоим отношениям A и B.

Результатом разности отношений A и B будет отношение с тем же заголовком, что и у совместимых по типу отношений A и B, и телом, состоящим из кортежей, принадлежащих отношению A и не принадлежащих отношению B.

При выполнении прямого произведения двух отношений производится отношение, кортежи которого являются конкатенацией (сцеплением) кортежей первого и второго операндов.

Реляционное деление достаточно нетривиально описать, но на примере его смысл нагляден. В целом, из таблицы A берутся значения строк, для которых присутствуют все комбинации значений из таблицы B.

Операция соединения есть результат последовательного применения операций декартового произведения и выборки. Если в отношениях и имеются атрибуты с одинаковыми наименованиями, то перед выполнением соединения такие атрибуты необходимо переименовать.

#### Содержание работы:

**Задание 1.** Дана БД производственного предприятия, состоящая из четырех таблиц:

ПРЕДПРИЯТИЕ			
Пред#	Название	Рейтинг	Город
180	Электроника	230	Воронеж
230	Гормолзавод	300	Москва
150	Сельмаш	140	Воронеж
190	Хлебозавод	300	Курск
270	Рудгормаш	240	Москва

где Пред# – номер предприятия, номер общий по некоторым группам городов; Название – название предприятия; Рейтинг – рейтинг предприятия по некоторым показателям; Город – город, в котором находится предприятие.

ПРОДУКЦИЯ			
Прод#	Наименование	Количество	ГородВыпуска
10	Магнитофоны	12000	Воронеж
20	Кровати	15000	Москва
30	Тракторы	20000	Воронеж
40	Кухни	30000	Орел
50	Продукты	10000	Воронеж

где Прод# – номер продукции; Наименование – наименование продукции; Количество – стоимость продукции, выпускаемой в год в данном городе; ГородВыпуска – город, в котором указанная продукция выпускается.

Работник				
ТН	Фамилия	ГородПрожив	День_рожд	Пред#
55	Иванов	Воронеж	15.03.02	180
10	Петров	Москва	17.02.95	230
100	Сидоров	Воронеж	03.12.93	150
190	Иванов	Курск	18.04.91	190

где ТН – номер личности; Фамилия – фамилия человека; ГородПрожив – город проживания; День\_рожд – дата рождения данного человека; Пред# – номер предприятия, где работает данная личность.

ПРЕД_ПРОД			
Пред#	Прод#	Год	Выработка
150	30	2000	150
180	10	2000	100
190	50	2001	50
230	50	2001	120
270	20	2002	50

где Пред# – номер предприятия; Прод# – номер продукции; Год – год выпуска продукции; Выработка (тыс.руб) – количество продукции данного предприятия.

#### Вариант 1

1. Получить названия предприятий, производящих продукцию с номером 30
2. Выбрать информацию обо всех предприятиях, в т.ч. о работниках и продукции.
3. Выбрать фамилии людей, которые работают на хлебозаводе.
4. Определить номера предприятий из Воронежа с рейтингом выше 200
5. Выбрать имена предприятий, производящих все виды продукции.

#### Вариант 2

1. Получить имена предприятий, производящих продукцию всех сортов.
2. Выбрать название продукции, у которой количество потребления в городе находится в диапазоне от 12000 до 15000
3. Выбрать фамилии людей, у которых город проживания совпадает с городом нахождения предприятия.
4. Найти номера работников, работающих на одном предприятии.
5. Определить название предприятий, которые не производят продукцию с номером 50

#### Вариант 3

1. Получить номера предприятий, производящих по крайней мере ту продукцию, которую выпускает предприятие с номером 190
2. Выбрать название предприятий, у которых выработка продукции в 2001 г. на единицу работающего составила более 100 тыс. руб.
3. Определить фамилии людей, работающих на предприятиях в г. Воронеже.

4. Определить имена предприятий, производящих продукцию с номером «10».
5. Определить номера предприятий, производящих по крайней мере все виды продукции, производимые предприятием с номером 270

#### Вариант 4

1. Выбрать все пары названий городов, для которых предприятие и работники находятся в одном городе.
2. Выбрать название предприятий, которые производят продукты.
3. Определить название предприятий, производящих продукцию с номером 50 в 2001 году.
4. Определить номера предприятий, имеющих в списке работающих по крайней мере одного «Иванова».
5. Получить номера продукции, которая имеет количество более 15000 или производится предприятием с номером 270

#### Вариант 5

1. Получить имена предприятий, не производящих продукцию с номером 50
2. Выбрать названия городов, для которых предприятие из первого города, а интересующая продукция во втором городе.
3. Определить название продукции с номером 30, имеющей выработку на единицу работающего  $> 100$  тыс. руб.
4. Найти названия предприятий, производящих по крайней мере одну продукцию с номером 50
5. Найти названия предприятий, выпускающих одинаковую продукцию.

### **ПРАКТИЧЕСКАЯ РАБОТА № 3**

#### **Тема: Проектирование баз данных. Сбор сведений и системный анализ предметной области**

**Цель работы:** освоить принципы выделения информационных объектов при анализе предметной области; научиться устанавливать связи между информационными объектами.

**Оборудование:** ПК, интернет, программное обеспечение – MS Word, MS Visio, инструкции по выполнению работы.

#### **Справочный материал:**

Первым этапом проектирования БД любого типа является анализ предметной области, который заканчивается построением информационной структуры (концептуальной схемы). Предметная область – это часть реального мира, данные о которой разработчик отражает в определенной базе данных. На данном этапе анализируются запросы пользователей, выбираются информационные объекты и их характеристики, которые определяют содержание проектируемой БД.

На основе проведенного анализа структурируется предметная область. Анализ предметной области не зависит от программной и технической сред, в которых будет реализовываться БД.

Анализ предметной области целесообразно разбить на три фазы:

1. анализ требований и информационных потребностей;
2. выявление информационных объектов и связей между ними;
3. построение модели предметной области и проектирование схемы БД.

На этапе анализа концептуальных требований и информационных потребностей необходимо выполнить;

- а) анализ требований пользователей к базе данных (концептуальных требований);
- б) выявление имеющихся задач по обработке информации, которая должна быть представлена в базе данных (анализ приложений),
- в) выявление перспективных задач (перспективных приложений);
- г) документирование результатов анализа

Требования пользователей к разрабатываемой БД представляют собой список запросов с указанием их интенсивности и объемов данных. Эти сведения разработчики БД получают в диалоге с ее будущими пользователями. Здесь же выясняются требования к вводу, обновлению и корректировке информации. Требования пользователей уточняются и дополняются при анализе имеющихся и перспективных задач.

#### **Содержание работы:**

##### **Задание 1.** Описать предметную область «Общежитие»

Комендант общежития ведет журнал по вселению и выселению различных категорий жильцов, которыми могут быть студенты, преподаватели, молодые семейные пары, родители или родственники студентов, временно проживающие в общежитии на период посещения. Комендант ведет реестр, в котором отмечается ежемесячная оплата за проживание, и льготы каждого жильца, влияющие на оплату. Ведется учет информации о наличии инвентаря и

выдачи его на определенный срок жильцам общежития. Комендант при вселении регистрирует жильца, т.е. временно прописывает его по адресу проживания.

Важной областью исследования являются бизнес-правила и политика университета (деловой регламент), на балансе которого находится общежитие. Бизнес-правила или деловой регламент – это непреложные факты, которым всегда должно подчиняться проектируемое приложение. В настоящем примере можно сформулировать следующие бизнес-правила:

- при наличии свободных мест в общежитии декан подписывает заявление и направляет студента к коменданту;
- одиноким жильцам предлагаются КОМНАТЫ двух и трехместные;
- семейным жильцам предлагаются двухместные комнаты;
- коменданту студент предъявляет паспорт, медицинскую справку о состоянии здоровья, справку о перечне льгот, которые повлияют на оплату за проживание;
- перечень ЛЬГОТ определяет администрация университета по форме (код\_льготы, наименование\_льготы, %\_от\_оплаты);
- оплата за проживание в общежитии для каждой категории жильцов различна;
- оплата за комнату складывается из оплат каждым жильцом этой комнаты + оплата за электроэнергию;
- в конце каждого месяца формируется РЕЕСТР – документ по оплате за электроэнергию и потребление электроэнергии каждой комнатой;
- ЖИЛЕЦ может быть выселен из общежития за регулярные административные НАРУШЕНИЯ режима проживания, которые определены администрацией университета;
- инвентарь выделяется как на комнату, так и отдельному жильцу по требованию;
- в случае, если инвентарь не возвращается, то полагается наложить штраф на жильца или комнату в зависимости от того, на кого был оформлен инвентарь;
- сумма штрафа зависит от % износа инвентаря, который зафиксирован в справочнике ИНВЕНТАРЬ.

В результате анализа формируется иерархия функций (рис.1) и модель «сущность-связь».



Рис.1. Функциональная структура предметной области «Общежитие»

Ниже представлен краткий перечень вопросов, определяющий архитектуру потоков данных, предназначенных для моделирования передачи информации между источником и приемником:

1. Есть ли свободные комнаты (мужские/женские) в общежитии?
2. Есть ли свободные места (мужские/женские) в общежитии?
3. Сформировать отчет по оплате за месяц проживания в общежитии по каждой комнате.
4. Сформировать отчет о задолжниках по оплате за электроэнергию по месяцам.
5. Найти конкретного жильца по фамилии, или по дате вселения, или по городу, из которого он приехал.
6. Сколько студентов той или иной специальности в % соотношении проживает в общежитии?
7. Сколько стоит проживание в общежитии в течение года для каждой категории жильцов?
8. Определить дату выселения жильца и комнату, из которой он выселен.
9. Провести процедуру переселения жильца в свободную комнату.
10. Сформировать квитанцию по оплате за проживание конкретному жильцу.
11. Какие льготы предоставлены конкретному жильцу, если они у него есть?
12. Определить количество жильцов, которые не являются студентами.
13. Провести процедуру прописки жильца.
14. Зарегистрировать жильца.
15. Какой инвентарь выдан на конкретную комнату?
16. Какой инвентарь выдан конкретному жильцу?

В качестве источников и приемников позиционируются процессы, внешние объекты и накопители данных.

**Задание 2.** Описать предметную область «Аптека»



## ПРАКТИЧЕСКАЯ РАБОТА № 4

### Тема: Проектирование баз данных. Сбор сведений и системный анализ предметной области

**Цель работы:** освоить принципы выделения информационных объектов при анализе предметной области; научиться устанавливать связи между информационными объектами.

**Оборудование:** ПК, интернет, программное обеспечение – MS Word, MS Visio, инструкции по выполнению работы.

#### Содержание работы:

**Задание 1.** Проанализировать предложенные бизнес-правила (индивидуальные варианты), определить объекты предметной области, провести подробное словесное (текстовое) описание объектов предметной области и реальных связей, присутствующих между реальными объектами.

#### Предложенные бизнес-правила «АЭРОПОРТ».

Создаваемая информационная система предназначена для учета движения самолетов и пассажиропотока. В аэровокзале имеется расписание движения самолетов, которое включает: Номер рейса, Тип самолета, Маршрут, Пункты промежуточной посадки, Время отправления, Дни полета. В системе ведется учет: Количество свободных мест на каждом рейсе, Общий вес пассажиров, Вес ручной клади, Вес багажа.

Система формирует посадочную ведомость с учетом веса багажа и ручной клади. В системе имеется справочник типов самолетов, в котором учитываются: Количество мест, Суммарная грузоподъемность.

#### Текстовое описание предметной области

##### Задача «Аэропорт»:

1. Каждый аэропорт обслуживает рейсы разных авиакомпаний и имеет международный код и название.
2. Авиакомпания характеризуется названием. У каждой авиакомпании есть несколько рейсов, проходящих через этот аэропорт.
3. Каждому рейсу соответствует один самолёт («воздушное судно»), каждому самолёту — несколько рейсов.
4. Самолёт характеризуется номером, маркой, моделью, вместимостью.
5. Каждому рейсу соответствует несколько экипажей, выполняющих их в разное время согласно расписанию.
6. Рейсы могут быть терминальные — завершающиеся или начинающиеся в аэропорту и транзитные, которые используют аэропорт только для временной посадки для заправки и отдыха.
7. Рейс характеризуется номером, типом (терминальные/транзитные), аэропортом отправления, аэропортом назначения, временем отправления, временем прибытия, временем в пути, дальностью, периодичностью (по нечётным дням, по чётным, по выходным, каждый N понедельник/вторник/.../воскресенье).
8. Экипаж состоит из нескольких сотрудников авиакомпании. Каждый член экипажа имеет ФИО, должность (командир, пилот, стюардесса) и лётный стаж, исчисляющийся в количестве вылетов.

9. В аэропорту происходит 2 типа событий — вылет и посадка. Каждое событие — вылет или посадка — характеризуется состоянием: «ожидается»/«состоялось»/«отменён», датой, временем задержки/опережения\_\_\_\_\_.

### **Индивидуальные варианты.**

#### **Вариант 1. Бизнес-правила «РЕСТОРАН».**

Постоянным клиентам предоставляется возможность заказать столик заранее. Официант указывает столик, открывает гостевой счет и вводит заказы в соответствии с меню. Далее заказ автоматически обрабатывается, формируются марки на приготовление выбранных блюд и направляют их на производство, в соответствующие цеха кухни, в бар. Расчеты с посетителем сводятся к простой операции: на бланке печатается итоговый счет. Если клиент – постоянный посетитель, то соответствующие привилегии рассчитываются автоматически, затем указываются способ оплаты и полученная от клиента сумма.

#### **Вариант 2. Бизнес-правила «КИНОТЕАТР».**

Создаваемая информационная система предназначена для учета проданных билетов в кинотеатре. Кинотеатр имеет несколько залов. Сеансы планируются для каждого зала отдельно. Система формирует базу данных, включающую следующую информацию: Место и сеанс, Справочник кинозалов, Справочник сеансов и стоимость, Справочник фильмов, Справочник жанров. Система формирует отчеты: Отчет о посещаемости по месяцам, Отчет о популярности фильмов, Отчет о популярности жанров. Необходимо предусмотреть возврат билетов и денег.

#### **Вариант 3. Бизнес-правила «ГОСТИНИЦА».**

Номера в гостинице имеют разный уровень обслуживания и соответственно разную стоимость, (предоставление информации о свободных номерах и их стоимости). Клиенты могут бронировать номера по телефону или Интернету. За номерами прикреплен обслуживающий персонал. Необходимо вести учет обслуживания и оплаты номеров, (заказы в номер, телефонные звонки и т. д.). Клиент может несколько раз останавливаться в гостинице в разных номерах.

#### **Вариант 4. Бизнес-правила «ФИТНЕС – КЛУБА».**

Они предлагают пакеты услуг – абонементы. Подразумевая предоплату определенного набора услуг. Абонемент позволяет пользоваться ими в течение определенного времени. Для идентификации владельца абонемента используются клубные карты. Комплекс позволяет быстро и просто осуществлять резервирование ресурсов по просьбе постоянного клиента предприятия: как тренера, так и места — спортзала, солярия, бассейна для персональных тренировок или занятий.

#### **Вариант 5. Бизнес-правила «ОПТОВЫЙ СКЛАД».**

Создаваемая информационная система предназначена для учета деятельности оптового склада. Оптовый склад состоит из нескольких складских

помещений, каждое помещение имеет наименование, адрес и кладовщика. Склад принимает партии товаров от поставщиков и отпускает его клиентам мелкими партиями. Требуется вести (количественный и стоимостной) учет поступающих и отпускаемых товаров, поставщиков и клиентов, формировать приходные и расходные накладные. Сведения о товаре: Артикул, Наименование полное, Наименование сокращенное, Производитель, Поставщик, Количество, Цена. Сведения о поставщике и клиенте: Наименование, Адрес, Телефон. Накладная включает: Номер, Дата, Клиент, Список товаров, Общая сумма, Кладовщик. В системе формируются отчеты о поступлении и отпуске товаров на складе за произвольный период.

#### Вариант 6. Бизнес-правила «РЕКЛАМНОЕ АГЕНТСТВО».

Создаваемая информационная система должна вести учет деятельности рекламного агентства. Рекламное агентство регистрирует заявки от рекламодателей и публикует рекламы в печатных изданиях. О рекламодателе регистрируются следующие данные: Наименование, Адрес, Руководитель, Телефон, Заявка, Оплата, Издание, Место размещения рекламы. Заявка включает: Вид рекламы, Объем, Желаемые издания, Количество выходов рекламы, Дополнительная информация. Заявка от рекламодателя может содержать публикацию в несколько печатных изданиях и на различные даты выхода. Справочник печатных изданий включает: Наименование, Виды реклам, Стоимость рекламы. Требуется вести списки печатных изданий с их расценками на рекламу, списки рекламодателей, заявок. Система должна обеспечить оперативный просмотр списка заявок (печатные издания, рекламодатель, стоимость) на любую вводимую дату, а также формирование отчета о заявленных и выполненных рекламах.

#### Вариант 7. Бизнес-правила «МАГАЗИН “ЦВЕТЫ”».

Создаваемая информационная система предназначена для учета деятельности магазина по продаже цветов. В системе формируется база данных отдельных цветов и готовых букетов: Наименование цветка или букета, Поставщик цветов, Состав букета, Стоимость, Срок поступления, Срок и место хранения (выставочный зал, склад), Дата продажи. В системе ведется учет бракованных и увядших цветов. Формируется отчет о движении товара за заданный период времени.

#### Вариант 8. Бизнес-правила «АДМИНИСТРАТОР ГОСТИНИЦЫ».

Создаваемая информационная система предназначена для учета деятельности гостиницы. В гостинице имеется список номеров: Место нахождения номера, Класс, Число мест, Признак занятости места, Дата освобождения номера. Каждый гость проходит регистрацию: Паспортные данные, Даты приезда и отъезда, Номер, Место, Цель приезда, Организация, в которую прибыл (в случае командировки). Администратор гостиницы осуществляется поселение гостя: выбор подходящего номера (при наличии свободных мест), регистрация, оформление квитанции. В системе автоматически формируется квитанция об оплате услуг гостиницы. Система должна предусмотреть оформление дополнительной квитанции в случае продления гостем срока проживания в гостинице. В системе имеется

возможность поиска гостя по произвольному признаку и формируется отчет о текущем состоянии номеров гостиницы (номер, место, не занят/ занят и кем, дата отъезда).

#### Вариант 9. Бизнес-правила «МУЗЫКАЛЬНЫЙ МАГАЗИН».

Создаваемая информационная система предназначена для учета музыкальных произведений в магазине. В системе формируются: База групп и исполнителей, База песен, База дисков с перечнем песен (в виде ссылок). База групп и исполнителей содержит: Наименование группы или исполнителя, Страна, Год образования группы или год начала творческого пути, Краткое содержание творческого пути. База песен содержит: Название, Автор текста, Автор музыки, Время звучания. База дисков содержит: Название диска, Перечень песен (название, исполнитель, время звучания, номер трека). Система имеет возможность поиска всех песен заданной группы (исполнителя). Имеется возможность выбора всех дисков, где встречается заданная песня.

#### Вариант 10. Бизнес-правила «АВТОЗАПРАВКА (АЗС)».

Создаваемая информационная система предназначена для учета деятельности автозаправки. На автозаправке имеются несколько колонок для заправки топливом: АИ-98, АИ-95, АИ-92, АИ-80, АИ-76, Дизельное топливо. В базе данных должна быть информация: О колонках, О видах бензина, О ценах и остатках. Необходимо учитывать отпуск топлива по чеку: Номер колонки, Тип топлива, Количество, Цена за литр, Стоимость. Предусмотреть отпуск топлива по дисконтной карточке со скидкой, при этом необходимо учитывать: Номер карточки, Общее количество отпущенного топлива, Скидка в %. Размер скидок зависит от общего количества заправленного топлива. В 19 часов – “пересменка” операторов АЗС, печатается отчет об отпуске топлива за время от 19 часов предыдущего дня до 19 часов текущего дня.

## ПРАКТИЧЕСКАЯ РАБОТА № 5

**Тема:** Разработка концептуальной ER-модели предметной области.

**Цель работы:** научиться разрабатывать концептуальную ER-модель на основании описания предметной области

**Оборудование:** ПК, интернет, программное обеспечение – MS Word, MS Visio, инструкции по выполнению работы.

### **Справочный материал:**

Работа по проектированию базы данных включает выбор:

- таблиц, которые будут входить в базу данных;
- столбцов, принадлежащих каждой таблице;
- взаимосвязей между таблицами и столбцами.

Конструирование базы данных связано в конечном итоге с построением ее логической структуры, а именно: взаимосвязанных нормализованных реляционных таблиц, для формирования которых используется система управления базами данных (СУБД).

Основой для построения логической структуры является концептуальное моделирование, т.е. переход от описания реального мира к модели этого мира. Синтез концептуальной модели производится при помощи ряда методик. Одной из наиболее популярных семантических моделей данных является модель «Сущность-Связь» или «Объект-Отношение». Часто ее называют ER–модель, предложенная Петером Пин-Шен Ченом в 1976 г. Построение концептуальной модели заключается в выделении объектов и установлении между ними связей.

### Типы сущностей

*Типы сущностей* — объект или концепция, которые характеризуются на данном предприятии как имеющие независимое существование.

*Сущность* — экземпляр типа сущности, который может быть идентифицирован уникальным образом.

*Слабый тип сущности* — тип сущности, существование которого зависит от какого-то другого типа сущности.

*Сильный тип сущности* — тип сущности, существование которого не зависит от какого-то другого типа сущности.

### Атрибуты

*Атрибут* — свойство типа сущности или связи.

*Домен атрибута* — набор значений, которые могут быть присвоены атрибуту.

*Простой атрибут* — атрибут, состоящий из одного компонента с независимым существованием.

*Составной атрибут* — атрибут, состоящий из нескольких компонентов, каждый из которых характеризуется независимым существованием.

*Однозначный атрибут* — атрибут, который содержит одно значение для одной сущности.

*Многозначный атрибут* — атрибут, который содержит несколько значений для одной сущности.

*Производный атрибут* — атрибут, который представляет значение, производное от значения связанного с ним атрибута или некоторого множества атрибутов, принадлежащих некоторому (не обязательно данному) типу сущности.

#### Ключи

*Потенциальный ключ* — атрибут или набор атрибутов, который уникально идентифицирует отдельные экземпляры типа сущности.

*Первичный ключ* — потенциальный ключ, который выбран в качестве первичного ключа.

*Составной ключ* — потенциальный ключ, который состоит из двух или больше атрибутов.

#### Типы связей

*Тип связи* — осмысленная ассоциация между сущностями разных типов.

*Связь* — ассоциация между сущностями, включающая по одной сущности из каждого участвующего в связи типа сущности.

*Степень связи* — количество сущностей, которые охвачены данной связью.

#### Графические обозначения основных элементов модели:

1. Сущности обозначаются с помощью прямоугольников.
2. Атрибуты обозначаются в виде овалов, связанных с сущностями, к которым они принадлежат.
3. Связи обозначаются с помощью ромбов, соединённых линиями с участвующими в них сущностями.
4. Имена ключевых атрибутов подчёркиваются.
5. Овалы производных атрибутов отображаются прерывистой линией.

#### Правила трансляции текстовой модели в ER-модель:

1. Существительное, образующее некоторое независимое понятие, отображается в сущность-прямоугольник.
2. Существительное или фраза, представляющая собой некоторое свойство понятия, отображается в атрибут-овал этого понятия.
3. Глаголы, описывающие взаимосвязи между понятиями, отображаются в связи-ромбы.

#### **Содержание работы:**

**Задание 1.** На основе анализа проведенного подробного словесного (текстового) описания предметной области и реальных связей между реальными объектами (Практическая работа № 3, задание 1): Транслировать текстовую модель предметной области в концептуальную ER- модель предметной области.

Список объектов для предметной области «Общежитие» будет выглядеть таким образом: Общежитие, Комната, Жилец, Инвентарь, Льгота, Прописка, Оплата.

Каждый из этих объектов обладает собственными свойствами, например: Номер общежития, Адрес, Телефон, ФИО коменданта – это свойства или вторичные атрибуты объекта Общежитие, которые являются потенциальными столбцами будущей реляционной таблицы. Названия столбцов должны быть

предельно ясными и краткими. Концептуальная модель базы данных «Общежитие» иллюстрирует влияние объектов друг на друга и представлена на рис.1. Прямоугольникам соответствуют экземпляры объектов (сущностей), а стрелкам – связи, которые несут элементы информации:

- тип связи;
- родительская и дочерняя (зависимая) сущности;
- мощность связи.



Рис.1. Концептуальная модель базы данных «Общежитие»

Выделяются сильные и слабые сущности. Сильные соответствуют объектам, которые представлены в предметной области вне зависимости от наличия других объектов (Общежитие, Комната, Жилец, Инвентарь, Льгота). Слабые сущности существуют только при наличии связанных с ними других объектов (Прописка, Оплата). Сущности соответствует набор атрибутов – это ее свойства. Атрибуты делятся на простые (Фамилия, Имя, Отчество) и сложные (Адрес, ФИО коменданта); однозначные (Дата рождения) и многозначные (Пол, Категория, Семейное положение); исходные (Стоимость) и производные (% от льготы).

Следует обратить внимание на основные типы связей: идентифицирующая, неидентифицирующая, полная категория, неполная категория, многие-ко-многим.

Связь называется идентифицирующей, если экземпляр дочерней сущности идентифицируется через ее связь с родительской сущностью. Дочерняя сущность при этом всегда является зависимой: ЖИЛЕЦ=>ПРОПИСКА.

Связь называется неидентифицирующей, если экземпляр дочерней сущности идентифицируется иначе, чем через связь с родительской сущностью: ЖИЛЕЦ=>ЛЬГОТА.

Ситуация, когда экземпляру одной сущности соответствует один или несколько экземпляров второй сущности, а экземпляру второй сущности соответствует один или несколько экземпляров первой сущности, отражается связью многие-ко-многим между данными сущностями: ЖИЛЕЦ=>ЛЬГОТА.

Мощность связи представляет собой отношение количества экземпляров родительской сущности к соответствующему количеству экземпляров дочерней сущности.

На рис.1 число  $N$  показывает максимальное количество экземпляров сущности, с которым может быть связан один экземпляр другой сущности.

Между объектами ОБЩЕЖИТИЕ и КОМНАТА максимальная мощность связи  $1:N$ , т.е. одно общежитие имеет много комнат.

Между объектами КОМНАТА и ЖИЛЕЦ максимальная мощность связи  $1:N$ , т.е. в одной комнате могут проживать несколько жильцов.

Между объектами ЖИЛЕЦ и ПРОПИСКА максимальная мощность связи  $1:1$ , т.е. каждый жилец может быть прописан и только в одной комнате.

Между объектами ЖИЛЕЦ и ОПЛАТА максимальная мощность связи  $1:N$ , т.е. каждый жилец вносит несколько оплат в течение года.

Между объектами КОМНАТА и ИНВЕНТАРЬ максимальная мощность связи  $1:N$ , т.е. за одной комнатой закреплен разнообразный инвентарь.

Между объектами ЖИЛЕЦ и ЛЬГОТА максимальная мощность связи  $N:N$ , т.е. каждому жильцу предоставлены льготы, и конкретная льгота может быть предоставлена многим жильцам.

**Задание 2.** На основе анализа проведенного подробного словесного (текстового) описания предметной области и реальных связей между реальными объектами (Практическая работа № 3, задание 2): Транслировать текстовую модель предметной области в концептуальную ER- модель предметной области.



## ПРАКТИЧЕСКАЯ РАБОТА № 6

**Тема:** Разработка концептуальной ER-модели предметной области.

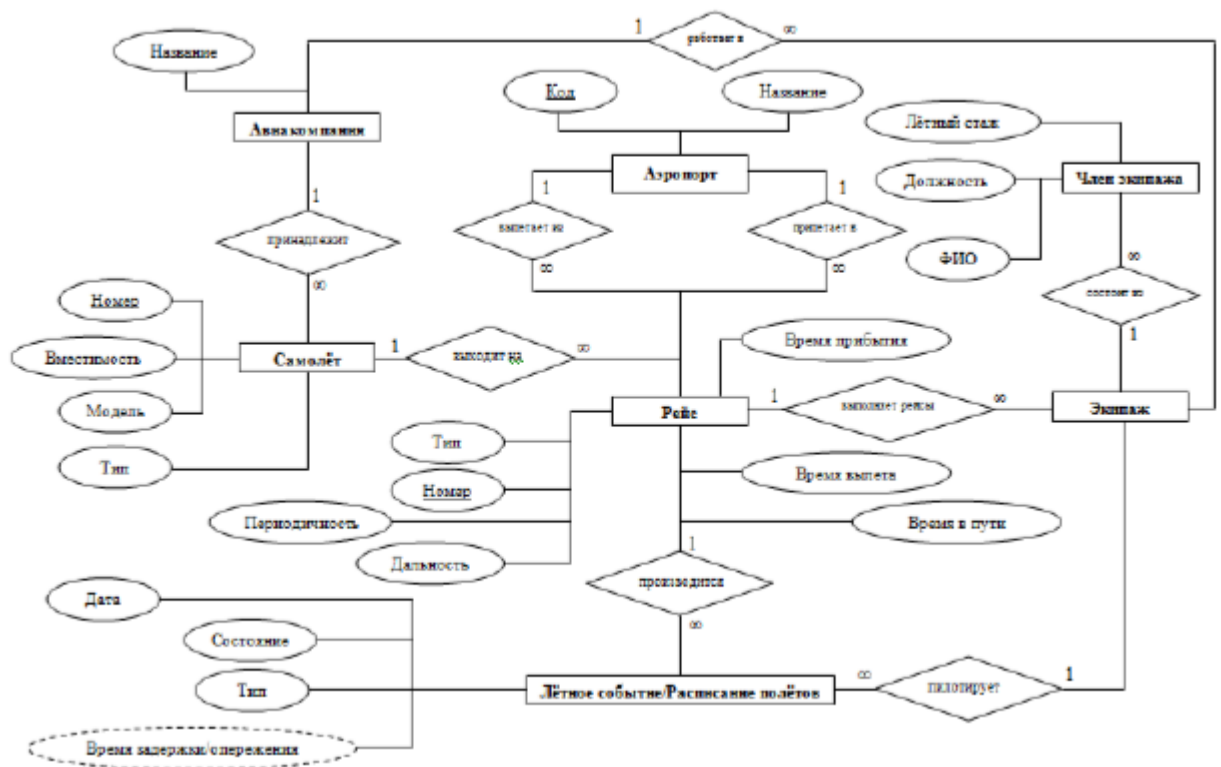
**Цель работы:** научиться разрабатывать концептуальную ER-модель на основании описания предметной области.

**Оборудование:** ПК, интернет, программное обеспечение – MS Word, MS Visio, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** На основе анализа проведенного подробного словесного (текстового) описания предметной области и реальных связей между реальными объектами (практическая работа № 4 - согласно варианту работы): Транслировать текстовую модель предметной области в концептуальную ER-модель предметной области.

Концептуальная ER –модель предметной области «Аэропорт» (результат данной практической работы № 4).



## ПРАКТИЧЕСКАЯ РАБОТА № 7

**Тема:** Создание логической реляционной модели базы данных

**Цель работы:** научиться создавать логическую модель базы данных на основании построенной концептуальной модели предметной области.

**Оборудование:** ПК, интернет, программное обеспечение – MS Word, MS Visio, инструкции по выполнению работы.

### **Справочный материал:**

На заключительном этапе проектирования базы данных конструируется логическая модель на основании концептуальной. *Логическая модель* – это взаимосвязанные реляционные таблицы.

Одна из задач проектирования базы данных - это обеспечение способа идентификации отдельных строк таблицы. Строки отличаются друг от друга по значению первичного ключа таблицы.

*Первичный ключ* – это атрибут или набор атрибутов, однозначно определяющий строку.

*Внешний ключ* — атрибут или множество атрибутов внутри отношения, которое соответствует потенциальному ключу некоторого (возможно, того же самого) отношения.

С точки зрения теории реляционных баз данных, существуют следующие понятия.

### Основные элементы реляционной модели

*Отношение* — плоская таблица, состоящая из столбцов и строк.

*Атрибут* — поименованный столбец отношения.

*Домен* — набор допустимых значений для одного или нескольких атрибутов.

*Кортеж* — строка отношения.

*Степень* — количество атрибутов в отношении.

*Кардинальность* — количество кортежей, которое содержит отношение.

*Реляционная БД* — набор нормализованных отношений.

### Реляционная целостность

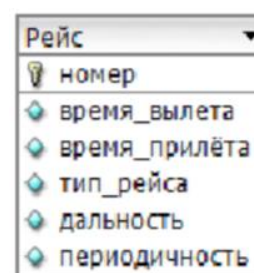
*Определитель* — значение атрибута в данный момент неизвестно или неприемлемо.

*Целостность сущностей* — в базовом отношении ни один атрибут первичного ключа не может содержать отсутствующих значений, обозначаемых оператором NULL.

*Ссылочная целостность* — значение внешнего ключа должно соответствовать значению потенциального ключа некоторого кортежа либо задаваться определителем NULL.

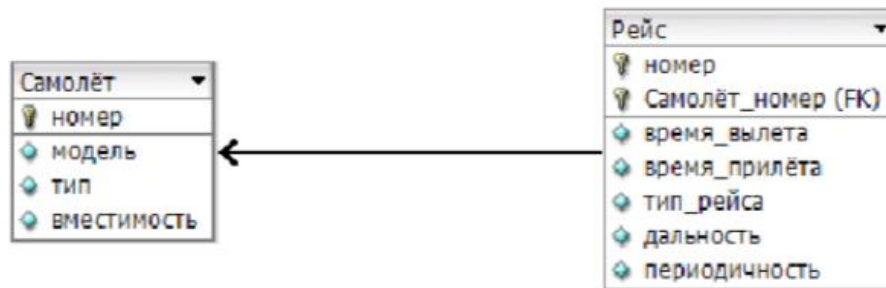
На диаграмме логической модели базы данных присутствуют следующие элементы.

Таблицы, представленные в виде прямоугольника, в верхней части которого располагается название таблицы, отчёркнутое линией, а в нижней — список полей таблицы с указанием ключей:



Рейс
номер
время_вылета
время_прилёта
тип_рейса
дальность
периодичность

Помимо первичных ключей, рассмотренных нами в ER-моделировании, логическая модель включает в себя внешние, либо вторичные ключи, которые необходимы для реализации связей между таблицами. Связи между таблицами изображаются в виде прямых, ломаных или кривых линий со стрелками, ведущих от внешних к первичным ключам связываемых таблиц:



### Правила преобразования ER-диаграммы в логическую модель

Для построения такой диаграммы существуют следующие правила:

1. Сущность становится таблицей с соответствующим именем.
2. Атрибут становится столбцом с таким же именем.
3. Связи типа 1:1 преобразуются одним из следующих вариантов:
  - 3.1. Связанные сущности-таблицы сливаются в одну таблицу.
  - 3.2. В одну из таблиц добавляется столбец – внешний ключ, содержащий ссылку-значение на первичный ключ другой таблицы.
  - 3.3. В обе таблицы добавляются столбцы – внешние ключи, содержащие ссылки на первичные ключи других таблиц.
4. Связи типа 1:N реализуются в виде добавления столбца – внешнего ключа в ту таблицу, которой соответствует N единиц сущностей отношения. Такая таблица называется подчинённой, или дочерней, а таблица, соответствующая одной сущности отношения — родительской или главной.
5. Связи типа M:N реализуются с помощью дополнительно вводимой вспомогательной таблицы, используемой лишь для хранения пар внешних ключей, ссылающихся на первичные ключи связываемых таблиц. Обычно такая таблица получает составное название из названий связываемых ею таблиц вида «Таблица1\_Таблица2».

### **Содержание работы:**

**Задание 1.** Создать логическую модель по построенной концептуальной модели (задание 1 из Практической работы № 5).

Первичные ключи. Для таблиц ОБЩЕЖИТИЕ, КОМНАТА, ИНВЕНТАРЬ, ЛЬГОТА, ЖИЛЕЦ первичными ключами являются соответственно Номер общежития, Номер комнаты, Номер инвентаря, Код льготы, Номер ордера. Таблицы ПРОПИСКА и ОПЛАТА не имеют явно выраженных первичных ключей. На рис.1 объекты ПРОПИСКА и ОПЛАТА связаны с объектом ЖИЛЕЦ в отношении 1:1 и 1:N соответственно.

На основании правила, приведенного ниже, следует добавить в качестве первичного ключа Номер ордера в эти таблицы.

**Правило.** Если экземпляр одной сущности связан с экземпляром другой сущности в отношении 1:1 или 1:N, то для связи необходимо первичный ключ

одной сущности добавить во вторую; если экземпляр одной сущности связан с экземпляром другой сущности в отношении N:N, то для связи необходимо создать дополнительную таблицу связи, в которую добавить первичные ключи из обеих таблиц.

Таблица ЖИЛЕЦ\_ЛЬГОТА – это дополнительная таблица, которая позволяет реализовать связь многие-ко-многим между таблицами ЖИЛЕЦ и ЛЬГОТА.

Это правило также позволяет определить внешние и родительские ключи для связи реляционных таблиц. Так для связи таблиц ОБЩЕЖИТИЕ и КОМНАТА в таблицу КОМНАТА добавляется ключ Номер общежития и в этой таблице называется внешним, а в таблице ОБЩЕЖИТИЕ – родительским, кроме того, что это и первичный ключ. Рассуждая аналогичным образом, определяются внешние ключи для всех остальных таблиц. Логическая модель данных представлена на рис.1.

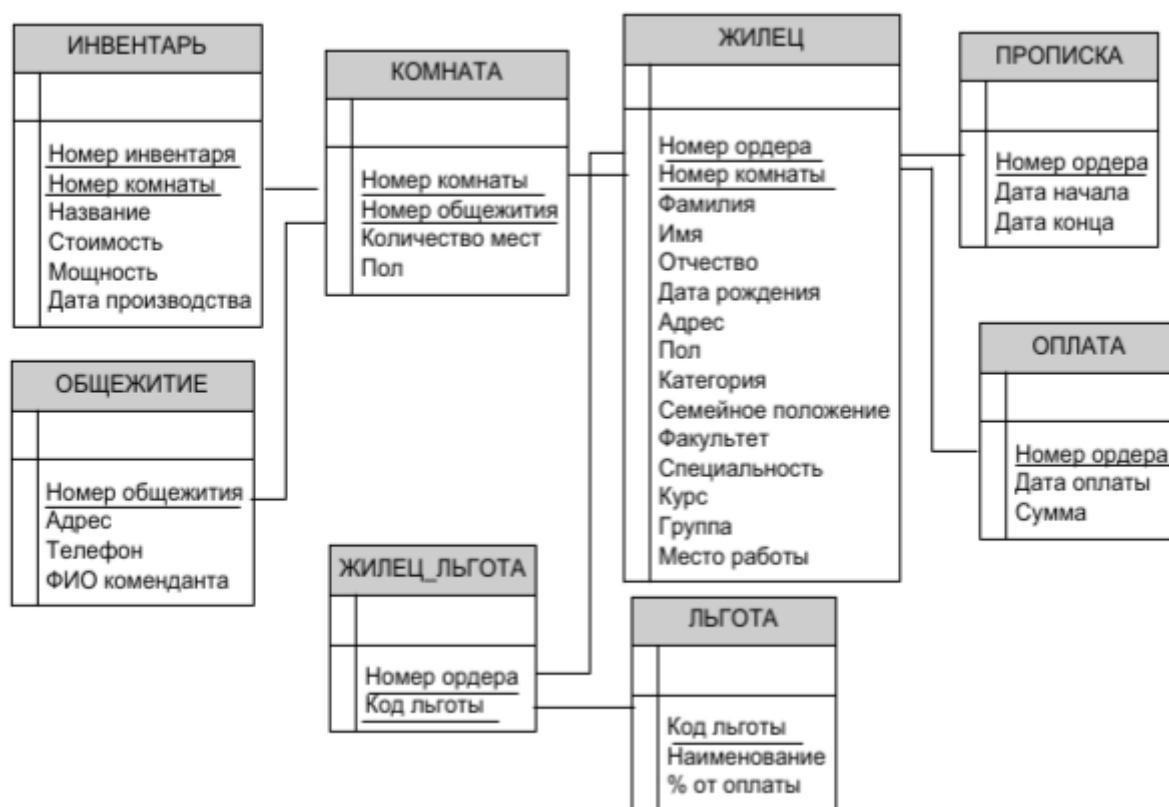


Рис.1. Логическая модель базы данных «Общежитие»

Однако на этом проектирование баз данных не заканчивается. Далее необходимо провести нормализацию, чтобы уменьшить избыточность данных. При этом проектировщика не должно смущать наличие в базе одинаковых столбцов первичных и внешних ключей. Такое преднамеренное дублирование - это не то же самое, что избыточность. Эта поддержка непротиворечивости между первичными и внешними ключами связана с понятием целостности данных. На рис.1 ключи в таблицах выделены подчеркиванием.

**Задание 2.** Создать логическую модель по построенной концептуальной модели (задание 2 из Практической работы № 5).

## ПРАКТИЧЕСКАЯ РАБОТА № 8

**Тема:** Создание логической реляционной модели базы данных

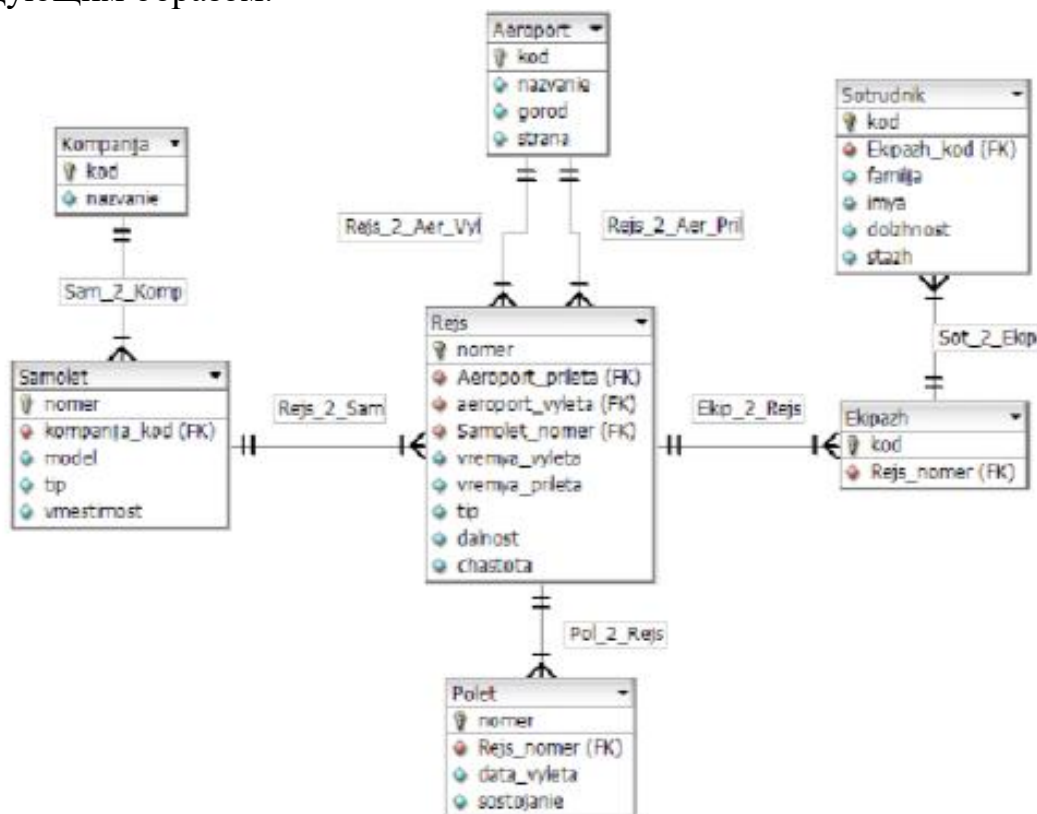
**Цель работы:** научиться создавать логическую модель базы данных на основании построенной концептуальной модели предметной области.

**Оборудование:** ПК, интернет, программное обеспечение – MS Word, MS Visio, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** Создать логическую модель по построенной концептуальной модели (индивидуальные варианты из Практической работы № 6).

Полная логическая реляционная модель базы данных для задачи «Аэропорт» (смотри Практический работы № 4 и № 6) будет выглядеть следующим образом.



## ПРАКТИЧЕСКАЯ РАБОТА № 9

### Тема: Нормализация баз данных

**Цель работы:** научиться проводить нормализацию отношений на этапе логического моделирования, используя декомпозицию исходных отношений, которые входят в логическую реляционную модель базы данных.

**Оборудование:** ПК, интернет, программное обеспечение – MS Word, MS Visio, инструкции по выполнению работы.

#### **Справочный материал:**

Согласно определению, реляционная база данных представляет собой набор нормализованных отношений (таблиц).

Под нормализованностью понимают соответствие отношений определённому набору правил, а процесс приведения модели БД в нормализованную форму называют нормализацией.

*Нормализация* — метод создания набора отношений с заданными свойствами на основе требований к данным, проистекающих из логики организации предметной области.

#### Избыточность данных и аномалии обновления

Известно, что одну и ту же предметную область можно описать в виде различных логических моделей. Как определить, какая из моделей будет более удачной, а какая — менее?

Неправильные модели приводят к ряду проблем при работе с базой данных, созданной на основании такой модели, главными из которых являются неоднозначность и большой объём лишней работы при обработке, вызванные избыточностью или наоборот, потерей части информации в модели:

1. Аномалии вставки,
2. Аномалии удаления,
3. Аномалия обновления,
4. Свойство соединения без потерь,
5. Свойство сохранения зависимости.

#### Функциональные зависимости

*Функциональная зависимость* — описывает связь между атрибутами отношения.

Например, если в отношении R, содержащем атрибуты A и B, атрибут B функционально зависит от атрибута A (что обозначается как  $A \rightarrow B$ ), то каждое значение атрибута A связано только с одним значением атрибута B.

*Детерминант* — детерминантом функциональной зависимости называется атрибут или группа атрибутов, расположенная на диаграмме функциональной зависимости слева от символа стрелки.

#### Процесс нормализации

*Ненормализованная форма* — таблица, содержащая одну или несколько повторяющихся групп данных.

*Первая нормальная форма* — отношение, в котором на пересечении каждой строки и каждого столбца содержится только одно значение.

*Полная функциональная зависимость* — в некотором отношении атрибут B называется полностью функционально зависимым от атрибута A, если

атрибут В функционально зависит от полного значения атрибута А и не зависит ни от какого подмножества атрибута А.

*Частичная функциональная зависимость* — такая зависимость  $A \rightarrow B$ , если в А есть некий атрибут, при удалении которого эта зависимость сохраняется.

*Вторая нормальная форма* — отношение, которое находится в первой нормальной форме и каждый атрибут которого, не входящий в состав первичного ключа, характеризуется полной функциональной зависимостью от этого ключа.

*Транзитивная зависимость* — если для атрибутов А, В и С некоторого отношения существуют зависимости вида  $A \rightarrow B$  и  $B \rightarrow C$ , то говорят, что атрибут С транзитивно зависит от атрибута А через атрибут В (при условии, что атрибут А функционально не зависит ни от атрибута В, ни от атрибута С).

*Третья нормальная форма* — отношение, которое находится в первой и второй нормальных формах и не имеет не входящих в первичный ключ атрибутов, которые находились бы в транзитивной функциональной зависимости от этого первичного ключа.

*Нормальная форма Бойса-Кодда (НФБК)* — отношение находится в НФБК тогда и только тогда, когда каждый его детерминант является потенциальным ключом.

### **Содержание работы:**

**Задание 1.** Привести к 3 нормальной форме базы данных, созданных по индивидуальным вариантам (практическая работа № 8).

Последовательно разберём полученные в результате преобразования таблицы «Аэропорт» на соответствие 1-3 нормальным формам.

#### Приведение модели в 1-ю нормальную форму

Поскольку ещё в процессе анализа мы избегали использования многозначных атрибутов, то в нашей модели нет таблиц, в ячейках которых хранилось бы более одного значения. Следовательно, наша модель уже находится в 1-й нормальной форме (1НФ).

#### Приведение модели во 2-ю нормальную форму

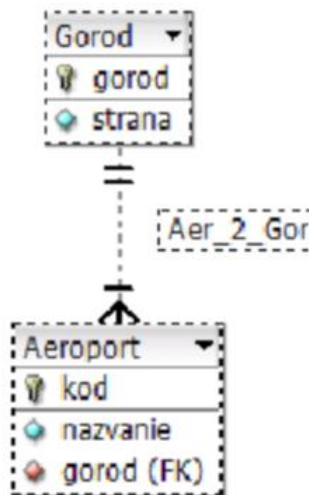
Согласно определению 2НФ, каждый неключевой атрибут каждой таблицы должен функционально зависеть от первичного ключа. Последовательно рассматривая каждую таблицу, убеждаемся, что это действительно так.

#### Приведение модели в 3-ю нормальную форму

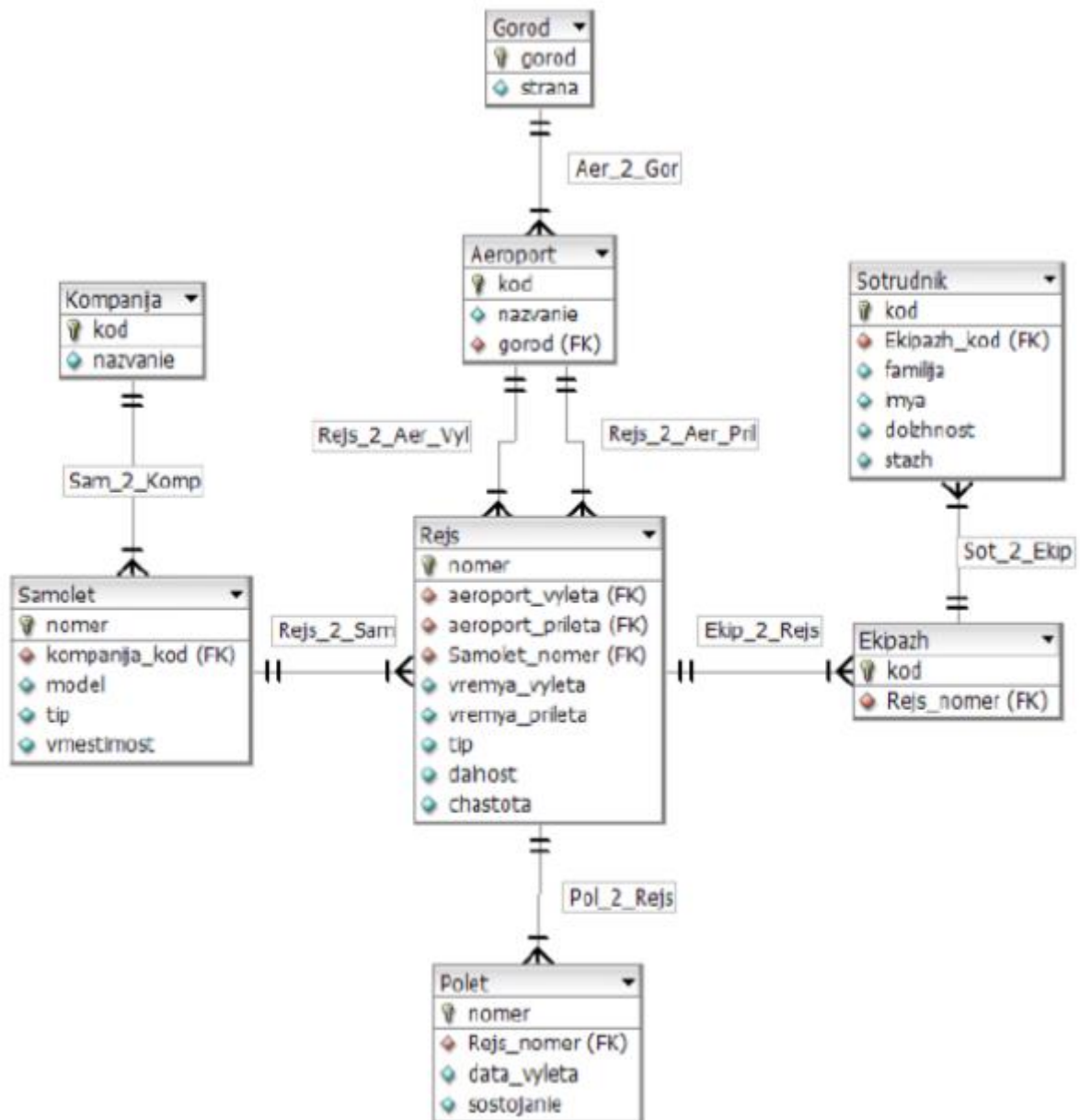
Рассмотрим, зависят ли атрибуты таблиц ТОЛЬКО от первичного ключа, или встречаются ещё и зависимости от других, неключевых атрибутов, называемые транзитивными.

Последовательно рассматривая каждую таблицу, обнаруживаем, что в таблице «Аэропорт» наблюдается транзитивная зависимость, а именно — атрибут «страна» зависит не только от первичного ключа, но и от атрибута «город», т.е. налицо транзитивная зависимость: «код аэропорта»  $\rightarrow$  «город»  $\rightarrow$  «страна». Для её исключения необходимо произвести декомпозицию таблицы

«Аэропорт» на две, а именно — вынести повторяющиеся группы «город», «страна» в таблицу «Город»:



Таким образом, мы привели нашу схему в 3-ю нормальную форму:





## **ПРАКТИЧЕСКАЯ РАБОТА № 10**

### **Тема: Нормализация баз данных**

**Цель работы:** научиться проводить нормализацию отношений на этапе логического моделирования, используя декомпозицию исходных отношений, которые входят в логическую реляционную модель базы данных.

**Оборудование:** ПК, интернет, программное обеспечение – MS Word, MS Visio, инструкции по выполнению работы.

#### **Содержание работы:**

**Задание 1.** Привести к 3 нормальной форме базы данных, созданных в заданиях 1 и 2 из Практической работы № 7.

- 1) Что такое нормализация и декомпозиция?
- 2) Каким аномалиям подвержены ненормализованные отношения?
- 3) Провести нормализацию вариантов отношений, представленных в приложении 2
- 4) Представить отчет с подробным анализом процесса декомпозиции – переходом по уровням нормализации.

## ПРАКТИЧЕСКАЯ РАБОТА № 11

**Тема: Преобразование реляционной базы данных в сущности и связи**

**Цель работы:** выработать практические навыки моделирования предметной области и построении ER-модели данных, закрепить технологию проектирования БД, закрепить основные понятия теории реляционных баз данных, освоить технологию построения ER-диаграмм, научиться получать реляционные БД из ER-диаграмм

**Оборудование:** ПК, интернет, программное обеспечение – MS Word, MS Visio, инструкции по выполнению работы.

### **Справочный материал:**

#### Пример проектирования реляционной базы данных

В качестве примера возьмем базу данных компании, которая занимается издательской деятельностью.

#### 1. Инфологическое проектирование

##### 1. 1. Анализ предметной области

База данных создаётся для информационного обслуживания редакторов, менеджеров и других сотрудников компании. БД должна содержать данные о сотрудниках компании, книгах, авторах, финансовом состоянии компании и предоставлять возможность получать разнообразные отчёты. В соответствии с предметной областью система строится с учётом следующих особенностей:


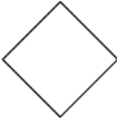
- каждая книга издаётся в рамках контракта;
- книга может быть написана несколькими авторами;
- контракт подписывается одним менеджером и всеми авторами книги;
- каждый автор может написать несколько книг (по разным контрактам);
- порядок, в котором авторы указаны на обложке, влияет на размер гонорара;
- если сотрудник является редактором, то он может работать одновременно над несколькими книгами;
- у каждой книги может быть несколько редакторов, один из них – ответственный редактор;
- каждый заказ оформляется на одного заказчика;
- в заказе на покупку может быть перечислено несколько книг.

Для инфологического проектирования воспользуемся методом «сущность-связь». Для того, чтобы представить, как устроена предметная область нужно задать множество объектов реального мира (главная проблема что считать объектом). Объект – семантическое понятие, которое может быть полезно при обсуждении устройств реального мира. Сущность реального мира – объекты – не обязательно материальны – важно понятие существенно и различимо для других. Между объектами могут возникать связи трех видов:

- один-к-одному 1:1 (пациент: место в палате);
- один-к-многим 1:n и многие к одному n:1;
- многие-ко-многим n:n (пациент : хирург).

При построении моделей используются следующие геометрические

фигуры:

Элемент ER-модели	Условно графическое представление
Объект	
Связь	
Атрибут	

Выделим базовые сущности этой предметной области:

– Сотрудники компании. Атрибуты сотрудников – ФИО, табельный номер, пол, дата рождения, паспортные данные, ИНН, должность, оклад, домашний адрес и телефоны. Для редакторов необходимо хранить сведения о редактируемых книгах; для менеджеров – сведения о подписанных контрактах.

– Авторы. Атрибуты авторов – ФИО, ИНН (индивидуальный номер налогоплательщика), паспортные данные, домашний адрес, телефоны. Для авторов необходимо хранить сведения о написанных книгах.

– Книги. Атрибуты книги – авторы, название, тираж, дата выхода, цена одного экземпляра, общие затраты на издание, авторский гонорар.

– Контракты будем рассматривать как связь между авторами, книгами и менеджерами. Атрибуты контракта – номер, дата подписания и участники.

Для отражения финансового положения компании в системе нужно учитывать заказы на книги. Для заказа необходимо хранить номер заказа, заказчика, адрес заказчика, дату поступления заказа, дату его выполнения, список заказанных книг с указанием количества экземпляров.

ER–диаграмма издательской компании приведена на рисунке ниже (базовые сущности на рисунках выделены полужирным шрифтом).

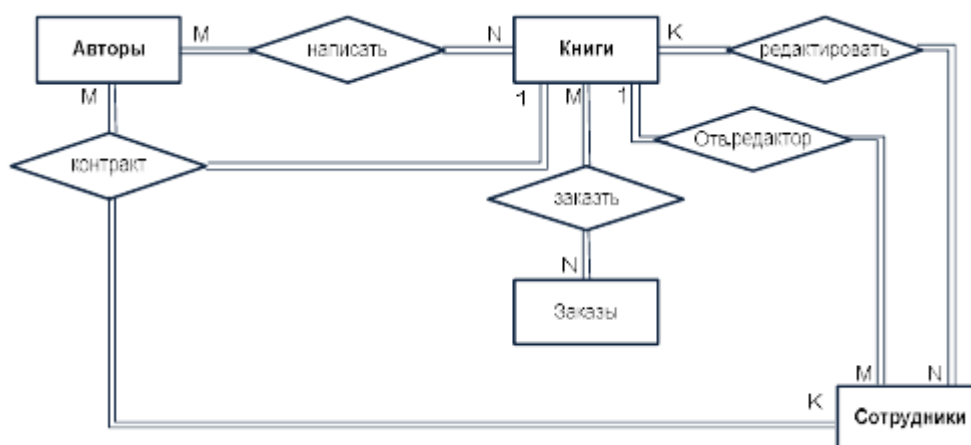


Рис. 1. – ER–диаграмма издательской компании

## 1.2. Логическое проектирование реляционной БД

### 1. 2.1. Преобразование ER–диаграммы в схему базы данных

База данных создаётся на основании схемы базы данных. Для преобразования ER–диаграммы в схему БД приведём уточнённую ER–диаграмму, содержащая атрибуты сущностей.

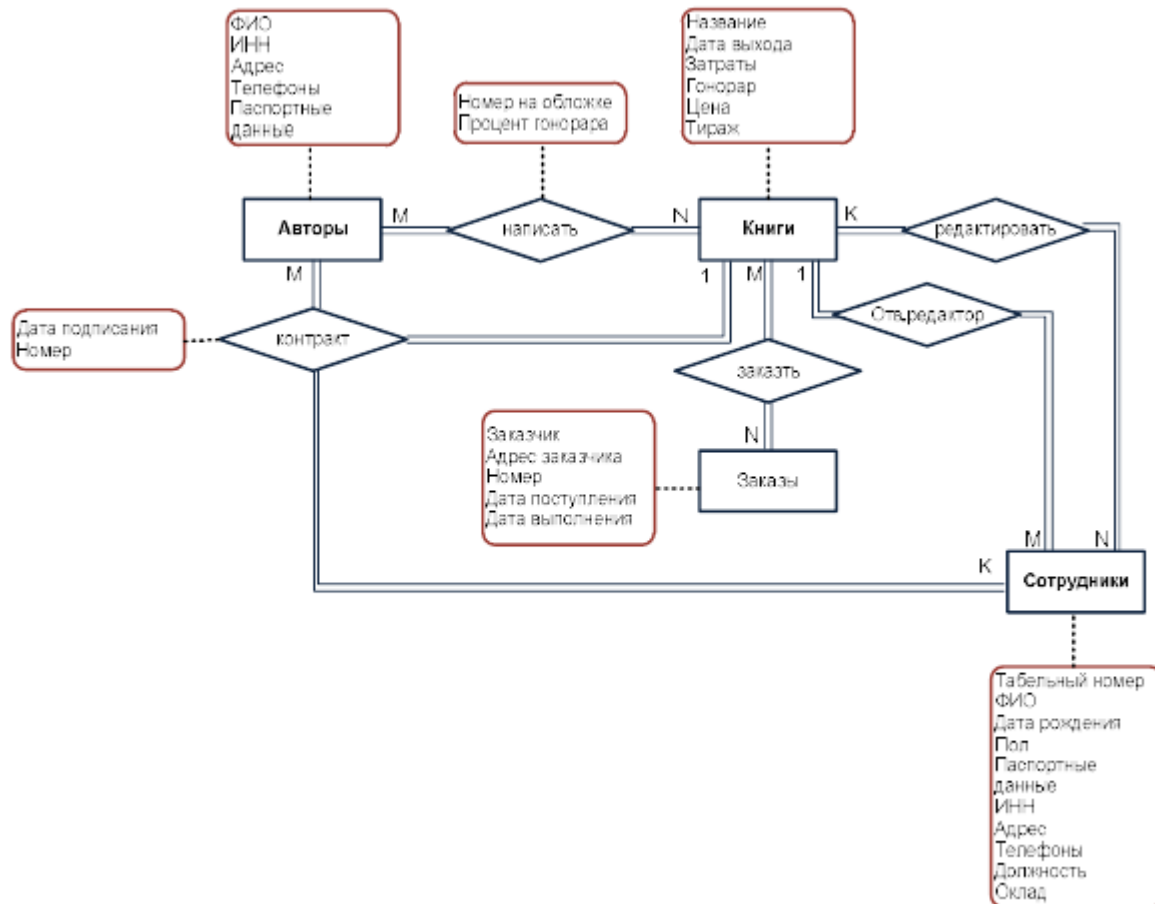


Рис.2 – Уточнённая ER–диаграмма издательской компании

Преобразование ER–диаграммы в схему БД выполняется путем сопоставления каждой сущности и каждой связи, имеющей атрибуты, отношения (таблицы БД). Будем использовать обозначения, представленные на рисунке 3.

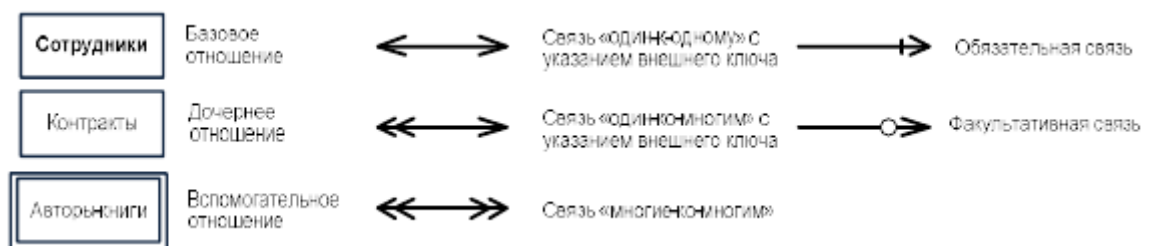


Рис.3 – Обозначения, используемые на схеме базы данных  
Полученная схема реляционной БД приведена на рисунок 4

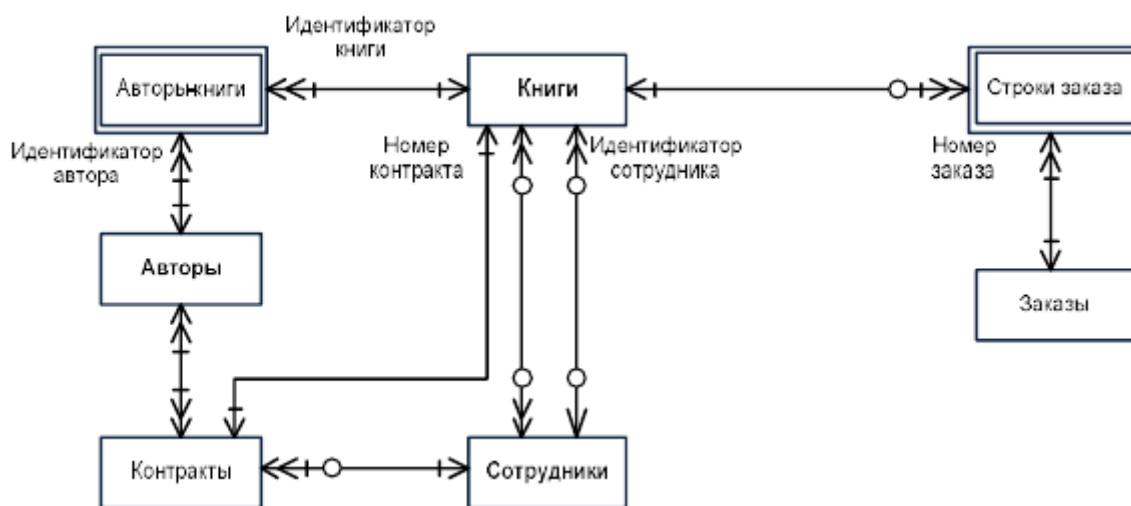


Рис. 4 – Схема реляционной БД, полученная из ER–диаграммы издательской компании

На схеме (рисунок 4) есть связь типа 1:1 – обязательная связь между КНИГАМИ и КОНТРАКТАМИ. Такие отношения следует объединять в одно.

Дополнительный эффект от объединения этих отношений – слияние связей авторы–контракты и авторы–книги: ведь в нашем случае контракт заключается именно для написания книги.

Связь типа 1:n (один-ко-многим) между отношениями реализуется через внешний ключ. Ключ вводится для того отношения, к которому осуществляется множественная связь (КНИГИ).

Связь редактировать между отношениями КНИГИ и СОТРУДНИКИ принадлежит к типу n:m (многие-ко-многим). Этот тип связи реализуется через вспомогательное отношение, которое является соединением первичных ключей соответствующих отношений.

Бинарная связь между отношениями не может быть обязательной для обоих отношений. После объединения сущностей КНИГИ и КОНТРАКТЫ остаётся три связи, обязательные для всех участников: между авторами и книгами и между заказами и строками заказов. Такой тип связи означает, что, например, прежде чем добавить новый заказ в отношение ЗАКАЗЫ, нужно добавить новую строку в отношение СТРОКИ ЗАКАЗА, и наоборот.

Поэтому для такой связи необходимо снять с одной стороны условие обязательности. Так как все эти связи будут реализованы с помощью внешнего ключа, снимем условие обязательности связей для отношений, содержащих первичные ключи.

Уточнённая схема реляционной БД издательской компании приведена на рисунке 5.

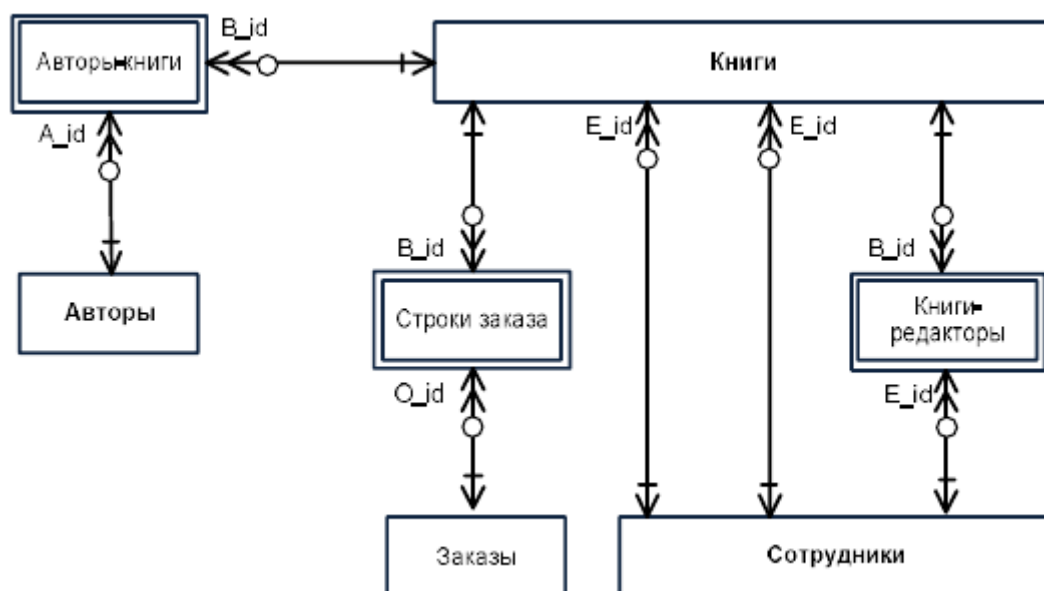


Рис. 5 – Уточнённая схема реляционной БД издательской компании

Схема на рисунке 5 содержит цикл «сотрудники–книги–сотрудники». Цикл допустим только в том случае, если связи, входящие в него, независимы друг от друга. Примем для нашей предметной области, что ответственный редактор книги может являться также просто редактором этой же книги или не входить в число редакторов. При этом цикл не приводит к нарушению логической целостности данных.

#### 1. 2.2. Составление реляционных отношений

Каждое реляционное отношение соответствует одной сущности (объекту предметной области) и в него вносятся все атрибуты сущности. Для каждого отношения необходимо определить первичный ключ и внешние ключи (если они есть). В том случае, если базовое отношение не имеет потенциальных ключей, вводится суррогатный первичный ключ, который не несёт смысловой нагрузки и служит только для идентификации записей.

Примечание: суррогатный первичный ключ также может вводиться в тех случаях, когда потенциальный ключ имеет большой размер (например, длинная символьная строка) или является составным (не менее трёх атрибутов).

Потенциальными ключами отношения АВТОРЫ являются атрибуты

Паспортные данные и ИНН. Первый хранится как длинная строка, а последний по условиям предметной области не является обязательным. Поэтому для авторов необходимо ввести суррогатный ключ – A\_id. Книги можно идентифицировать по атрибуту Контракт: его номер обязателен и уникален. Потенциальные ключи отношения СОТРУДНИКИ – атрибуты ИНН, Паспортные данные, Табельный номер, причём все они обязательные. Табельный номер занимает меньше памяти, чем ИНН, поэтому он и будет первичным ключом. Кортежи отношения ЗАКАЗЫ можно идентифицировать ключом Номер заказа.

Потенциальными ключами вспомогательных отношений являются комбинации первичных ключей соответствующих базовых отношений.

Отношения приведены в таблице 1 - 7. Для каждого отношения указаны атрибуты с их внутренним названием, типом и длиной. Типы

данных обозначаются так: N – числовой, C – символьный, D – дата (последний имеет стандартную длину, зависящую от СУБД, поэтому она не указывается).

Таблица 1.1 – Схема отношения СОТРУДНИКИ (Employees)

Содержание поля	Имя поля	Тип, длина	Примечания
Табельный номер	E_ID	N(4)	первичный ключ
Фамилия, имя, отчество	E_NAME	C(50)	обязательное поле
Дата рождения	E_BORN	D	
Пол	E_SEX	C(1)	обязательное поле
Паспортные данные	E_PASSP	C(50)	обязательное поле
ИНН	E_INN	N(12)	обязательное уникальное поле
Должность	E_POST	C(30)	обязательное поле
Оклад	E_SALARY	N(8,2)	обязательное поле
Адрес	E_ADDR	C(50)	
Телефоны	E_TEL	C(30)	многозначное поле

Таблица 1.2 – Схема отношения КНИГИ (Books)

Содержание поля	Имя поля	Тип, длина	Примечания
Номер контракта	B_CONTRACT	N(6)	первичный ключ
Дата подписания контракта	B_DATE	D	обязательное поле
Менеджер	B_MAN	N(4)	внешний ключ (к Employees)
Название книги	B_TITLE	N(40)	обязательное поле
Цена	B_PRICE	N(6,2)	цена экземпляра книги
Затраты	B_ADVANCE	N(10,2)	общая сумма затрат на книгу
Авторский гонорар	B_FEE	N(8,2)	общая сумма гонорара
Дата выхода	B_PUBL	D	
Тираж	B_CIRCUL	N(5)	
Ответственный редактор	B_EDIT	N(4)	внешний ключ (к Employees)

Таблица 1.3 – Схема отношения АВТОРЫ (Authors)

Содержание поля	Имя поля	Тип, длина	Примечания
Код автора	A_ID	N(4)	суррогатный первичный ключ
Фамилия, имя, отчество	A_NAME	C(50)	обязательное поле
Паспортные данные	A_PASSP	C(50)	обязательное поле
ИНН	A_INN	N(12)	уникальное поле
Адрес	A_ADDR	C(50)	обязательное поле
Телефоны	A_TEL	C(30)	многозначное поле

Таблица 1.4 – Схема отношения ЗАКАЗЫ (Orders)

Содержание поля	Имя поля	Тип, длина	Примечания
Номер заказа	O_ID	N(6)	первичный ключ
Заказчик	O_COMPANY	C(40)	обязательное поле
Дата поступления заказа	O_DATE	D	обязательное поле
Адрес заказчика	O_ADDR	C(50)	обязательное поле
Дата выполнения заказа	O_READY	D	

Таблица 1.5 – Схема отношения КНИГИ–АВТОРЫ (Titles)

Содержание поля	Имя поля	Тип, длина	Примечания
Код книги (№ контракта)	B_ID	N(6)	внешний ключ (к Books)
Код автора	A_ID	N(4)	внешний ключ (к Authors)
Номер в списке	A_NO	N(1)	обязательное поле
Гонорар	A_FEE	N(3)	процент от общего гонорара

Таблица 1.6 – Схема отношения КНИГИ–РЕДАКТОРЫ (Editors)

<i>Содержание поля</i>	<i>Имя поля</i>	<i>Тип, длина</i>	<i>Примечания</i>
Код книги (№ контракта)	B_ID	N(6)	внешний ключ (к Books)
Код редактора	E_ID	N(4)	внешний ключ (к Employees)

Таблица 1.7 – Схема отношения СТРОКИ ЗАКАЗА (Items)

<i>Содержание поля</i>	<i>Имя поля</i>	<i>Тип, длина</i>	<i>Примечания</i>
Номер заказа	O_ID	N(6)	внешний ключ (к Orders)
Код книги (№ контракта)	B_ID	N(6)	внешний ключ (к Books)
Количество	B_COUNT	N(4)	обязательное поле

### **Порядок выполнения работы и содержание отчета:**

1. Выбрать вариант задания
2. Провести инфологическое проектирование проанализировав предметную область согласно варианту задания. Разработать диаграмму «Сущность-связь»
3. Осуществить процесс логического проектирования, подробно расписав процесс преобразования диаграммы «Сущность-связь» в схему отношений.
4. Подготовить отчет о проделанной работе. Структура отчета:
  1. титульный лист;
  2. задание;
  3. описание процесса проектирования (инфологическое проектирование и логическое проектирование, аналогично примеру, представленному в данном методическом указании);
  4. заключение.

### **Варианты заданий:**

1. музей;
2. картинная галерея;
3. книжный склад;
4. компания по сбыту лекарственных препаратов;
5. кулинария;
6. минимаркет;
7. продуктовый магазин;
8. строительная компания;
9. таксопарк;
10. товары-почтой;



## **ПРАКТИЧЕСКАЯ РАБОТА № 12**

**Тема: Преобразование реляционной базы данных в сущности и связи**

**Цель работы:** выработать практические навыки моделирования предметной области и построении ER-модели данных, закрепить технологию проектирования БД, закрепить основные понятия теории реляционных баз данных, освоить технологию построения ER-диаграмм, научиться получать реляционные БД из ER-диаграмм

**Оборудование:** ПК, интернет, программное обеспечение – MS Word, MS Visio, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** По заданному описанию предметной области построить концептуальную модель базы данных.

1. Выделите типы сущностей;
2. Выделите типы связей и определите для них показатели кардинальности и степень участия сторон;
3. Выделите атрибуты и свяжите их типами сущностей и связей;
4. Определите потенциальные и первичные ключи сущностей;
5. Нарисуйте ER-диаграмму.
6. Проанализируйте информационные задачи и группы пользователей.

**Индивидуальные задания к практической работе**

Вариант 1. Задача – организация учебного процесса в вузе:

- Студенты: паспортные данные, адрес, дата зачисления, номер приказа, факультет, группа, является ли старостой, кафедра (специализация), изучаемые (изученные) предметы, оценки, задолженности, стипендия.
- Учебные курсы: название, факультет(ы), групп(ы), кафедра, семестр(ы), форма отчетности, число часов.
- Преподаватели: паспортные данные, адрес, телефон, фотография, кафедра, должность, ученая степень, начальник (зав. кафедрой), предмет(ы), число ставок, зарплата.

Вариант 2. Учет и выдача книг в библиотеке вуза:

- Книги: авторы, название, раздел (техническая, общественно-политическая и т.п.), место и год издания, издательство, количество страниц, иллюстрированность, цена, дата покупки, номер сопроводительного документа (чек, счет/накладная), вид издания (книги, учебники, брошюры, периодические издания), инвентарный номер (есть только для книг и некоторых учебников), длительность использования читателями (год, две недели, день), электронная версия книги или ее реферата (отсканированный текст).
- Читатели: номер читательского билета, ФИО, год рождения, адрес, дата записи, вид (студент, аспирант, преподаватель, сотрудник), курс, номер группы, названия взятых книг и даты их выдачи.

Вариант 3. Отдел кадров некоторой компании.

- Сотрудники: ФИО, паспортные данные, фотография, дом. и моб. телефоны, отдел, комната, раб. телефоны (в т.ч. местный), подчиненные сотрудники, должность, тип(ы) работы, задание(я), проект(ы), размер зарплаты, форма зарплаты (почасовая, фиксированная).

- Отделы: название, комната, телефон(ы), начальник, размер финансирования, число сотрудников.
- Проекты: название, дата начала, дата окончания, размер финансирования, тип финансирования (периодический, разовый), задачи и их исполнители, структура затрат и статьи расходов.

#### Вариант 4. Отдел поставок некоторого предприятия.

- Поставщики: название компании, ФИО контактного лица, расчетный счет в банке, телефон, факс, поставляемое оборудование (материалы), даты поставок (по договорам и реальные), метод и стоимость доставки.
- Сырье: тип, марка, минимальный запас на складе, время задержки, цена, продукты, при производстве которых используется, потребляемые объемы (необходимый, реальный, на единицу продукции).

Вариант 5. Задача – информационная поддержка деятельности гостиницы.

БД должна осуществлять:

- ведение списка постояльцев;
- учет забронированных мест;
- ведение архива выбывших постояльцев за последний год.

Необходимо предусмотреть:

- получение списка свободных номеров (по количеству мест и классу);
- получение списка номеров (мест), освобождающихся сегодня и завтра;
- выдачу информации по конкретному номеру;
- автоматизацию выдачи счетов на оплату номера и услуг;
- получение списка забронированных номеров;
- проверку наличия брони по имени клиента и/или названию организации

Вариант 6. Задача – информационная поддержка деятельности аптечного склада.

В аптечном складе хранятся лекарства. Сведения о лекарствах содержатся в специальной ведомости: наименование лекарственного препарата; количество (в шт.); цена; срок хранения на складе (в месяцах). Лекарства поступают на склад ежедневно от разных поставщиков, отпускаются два раза в неделю по предварительным заказам аптек. Выяснить, сколько стоит самый дорогой и самый дешевый препарат; сколько препаратов хранится на складе более 3 месяцев; сколько стоят все препараты, хранящиеся на складе, отыскать препараты, остаток которых равен нулю, ниже требуемого по заказам.

#### Вариант 7. Кинотеатры (информация для зрителей).

- Фильмы: название, описание, жанр (категория), длительность, популярность (рейтинг, число проданных билетов в России и в мире), показывается ли сейчас (сегодня, на текущей неделе), в каких кинотеатрах показывается, цены на билеты (в т.ч. средние).
- Кинотеатры: название, адрес, схема проезда, описание, число мест (в разных залах, если их несколько), акустическая система, широкоэкранность, фильмы и цены на них: детские и взрослые билеты в зависимости от сеанса (дневной,

вечерний и т.п.) и от категории мест (передние, задние и т.п.); сеансы показа фильмов (дата и время начала).

Вариант 8. Ресторан (информация для посетителей).

- Меню: дневное или вечернее, список блюд по категориям.
- Блюда: цена, название, вид кухни, категории (первое, второе и т.п.; мясное, рыбное, салат и т.п.), является ли вегетарианским, компоненты блюда, время приготовления, есть ли в наличии.
- Компоненты блюд: тип (гарнир, соус, мясо и т.п.), калорийность, цена, рецепт, время приготовления, есть ли в наличии, ингредиенты (продукты) и их расходы на порцию.

Вариант 9. Задача - информационная поддержка деятельности склада.

База данных должна содержать информацию о наименовании товара, его поставщике, количестве, цене товара, конечном сроке реализации, сроке хранения на складе. Торговый склад производит уценку хранящейся продукции. Если продукция хранится на складе дольше 10 месяцев, то она уценивается в 2 раза, а если срок хранения превысил 6 месяцев, но не достиг 10, то в 1,5 раза. Ведомость уценки товаров должна содержать информацию: наименование товара, количество товара(шт.), цена товара до уценки, срок хранения товара, цена товара после уценки, общая стоимость товаров после уценки.

Вариант 10. Задача – информационная поддержка деятельности адвокатской конторы. БД должна осуществлять:

- ведение списка адвокатов;
- ведение списка клиентов;
- ведение архива законченных дел.

Необходимо предусмотреть:

- получение списка текущих клиентов для конкретного адвоката;
- определение эффективности защиты (максимальный срок минус полученный срок) с учетом оправданий, условных сроков и штрафов;
- определение неэффективности защиты (полученный срок минус минимальный срок);
- подсчет суммы гонораров (по отдельным делам) в текущем году;
- получение для конкретного адвоката списка текущих клиентов, которых он защищал ранее (из архива, с указанием полученных сроков и статей).

## **ПРАКТИЧЕСКАЯ РАБОТА № 13**

**Тема: Задание ключей. Создание основных объектов БД**

**Цель работы:** научить создавать ключи, создавать основные объекты БД.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** Спроектировать базу данных предметной области "Библиотека".

### 1. Назначение и предметная область

База данных предназначена для хранения данных о приобретенных библиотекой изданиях (монографиях, справочниках, сборниках статей и т.п.), информации о местонахождении отдельных экземпляров (переплетов) каждого издания и сведений о читателях.

Для ведения библиотечных каталогов, организации поиска требуемых изданий и библиотечной статистики в базе должны храниться сведения, большая часть которых размещаются в аннотированных каталожных карточках.

Анализ запросов на литературу (как читателями, так и сотрудниками библиотек) показывает, что для поиска подходящих изданий (по тематике, автору, художнику, издательству и т.п.) и отбора нужного (например, по аннотации) следует выделить следующие атрибуты каталожной карточки:

1. Автор (фамилия и имена (инициалы) или псевдоним каждого автора издания).
2. Название (заглавие) издания.
3. Номер тома (части, книги, выпуска).
4. Вид издания (сборник, справочник, монография, ...).
5. Составитель (фамилия и имена (инициалы) каждого из составителя издания).
6. Язык, с которого выполнен перевод издания.
7. Переводчик (фамилия и инициалы каждого переводчика).
8. Под чьей редакцией (фамилия и имена (инициалы) каждого из титульных редакторов).
9. Художник (фамилия и имена (инициалы) каждого художника-иллюстратора) – для художественных изданий, иллюстрируемых оригинальными рисунками.
10. Повторность издания (второе, одиннадцатое и т.п.).
11. Характер переиздания (исправленное, дополненное, переработанное, стереотипное и т.п.).
12. Место издания (город).
13. Издательство (название издательства).
14. Год выпуска издания.
15. Издательская аннотация или реферат.
16. Библиотечный шифр (например, ББК 32.973).
17. Авторский знак (например, Д27).

Библиотечный шифр и авторский знак используются при составлении каталогов и организации расстановки изданий на полках: по содержанию (в соответствии с библиотечным шифром) и алфавиту (в соответствии с авторским знаком).

Библиотечно-библиографическая классификация (ББК) распределяет издания по отраслям знания в соответствии с их содержанием. В ней используется цифро-буквенные индексы ступенчатой структуры. Каждый из девяти классов (1. Марксизм-ленинизм; 2. Естественные науки; 3. Техника. Технические науки; 4. Сельское и лесное хозяйство; 5. Здравоохранение; 6-8. Общественные и гуманитарные науки; 9. Библиографические пособия. Справочные издания. Журналы.) делится на подклассы и следующие ступени деления:

3. Техника. Технические науки.

32. Радиоэлектроника.

32.97. Вычислительная техника.

32.973. Электронные вычислительные машины и устройства.

32.973.2. Электронно-вычислительные машины и устройства дискретного действия.

Шифр ББК используется при выделении хранимым изданиям определенных комнат, стеллажей и полок, а также для составления каталогов и статистических отчетов.

Авторский знак, состоящий из первой буквы фамилии (псевдонима) автора или названия издания (для изданий без автора) и числа, соответствующего слогу, наиболее приближающегося по написанию к первым буквам фамилии (названия), упрощает расстановку книг на полках в алфавитном порядке.

К объектам и атрибутам, позволяющим охарактеризовать отдельные экземпляры изданий (переплеты), места их хранения и читателей, можно отнести:

1. Номер комнаты (помещения для хранения переплетов).
2. Номер стеллажа в комнате.
3. Номер полки на стеллаже.
4. Номер (инвентарный номер) переплета.
5. Дата приобретения конкретного переплета.
6. Цена конкретного переплета.
7. Дата размещения конкретного переплета на конкретном месте.
8. Дата изъятия переплета с установленного места.
9. Номер читательского билета (формуляра).
10. Фамилия читателя.
11. Имя читателя.
12. Отчество читателя.
13. Адрес читателя.
14. Телефон читателя.
15. Дата выдачи читателю конкретного переплета.
16. Срок, на который конкретный переплет выдан читателю.
17. Дата возврата переплета.

Описание предметной области можно свести к заполнению следующей таблицы:

Таблица 1. Описание предметной области

Объект	Атрибут	Описание атрибута
1.	1.1.	
	1.2.	
2.	2.1.	
	2.2.	

## 2. Построение инфологической модели

Анализ определенных выше объектов и атрибутов позволяет выделить сущности проектируемой базы данных и, приняв решение о создании реляционной базы данных, построить ее инфологическую модель на языке «Таблицы-связи».

К стержневым сущностям можно отнести:

1. Создатели (Код создателя, Создатель). Эта сущность отводится для хранения сведений об основных людях, принимавших участие в подготовке рукописи издания (авторах, составителях, титульных редакторах, переводчиках и художниках). Такое объединение допустимо, так как данные о разных создателях выбираются из одного домена (фамилия и имена) и исключает дублирование данных (один и тот же человек может играть разные роли в подготовке разных изданий). Например, С.Я.Маршак писал стихи (Сказка о глупом мышонке) и пьесы (Двенадцать месяцев), переводил Дж.Байрона, Р.Бернса, Г.Гейне и составлял сборники стихов.

Так как фамилия и имена (инициалы) создателя могут быть достаточно громоздкими (М.Е. Салтыков-Щедрин, Франсуа Рене де Шатобриан, Остен Жюль Жан-Батист Ипполит и т.п.) и будут многократно встречаться в разных изданиях, то их целесообразно нумеровать и ссылаться на эти номера. Для этого вводится целочисленный атрибут "Код\_создателя", который будет автоматически наращиваться на единицу при вводе в базу данных нового автора, переводчика или другого создателя.

Аналогично создаются: Код\_издательства, Код\_заглавия, Вид\_издания, Код\_характера, Код\_языка, Номер\_билета, Номер\_переплета, Код\_места и Код\_издания, замещающие от одного до девяти атрибутов.

2. Издательства (Код\_издательства, Название, Город).

3. Заглавия (Код\_заглавия, Заглавие). Выделение этой сущности позволит сократить объем данных и снизить вероятность возникновения противоречивости (исключается необходимость ввода длинных текстовых названий для различных томов собраний сочинений, повторных изданий, учебников и т.п.).

4. Вид\_издания (Вид\_издания, Название\_вида).

5. Характеры (Код\_характера, Характер\_переиздания).

6. Языки (Код\_языка, Язык, Сокращение). Кроме названия языка хранится его общепринятое сокращение (англ., исп., нем., фр.), если оно существует.

7. Места (Код\_места, Номер\_комнаты, Номер\_стеллажа, Номер\_полки). Один из кодов этой сущности (например, "-1") отведен для описания обобщенного места, находящегося за стенами хранилища книг (издание выдано читателю, временно передано другой библиотеке или организации).

8. Читатели (Номер\_билета, Фамилия, Имя, Отчество, Адрес, Телефон).

Две ключевые сущности, описывающие издание и его конкретные экземпляры, оказываются зависимыми от других сущностей и попадают в класс обозначений:

1. Издание (Код\_издания, Код\_заглавия, Вид\_издания, Номер\_тома, Авторский\_знак, Библиотечн\_шифр, Повторность, Код\_издательства, Год\_издания, Аннотация) [Заглавия, Вид\_издания, Издательства];
2. Переплеты (Номер\_переплета, Код\_издания, Цена, Дата\_приобретения) [Издания]

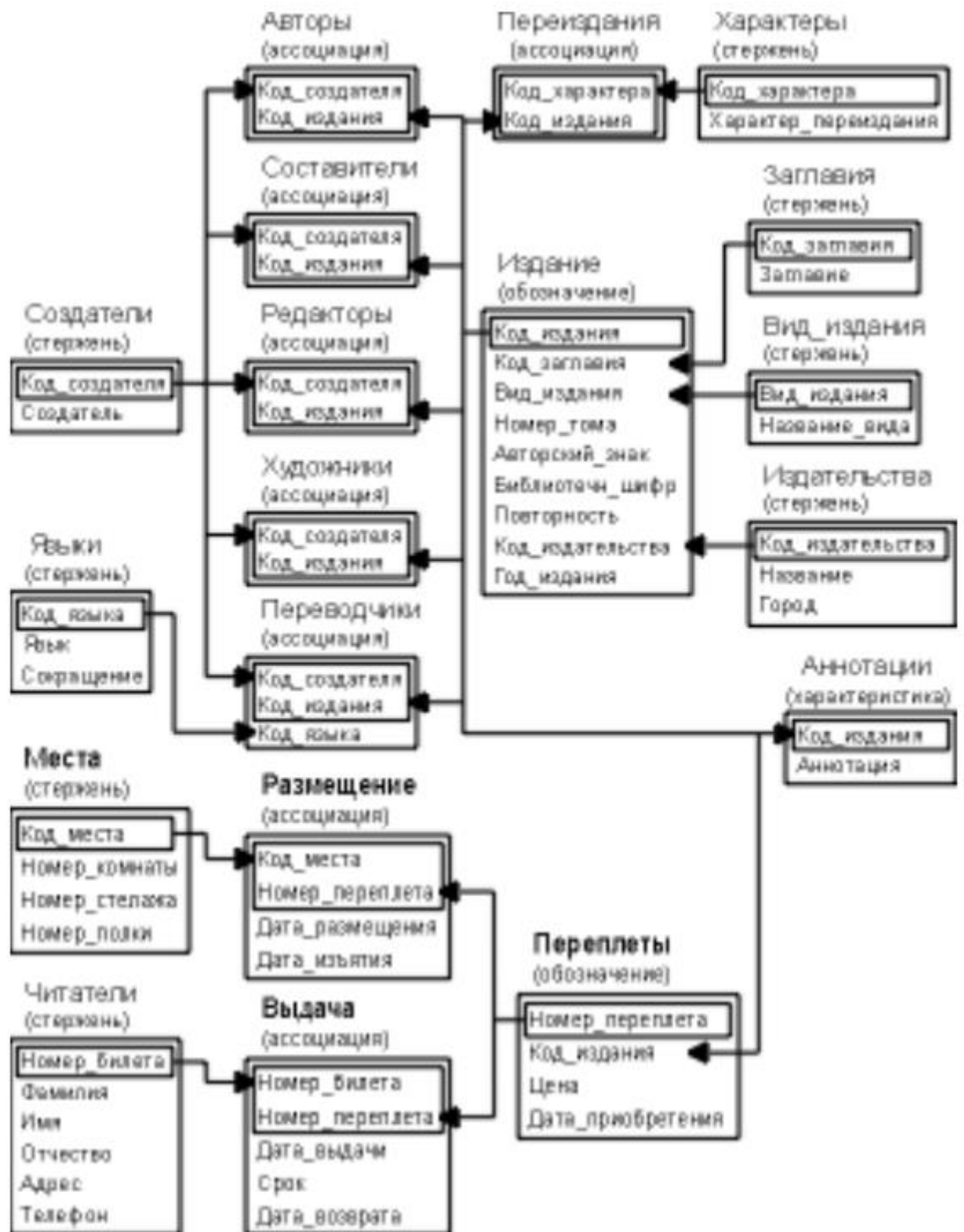
Стержневые сущности и обозначения связаны между собой ассоциациями:

1. Авторы [Создатели М, Издание N] (Код\_создателя, Код\_издания).
2. Составители [Создатели М, Издания N] (Код\_создателя, Код\_издания).
3. Редакторы [Создатели М, Издания N] (Код\_создателя, Код\_издания).
4. Художники [Создатели М, Издания N] (Код\_создателя, Код\_издания).
5. Переводчики [Создатели М, Издания N] (Код\_создателя, Код\_издания, Язык).
6. Переиздания [Характеры М, Издания N] (Код\_характера, Код\_издания).
7. Размещение [Места М, Переплеты N] (Код\_места, Номер\_переплета, Дата\_размещения, Дата\_изъятия).
8. Выдача [Читатели М, Переплеты N] (Номер\_билета, Номер\_переплета, Дата\_выдачи, Срок, Дата\_возврата).

И, наконец, для уменьшения объема часто используемого обозначения "Издания" из него выделена характеристика:

1. Аннотации (Код\_издания, Аннотация) {Издание}.

Инфологическая модель базы данных "Библиотека", построенная с помощью языка "Таблицы-связи" представлена на рисунке.



**Задание 2.** Спроектировать базу данных для представленной предметной области.

Пусть требуется создать программную систему, предназначенную для директора продовольственного магазина. Такая система должна обеспечивать хранение сведений о магазине, об имеющихся в нем товарах, о торговых базах и товарах, хранящихся на этих базах. Магазин осуществляет закупку товаров на разных базах, предпочитая при этом закупать одни виды товара на одних базах, а другие на других. Магазин характеризуется классом, номером и имеет



несколько отделов. Каждый товар в каждом магазине продается, по крайней мере, в одном отделе.

Каждый отдел имеет заведующего. Товары, имеющиеся в магазине и хранящиеся на базах, характеризуются ценой, сортом и количеством. Розничные цены в магазине зависят от класса магазина.

Директор магазина должен иметь возможность изменить цену товара по своему усмотрению, осуществить закупку недостающего товара на базе. Он может также закрыть один из отделов или открыть новый, при этом товары могут перемещаться из отдела в отдел. Директору могут потребоваться следующие сведения:

- Какие товары имеются в магазине (на базе)?
- Какие отсутствующие товары может заказать магазин на базе?
- Какие товары, и в каком количестве имеются в отделе магазина?
- Список заведующих отделами магазина?
- Суммарная стоимость товара в каждом отделе?
- На каких базах, и в каких количествах есть товар нужного наименования?

Необходимо предусмотреть возможность выдачи документа, представляющего собой заявку на закупку товара на базе, и создания ежемесячного отчета о работе магазина с подсчетом прибыли. Отчет, сгруппированный по отделам, должен содержать перечень товаров, закупленных в отчетный месяц на базах (количество, наименование и сорт товара), а также перечень проданных товаров.

## ПРАКТИЧЕСКАЯ РАБОТА № 14

### Тема: Создание проекта БД. Создание БД. Редактирование и модификация таблиц

**Цель работы:** Получение навыков работы по созданию структуры таблиц, модификации структуры таблиц, заполнению таблиц.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### **Справочный материал:**

##### *Разработка структуры БД*

Выполнение начинается с разработки структуры БД. На этом этапе должны быть детально проанализированы условия задания и, на их основе, определено количество таблиц, необходимых для описания всех характеристик анализируемой предметной области. Кроме того, необходимо определить какие поля в таблицах будут использованы в качестве ключевых, а также определить каким образом будет осуществляться связь между таблицами. Если невозможно установить связи посредством использования ключевых полей, определить таблицы, которые будут использоваться только для связи между другими таблицами.

##### *Создание таблиц.*

Для каждого поля конкретной таблицы необходимо определить его тип и размер и тщательно проверить, удовлетворяет ли диапазон значений выбранного типа тем значениям, которые может реально принимать данное поле. При необходимости, для некоторых полей можно установить *Условие на значение* и задать сообщение, выдаваемое на экран в случае несоответствия введенного значения заданному условию или присвоить значения, принимаемые по умолчанию. Можно также определить формат вводимой информации для конкретных полей. Заполнить соответствующей информацией каждый из разделов создаваемой структуры таблицы: *Имя поля*, *Тип данных* и *Описание*. Раздел описаний необязателен для заполнения, но информация, введенная в данный раздел отображается в строке состояния при вводе данных для конкретного поля, облегчая процесс ввода.

##### *Сохранение таблиц*

По окончании создания структуры таблицы в режиме *Конструктора* ее необходимо сохранить. Для сохранения выполнить: *Файл/Сохранить*.

##### *Заполнение таблиц.*

Открыть таблицу в режиме *Таблицы*. Заполнить необходимой информацией, подготовив для заполнения не менее десяти записей для основной таблицы. Сохранение не требуется, т.к. сохранение производится сразу при переходе к следующей записи. Закрыть заполненную таблицу. Аналогично поступить с остальными таблицами.

##### *Завершение работы с БД.*

Для завершения работы с БД необходимо закрыть окно БД, затем закрыть окно приложения.

#### **Содержание работы:**

**Задание 1.** Создать структуры таблиц, ключевые и индексные поля. Заполнить таблицы данными, установить связи, удалить данные, восстановить их.

**Постановка задачи:** Создать базу данных ОТДЕЛ КАДРОВ, поместив в нее три таблицы:

СОТРУДНИК, СОСТАВ СЕМЬИ и ШТАТНОЕ РАСПИСАНИЕ, содержащие информацию о сотрудниках предприятия.

**Описание прикладной области Отдел кадров предприятия.**

Анализ предметной области показывает, что для автоматизации работы отдела кадров целесообразно создать базу данных ОТДЕЛ КАДРОВ, состоящую из трех таблиц: СОТРУДНИК, СОСТАВ СЕМЬИ, ШТАТНОЕ РАСПИСАНИЕ. Таблицы будут связаны между собой следующим образом: Таблица СОТРУДНИК с таблицей СОСТАВ СЕМЬИ связываются по полю Идент код, а с таблицей ШТАТНОЕ РАСПИСАНИЕ - по полю Должн.

Наименование таблицы	Структура таблицы
Сотрудник	Идентификационный код, Фамилия, Имя, Отчество, Пол, Дата рождения, Место рождения, Образование, Должность, Стаж работы, Семейное положение, Дата зачисления на работу, Телефон, Домашний адрес
Состав семьи	Идентификационный код, Взаимоотношения, Фамилия, Имя, Отчество, Год рождения
Штатное расписание	№ по порядку, Название подразделения, Должность, Количество штатных единиц, Должностной оклад, Фонд з/платы за месяц, Фонд з/платы за год.

Состав и характеристика полей таблицы “Штатное расписание”.

Название поля	Имя поля	Характеристики поля	
		Тип данных	Возможности
№ по порядку	НПП	Числовой	Длинное целое, обязательное
Название подразделения	Назв подраз	Текстовый	30 символов, обязательное
Должность	Должность	Текстовый	15 символов, обязательное
Количество штатных единиц	Кол ед	Числовой	Длинное целое, обязательное
Должностной оклад	Оклад	Числовой	Длинное целое, обязательное
Фонд з/платы за месяц	ФЗПМ	Числовой	Длинное целое, обязательное
Фонд з/платы за год	ФЗПГ	Числовой	Длинное целое, обязательное

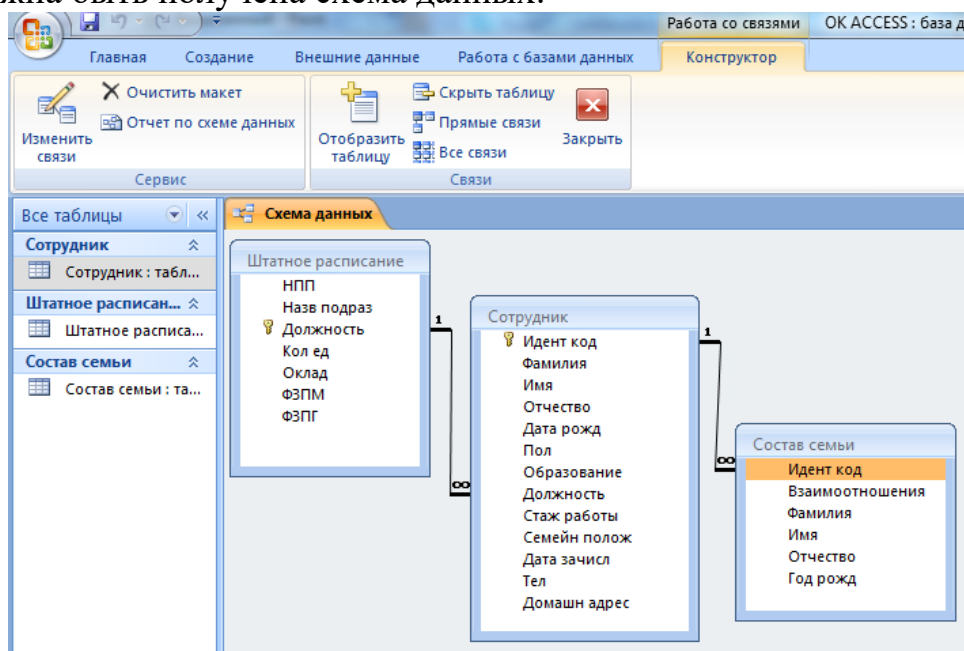
Состав и характеристика полей таблицы “Сотрудник”.

Название поля	Имя поля	Характеристики поля	
		Тип данных	Возможности
Идентификационный код	<u>Идент код</u>	Текстовый	10 символов, обязательное
Фамилия	Фамилия	Текстовый	20 символов, обязательное
Имя	Имя	Текстовый	15 символов, обязательное
Отчество	Отчество	Текстовый	15 символов, обязательное
Пол	Пол	Текстовый	50 символов, не обязательное
Дата рождения	Дата рожд	Дата/время	Маска ввода 00.00.0000, не обязательное
Место рождения	Место рожд	Текстовый	15 символов, не обязательное
Образование	Образование	Текстовый	15 символов, обязательное
Должность	Должность	Мастер подстановок	15 символов, индексированное, допускается совпадение, обязательное
Стаж работы	Стаж работы	Числовой	Длинное целое, обязательное
Семейное положение	Семейн полож	Текстовый	11 символов, не обязательное
Дата зачисления на работу	Дата зачисл	Дата/время	Маска ввода 00.00.0000, не обязательное
Телефон	Тел	Текстовый	8 символов, не обязательное
Домашний адрес	Домашн адрес	Поле МЕМО	не обязательное

Состав и характеристика полей таблицы “Состав семьи”.

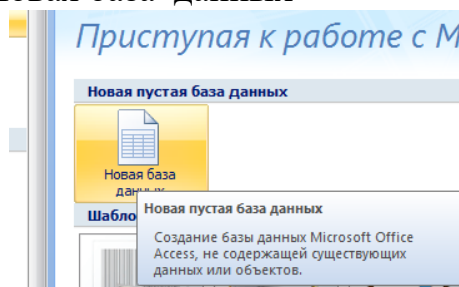
Название поля	Имя поля	Характеристики поля	
		Тип данных	Возможности
Идентификационный код	Идент код	Мастер подстановок	10 символов, обязательное
Взаимоотношения	Взаимоотношения	Текстовый	10 символов, не обязательное
Фамилия	Фамилия	Текстовый	20 символов, обязательное
Имя	Имя	Текстовый	15 символов, обязательное
Отчество	Отчество	Текстовый	15 символов, обязательное
Год рождения	Год рожд	Дата/время	Маска ввода 00.00.0000, обязательное

Должна быть получена схема данных:

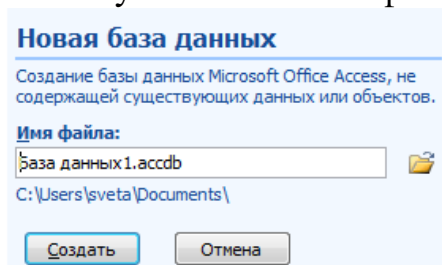



#### Порядок выполнения:

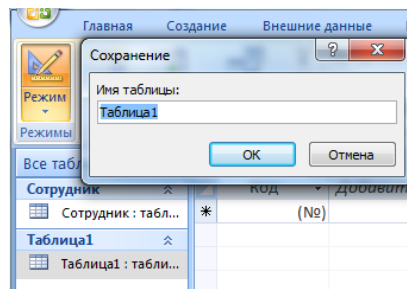
1. Запустить Microsoft Access
2. щелкаем на пиктограмме Новая база данных



3. В правой части окна появится информация об имени файла и указана директория для его хранения. По умолчанию имя файла - **База данных1.accdb**.

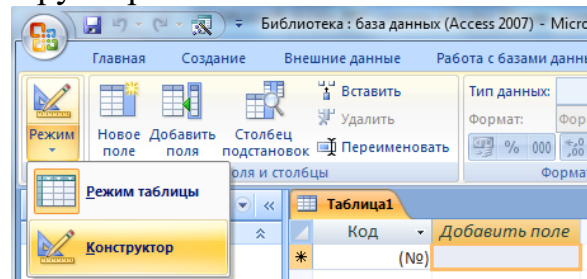


4. Далее щелкнуть справа по пиктограмме  и ввести имя файла *Библиотека*
- и в верхней части окна открыть свою папку:
5. В результате получаем:

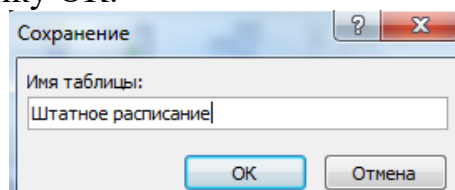


6. Нажимаем кнопку *Создать*

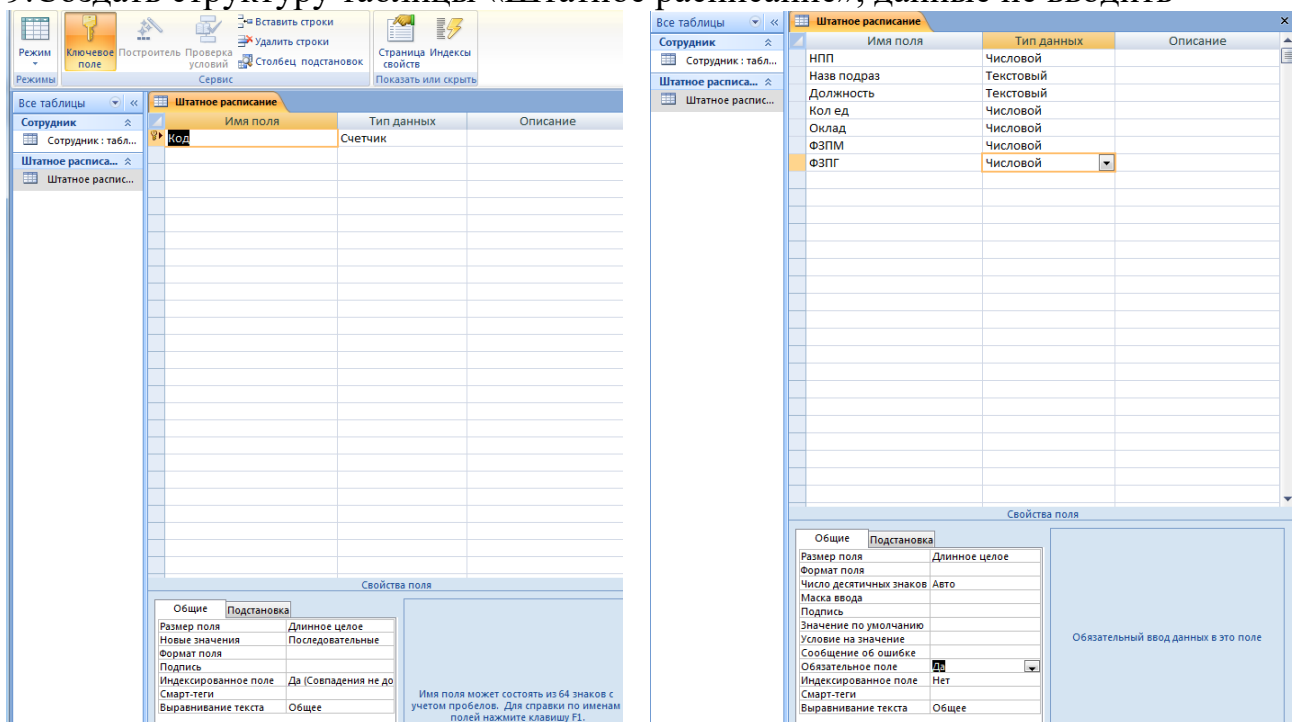
7. Далее необходимо перейти в режим Конструктор и создать структуру первой таблицы базы данных. Для этого необходимо щелкнуть на пиктограмме Режим и выбрать режим Конструктор.



8. Откроется окно Сохранение, в котором надо указать имя Штатное расписание и нажать кнопку ОК.

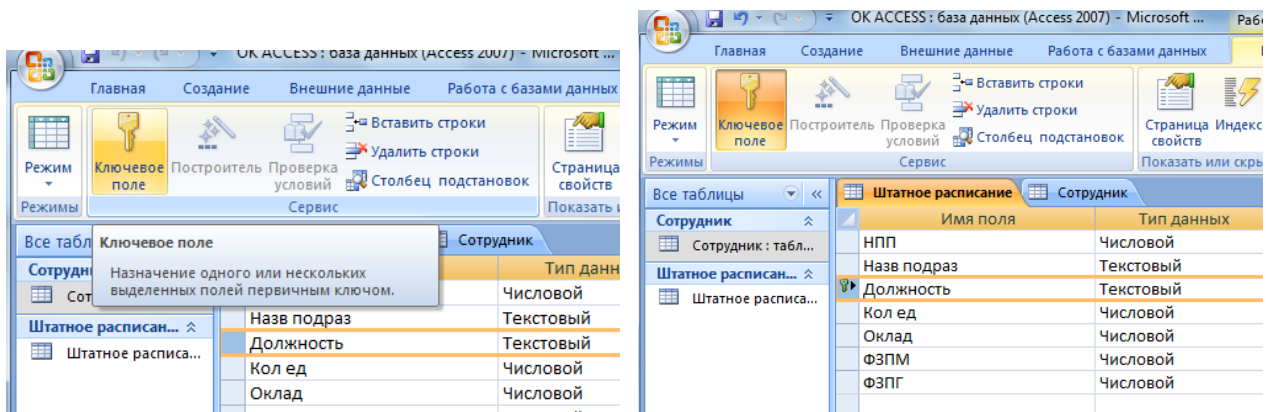


9. Создать структуру таблицы «Штатное расписание», данные не вводить



После создания структуры таблицы необходимо задать ключевое поле.

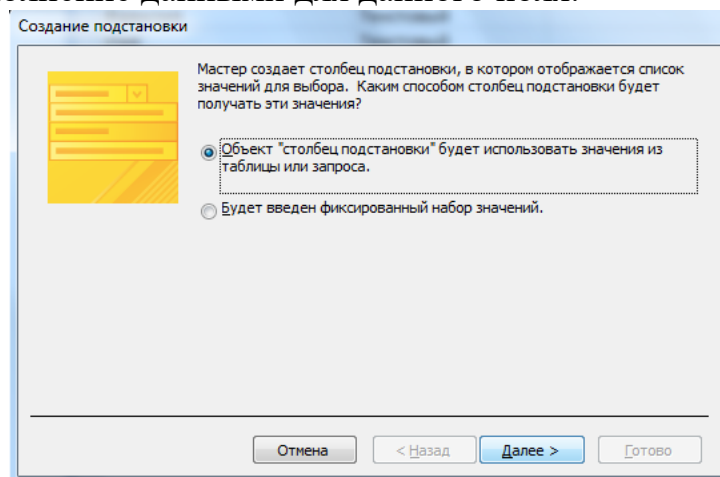
Как ключевое поле выбираем поле **Должность**, т.к. оно не содержит записей, что повторяются, а также будет использовано для связи с таблицей «Сотрудник».



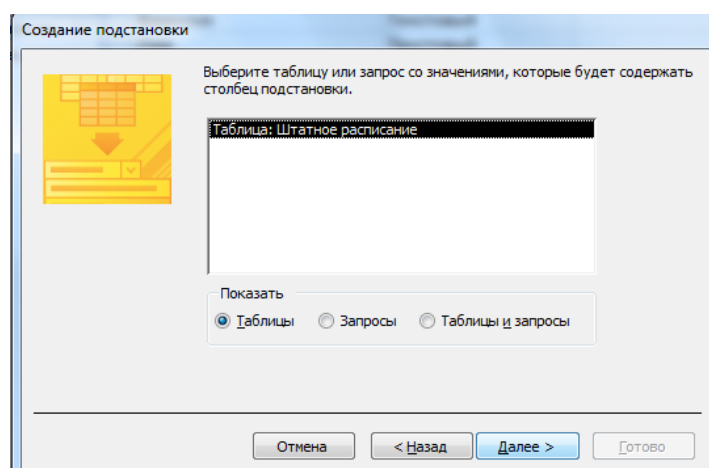
10. Создать структуру таблицы «Сотрудник», данные не вводить:

- меню Создание
- Таблица
- Конструктор
- имя Сотрудник и т.д.

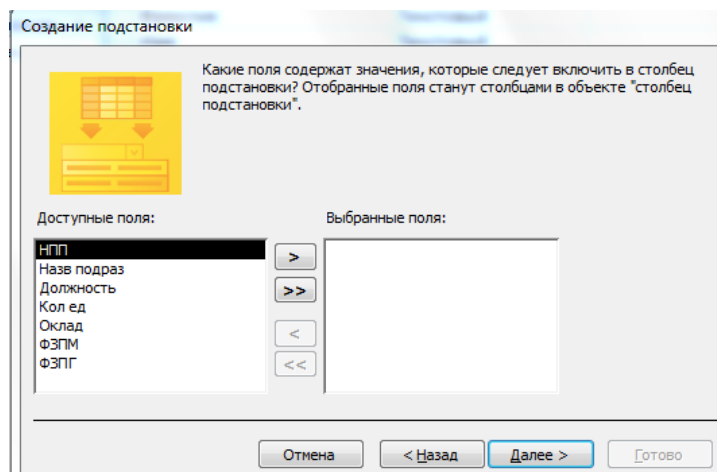
Для поля **Должность** выбираем тип **Мастер подстановок**. Это позволит облегчить заполнение данными для данного поля.



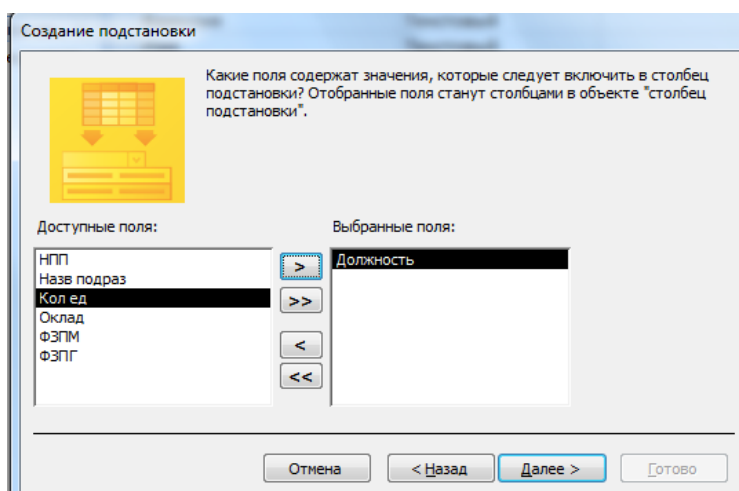
Далее



Далее

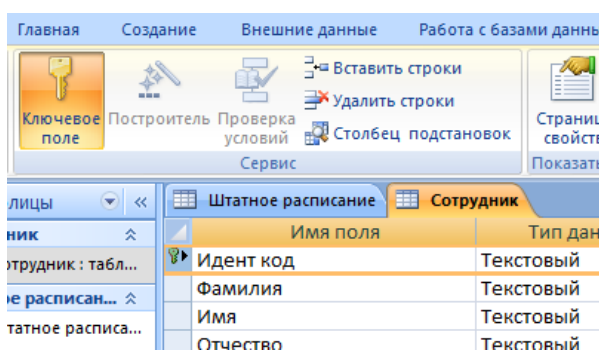


Далее



Готово

Как ключевое поле выбираем поле **Идент код**, т.к. оно не содержит записей, что повторяются, а также будет использовано для связи с таблицей "Состав семьи".



11. Создать структуру таблицы «Состав семьи», данные не вводить:

При создании поля **Идент код** как столбец подстановки используется поле **Идент код** из таблицы "Сотрудник".



Создание подстановки

Выберите таблицу или запрос со значениями, которые будут содержать столбец подстановки.

Таблица: Сотрудник  
Таблица: Штатное расписание

Показать  
☒ Таблицы  
☐ Запросы  
☐ Таблицы и запросы

Отмена < Назад Далее > Готово

Создание подстановки

Какие поля содержат значения, которые следует включить в столбец подстановки? Отобранные поля станут столбцами в объекте "столбец подстановки".

Доступные поля: Фамилия, Имя, Отчество, Дата рожд, Пол, Образование, Должность, Стаж работы

Выбранные поля: Идент код

Отмена < Назад Далее > Готово

Поле **Иден код** выбрать как индексное поле. Для этого в разделе **Свойства поля** выбрать строку **Индексированное поле** и выбрать из выпадающего списка **Да (допускаются совпадения)**.

Общие	Подстановка
Размер поля	10
Формат поля	
Маска ввода	
Подпись	
Значение по умолчанию	
Условие на значение	
Сообщение об ошибке	
Обязательное поле	Нет
Пустые строки	Да
Индексированное поле	Нет
Сжатие Юникод	Нет
Режим IME	Да (Допускаются совпадения)
Режим предложений IME	Да (Совпадения не допускаются)
Смарт-теги	

Таблицы будут связаны между собой таким образом: таблица **Сотрудник** с таблицей **Состав семьи** связываются по полю **Идент код**, с таблицей **Штатное расписание** – по полю **Должность**.

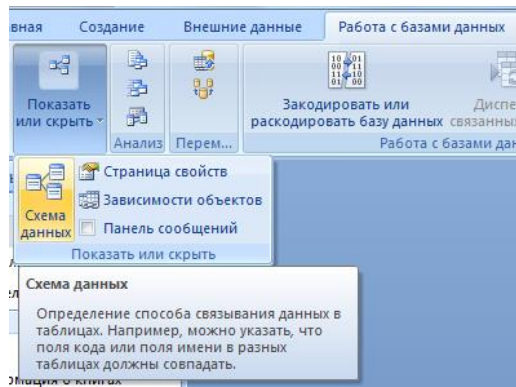
12.Закрывать все созданные структуры таблиц

**Задание 2.** Создать связи между таблицами.

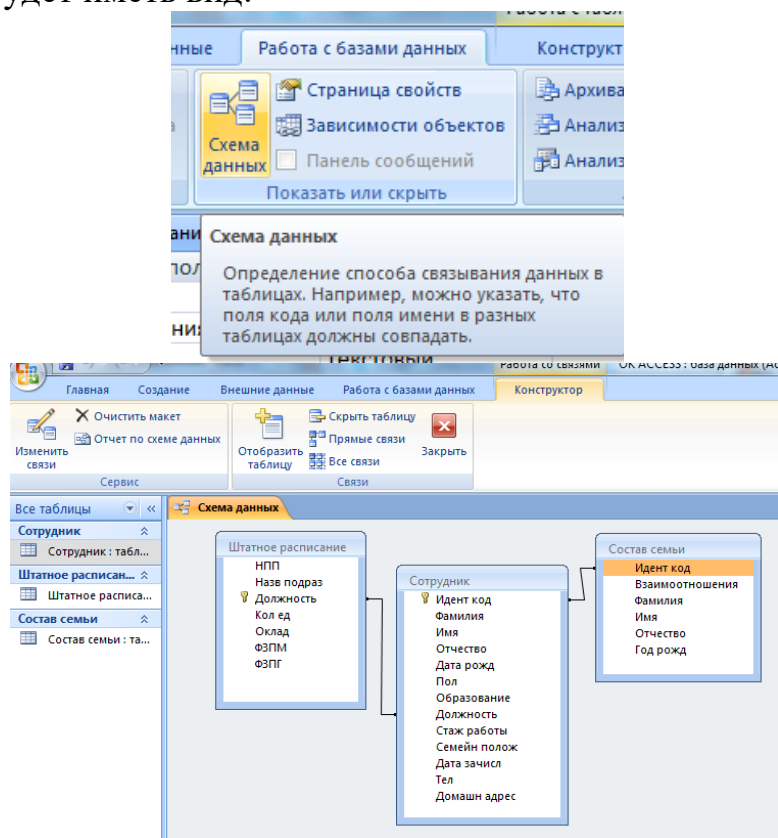
1.Создание связей между таблицами:

- меню Работа с базами данных

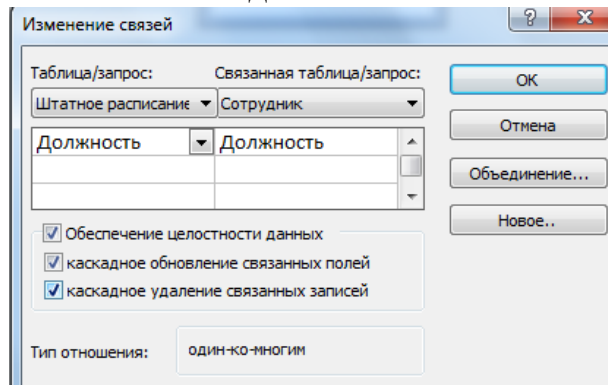
- Показать или скрыть



- Схема данных, появится окно Добавление таблицы
- Выделить все таблицы, Добавить каждую
- кнопка Заккрыть
- Схема данных будет иметь вид:



Дважды щелкнуть по линии соединения



Изменение связей

Таблица/запрос: Связанная таблица/запрос:

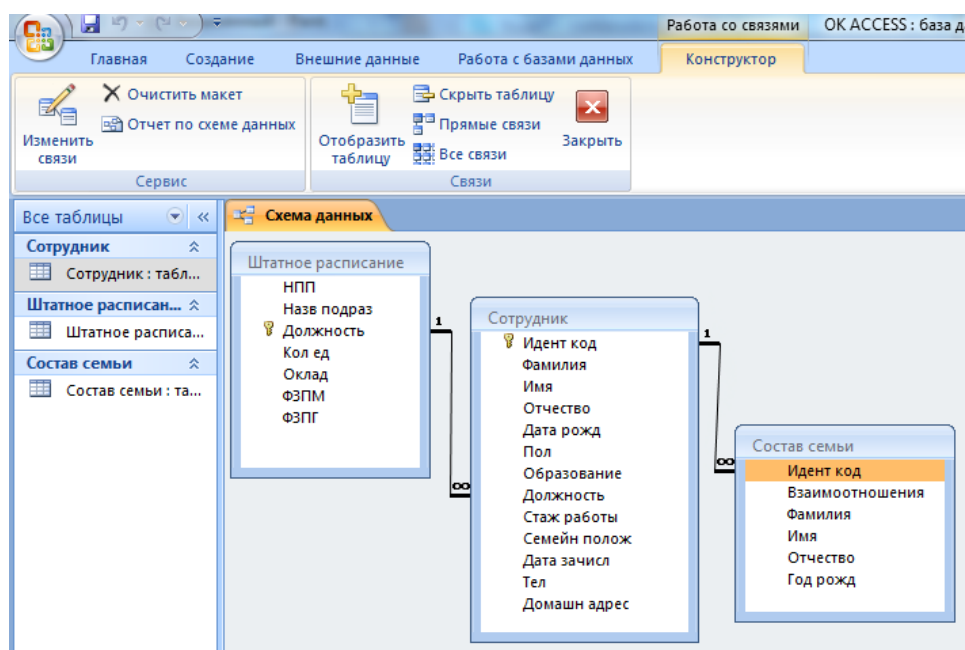
Сотрудник Состав семьи

Идент код Идент код

☒ Обеспечение целостности данных  
☒ каскадное обновление связанных полей  
☒ каскадное удаление связанных записей

Тип отношения: один-ко-многим

OK  
Отмена  
Объединение...  
Новое..



**Задание 3.** Внести данные во все таблицы

Таблица-объект ШТАТНОЕ РАСПИСАНИЕ

Нпп	Назв подр	Должн	Кол ед	Оклад	ФЗПМ	ФЗПГ
1	Дирекция	бухгалтер	2	230	460	5520
2	Дирекция	гл.бухгалтер	1	430	430	5160
3	Дирекция	директор	1	530	530	6360
4	Уч.кафедра	диспетчер	1	100	100	1200
5	Уч.кафедра	доцент	1	500	500	6000
6	Уч.кафедра	зав.кафедрой	1	430	430	5160
7	Дирекция	зам.директора	1	500	500	6000
8	Уч.кафедра	методист	2	200	400	4800
9	Дирекция	начальник ОК	1	150	150	1800
10	Уч.кафедра	преподаватель	4	350	1800	21600
11	Уч.кафедра	статистик	1	100	100	1200
12	Уч.кафедра	специалист	2	150	300	3600

Таблица-объект СОТРУДНИК

Идент код	Фамилия	Имя	Отчество	Пол	Дата рожд	Место рожд	Образов	Должн	Сем полож	Дата зач	Телефон
1314152347	Старченко	Светлана	Борисовна	ж	22.04.43	г.Казань	высшее	Бухгалтер	замужем	24.09.90	65-12-13
1545678990	Архипов	Сергей	Иванович	м	23.03.49	г.Харьков	высшее	Гл. бухгалтер	женат	10.12.88	нет
1624790203	Круговой	Геннадий	Иванович	м	22.04.45	г.Омск	высшее	Директор	вдовец	10.12.88	68-14-13
2748576413	Царева	Анна	Николаевна	ж	30.07.50	г.Харьков	ср.техн.ич.	Диспетчер	замужем	01.01.96	47-23-15
2934789231	Каменев	Татьяна	Дмитриевна	ж	24.06.59	г.Курск	высшее	Доцент	замужем	30.12.90	65-67-72
2955443781	Безродный	Владимир	Михайлович	м	05.09.53	г.Харьков	высшее	Зав. кафедрой	женат	01.09.92	32-32-14
1014654788	Садчиков	Аркадий	Викторович	м	10.01.57	г.Тамбов	высшее	Зам. директора	холост	10.12.88	10-12-10
2055894321	Бронзов	Станислав	Иванович	м	12.11.60	г.Москва	ср.техн.ич.	Методист	женат	31.08.94	23-10-70
1178943214	Мапошенко	Юрий	Николаевич	м	21.11.64	г.Омск	высшее	Начальник ОК	женат	31.08.94	43-35-13
2200987654	Коваль	Александр	Николаевна	ж	31.03.65	г.Киев	высшее	Преподаватель	замужем	01.10.92	47-67-33
2233668943	Строков	Олег	Викторович	м	05.08.65	г.Орел	ср.техн.ич.	Статистик	женат	10.09.92	69-05-03
2314743296	Бородулин	Андрей	Васильевич	м	31.12.69	г.Киев	высшее	Специалист	холост	31.08.95	27-14-12

МЕМО-поле Таблицы СОТРУДНИК

Адрес
ул.Гв.Широнинцев 21,кв.30
пер.Хрустальный 8
ул.Светлая 14,кв.55
ул.Артема 24, кв.1
ул.Героев труда 28-Б,кв.76
пр.Правды 44, кв.55
пер.Короленко 2, кв.1
ул. Революции 6, кв.2
ул.Пушкинская 54,кв2
ул.Иванова 5, кв.2
пр. Косиора 162, кв161
пр.Гагарина 117, кв.20

Таблица-объект СОСТАВ СЕМЬИ

Идент код	Отношение	Фамилия	Имя	Отчество	Дата рожд
1314152347	отец	Старченко	Николай	Иванович	12/01/1917
1314152347	мать	Старченко	Людмила	Яковлевна	25/12/1920
1545678990	сын	Архипов	Дмитрий	Сергеевич	01/09/1988
2748576413	муж	Царев	Петр	Алексеевич	14/11/1948
2934789231	муж	Каменев	Александр	Иванович	15/02/1952
2955443781	дочь	Безродная	Алла	Владимировна	24/06/1991
1014654788	мать	Садчикова	Мария	Ивановна	29/04/1930
2055894321	дочь	Бронзова	Инна	Станиславовна	15/12/1998
1178943214	сын	Мапошенко	Игорь	Юрьевич	22/06/1992
1178943214	сын	Мапошенко	Владимир	Юрьевич	23/08/1995
2233668943	дочь	Строкова	Юлия	Олеговна	28/07/1985
2233668943	дочь	Строкова	Наталия	Олеговна	14/03/1990

## ПРАКТИЧЕСКАЯ РАБОТА № 15

### Тема: Создание проекта БД. Создание БД. Редактирование и модификация таблиц

**Цель работы:** Получение навыков работы по созданию структуры таблиц, модификации структуры таблиц, заполнению таблиц.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### Содержание работы:

**Задание 1.** Создать БД «Студенты».

1. Вызвать программу MS Access.
2. В окне системы управления базы данных щелкнуть по значку *Новая база данных*. Справа в появившемся окне дать имя новой базе данных «Студенты» и щелкнуть по значку папки, находящемуся справа от окна названия. Откроется окно сохранения, найдите свою папку и сохраните в нее новый файл базы данных «Студенты». Затем нажмите на кнопку «Создать».
3. Появится окно *Таблица* (Рисунок 1).

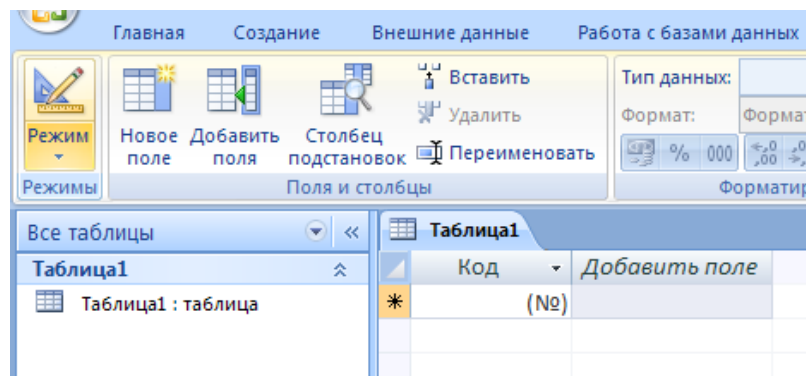




Рисунок 1

4. В появившемся окне откройте меню команды **Режим** и выберите вариант **Конструктор**  и сохраните будущую таблицу под названием **Ведомость успеваемости**. Появится окно Конструктора.
5. Заполните поля в **Конструкторе** данными из *таблицы 1*. Тип данных можно выбрать из меню, появившемся при нажатии на кнопку  в ячейке справа.

**Обратите внимание:** ключевое поле «Счетчик» внесен в таблицу автоматически. Если напротив поля отсутствует значок ключа, то на панели инструментов щелкните по этому значку.



Таблица 1.

Ведомость успеваемости	
Имя поля	Тип данных
Код	Счетчик
Фамилия	Текстовый
Имя	Текстовый
Математика	Числовой
Менеджмент	Числовой
Сервисная деятельность	Числовой
Информационные технологии	Числовой
Стандартизация	Числовой
Гостиничная индустрия	Числовой
Пропуски по неуважительной	Числовой
Пропуски по уважительной п	Числовой

6. Перейдите в режим таблицы, щелкнув по кнопке **Режим** на панели инструментов. Введите данные в этом режиме, заполняя клетки таблицы. Значение поля **Код** будет меняться автоматически.

7. Заполните базу данных значениями из *таблицы 2*. Напротив каждой фамилии выставьте по всем дисциплинам оценки от 2 до 5


Таблица 2

Код	Фамилия	Имя	Математика	Менеджмент	Сервисная деятельность	Информационные технологии	Стандартизация	Гостиничная индустрия	Пропуски по неуважительной причине	Пропуски по уважительной причине
1	Иванникова	Анна								
2	Баранова	Ирина								
3	Корнилова	Ольга								
4	Воробьев	Алексей								
5	Воробьев	Олег								
6	Скоркин	Александр								
7	Володина	Нина								
8	Новоселов	Алексей								
9	Петрова	Елена								
10	Чернова	Кристина								
11	Терецинка	Инна								
12	Истратов	Максим								
13	Бондарь	Ольга								
14	Ревин	Олег								
15	Шарова	Оксана								

8. Выполните редактирование ячеек: – Замените фамилию Иванникова на Иванова.

9. Отсортируйте:

а) *фамилии – по алфавиту* (поставьте маркер на любую фамилию в столбце

Фамилия и щелкните мышкой по кнопке  на панели инструментов или произведите сортировку с помощью контекстного меню)

б) *имя – по алфавиту*

10. Сохраните текущую таблицу, щелкнув по кнопке «крестик» в правом верхнем углу окна таблицы.

11. Откройте снова свою базу данных.

12. Выполните поиск записей по образцу: найти студентку по фамилии Володина. Для этого установите курсор в поле фамилия, щелкните на кнопке Бинокль на панели инструментов меню Главная и в появившемся диалоговом окне введите в поле Образец фамилию Володина и щелкните по кнопке Найти.

*Примечание:* Если требуется найти следующую подобную запись, то щелкните мышкой по кнопке Найти далее. По окончании работы щелкните по кнопке Отмена.

13. Переименуйте поле «Математика» на «Информатика» с помощью контекстного меню. (Верните все как было назад).

14. Скройте столбец Пропуски по неуважительным причинам потом отобразите его назад.

15. Войдите в режим Конструктора и назначьте полю Пропуски по неуважительным причинам и Пропуски по уважительным причинам. Маску ввода 00 «часов». Заполните эти поля данными от 0 до 99

16. Завершите работу с Access.



## ПРАКТИЧЕСКАЯ РАБОТА № 16

**Тема: Создание ключевых полей. Задание индексов. Установление и удаление связей между таблицами**

**Цель работы:** Создание ключевых полей, индексированных полей, установка связей между таблицами. Удаление информации из связанных таблиц и восстановление этой информации

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

**Справочный материал:**

*Создание индексов и ключевых полей.*

Информацию в таблицах можно упорядочить, создав индекс для конкретного поля или нескольких полей. Желательно, чтобы для таблиц были созданы ключевые поля. Для установления связей между таблицами наличие таких полей обязательно. Ключевое поле может быть простым или составным, т.е. состоять из нескольких полей для однозначной идентификации каждой записи в таблице.

*Установка связей между таблицами.*

Выполнить команду *Работа с базами данных /Схема данных*.

1. Появится окно *Схема данных*. Если связи устанавливаются впервые, оно будет содержать диалоговое окно *Добавление таблицы*. Если окно *Добавление таблицы* отсутствует, его можно открыть, пиктограмму *Добавить таблицу*, либо одноименный пункт в контекстном меню (правая клавиша мыши по рабочей области).
2. Выбрать таблицу, которая будет использоваться для установки связей, затем выполнить щелчок на кнопке *Добавить*, для добавления таблицы в окно *Схема данных*.
3. Повторить действия, описанные в п.2 для каждой таблицы, участвующей в установке связи.
4. Для создания связей между таблицами переместить поле (или поля), которое необходимо связать на соответствующее поле другой таблицы. В большинстве связей ключевое поле первой таблицы связывается с аналогичным полем второй таблицы. После перемещения поля появится диалоговое окно *Связи*.
5. В диалоговом окне представлены названия таблиц, между которыми устанавливаются связи и имена полей для связи. Полям, на основе которых создаются связи между таблицами, не обязательно иметь одинаковые имена, однако они должны быть одного типа. Исключение составляют поля счетчиков, которые можно связывать с числовыми полями.
6. Для автоматической поддержки целостности БД установить флажок *Обеспечение целостности данных*. Кроме этого флажка в окне представлены и другие:

– Каскадное обновление связанных полей. При включении данного режима изменения, сделанные в связанном поле первой таблицы, автоматически вносятся в поля связанной таблицы, содержащей те же данные.

– Каскадное удаление связанных полей. При включении данного режима удаление записей в первой таблице приводит к удалению соответствующих записей связанной таблицы.

7. Выполнить щелчок на кнопке *Создать*. Затем закрыть окно *Связи*. При запросе о сохранении связи выполнить щелчок на кнопке *Да*.

### Содержание работы:


**Задание 1.** Создать связь один-ко-многим.

1. Откройте учебную базу данных «Студенты», созданную в Практической работе № 15.

2. Создайте таблицу **Преподаватели** в *Режиме таблицы*. Для этого в меню *Создание* выберите кнопку **Таблица**. В появившейся таблице сделайте следующее:

- Добавьте два поля – Поле 1 и Поле 2, выполнив команду через контекстное меню.
- Переименуйте Поле 1 на **Предмет**. Для этого поставьте курсор в любую ячейку столбца Поля 1 и выполните команду *Переименовать столбец* из контекстного меню. Или щелкните два раза по имени поля, удалите старое название и введите новое.
- Переименуйте аналогично Поле 2 на **Преподаватель**.

3. Сохраните таблицу с именем **Преподаватели**, щелкнув по кнопке *Сохранить* (дискетка  на панели инструментов).

4. Перейдите в режим **Конструктор** и удалите строку с ключевым словом **Счетчик**. Посмотрите как заданы поля. Сделайте поле **Предмет** ключевым, поместив курсор на имя этого поля и щелкнув по кнопке  - *Ключевое поле*. Тип данных поля задайте *текстовым*.

5. Перейдите в *Режим таблицы* и заполните таблицу **Преподаватели** записями из *Таблицы 1*.

Таблица 1

предмет	преподаватель	Д
Математика	Бекетова Н.И.	
Менеджмент	Казумова Н.С.	
Сервисная деятельность	Бессарабова Т.В	
Информационные технологии	Бабич О.А.	
Стандартизация	Казарян Г.Г.	
Гостиничная индустрия	Казарян Г.Г.	
*		

6. Закройте таблицу **Преподаватели**, сохранив все изменения.

7. Используя *Шаблон таблиц*, создайте таблицу **Личные данные** студентов с ключевым полем. Для этого:


- Находясь на закладке *Создание* щелкните по кнопке *Шаблоны таблиц, Контакты*. Появится таблица уже с готовыми полями.
- Переименуйте предложенные поля на следующие поля: *Код студента, Фамилия, Имя, Город, Адрес, Телефон, Дата рождения, Фотография, Любимый предмет*, лишние поля удалите.

- Сохраните полученную таблицу под названием **Личные данные**. Ключевое поле задано автоматически.

8. Внесите данные в новую таблицу, заполнив поля *Фамилия*, *Имя*, *Город*, *Адрес*, *Телефон*, *Дата рождения*.

**ПРИМЕЧАНИЕ.** Поля *Фамилия* и *Имя* можно скопировать из таблицы *Ведомость успеваемости*. В поле *Город* внесите четыре разных города (например, Саратов, Петровск, Пенза, Энгельс)


9. Перейдите в режим *Конструктор* и назначьте типы данных: для поля *Телефон* – *числовой*, для поля *Дата рождения* – *дата/время*, для поля *Фотография* – *поле объекта OLE*, для остальных – *текстовый*.

Для поля *Любимый предмет* выполните свойство выбор предмета из списка с помощью *Мастера подстановок*. Для этого в строке *Любимый предмет* в поле *Тип данных* – *текстовый* щелкните по кнопке  и в ниспадающем меню выберите команду **Мастер подстановок**.


- В диалоговом окне *Создание подстановки* поставьте флажок напротив способа *Будет введен фиксированный набор значений* и нажмите *Далее*.

- В следующем окне внесите в столбец все предметы (предметы из таблицы *Преподаватели*), нажмите *Далее*.

- В последнем окне, не изменяя имени столбца нажмите *Готово*.

10. Перейдите в режим таблицы и выберите для каждого студента с помощью кнопки  из списка любимый предмет.

11. Создайте **схему данных**, т.е. установите связи между таблицами.

Щелкните по кнопке  - *Схема данных* на панели инструментов меню **Работа с базами данных**. В окне *Отобразить таблицу* выделите таблицу *Ведомость успеваемости* и щелкните по кнопке *Добавить*. Также добавьте таблицы *Преподаватели* и *Личные данные*. В окне *Схема данных* появится условный вид этих таблиц. Закройте окно *Добавление таблицы*.

- Поставьте мышку на имя поля *Предметы* в таблице *Преподаватели*, и не отпуская кнопку мыши перетащите его на поле *Любимый предмет* таблицы *Личные данные*. Отпустите мышку. Появится диалоговое окно *Связи*, в котором включите значки «Обеспечение целостности данных», «Каскадное обновление связанных полей» и «Каскадное удаление связанных полей». Щелкните по кнопке *Создать*. Появится связь «**один-ко-многим**».

- Поставьте мышку на имя поля *Код студента* в таблице *Личные данные* и перетащите его, не отпуская мышки, на поле *Код* таблицы *Ведомость успеваемости*. В появившемся окне *Связи* включите значок «Обеспечение целостности данных» и щелкните по кнопке *Создать*. Появится связь «**один-к-одному**».

- Закройте схему данных, сохранив ее.

**Задание 2.** Создать таблицы, представляющие собой фрагмент базы данных учебного центра.

Таблица «Курс» с полями: «Код курса» - поле типа «Счетчик», создать автоматически как ключ при завершении описания таблицы; «Наименование

курса» - текстовое поле длиной 120 символов; «Продолжительность» - числовое поле, размер поля – целое; «Стоимость обучения» - поле денежного типа.

Таблица «Преподаватель» с полями: «Код преподавателя» - поле типа «Счетчик», ключевое поле; «ФИО преподавателя» - текстовое поле, 50 символов; «Дата рождения» - поле типа «Дата/время», «Должность» - текстовое поле, 25 символов, «Научно-педагогический стаж» - числовое; «Общий стаж работы» - числовое; «Контактный телефон» - текстовое поле, 15 символов.

Таблица «Владение предметами» с полями: «Код преподавателя», «Код курса» - числовое поле, размер поля – длинное целое. Создать составной ключ, включающий оба эти поля.

Таблица «График учебного процесса» с полями: «Код потока» - поле типа «Счетчик», создать автоматически как ключ при завершении описания таблицы; «Код курса» - числовое поле, размер поля – длинное целое; «Дата начала» - поле типа «Дата/время»; «Дата завершения» - поле типа «Дата/время»; «Время начала» - поле типа «Дата/время»; «Время завершения» - поле типа «Дата/время».

Таблица «Слушатель» с полями: «Код слушателя» - числовое поле, размер поля – длинное целое, поле ключа; «ФИО слушателя» - текстовое поле, 50 символов; «Контактный телефон» - текстовое поле, 10 символов.

Таблица «Запись на курс» с полями: «Код потока» - числовое поле, размер поля – длинное целое; «Код слушателя» - числовое поле, размер поля – длинное целое. Создать составной ключ, включающий оба эти поля.

Создать связь один-ко-многим для всех таблиц БД.

## ПРАКТИЧЕСКАЯ РАБОТА № 17

**Тема: Редактирование, добавление и удаление записей в таблице.**

**Применение логических условий к записям. Открытие, редактирование и пополнение табличного файла**

**Цель работы:** закрепить навыки по редактированию таблиц; познакомиться с основными видами запросов; научиться создавать запросы на выборку различными способами; научиться создавать сложные запросы; научиться создавать перекрестные запросы

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

### **Справочный материал:**

Запрос – это средство, с помощью которого извлекается из базы данных информация, отвечающая определенным критериям. Результаты запроса представляют не все записи из таблицы, а только те, которые удовлетворяют запросу.

Запросы состоят из ряда условий, каждое условие состоит из трех элементов:

1. поле, которое используется для сравнения;
2. оператор, описывающий тип сравнения;
3. величина, с которой должно сравниваться значение поля

Запросы могут быть простые, сложные перекрестные.

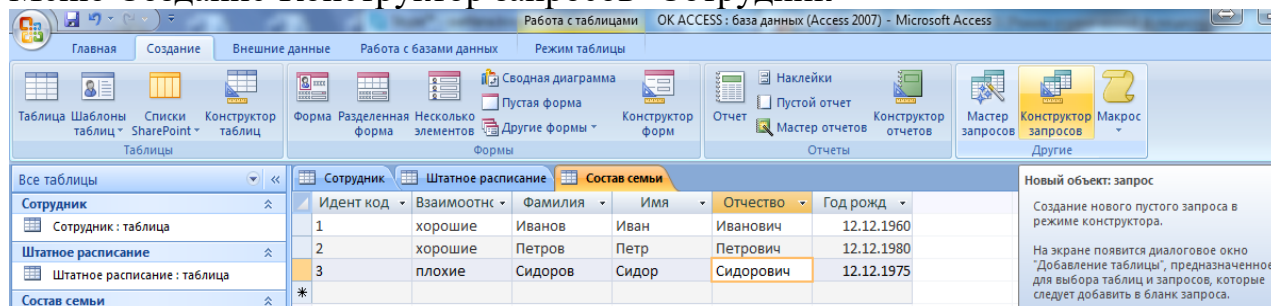
### **Содержание работы:**

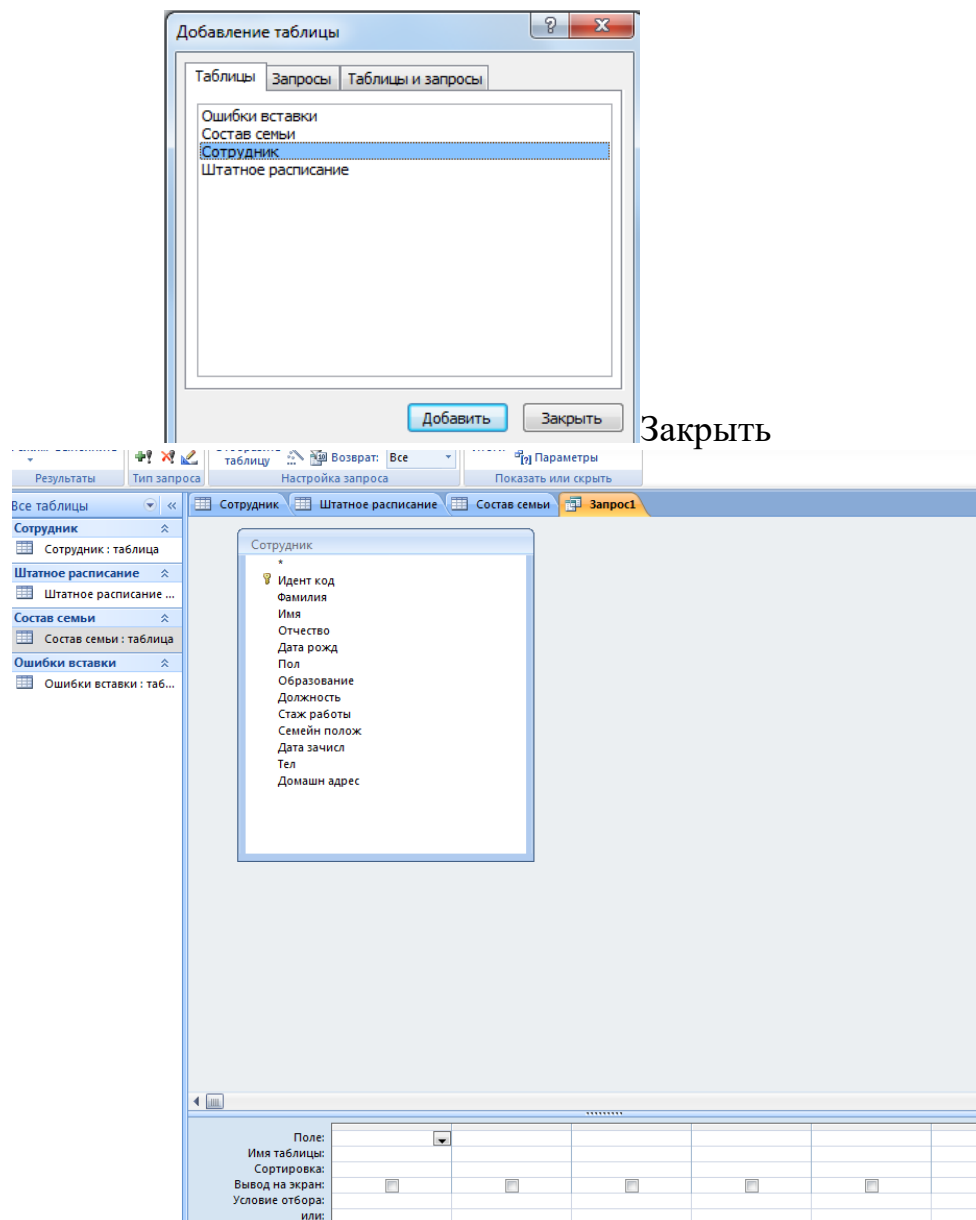
**Задание 1.** Создать запрос на выборку данных из одной таблицы

1.Создадим запрос, что содержит поля: Идент код, Фамилия, Имя, Отчество, Дата нар, который отображает список только тех сотрудников, фамилии которых начинаются с буквы "К". Список отсортируем по дате рождения по возрастанию.

Для этого необходимо выполнить такую последовательность действий.

Меню Создание-Конструктор запросов -Сотрудник





2.Выбираем объект **Запросы**, щелкаем пункт меню **Создать**. Открывается окно **Новый запрос**, в котором выбираем режим создания запроса **Конструктор**. Открывается окно **Запросы: запрос на выборку** и активизируется окно **Добавление таблицы**, в котором следует выбрать из списка таблицу **Сотрудник** (щелкнув мышью на имя таблицы), после чего нажать на кнопку **Добавить** и закрыть окно **Добавление таблицы**.

3.Дальше необходимо выбрать нужные поля и задать способы сортировки и условия отбора из таблицы. Для этого:

- выделить поля **Идент код**, **Фамилия**, **Имя**, **Отчество**, **Дата рожд** с помощью мыши в комбинации с клавишами **SHIFT** или **CTRL** и отбуксировать на бланк построения запроса **QBE** в строку **Поле**. Поля можно перемещать в бланк **QBE** и в одиночку.

Сотрудник Штатное расписание Состав семьи **Запрос1**

Сотрудник

- Идент код
- Фамилия
- Имя
- Отчество
- Дата рожд
- Пол
- Образование
- Должность
- Стаж работы
- Семейн полож
- Дата зачисл
- Тел
- Домашн адрес

Поле:	Идент код	Фамилия	Имя	Отчество	Дата рожд
Имя таблицы:	Сотрудник	Сотрудник	Сотрудник	Сотрудник	Сотрудник
Сортировка:					
Вывод на экран:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Условие отбора:					
или:					

- для поля **Дата рожд** установить сортировку записей по возрастанию. Для этого щелкнуть в строке **Сортировка** в столбце поля **Дата рожд**, появится кнопка со стрелкой, нажатие на которую раскрывает окно выбора типа сортировки. Выбрать тип сортировки **по возрастанию**;

во	Дата рожд
ник	Сотрудник
<input checked="" type="checkbox"/>	по возрастанию по убыванию (отсутствует)

- для реализации в запросе условия выбора сотрудников, фамилии которых начинаются с буквы "К", в строке **Условие отбора** в столбце **Фамилия** ввести условие **Like "К\*"**.

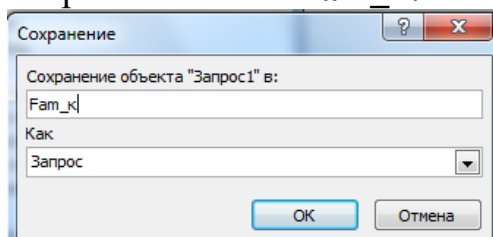
Все таблицы << Сотрудник Штатное расписание Состав семьи **Fam\_k**

Сотрудник

- Идент код
- Фамилия
- Имя
- Отчество
- Дата рожд
- Пол
- Образование
- Должность
- Стаж работы
- Семейн полож
- Дата зачисл
- Тел
- Домашн адрес

Поле:	Идент код	Фамилия	Имя	Отчество	Дата рожд
Имя таблицы:	Сотрудник	Сотрудник	Сотрудник	Сотрудник	Сотрудник
Сортировка:					по возрастанию
Вывод на экран:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Условие отбора:		Like "К*"			
или:					

Закрывать окно конструктора запроса, сохранить в памяти с именем. В окне базы данных появится файл запроса с именем **Fam\_k**.

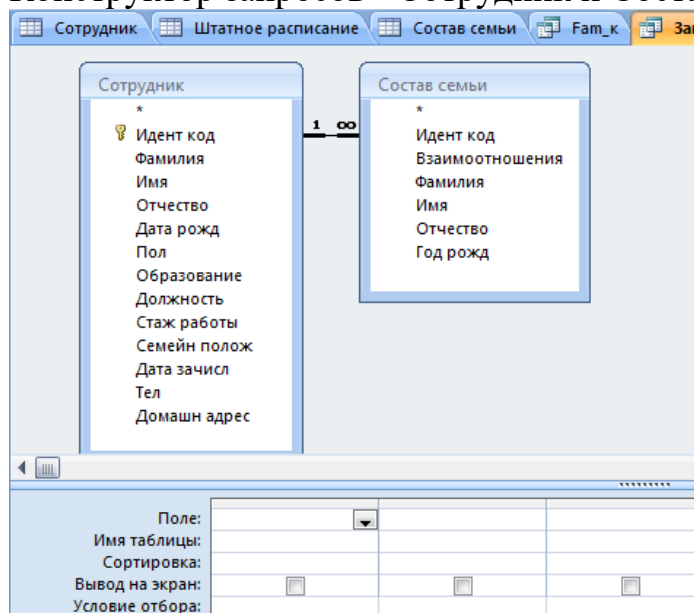


### Файл - Сохранить как

Выполнить запрос на выборку. Для этого выделить запрос **Fam\_k** и щелкнуть по кнопке **Открыть**. На экран выводится таблица, в которой отображаются все записи с фамилиями, которые начинаются на букву К, записи отсортированные по дате рождения по возрастанию.

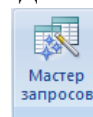
Идент код	Фамилия	Имя	Отчество	Дата рожд
1	К	К	К	01.01.2000
*				

**Задание 2.** Создать запрос на выборку данных с из двух или трех таблиц.  
1. Меню Создание-Конструктор запросов - Сотрудник и Состав семьи, Закрывать



2. Создадим запрос, в результате выполнения которого будут полученные сведения о сотрудниках, которые не имеют родственников.

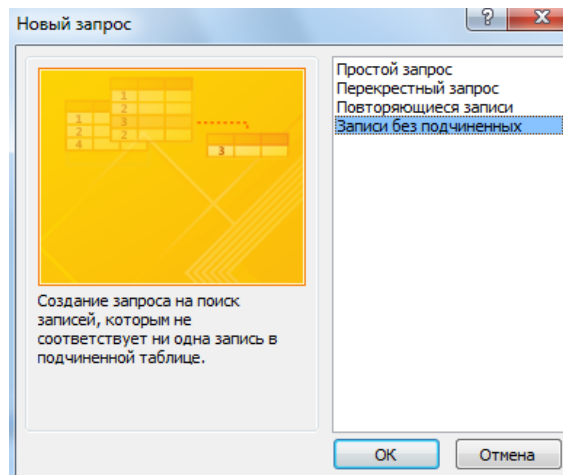
3. Для отчета включим поля, которые содержат идентификационный код, фамилию, имя, отчество рабочего, а также его дату рождения.



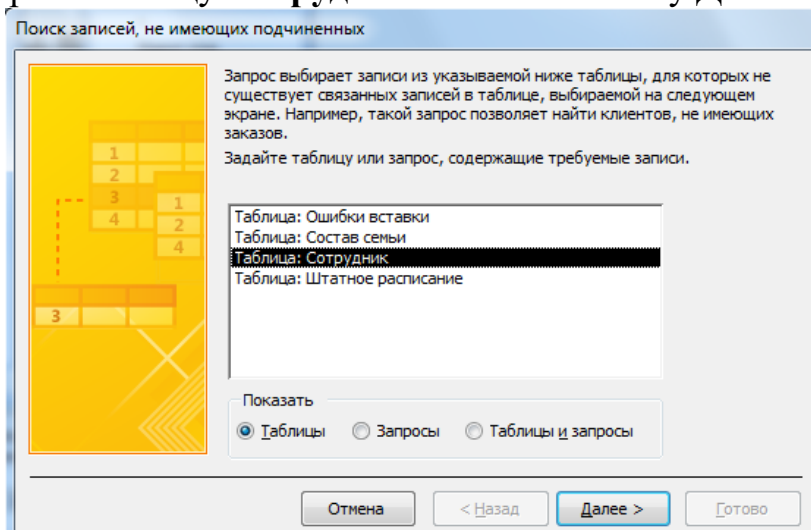
4. При выбранной вкладке **Запрос** щелкнуть по кнопке

5. Открывается окно **Новый запрос**, в котором выбрать режим создания запроса **Записи без подчиненных**.

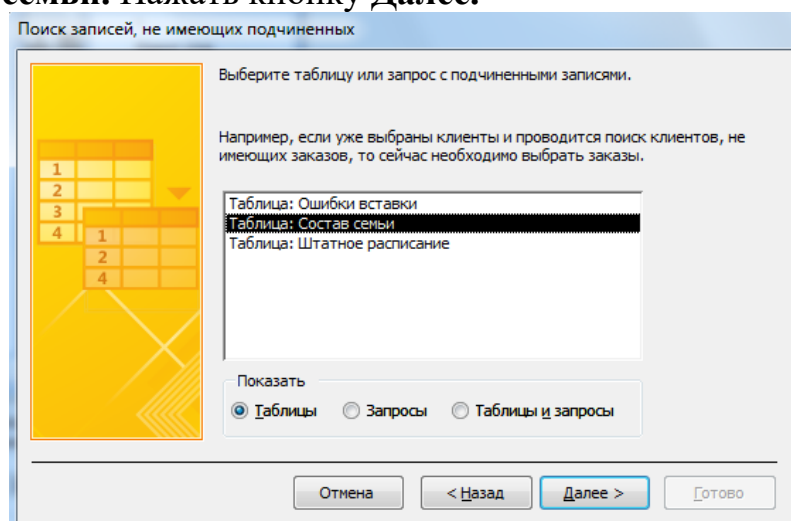




6. В первом окне с названием "Поиск записей, не имеющих подчиненных" мастер выведет на экран список для выбора основной таблицы, в котором выбрать таблицу **Сотрудник** и нажать кнопку **Далее**.



7. В следующем окне выбрать таблицу, что содержит подчиненные записи. Это таблица **Состав семьи**. Нажать кнопку **Далее**.



8. В следующем окне мастера проверить, что таблицы **Сотрудник** и **Состав семьи** связаны по полю **Идент код** (поля, по которым связанные таблицы, выделены). Если это не так, в каждом списке полей обеих таблиц выделить

поле **Идент код** и щелкнуть на кнопку «<=>», что расположена между списками. Нажать кнопку **Далее**.

Поиск записей, не имеющих подчиненных

Какие данные содержатся в обеих таблицах?  
Например, и таблица "Клиенты", и таблица "Заказы" содержат поле "Клиент". Соответствующие поля могут иметь и различные имена.  
Выберите подходящее поле в каждой таблице и нажмите кнопку <=>.

Поля в 'Сотрудник':

- Идент код
- Фамилия
- Имя
- Отчество
- Дата рожд
- Пол
- Образование
- Должность

Поля в 'Состав семьи':

- Идент код
- Взаимоотношения
- Фамилия
- Имя
- Отчество
- Год рожд

Соответствие: Идент код <=> Идент код

Отмена < Назад Далее > Готово

9. На экране появится новое окно, в котором отображенные поля, которые могут быть включены к отчету. В этом окне в левом поле в списке выделить по очереди поля **Идент код**, **Фамилия**, **Имя**, **Отчество**, **Дата рожд**, которые должны отображаться в отчете, и перенести их в левое поле с помощью кнопки «>».

Поиск записей, не имеющих подчиненных

Выберите поля для отображения в результате выполнения запроса:

Доступные поля:

- Пол
- Образование
- Должность
- Стаж работы
- Семейн полож
- Дата зачисл
- Тел
- Домашн адрес

Выбранные поля:

- Идент код
- Фамилия
- Имя
- Отчество
- Дата рожд

Отмена < Назад Далее > Готово

Далее

Поиск записей, не имеющих подчиненных

Задайте имя запроса:

**Сотрудник без подчиненных в Состав семьи**

Указаны все сведения, необходимые для создания запроса с помощью мастера.

Дальнейшие действия после создания запроса:

- ☒ Просмотреть результаты запроса.
- ☐ Изменить структуру запроса.

Отмена < Назад Далее > Готово

Дальше нажать кнопку **Готово**.

10.Проверяем исправленный отчет и храним его под именем **Без родственников**.

Идент код	Фамилия	Имя	Отчество	Дата рожд
1	К	К	К	01.01.2000
*				

### Задание 3. Создать параметрический запрос.

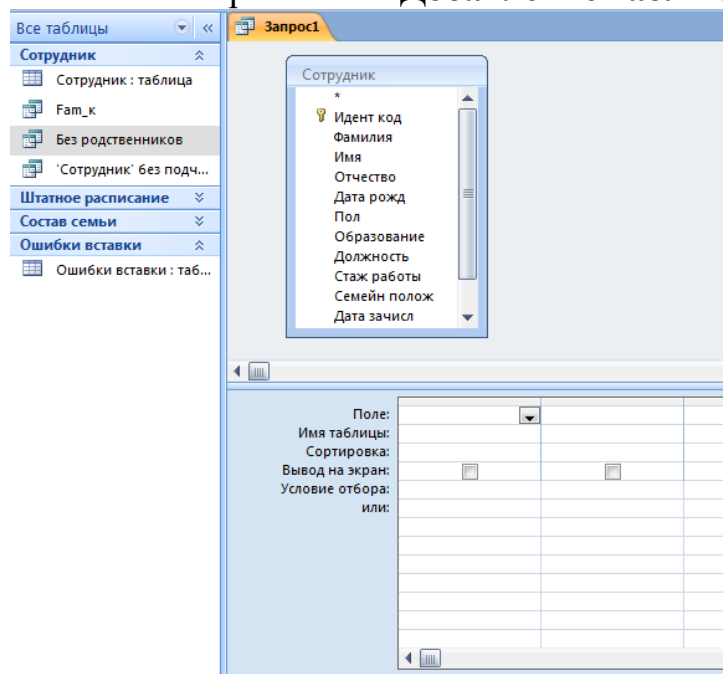
При выполнении параметрического запроса выводится диалоговое окно с приглашением ввести параметр для условия отбора записей. Параметров может быть несколько.

1.Создадим запрос, в результате выполнения которого будут выводиться поля **Фамилия, Имя, Отчество, Идент код** и **Стаж работы** сотрудника, фамилия которого будет указана в запросе как параметр отбора.

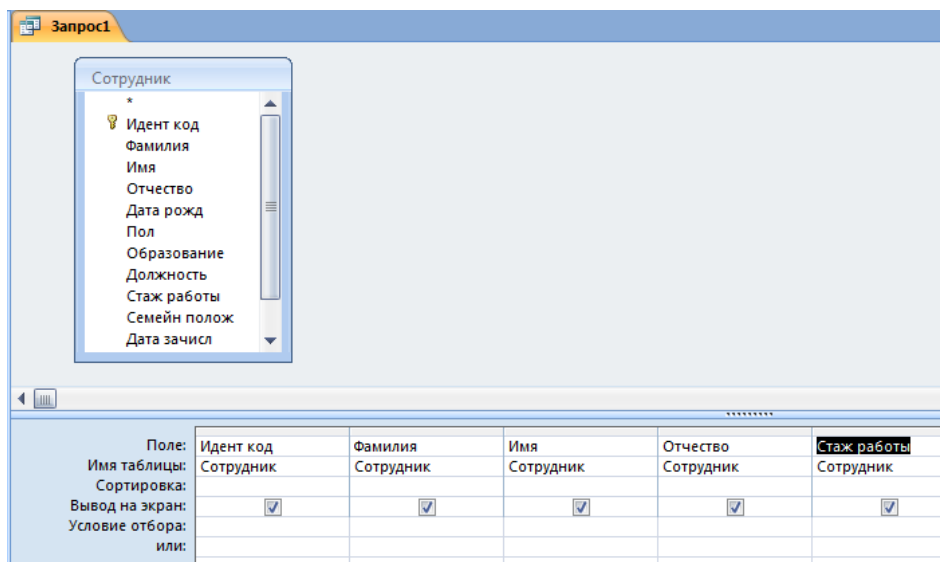
2.При выбранном режиме работы **Запрос** щелкнуть по кнопке **Создать. Меню Создание-Конструктор запросов -Сотрудник, Заккрыть**

3.Открывается окно **Новый запрос**, в котором выбрать режим создания запроса **Конструктор**.

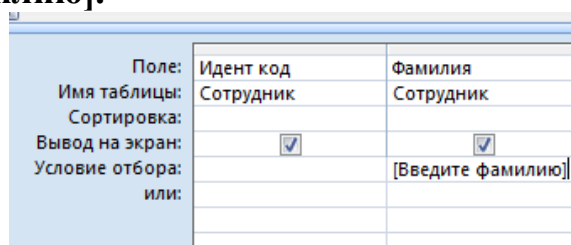
4.Открывается окно **Запрос 1: запрос на выборку** и активизируется окно **Добавление таблицы**, в котором выбрать таблицу **Сотрудник**, щелкнуть по кнопке **Добавить**, после чего закрыть окно **Добавление таблицы**.



5.С помощью мыши переместить поля **Фамилия, Имя, Отчество** и **Идент код, Стаж работы** из выбранной таблицы в бланк построения запроса.

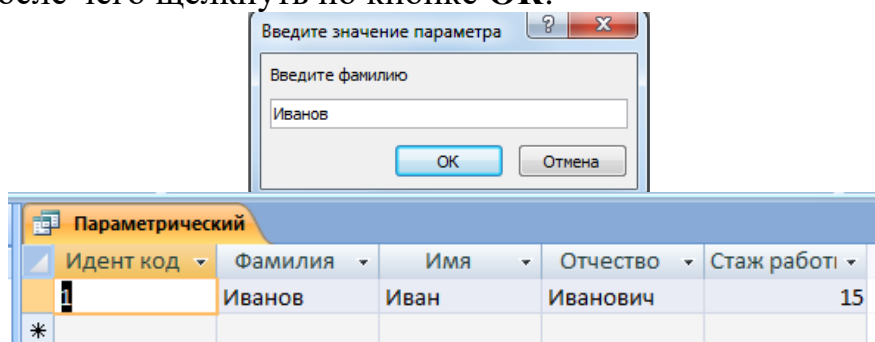


6. В столбце **Фамилию** в строке **Условие отбора** ввести в квадратных дужках сообщения, которое будет выводиться на экран при выполнении запроса, а именно: **[Введите фамилию]**.



7. Сохранить как с именем **Параметрический**

8. Выполните запрос, щелкнув по пункту меню **Открыть**. На экране появится окно **Введите значение параметра** для ввода фамилии сотрудника, информацию о котором необходимо получить, например об Иванове или Сидорове, после чего щелкнуть по кнопке **ОК**.



На экране появится таблица с данными о выбранном сотруднике.

## ПРАКТИЧЕСКАЯ РАБОТА № 18

**Тема: Проведение сортировки и фильтрации данных. Поиск данных по одному и нескольким полям. Поиск данных в таблице.**

**Цель работы:** научиться создавать таблицы с помощью конструктора (создание макета таблицы с указанными полями, заполнение таблицы, использование мастера подстановок, установка поля первичного ключа), сортировать данные таблицы, связывать таблицы, осуществлять поиск данных в таблице по заданным критериям.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

### **Справочный материал:**

#### Сортировка записей

Существует два вида сортировки, которые можно выполнить: простая сортировка и сложная сортировка.

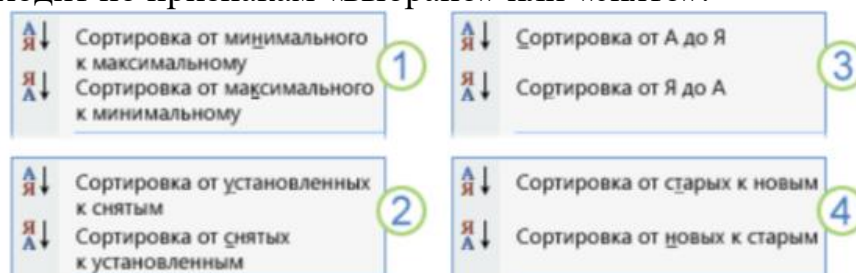
*Простая сортировка.* При сортировке в режиме формы, в режиме таблицы выполняется простая сортировка, то есть все записи поля сортируются по возрастанию или по убыванию (но не в том и другом порядке сортировки одновременно).

*Сложная сортировка.* Если нужно провести сортировку записей в режиме конструктора запроса, в окне расширенного фильтра, в режиме конструктора отчета, в режиме конструктора страницы, в режиме сводной диаграммы или сводной таблицы, можно выполнить сложную сортировку. Это означает, что по некоторым полям допускается сортировка по возрастанию, а по другим полям сортировка по убыванию.

#### Сортировка таблицы, запроса или формы

1. Укажите поля для сортировки. Чтобы выполнить сортировку по двум или более полям, укажите, какие из полей будут использоваться в качестве внутренних и внешних полей сортировки.
2. Щелкните правой кнопкой мыши столбец или элемент управления, соответствующий внутреннему полю, и выберите одну из команд сортировки. Команды зависят от типа данных, содержащихся в выбранном поле.

*Примечание.* При сортировке данных по полю с логическим типом данных значения «Да», «Истина» или «Включено» считаются «выбранными», а значения «Нет», «Ложь» или «Отключено» — «снятыми». По умолчанию этот тип поля отображается в виде флажка, но пользователь может настроить отображение поля в виде текстового поля или поля со списком. При смене вида отображения поля на текстовое поле или поля со списком сортировка происходит по признакам «выбрано» или «снято».



1. Числовой, Денежный, Счетчик
2. Текстовый, Поле МЕМО, Гиперссылка
3. Логический
4. Дата/время

3. Повторите предыдущий шаг для каждого поля сортировки, включая последнее внешнее поле сортировки. Записи переупорядочиваются в соответствии с порядком сортировки.

Если текстовое поле содержит значения и пустые строки, при сортировке по возрастанию сначала отображаются записи со значением Null, потом записи с пустыми строками, а затем записи с непустыми значениями.

Если значение в поле начинается со специального знака, такого как дефис, скобки или другого символа, при сортировке по возрастанию соблюдаются следующие правила:

- Значения, начинающиеся с пробела, отображаются перед алфавитно-цифровыми значениями.
- Значения в скобках отображаются после значений, начинающихся с пробелов, но перед алфавитно-цифровыми значениями.
- Значения, начинающиеся со знака «минус» (-), отображаются перед значениями со знаком «плюс» (+).
- Для всех других знаков порядок сортировки определяется на основе кодов ASCII этих знаков. Например, для знака доллара (\$) используется код 36, а для знака равенства (=) — 61, поэтому значения, начинающиеся с \$, отображаются перед значениями, начинающимися с =.

Для переопределения этого порядка можно проигнорировать первый знак для всех значений в этом поле. Этот метод удобно использовать, если значения в поле всегда начинаются с одного специального знака, например, знака «минус (-)», или с одинакового количества специальных знаков — то есть заранее известно, сколько знаков игнорировать. Если количество знаков, которое необходимо игнорировать, изменяется, можно определить специальный (пользовательский) порядок сортировки.

Имейте в виду, что нельзя удалить порядок сортировки только в одном поле. Чтобы отменить сортировку во всех полях сортировки, на вкладке Главная в группе Сортировка и фильтр нажмите кнопку Очистить все сортировки, а затем примените необходимый порядок сортировки.

#### Использование Мастера подстановок при вводе данных в таблицы

Мастер подстановок позволяет формировать для нужного поля список значений, который может содержать данные другой таблицы или запроса, либо состоять из фиксированного набора значений. В обоих случаях Мастер подстановок облегчает ввод данных и обеспечивает их достоверность.

#### **Содержание работы:**

**Задание 1.** Создать базу данных, состоящую из двух таблиц, в которых бы хранились информация о студентах и их родителях какой – либо группы.

1. Откройте приложение MS Access.
2. Выберите пункт Новая база данных.

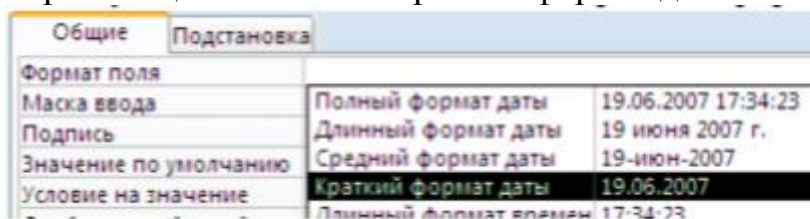
3. На панели задач справа введите в окне Имя файла имя файла базы данных – ГРУППА. Выполните щелчок на кнопке Создать.

4. Создайте в этой базе данных в режиме Конструктор таблицу «Сведения о студентах» командой вкладка Создание – панель Таблицы – Конструктор таблиц.

5. В окне конструктора таблиц задаются поля таблицы и их свойства. Создайте структуру таблицы с полями, представленные ниже.

Имя поля	Тип данных	Описание
Фамилия	Текстовый	
Имя	Текстовый	
Код студента	Текстовый	На первом месте этого поля указывается первая буква фамилии, затем цифры без пробела.
Пол	Текстовый	
Дата рождения	Время/Дата	
Район	Текстовый	
Адрес	Текстовый	Улица, дом, квартира
Домашний телефон	Текстовый	

6. Для поля Дата рождения установите свойства поля – Краткий формат даты, для этого установите курсор в это поле и в свойства поля на вкладке общие выберите из раскрывающегося списка Краткий формат даты.



7. Объявите поле Код студента созданной таблицы ключевым полем командой вкладка Работа с таблицами/Конструктор – панель Сервис – Ключевое поле.

8. Сохраните структуру таблицы командой кнопка Office – Сохранить. Дайте ей имя «Сведения о студентах».

9. Создайте для поля Район таблицы «Сведения о студентах» список значений, используя мастер подстановок. Для этого:

а. В режиме конструктора для поля Район выберите тип данных Мастер подстановок.

б. Укажите пункт будет введён фиксированный набор значений и нажмите Далее.

с. Введите в столбец все районы нашего города и нажмите Далее и Готово.

10. Сохраните ещё раз структуру таблицы.

11. Перейдите в режим таблицы и заполните таблицу «Сведения о студентах» (8 записей).

При заполнении поля Район воспользуйтесь созданным списком. (См. образец):



Фамилия	Имя	Код студент	Пол	Дата рождения	Район	Адрес	Домашний
Анисимова	Татьяна	A21	Ж	28.10.1985	Центральный	Дружбы 46-28	45-67-23
Белкин	Константин	B25	М	15.08.1986	Орджоникидз	Авиаторов 89	61-90-09
Гусев	Владимир	G62	М	05.11.1986	Новокузнецкий	до лет ВЛКСМ	53-78-90
Иванов	Петр	I34	М	16.07.1894	Центральный	Орджоникидз	56-87-90
Максимова	Алена	M34	Ж	07.03.1986	Нуйбышевский	Гореза 56-78	57-90-87
Москаев	Алексей	M45	М	14.12.1985	Заводской	Челюскина 34	35-45-78
Светлова	Екатерина	S45	Ж	13.07.1896	Кузнецкий	Горьковская 1	12-13-14
Тереньцова	Елизавета	T56	Ж	03.05.1986	Центральный	Косыгина 45-5	61-76-45
Тулунов	Руслан	T43	М	23.11.1985	Центральный	Металлургов	45-67-23

12. Произвести сортировку записей в таблице по полю Фамилия по возрастанию. Для этого поставьте курсор в поле Фамилия и выполните команду вкладка Главная – панель Сортировка и фильтр – Сортировка по возрастанию.

13. Создайте в этой же базе данных ещё одну таблицу «Сведения о родителях» с помощью конструктора с полями, представленные в следующей таблице:

Имя поля	Тип данных
Код студента	Текстовый
Сведения о матери	Текстовый
Сведения об отце	Текстовый

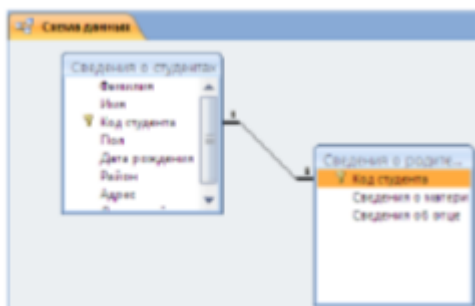
14. Создайте для поля Код студента поле со списком Код студента, Фамилии, Имя, используя мастер подстановок. См. выполнение п. 9, только указать пункт Объект будет использовать значение из таблицы, добавить указанные поля, используя одинарную стрелку, ключевое поле не скрывать.

15. В качестве ключевого поля выберите Код студента.

16. Заполните таблицу данными. (см. образец)

Код студента	Сведения о матери	Сведения об отце
A21	Анисимова Татьяна Николаевна, ООО "Ацтек", 54-44-55	Анисимов Александр Петрович, КМН, 45-67-54
A21	Анисимова Татьяна	Белкин Петр Николаевич, ОАО "Терем", 45-7-33
B25	Белкин Константин	Гусев Николай Сергеевич, ООО "Рурьер", 55-43-38
G62	Гусев Владимир	

17. Установить связь «один к одному» между таблицами «Сведения о студентах» и «Сведения о родителях» по полю Код студента командой вкладка Работа с таблицами/ Режим таблицы – панель Связи – команда Схема данных.



**Задание 2.** Доработать базу данных из задания 1.

1. Откройте базу данных, созданную в задании 1.

2. Создайте новые таблицы Студенты и Экзамены со следующими структурами:



Таблица Студенты

Имя поля	Тип поля
Номер зачетки	Числовой
Фамилия	Текстовый
Имя	Текстовый
Отчество	Текстовый
Факультет	Текстовый
Курс	Числовой
Группа	Числовой
Дата рождения	Дата\Время
Стипендия	Числовой

В бланке Свойства обязательно указать длину текстовых полей, формат числовых полей и дат. Поле Номер зачетки в таблице Студенты объявить ключевым и индексированным со значением Совпадения не допускаются.

Заполнить таблицу содержимым (10-15 записей).

Таблица Экзамены

Имя поля	Тип поля
Номер зачетки	Мастер подстановок..
Предмет	Текстовый
Оценка	Числовой
Дата сдачи	Дата\Время

Обязательно определить нужные формат и длину полей в бланке Свойства.

Тип поля Номер зачетки определяется Мастером подстановок, используя для подстановки данные из таблицы Студенты. В качестве доступных при подстановке полей выбрать Фамилию и Имя.

Поле Номер зачетки в таблице Экзамены объявить индексированным со значением Совпадения допускаются.

При сохранении структуры неключевой таблицы Access может предупредить об отсутствии ключевого поля и предложит создать это поле сейчас. В данном случае следует отказаться от этого.

После определения структур обеих таблиц вызвать окно Схема данных и добавить в схему данных обе таблицы (Студенты и Экзамены). Установить в окне схемы данных связь между таблицами по полю Номер зачетки. В окне Связи включить переключатель Определение целостности данных. После этого нужно указать тип связи: Один-ко-многим и включить опции Каскадное обновление связанных полей и Каскадное удаление связанных полей. После этого закрыть окно Связи.

В режиме таблицы ввести данные в таблицу Экзамены, используя созданный с помощью Мастера подстановок список в поле Номер зачетки. Для проверки соблюдения целостности данных при работе с таблицами нужно:

\*\*\*изменить значение ключевого поля (Номер зачетки) для одной из записей в таблице Студенты. Перейти в таблицу Экзамены и проверить, изменилось ли в ней значение общего поля для соответствующих записей;

\*\*\*удалить одну из записей в таблице Студенты. Перейти в таблицу Экзамены и проверить, удалены ли в ней соответствующие записи.

После редактирования таблицы Студенты нужно сначала сохранить в ней изменения, а затем переходить в неключевую таблицу.

**Задание 3.** В режиме Конструктора таблицы Студенты произвести в ее структуре следующие изменения:

\* добавить поля Город, Адрес, Телефон;

\* определить тип поля Факультет с помощью Мастера подстановок, взяв в качестве источника данных фиксированный набор значений (список всех факультетов ВУЗа).

\* для поля Город в свойстве Значение по умолчанию задать значение: Уфа.

\* для поля Курс ввести условие на значение:  $>0, \leq 5$  и задать соответствующее сообщение об ошибке.

Данные в поля Город, Адрес, Телефон вводить в режиме таблицы.

Отредактировать значения в поле Факультет, используя список значений, созданный Мастером подстановок.

Для нескольких записей использовать значение по умолчанию в поле Город.

В одной из записей попробовать внести в поле Курс значение, большее 5.

Вызвать окно для изменения схемы данных. Скрыть одну из таблиц, включенных в схему данных (например, Экзамены). Затем отобразить все прямые связи.

Изменить макет таблицы Студенты:

- зафиксировать столбцы Фамилия и Номер зачетки.
- поле Город поставить после поля Отчество;
- скрыть столбцы Адрес, Телефон и Стипендия;
- оставить для столбцов только вертикальную сетку;
- установить произвольно цвет фона для записей;
- изменить шрифт для записей таблицы на курсив.

Отсортировать таблицу Студенты по следующим признакам:

- возрастанию в поле Фамилия;
- убыванию в поле Стипендия;
- возрастанию в поле Факультет и убыванию в полях Курс и Группа.

Найти в таблице Студенты все записи, удовлетворяющие следующим условиям:

- студенты, чьи фамилии начинаются с определенной буквы;
- студенты, обучающиеся на одном курсе определенного факультета.

Найти записи для студентов определенного факультета и заменить для них название этого факультета. Например, эконом. на экономический.

**Задание 4.** Создать простой запрос - выбрать несколько произвольных полей из таблицы Студенты.

С помощью Конструктора создать запросы, удовлетворяющие условиям:

- единственное значение факультета;
- два различных факультета;
- фамилии студентов, начинающиеся с определенной буквы (использовать шаблоны);
- фамилии студентов, заканчивающиеся на “ов”;
- фамилии студентов одного факультета и одного курса;
- фамилии и имена студентов, проживающие в одном из городов или обучающиеся на одном из факультетов;
- фамилии студентов, у которых стипендия больше 400 рублей;
- фамилии студентов, занимающиеся не в 1-ой группе и стипендия которых в пределах от 200 до 500 р.

*Примечание:* В запрос должны быть включены поля Фамилия, Имя, Отчество и те поля, где вводятся критерии.

После того как был задан критерий для запроса, запрос нужно выполнить и сохранить под именем, подходящим по смыслу.

Для запросов с полем типа Дата/время добавить поле Дата рождения и выбрать записи, удовлетворяющие условиям:

- дата больше 1.1.80;
- дата в интервале значений и задан факультет;
- фамилии и имена студентов, родившихся в 80-х годах;
- вычислить возраст студентов;
- фамилии и имена студентов, родившихся в первой половине месяца;

Создать итоговый запрос:

- оставить в запросе поля Факультет, Стипендия, Номер зачетки, вычислить максимальное значение стипендии для каждого факультета и подсчитать количество студентов на каждом факультете (используя Count).

Запрос с вычисляемыми полями:

- включить в запрос вычисляемое поле, которое является результатом сцепления текстовых полей Фамилия, Имя, Отчество. Назвать поле Ф. И. О. студента.
- используя построитель выражений, подсчитать надбавку студентам, равную 15% от стипендии;

Создать запрос, в котором используются поля из двух ранее созданных и связанных таблиц, задав ему имя Запрос для 2-х таблиц –убрать несколько полей таблицы Студенты и добавить поля Предмет и Оценка из таблицы Успеваемость;

- выбрать поле Фамилия, предмет и Оценка, вычислить минимальное значение по полю Оценка;
- сгруппировать по номеру зачетки и вычислить среднюю оценку для каждого студента.

## **ПРАКТИЧЕСКАЯ РАБОТА № 19**

### **Тема: Создание запросов**

**Цель работы:** научиться создавать разные виды запросов – сложные запросы на выборку, обновление, удаление и добавление данных, перекрестных запросов.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### **Справочный материал:**

В перекрестном запросе отображаются результаты статистических расчетов (например, суммы, количество записей, средние значения), выполненных по данным из одного поля таблицы. Например, необходимо вычислить средний стаж работы ассистентов, доцентов и профессоров, работающих на разных кафедрах.

Перекрестный запрос создает таблицу, в которой заголовками строк будут служить должности, заголовками столбцов - названия кафедр, а в ячейках будут рассчитаны средние значения стажа преподавателей.

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

- запрос на удаление удаляет группу записей, удовлетворяющих заданным условиям, из одной или нескольких таблиц; с помощью запроса на удаление можно удалять только всю запись, а не отдельные ее поля;
- запрос на обновление записей вносит общие изменения в группу записей одной или нескольких таблиц;
- запрос на добавление добавляет группу записей из одной или нескольких таблиц в конец одной или нескольких таблиц;
- запрос на создание таблицы создает новую таблицу на основе всех или части данных из одной или нескольких таблиц.

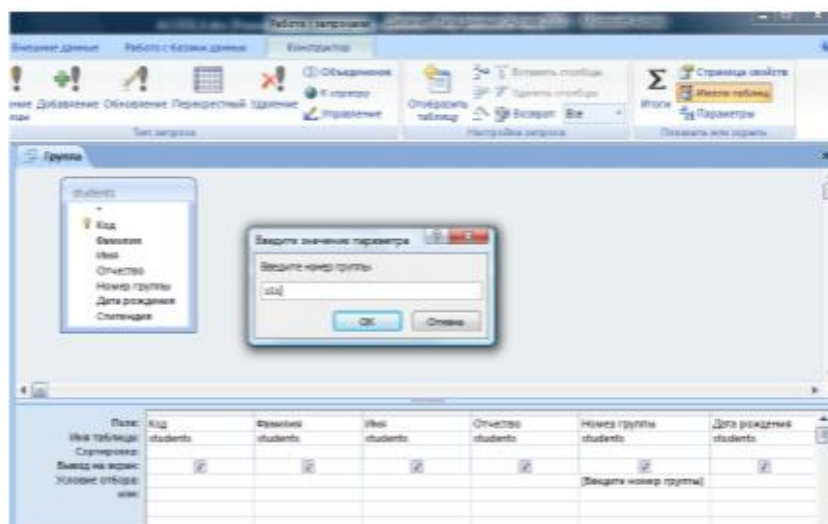
#### **Содержание работы:**

**Задание.** Создать запросы.

1. Открыть базу данных, созданную в Практической работе № 18.
2. Разработайте запрос с параметрами о студентах заданной группы, в котором при вводе в окно параметров номера группы (например, 151 или 152) на экран должен выводиться состав этой группы.

Для создания запроса с параметрами о студентах заданной группы:

- создайте простой запрос на основании таблицы Студенты, выбрав все поля таблицы; выведите все поля таблицы (подробный отчет); задайте имя запроса Группа;
- перейдите в режим конструктора; в строке Условия отбора для поля Номер группы введите фразу: [Введите номер группы]; откройте запрос двойным щелчком по названию в левой панели:



- Сохраните запрос и закройте таблицу запроса. Выполните запрос Группа со значением параметра 152; просмотрите результаты запроса.

3. Создайте запрос, в котором выводятся оценки студентов заданной группы по заданной дисциплине.

Для создания запроса, в котором выводятся оценки студентов заданной группы по заданной дисциплине:

- создайте простой запрос на основании таблицы Студенты, выбрав поля Фамилия, Имя, Отчество, Группа; в таблице Экзамены выберите поля Предмет и Оценка; таким образом сформированы шесть полей запроса, связанных между собой посредством схемы данных;
- выберите подробный отчет; введите имя запроса Оценки группы и измените макет запроса;
- в строке Условия отбора для поля Группа введите фразу: [Введите номер группы], в строке Условия отбора для поля Предмет введите фразу: [Введите название предмета];
- выполните запрос, например, для группы 152 и предмета Информатика.

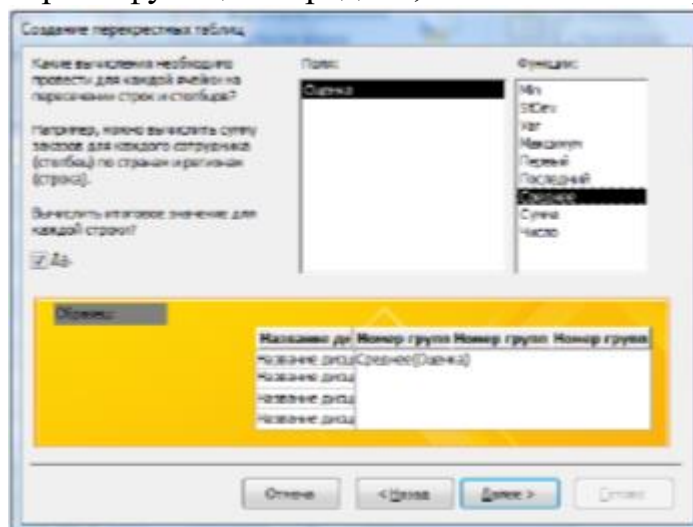
4. Создайте перекрестный запрос, в результате которого создается выборка, отражающая средний балл по дисциплинам в группах.

Для создания перекрестного запроса на выборку, отражающую средний балл по дисциплинам в группах: создайте простой запрос на основе таблицы Студенты (поле Группа), таблицы Экзамены (поля Предмет и Оценка); выберите подробный отчет и задайте имя запроса Оценки по предметам в группе; полученная таблица запроса показана на рисунке:

Группа	Название д.	Оценка
152	Информатика	4
153	Математика	5
151	Физика	4
151	Экономика	4
151	Информатика	5
151	Математика	5
152	Физика	4
151	Экономика	4
151	Информатика	3
151	Математика	5
151	Физика	4
151	Экономика	3

Сохраните запрос и закройте таблицу запроса. Создайте перекрестный запрос, выбрав на вкладке Создать пункт Мастер запросов, в мастере выберите

Перекрестный запрос. Выберите запрос Оценки по предметам в группе; выберите последовательно поля: Название дисциплины (Далее >) и Номер группы (Далее >); выберите функцию Среднее, как показано на рисунке:



Введите название запроса Средние оценки и нажмите кнопку Готово. Откроется таблица перекрестного запроса:

Группа	Название дисциплины	Среднее
151	Информатика	4,2
	Математика	4,5
	Физика	4,5
	Экономика	4
152	Информатика	4,2
	Математика	4,5
	Физика	4,6
	Экономика	4

Сохраните запрос и закройте таблицу запроса.

5. Разработайте запрос на увеличение на 10% стипендии тех студентов, которые получают менее 500 руб.

Для создания запроса на изменение стипендии студентов: создайте простой запрос на основе таблицы Студенты, поля Стипендия; выберите подробный отчет и задайте имя запроса Изменение стипендии; измените макет запроса; в строке Условия отбора введите <500.

На вкладке Конструктор выберите пункт Обновление; в строке Обновление в поле Стипендия введите значение [Стипендия]\*1,1

Выполните запрос, подтвердив готовность на обновление данных; сохраните и закройте запрос; откройте таблицу Студенты и просмотрите изменение стипендии у студентов, получавших менее 500 р.

6. Создайте запрос на удаление из базы данных отчисленного студента, например, гр. 152 Перлова Кирилла Николаевича.

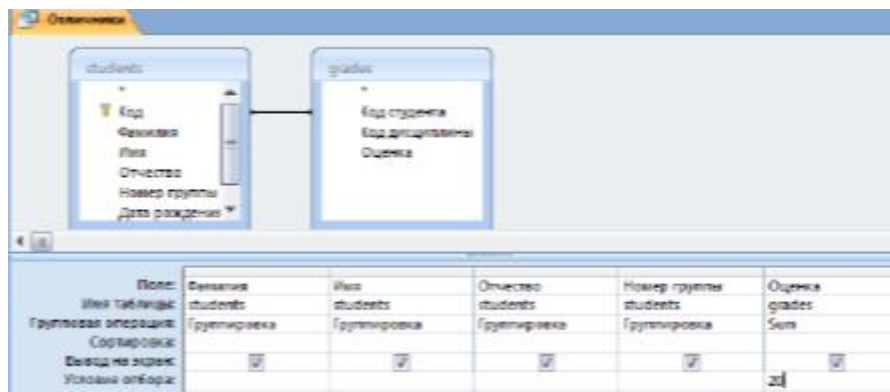
Для создания запроса на удаление из базы данных студента гр. 152 Перлова Кирилла Николаевича:

- создайте простой запрос на основании таблицы Студенты, выбрав поля Фамилия, Имя, Отчество, Группа; выберите подробный отчет; задайте имя запроса Отчисленные студенты; измените макет запроса;

- в окне запроса на выборку в строке Условия отбора введите: в поле Фамилия - Перлов, в поле Имя - Кирилл, в поле Отчество - Николаевич, в поле Номер группы - 152;
- на вкладке Конструктор выберите режим Удаление; перейдите в режим таблицы и проверьте правильность введенных условий отбора;
- перейдите в режим конструктора, на вкладке Конструктор выберите пункт Выполнить;
- сохраните запрос; откройте форму Студенты и удостоверьтесь в удалении записи о студенте Перлове.

7. Разработайте запрос на создание таблицы отличников; отличниками считаются студенты, набравшие за четыре экзамена 20 баллов.

Для создания таблицы отличников: создайте простой запрос на основании таблицы Студенты, выбрав поля Фамилия, Имя, Отчество, Группа, и таблицы Экзамены (поле Оценка); выберите подробный отчет; задайте имя запроса Отличники; измените макет запроса; на вкладке Конструктор выберите команду Итоги; в строке Групповые операции поля Оценка выберите функцию SUM (для создания этого запроса воспользуемся групповой операцией; операция SUM позволит просуммировать оценки студентов по всем дисциплинам); в строке Условия отбора поля Оценка введите 20, как показано ниже:



Просмотрите создаваемую таблицу. Перейдите в режим конструктора; на ленте Конструктор выполните Создание таблицы; в текущей базе данных создайте таблицу с именем Студенты-отличники. Сохраните и закройте запрос на создание таблицы. В левой панели откройте запрос Отличники:



Подтвердите создание новой таблицы с выбранными записями. Откройте полученную таблицу Студенты-отличники и проверьте правильность данных в таблице.

8. Для запросов Группа и Оценки группы разработайте формы для удобного просмотра данных.



## **ПРАКТИЧЕСКАЯ РАБОТА № 20**

### **Тема: Создание формы. Управление внешним видом формы**

**Цель работы:** создавать форму с помощью мастера форм, функции автоформата, а также в режиме конструктора; настраивать внешний вид формы и последовательность перехода по ее полям с учетом предпочтений большинства пользователей и особенностей данных.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### **Справочный материал:**

1. Форма — это объект базы данных, который можно использовать для создания интерфейса пользователя для работы с базой данных.
2. Формы Access предоставляют функциональные возможности для выполнения многих задач, которые нельзя выполнить другими средствами. Формы позволяют выполнять проверку корректности данных при вводе, проводить вычисления, и обеспечивают доступ к данным в связанных таблицах с помощью подчиненных форм. Создание форм, содержащих необходимые элементы управления, существенно упрощает процесс ввода данных и позволяет предотвратить ошибки.
3. Формы предоставляют более удобный способ просмотра и правки данных в таблицах, чем режим Таблицы. Формы содержат так называемые элементы управления, с помощью которых осуществляется доступ к данным в таблицах.
4. Элементами управления являются текстовые поля для ввода и правки данных, кнопки, флажки, переключатели, списки, надписи, а также рамки объектов для отображения графики и объектов OLE.

#### **Содержание работы:**


##### **Задание 1.** Создать форму к базе данных

1. Откройте базу данных «Библиотека» (созданную в Практической работе 14). Создайте форму с помощью Мастера форм на базе таблицы Сотрудник.
2. Откройте таблицу Сотрудник. Выберите закладку Формы, щелкните мышкой по кнопке Другие формы. В появившемся диалоговом окне выберите Мастер форм.
3. В поле Таблицы/Запросы выберите таблицу Сотрудник, в поле Доступные поля выберите поля Фамилия, Имя и перенесите их стрелкой в поле Выбранные поля. Также перенесите поля Дата рождения Номер телефона, Домашний адрес, щелкните по кнопке Далее.
4. Выберите внешний вид формы – Табличный, щелкните по кнопке Далее.
5. Выберите требуемый стиль (н-р, Обычная), щелкните по кнопке Далее.
6. Задайте имя формы Личные данные и щелкните по кнопке Готово. В результате получите форму, в которой можно менять данные и вводить новые значения.
7. Закройте форму.
8. Аналогично создайте формы к двум другим таблицам БД, разных видов.

##### **Задание 2.** Создать форму с помощью инструмента Пустая форма

1. Создайте форму Личные данные с помощью инструмента Пустая форма



2. На вкладке Создание в группе Формы щелкните Пустая форма. Access открывает пустую форму в режиме макета и отображает область Список полей.
  3. В области Список полей щелкните знак плюс (+) рядом с таблицей или таблицами, содержащими поля, которые нужно включить в форму.
  4. Чтобы добавить поле к форме, дважды щелкните его или перетащите его на форму. Чтобы добавить сразу несколько полей, щелкните их последовательно, удерживая нажатой клавишу CTRL. Затем перетащите выбранные поля на форму.
  5. Закройте окно списка полей.
  6. Перейдите в режим Конструктора
- Примечание 1. Размер окошка для названия поля и для его значений меняются мышкой. Для этого выделите черный квадратик рамки (рамка станет цветной), установите курсор на границу рамки и с помощью двунаправленной стрелки измените размеры рамки.
- Примечание 2. С помощью кнопок панели инструментов Шрифт меняйте соответственно цвет фона, текста, линии/границы и т.д.
7. Расположите элементы удобно по полю.
  8. Задайте размер текста поля Фамилия равным 24 пт, шрифт - синего цвета.
  9. Сохраните форму с именем Данные студентов.
  10. Посмотрите все способы представления форм: в режиме Конструктора, режиме Макета и режиме Форм.
  11. Закройте форму, сохранив ее.
  12. Добавьте в таблицу Личные данные логическое поле Институт (т.е., собирается ли в дальнейшем учащийся поступать в институт). Значение этого поля ДА или НЕТ.
  13. Откройте таблицу Личные данные в режиме Конструктор. Добавьте поле с именем Институт и типом Логический. Закройте таблицу.
  14. Перейдите на закладку Формы и откройте форму Личные данные в режиме Конструктор
  15. Щелкните по кнопке Список полей на панели инструментов, выделите название Институт и перетащите его мышкой в область данных, появится значок  и надпись Институт.
  16. Расположите новые элементы по правилам оформления формы (с помощью мыши).
  17. Закройте Список полей
- Примечание 3. Если флажок установлен, поле в таблице имеет значение ДА, если снят, то НЕТ.
18. Перейдите в режим Раздельная форма и посмотрите записи. Установите флажки у восьми разных сотрудников.
  19. Закройте форму, ответив утвердительно на вопрос о сохранении.

**Задание 3.** Создайте 3 любые формы к базе данных Студенты.

## ПРАКТИЧЕСКАЯ РАБОТА № 21

### Тема: Создание кнопочной формы

**Цель работы:** научиться создавать формы ввода-вывода; научиться создавать кнопочные формы.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### **Справочный материал:**

##### Создание кнопочной формы.

Кнопочное меню представляет собой форму, на которой расположены элементы управления – кнопки с поясняющими надписями. Щелчок на кнопке открывает соответствующую таблицу, запрос, форму или отчет. Меню – удобный инструмент работы с базами данных, и он практически всегда присутствует в базах, созданных для предприятий или фирм.

Панель инструментов Конструктор форм (Form Design) в Access содержит кнопки, предназначенные для разработки форм.



Основное назначение кнопок панели элементов (Toolbox):

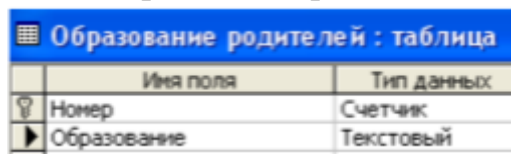
- выбор объектов (Select Objects) — выделение щелчком мыши элемента, раздела или формы, выделение группы элементов путем обвода курсором мыши рамки вокруг них;
- мастера (Control Wizards) — включение или отключение мастера для создания элементов управления (например, элементов Список, Поле со списком);
- надпись (Label) — создание текстов постоянных заголовков, примечаний, инструкций, не связанных с другими элементами управления;
- поле (Text Box) — создание элемента типа Свободный (Unbound), который может быть затем связан с полем таблицы или запроса, или использован для создания вычисляемого поля формы;
- группа переключателей (Option Group) — размещение набора флажков, переключателей или выключателей;
- выключатель (Toggle Button) — создание выключателей;
- переключатель (Option Button) — выбор альтернативных значений;
- флажок (Check Box) — выбор набора из возможных значений;
- поле со списком (Combo Box) — создание поля с раскрывающимся списком значений поля из записей некоторой таблицы базы (значение может вводиться в поле пользователем или выбираться из списка);
- список (List Box) — создание всегда раскрытого списка значений, которые при связи с полем таблицы базы являются единственным источником ввода в поле;
- кнопка (Command Button) — создание командной кнопки, с помощью которой может быть выполнено одно из действий, например, переход по записям, открыта форма, напечатан отчет и реализованы другие функции Access;

- рисунок (Image) — для отображения не редактируемого рисунка, не являющегося объектом OLE;
- свободная рамка объекта (Unbound Object Frame) — отображение свободного объекта OLE, который остается неизменным при переходе по записям;
- присоединенная рамка объекта (Bound Object Frame) — отображение объектов OLE, сохраненных в поле базового источника записей формы;
- разрыв страницы (Page Break) — начало нового экрана в форме, новой страницы в печатной форме (отчете);
- вкладка (Tab Control) — создание вкладок в форме, на каждой из которых могут размещаться свои элементы управления;
- подчиненная форма/отчет (Subform/Subreport) — вывод данных из таблиц, связанных с таблицей-источником формы;
- линия (Line) — разграничение разделов в форме (отчете);
- прямоугольник (Rectangle) — создание рамки при оформлении;
- другие элементы (More Controls) — открытие обширного списка дополнительных элементов, при выборе из которого в форме будет создан соответствующий элемент.

### Содержание работы:

**Задание 1.** Создать и заполнить базу данных «Классный руководитель».

1. Запустите Microsoft Access - Новая база данных - Имя файла: Классный руководитель – и Создать.
2. Создание таблицы в режиме конструктора - Выбрать тип данных (см. Рис. 1.) - Сохранить таблицу, как “Образование родителей”




Имя поля	Тип данных
Номер	Счетчик
Образование	Текстовый

Рис.1

Образование родителей можно заполнить в табличном режиме как: высшее, средне- специальное, среднее.

3. Создание таблицы в режиме конструктора - Ввести данные согласно Рис. 2
- Примечание. Для поля “Место работы папы” и “Место работы мамы” в опции Общие размер задать 255 (символов) - Сохранить таблицу, как: “Личные данные родителей”.



Имя поля	Тип данных
Номер	Счетчик
Папа	Текстовый
Образование папы	Числовой
Телефон Папы	Текстовый
Место работы Папы	Текстовый
Адрес эл почты папы	Текстовый
Мама	Текстовый
Образование мамы	Числовой
Телефон Мамы	Текстовый
Адрес эл почты мамы	Текстовый
Место работы Мамы	Текстовый
Количество детей	Числовой

Рис.2

4. Создание таблицы в режиме конструктора - Ввести данные - Примечание. Для Дата Рождения формат задать согласно Рис.3. (Маска ввода позволит вводить только последние две цифры года рождения) Для полей: место рождения, место жительства, кружки/секции – задать формат поля в 255 символов. - Сохранить таблицу “Личные данные ученика”.

Имя поля	Тип данных
Номер Ученика	Счетчик
Фамилия	Текстовый
Имя	Текстовый
Отчество	Текстовый
Дата Рождения	Дата/время
Место Рождения	Текстовый
Место жительства	Текстовый
Дом телефон	Текстовый
Адрес эл почты	Текстовый
Фотография ученика	Поле объекта
Кружки, секции	Текстовый
Многодетная семья	Логический
Неполная семья	Логический
Семья с неработающими родителями	Логический
Социально-правовая ситуация	Логический
Примечание	Поле МЕМО
Номер Родителей	Числовой

Общие Подстановка

Формат поля Краткий формат даты

Маска ввода 99.99.00;0

Подпись Дата Рождения

Рис.3

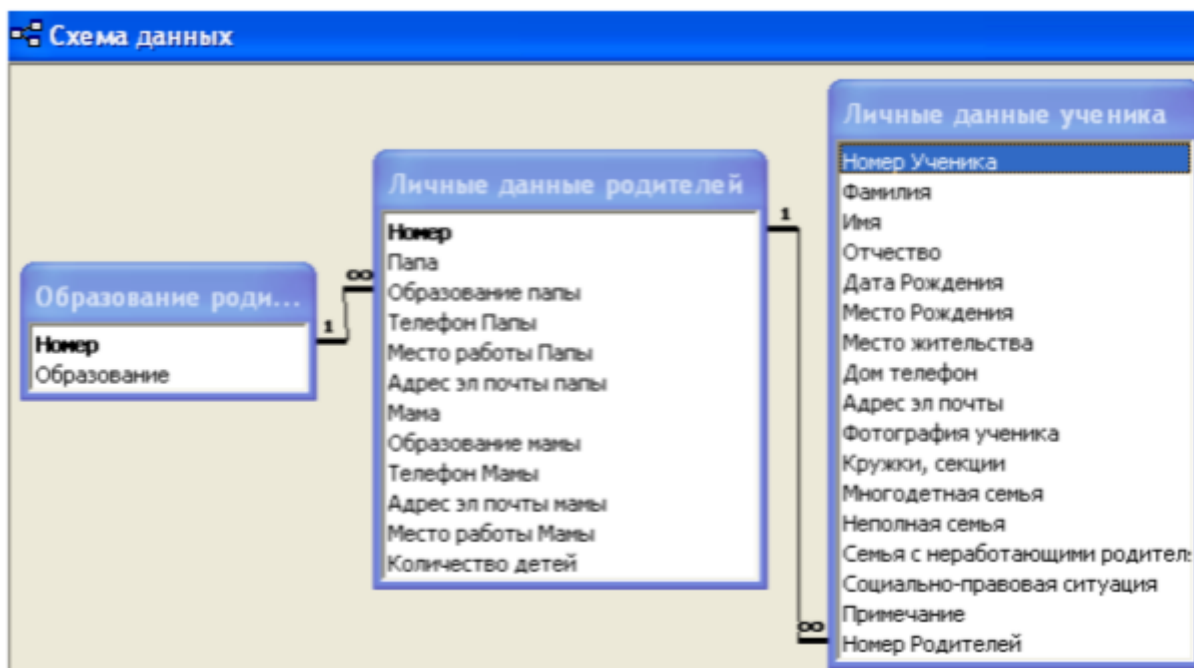
5. И, наконец, последняя таблица: “Информация о состоянии здоровья” заполняется аналогично, см. Рис. 4

Имя поля	Тип данных
Номер	Счетчик
Осанка	Текстовый
Слух	Текстовый
Зрение	Текстовый
Хронические заболевания	Текстовый
Прочее	Текстовый

Рис.4

6. Создание схемы данных.

Выполните команду: Сервис - Схема данных - Поочередно выберите таблицы: “Образование родителей”, “Личные данные родителей”, “Личные данные ученика” - Добавить. Далее, ухватив мышкой за название нужного поля, удерживая левую кнопку мыши, перетащите в нужное поле соседней таблицы. Тип отношений, Microsoft Access предложит сам: один-ко-многим, останется только включить флажок обеспечение целостности данных.



Связывание таблиц необходимо для того, чтобы в последствии, при создании запросов на выборку данных, мы смогли использовать данные из трех таблиц. Причем, таблица “Личные данные родителей” является родительской по отношению к таблице “Личные данные ученика”, это связано с тем, что в одном классе могут учиться несколько детей одних родителей, т.е. одна учетная запись (номер) родителей, связана по схеме один-ко-многим с несколькими учетными записями учеников.

7.Создадим список, по которому возможно осуществить выбор образования родителей. Установите курсор на таблице “Личные данные родителей” и вызовите Конструктор -поле: Образование папы, тип данных: числовой - выберите внизу опцию Подстановка –настройте согласно рисунка

Личные данные родителей : таблица

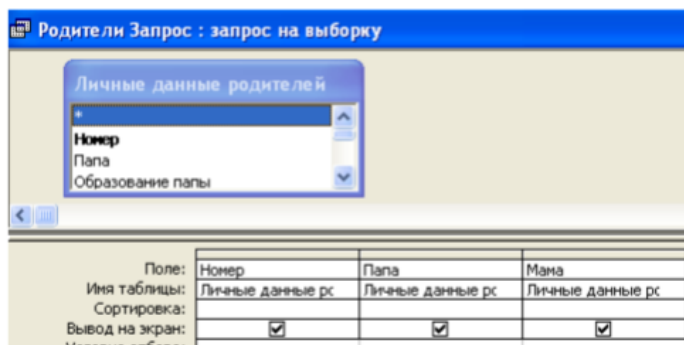
Имя поля	Тип данных
Номер	Счетчик
Папа	Текстовый
Образование папы	Числовой
Телефон Папы	Текстовый
Место работы Папы	Текстовый
Адрес эл почты папы	Текстовый
Мама	Текстовый
Образование мамы	Числовой
Телефон Мама	Текстовый
Адрес эл почты мамы	Текстовый
Место работы Мама	Текстовый
Количество детей	Числовой

Общие	Подстановка
Тип элемента управления	Поле со списком
Тип источника строк	Таблица или запрос
Источник строк	Образование родителей
Присоединенный столбец	1
Число столбцов	2
Заглавия столбцов	Да
Ширина столбцов	80% 2,54см
Число строк списка	8
Ширина списка	2,54см
Ограничить список	Да

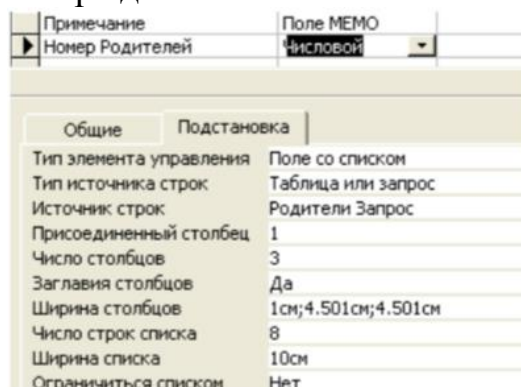
То же самое проделать для поля “Образование мамы”. Теперь, при вводе данных об образовании родителей, достаточно будет выбрать значения из списка указанных в таблице: “Образование родителей”. Закройте и сохраните.

8. Создадим Запрос Родители. Итак, перейдите в объект базы данных Запросы - Создать - Конструктор - Таблицы: Личные данные родителей - Имя таблицы: Личные данные родителей - Поле: номер, папа, мама.



Сохранить под именем “Родители”.

9. Вернемся в объект таблицы - таблица: Личные данные ученика - Конструктор – Номер родителей, тип данных: Числовой - опция: Постановка - Выполните настройки выбора данных согласно Рис.

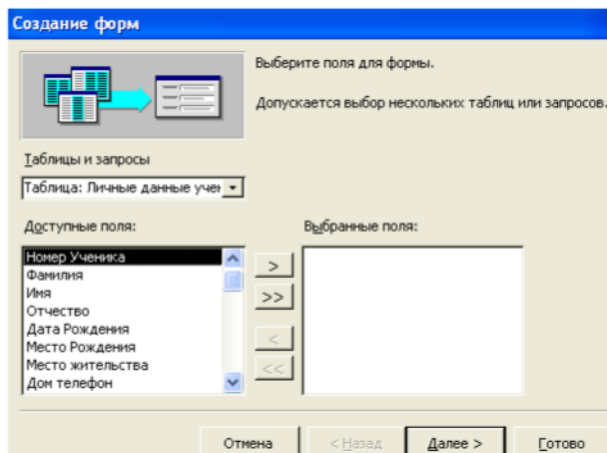


Во время ввода, редактирования данных на ученика, пользователь сможет выбрать номер родителей, причем в списке будет указан не только номер, но и ФИО папы и мамы. С таблицами все. Имеем 4 таблицы и один запрос, т.е. структура базы данных создана!

10. Создание запросов и форм базы данных.

Для того чтобы ввести данные в таблицы, как правило, создают формы. Создание форм “Личные данные ученика”, “Личные данные родителей” и “Информация о состоянии здоровья”.

Создать - Мастер форм - таблица: Личные данные ученика - двойной стрелкой выбираем все - в один столбец - Готово.





Аналогично создаем форму “Личные данные родителей” и “Информация о состоянии здоровья”, в каждую из которых помещаем данные соответствующих таблиц.

Итак, выделяем форму “Личные данные ученика”, запускаем панель элементов.

ичные данные ученика : форма

Заголовок формы

Область данных

Номер

Фамилия

Имя

Отчество

Дата Рождения

Место Рождения

Место жительства

Дом телефон

Адрес эл почты

Номер Родителей:

Номер

Фотография ученика

Многодетная семья

Социально-правовая ситуация

Семья с неработающими родителями

Неполная семья

Кружки, секции

Кружки, секции

Примечание

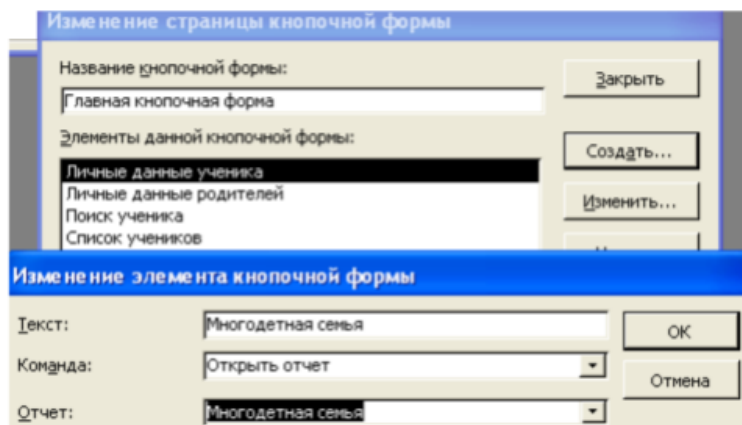
Панель элементов

Кнопка

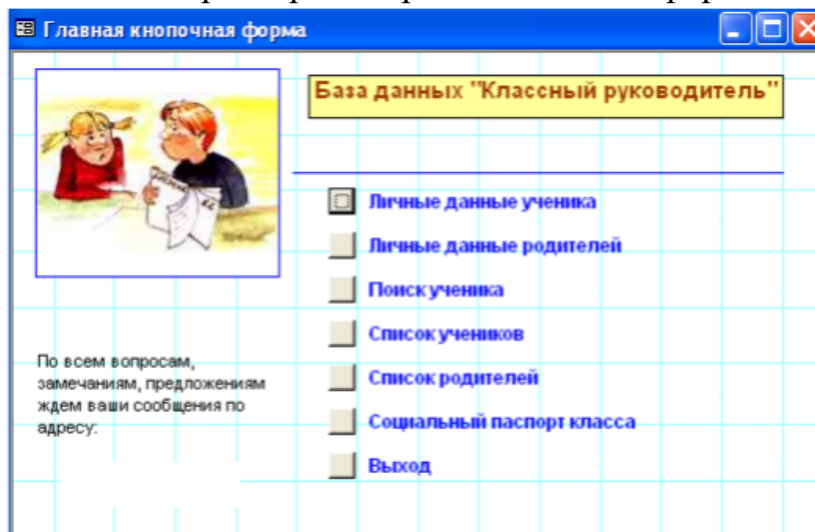
Для удобства работы с формой “Личные данные родителей”, можно создать отдельный запрос (далее форму на запрос) на выборочные данные ученика и внедрить кнопку, открывающую данную форму.

### 13. Создание главной кнопочной формы.

Выполните команду: Офис – Параметры Access – Настройка – Выбрать команды из – Вкладка «Работа с базами данных» - Диспетчер кнопочных форм – Добавить – Панель быстрого доступа. И поочередно создайте кнопки: Личные данные ученика (форма), Личные данные родителей (форма), Поиск ученика (форма), Список класса (отчет), Список родителей (отчет), Многодетная семья (отчет) и т.д. В принципе, можно создать отдельную форму: Социальный паспорт класса, и в нее внести отчеты: многодетная семья, социально-правовая ситуация и т.д., и затем внести ее в главную кнопочную форму.




14. Для того чтобы при запуске нашей базы данных запускалась главная кнопочная форма, необходимо выполнить: Офис – Параметры Access – Текущая база данных – Форма просмотра – Кнопочная форма



## Задание 2. Создать кнопочную форму с помощью Конструктора

1. Щелкните по кнопке Создать. Выберите Конструктор. Появится пустая форма. Задайте мышкой ширину формы, равную 10см, а высоту – 7см.
2. Выберите на панели инструментов Элементы управления > кнопку Аа – Надпись. Курсор мышки примет вид крестика с «приклеенной» буквой А. Щелкните мышкой по месту начала надписи и введите: БАЗА ДАННЫХ «БИБЛИОТЕКА».
3. Нажмите клавишу Enter. Выберите размер букв 18, а выравнивание - по центру. Цвет фона – голубой. Растяните мышкой надпись на ширину окна.



4. Выберите на панели элементов значок  - Кнопка. Щелкните мышкой по тому месту области данных, где должна быть кнопка. Появится диалоговое окно Создание кнопок.

5. Выберите категорию Работа с формой, а действие Открыть форму, и щелкните по кнопке Далее.

6. Выберите форму Штатное расписание, открываемую этой кнопкой щелкните по кнопке Далее. В следующем окне также щелкните по кнопке Далее.

7. В следующем окне поставьте переключатель в положение Текст, наберите в поле слова Штатное расписание и щелкните по кнопке Далее.

Примечание 4. Размер и расположение кнопок можно менять мышкой в режиме Конструктор.

8. Аналогично создайте кнопки для форм Сотрудники и Состав семьи.

9. Перейдите в режим формы. Теперь при щелчке мышью по соответствующим кнопкам будут открываться соответствующие формы для работы.

10. Закройте форму.

## ПРАКТИЧЕСКАЯ РАБОТА № 22

### Тема: Создание меню различных видов. Модификация и управление меню. Макросы и модули

**Цель работы:** овладение навыками создания и модификации структуры меню

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.


#### Справочный материал:

Для хорошего тона создают общий интерфейс базы данных, которая сдаётся в эксплуатацию. Пользователь может выбрать направление на общем интерфейсе, а затем переходить к группе задач, которые его интересуют. В учебных целях, создадим ещё одну форму, которая будет начальной для работы с базой данных. Конечно, рассмотренные варианты, создания интерфейсов с использованием макросов, гиперссылок, формы навигации, могут лечь в основу общего интерфейса, но мы постараемся создать новый вид интерфейса, основанный на элементах управления, которые ещё не рассматривались.

#### Содержание работы:

**Задание 1.** Создать меню для разработанной базы данных Библиотека.

##### 1. Подготовка формы

Создайте основу формы, чтобы в дальнейшем её дополнять новыми элементами, для этого достаточно спроектировать заголовок формы. Откройте форму в режиме Конструктора, в поле заголовка вставьте рисунок, отражающий логотип организации, воспользовавшись пиктограммой , отмасштабируйте рисунок. Обратите внимание, что одновременно с рисунком на поле заголовка появилось окно для надписи, заполните его (не забывайте про свойства этого элемента), как показано на рисунке 1. Сохраните форму, например под именем «Общая форма».

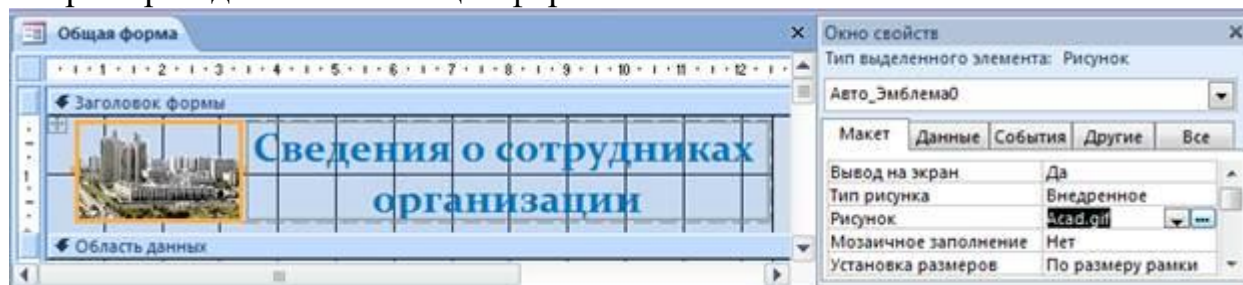


Рис. 1. Создание заголовка для общей формы

##### 2. Использование группы переключателей для выбора варианта просмотра объектов базы данных

Откройте созданную форму в режиме Конструктор. На поле «Область данных» перенесите элемент управления «Группа переключателей» (Рис. 2), предварительно раскрыв на панели список с элементами управления. Переключатели обычно применяются в тех случаях, когда предлагается выбрать один вариант из нескольких предложенных (как правило, количество переключателей создают небольшое, в противном случае, удобнее использовать элемент управления – Список).

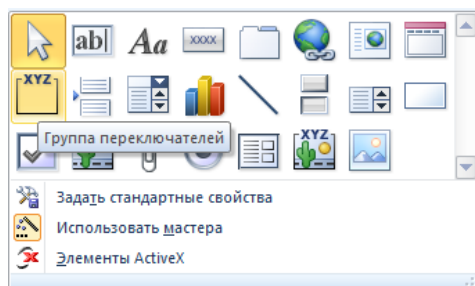


Рис. 2. Раскрытый список элементов управления

В появившемся окне (Рис. 3) каждая строка соответствует названию переключателя. Чтобы добавить переключатель, заполните текстом строку со звёздочкой (Рис. 3) и нажмите на кнопку **Далее >**. Мастер создания переключателей будет последовательно предлагать диалоговые окна, в которых следует выбирать варианты решения (Рис. 4, 5).

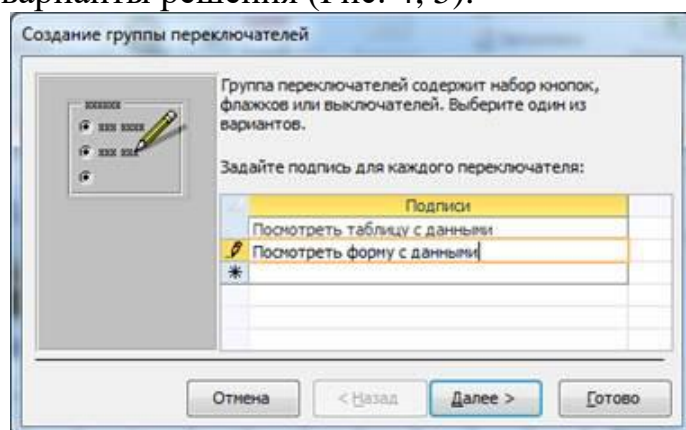


Рис. 3. Ввод текста около переключателя

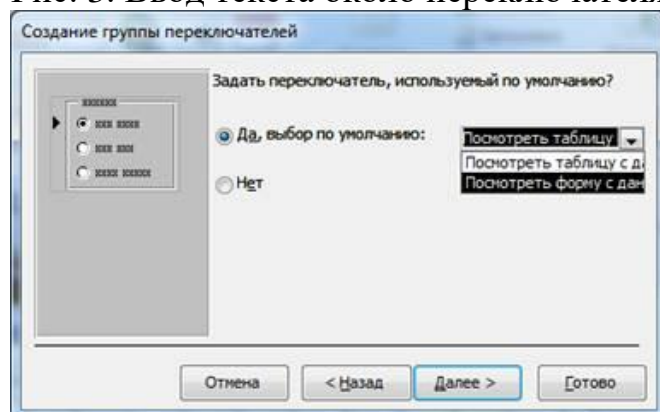


Рис. 4. Задание начального состояния переключателя

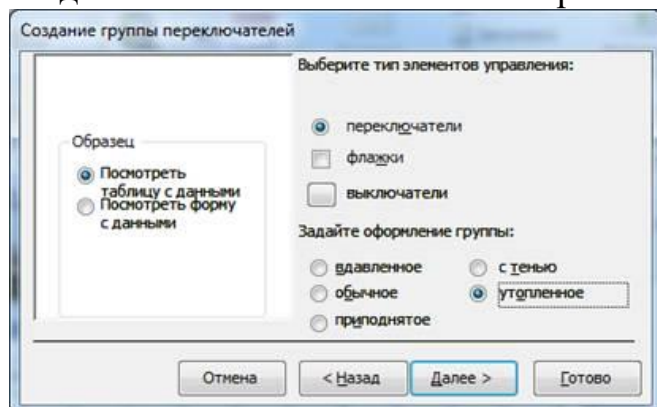



Рис. 5. Выбор варианта представления блока с переключателями

Создайте два элементарных макроса, которые будут использоваться для открытия таблицы «Личные сведения» при щелчке по первому переключателю и для открытия формы, например, «Форма с макросами». Напомним, что для создания независимого макроса, необходимо на вкладке «Создание» щёлкнуть по значку  - Макросы. Открыть список макрокоманд, выбрать **ОткрытьФорму**, и заполнить бланк макроса (Рис. 6). В рассматриваемом примере показан макрос для открытия формы, этот макрос понадобится для подключения к переключателю номер 2. Сохраните макрос, например с именем «Подключить форму». Для переключателя с номером 1 потребуется макрос для открытия таблицы, создайте новый по аналогии с примером и сохраните под именем, например, «ЛичныеСвед».

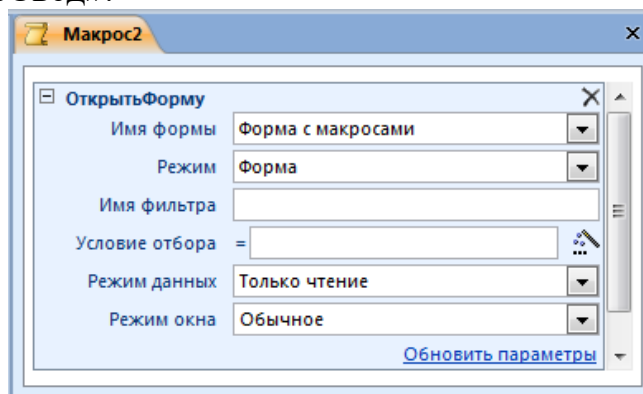



Рис. 6. Бланк макроса для открытия формы «Форма с гиперссылками»

Проведите операцию назначения макроса событию, которое вызывает пользователь при взаимодействии с переключателем. Выделите на форме в режиме Конструктор первый переключатель, в окне свойств активизируйте ярлык «События», для строки «Кнопка вниз» выберите из списка  макрос «ЛичныеСвед» (Рис. 7). Почему выбрали событие «Кнопка вниз»? Когда создавали поле с переключателями, тогда поставили отметку, что первый будет активным, это значит, что он получил фокус, если этому событию назначить макрос, то при открытии формы, автоматически будет открываться таблица «Личные сведения», что бы этого не случилось, выбрано событие, которое имитирует нажатие мышкой на кнопку в элементе переключателя.

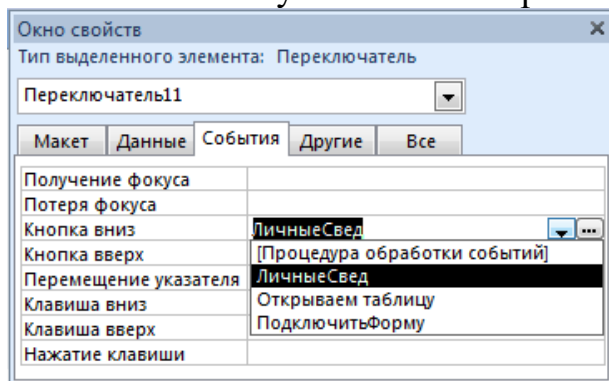


Рис. 7. Назначение макроса событию «Кнопка вниз»

Второй переключатель становится активным только после того, как по кнопке будет проведён щелчок мыши, это значит, либо в нём устанавливается фокус, либо кнопка идёт вниз (Рис. 8). Поэтому макрос «ПодключитьФорму» можно привязать к одному из указанных событий. Кстати, событие «Кнопка

вверх» происходит в тот момент, когда производится нажатие на любую другую кнопку, следовательно, можно написать макрос для этого события, который будет управлять закрытием формы, таблицы или запроса.

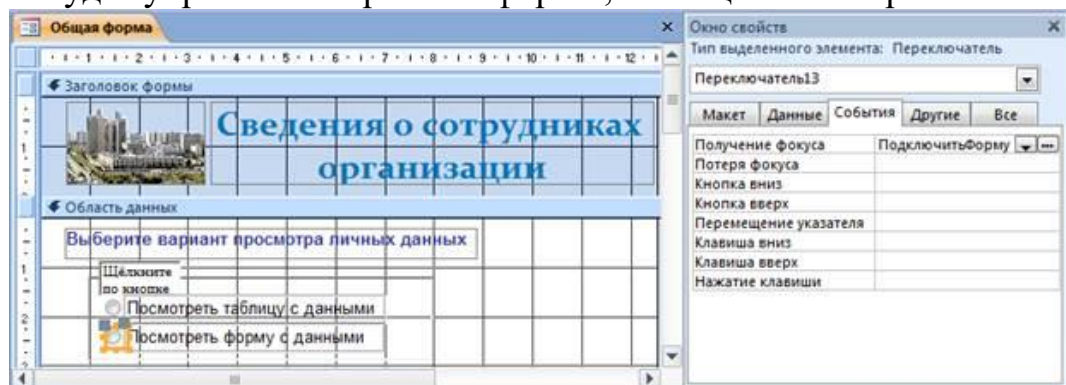








Рис. 8. Расположение группы переключателей на общей форме

### 3. Использование списков на форме

Среди элементов управления существует два элемента:  - Поле со списком и  - Список. Которые имеют аналогичные свойства, разница заключается в том, что поле первый элемент имеет кнопку для раскрытия списка, а второй имеет только линейку прокрутки. Но, при внедрении на форму списка с помощью Мастера, можно добиться различных эффектов. Рассмотрим пример, когда с помощью элемента управления  «Список» на форме можно отобразить таблицы с данными.

3.1. Откройте форму «Общая форма». Раскройте окно с элементами управления проверьте состояние команды  **Использовать мастера**, этот элемент должен быть активным. Поместите на форму элемент управления  - Список. В окне мастера «Создание списков» выберите отметку  **Объект "список" получит значения из другой таблицы или другого запроса.** (Рис. 9).

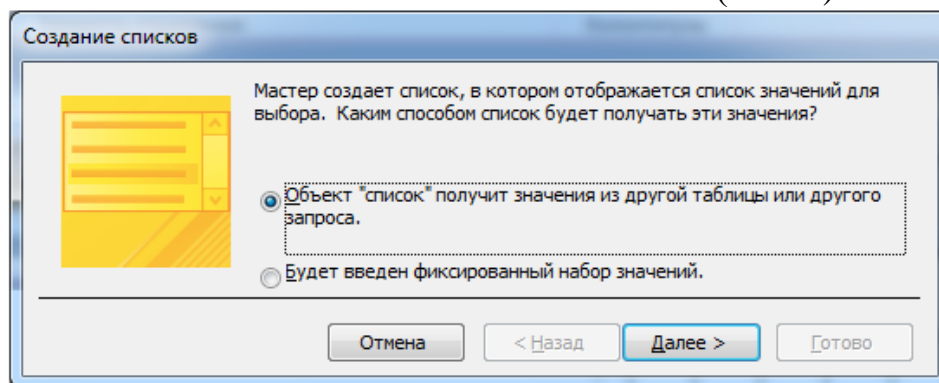



Рис. 9. Окно мастера для создания списка на форме

3.2. Напомним, что при работе с Мастером следует в последующих окнах выбирать вариант, который устраивает разработчика и нажимать на кнопку  **Далее >**. На последующих рисунках отображены шаги работы с Мастером при создании списка на форме. Так, прежде всего, следует выбрать источник данных, для примера это будет таблица «Личные сведения» (Рис. 10). На следующем этапе (Рис. 11) в будущий список переносят в любой



последовательности наименования полей из источника данных. Для удобного восприятия данных, устанавливается порядок сортировки (Рис. 12).

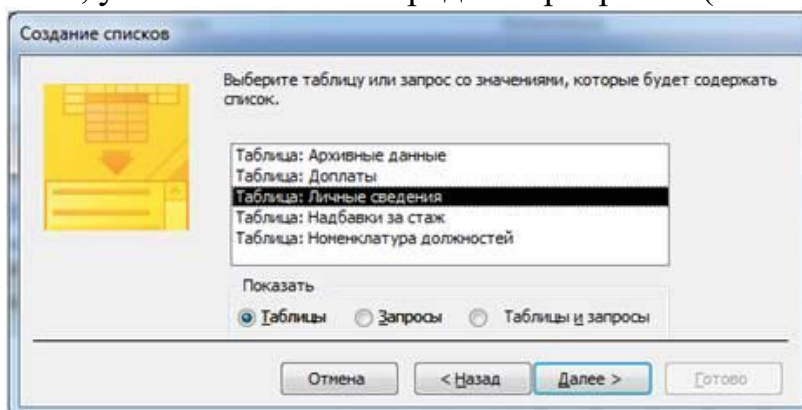


Рис. 10. Выбор наименования таблицы, поля из которой понадобятся

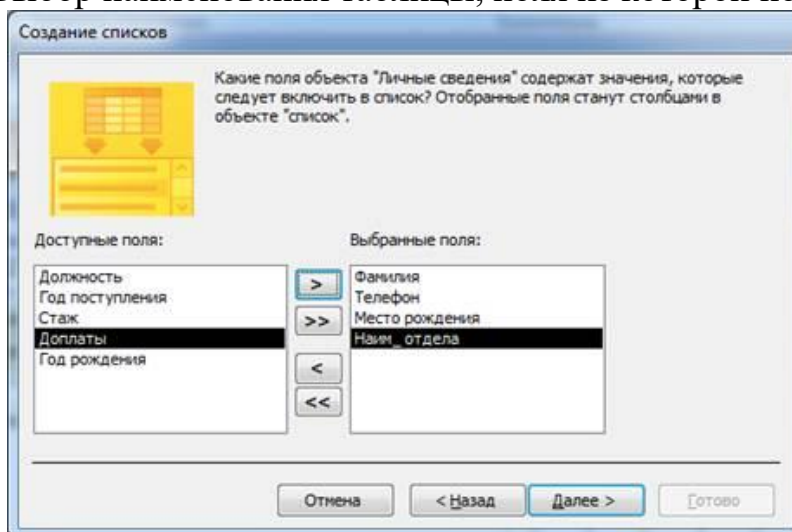


Рис. 11. Отбор полей для таблицы

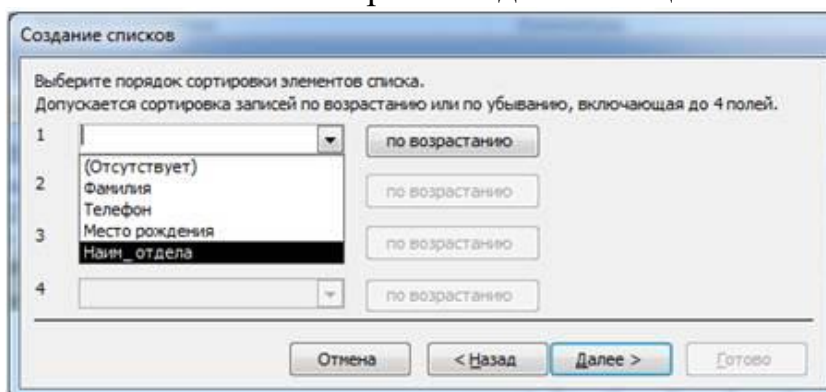


Рис. 12. Установка порядка сортировки в таблице

На этапе просмотра готовой таблицы (Рис. 13) уберите отметку в элементе ☐ Скрыть ключевой столбец (рекомендуется), так как в таблице «Личные сведения» ключевым элементом было назначено поле «Фамилия».

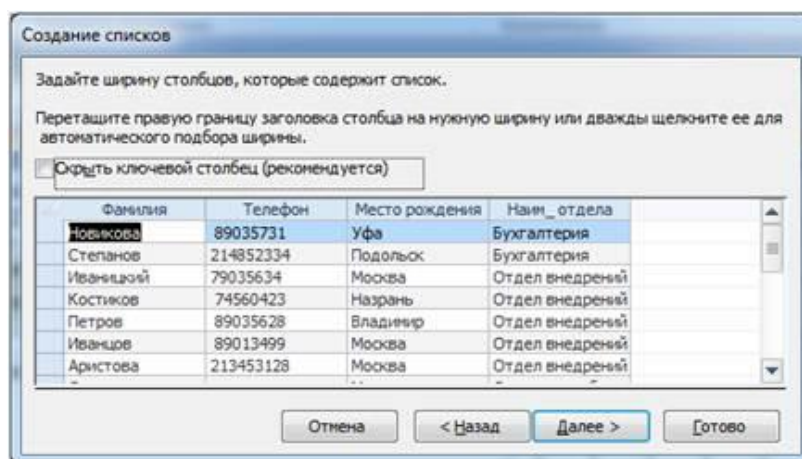


Рис. 13. Отображение таблицы в Мастере

Обратите внимание, что на следующем шаге будет предложено выбрать поле, которое в дальнейшем можно будет использовать для формирования запроса к данной таблице, например, выделите поле «Место рождения» (Рис. 14).

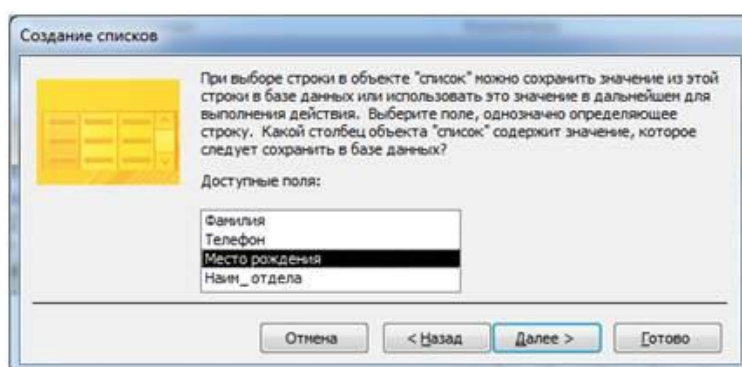


Рис. 14. Выбор столбца в объекте для дальнейшего использования

На заключительном этапе работы с Мастером (Рис. 15) требуется задать заголовок для списка, который будет отображаться в виде таблицы на форме.

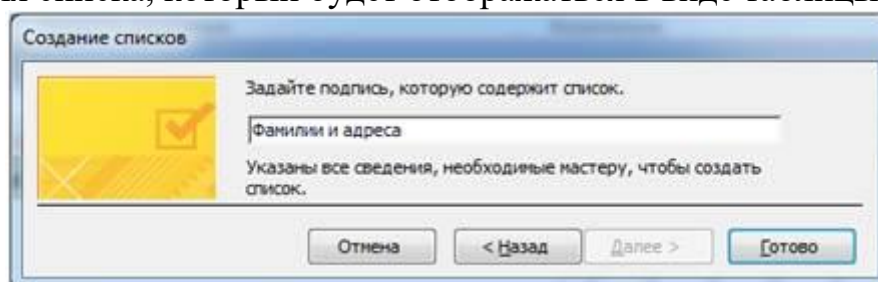



Рис. 15. Задание подписи, который будет содержать созданный список

Нажав на кнопку **Готово**, вы увидите в режиме Конструктор форму (Рис. 16), на которой будут только заготовка для списка и его заголовок. Можете воспользоваться окном свойств этого элемента, что бы установить те параметры, которые вас устраивают. Чтобы увидеть на форме результаты творчества, перейдите в режим формы , на которой отобразится список, созданный с помощью Мастера (Рис. 17).

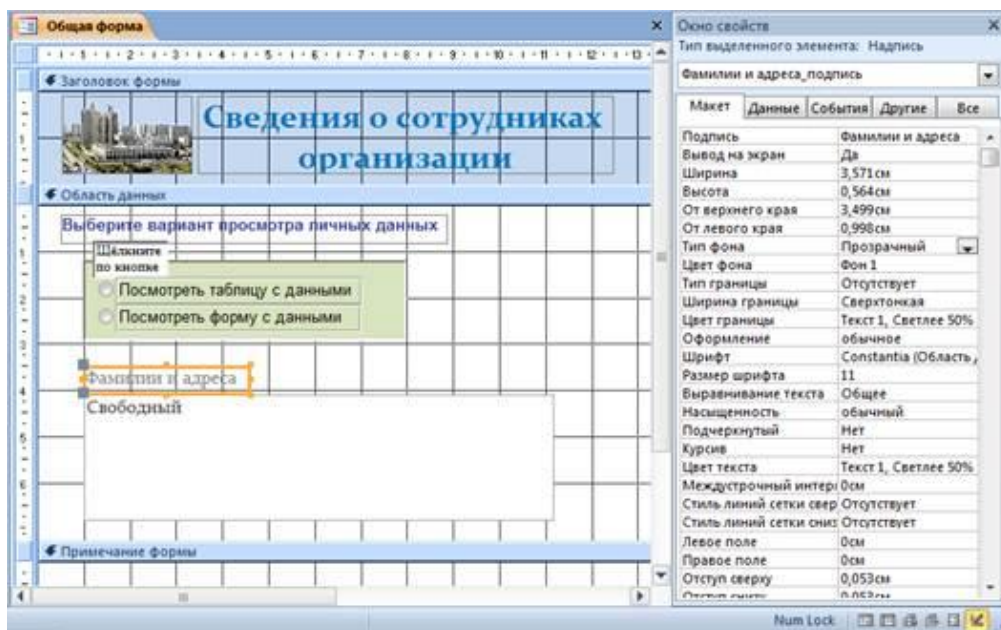


Рис. 16. Отображение созданного списка на форме в режиме Конструктор

Фамилии и адреса

Новикова	89035731	Уфа	Бухгалтерия
Степанов	214852334	Подольск	Бухгалтерия
Иваницкий	79035634	Москва	Отдел внедрений
Костиков	74560423	Назрань	Отдел внедрений
Петров	89035628	Владимир	Отдел внедрений
Иванцов	89013499	Москва	Отдел внедрений

Рис. 17. Список на форме, который был создан с помощью Мастера



Как видите, без использования, каких либо средств, на форме можно расположить список, сформированный в виде таблицы. Следует отметить, что данный элемент управления позволяет подключать программные модули, написанные на Visual Basic, чтобы вести обработку данных.

#### 4. Создание программных кодов для обработки событий




Событием называется действие, которое вызывает пользователь или генерирует система. Событие производится над объектом, поэтому необходимо совершить действие, например, щёлкнуть мышкой, нажать на клавишу, передвинуть указатель мыши. Система совершает действия при загрузке формы, закрытии окон и т.п. Общение пользователя с интерфейсом любого приложения состоит из цепочки событий, которые он совершает. После того, как событие совершено, необходимо подключить программу, в которой заложен алгоритм изменения свойств определённого элемента управления, либо осуществление логических операций, либо проведение преобразования данных (в том числе и выполнения вычислений). Программные коды формируются с помощью макрокоманд, когда создаётся макрос. Более интересно создавать программные модули с помощью языка программирования Visual Basic (VB). В данном разделе автор поставил перед собой задачу – показать некоторые приёмы разработки программ обработки событий с помощью самостоятельного использования VBE – Visual Basic Editor. Безусловно, для серьёзных программ требуются знания в области алгоритмизации и программирования, а так же опыт работы



с VB (автор не теряет надежду, что большинство, изучающих практику работы с приложениями MS Office, начнут серьезно заниматься программированием).

Попытаемся использовать, накопленный опыт по разработке интерфейсов базы данных, при создании программных модулей. В качестве учебной задачи, остановимся на совершенствовании начальной формы под именем «Общая форма». Для этого разместим на форме элементы управления:  - Поле со списком и  - Кнопка. Затем напишем небольшую программу для этих элементов управления, а затем покажем, как можно обойтись без кнопки. В поле со списком должны войти наименования форм, которые созданы для подразделений организации. А при выборе из списка необходимого наименования формы, должна открываться форма. Управлять этим процессом будем с помощью кнопки.

#### 5. Размещение элементов управления на форме

Создадим на форме Поле со списком с помощью Мастера. Прежде убедитесь в том, что команда  **Использовать мастера** находится в активном состоянии (её можно увидеть в окне с элементами управления). На этот раз в элементах управления выберите , после открытия окна выберите режим  **Будет введен фиксированный набор значений.** (Рис. 18).

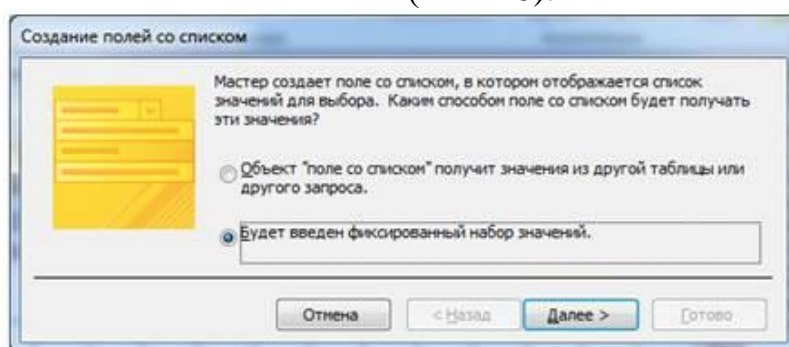


Рис. 18. Выбор варианта формирования списка

На следующем шаге заполните строки для столбца (выберите один столбец), в список введите наименования готовых форм, которые обозначены в окне переходов базы данных. Для начала введите три строки (Рис. 19), в дальнейшем будет показано, как такой список можно дополнить новыми записями. Введите заголовок для списка (Рис. 20).

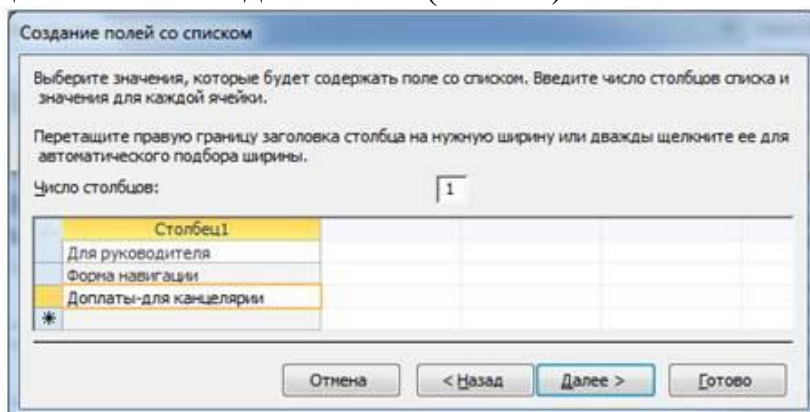


Рис. 19. Заполнение строк таблицы наименованиями форм

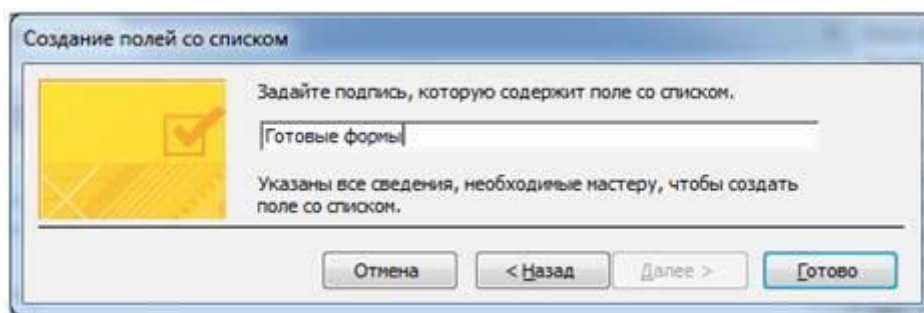





Рис. 20. Ввод заголовка для списка

В режиме Конструктор на форме «Общая форма» появится только заголовок для списка и окно списка. Опять воспользуйтесь окном свойств и отредактируйте текст, размер, фон и т.п. Увидеть результат создания списка можно после того, как форма будет запущена. Если возникает необходимость дополнить список новыми записями, то это можно сделать непосредственно в

режиме  - Форма. Раскройте список, появится значок  - Изменить элементы списка (Рис. 21 слева), щёлкните по значку, после чего в него можно добавлять новые наименования, удалять и редактировать записи (Рис. 21 – справа). В примере добавлена новая строка «Список для отдела кадров». Если вы находитесь в режиме Конструктор, то щёлкните правой кнопкой мыши по элементу список, а затем в раскрывшемся меню по строке  Изменить элементы списка...

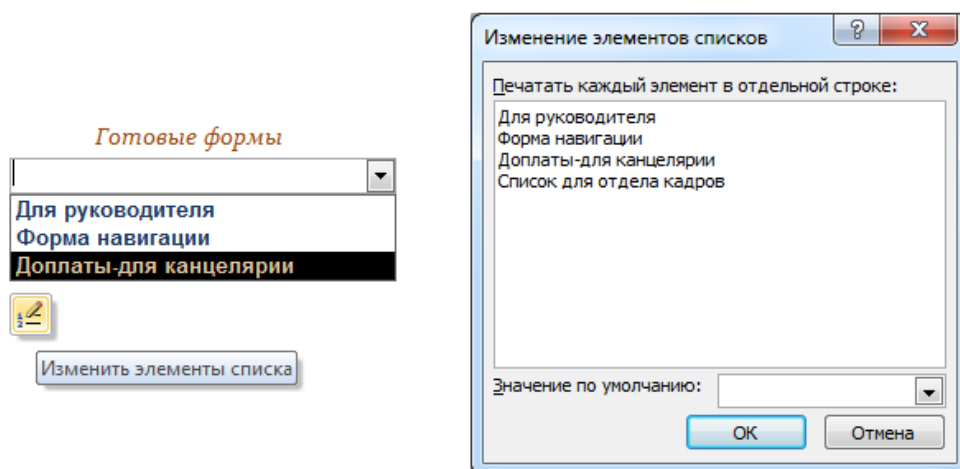


Рис. 21. Вызов диалогового окна для изменения элементов списка



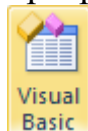
Теперь, на форме создайте элемент  - Кнопка, в этом случае  Использовать мастера следует отключить, т.к. в режиме Конструктор можно использовать свойства этого элемента и обойтись без Мастера создания кнопок, это будет гораздо быстрее. На рисунке 22 показан фрагмент формы с окном свойств кнопки. При работе со свойствами: *Подпись*, *Расположение подписи к рисунку*, *Рисунок* (открыто окно для выбора варианта рисунка, который можно разместить на кнопке) и другие, которые позволяют добиться определённого результата.



Рис. 22. Размещение рисунка и подписи на кнопке с помощью окна свойств

## 6. Разработка программных кодов

Цель создания программных кодов – организовать открытие готовых форм при выборе определённого названия из списка на форме. Чтобы создать программу, необходимо открыть редактор Visual Basic for Applications (VBA).



Эту операцию можно выполнить с помощью кнопки на панели, предварительно открыв вкладку «Работа с базами данных», после чего будет открыт редактор, внешний вид которого показан на рисунке 23. Редактор имеет собственный интерфейс, на панели которого располагается строка меню и кнопки быстрого вызова. Основное поле редактора делится на несколько областей, в рассматриваемом примере их три. Область проектов слева (Project), в которой отображается форма базы данных «Общая форма». Вторая область – Свойства (Properties). Третья область – Коды (Code), в которой создаётся программа (эта область содержит созданную программу для обработки событий при работе с формой и элементами управления на ней).

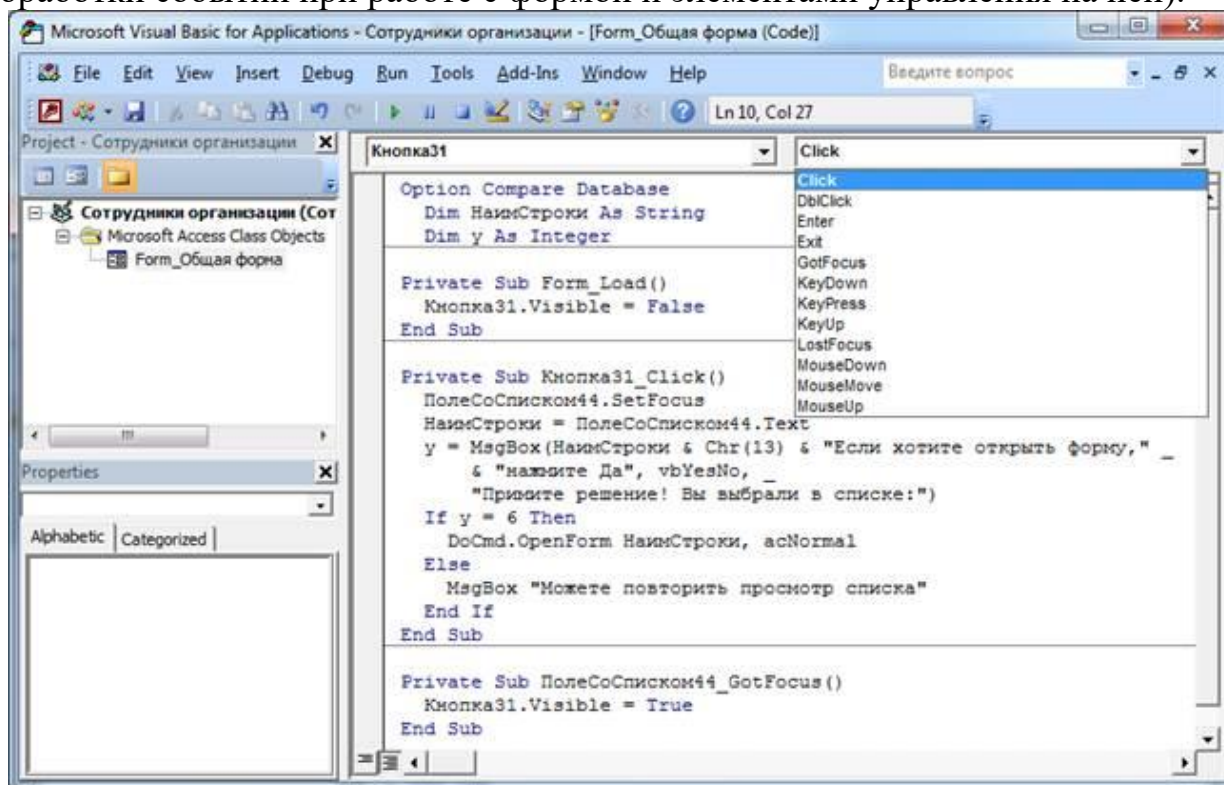


Рис. 23. Программные коды описания переменных и загрузки формы

Напомним, что Visual Basic относится к событийно-процедурным языкам программирования. Их особенностью является то, что логика программы основывается на выделении процедур, которые обрабатывают события, вызываемые либо системой, либо пользователем. При составлении программы сначала вводят общее (General) объявление переменных (Declaration) для формы, а затем создают отдельные процедуры. В рассматриваемом примере понадобится использовать две переменные, первая переменная - «НаимСтроки» предназначена для временного сохранения текста из строки, которую выбрали в раскрывающемся списке элемента на форме «Поле со списком». Вторая переменная понадобится для анализа нажатия на кнопку в диалоговом окне (о нём будет сказано ниже), эту переменную в программе обозначим буквой у, присвоим ей тип данных – целый (Integer), как показано на рисунке 24. Переменные вступят в действие после события – открыть форму «Общая форма».

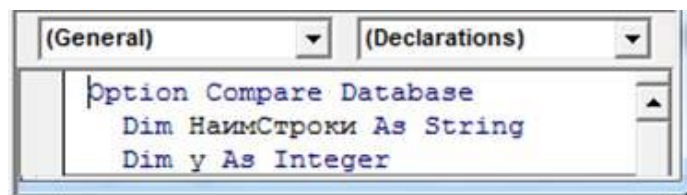
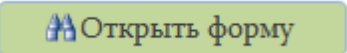


Рис. 24. Программа с общим описанием переменных для программы

Одновременно с загрузкой формы, мы решили скрыть на форме

кнопку , это делается для того, чтобы не возникла ошибка при первом нажатии на кнопку пользователя, когда в строке списка ничего нет. Процедура, которая отвечает за процесс, скрытия кнопки на форме (Рис. 25), состоит всего из одного оператора: Кнопка31.Visible = False кнопке, которой система присвоила номер 31, устанавливаем в свойстве Visible (Видимый) параметр – False (скрыть).

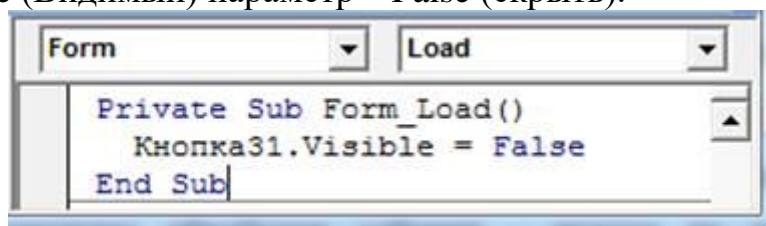


Рис. 25. Процедура открытия формы


В программе есть ещё одна процедура (Рис. 26), которая необходима для отображения кнопки на форме в тот момент, когда поле со списком получает фокус, т.е. это поле активно.

```

Private Sub ПолеСоСписком44_GotFocus()
    Кнопка31.Visible = True
End Sub
  
```

Рис. 26. Процедура отображения кнопки на форме, в случае наведения мышки на поле списка (получение фокуса)



Посмотрите программу для обработки события – нажать на кнопку  Открыть форму, которая находится на форме целиком, для того, что бы легче было ориентироваться при её создании и правки (Рис. 27).

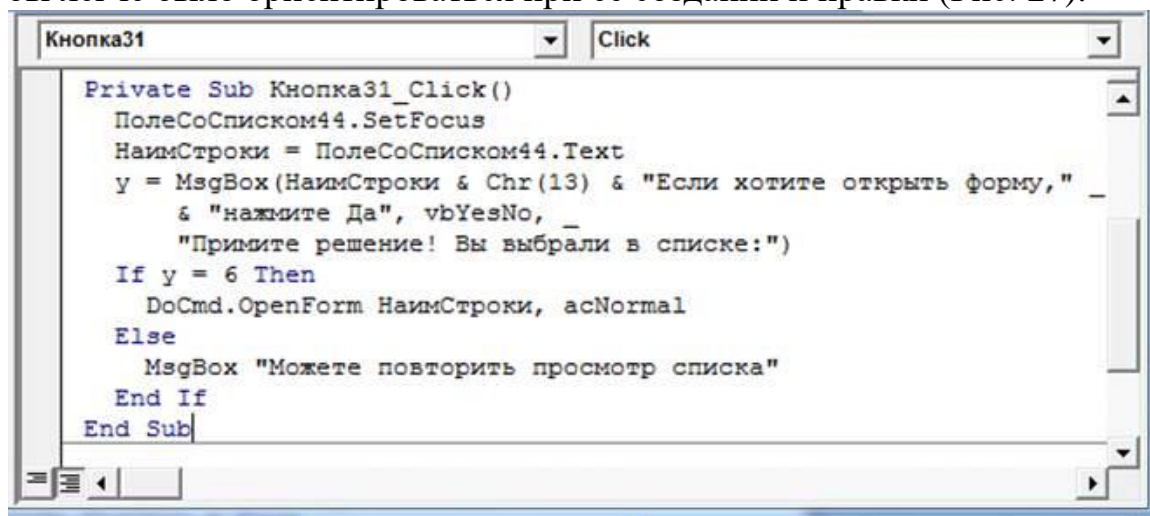


Рис. 27. Процедура обработки события – нажать на кнопку

Процедура Private Sub Кнопка31\_Click() отвечает за перехват выделенного наименования формы в строке списка, подготовки диалогового окна для пользователя о принятии решения по выполнению команды загрузки формы на экран компьютера. Поэтому, первым оператором в процедуре стоит ПолеСоСписком44.SetFocus – установить фокус в строке поля со списком. Обратите внимание, что в программе наименования объектов и переменных пишут слитно (символ пробел недопустим). Вторая строка программы обозначает, что переменной НаимСтроки присваивается значение, которое находится в активной строке ПолеСоСписком44, после точки идёт указание свойства этого объекта – Text (то, что находится в виде записи в строке списка, преобразуется в текстовую переменную). Переменной у присваивается значение из функции MsgBox() - вывод на экран сообщения. На рисунке 27 эта функция занимает три строки (можно всё записать в одной строке), это сделано, что бы было удобнее читать содержимое функции, которое находится в круглых скобках. Признаком переноса строки в программном коде является сочетание двух символов – *Пробел* и *знак подчёркивания*. С помощью функции MsgBox() формируется диалоговое окно, в котором на основном поле появляется надпись: текст, который находится в переменной НаимСтроки, переход на новую строку (функция Chr(13)), продолжение текста – «Если хотите открыть форму, нажмите Да»; далее идёт стандартное описание кнопок VbYesNo, которые появятся на диалоговом окне («Да» и «Нет»); сообщение для информационной строки «Примите решение! Вы выбрали в списке:». При нажатии на кнопку «Да», функция сгенерирует целое число 6, которое будет присвоено переменной у. Далее, используется конструкция условного оператора If (запись условия) Then. После ключевого слова Then вставляют операторы, которые будут задействованы, если условие выполняется. Ключевое слово Else необходимо, чтобы после него вставить операторы, которые будут задействованы, если условие не будет выполнено.

Заканчивается условный оператор командой - End If . В данном примере, если условие будет выполнено (нажата кнопка «Да»), то выполнится открытие объекта базы данных форма с наименованием, которое содержится в переменной НаимСтроки. В противном случае, на экране пользователя появится окно с сообщением «Можете повторить просмотр списка»

В результате разработки главной формы («Общая форма»), пользователи получают возможность использовать все виды таблиц, запросов и форм, к которым можно обратиться с главного интерфейса базы данных, а так же с интерфейсов, которые предназначены для отдельных подразделений организации. На рисунке 28 представлен интерфейс пользователя - «Общая форма» с сообщением в диалоговом окне о выборе другой формы.

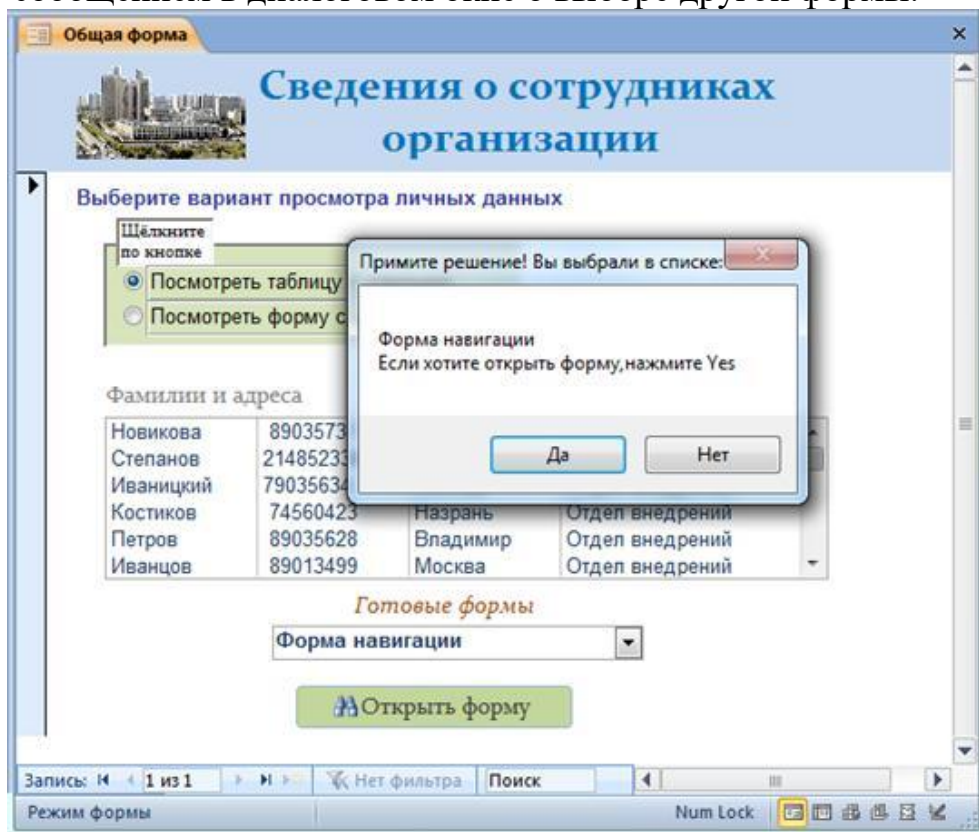


Рис. 28. Внешний вид общей формы в базе данных

## ПРАКТИЧЕСКАЯ РАБОТА № 23

**Тема: Создание меню различных видов. Модификация и управление меню.**

### Макросы и модули

**Цель работы:** отработать навыки создания макросов и модулей.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

### Справочный материал:

**Макросы.** Макрос представляет набор макрокоманд, который создается для автоматизации часто выполняемых задач. Макросы нужны пользователям для начального освоения СУБД, чтобы впоследствии перейти к программированию на VBA.

Макросы в Access создаются в специально предназначенном для этого окне Конструктора макросов. Для создания макроса необходимо выбрать вкладку Создать и в раскрывающейся кнопке Макрос выбрать команду Макрос. Откроется окно конструктора макросов, которое состоит из панели описаний, расположенной в верхней части окна, и панели аргументов – в нижней. Панель описаний по умолчанию содержит три столбца Макрокоманда, Аргументы и Примечание.

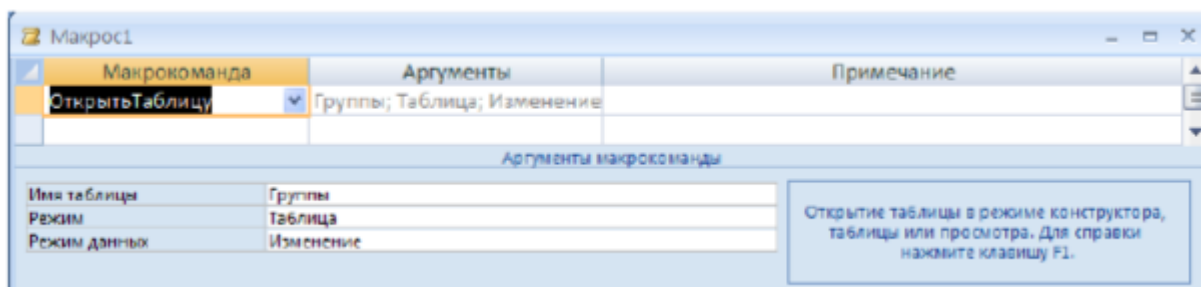
В поле Макрокоманда можно выбрать необходимую команду (ОткрытьТаблицу, ОткрытьЗапрос, ОткрытьОтчет, ЗапускМакроса и т.д.) нажатием кнопкой мыши. Иначе в строку Макрокоманда можно просто перетащить любой объект базы данных (таблицу, форму и т.д.).

После выбора макрокоманды на панели аргументов окна конструктора макросов могут появиться строки, предназначенные для задания значений аргументов соответствующей макрокоманды. Набор строк на этой панели зависит от конкретной макрокоманды. Заданные значения аргументов также появляются во втором столбце Аргументы панели описаний.

Столбец Примечание служит для ввода комментария, который описывает выполняемое действие.

Рассмотрим простой способ создания макрокоманды. Например, разработаем ее для открытия таблицы Группы базы данных Студент.

1. Вы уже открыли окно Конструктора макросов. Теперь с помощью мыши перетащите таблицу Группы из Области переходов в столбец Макрокоманда. В результате в нем появится макрокоманда Открыть таблицу, причем Access автоматически заполнит поля панели аргументов:



2. В поле Режим данных задайте значение Только чтение, что позволит сделать записи этой таблицы недоступными для редактирования. Для того чтобы ввести значение аргумента макрокоманды вы можете выбрать аргумент из

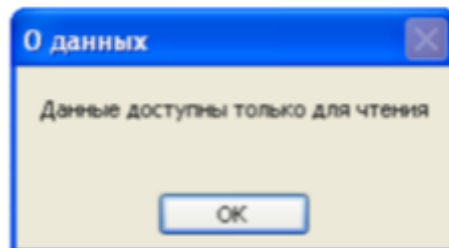
списка или ввести в его поле выражение. Справа от полей некоторых аргументов расположена кнопка построителя, в окне которого вводятся различные выражения. Перед выражением нужно ставить знак равенства (=), за исключением аргумента Выражение макрокоманды ЗадатьЗначение и аргумента Число повторов макрокоманды ЗапускМакроса.

Макрос может содержать несколько макрокоманд, которые выполняются последовательно. Добавим к созданной нами макрокоманде, которая открывает таблицу Группы в режиме Только чтение, добавим макрокоманду, сообщающую пользователю о том, что таблица Товары доступна только для просмотра информации.

1. Перейдите на следующую строку окна конструктора макросов и в столбце Макрокоманда выберите макрокоманду Сообщение.

2. В поле Сообщение панели аргументов введите текст «Данные доступны только для просмотра». Аргумент Сигнал служит для определения того, будет ли вывод сообщения сопровождаться звуковым сигналом. В аргументе Тип выберите вид диалогового окна. Укажите, например, значение Предупреждающее!. В поле аргумент Заголовок, позволяющего задать текст заголовка окна сообщения, введите текст «О данных».

3. Сохраните макрос под именем Открыть группы. После запуска макроса будет отображена таблица Группы, открытая в режиме просмотра, и сообщение, представленное на рисунке:



### **Содержание работы:**

**Задание 1.** Создать меню для разработанной базы данных «Студенты»



## **ПРАКТИЧЕСКАЯ РАБОТА № 24**

### **Тема: Обращение к объектам БД с помощью встроенного языка программирования VBA**

**Цель работы:** Приобрести навыки обращения к элементам базы данных (таблицам, запросам) с помощью языка программирования Visual Basic.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### **Справочный материал:**

Чтобы создать приложение, которое показывает, редактирует и обновляет данные из различных баз данных, включая Microsoft Access, dBASE. Его можно применить, чтобы получить доступ к Microsoft Excel, Lotus 1-2-3 или стандартным текстовым файлам ASCII так, как если бы они были настоящими базами данных. В дополнение к тому, элемент управления данными позволяет получить доступ к удаленным базам данных ODBC и управлять ими (например, базами данных Microsoft SQL Server или Oracle).

Элемент управления данными осуществляет доступ к данным, используя тот же процессор баз данных Microsoft Jet, что работает в Microsoft Access. Эта технология обеспечивает прямой доступ ко многим стандартным форматам баз данных, и позволяет писать приложения обработки данных без программирования.

Элемент управления данными может без программирования решать следующие задачи:

- связаться с локальной или удаленной базой данных;
- открыть указанную таблицу базы данных;
- передавать поля данных в связанные элементы управления, где их можно выводить на экран или менять в них значения;
- добавить новые записи или обновить базу данных
- перехватывать ошибки, возникающие при обращении к данным;
- закрыть базу данных.

Чтобы создать простое приложение базы данных необходимо:

1. Добавить элемент управления данными на форму.
2. Установить его свойства (указать базу данных и таблицу, из которых нужно получать данные).
3. Добавить связанные элементы управления (например, окна с текстом, окна списков, и другие элементы управления, которые необходимо связать с элементом управления данными.)
4. Указать в свойствах связанных элементов управления источник данных и поля данных, которые предполагается выводить.

Элемент управления данными предлагает высокий уровень функциональности, которым можно воспользоваться, не написав ни строчки программы – просто устанавливая его свойства и управляя ими, и включая в проект связанные с данными элементы управления. Однако для расширения функций элемента управления данными в программе на Visual Basic можно самостоятельно манипулировать элементом управления данными и создаваемым им объектом Recordset.

Когда приложение запущено, элемент управления данными работает вместе с базой данных, предоставляя доступ к текущей совокупности записей, или набору записей (Recordset). То есть, элемент управления данными создает объект Recordset. Как и у любого объекта, у этого объекта есть свойства, которые его характеризуют, и методы (действия, которые объект может совершать). Навигация предполагает передвижение или изменение текущей записи в наборе записей. Навигация осуществляется с помощью методов.

Таблица 1. Методы объекта Recordset

Метод	Описание	Пример
AddNew	Добавление новой пустой записи к таблице базы данных	Data1.Recordset.AddNew
MoveNext	Перемещение указателя записи на следующую запись в объекте Recordset	Data1.Recordset.MoveNext
MoveFirst	Перемещение указателя записи на первую запись в объекте Recordset	Data1.Recordset.MoveFirst
MoveLast	Перемещение указателя записи на последнюю запись в объекте Recordset	Data1.Recordset.MoveLast
Update	Сохранение данных в новой записи. Добавление новой записи к таблице.	Data1.Recordset.Update
Delete	Удаление текущей записи	Data1.Recordset.Delete
Close	Закрытие набора записей	Data1.Recordset.Close
Edit	Редактирование текущей записи	Data1.Recordset.Edit

Обращение к данным, находящимся в полях, осуществляется с помощью свойства Fields.

Таблица 2. Свойства объекта Recordset

Свойство	Описание	Пример
Fields	Значение поля. К полю можно обращаться по имени или по номеру. Поля нумеруются, начиная с нуля.	Data3.Recordset.Fields("Направление").Value = направление <b>ИЛИ</b> ФИО = Data2.Recordset.Fields(2).Value
Eof	Признак окончания таблицы. Если достигнута последняя запись в таблице, то свойство принимает значение True, иначе – False	If Data2.Recordset.EOF = True then Form1.print "Вся таблица просмотрена" End IF

Синтаксис, применяемый для управления объектами DAO Database, аналогичен тому, который используется для управления другими объектами Visual Basic:



Таким образом, если требуется написать программу перехода к последней записи в наборе записей, можно обратиться к набору записей, как к объекту, а затем применить к нему метод `MoveLast`: `Data1.Recordset.MoveLast`

Если нужно считать значение определенного поля в текущей записи, нужно написать `MyString = Data1.Recordset.Fields("Title").Value`

Пример 1. Вывести все записи из таблицы БД в текстовое окно

```
Data22.Recordset.MoveFirst
Do While Data22.Recordset.EOF = false
Номер_маршрута = Data22.Recordset.Fields("Номер").Value
Схема = Data22.Recordset.Fields("Схема").Value
Text1.Text=Text1.Text &Номер_маршрута&Схема
Data22.Recordset.MoveNext
Loop
Data22.Recordset.Close
```

Пример 2. Добавить новую запись к таблице

```
Data3.Recordset.AddNew
Data3.Recordset.Fields("ID ОП").Value = ID_ОП
Data3.Recordset.Fields("ID соседнего ОП").Value = ID_ОП_соседнего
Data3.Recordset.Fields("ID маршрута").Value = ID_маршрута
Data3.Recordset.Fields("Направление").Value = направление
Data3.Recordset.Update
```

Пример 3. Найти автора с минимальной премией

```
Data1.Recordset.MoveFirst
Min=100000000
Do While Data1.Recordset.EOF = false
Автор = Data1.Recordset.Fields("ФИО автора").Value
Премия = Data1.Recordset.Fields("премия").Value
If Премия<min then min=Премия
Найден_автор=Автор
End If
Data1.Recordset.MoveNext
```

Loop

Text1.Text="Найден автор" & Найден\_автор & "Премия=" & Премия

### **Содержание работы:**

**Задание 1.** Разместить на форме командные кнопки. Добавить элементы для вывода данных (например, надписи). Написать программу на языке VBA.

1. Открыть базу данных Библиотека.
2. Вывести на экран список всех сотрудников.
3. Вывести на экран всю информацию о сотруднике, фамилию которого вводит пользователь.

Примечание: Можно использовать функцию Inputbox.

Синтаксис: ФИО= Inputbox ("Введи фамилию для поиска")

4. Добавить новую запись к таблице «Сотрудник».

Примечание: Перед выполнением следующих заданий необходимо выполнить резервное копирование БД!

5. Удалить сотрудника, ФИО которого введены с клавиатуры.
6. Изменить регистр данных, находящихся в таблице «Сотруднике» на верхний.

Примечание: Можно использовать функции UCase (см. справку Access)

7. Найти количество записей в таблице «Штатное расписание».
8. Найти количество женатых сотрудников.

Примечание: Можно использовать функции DatePart (см. справку Access)

9. Для всех женатых сотрудников добавить к характеристике сотрудника слова «Женат (или замужем)».
10. Найти суммарное количество всех сотрудников.

## **ПРАКТИЧЕСКАЯ РАБОТА № 25**

**Тема: Обращение к объектам БД с помощью встроенного языка программирования VBA**

**Цель работы:** Приобрести навыки обращения к элементам базы данных (таблицам, запросам) с помощью языка программирования Visual Basic.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** Разместить на форме к базе данных «Студенты» командные кнопки. Добавить элементы для вывода данных (например, надписи). Написать программу на языке VBA.

**Задание 2.** Разместить на форме к базе данных «Классный руководитель» командные кнопки. Добавить элементы для вывода данных (например, надписи). Написать программу на языке VBA.

## ПРАКТИЧЕСКАЯ РАБОТА № 26

### Тема: Создание интерфейса входной формы

**Цель работы:** овладение навыками создания интерфейса с пользователем.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### Содержание работы:

**Задание 1.** Создать для каждой таблицы базы данных Библиотека формы, содержащие кнопки, поля ввода, поля со списком. Добавить изображение, кнопки закрытия формы и выхода из приложения.

Например, для таблицы Сотрудники



**Задание 2.** Создать для каждой таблицы, созданных баз данных «Студенты» и «Классный руководитель», формы, содержащие кнопки, поля ввода, поля со списком. Добавить изображение, кнопки закрытия формы и выхода из приложения.

## ПРАКТИЧЕСКАЯ РАБОТА № 27

**Тема:** Создание файла проекта базы данных. Использование исполняемого файла проекта БД, приемы создания и управления

**Цель работы:** научить применять операторы языка SQL для создания БД

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** Создать базу данных с именем prim, используя язык SQL

CREATE DATABASE prim

Создаем таблицу «Комнаты» с именем room, состоящую из двух полей:

- номер комнаты roomnum, символьное поле типа CHAR длиной 14 символов;

- номер телефона tel, тип поля INTEGER длиной 4 знака.

Первичным ключом таблицы является поле roomnum.

CREATE TABLE room ;

(roomnum C(14) PRIMARY KEY, ;

Tel

I(4) )

Создаем таблицу «Сотрудники» с именем person, состоящую из шести полей:

- табельный номер tubnum, символьное поле типа CHAR длиной 4 символа;

- ФИО fio, символьное поле типа CHAR длиной 15 символов с ограничением на обязательное заполнение данными;

- должность jt, символьное поле типа CHAR длиной 15 символов;

- дата рождения birthd, поле типа DATE (дата: год — месяц — день);

- город city, символьное поле типа CHAR длиной 15 символов;

- номер комнаты roomnum, символьное поле типа CHAR длиной 14 символов.

Первичным ключом таблицы является поле tubnum. Вторичный ключ roomnum с именем perroom связывает данную таблицу с таблицей room с заданным первичным ключом по аналогичному полю roomnum:

FOREIGN KEY roomnum TAG perroom REFERENCES room

CREATE TABLE person;

( tubnum C(4) PRIMARY KEY,;

fio C(25) NOT NULL,;

jt C(15),;

birthd D,;

city C(15),;

roomnum C(4),;

FOREIGN KEY roomnum TAG perroom REFERENCES room)

Создаем таблицу «Дети» с именем baby, состоящую из трех полей:

- табельный номер tubnum, символьное поле типа CHAR длиной 4 символа;

- имя ребенка bname, символьное поле типа CHAR длиной 10 символов;

– возраст age, числовое поле типа NUMERIC с длиной действительной части 4 символа;

Первичным ключом таблицы является конкатенация полей tubnum и bname, ключ имеет имя baby. Вторичный ключ tubnum с именем btub связывает данную таблицу с таблицей person с заданным первичным ключом по аналогичному полю tubnum.

```
CREATE TABLE baby;  
( tubnum C(4),;  
  bname C(10),;  
  age N(4,0),;  
  PRIMARY KEY tubnum+bname TAG baby,;  
  FOREIGN KEY tubnum TAG btub REFERENCES person)
```

Создаем представление с именем mbaby, содержащее поле fio из таблицы person и поле bname из таблицы baby. Представление образует временную таблицу, содержащую список имен детей с ФИО сотрудника, являющегося родителем данного ребенка:

```
CREATE VIEW mbaby;  
AS SELECT person.fio, baby.bname;  
FROM person, baby;  
WHERE person.tubnum= baby.tubnum
```

**Задание 2.** Создать базу данных, используя язык SQL.

Создать базу данных, состоящую из следующих трех таблиц с перечисленными полями.

Таблица 1 Книги: Регистрационный номер; Название книги; Название издательства; Год издания; Количество страниц.

Первичным ключом таблицы является поле «Регистрационный номер». Внешним ключом является поле «Название издательства»

Таблица 2 Авторы: Регистрационный номер; ФИО; Год рождения; Город проживания.

Первичным ключом таблицы является конкатенация полей «Регистрационный номер» и «ФИО». Внешним ключом таблицы является поле «Регистрационный номер».

Таблица 1 связана с Таблицей 2 через ключи, образованные полями «Регистрационный номер».

Таблица 3 Издательства: Название издательства; Владелец издательства; Адрес издательства; Город издательства.

Первичным ключом таблицы является поле «Название издательства».

Таблица 3 связана с Таблицей 1 через ключи, образованные полями «Название издательства». В БД должно быть представление, построенное на основе запроса, содержащего «Название книги», «Название издательства», «Владелец издательства».



## ПРАКТИЧЕСКАЯ РАБОТА № 28

**Тема:** Создание файла проекта базы данных. Использование исполняемого файла проекта БД, приемы создания и управления

**Цель работы:** научить применять операторы языка SQL для создания БД

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** Создать базу данных, используя язык SQL.

Создать БД, состоящую из следующих трех таблиц с перечисленными полями.

Таблица 1 Автомобили: Регистрационный номер, Название марки и модели; Год выпуска; Цвет кузова.

Первичным ключом таблицы является поле «Регистрационный номер». Внешним ключом является поле «Название марки и модели».

Таблица 2 Владельцы: Регистрационный номер; ФИО; Год рождения; Город проживания; Адрес.

Первичным ключом таблицы является конкатенация полей «Регистрационный номер» и «ФИО». Внешним ключом таблицы является поле «Регистрационный номер»

Таблица 1 связана с Таблицей 2 через ключи, образованные полями «Регистрационный номер».

Таблица 3 Марки и модели: Название марки и модели; Мощность двигателя; Тип коробки передач; Тип кузова.

Первичным ключом таблицы является поле «Название марки и модели». Таблица 3 связана с Таблицей 1 через ключи, образованные полями «Название марки и модели». В БД должно быть представление, построенное на основе запроса, содержащего «Регистрационный номер», «ФИО», «Город проживания», «Адрес».

**Задание 2.** Создать базу данных, используя язык SQL

Создать базу данных, состоящую из следующих трех таблиц с перечисленными полями.

Таблица 1 Песни: Название песни; Язык песни; Музыкальный стиль; Время звучания.

Первичным ключом таблицы является поле «Название песни».

Таблица 2 Авторы: Псевдоним; ФИО (должно быть обязательно заполнено); Дата рождения; Город проживания. Адрес.

Первичным ключом таблицы является поле «Псевдоним».

Таблица 3 Исполнители: Псевдоним; ФИО (должно быть обязательно заполнено); Дата рождения; Город проживания; Адрес; Характеристика голоса.

Первичным ключом таблицы является поле «Псевдоним».

Таблица 4 Связи: Название песни; Псевдоним автора; Псевдоним исполнителя;

Первичным ключом таблицы является конкатенация полей «Название песни», «Псевдоним автора», «Псевдоним исполнителя». Вторичными ключами

являются поля «Название песни», «Псевдоним автора», «Псевдоним исполнителя».

Таблица 4 связана с Таблицами 1, 2 и 3 через ключи, образованные полями «Название песни», «Псевдоним автора», «Псевдоним исполнителя» с соответствующими полями Таблиц 1, 2 и 3 В БД должно быть представление, построенное на основе запроса, содержащего «Название песни», «ФИО» (исполнителя).

## ПРАКТИЧЕСКАЯ РАБОТА № 29

### Тема: Модификация содержимого БД. Добавление, удаление и обновление данных

**Цель работы:** научить добавлять данные к созданным таблицам с помощью оператора языка SQL – INSERT INTO; обновлять и удалять данные и таблицы.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### Справочный материал:

##### Пример 1. Добавление целых строк

Оператор **INSERT** используется для вставки (добавления) строк в таблицу базы данных. Добавление можно осуществить несколькими способами:

- добавить одну полную строку
- добавить часть строки
- добавить результаты запроса.

Итак, чтобы добавить новую строку в таблицу, нам необходимо указать название таблицы, перечислить названия колонок и указать значение для каждой колонки с помощью конструкции **INSERT INTO название\_таблицы (поле1, поле2 ... ) VALUES (значение1, значение2 ...)**. Рассмотрим на примере.  
**INSERT INTO Sellers (ID, Address, City, Seller\_name, Country) VALUES ('6', '1st Street', 'Los Angeles', 'Harry Monroe', 'USA')**

ID	Address	City	Seller_name	Country
1	500 Park Street	Montreal	Michelle Green	Canada
2	1000 5th Avenue	San Francisco	Kim Howard	USA
3	42 Galaxy Road	New York	John Smith	USA
4	123 Main Street	Toronto	Denise L. Stephens	Canada
5	4th Avenue	Ottawa	Semuel Piter	Canada
6	1st Street	Los Angeles	Harry Monroe	USA

Также можно изменять порядок указания названий колонок, однако одновременно нужно менять и порядок значений в параметре **VALUES**.

##### Пример 2. Добавление части строк

В предыдущем примере при использовании оператора **INSERT** мы явно отмечали имена столбцов таблицы. Используя данный синтаксис, мы можем пропустить некоторые столбцы. Это значит, что вы вводите значение для одних столбцов но не предлагаете их для других. Например:

**INSERT INTO Sellers (ID, City, Seller\_name) VALUES ('6', 'Los Angeles', 'Harry Monroe')**

ID	Address	City	Seller_name	Country
1	500 Park Street	Montreal	Michelle Green	Canada
2	1000 5th Avenue	San Francisco	Kim Howard	USA
3	42 Galaxy Road	New York	John Smith	USA
4	123 Main Street	Toronto	Denise L. Stephens	Canada
5	4th Avenue	Ottawa	Semuel Piter	Canada
6		Los Angeles	Harry Monroe	

В данном примере мы не указали значение для двух столбцов **Address** и **Country**. Вы можете исключать некоторые столбцы из

оператора **INSERT INTO**, если это позволяет производить определение таблицы. В этом случае должно соблюдаться одно из условий: этот столбец определен как допускающий значение **NULL**(отсутствие какого-либо значения) или в определение таблицы указанное значение по умолчанию. Это означает, что, если не указано никакое значение, будет использовано значение по умолчанию. Если вы пропускаете столбец таблицы, которая не допускает появления в своих строках значений **NULL** и не имеет значения, определенного для использования по умолчанию, СУБД выдаст сообщение об ошибке, и эта строка не будет добавлена.

#### Пример 3. Добавление отобранных данных

В предыдущей примерах мы вставляли данные в таблицы, прописывая их вручную в запросе. Однако оператор **INSERT INTO** позволяет автоматизировать этот процесс, если мы хотим вставлять данные из другой таблицы. Для этого в SQL существует такая конструкция как **INSERT INTO ... SELECT ...**. Данная конструкция позволяет одновременно выбирать данные из одной таблицы, и вставить их в другую. Предположим мы имеем еще одну таблицу **Sellers\_EU** с перечнем продавцов нашего товара в Европе и нам нужно их добавить в общую таблицу **Sellers**. Структура этих таблиц одинакова (то же количество колонок и те же их названия), однако другие данные. Для этого мы можем прописать следующий запрос:

```
INSERT INTO Sellers (ID, Address, City, Seller_name, Country) SELECT ID, Address, City, Seller_name, Country FROM Sellers_EU
```

Нужно обратить внимание, чтобы значение внутренних ключей не повторялись (поле **ID**), в противном случае произойдет ошибка. Оператор **SELECT** также может включать предложения **WHERE** для фильтрации данных. Также следует отметить, что СУБД не обращает внимания на названия колонок, которые содержатся в операторе **SELECT**, для нее важно только порядок их расположения. Поэтому данные в первом указанном столбце, что были выбраны из-за **SELECT**, будут в любом случае заполнены в первый столбец таблицы **Sellers**, указанной после оператора **INSERT INTO**, независимо от названия поля.

#### Пример 4. Копирование данных из одной таблицы в другую

Часто при работе с базами данных возникает необходимость в создании копий любых таблиц, с целью резервирования или модификации. Чтобы сделать полную копию таблицы в SQL предусмотрен отдельный оператор **SELECT INTO**. Например, нам нужно создать копию таблицы **Sellers**, нужно будет прописать запрос следующим образом:

```
SELECT * INTO Sellers_new FROM Sellers
```

Усі об'єкти Access	ID	Address	City	Seller_name	Country
Таблиці	1	500 Park Street	Montreal	Michelle Green	Canada
Customers	2	1000 5th Avenue	San Francisco	Kim Howard	USA
Sellers	3	42 Galaxy Road	New York	John Smith	USA
Sellers_new	4	123 Main Street	Toronto	Denise L. Stephens	Canada
Sumproduct	5	4th Avenue	Ottawa	Semuel Piter	Canada
	6	1st Street	Los Angeles	Harry Monroe	USA

В отличие от предыдущей конструкции **INSERT INTO ... SELECT ...**, когда данные добавляются в существующую таблицу, конструкция **SELECT ... INTO ... FROM ...** копирует данные в новую таблицу. Также можно сказать, что первая конструкция импортирует данные, а вторая - экспортирует. При использовании конструкции **SELECT ... INTO ... FROM ...** следует учитывать следующее:

- можно использовать любые предложения в операторе **SELECT**, такие как **GROUP BY** и **HAVING**
- для добавления данных из нескольких таблиц можно использовать объединение
- данные возможно добавить только одну таблицу, независимо от того, из скольких таблиц они были взяты.

#### Пример 5. Обновление таблиц

Для того, чтобы изменить таблицу в SQL используется оператор **ALTER TABLE**. При использовании данного оператора необходимо ввести следующую информацию:

- имя таблицы, которую мы хотим изменить
- перечень изменений, которые мы хотим сделать.

Для примера давайте добавим новую колонку в таблицу **Sellers**, в которой будем указывать телефон реализатора:

**ALTER TABLE Sellers ADD Phone CHAR (20)**

ID	Address	City	Seller_name	Country	Phone
1	500 Park Street	Montreal	Michelle Green	Canada	
2	1000 5th Avenue	San Francisco	Kim Howard	USA	
3	42 Galaxy Road	New York	John Smith	USA	
4	123 Main Street	Toronto	Denise L. Stephens	Canada	
5	4th Avenue	Ottawa	Semuel Piter	Canada	
6	1st Street	Los Angeles	Harry Monroe	USA	

#### Пример 6. Удаление данных

Кроме добавления столбцов, мы можем их удалять. Давайте теперь удалим поле **Phone**. Для этого пропишем следующий запрос:

**ALTER TABLE Sellers DROP COLUMN Phone**

#### **Содержание работы:**

**Задание 1.** В созданные таблицы из Практической работы №28, добавить по 5 строк.

**Задание 2** В каждой созданной таблице из Практической работы №28 обновить данные.

**Задание 3.** Из каждой созданной таблицы из Практической работы №28 удалить любые данные.

## ПРАКТИЧЕСКАЯ РАБОТА № 30

### Тема: Создание простых запросов к базе данных

**Цель работы:** научить создавать простые запросы к базе данных на языке SQL

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### Содержание работы:

**Задание 1.** Создать базу данных Туры и простые запросы к ней

1.Создайте запрос в режиме конструктора и перейдите в режим SQL. Введите следующую команду для создания таблицы:

```
CREATE TABLE [Туры] (  
[Код] INTEGER NOT PRIMARY KEY,  
[Страна] VARCHAR(50) NOT NULL,  
[Транспорт] VARCHAR(20) NOT NULL,  
[Цена] MONEY NOT )
```

В этой команде требуется создать таблицу (CREATE TABLE) с именем «Туры». В таблице должно быть 4 поля:

Код – целое число (INTEGER), непустое (NOT NULL), первичный ключ таблицы (PRIMARY KEY)

Страна – строка длиной до 50 символов, непустое

Транспорт – строка длиной до 20 символов, непустое

Цена – поле для хранения денежной суммы (MONEY)

Названия таблиц и полей заключаются в квадратные скобки! Если эти названия состоят из одного слова, скобки можно не ставить:

```
CREATE TABLE Туры (  
Код INTEGER NOT PRIMARY KEY,  
Страна VARCHAR(50) NOT NULL,  
Транспорт VARCHAR(20) NOT NULL,  
Цена MONEY NOT NULL)
```

2. Выполните эту команду (вкладка Конструктор – Выполнить). Проверьте, что таблица действительно создана.

3. Выполните команду для добавления в базу новой записи:

```
INSERT INTO Туры  
VALUES (1, 'Финляндия', 'автобус', 1200)
```

Эта команда вставляет (INSERT) в таблицу «Туры» одну запись. После ключевого слова VALUES в скобках перечислены через запятую значения полей в том порядке, в котором они задавались при создании таблицы. Символьные строки в значениях полей заключаются в апострофы или двойные кавычки!

4. Выполните ещё одно добавление записи:

```
INSERT INTO Туры  
VALUES (1, 'Норвегия', 'самолёт', 15000)
```

Какая ошибка произошла? В чём её причина?

Исправьте ошибку и добавьте новую запись правильно. Аналогично добавьте в таблицу ещё несколько записей:

Страна	Транспорт	Цена
Швеция	паром	9000 р.
Германия	автобус	15700 р.
Греция	самолёт	23000 р.
Норвегия	автобус	8000 р.
Германия	самолёт	19000 р.

5. Выполните запрос на выборку данных: `SELECT * FROM Туры`

Посмотрите на результат. Этот оператор выберет все поля (\*) всех записей из таблицы «Туры».

6. Вместо \* можно указать через запятую список нужных полей:

`SELECT Страна, Цена FROM Туры`

Проверьте результат выполнения этого запроса.

7. Чаще всего нужно выбрать только записи, удовлетворяющие некоторому условию отбора. Для этого используется ключевое слово `WHERE`, после которого стоит условие: `SELECT * FROM Туры WHERE Страна = 'Норвегия'`

Проверьте работу этого оператора.

8. Составьте запрос, который выбирает из таблицы «Туры» значения полей «Страна», «Транспорт» и «Цена» для всех автобусных туров.

9. Составьте запрос, который выбирает из таблицы «Туры» значения всех полей для туров с ценой меньше 10000 руб

10. Для того, чтобы отсортировать данные по некоторому полю, в запросе после ключевых слов `ORDER BY` (англ. «упорядочить по») указывают название этого поля: `SELECT * FROM Туры ORDER BY Цена`

Проверьте работу этого запроса.

Если в конце предыдущего запроса добавить слово `DESC` (англ. «descending» – нисходящий), сортировка выполняется в обратном порядке.

11. Составьте запрос, который выбирает из таблицы «Туры» значения всех полей для туров с ценой больше 10000 руб. и сортирует результаты по убыванию цены.

12. В запросах можно использовать стандартные функции. Например, функция `MIN` определяет минимальное значение заданного поля среди всех записей:

`SELECT MIN(Цена) FROM Туры`

Результат этого запроса – одно число.

13. Составьте запрос, который находит минимальную цену для туров в Норвегию:

14. Результаты запросов можно использовать в других запросах – получается вложенный запрос. Например, запрос `SELECT * FROM Туры WHERE Цена = (SELECT MIN(Цена) FROM Туры WHERE Страна = 'Норвегия')` вернет данные о самом дешевом туре.

15. Составьте запрос, который находит тур минимальной цены на самолёте:

Изменение и удаление данных

16. Для изменения записей используется оператор `UPDATE`. Запрос, приведенный ниже, увеличивает цены всех туров на 10%:

`UPDATE Туры SET Цена = Цена*1.1`

Проверьте, что данные в таблице «Туры» действительно изменились.

17. Авиакомпании в данный момент представляют скидку на билеты, так что цены всех туров на самолётах составляют 80% от исходных. Составьте и выполните соответствующий запрос. Какая стоимость получилась у тура в Грецию?

18. Скопируйте таблицу «Туры», назвав копию «Туры2». Удалите все туры в Германию с помощью запроса `DELETE FROM Туры2 WHERE Страна = 'Германия'`

Проверьте, что данные в таблице «Туры2» действительно изменились.

19. Удалите таблицу «Туры2», которая больше не нужна, с помощью запроса `DROP TABLE Туры2`

Проверьте, что таблица «Туры2» удалена из списка таблиц.



## **ПРАКТИЧЕСКАЯ РАБОТА № 31**

**Тема: Создание простых запросов к базе данных**

**Цель работы:** научить создавать простые запросы к базе данных на языке SQL

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** Создать 10 простых запросов (по аналогии запросов из Практической работы № 30) к базам данных, созданным в Практической работе № 28.

## ПРАКТИЧЕСКАЯ РАБОТА № 32

### Тема: Создание сложных запросов к базе данных

**Цель работы:** научить создавать сложные запросы и подзапросы, комбинированные запросы на языке SQL

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### Справочный материал:

##### Пример 1. Фильтрация с помощью подзапросов

Таблицы баз данных, которые используются в СУБД Access, являются реляционными таблицами, т.е. все таблицы можно связать между собой по общим полям. Допустим у нас хранятся данные в двух разных таблицах и нам нужно выбрать данные в одной из них, в зависимости от того, какие данные в другой. Для этого создадим еще одну таблицу в нашей базе данных. Это будет, например, таблица **Sellers** с информацией о поставщиках:

ID	Address	City	Seller_name	Country
1	500 Park Street	Montreal	Michelle Green	Canada
2	1000 5th Avenue	San Francisco	Kim Howard	USA
3	42 Galaxy Road	New York	John Smith	USA
4	123 Main Street	Toronto	Denise L. Stephens	Canada

Теперь мы имеем две таблицы – **Sumproduct** и **Sellers**, которые имеют одинаковое поле **City**. Предположим, нам нужно посчитать, сколько товаров было продано только в Канаде. Сделать это нам помогут подзапросы. Итак, сначала напишем запрос для выборки городов, которые находятся в Канаде:

**SELECT City FROM Sellers WHERE Country = 'Canada'**

City
Montreal
Toronto

Теперь передадим эти данные в следующий запрос, который будет выбирать данные из таблицы **Sumproduct**:

**SELECT SUM(Quantity) AS Qty\_Canada FROM Sumproduct WHERE City IN ('Montreal','Toronto')**

Qty_Canada
10222

Также мы можем объединить эти два запроса в один. Таким образом, один запрос, который выводит данные, будет главным, а второй запрос, который передает входные данные, будет вспомогательным (подзапросом). Для вложения подзапроса используем конструкцию **WHERE ... IN (...)**:

**SELECT SUM(Quantity) AS Qty\_Canada FROM Sumproduct WHERE City IN (SELECT City FROM Sellers WHERE Country = 'Canada')**

Qty_Canada
10222

Видим, что мы получили аналогичные данные, как и с помощью двух отдельных запросов. Таким же образом, мы можем увеличивать глубину вложенности запросов, вкладывая подзапросы сколько угодно раз.

### Пример 2. Использование подзапросов в качестве расчетных полей

Мы также можем использовать подзапросы в качестве расчетных полей. Отразим, например, количество реализованной продукции по каждому продавцу с помощью следующего запроса:

**SELECT Seller\_name, (SELECT SUM(Quantity) FROM Sumproduct WHERE Sellers.City = Sumproduct.City) AS Qty**FROM Sellers

Seller_name	Qty
Michelle Green	5129
Kim Howard	5347
John Smith	3897
Denise L. Stephens	5093

Первый оператор **SELECT** отражает два столбца - **Seller\_name** и **Qty**. Поле **Qty** является расчетным, оно формируется в результате выполнения подзапроса, который взят в круглые скобки. Этот подзапрос выполняется по одному разу для каждой записи в поле **Seller\_name** и в общем будет выполнен четыре раза, поскольку выбрано имена четырех продавцов.

Также в подзапросе, предложение **WHERE** выполняет функцию объединения, поскольку с помощью **WHERE** мы соединили две таблицы по полю **City**, используя полные названия столбцов (*Таблица.Поле*).

#### **Содержание работы:**

**Задание 1.** Создать по 5 запросов к созданным таблицам из Практической работы № 27

## **ПРАКТИЧЕСКАЯ РАБОТА № 33**

**Тема:** Создание сложных запросов к базе данных

**Цель работы:** научить создавать сложные запросы и подзапросы, комбинированные запросы на языке SQL

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** Создать по 5 запросов к созданным таблицам из Практической работы № 28

## ПРАКТИЧЕСКАЯ РАБОТА № 34

**Тема:** Символы подстановки и регулярные выражения (LIKE)

**Цель работы:** научить применять оператор Like для поиска и отбора данных по шаблону.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

**Справочный материал:**

Часто, для фильтрации данных, нужно осуществлять выборку не по точному совпадению условия, а по приближенному значению. То есть когда, например, мы ищем товар, название которого соответствует определенному шаблону или содержит определенные символы или слова. Для таких целей в **SQL** существует оператор **LIKE**, который ищет приближенные значения. Для конструирования такого шаблона используются **метасимволы** (специальные символы для поиска части значения), а именно: "знак процента" (%) или "звездочка" (\*), "символ подчеркивания" (\_) или "знак вопроса" (?), "квадратные скобки" ([ ]).

Пример 1. Метасимвол знак процента (%) или звездочка (\*)

Из нашей таблицы, например, отберем записи, относящиеся только к товарам, содержащих в своем названии слово *Skis* (лыжи). Для этого составим соответствующий шаблон:

**SELECT \* FROM Sumproduct WHERE Product LIKE '\*Skis\*'**

ID	Month	Product	City	Quantity	Amount
10	April	Skis Long	Montreal	854	\$209 230,00
11	April	Skis Long	Toronto	25	\$6 125,00
12	April	Skis Long	San Francisco	663	\$162 435,00
13	April	Skis Long	New York	21	\$5 145,00
14	April	Skis Short	Montreal	21	\$4 389,00
15	April	Skis Short	Toronto	4	\$836,00
16	April	Skis Short	San Francisco	522	\$109 098,00
17	April	Skis Short	New York	136	\$28 424,00
30	February	Skis Long	Montreal	522	\$127 890,00
31	February	Skis Long	Toronto	125	\$30 625,00
32	February	Skis Long	San Francisco	663	\$162 435,00
33	February	Skis Long	New York	21	\$5 145,00

Как видим, **СУБД** отобрала только те записи, где в колонке **Product** были товары, содержащие слово *Skis*. Также отметим, что в данном примере используется метасимвол "звездочка" (\*), поскольку **СУБД Access** не поддерживает "знак процента" (%) для оператора **LIKE**.

Пример 2. Метасимвол знак подчеркивания (\_) или знак (?)

Знак подчеркивания или вопросительный знак применяется для того, чтобы заменить один символ в слове. Давайте в слове *Bikes* заменим все гласные буквы на "вопросительный знак" (?) и посмотрим на результат:

**SELECT \* FROM Sumproduct WHERE Product LIKE 'B?k?s'**

ID	Month	Product	City	Quantity	Amount
2	April	Bikes	Montreal	12	\$4 500,00
3	April	Bikes	Montreal	56	\$21 000,00
4	April	Bikes	San Francisco	854	\$320 250,00
5	April	Bikes	New York	25	\$9 375,00
22	February	Bikes	Montreal	663	\$248 625,00
23	February	Bikes	San Francisco	21	\$7 875,00
24	February	Bikes	San Francisco	54	\$20 250,00
25	February	Bikes	New York	658	\$246 750,00
42	January	Bikes	Montreal	75	\$28 125,00

Мы использовали метасимвол "вопросительный знак" (?), поскольку СУБД Access не поддерживает "знак подчеркивания" (\_) для оператора LIKE.

Пример 3. Метасимвол квадратные скобки ([ ])

Метасимвол "квадратные скобки" ([ ]) используется для одновременного указания набора символов, по которым нужно выполнить поиск.

**SELECT \* FROM Sumproduct WHERE City LIKE '[TN]\*'**

ID	Month	Product	City	Quantity	Amount
5	April	Bikes	New York	25	\$9 375,00
7	April	Skates	Toronto	854	\$84 546,00
9	April	Skates	New York	663	\$65 637,00
11	April	Skis Long	Toronto	25	\$6 125,00
13	April	Skis Long	New York	21	\$5 145,00
15	April	Skis Short	Toronto	4	\$836,00
17	April	Skis Short	New York	136	\$28 424,00
19	April	Snow Board	Toronto	522	\$160 776,00
21	April	Snow Board	New York	663	\$204 204,00

В примере выше, мы отобрали записи, где в поле **City** названия городов начинаются с буквы **T** или **N**. Также, в данном случае, мы можем использовать еще один метасимвол, который выполняет обратное действие. Добавим в наше регулярное выражение восклицательный знак (!), что будет означать "не равно" (для СУБД Access) или знак степени (^) (для других СУБД).

**SELECT \* FROM Sumproduct WHERE City LIKE '[!TN]\*'**

ID	Month	Product	City	Quantity	Amount
2	April	Bikes	Montreal	12	\$4 500,00
3	April	Bikes	Montreal	56	\$21 000,00
4	April	Bikes	San Francisco	854	\$320 250,00
6	April	Skates	Montreal	56	\$5 544,00
8	April	Skates	San Francisco	25	\$2 475,00
10	April	Skis Long	Montreal	854	\$209 230,00
12	April	Skis Long	San Francisco	663	\$162 435,00
14	April	Skis Short	Montreal	21	\$4 389,00
16	April	Skis Short	San Francisco	522	\$109 098,00

То есть, последний созданный нами запрос будет читаться как: выбрать все колонки из таблицы **Sumproduct** и только те записи, где в поле **City** названия городов не начинаются на буквы **T** или **N**. Дополнительно

отметим, что набор букв в метасимволе "квадратные скобки" отвечает только за одну позицию в тексте.

Мы можем получить аналогичный результат, если воспользоваться уже известным нам оператором **NOT**, однако с восклицательным знаком (!) запись будет короче.

### **Содержание работы:**

**Задание 1.** Применить метасимволы ("знак процента" (%), "знак вопроса" (?), "квадратные скобки" ([ ])) к созданным таблицам по выбранным вариантам заданий из Практической работы № 28.

## ПРАКТИЧЕСКАЯ РАБОТА № 35

### Тема: Вычисляемые поля

**Цель работы:** научить применять операторы языка SQL для создания вычисляемых полей.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### Справочный материал:

##### Пример 1. Выполнение математических операций

Одним из способов использования расчетных полей является выполнение математических операций над выбранными данными. Давайте на примере рассмотрим как это происходит, используя снова нашу таблицу **Sumproduct**. Предположим, нам нужно вычислить среднюю цену приобретения каждого товара. Для этого нужно переделить колонку **Amount** (сумма) на **Quantity** (количество):

**SELECT DISTINCT Product, Amount/Quantity FROM Sumproduct**

Product	Expr1001
Bikes	375
Skates	99
Skis Long	245
Skis Short	209
Snow Board	308

Как видим, СУБД отобразила все наименования товаров и отобразила их среднюю стоимость в отдельном столбце, который был создан во время выполнения запроса. Также можно заметить, что мы использовали дополнительный оператор **DISTINCT**, который нам нужен для отображения уникальных названий товаров (без него мы бы получили дублирование записей).

##### Пример 2. Использование псевдонимов

В предыдущем примере мы рассчитывали среднюю стоимость покупки каждого товара и отображали значение в расчетном столбце. Однако в дальнейшем, нам неудобно обращаться к этому полю, так как его название является неинформативным для нас (СУБД дала название полю - **Expr1001**). Однако мы можем назвать поле самостоятельно, заранее указав его название в запросе, то есть дать псевдоним. Давайте перепишем предыдущий пример и укажем псевдонима для расчетного поля:

**SELECT DISTINCT Product, Amount/Quantity AS AvgPrice FROM Sumproduct**

Product	AvgPrice
Bikes	375
Skates	99
Skis Long	245
Skis Short	209
Snow Board	308

Видим, наше расчетное поле получило собственное название **AvgPrice**. Для этого мы использовали оператор **AS**, после которого указали необходимое



нам название. Стоит отметить, что в SQL поддерживаются только основные математические операции: сложение (+), вычитание (-), умножение (\*), деление (/). Также для изменения очередности выполнения операции можно использовать круглые скобки.

Часто псевдонимы используют не только чтобы называть расчетные поля, но и для переименования действующих. Это может быть необходимым, если действующее поле имеет длинное название или название не достаточно информативным.

### Пример 3. Соединение полей (конкатенация)

Кроме математических операций мы можем объединять текст и выводить его в отдельном поле. Давайте рассмотрим, каким образом можно осуществить склеивание (конкатенацию) текста. Имеем такой пример:

**SELECT Month + ' ' + Product AS NewField, Quantity FROM Sumproduct**

NewField	Quantity
April Bikes	12
April Bikes	56
April Bikes	854
April Bikes	25
April Skates	56
April Skates	854
April Skates	25
April Skates	663
April Skis Long	854
April Skis Long	25
April Skis Long	663
April Skis Long	21

В этом примере мы соединили значение в двух столбцах и вывели результат в новое поле **NewField**.

### **Содержание работы:**

**Задание 1.** В созданных таблицах из Практической работы № 27 применить расчетные поля (если в таблицах нет числовых значений, то добавьте в них такие поля)

## **ПРАКТИЧЕСКАЯ РАБОТА № 36**

### **Тема: Вычисляемые поля**

**Цель работы:** научить применять операторы языка SQL для создания вычисляемых полей.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### **Содержание работы:**

**Задание 1.** В созданных таблицах из Практической работы № 28 применить расчетные поля (если в таблицах нет числовых значений, то добавьте в них такие поля)

## ПРАКТИЧЕСКАЯ РАБОТА № 37

### Тема: Проведение сортировки и фильтрации данных. Поиск данных по одному и нескольким полям. Поиск данных в таблице. Группировка данных (GROUP BY)

**Цель работы:** изучить операторы языка SQL для сортировки, фильтрации и группировки данных.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

**Справочный материал:**

Пример 1. Сортировка выбранных данных.

Давайте всю нашу таблицу посортируем по сумме реализации продукции, а именно по столбцу **Amount**.

**SELECT \* FROM Sumproduct ORDER BY Amount**

ID	Month	Product	City	Quantity	Amount
47	January	Skates	New York	4	\$396,00
15	April	Skis Short	Toronto	4	\$836,00
35	February	Skis Short	Toronto	4	\$836,00
74	March	Skis Short	Montreal	4	\$836,00
52	January	Skis Long	San Francisco	4	\$980,00
41	February	Snow Board	New York	4	\$1 232,00
18	April	Snow Board	Montreal	4	\$1 232,00
79	March	Snow Board	Toronto	4	\$1 232,00
62	March	Bikes	Montreal	4	\$1 500,00
28	February	Skates	San Francisco	21	\$2 079,00
26	February	Skates	Montreal	21	\$2 079,00
46	January	Skates	Montreal	21	\$2 079,00
8	April	Skates	San Francisco	25	\$2 475,00

Видим, что запрос посортировал записи по возрастанию в поле **Amount**. Обязательно нужно соблюдать последовательность расположения операторов, т.е. оператор **ORDER BY** должен идти в самом конце запроса. В противном случае будет получено сообщение об ошибке.

Пример 2. Сортировка по нескольким полям.

Теперь посортируем наш пример дополнительно за еще одним полем. Пусть это будет поле **City**, которое отображает место реализации продукции.

**SELECT \* FROM Sumproduct ORDER BY Amount, City**

ID	Month	Product	City	Quantity	Amount
47	January	Skates	New York	4	\$396,00
74	March	Skis Short	Montreal	4	\$836,00
35	February	Skis Short	Toronto	4	\$836,00
15	April	Skis Short	Toronto	4	\$836,00
52	January	Skis Long	San Francisco	4	\$980,00
18	April	Snow Board	Montreal	4	\$1 232,00
41	February	Snow Board	New York	4	\$1 232,00
79	March	Snow Board	Toronto	4	\$1 232,00
62	March	Bikes	Montreal	4	\$1 500,00
26	February	Skates	Montreal	21	\$2 079,00
46	January	Skates	Montreal	21	\$2 079,00
28	February	Skates	San Francisco	21	\$2 079,00

Очередность сортировки будет зависеть от порядка расположения полей в запросе. То есть, в нашем случае сначала данные будут рассортированы по колонке **Amount**, а затем по **City**.

Пример 3. Направление сортировки.

Несмотря на то, что по умолчанию оператор **ORDER BY** сортирует по возрастанию, мы можем также прописать сортировки значений по убыванию. Для этого в конце каждого поля проставляем оператор **DESC** (что является сокращением от слова **DESCENDING**).

**SELECT \* FROM Sumproduct ORDER BY Amount DESC, City**

ID	Month	Product	City	Quantity	Amount
4	April	Bikes	San Francisco	854	\$320 250,00
22	February	Bikes	Montreal	663	\$248 625,00
25	February	Bikes	New York	658	\$246 750,00
70	March	Skis Long	Montreal	854	\$209 230,00
10	April	Skis Long	Montreal	854	\$209 230,00
21	April	Snow Board	New York	663	\$204 204,00
59	January	Snow Board	Toronto	663	\$204 204,00
63	March	Bikes	Montreal	522	\$195 750,00
12	April	Skis Long	San Francisco	663	\$162 435,00
32	February	Skis Long	San Francisco	663	\$162 435,00
71	March	Skis Long	Toronto	663	\$162 435,00
58	January	Snow Board	Montreal	522	\$160 776,00
80	March	Snow Board	San Francisco	522	\$160 776,00

В данном примере, значение в поле **Amount** были посортированы по убыванию, а в поле **City** - по возрастанию. Оператор **DESC** применяется только для одного столбца, поэтому при необходимости его нужно прописывать после каждого поля, которое принимает участие в сортировке.

Пример 4. Простое фильтрование оператором WHERE.

Давайте из нашей таблицы, например, отберем записи, относящиеся только к определенному товару. Для этого мы укажем дополнительный параметр отбора, который будет фильтровать значение по колонке **Product**.

**SELECT \* FROM Sumproduct WHERE Product = 'Bikes'**

ID	Month	Product	City	Quantity	Amount
2	April	Bikes	Montreal	12	\$4 500,00
3	April	Bikes	Montreal	56	\$21 000,00
4	April	Bikes	San Francisco	854	\$320 250,00
5	April	Bikes	New York	25	\$9 375,00
22	February	Bikes	Montreal	663	\$248 625,00
23	February	Bikes	San Francisco	21	\$7 875,00
24	February	Bikes	San Francisco	54	\$20 250,00
25	February	Bikes	New York	658	\$246 750,00
42	January	Bikes	Montreal	75	\$28 125,00
43	January	Bikes	Toronto	12	\$4 500,00
44	January	Bikes	San Francisco	136	\$51 000,00
45	January	Bikes	New York	21	\$7 875,00
62	March	Bikes	Montreal	4	\$1 500,00
63	March	Bikes	Montreal	522	\$195 750,00
64	March	Bikes	San Francisco	125	\$46 875,00
65	March	Bikes	New York	212	\$79 500,00

Как видим, условие отбора взято в одинарные кавычки, что является обязательным при фильтровании текстовых значений. При фильтровании числовых значений кавычки не нужны.

**Пример запроса для отбора числовых значений:**

**SELECT \* FROM Sumproduct WHERE Amount > 40000 ORDER**

**BY Amount**

ID	Month	Product	City	Quantity	Amount
39	February	Snow Board	Toronto	136	\$41 888,00
61	January	Snow Board	New York	136	\$41 888,00
64	March	Bikes	San Francisco	125	\$46 875,00
44	January	Bikes	San Francisco	136	\$51 000,00
48	January	Skates	San Francisco	522	\$51 678,00
9	April	Skates	New York	663	\$65 637,00
27	February	Skates	Toronto	663	\$65 637,00
65	March	Bikes	New York	212	\$79 500,00
7	April	Skates	Toronto	854	\$84 546,00
67	March	Skates	Toronto	854	\$84 546,00
36	February	Skis Short	San Francisco	522	\$109 098,00
16	April	Skis Short	San Francisco	522	\$109 098,00

В этом примере мы отобрали записи, в которых выручка от реализации составила более **40 тыс. \$** и, дополнительно, все записи посортировали по возрастанию по полю **Amount**.

Пример 5. Фильтрация по диапазону значений (BETWEEN).

Для отбора данных, которые лежат в определенном диапазоне, используется оператор **BETWEEN**. В следующем запросе будут отобраны все значения, лежащие в пределах от **1000 \$** в **2000 \$** включительно, в поле **Amount**.

**SELECT \* FROM Sumproduct WHERE Amount BETWEEN 1000 AND 2000**

ID	Month	Product	City	Quantity	Amount
18	April	Snow Board	Montreal	4	\$1 232,00
41	February	Snow Board	New York	4	\$1 232,00
62	March	Bikes	Montreal	4	\$1 500,00
79	March	Snow Board	Toronto	4	\$1 232,00

Очередность сортировки будет зависеть от порядка расположения полей в запросе. То есть, в нашем случае сначала данные будут посортированы по колонке **Amount**, а затем по **City**.

Пример 6. Выборка пустых записей (IS NULL).

В **SQL** существует специальный оператор для выборки пустых записей (называется **NULL**). Пустой записью считается любая ячейка в таблице, в которую не введены какие-либо символы. Если в ячейку введен **0** или **пробел**, то считается, что поле заполнено.

**SELECT \* FROM Sumproduct WHERE Amount IS NULL**

ID	Month	Product	City	Quantity	Amount
5	April	Bikes	New York	25	
19	April	Snow Board	Toronto	522	

В примере выше, мы нарочно удалили два значения в поле **Amount**, чтобы продемонстрировать работу оператора **NULL**.

**Содержание работы:**

**Задание 1.** Примените сортировку к каждой таблице, созданным в Практической работе № 27.

**Задание 2.** Примените все виды фильтрации к таблицам, созданным в Практической работе № 27.

**Задание 3.** В созданных в Практической работе № 27 таблицах применить группировку по полям

## ПРАКТИЧЕСКАЯ РАБОТА № 38

**Тема: Проведение сортировки и фильтрации данных. Поиск данных по одному и нескольким полям. Поиск данных в таблице. Группировка данных (GROUP BY)**

**Цель работы:** изучить операторы языка SQL для сортировки, фильтрации и группировки данных.

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

**Справочный материал:**

Пример 1. Расширенная фильтрация (AND, OR).

Язык SQL не ограничивается фильтрацией по одному условию, для собственных целей вы можете использовать достаточно сложные конструкции для выборки данных одновременно по многим критериям. Для этого в SQL есть дополнительные операторы, которые расширяют возможности оператора **WHERE**. Такими операторами являются: **AND**, **OR**, **IN**, **NOT**. Приведем несколько примеров работы данных операторов.

**SELECT \* FROM Sumproduct WHERE Amount > 40000 AND City = 'Toronto'**

ID	Month	Product	City	Quantity	Amount
7	April	Skates	Toronto	854	\$84 546,00
19	April	Snow Board	Toronto	522	\$160 776,00
27	February	Skates	Toronto	663	\$65 637,00
39	February	Snow Board	Toronto	136	\$41 888,00
59	January	Snow Board	Toronto	663	\$204 204,00
67	March	Skates	Toronto	854	\$84 546,00
71	March	Skis Long	Toronto	663	\$162 435,00
75	March	Skis Short	Toronto	522	\$109 098,00

**SELECT \* FROM Sumproduct WHERE Month= 'April' OR Month= 'March'**

ID	Month	Product	City	Quantity	Amount
15	April	Skis Short	Toronto	4	\$836,00
74	March	Skis Short	Montreal	4	\$836,00
18	April	Snow Board	Montreal	4	\$1 232,00
79	March	Snow Board	Toronto	4	\$1 232,00
62	March	Bikes	Montreal	4	\$1 500,00
8	April	Skates	San Francisco	25	\$2 475,00
77	March	Skis Short	San Francisco	21	\$4 389,00
14	April	Skis Short	Montreal	21	\$4 389,00
2	April	Bikes	Montreal	12	\$4 500,00
13	April	Skis Long	New York	21	\$5 145,00
72	March	Skis Long	San Francisco	21	\$5 145,00

Давайте объединим операторы **AND** и **OR**. Для этого сделаем выборку велосипедов (**Bikes**) и коньков (**Skates**), которые были проданы в марте (**March**).

**SELECT \* FROM Sumproduct WHERE Product = 'Bikes' OR Product = 'Skates' AND Month= 'March'**



ID	Month	Product	City	Quantity	Amount
2	April	Bikes	Montreal	12	\$4 500,00
3	April	Bikes	Montreal	56	\$21 000,00
4	April	Bikes	San Francisco	854	\$320 250,00
5	April	Bikes	New York	25	\$9 375,00
22	February	Bikes	Montreal	663	\$248 625,00
23	February	Bikes	San Francisco	21	\$7 875,00
24	February	Bikes	San Francisco	54	\$20 250,00
25	February	Bikes	New York	658	\$246 750,00
42	January	Bikes	Montreal	75	\$28 125,00
43	January	Bikes	Toronto	12	\$4 500,00
44	January	Bikes	San Francisco	136	\$51 000,00
45	January	Bikes	New York	21	\$7 875,00
62	March	Bikes	Montreal	4	\$1 500,00
63	March	Bikes	Montreal	522	\$195 750,00
64	March	Bikes	San Francisco	125	\$46 875,00
65	March	Bikes	New York	212	\$79 500,00
66	March	Skates	Montreal	56	\$5 544,00
67	March	Skates	Toronto	854	\$84 546,00
68	March	Skates	San Francisco	212	\$20 988,00
69	March	Skates	New York	56	\$5 544,00

Видим, что в нашу выборку попало за много значений (кроме марта (**March**), также январь (**January**), февраль (**February**) и апрель (**April**)). Оператор **AND** имеет более высокий приоритет, чем оператор **OR**, поэтому сначала были отобраны записи с коньками, которые проданные в марте, а потом все записи, касающиеся велосипедов.

Итак, чтобы получить правильную выборку, нам нужно изменить приоритеты выполнения команд. Для этого используем **скобки**, как в математике. Тогда, сначала будут обработаны операторы в скобках, а затем - все остальные.

**SELECT \* FROM Sumproduct WHERE (Product = 'Bikes' OR Product = 'Skates') AND Month= 'March'**

ID	Month	Product	City	Quantity	Amount
62	March	Bikes	Montreal	4	\$1 500,00
63	March	Bikes	Montreal	522	\$195 750,00
64	March	Bikes	San Francisco	125	\$46 875,00
65	March	Bikes	New York	212	\$79 500,00
66	March	Skates	Montreal	56	\$5 544,00
67	March	Skates	Toronto	854	\$84 546,00
68	March	Skates	San Francisco	212	\$20 988,00
69	March	Skates	New York	56	\$5 544,00

Пример 2. Расширенная фильтрация (оператор IN).

**SELECT \* FROM Sumproduct WHERE ID IN (4, 12, 58, 67)**

ID	Month	Product	City	Quantity	Amount
4	April	Bikes	San Francisco	854	\$320 250,00
12	April	Skis Long	San Francisco	663	\$162 435,00
58	January	Snow Board	Montreal	522	\$160 776,00
67	March	Skates	Toronto	854	\$84 546,00

Оператор **IN** выполняет ту же функцию, что и **OR**, однако имеет ряд преимуществ:

- При работе с длинными списками, предложение с **IN** легче читать;



- Используется меньшее количество операторов, что ускоряет обработку запроса;
- Самое важное преимущество **IN** в том, что в его конструкции можно использовать дополнительную конструкцию **SELECT**, что открывает большие возможности для создания сложных подзапросов.

Пример 3. Расширенная фильтрация (оператор NOT).

**SELECT \* FROM Sumproduct WHERE NOT City IN ('Toronto', 'Montreal')**

ID	Month	Product	City	Quantity	Amount
47	January	Skates	New York	4	\$396,00
52	January	Skis Long	San Francisco	4	\$980,00
41	February	Snow Board	New York	4	\$1 232,00
28	February	Skates	San Francisco	21	\$2 079,00
8	April	Skates	San Francisco	25	\$2 475,00
77	March	Skis Short	San Francisco	21	\$4 389,00
57	January	Skis Short	New York	21	\$4 389,00
13	April	Skis Long	New York	21	\$5 145,00
72	March	Skis Long	San Francisco	21	\$5 145,00
33	February	Skis Long	New York	21	\$5 145,00
69	March	Skates	New York	56	\$5 544,00
40	February	Snow Board	San Francisco	21	\$6 468,00

Ключевое слово **NOT** позволяет убрать ненужные значения из выборки. Также его особенностью является то, что оно проставляется перед названием столбца, участвующего в фильтровании, а не после.

Группировка данных позволяет разделить все данные на логические наборы, благодаря чему становится возможным выполнение статистических вычислений отдельно в каждой группе.

Пример 4. Создание групп (GROUP BY)

Группы создаются с помощью предложения **GROUP BY** оператора **SELECT**. Рассмотрим на примере.

**SELECT Product,**

**SUM(Quantity) AS Product\_num FROM Sumproduct GROUP BY Product**

Product	Product_num
Bikes	3450
Skates	4376
Skis Long	5251
Skis Short	2879
Snow Board	3510

Данным запросом мы извлекли информацию о количестве реализованной продукции в каждом месяце. Оператор **SELECT** приказывает вывести два столбца **Product** - название продукта и **Product\_num** - расчетное поле, которое мы создали для отображения количества реализованной продукции (формула поля **SUM (Quantity)**). Предложение **GROUP BY** указывает СУБД сгруппировать данные по столбцу **Product**. Стоит также отметить, что **GROUP BY** должен идти после предложения **WHERE** и перед **ORDER BY**.

### Пример 5. Фильтрующие группы (HAVING)

Так же, как мы фильтровали строки в таблице, мы можем осуществлять фильтрацию по сгруппированным данным. Для этого в **SQL** существует оператор **HAVING**. Возьмем предыдущий пример и добавим фильтрацию по группам.

```
SELECT Product,  
SUM(Quantity) AS Product_num FROM Sumproduct GROUP  
BY Product HAVINGSUM(Quantity)>4000
```

Product	Product_num
Skates	4376
Skis Long	5251

Видим, что после того, как была посчитана количество реализованного товара в разрезе каждого продукта, СУБД "отсекла" те продукты, которых было реализовано меньше 4000 шт.

Как видим, оператор **HAVING** очень похож на оператора **WHERE**, однако между собой они имеют существенное отличие: **WHERE** фильтрует данные до того, как они будут сгруппированы, а **HAVING** - осуществляет фильтрацию после группировки. Таким образом, строки, которые были изъяты предложением **WHERE** НЕ будут включены в группу. Итак, операторы **WHERE** и **HAVING** могут использоваться в одном предложении.

Рассмотрим пример:

```
SELECT Product,  
SUM(Quantity) AS Product_num FROM Sumproduct WHERE Product<>'Skis  
Long' GROUP BYProduct HAVING SUM(Quantity)>4000
```

Product	Product_num
Skates	4376

Мы к предыдущему примеру добавили оператор **WHERE**, где указали товар **Skis Long**, что в свою очередь повлияло на группирование оператором **HAVING**. Как результат видим, что товар **Skis Long** не попал в перечень групп с количеством реализованной продукции больше 4000 шт.

### Пример 6. Группировка и сортировка

Как и при обычной выборке данных, мы можем сортировать группы после группировки оператором **HAVING**. Для этого мы можем использовать уже знакомый нам оператор **ORDER BY**. В данной ситуации его применения аналогичное предыдущим примерам. К примеру:

```
SELECT Product,  
SUM(Quantity) AS Product_num FROM Sumproduct GROUP  
BY Product HAVINGSUM(Quantity)>3000 ORDER BY SUM(Quantity)
```

или просто укажем номер поля по порядку, по которому хотим сортировать:

```
SELECT Product,  
SUM(Quantity) AS Product_num FROM Sumproduct GROUP  
BY Product HAVINGSUM(Quantity)>3000 ORDER BY 2
```

Product ▾	Product_num ▾
Bikes	3450
Snow Board	3510
Skates	4376
Skis Long	5251

Видим, что для сортировки сводных результатов нам нужно просто прописать предложения с **ORDER BY** после оператора **HAVING**. Однако есть один нюанс. **СУБД Access** не поддерживает сортировку групп по псевдонимами колонок, то есть в нашем примере, чтобы сортировать значения, мы не сможем в конце запроса прописать **ORDER BY Product\_num**.

#### **Содержание работы:**

**Задание 1.** Примените сортировку к каждой таблице, созданным в Практической работе № 28.

**Задание 2.** Примените все виды фильтрации к таблицам, созданным в Практической работе № 28.

**Задание 3.** В созданных в Практической работе № 28 таблицах применить группировку по полям

## ПРАКТИЧЕСКАЯ РАБОТА № 39

### Тема: Разработка базы данных на языке SQL по индивидуальным заданиям

**Цель работы:** закрепить знания по разработке базы данных, используя язык SQL

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### Содержание работы:

**Задание 1.** Создайте новую базу данных с использованием языка SQL, сохраните ее в личной папке под именем Магазин.

1. Создайте таблицу Товары следующей структуры:

Код товара	Название	Марка товара	Цена
1	телевизор		
2	телевизор		
3	телевизор		
4	телевизор		
5	DVD-плеер		
6	DVD-плеер		
7	видеокамера		
8	видеокамера		
9	музыкальный центр		
10	музыкальный центр		

2. Укажите первичный ключ в поле Код товара

3. Заполните таблицу данными.

4. Выполните сортировку данных таблицы по возрастанию цены.

5. Используя Фильтр, выберите данные по телевизорам, цена которых не превосходит определенного значения, например, меньше 300 евро.

6. Отобразите все данные таблицы.

7. В базе данных Магазин создайте таблицу Поставщики

Код поставщика	Название фирмы	Телефон	Факс	Адрес
1010	М-видео	(095) 207-9475	(095) 207-1045	Мира 78/11
1020	Портал	(095) 158-7862	(095) 158-7895	Труда 136
1030	ПК-Мир	(095) 296-0590	(095) 296-0486	Первая 12

8. Объявите ключевым поле Код поставщика.

9. Создайте Запрос для отображения названий товаров, их цен и названий фирм – поставщиков.

10. Создайте Запрос для отображения в алфавитном порядке названий фирм, поставляющих телевизоры, названия и марки этого товара, а также его цены.

11. Создайте Запрос для отображения средних цен на все товары (функция Avg групповых операций).

12. Создайте Запрос для отображения оптовых цен со скидкой 5% на каждый товар, с указанием названия фирмы и ее адреса (Создание вычисляемого поля).

13. Создайте Запрос на обновление в таблице Товары1 цен с учетом сезонных скидок в 10%.

14. Создайте Запрос на создание таблицы DVD-плееры, отображающей данные о ценах на DVD-плееры, марке товара, а также о названиях и телефонах фирм, их поставляющих.

## **ПРАКТИЧЕСКАЯ РАБОТА № 40**

### **Тема: Разработка базы данных на языке SQL по индивидуальным заданиям**

**Цель работы:** закрепить знания по разработке базы данных, используя язык SQL

**Оборудование:** ПК, интернет, программное обеспечение – MS Access, инструкции по выполнению работы.

#### **Содержание работы:**

**Задание 1.** Создать БД, используя язык SQL, по выбранным вариантам. К базе данных создать 15 запросов – простых и сложных, с вычисляемыми полями, с символами подстановки.

#### **Индивидуальные варианты:**

Вариант 1. База данных «РЕСТОРАН».

Постоянным клиентам предоставляется возможность заказать столик заранее. Официант указывает столик, открывает гостевой счет и вводит заказы в соответствии с меню. Далее заказ автоматически обрабатывается, формируются марки на приготовление выбранных блюд и направляют их на производство, в соответствующие цеха кухни, в бар. Расчеты с посетителем сводятся к простой операции: на бланке печатается итоговый счет. Если клиент – постоянный посетитель, то соответствующие привилегии рассчитываются автоматически, затем указывается способ оплаты и полученная от клиента сумма.

Вариант 2. База данных «КИНОТЕАТР».

Создаваемая информационная система предназначена для учета проданных билетов в кинотеатре. Кинотеатр имеет несколько залов. Сеансы планируются для каждого зала отдельно. Система формирует базу данных, включающую следующую информацию: Место и сеанс, Справочник кинозалов, Справочник сеансов и стоимость, Справочник фильмов, Справочник жанров. Система формирует отчеты: Отчет о посещаемости по месяцам, Отчет о популярности фильмов, Отчет о популярности жанров. Необходимо предусмотреть возврат билетов и денег.

Вариант 3. База данных «ГОСТИНИЦА».

Номера в гостинице имеют разный уровень обслуживания и соответственно разную стоимость, (предоставление информации о свободных номерах и их стоимости). Клиенты могут бронировать номера по телефону или Интернету. За номерами прикреплен обслуживающий персонал. Необходимо вести учет обслуживания и оплаты номеров, (заказы в номер, телефонные звонки и т. д.). Клиент может несколько раз останавливаться в гостинице в разных номерах.

Вариант 4. База данных «ФИТНЕС – КЛУБ».

Они предлагают пакеты услуг – абонементы. Подразумевая предоплату определенного набора услуг. Абонемент позволяет пользоваться ими в течение определенного времени. Для идентификации владельца абонемента используются клубные карты. Комплекс позволяет быстро и просто осуществлять резервирование ресурсов по просьбе постоянного клиента

предприятия: как тренера, так и места — спортзала, солярия, бассейна для персональных тренировок или занятий.

#### Вариант 5. База данных «ОПТОВЫЙ СКЛАД».

Создаваемая информационная система предназначена для учета деятельности оптового склада. Оптовый склад состоит из нескольких складских помещений, каждое помещение имеет наименование, адрес и кладовщика. Склад принимает партии товаров от поставщиков и отпускает его клиентам мелкими партиями. Требуется вести (количественный и стоимостной) учет поступающих и отпускаемых товаров, поставщиков и клиентов, формировать приходные и расходные накладные. Сведения о товаре: Артикул, Наименование полное, Наименование сокращенное, Производитель, Поставщик, Количество, Цена. Сведения о поставщике и клиенте: Наименование, Адрес, Телефон. Накладная включает: Номер, Дата, Клиент, Список товаров, Общая сумма, Кладовщик. В системе формируются отчеты о поступлении и отпуске товаров на складе за произвольный период.

#### Вариант 6. База данных «РЕКЛАМНОЕ АГЕНТСТВО».

Создаваемая информационная система должна вести учет деятельности рекламного агентства. Рекламное агентство регистрирует заявки от рекламодателей и публикует рекламы в печатных изданиях. О рекламодателе регистрируются следующие данные: Наименование, Адрес, Руководитель, Телефон, Заявка, Оплата, Издание, Место размещения рекламы. Заявка включает: Вид рекламы, Объем, Желаемые издания, Количество выходов рекламы, Дополнительная информация. Заявка от рекламодателя может содержать публикацию в несколько печатных изданиях и на различные даты выхода. Справочник печатных изданий включает: Наименование, Виды реклам, Стоимость рекламы. Требуется вести списки печатных изданий с их расценками на рекламу, списки рекламодателей, заявок. Система должна обеспечить оперативный просмотр списка заявок (печатные издания, рекламодатель, стоимость) на любую вводимую дату, а также формирование отчета о заявленных и выполненных рекламах.

#### Вариант 7. База данных «МАГАЗИН “ЦВЕТЫ”».

Создаваемая информационная система предназначена для учета деятельности магазина по продаже цветов. В системе формируется база данных отдельных цветов и готовых букетов: Наименование цветка или букета, Поставщик цветов, Состав букета, Стоимость, Срок поступления, Срок и место хранения (выставочный зал, склад), Дата продажи. В системе ведется учет бракованных и увядших цветов. Формируется отчет о движении товара за заданный период времени.

#### Вариант 8. База данных «АЭРОПОРТ».

Создаваемая информационная система предназначена для учета деятельности гостиницы. В гостинице имеется список номеров: Место нахождения номера, Класс, Число мест, Признак занятости места, Дата освобождения номера. Каждый гость проходит регистрацию: Паспортные данные, Даты приезда и отъезда, Номер, Место, Цель приезда, Организация, в которую прибыл (в случае командировки). Администратор гостиницы

осуществляется поселение гостя: выбор подходящего номера (при наличии свободных мест), регистрация, оформление квитанции. В системе автоматически формируется квитанция об оплате услуг гостиницы. Система должна предусмотреть оформление дополнительной квитанции в случае продления гостем срока проживания в гостинице. В системе имеется возможность поиска гостя по произвольному признаку и формируется отчет о текущем состоянии номеров гостиницы (номер, место, не занят/ занят и кем, дата отъезда).

#### Вариант 9. База данных «МУЗЫКАЛЬНЫЙ МАГАЗИН».

Создаваемая информационная система предназначена для учета музыкальных произведений в магазине. В системе формируются: База групп и исполнителей, База песен, База дисков с перечнем песен (в виде ссылок). База групп и исполнителей содержит: Наименование группы или исполнителя, Страна, Год образования группы или год начала творческого пути, Краткое содержание творческого пути. База песен содержит: Название, Автор текста, Автор музыки, Время звучания. База дисков содержит: Название диска, Перечень песен (название, исполнитель, время звучания, номер трека). Система имеет возможность поиска всех песен заданной группы (исполнителя). Имеется возможность выбора всех дисков, где встречается заданная песня.

#### Вариант 10. База данных «АВТОЗАПРАВКА (АЗС)».

Создаваемая информационная система предназначена для учета деятельности автозаправки. На автозаправке имеются несколько колонок для заправки топливом: АИ-98, АИ-95, АИ-92, АИ-80, АИ-76, Дизельное топливо. В базе данных должна быть информация: О колонках, О видах бензина, О ценах и остатках. Необходимо учитывать отпуск топлива по чеку: Номер колонки, Тип топлива, Количество, Цена за литр, Стоимость. Предусмотреть отпуск топлива по дисконтной карточке со скидкой, при этом необходимо учитывать: Номер карточки, Общее количество отпущенного топлива, Скидка в %. Размер скидок зависит от общего количества заправленного топлива. В 19 часов – “пересменка” операторов АЗС, печатается отчет об отпуске топлива за время от 19 часов предыдущего дня до 19 часов текущего дня.

## **Информационное обеспечение обучения**

### **Печатные издания:**

#### **Основные учебные издания:**

1. Данилова, Л. Ф. Проектирование и разработка баз данных: практикум для СПО / Л. Ф. Данилова, А. Н. Полетайкин. — 2-е изд. — Саратов: Профобразование, 2025. — 172 с. — ISBN 978-5-4488-2589-7. — Текст: электронный // Электронный ресурс цифровой образовательной среды СПО PROФобразование: [сайт]. — URL: <https://profspo.ru/books/152770>
2. Кумскова, И. А., Базы данных: учебник / И. А. Кумскова. — Москва: КноРус, 2026. — 400 с. — ISBN 978-5-406-15045-0. — URL: <https://book.ru/book/958783>

#### **Дополнительные учебные издания:**

3. Молдованова, О. В. Информационные системы и базы данных: учебное пособие для СПО / О. В. Молдованова. — 2-е изд. — Саратов: Профобразование, 2024. — 177 с. — ISBN 978-5-4488-1177-7. — Текст: электронный // Электронный ресурс цифровой образовательной среды СПО PROФобразование: [сайт]. — URL: <https://profspo.ru/books/139095>
4. Ткаченко, С. Н., Основы проектирования баз данных: учебник / С. Н. Ткаченко. — Москва: КноРус, 2026. — 176 с. — ISBN 978-5-406-14991-1. — URL: <https://book.ru/book/958706>

### **Интернет-ресурсы:**

#### **Электронно-библиотечная система:**

5. ЭБС «Znanium»
6. ЭБС «PROФобразование»
7. ЭБС «Book.ru»