МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ЛУГАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ ВЛАДИМИРА ДАЛЯ»

Стахановский инженерно-педагогический институт (филиал)
федерального государственного бюджетного образовательного учреждения
высшего образования «Луганский государственный университет имени
Владимира Даля»

Кафедра информационных систем

КОНСПЕКТ ЛЕКЦИЙ

по дисциплине

«ПРОИЗВОДСТВЕННОЕ ОБУЧЕНИЕ»

для студентов направления подготовки
Профессиональное обучение (по отраслям), профили
«Информационные технологии и системы»,
«Профессиональная психология»
(в 2-х частях). Часть 2.

Луганск 2023

УДК 004.912, 004.62, 004.63

Рекомендовано к изданию Учебно-методическим советом
ФГБОУ ВО «ЛГУ им. В. ДАЛЯ»
(протокол № от . .2023 г.)

Конспект лекций по дисциплине «**Производственное обучение**» для студентов направления подготовки **Профессиональное обучение** (по отраслям), профили **«Информационные технологии и системы», «Профессиональная психология»** (в 2-х частях). Часть 2. / Сост.: М.В. Авершина. – **Стаханов**: ФГБОУ ВО «ЛГУ им. В. Даля», 2023. – 56 с.

Конспект лекций содержит материал по 2 темам дисциплины, описание которых сопровождается теоретическими сведениями. К каждой теме приведены вопросы для самопроверки, список рекомендованной литературы.

Предназначен для студентов профилей «Информационные технологии и системы», «Профессиональная психология».

Составитель: ст. преп. Авершина М.В.

Ответственный за выпуск: доц. Карчевский В.П.

Рецензент: доц. Авершин А.А.

Содержание

Тема 1
Базы данных и СУБД

Цели:

- ознакомиться с основными понятиями БД и СУБД;
- изучить категории пользователей банков данных;
- рассмотреть основные модели баз данных;
- ознакомиться с элементами реляционных баз данных.

План

1. Основные понятия баз данных.
2. Пользователи банков данных.
3. Архитектура базы данных.
4. Модели баз данных.
5. Основные подходы к хранению данных.
6. Элементы реляционных баз данных.
7. Языковые средства баз данных.

1. Основные понятия баз данных

Одним из важнейших условий обеспечения эффективного функционирования любого предприятия или организации является наличие развитой *информационной системы*.

Информационная система представляет собой систему, реализующую автоматизированный сбор, обработку и манипулирование данными и включающую технические средства обработки данных, программное обеспечение и обслуживающий персонал. Современной формой информационных систем являются *банки данных*.

Банк данных – это система специальным образом организованных данных – баз данных, а также технических, программных, языковых и организационно-методических средств, предназначенных для обеспечения централизованного накопления и коллективного многоцелевого использования данных.

Основными компонентами банка данных являются:

- вычислительная система (технические средства и операционная система);
- база данных (непосредственно вся информация);
- система управления базой данных, СУБД (программное обеспечение для организации хранения и использования информации);
- набор прикладных программ.

К основным функциям банка данных относятся:

- хранение данных и их защита;
- изменение (обновление, добавление и удаление) хранимых данных;
- поиск и отбор данных по запросам пользователей;
- обработка данных и вывод результатов.

База данных (БД) является ядром банка данных и представляет совокупность взаимосвязанных и вместе хранящихся данных из определенной предметной области, организованных специальным образом и хранимых во внешней памяти (файлах базы данных).

В компьютерных базах данных может содержаться любая информация: от простого текста (например, фамилия, имя и адрес) до сложной структуры, включая рисунки, звуки и изображения. Хранение данных в заранее известном формате позволяет извлекать данные в желаемом формате благодаря использованию разных методов обработки. Функционирование базы данных обеспечивает *администратор базы данных*.

Администратор базы данных – лицо, отвечающее за выработку требований к базе данных, её проектирование, реализацию, эффективное использование и сопровождение, включая управление учётными записями пользователей БД и защиту от несанкционированного доступа. Не менее важной функцией администратора БД является поддержка целостности базы данных.

Целостность БД – свойство БД, означающее, что база данных содержит полную и непротиворечивую информацию, необходимую и достаточную для корректного функционирования приложений.

Система управления базой данных (СУБД) – это совокупность языковых и программных средств, предназначенных для создания, ведения и совместного использования БД многими пользователями.

К функциям СУБД относится:

– перевод схемы, определяющей структуру данных и записанной на языке определения данных в некоторое внутреннее представление, используемой системой при дальнейшей работе с данными;

– создание БД (загрузка данных в БД);

– реализация запросов пользователей (формулируемых на специальном языке, принятом в данной СУБД) на сортировку и отбор по заданным критериям, а также извлечение некоторой части БД, что может сопровождаться редактированием и обработкой информации;

– обновление некоторой части БД без изменения структуры данных;

– обеспечение защиты данных и приоритетов в их использовании.

Можно сказать, что основная функция СУБД – это *предоставление пользователю БД возможности работы с ней, не вникая в детали на уровне аппаратного обеспечения*. То есть все запросы пользователя к БД, добавление и удаление данных, выборки, обновление данных – все это обеспечивает СУБД.

Программы, с помощью которых пользователи работают с базой данных, называются **приложениями**. В общем случае с одной базой данных могут работать множество различных приложений. Например, если база данных моделирует некоторое предприятие, то для работы с ней может быть создано приложение, которое обслуживает подсистему учета кадров, другое приложение может использоваться для расчета заработной платы

сотрудников, третье предназначено для планирования производственного процесса и т. д. При рассмотрении приложений, работающих с одной базой данных, предполагается, что они могут работать параллельно и независимо друг от друга, и именно СУБД призвана обеспечить работу множества приложений с единой базой данных таким образом, чтобы каждое из них выполнялось корректно, но учитывало все изменения в базе данных, вносимые другими приложениями. Приложения могут создаваться как в среде СУБД, так и вне СУБД – с помощью системы программирования, использующей средства доступа к БД (например, Delphi или C++ Builder).

Для работы с базой данных во многих случаях можно обойтись только средствами СУБД, например, создавая запросы и отчеты. Приложения разрабатывают главным образом в случаях, когда требуется обеспечить удобство работы с БД неквалифицированным пользователям или интерфейс СУБД не устраивает пользователя.

2. Пользователи банков данных

Как любой программно-организационный и технический комплекс, банк данных существует во времени и в пространстве. Он имеет определенные стадии своего развития:

1. Проектирование.
2. Реализация.
3. Эксплуатация.
4. Модернизация и развитие.
5. Полная реорганизация.

На каждом этапе своего существования с банком данных связаны разные категории пользователей. Определим основные категории пользователей и их роль в функционировании банка данных.

Конечные пользователи

Это основная категория пользователей, в интересах которых и создается банк данных. В зависимости от особенностей создаваемого банка данных круг конечных пользователей может существенно различаться. Это могут быть случайные пользователи, обращающиеся к БД время от времени за получением некоторой информации, а могут быть регулярные пользователи. В качестве случайных пользователей могут рассматриваться, например, возможные клиенты фирмы, просматривающие каталог продукции или услуг с обобщенным или подробным их описанием.

Регулярными пользователями могут быть сотрудники организации, работающие со специально разработанными для них программами, которые обеспечивают автоматизацию их деятельности при выполнении своих должностных обязанностей. Главный принцип состоит в том, что от конечного пользователя не должно требоваться каких-либо специальных знаний в области вычислительной техники и языковых средств.

Администраторы банка данных

Это группа пользователей, которая на начальной стадии разработки банка данных отвечает за его оптимальную организацию с точки зрения одновременной работы множества конечных пользователей, на стадии

эксплуатации отвечает за корректность работы данного банка информации в многопользовательском режиме. На стадии развития и реорганизации эта группа пользователей отвечает за возможность корректной реорганизации банка без изменения или прекращения его текущей эксплуатации. Таким образом, пользователи этой группы отвечают за создание БД, технический контроль, обеспечение быстродействия системы, ее техническое обслуживание.

Разработчики и администраторы приложений (прикладные программисты)

Это группа пользователей, которая функционирует во время проектирования, создания и реорганизации банка данных. Администраторы приложений координируют работу программистов при разработке конкретного приложения или группы приложений, объединенных в функциональную подсистему. Разработчики конкретных приложений работают с той частью информации из базы данных, которая требуется для конкретного приложения, они отвечают за написание прикладных программ, использующих БД. Для этих целей применимы различные языки программирования.

Не в каждом банке данных могут быть выделены все типы пользователей. Так при разработке информационных систем с использованием настольных СУБД администратор банка данных, администратор приложений и разработчики часто существует в одном лице. Однако при построении современных сложных корпоративных баз данных, которые используются для автоматизации всех или большей части бизнес-процессов в крупной фирме или корпорации, могут существовать и группы администраторов приложений, и отделы разработчиков.

3. Архитектура базы данных

В процессе научных исследований, посвященных тому, как именно должны быть устроена СУБД, предлагались различные способы реализации. Самым жизнеспособным из них оказалась предложенная американским комитетом по стандартизации ANSI (American National Standards Institute) трехуровневая система организации БД, в соответствии с которой выделяют три уровня представления данных (рис. 1.1).

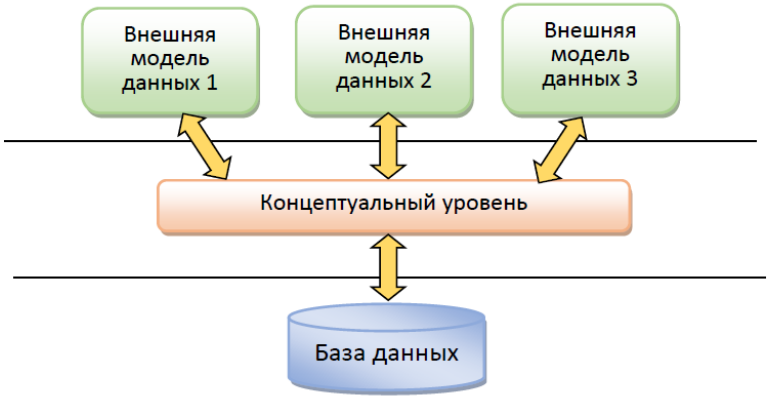

Рисунок 1.1 – Трехуровневая система организации БД

Внешний уровень

Является самым верхним уровнем или уровнем пользователя. Это совокупность внешних представлений данных, которые обрабатывают приложения и какими их видит пользователь на экране. Это может быть таблица с отсортированными данными, с примененным фильтром, форма, отчет, результат запроса. Внешние представления взаимосвязаны, т.е. из одного внешнего представления можно получить другое.

Концептуальный уровень

Является центральным. Здесь БД представлена в наиболее общем виде, который объединяет данные, используемые всеми приложениями. Фактически концептуальный уровень отражает обобщенную модель предметной области (объектов реального мира), для которой создавалась БД.

Физический уровень

Это собственно данные, расположенные на внешних носителях.

4. Модели баз данных

Основная задача проектирования базы данных состоит в устранении необходимости переделывания созданной структуры по мере развития системы. Для ее решения создается комплекс взаимосвязанных **моделей данных**.

Модель данных – это некоторая абстракция, которая будучи приложима к конкретным данным, позволяет пользователям и разработчикам трактовать их уже как информацию, т.е. сведения, содержащие не только данные, но и взаимосвязь между ними.

Первым этапом проектирования является разработка концептуальной модели, когда определяется, какие именно данные необходимо хранить в БД – отражается предметная область в виде совокупностей информационных объектов и их структурных связей.

Далее строится внутренняя модель, когда решается, как данные должны быть представлены в БД – создается соответствующая структура хранения, а также определяется отображение между внутренней и концептуальными схемами.

Впоследствии создается внешняя модель, когда осуществляется представление необходимых данных для пользователей, а также определяется отображением между внешней и концептуальными схемами.

Модель данных должна быть разработана таким образом, чтобы по возможности быть стабильной. Известны три основные модели данных:

1) **Иерархическая модель** предполагает организацию данных в виде древовидной структуры. На самом верхнем уровне структуры находится корень дерева, не имеющий вышестоящих узлов. Остальные узлы связаны между собой через исходный узел, находящийся выше (рис. 1.2).

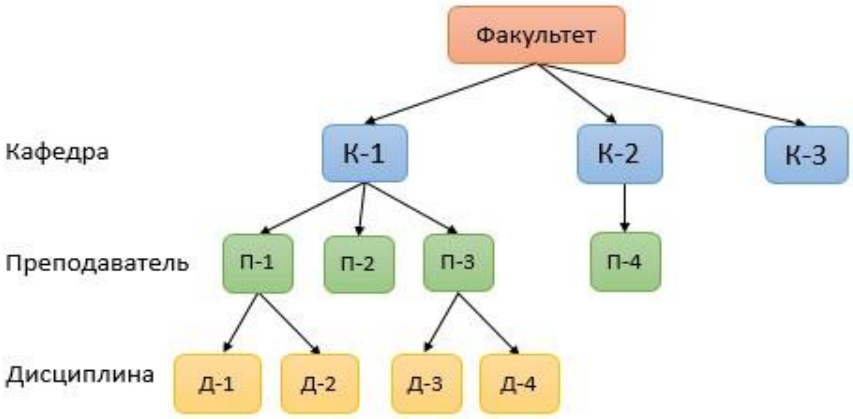

Рисунок 1.2 – Пример иерархической структуры данных

2) **Сетевая модель** предполагает организацию данных в виде сетевой структуры, когда любой элемент может быть связан с любым другим элементом (рис. 1.3).

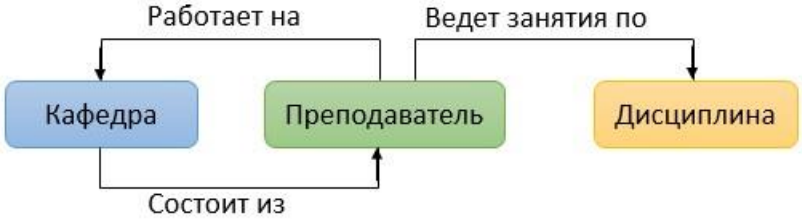

Рисунок 1.3 – Пример сетевой модели данных

3) **Реляционная модель** (от слова relation – отношение) предполагает использование двумерных таблиц (отношений), связь между которыми осуществляется посредством значений одного или нескольких совпадающих полей. При этом каждая строка таблицы уникальна, что обеспечивается использованием ключей, содержащих одно или несколько полей таблицы (рис. 1.4).

Факультет

Код_фак	Название_фак
10	Математический
20	Физический
30	Исторический

Кафедра

Код_фак	Код_каф	Название_каф
10	1	Прикладной математики и кибернетики
10	2	Математического анализа
10	3	Геометрии и топологии
20	4	Электроники
20	5	Общей физики

Рисунок 1.4 – Пример реляционной модели данных

5. Основные подходы к хранению данных

Какая бы модель не была использована для хранения и обработки данных необходимо, чтобы выполнялись правила полноты, непротиворечивости и целостности данных.

Полнота данных – база данных должна обеспечивать полное и адекватное описание предметной области. При этом должен соблюдаться

принцип минимальной избыточности. Особое внимание на полноту обращается на этапе проектирования базы данных.

Непротиворечивость данных – данные, которые хранятся в базе данных, должны проверяться на правильность при вводе, существует запрет на дублирование данных.

Целостность данных:

– при описании связей должна обеспечиваться правильность ссылок между таблицами, что обеспечивается каскадным обновлением и удалением;

– блокировка модифицируемых записей, при одновременной работе с БД;

– механизм транзакций (последовательность операций над БД), позволяющий вернуться на несколько шагов назад, отменив последние действия, осуществив «откат».

Основываясь на физическом представлении организации хранения данных, можно выделить следующие виды архитектуры для хранения данных:

Локальные базы данных

Располагаются на компьютере, на котором работает пользователь. Вся информация используется в монопольном режиме. Пользователь сам регулирует доступ к данным.

Централизованные базы данных

Централизованная база данных хранится на центральном компьютере, пользователи и прикладные программы имеют удаленный доступ к базе данных. Преимущества централизованной БД – минимальные затраты на корректировку. Такая система предпочтительна, если важны требования к безопасности и целостности данных. Недостатком является сложность в обслуживании, увеличение времени отклика, затраты на передачу данных, неисправность центральной системы выводит из рабочего состояния всю сеть.

Централизованные базы данных реализуются на базе двух архитектур с сетевым доступом:

– архитектура «файл – сервер» предполагает выделение одной из машин в сети в качестве центральной (сервер файлов), на которой хранится совместно используемая централизованная база данных. Остальные машины сети исполняют роль рабочих станций, на которых в основном и производится обработка данных, получаемых в виде файлов базы данных в соответствии с запросами пользователей;

– архитектура «клиент – сервер[1]» стала стандартом для современных СУБД, когда сервер владеет и распоряжается информационными ресурсами системы, а клиент пользуется ими. Центральная машина (сервер базы данных) помимо хранения базы данных обеспечивает выполнение основного объема обработки данных. Запрос клиента (рабочей станции) порождает поиск и извлечение данных на сервере, которые затем транспортируются по сети к клиенту (в отличие от передаваемых файлов в предыдущей архитектуре).

[1] Компьютер, располагающий ресурсами и предоставляющий их, называется **сервером**. Компьютер, который обращается к серверу за данными или требованием решения задачи, называется **клиентом**.

Распределенные базы данных

Распределенная база данных предполагает хранение и управление данными в нескольких узлах компьютерной сети и передачу данных между ними в процессе выполнения запросов. На разных компьютерах могут храниться не только различные таблицы, но и разные фрагменты одной огромной таблицы. При этом для пользователя не имеет значения, как организовано хранение данных.

6. Элементы реляционных баз данных

Основным объектом реляционных баз данных является *таблица*. Простейшая база данных имеет хотя бы одну таблицу. Структуру любой двумерной таблицы составляют столбцы и строки, аналогами которых в базе данных являются *поля* и *записи* (рис. 1.5).

Рисунок 1.5 – Пример реляционной БД

Поле – это элементарная единица логической организации данных, которая соответствует неделимой единице информации (реквизиту). Поле – наименьший поименованный элемент информации, хранящейся в БД и рассматриваемой как единое целое.

Поле обладает следующими характеристиками:

- *имя* – определяет, как следует обращаться к данным этого поля;
- *тип* – определяет тип данных, которые могут содержаться в поле;
- *размер* – определяет предельную длину размещаемых в поле данных;
- *формат* – определяет способ форматирования данных в поле.

С полями базы данных можно производить следующие операции:

- описание (указание имени, типа и длины поля);
- редактирование (изменение имени, типа и длины поля);
- манипуляция (добавление, перемещение и удаление полей).

Совокупность полей базы данных определяет ее *структуру*. Изменив состав полей (или их свойства), мы изменяем структуру БД и, соответственно, получаем новую БД.

Единицей хранения и доступа к базе данных является *запись*. Записью, например, может быть библиографическая карточка в электронном каталоге, листок по учету кадров в базе данных отдела кадров, реферат статьи в автоматизированном реферативном журнале, чертеж детали в системе автоматизированного проектирования.

Запись – это совокупность логически связанных полей, соответствующих одному объекту.

С записями можно производить следующие операции:

- ввод данных в поля записей;
- редактирование записей;
- индексирование записей;
- сортировка записей;
- поиск записей по одному или нескольким критериям.

Таблица – это совокупность записей одной структуры.

В структуре записи указываются *ключевые поля*, которые могут быть простыми или составными. Одно или несколько полей, комбинация значений которых однозначно определяет каждую запись в таблице, называется **первичным (главным) ключом**. При этом в таблице не может быть одинаковых первичных ключей. Поле «Код» в примере, приведенном на рисунке 1.5, однозначно определяет запись и является первичным ключом. Он является простым, так как состоит из одного поля.

С таблицами можно производить следующие операции:

- *выборка* – выполняется над одной таблицей (результирующее отношение содержит подмножество записей, удовлетворяющих некоторому условию);
- *объединение* – выполняется над двумя таблицами (результирующее отношение включает все записи первой таблицы и недостающие кортежи второго отношения);
- *пересечение* – выполняется над двумя таблицами (результирующее отношение включает все записи первой таблицы, которые есть также и во втором отношении);
- *соединение* – выполняется над двумя таблицами, в каждой из которых выделяется атрибут, по которому будет производится объединение (результирующее отношение включает все атрибуты исходных таблиц).

Для создания базы данных средствами любой СУБД необходимо выполнить четыре этапа:

- *создание структуры базы данных*, т.е. определение перечня полей, из которых состоит каждая таблица, их типов (числовой, текстовый, логический и т.д.) и размеров, а также определение ключевых полей для обеспечения необходимых связей между данными;
- *ввод и редактирование данных в таблицах* с помощью представляемой по умолчанию стандартной формы в виде таблицы или с помощью специально создаваемых экранных форм;
- *обработка содержащихся в таблицах данных* с помощью запросов;
- *вывод результатной информации* с использованием отчетов.

Названные этапы реализуются с помощью различных команд.

Команды для работы с файлами обеспечивают:

- создание новых и открытие уже существующих баз данных;
- сохранение и переименование ранее созданных объектов;
- печать объектов базы данных.

Команды редактирования обеспечивают:

– копирование объектов;
– перемещение объектов;
– удаление объектов;
– вставку рисунков, диаграмм и созданных в других программах объектов;
– поиск и замену информации в документе или его части.

Команды форматирования обеспечивают:

– выравнивание данных;
– установку различных видов шрифтов;
– выбор толщины и цвета линий, фона и др.

Команды для работы с несколькими окнами обеспечивают:

– работу сразу с несколькими окнами;
– изменение расположения и размеров окна;
– деление одного большого окна на части и их фиксацию.

7. Языковые средства баз данных

Описание базы данных обеспечивается **языком описания данных (языком определения данных)**. Кроме того, для выполнения управления данными используется **язык манипулирования данными**, который содержит набор команд управления данными и позволяет выполнять операции над данными из базы данных: заносить, выбирать, модифицировать и удалять их. Благодаря языковым средствам системы пользователи получают доступ к функциональным возможностям используемых моделей данных.

В современных СУБД обычно поддерживается **единый интегрированный язык**, содержащий все необходимые средства для работы с базами данных. Имеются примеры языков СУБД, объединяющих описание и манипулирование данными. В реляционных СУБД таким языком является разработанный компанией IBM язык SQL.

SQL (Structured Query Language – структурированный язык запросов) – это язык программирования, который применяется для взаимодействия пользователя с базой данных.

В настоящее время SQL используется для реализации всех функциональных возможностей СУБД. Возможности языка SQL:

– организация данных (позволяет изменять структуру представления данных, устанавливать соотношения между элементами базы данных);
– чтение данных (позволяет читать данные из базы и пользоваться ими);
– обработка данных (позволяет изменять базу данных: добавлять в нее новые данные, обновлять или удалять уже имеющиеся);
– совместное использование данных (позволяет пользоваться данными параллельно работающим пользователям, не мешая друг другу);
– управление доступом (ограничивает возможности пользователей по изменению данных и защищает их от несанкционированного доступа);
– обеспечение целостности данных (защищает базу данных от разрушения из-за несогласованных действий или отказа системы).

Некоторые СУБД располагают автономными языками, которые не только реализуют функции определения и манипулирования данными, но и обладают средствами, свойственными традиционным языкам программирования. Благодаря этому они могут использоваться как средства создания прикладных программ и для формулировки запросов пользователей к базе данных.

Контрольные вопросы

1. Какую систему называют информационной?
2. Что называется банком данных, базой данных?
3. Перечислите основные компоненты и функции банка данных.
4. Приведите примеры баз данных.
5. Что такое целостность БД?
6. Стадии развития банка данных?
7. Опишите основные категории пользователей и их роль в функционировании банка данных
8. Перечислите основные модели данных.
9. Нарисуйте трехуровневую систему организации БД.
10. Перечислите основные модели представления данных.
11. Приведите пример иерархической модели представления данных.
12. Что называется СУБД?
13. Перечислите основные функции СУБД.
14. Что называется таблицей, записью БД, полем БД?
15. Приведите основные характеристики поля реляционной БД.
16. Как классифицируются базы данных по архитектуре хранения данных?
17. Что такое SQL?
18. Перечислите операции, которые можно выполнят с таблицами реляционной БД.
19. Что такое ключевое поле?
20. Установите соответствие между объектами, с которыми работает СУБД, и их определениями. К каждой позиции, данной в первом столбце, подберите соответствующую позицию из второго столбца.

Объект БД		**Определение**	
1.	Запись	А	Совокупность таблиц, связанных общими характеристиками описываемых объектов
2.	Таблица	Б	Множество значений одного параметра объектов, описываемых базой данных
3.	Поле	В	Совокупность экземпляров писей одной структуры
4.	База данных	Г	Совокупность характеристик объекта, описываемого базой данных

Список использованных и рекомендованных источников

1. Мердина О.Д. Базы данных : учебное пособие / О.Д. Мердина. – СПб. : Изд-во СПбГЭУ, 2019. – 99 с.

2. Нестеров С. А. Базы данных : учебник и практикум / С. А. Нестеров. — М. : Издательство Юрайт, 2019. — 230 с

3. Стружкин Н. П. Базы данных: проектирование. Практикум : учеб. пособие / Н. П. Стружкин, В. В. Годин. — М. : Издательство Юрайт, 2018. — 291 с.

Тема 2
Работа пользователя в Microsoft Access

Цели:

– ознакомиться с основными элементами интерфейса СУБД MS Access;

– рассмотреть основные объекты СУБД MS Access, изучить их назначение;

– изучить этапы создания основных объектов MS Access.

План

1. Интерфейс MS Access.
2. Этапы проектирования базы данных.
3. Объекты базы данных MS Access.
   - 3.1. Таблицы
     - 3.1.1. Типы данных.
     - 3.1.2. Структура базы данных.
     - 3.1.3. Создание таблиц.
     - 3.1.4. Задание ключевого поля.
     - 3.1.5. Свойства полей.
     - 3.1.6. Создание подстановок с помощью мастера подстановок.
     - 3.1.7. Создание маски ввода.
     - 3.1.8. Связи между таблицами.
     - 3.1.9. Работа с данными таблицы.
   - 3.2. Запросы.
     - 3.2.1. Запрос на выборку.
     - 3.2.2. Запрос с параметром.
     - 3.2.3. Вычисления в запросах.
     - 3.2.4. Перекрестный запрос.
     - 3.2.5. Запрос на создание таблицы.
     - 3.2.6. Запрос на обновление.
     - 3.2.7. Запрос на добавление записей.
     - 3.2.8. Запрос на удаление записей.
   - 3.3. Создание формы.
     - 3.3.1. Формы для связанных таблиц.
     - 3.3.2. Встраивание объектов в форму.
     - 3.3.3. Создание элементов формы или отчета.
   - 3.4. Создание отчета.

При создании базы данных в Microsoft Access применяются следующие основные объекты:

– **Таблица** – это объект, предназначенный для хранения данных в виде записей (строк) и полей (столбцов). Обычно каждая таблица используется для хранения сведений по одному конкретному вопросу.

– **Форма** – объект Microsoft Access, предназначенный в основном для ввода данных. В форме можно разместить элементы управления, применяемые для ввода, изображения и изменения данных в полях таблицы.

– **Запрос** – объект, позволяющий получить нужные сведения из одной или нескольких таблиц.

– **Отчет** – объект базы данных Microsoft Access, предназначенный для печати данных.

– **Макросы** – программы, в которых производится такая обработка информации, которая недоступна каждому из перечисленных выше средств по отдельности.

– **Кнопочная форма** – форма особого вида, в которой располагается меню вызова средств, отвечающих на типовые вопросы, возникающие в процессе работы с базой.

Начинать следует с создания таблиц. В режиме таблицы добавляются, редактируются или просматриваются табличные данные. Также можно проверить орфографию и напечатать табличные данные, отфильтровать и отсортировать записи, изменить внешний вид таблицы или изменить структуру таблицы, добавив или удалив столбцы. В режиме конструктора таблицы можно создать целую таблицу, добавляя новые поля или удаляя и настраивая существующие поля таблицы.

С помощью запросов можно просматривать, анализировать и изменять данные из нескольких таблиц. Они также используются в качестве источника данных для форм и отчетов. Наиболее часто используется запрос на выборку. При его выполнении данные, удовлетворяющие условиям отбора, выбираются из одной или нескольких таблиц и выводятся в определенном порядке. Запрос можно создать с помощью мастера или самостоятельно с помощью конструктора.

Формы являются объектом базы данных, которые обычно используются для отображения данных в базе данных. Форма может быть кнопочной, открывающая другую форму или отчеты базы данных. Большинство форм являются присоединенными к одной или нескольким таблицам и запросам из базы данных.

Отчет – это гибкое и эффективное средство для организации данных при выводе на печать. С помощью отчета имеется возможность вывести необходимые сведения в том виде, в котором требуется. С помощью отчетов возможно выполнение следующих действий:

– добавление в отчет эмблемы или рисунка;
– группировка записей по различным условиям;
– вычисление итоговых значений;
– представление данных на диаграмме.

Макросы могут быть полезны для автоматизации часто выполняемых задач. Например, при нажатии пользователем кнопки можно запустить макрос, который распечатает отчет.

1. Интерфейс MS Access

Интерфейс MS Access показан на рисунке 2.1. Главный элемент пользовательского интерфейса MS Access 2016 представляет собой **Ленту**, которая идет вдоль верхней части окна каждого приложения. Лента управления содержит вкладки. По умолчанию их пять: **Файл, Главная, Создание, Внешние данные, Работа с базами данных.** Каждая вкладка связана с видом выполняемого действия.

Панель быстрого доступа. Расположена в верхней части окна Access. По умолчанию на панели быстрого доступа расположены четыре кнопки управления.

Область навигации. Расположена по левому краю окна Access. Она предназначена для отображения объектов или групп объектов открытой базы данных, а также для перехода от объекта к объекту. Чтобы раскрыть группу объектов следует щелкнуть мышкой по кнопке . Управлять объектами можно командами ленты и командами контекстного меню.

Область документов, в которой отображается каждый объект базы данных, открываемый в любом режиме.

Строка состояния. Расположена вдоль нижней границы окна Access, отображает кнопки переключения в различные режимы работы с активным объектом.

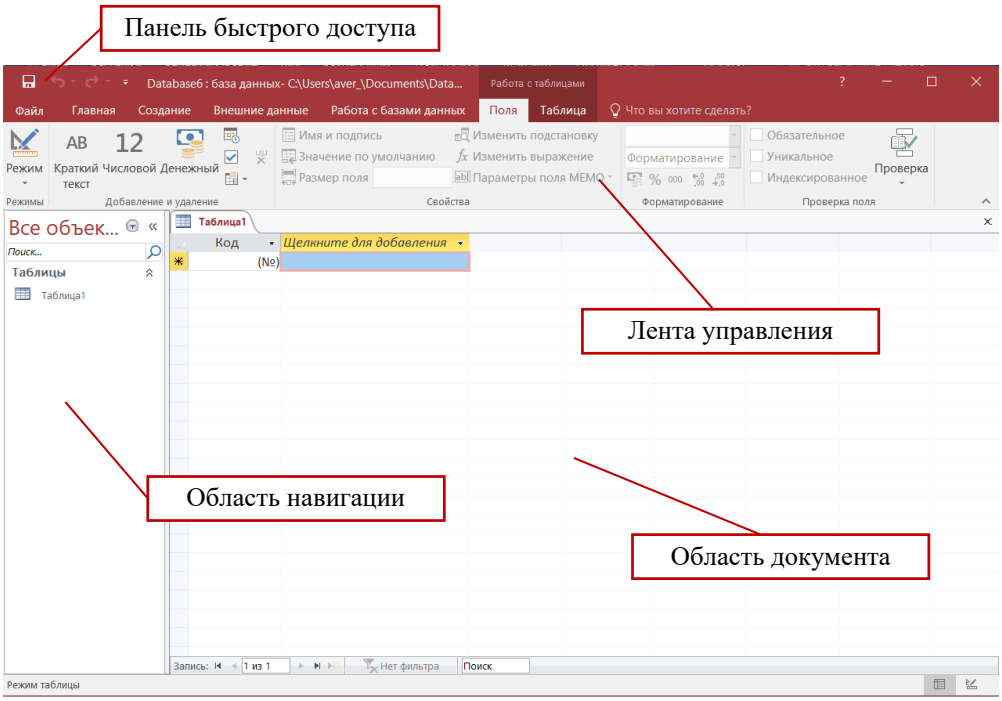

Рисунок 2.1 – Интерфейс программы MS Access

2. Этапы проектирования базы данных

Прежде чем приступить к созданию таких объектов базы данных, как таблицы, формы и отчеты, нужно разработать их проект. Главное назначение проекта – выработка четкого пути, по которому нужно следовать при его реализации. База данных – достаточно сложный объект, и время, затраченное на ее планирование, может значительно сократить сроки ее разработки. Отсутствие продуманной структуры базы данных приводит к необходимости

постоянной переделки и перенастраиванию объектов базы данных, таких, как формы и таблицы.

Проектирование базы данных целесообразно начать с краткого описания отчетов, списков и других документов, которые необходимо получить с помощью БД. Далее следует разработать эскиз объектов, требуемых для получения необходимых результатов и определить связи между этими объектами.

При разработке эскиза необходимо ответить на следующие вопросы:

Какими данными мы располагаем?

Какие данные будут содержать таблицы?

Какой тип и какие свойства должны иметь данные в каждом поле таблицы?

Как эти таблицы будут связаны друг с другом?

Законченный план должен содержать подробное описание всех таблиц (имена полей, типы данных и их свойства), а также связей между ними.

Проектирование предусматривает этапы создания проекта базы данных от концепции до реального воплощения. Этапы проектирования базы данных:

1. **Исследование предметной области и формулировка основных допущений (накладываемых условий).** На этом этапе составляется список всех форм и отчетов, которые могут быть затребованы пользователями вашей БД.

2. **Анализ данных.** Составить перечень всех элементов данных, входящих в формы и отчеты и сгруппировать их в таблицы БД.

3. **Установление взаимосвязей между элементами данных.** Определить первичные и вторичные (внешние) ключи отношений. Организовать поля данных в таблицах, причем это необходимо сделать, следуя 4-м правилам нормализации:

Правило 1: Каждое поле таблицы должно представлять уникальный тип информации. Это правило означает, что необходимо избавиться от повторяющихся полей и разделить составные поля на отдельные элементы данных.

Правило 2: Каждая таблица должна иметь уникальный идентификатор или первичный ключ, который может состоять из одного или нескольких полей.

Правило 3: В таблице не должно быть данных не относящихся к объекту, определяемому первичным ключом.

Правило 4: Независимость полей. Это правило означает возможность изменять значения любого поля (не входящего в первичный ключ) без воздействия на данные других полей. Результатом 3 этапа должна явиться группа таблиц, удовлетворяющих правилам нормализации. На этом же этапе необходимо установить связи между таблицами.

3. Объекты базы данных MS Access

3.1. Таблицы

3.1.1. Типы данных

Информация, хранящаяся в базе данных, может быть самой разной по смыслу и по форме. Access поддерживает следующие типы данных для полей таблиц.

Текстовый. Может содержать строку символов длиной до 255 символов.

Поле Memo. Может содержать произвольный многострочный текст размером до 64000 символов.

Числовой. Данное поле может содержать некоторое числовое значение. Конкретный числовой тип определяется свойством «размер поля». Возможные значения числового типа приведены в табл. 2.1.

Таблица 2.1. – Типы числовых данных

Тип	Размер, байт	Диапазон значений
Байт	1	от 0 до 255
Целое	2	от –32768 до 32767
Длинное целое	4	от –2147483648 до 2147483647
Одинарное с плавающей точкой	4	От $-3.402823\ 10^{38}$ до $-1.401298\ 10^{-45}$ для отрицательных от $1.401298\ 10^{-45}$ до $3.402823\ 10^{38}$ для положительных
Двойное с плавающей точкой	8	от $-1.79769313486231\ 10^{308}$ до $-4.94065645841247\ 10^{-324}$ для отрицательных, от $4.94065645841247\ 10^{-324}$ до $1.79769313486231\ 10^{308}$ для положительных
Код репликации	16	Глобальный уникальный идентификатор (GUID).
Действительное	12	От $-10^{38} + 1$ до $10^{38} - 1$

Поля типа **Действительное** могут хранить целые числа, задаваемые десятичными цифрами. Количество десятичных цифр может устанавливаться свойством **Точность** в пределах от 1 до 28.

Дата/время. Содержит дату и время в диапазоне от 100 до 9999 года.

Денежный. Применяется для значений валют. Предотвращает округления при проведении вычислений. Может иметь до 15 цифр в целой части и до 4 в дробной.

Счётчик. Обеспечивает автоматическую вставку последовательных (увеличивающихся на 1) или случайных чисел при добавлении записи. Гарантируется, что значения счетчика не будут повторяться.

Логический. Содержит только одно из двух значений: **Да/Нет**, **Истина/Ложь**, **Вкл/Выкл.**

Поле объекта OLE. Может содержать двоичные объекты, например, документы MS Word, MS Excel, рисунки, звуки и другие двоичные данные, созданные в программах, использующих протокол OLE (Object Linking and Embedding – связывание и встраивание объектов). Технология OLE обеспечивает обмен данными между различными программами, например, с помощью этой технологии в текстовый документ можно вставить рисунок,

созданный в каком-либо графическом редакторе. Объекты, хранимые в базе данных, могут быть связанными или внедренными. Для отображения объекта OLE на форме или в отчёте необходимо использовать присоединённую рамку объекта.

Гиперссылка. Гиперссылка хранит путь к месту назначения, например, к объекту, документу или Web-странице в Internet

Мастер подстановок. Этот режим не является самостоятельным типом. Его выбор запускает мастер, который позволяет создать список или назначить таблицу или запрос, из которого можно будет выбирать значения для поля. Тип данных устанавливается по значениям, выбранным в процессе работы мастера.

3.1.2. Структура базы данных

Прежде, чем создавать базу данных, нужно продумать её структуру, т.е. решить вопрос о том, какие таблицы с какими полями должны в нее входить.

В качестве примера рассмотрим разработку базы данных «Семестровая успеваемость» для учета семестровой успеваемости студентов СИПИМ.

Краткое описание предметной области

В учебном заведении формируются группы студентов. Студенты в каждом семестре сдают зачеты и экзамены по учебным дисциплинам в соответствии с учебным планом. Преподаватели экзаменуют студентов и выставляют оценки в соответствии с перечнем возможных оценок.

Проанализировав предметную область, устанавливаем, что в базе данных должны быть таблицы: Группа, Дисциплина, Студент и Успеваемость.

Назначим для полей этих таблиц подходящие типы данных (табл. 2.2).

Таблица 2.2 – Структура таблиц БД **Семестровая успеваемость**

Имя поля	Тип поля	Описание
Таблица *Группа*		
Код группы	Счетчик	Уникальный номер
Группа	Текстовый	Название группы
Куратор	Текстовый	ФИО куратора группы
Таблица *Дисциплина*		
Код дисциплины	Счетчик	Уникальный номер
Дисциплина	Текстовый	Название дисциплины
Семестр	Числовой	Номер семестра
Форма контроля	Текстовый	Форма контроля, предусмотренного по дисциплине
Преподаватель	Текстовый	ФИО преподавателя
Таблица *Студент*		
№ зачетной книжки	Текстовый	Уникальный номер зачетной книжки
ФИО студента	Текстовый	ФИО студента
Код группы	Числовой	Код группы (из таблицы Группа)
Дата рождения	Дата и время	Дата рождения студента

Телефон	Текстовый	Телефон студента
Email	Текстовый	Адрес электронной почты студента
Таблица Успеваемость		
Код студента	Числовой	ФИО студента (из таблицы Студент)
Код дисциплины	Числовой	Название дисциплины (из таблицы Дисциплина)
Оценка	Числовой	Оценка
Дата	Дата и время	Дата проведения контроля

При проектировании таблиц необходимо обеспечить наличие *ключей*. Ключ состоит из одного или нескольких полей, которые однозначно идентифицируют каждую запись в таблице. В таблице **Группа** таким полем может служить **Код Группы**, в таблице **Дисциплина – Код Дисциплины**, в таблице **Студент - № зачетной книжки**. В таблице **Успеваемость** ключом могут быть одновременно три поля: **Код студента, Код дисциплины** и **Дата**, один и тот же студент может сдать в определенную день одну и ту же дисциплину один раз. Ключевые поля при описании структур таблиц принято подчеркивать.

3.1.3. Создание таблиц

Для создания новой таблицы необходимо перейти на вкладку **Создание** и в группе **Таблицы** выбрать способ создания новой таблицы.

Создание таблиц в режиме конструктора таблиц

Режим конструктора позволяет полностью контролировать процесс создания таблицы.

На рисунке 2.2 показан бланк конструктора таблиц в процессе создания структуры таблицы **Группа**.

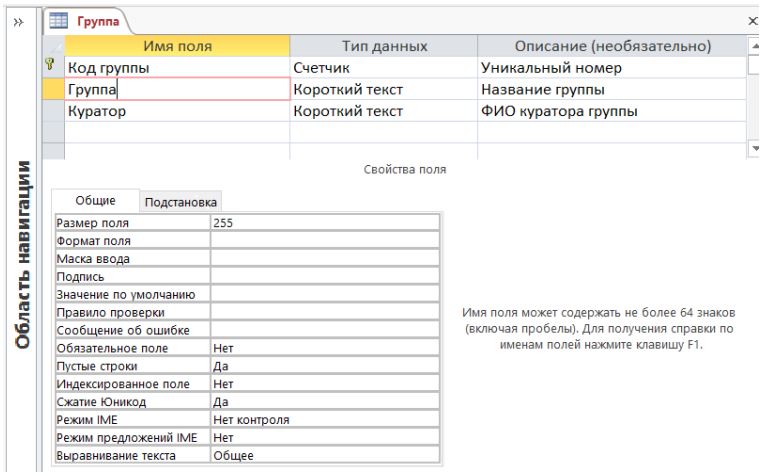

Рисунок 2.2 – Таблица **Группа** в режиме **Конструктора таблиц**.

Новое поле добавляется путем ввода его имени в свободную ячейку столбца **Имя поля**.

Создание таблиц в режиме Таблица

Таблицы можно создавать в режиме **Таблица**, для этого необходимо нажать кнопку **Таблица** в группе **Таблицы** на вкладке **Создание**.

На экране появится пустая таблица, в ячейки которой можно вводить значения. Access по введенному значению определит тип данных и включит его в структуру таблицы. При добавлении нового поля можно выбирать тип данных для этого поля.

3.1.4. Задание ключевого поля

В Microsoft Access можно выделить три типа ключевых полей: *счетчик, простой ключ и составной ключ.* Указание поля счетчика в качестве ключевого является наиболее простым способом создания ключевых полей. Если до сохранения созданной таблицы ключевые поля не были определены, то при сохранении будет выдано сообщение о создании ключевого поля. При нажатии кнопки **Да** будет создано ключевое поле счетчика.

Простой ключ определяется полем, содержащим уникальные значения, такие как СНИЛС или номер зачетной книжки. Ключевое поле не может содержать повторяющиеся или пустые значения. Если устранить повторы путем изменения значений невозможно, то следует либо добавить в таблицу поле счетчика и сделать его ключевым, либо определить составной ключ.

В случаях, когда невозможно гарантировать уникальность значений каждого поля, существует возможность создать **составной ключ**, состоящий из нескольких полей. Чаще всего такая ситуация возникает для таблицы, используемой для связывания двух таблиц в отношении «многие-ко-многим». Если определить подходящий набор полей для составного ключа сложно, следует добавить поле счетчика и сделать его ключевым.

Ключевое поле позволяет однозначно идентифицировать запись, то есть отличить одну запись от другой. По умолчанию при создании таблицы автоматически создается поле **Код**, имеющее тип **Счетчик**, и оно назначается ключевым, о чем говорит изображение ключа слева. При создании таблицы **Группа** заменим название поля **Код** на **Код группы**.

Щелкая по кнопке на вкладке **Конструктор** можно сделать поле ключевым или отменить это назначение.

При вводе данных в ключевые поля осуществляется проверка, чтобы не было совпадения значений в различных записях. По ключевому полю производится автоматическая индексация.

3.1.5. Свойства полей

Свойства полей устанавливаются с помощью вкладок **Общие** и **Подстановка** в нижней половине окна конструктора таблиц (рис.2.2). Рассмотрим эти свойства.

Размер поля. Для текстового поля это максимальное число символов (до 255). По умолчанию устанавливается размер 255 символов. При выборе размера надо учитывать, что в поле размером 20 символов нельзя будет ввести текст длиной 30 символов. С другой стороны, установка слишком большой длины может привести к бесполезному увеличению размера файла базы данных, если значения, хранимые в поле, будут существенно меньше его

установленной длины. Для числовых полей размер выбирается из списка возможных числовых типов данных, которые перечислены в табл. 2.2.

Формат устанавливает вид данных на экране, например, для типа **Дата/Время** можно выбрать представление даты вида *30 июня 2022 г.* или *6/30/22*.

Число десятичных знаков определяет количество цифр после десятичной точки для дробных чисел. Это значение влияет только на вид представления числовых величин, а не их способ хранения в памяти.

Маска ввода позволяет задать шаблон для ввода, который дает некоторую гарантию правильности ввода данных, она применяется для полей типа **Дата/Время** и текстовых. Маска ввода выводит на экран символы – местозаменители, показывает, сколько символов нужно ввести, включает разделительные символы (дефис, скобки). Например, для даты маска ввода может выглядеть так: --.--.--. Данная маска кодируется набором символов 99/99/00. Заполнитель 9 означает, что разрешается вводить только цифры, причём её ввод не является обязательным, заполнитель 0 требует обязательного ввода цифры. При выборе свойства **Маска ввода** справа появляется кнопка ..., по которой запускается мастер, помогающий создать маску.

Подпись используется в качестве заголовка столбца в режиме таблицы.

Значение по умолчанию позволяет автоматически вводить в поле какое-либо значение.

Свойство **Обязательное поле** имеет два значения **Да** и **Нет**. Если установить значение **Да**, Access потребует обязательного ввода какого-либо значения в данное поле.

Свойство **Индексированное поле** определяет, будет или нет проводиться индексация по данному полю. Индексация состоит в создании списка номеров записей, упорядоченных в соответствии со значениями поля. Наличие индекса ускоряет операции поиска и сортировки, но требует дополнительного места на диске.

3.1.6. Создание подстановок с помощью мастера подстановок

Сделать более простым ввод данных в таблицу позволяет **мастер подстановки**. Например, обеспечим возможность ввода значения поля **Форма контроля** таблицы **Дисциплина** путем выбора нужного вида контроля из списка. Перейдем в режим конструктора для таблицы **Дисциплина** и выберем для типа данных поля **Форма контроля** значение **Мастер подстановок…** (рис.2.3).

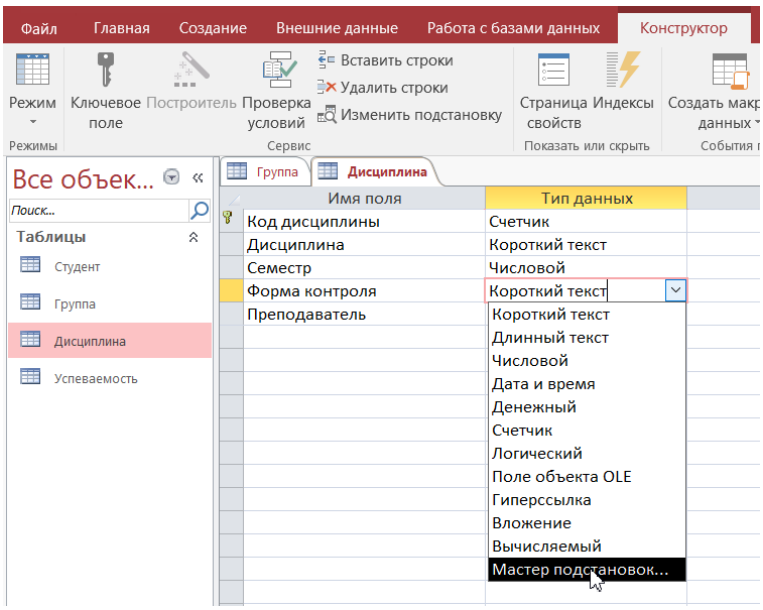

Рисунок 2.3 – Выбор режима **Мастер подстановок**

При использовании режима **Мастер подстановок** предъявляется несколько диалоговых окон. Первый шаг мастера показано на рисунке 2.4.

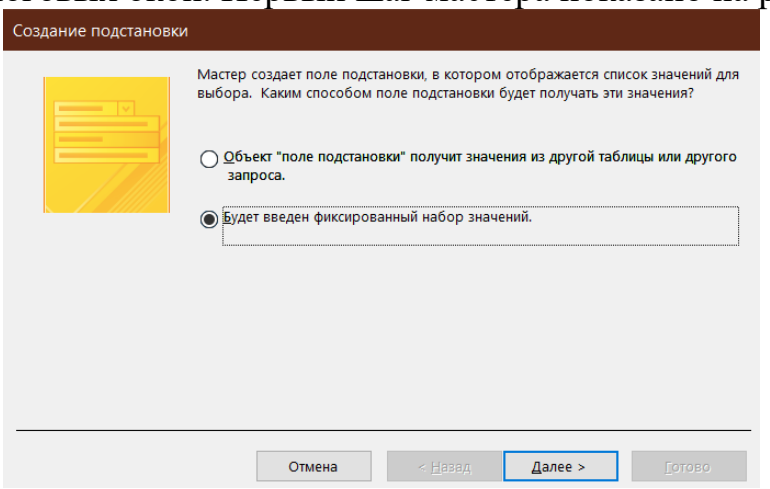

Рисунок 2.4 – Мастер подстановок. Шаг 1. Выбор источника значений

Выберем способ **Будет введен фиксированный набор значений**. На втором шаге (рис.2.5) вводим виды контроля знаний в столбец.

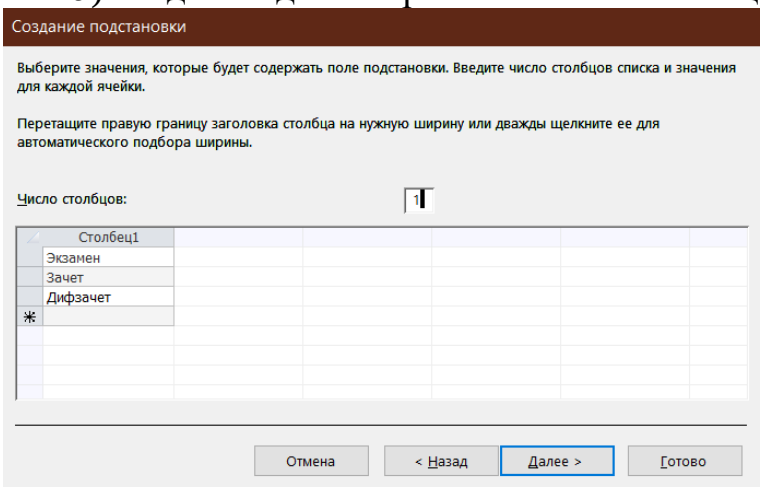

Рисунок 2.5 – Мастер подстановок. Шаг 2. Ввод возможных значений

На третьем шаге мастера подстановок задаем название столбца подстановки (рис.2.6).

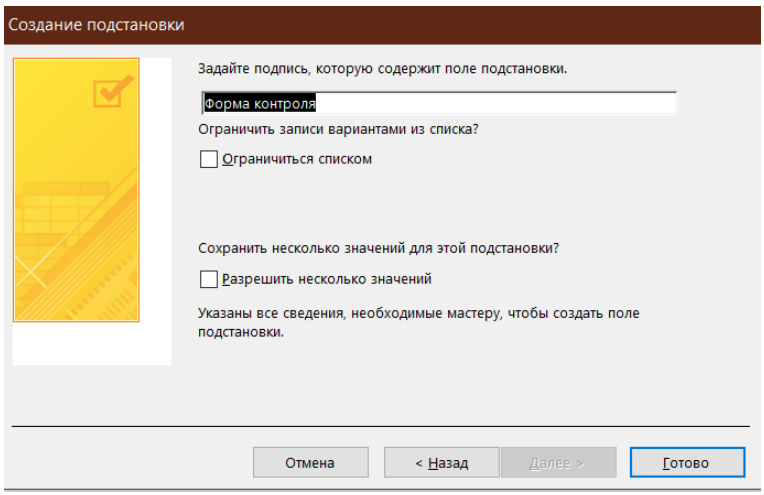

Рисунок 2.6 – Мастер подстановок. Шаг 3. Ввод подписи

В результате поле **Форма контроля** будет преобразовано в поле со списком. Это можно увидеть на вкладке **Подстановка** раздела **Свойства поля** (рис. 2.7).

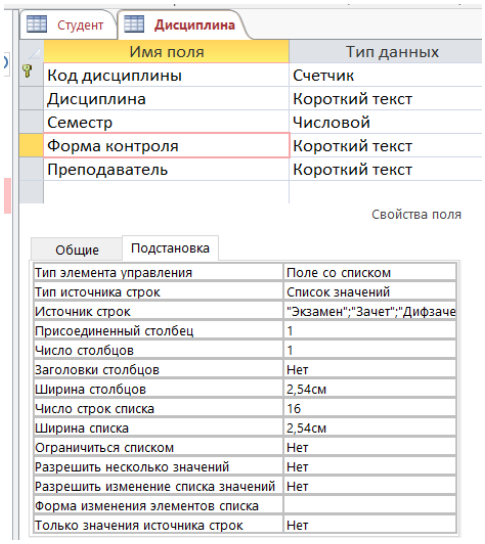

Рисунок 2.7 – Вкладка **Подстановка** поля **Форма контроля**

Таблица с использованием столбца подстановки показана на рисунке 2.8. Надпись, введенная в третьем ДО (рис.2.6), совпадает с названием поля таблицы, поэтому название поля не изменилось.

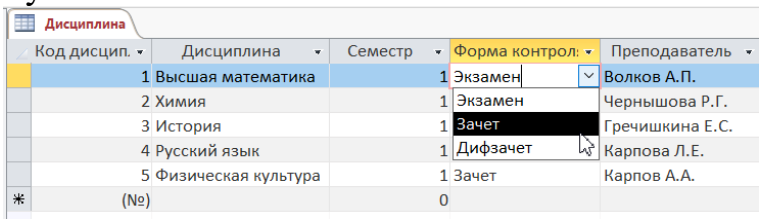

Рисунок 2.8 – Использование столбца подстановок в режиме таблицы

В рассматриваемой в качестве примера БД успеваемости студентов СИПИМ необходимо преобразовать в Поле со списком такие поля:

– **Код группы** в таблице **Студент**;
– **Код студента** в таблице **Успеваемость**;
– **Код дисциплины** в таблице **Успеваемость**;

Рассмотрим, например, создание подстановки в поле **Код группы** в таблице **Студент**. Для этого перейдем в режим конструктора для таблицы **Студент** и выберем для типа данных поля **Код группы Мастер подстановок**.

На первом шаге выберем **Объект «столбец подстановок» будет использовать значения из таблицы или запроса.** На втором шаге выберем таблицу Группа (рис. 2.9).

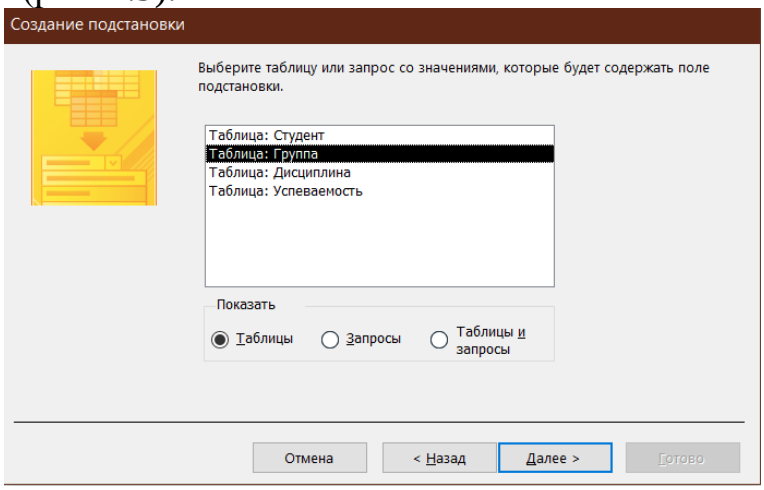

Рисунок 2.9 Мастер подстановок. Шаг 2. Выбор таблицы Группа.

На третьем шаге выбираем поля для подстановки **Код группы** и **Группа** (рис. 2.10).

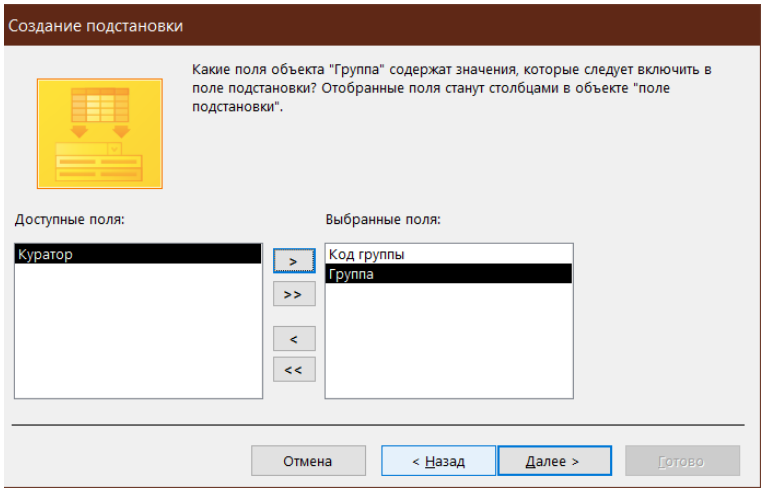

Рисунок 2.10 – Мастер подстановок. Шаг 3. Выбор полей таблицы **Группа**

На четвертом шаге при необходимости можно задать порядок сортировки элементов создаваемого списка. На пятом шаге задаем ширину столбцов (рис. 2.11). Обратите внимание, что мастер рекомендует скрыть ключевой столбец.

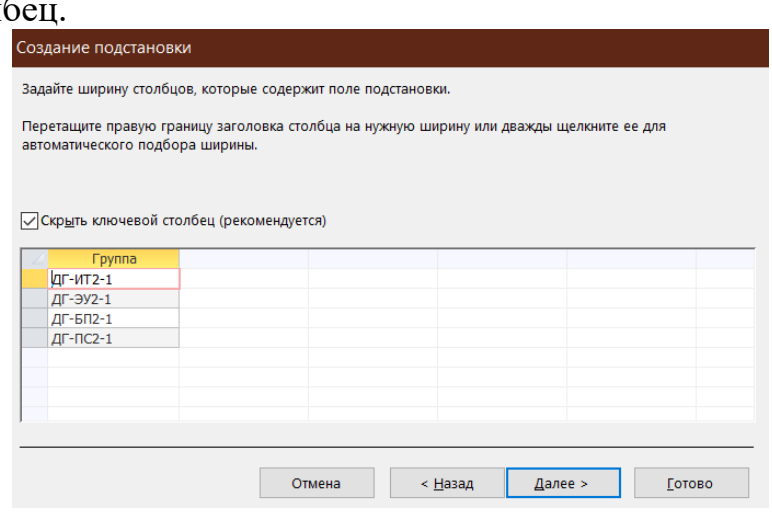

Рисунок 2.11 – Мастер подстановок. Шаг 5. Задание ширины столбцов.

На следующем шаге можно задать подпись, которую содержит столбец подстановки. После нажатия на кнопку **Готово** появляется извещение Мастера подстановок о том, что будет создана связь между таблицами.

Таким образом, с помощью Мастера подстановок создана связь между таблицами **Студент** и **Группа** по полю **Код группы**, а также поле со списком **Код группы** в таблице **Студент** свойства которого можно увидеть на вкладке **Подстановка** раздела **Свойства поля** (рис. 2.12).

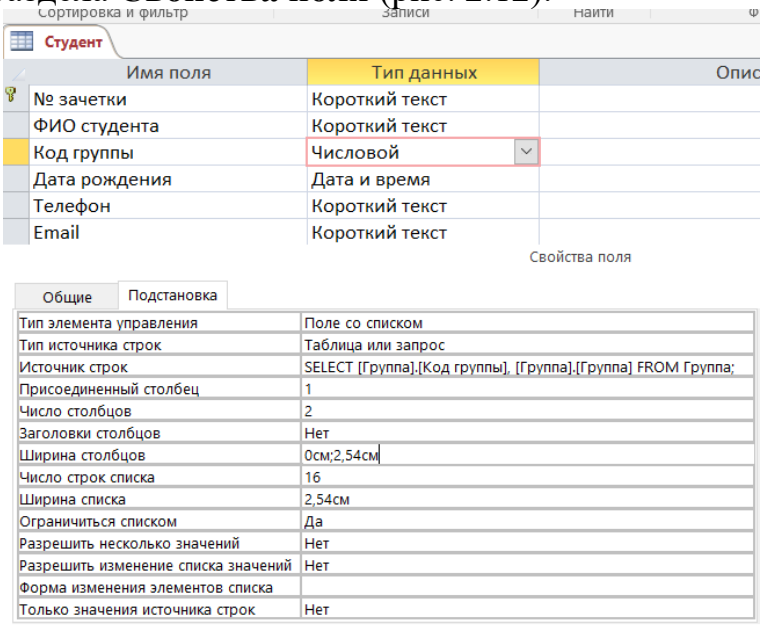

Рисунок 2.12 – Вкладка **Подстановка** поля **Код группы**

3.1.7. Создание маски ввода

Для поля *Телефон* таблицы *Студент* выберем свойство **Маска ввода** (рис.2.13).

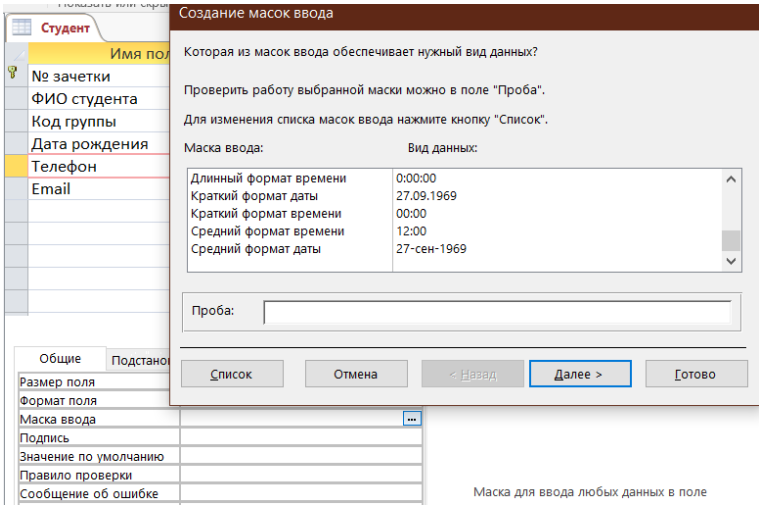

Рисунок 2.13 – Создание маски ввода

Так как в предустановленных масках ввода нет маски для ввода номера телефона необходимо нажать на кнопку **Список** для перехода в окно **Настройка масок ввода** (рис.2.14), где можно самостоятельно создать маску ввода.

В поле **Описание** зададим маске ввода название.

В поле **Маска ввода** приведены символы, кодирующие маску ввода. Цифра ноль (0) разрешает ввод в соответствующей позиции только цифры. То

есть десять нулей разрешают ввод трех цифр для кода и семи цифр для номера телефона.

В поле **Заполнитель** указывается символ, который будет отображаться в поле таблицы при его заполнении.

В поле **Образцы данных** – вводится образец номера телефона.

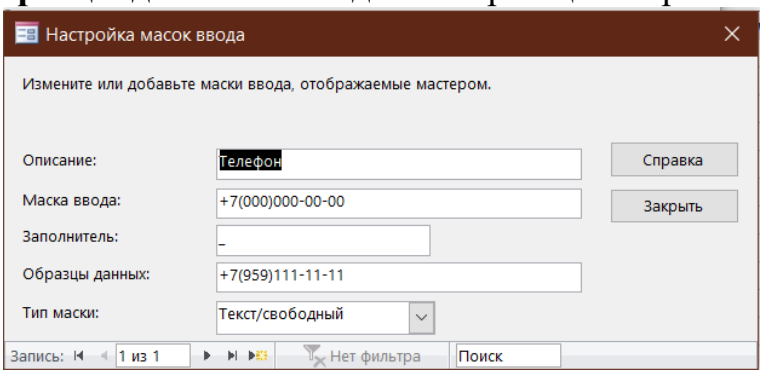

Рисунок 2.14 – Создание маски ввода

Пример поля таблицы показан в поле **Проба** на рисунке 2.15.

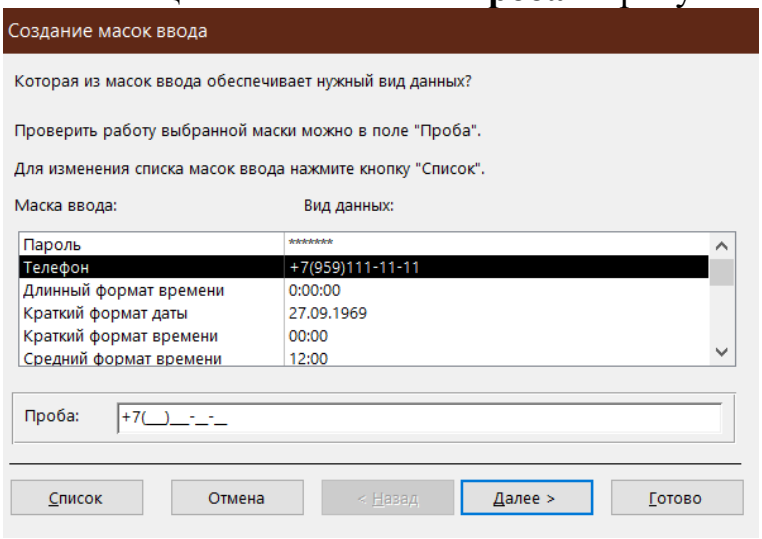

Рисунок 2.15 – Проба маски ввода

3.1.8. Связи между таблицами

Если между таблицами установлены связи, возможен контроль за данными, хранящимися в базе.

Связь возможна между таблицами, у которых имеются поля с *одинаковыми значениями*. Установка связей между таблицами позволяет обеспечить *целостность данных*. Для установления связей между таблицами:

1. Выберите вкладку **Работа с базами данных.**
2. В группе **Отношения** выберите **Схема данных.**
3. В появившемся диалоговом окне **Добавление таблицы** выберите таблицы, которые должны быть связаны. Названия каждой из таблиц со списками полей появятся в окне **Схема данных.**
4. Установите курсор в любую из таблиц на поле, по которому будет установлена связь и «перетащите» это поле на связующее поле другой таблицы. Тип данных[2], значения и свойства связываемых полей должны совпадать.

[2] Для поля Счетчик связующее поле может иметь числовой тип данных.

5. Активизируйте флажок **Обеспечение целостности данных.** Если установить флажок **Каскадное обновление связанных полей,** то при изменении ключевого поля главной таблицы автоматически будут изменяться и соответствующие значения связанных записей. Если установить флажок **Каскадное удаление связанных полей,** то при удалении записи в главной таблице будут удалены и все связанные записи в подчиненной таблице.

От полей, указанных при определении связи зависит тип создаваемой связи, который отображается в этом же окне.

Отношение **«один-к-одному»** создается в том случае, когда оба связываемых поля являются ключевыми или имеют уникальные индексы[3].

Отношение **«один-ко-многим»** создается в том случае, когда только одно из полей является ключевым или имеет уникальный индекс. В отношении «один-ко-многим» *главной таблицей* является таблица, которая содержит первичный ключ и составляет часть «один» в этом отношении. Таблица со стороны «много» является *подчиненной таблицей.* Связующее поле (или поля) в ней с таким же типом информации как в первичном ключе главной таблицы является полем *внешнего ключа.* Связь на схеме данных отображается в виде соединительной линии со специальными значками около таблиц: «1» вблизи главной таблицы, имеющей первичный ключ, «∞» вблизи подчиненной таблицы, имеющей внешний ключ.

Связь с отношением **«многие-ко-многим»** фактически представляет две связи с отношением «один-ко-многим» через третью таблицу, ключ которой состоит, по крайней мере, из двух полей, которые являются полями первичного ключа в двух других таблицах.

В случае если для какой-то из таблиц не было определено ключевое поле, то в поле Тип отношения отображается текст: «Не определено».

Для удаления или изменения связи в окне **Схема данных** выберите нужное действие в контекстном меню**.**

В рассматриваемой в примере БД **Семестровая успеваемость** использованы отношения «один-ко-многим» (рис.2.16).

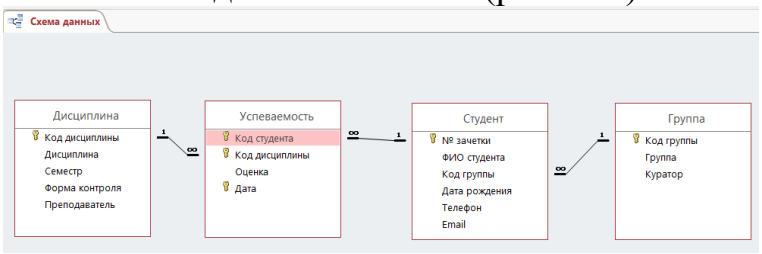

Рисунок 2.16 – Схема данных

3.1.9. Работа с данными таблицы

Ввод записей

Ввод записей выполняется в режиме работы с таблицами. Переход к табличному представлению БД осуществляется с помощью вкладка **Главная** – группа **Режимы** – кнопка **Режим**.

[3] Уникальный индекс - индекс, определенный для свойства Индексированное поле значением «Да (Совпадения не допускаются)». При этом ввод в индексированное поле повторяющихся значений становится невозможным. Для ключевых полей уникальный индекс создается автоматически.

Переход на нужное поле или запись

Для перехода между столбцами и к следующей записи используется клавиша **Tab** или комбинация клавиш **Shift+Tab.**

Для перехода между записями также служат кнопки переходов в нижнем левом углу окна, где также отображается общее количество записей и номер текущей записи (рис. 2.17).

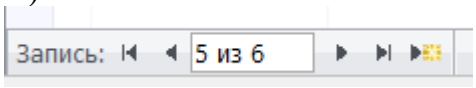

Рисунок 2.17 – Кнопки переходов между записями

Для перехода к конкретной записи вместо номера текущей записи нужно ввести требуемый номер и нажать клавишу **Enter.**

Переход к другой записи также может быть осуществлен с помощью команды **Перейти** (вкладка **Главная** – группа **Найти**).

Быстрый путь ввода данных

Копирование данных из предыдущей записи.

Для копирования данных из аналогичного поля предыдущей записи в текущую надо нажать **<Ctrl>+"** (кавычки).

Вставка текущего времени или даты.

Чтобы вставить текущую дату надо нажать **<Ctrl>+Ж.**

Чтобы вставить текущее время надо нажать **<Ctrl>+<Shift>+Ж.**

Сохранение данных

В MS Access изменения сохраняются автоматически при следующих действиях:

– переход к следующей записи;

– закрытие режима таблицы или формы.

Вставка в запись рисунка или объекта

Рисунок или объект добавляется из имеющегося файла либо создается в приложении OLE (например, в MS Paint), а затем вставляется в текущую запись.

Чтобы добавить рисунок или любой другой объект в запись:

1. Перейдите в режим Конструктора таблиц.
2. Добавьте поле объекта OLE.
3. В режиме Таблицы установите курсор в нужную клетку, правой клавишей вызовите контекстное меню и выполните команду **Вставить объект.**

Если объект вставляется из существующего файла:

1. В появившемся окне выберите переключатель **Создать из файла.**
2. Введите полное имя добавляемого файла в поле **Файл** или нажмите кнопку **Обзор** и выберите имя требуемого файла.
3. Нажмите кнопку **ОК**.

Если объект нужно создать:

1. Выберите тип создаваемого объекта в поле **Тип объекта** (например, Bitmap Image).
2. Нажмите кнопку **ОК**.

3. После создания рисунка или объекта в приложении OLE выполните команду **Выход** из приложения OLE.

Фильтрация данных

Фильтрация – удобный способ отображения нужных данных. Фильтры позволяют просмотреть только отдельные записи в форме, отчете, запросе или таблице либо напечатать некоторые записи из отчета, таблицы или запроса. С помощью фильтра можно ограничить объем отображаемых данных, не изменяя макет базовых объектов.

Так как после применения фильтра представление содержит только записи с выбранными значениями, остальные записи скрываются до очистки фильтра.

Для столбцов таблиц и элементов управления в формах и отчетах, связанных с выражениями, фильтрация не поддерживается.

Существует несколько типов фильтров, и некоторые из них очень легко применять и удалять. Обычные фильтры встроены в каждое представление MS Access. Доступность команд фильтра зависит от типа и значений поля.

Для каждого типа данных предусмотрено несколько готовых фильтров. Они доступны в виде команд меню в режимах таблицы и макета и в представлениях формы и отчета. Таблицу или форму можно отфильтровать не только с помощью этих фильтров, но и путем заполнения формы (фильтр по форме).

Пользователь, который может уверенно написать выражение самостоятельно, может добиться большей гибкости, создав собственные фильтры с помощью параметров вкладки документа Фильтр.

Ниже описаны доступные типы фильтров.

Обычные фильтры: используются для фильтрации по значению или диапазону значений.

Фильтрация по выделенному: позволяет отсортировать все строки в таблице, содержащие значение, которое совпадает с выделенным значением в строке. Используется в режиме таблицы.

Фильтр по форме: используется, если требуется отфильтровать несколько полей в форме или таблице либо найти конкретную запись.

Расширенный фильтр: позволяет задать пользовательские условия фильтра.

3.2. Запросы

Использование запросов позволяет осуществлять различные формы доступа к одной и той же информации. Запрос – это объект БД, допускающий многократное использование. Результат запроса – представленный в табличном виде набор данных. Запросы могут быть созданы как с помощью Мастера запросов, так и самостоятельно, с помощью Конструктора запросов.

Для создания нового запроса:

1. На вкладке **Создание** в группе **Запросы** выберите кнопку **Конструктор запросов.**

2. В диалоговом окне **Добавление таблицы** укажите имена таблиц, по полям которых будет производиться запрос, нажимая кнопку **Добавить** после каждого указанного имени таблицы.

3. Нажмите кнопку **Закрыть**.

В специальном бланке запроса указываются условия отбора выводимых на экран полей и записей одной или нескольких таблиц и порядок их отображения. В бланке запроса содержится 6 строк (табл. 2.3).

Таблица 2.3 – Бланк запроса

Поле	**Имя поля**
Имя таблицы	Имя таблицы
Сортировка	Место ввода инструкций сортировки
Вывод на экран	Определяет, будет ли отображено поле в результирующем наборе данных
Условие отбора	Содержит первое условие, ограничивающее набор записей
ИЛИ	Другие условия на ограничения набора записей

MS Access позволяет выполнять следующие типы запросов:

1. **QBE-запросы** (QBE=Query By Example - Запросы по образцу):

- запрос на выборку;
- перекрестный запрос;
- запрос на создание таблицы;
- запрос на обновление;
- запрос на добавление записей;
- запрос на удаление записей.

Каждый из этих типов указывается в дополнительной вкладке **Конструктор** группа **Тип запроса.**

2. **Запросы SQL** (Structured Query Language – структурированный язык запросов). SQL – стандартизированная форма составления запросов для обработки реляционных баз данных. При выполнении QBE-запросов они транслируются в соответствующие SQL-запросы.

3.2.1. Запрос на выборку

Запрос на выборку является самым распространенным типом запроса. Данный запрос определяет, какие записи или поля из одной или нескольких таблиц будут отображены при его выполнении (рис. 2.18).

Для выбора записей, удовлетворяющих определенным критериям:

1. В строке **Поле** щелкните в правой части поля на стрелке, указывающей вниз и выберите имя поля, по которому будет осуществляться запрос. Если запрос осуществляется по полям из разных таблиц, то сначала щелкните в строке **Таблица** и укажите нужную таблицу, что позволит ограничить список полей в строке **Поле**. Если запрос будет осуществляться по нескольким полям, отобразите их имена в свободных клетках строки **Поле**.

2. Проследите, чтобы в строке **Вывод на экран** флажок отображался бы галочкой.

3. В строке **Условие отбора** введите критерии выбора. (Для задания диапазона значений могут быть использованы операторы: > (больше), >= (больше или равно), < (меньше), <= (меньше или равно) и Between (между) Выражение 1 and Выражение 2 как с текстовыми и числовыми полями, так и с полями дат). Для ввода условия выборки можно использовать окно **Построитель выражений** (группа **Настройка запроса** кнопка **Построитель**).

4. Если это нужно, сохраните запрос для последующего использования.

Для выполнения запроса нажмите кнопку с восклицательным знаком **Выполнить** группа **Результаты.**

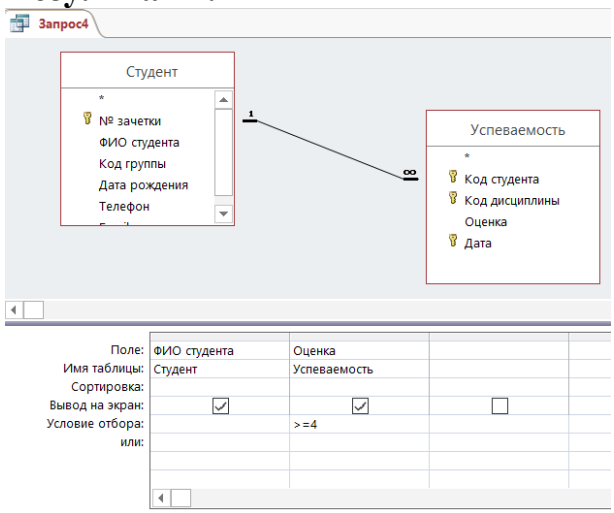

Рисунок 2.18 – Запрос на выборку

3.2.2. Запрос с параметром

Как правило, запросы с параметром создаются в тех случаях, когда предполагается выполнять этот запрос многократно, изменяя лишь условия отбора. В отличие от запроса на выборку, где для каждого условия отбора создается свой запрос и все эти запросы хранятся в БД, параметрический запрос позволяет создать и хранить один единственный запрос и вводить условие отбора (значение параметра) при запуске этого запроса, каждый раз получая новый результат. В качестве параметра может быть любой текст, смысл которого определяет значение данных, которые будут выведены в запросе. Значение параметра задается в специальном диалоговом окне. В случае, когда значение выводимых данных должно быть больше или меньше указываемого значения параметра, в поле **Условие отбора** бланка запроса перед параметром, заключенным в квадратные скобки ставится соответствующий знак. Можно также создавать запрос с несколькими параметрами, которые связываются друг с другом логическими операциями "И" и "ИЛИ". В момент запуска запроса на выполнение MS Access отобразит на экране диалоговое окно для каждого из параметров. Помимо определения параметра в бланке запроса, необходимо указать с помощью кнопки **Параметры** (группа **Показать или скрыть**) соответствующий ему тип данных.

1. Откройте в режиме **Конструктора** окно запроса и добавьте в него таблицу. Создайте запрос, «перетащив» необходимые поля в бланк запроса и задав условие выбора.

2. В качестве условия введите параметр, заключенный в квадратные скобки (например, [Введите название группы]).

3. Выберите команду **Параметры**.

4. В появившемся окне **Параметры запроса** введите без квадратных скобок параметр (для точности ввода воспользуйтесь «быстрыми» клавишами копирования и вставки из буфера обмена) и укажите соответствующий ему тип данных. Нажмите **ОК.**

5. Нажмите кнопку **Выполнить** (группа **Результаты**).

6. В появившемся окне укажите значение параметра.

7. Результат запроса будет содержать только те записи, которые удовлетворяют заданному значению параметра.

Пример запроса с параметром представлен на рисунке 2.19.

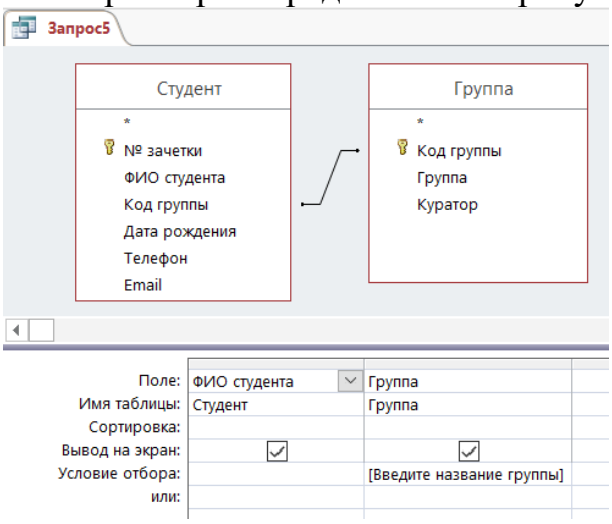

Рисунок 2.19 – Запрос с параметром

3.2.3. Вычисления в запросах

Запрос можно использовать для выполнения расчетов и подведения итогов из исходных таблиц.

Для создания вычисляемых полей используются математические и строковые операторы. При этом Access проверяет синтаксис выражения и автоматически вставляет следующие символы:

– квадратные скобки [], в них заключаются имена элементов управления; знаки номеров (#), в них заключаются распознанные даты;

– кавычки (""), в них заключается текст, не содержащий пробелов или знаков пунктуации.

Выражения, определяемые пользователем, дают возможность выполнять действия с числами, датами и текстовыми значениями в каждой записи с использованием данных из одного или нескольких полей. Например, обычное выражение позволяет найти разность значений двух полей типа даты, соединять несколько строковых значений в текстовом поле или умножить значения одного поля на итоговое значение.

Поле, содержимое которого является результатом расчета по содержимому других полей, называется **вычисляемым полем**. **Вычисляемое поле** существует только в результирующей таблице. Общий формат вычисляемого поля выглядит так:

Имя вычисляемого поля: ***Выражение для создания вычисляемого поля.***

Например**, Цена со скидкой:[Цена]*0,9**

Для расчетов с использованием формул, определяемых пользователем, требуется создать новое вычисляемое поле прямо в бланке запроса путем простого ввода выражения для вычисления в ячейку **Поле** пустого столбца бланка запроса.

После выполнения запроса вычисляемое поле, основанное на этом выражении, выводит на экран результат вычислений, а не само выражение.

1. В строку **Поле** пустого столбца бланка запроса введите выражение, начинающееся со знака «=» и состоящее из имен полей, записанных в квадратные скобки и какой-либо арифметической или другой операции.
2. После выполнения запроса в результирующей таблице появится новое поле с названием «Выражение 1», используемым в качестве имени вычисления выражения.
3. В режиме конструктора запроса измените имя «Выражение 1» на более значимое.

Для того чтобы ввести сложные вычисления используйте **Построитель выражения**, которое вызывается нажатием кнопки **Построитель** (группа **Настройка запроса**). Построитель выражений облегчает создание выражений, позволяя выбирать его составляющие элементы (арифметические операции, встроенные функции, названия полей имеющихся в БД таблиц и запросов и т.п.) при помощи кнопок и списков.

Результаты вычислений также могут быть использованы в условиях отбора для определения записей, которые выбираются в запросе, или для определения записей, над которыми производятся какие-либо действия.

Запросы позволяют производить **итоговые вычисления**. Для этих целей в Access предусмотрены **статистические функции SQL.** Статистическую функцию задают в строке **Групповая операция** бланка запросов, которая появляется при нажатии кнопки 

(группа **Показать или скрыть**). Заполняя ячейки в строке **Групповая операция**, можно выполнить расчеты для групп записей и вычислить сумму, среднее, количество или другой тип итогового значения для вычисляемого поля (табл. 2.4).

Таблица 2.4 – Функции в строке Групповая операция

Функция SQL	**Действие**
Sum	Суммирование значений определенного поля
Avg	Вычисление среднего значения данных определенного поля
Min	Вычисление минимального значения поля
Max	Вычисление максимального значения поля

Count	Вычисление количества записей, отобранных запросом по условию
First	Определяется первое значение в указанном поле записей, отобранных запросом
Last	Определяется последнее значение в указанном поле записей, отобранных запросом
StDev	Вычисляется стандартное отклонение значений данного поля, для всех записей, отобранных запросом
Var	Вычисляется вариация значений данного поля для всех записей, отобранных запросом

Для выполнения запроса на итоговое вычисление:

1. Находясь в режиме Конструктора Запроса, выберите команду **Итоги** (группа **Показать или скрыть**). В результате чего в бланке запроса появится строка **Групповая операция.**

2. Для соответствующего поля выберите нужную функцию из списка. Например, вычислим средний балл успеваемости студентов (рис. 2.20).

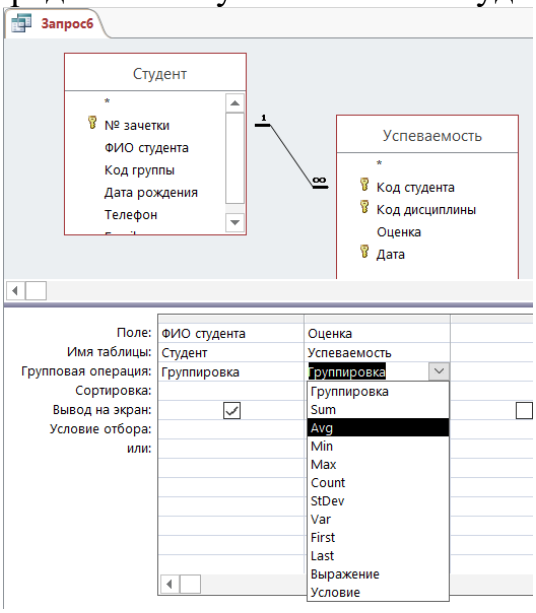

Рисунок 2.20 – Создание запроса «Вычисляемые поля»

3.2.4. Перекрестный запрос

Перекрестный запрос применяется в том случае, если необходимо объединить данные в формате строк-столбцов. В качестве заголовков для столбцов при проектировании таких запросов можно указать значения некоторых полей или выражений:

1. В режиме **Конструктора** сформируйте запрос, добавив таблицу, которая должна лежать в его основе.

2. Выберите команду **Перекрестный** (группа **Тип запроса**). Строка запроса **Вывод на экран** в бланке запроса изменится на новую строку **Перекрестная таблица** и перед ней появится строка **Групповая операция.**

3. В строке **Поле** укажите поле, значения которого в новой таблице должны появиться в виде строк; поле, значения которого в новой таблице должны появиться в виде столбцов и поле, содержимое которого в

перекрестной таблице необходимо индицировать в качестве значения. Полей, которые будут использованы в качестве заголовков, может быть несколько.

4. Щелкните мышью в строке **Перекрестная таблица** и выберите соответствующие значениям данных полей опции из разворачивающегося списка.

5. Для поля, содержимое которого индицируется в качестве значений, в строке Групповая операция введите необходимую функцию, например, автосуммирования (Sum), определения среднего значения (Avg) или количества (Count).

На основе данных перекрестного запроса можно строить диаграммы, представленные в виде формы.

3.2.5. Запрос на создание таблицы

БД на физическом уровне хранит только таблицы. Набор записей запросов физически не существует в БД. Access создает его из данных таблиц только во время выполнения запроса. Иногда возникает необходимость сохранить извлекаемые с помощью запроса на выборку данные в новой таблице:

1. Создайте новый запрос на выборку и проверьте его корректность, перейдя в режим **Таблица**. Для создания резервной копии таблицы (таблицы, содержащей те же поля и в том же количестве, что и в оригинале), чтобы не перетаскивать все поля таблицы в строку **Поле**, достаточно поместить туда из начала списка полей таблицы символ *, заменяющий все поля таблицы.

2. Преобразуйте запрос на выборку в запрос на создание новой таблицы.

Для этого, в группе **Тип запроса**, выберите команду **Создание таблицы.**

3. В появившемся окне введите имя новой таблицы и нажмите **ОК.**

4. Выполните запрос.

3.2.6. Запрос на обновление

Используя этот тип запроса, можно изменить в базовой таблице группу блоков данных, отобранную на основе определенных критериев:

1. Создайте новый запрос на выборку и проверьте его корректность, перейдя в режим **Таблица**.

2. Преобразуйте запрос на выборку в запрос на обновление. Для этого, вернувшись в режим **Конструктора**, выберите команду **Обновление** (группа **Тип запроса**).

3. В появившейся в бланке запроса строке **Обновление** в соответствующих столбцах задайте новые значения полей таблицы. В качестве таковых могут выступать и вычисляемые значения. В случае необходимости воспользуйтесь **Построителем выражений**.

4. Выполните запрос.

3.2.7. Запрос на добавление записей

С помощью этого типа запроса блоки данных одной таблицы (все или отобранные запросом) можно присоединить в конец другой таблицы:

1. Создайте новый запрос на выборку тех блоков данных, которые будут добавлены в некоторую таблицу и проверьте его корректность, перейдя в режим **Таблица**.

2. Преобразуйте запрос на выборку в запрос на добавление. Для этого, вернувшись в режим **Конструктора**, выберите команду **Добавление** (группа **Тип запроса**).

3. В появившемся окне введите имя таблицы, к которой нужно присоединить данные и нажмите **ОК**.

4. Выполните запрос.

3.2.8. Запрос на удаление записей

С помощью данного типа запроса можно удалить из базовой таблицы группу блоков данных, отобранных по определенным критериям. При этом следует тщательно проанализировать критерии отбора, поскольку эту операцию нельзя отменить:

1. Создайте новый запрос на выборку удаляемых блоков данных. Отбор блоков данных выполняется в соответствии с заданными в строке **Условие** критериями.

2. Проверьте корректность сформулированных условий, перейдя в режим **Таблица**.

3. Преобразуйте запрос на выборку в запрос на удаление записей. Для этого, вернувшись в режим **Конструктора**, выберите команду **Удаление** (группа **Тип запроса**).

4. В появившейся строке **Удалить** установите критерии отбора.

5. Выполните запрос.

3.3. Создание формы

Формы Access позволяют создавать пользовательский интерфейс для таблиц базы данных. Хотя для выполнения тех же самых функций можно использовать режим таблицы, формы предоставляют преимущества для представления данных в упорядоченном и привлекательном виде.

Форма представляет собой некий электронный бланк, в котором имеются поля для ввода данных. В Форме каждое поле можно разместить в точно заданном месте, выбрать для него цвет и заливку. В Форму можно помещать вычисляемые поля. OLE-обьекты можно увидеть только в форме или отчете. В Форме намного проще работать с большими текстами поля типа MEMO в текстовом окне с полосами прокрутки.

Форма строится на основе таблицы или запроса. При каждом открытии сохраненной формы обновляются данные запроса, на основе которого создается форма. Благодаря этому содержимое Формы всегда соответствует информации в таблицах и запросах.

Формы могут быть выведены на экран в трех видах: режим формы, режим макета и режим конструктора. Для перехода из одного режима в другой используются команды группы **Режимы**.

Microsoft Access предоставляет быстрый способ создания формы на основе таблицы с использованием **Мастера Форм**. Он задает пользователю вопросы о структуре и оформлении формы. Результатом диалога пользователя

и **Мастера Форм** является «готовая к использованию» форма. Для создания формы самостоятельно без помощи **Мастера Форм**:

1. В области навигации выберите таблицу, по которой будет создаваться форма.
2. На вкладке **Создание** в группе **Формы** выберите команду **Форма**.
3. Выберите режим **Конструктор**. При открытии окно конструктора содержит три области: заголовок формы, область данных, примечание формы.

Поля, размещенные в области данных, состоят из надписи поля и поля для ввода данных. Если выделить надпись или само поле, то ко второму элементу автоматически добавляется манипулятор перемещения и можно перемещать их в паре или по отдельности. В случае, когда нет необходимости в выводе надписи поля рядом с самим полем, удалить ее можно следующим образом: выделить объект Надпись и нажать клавишу **Delete**.

3.3.1. Формы для связанных таблиц

В таких формах можно одновременно отобразить информацию из двух (или более) связанных таблиц. Кроме того, такая форма позволяет выполнить редактирование данных, содержащихся в обеих таблицах.

В результате создания этой формы на экране выводятся только те записи подчиненной таблицы, которые связаны с текущей записью исходной (главной) таблицы:

1. Выберите команду **Мастер форм** (вкладка **Создание** группа **Формы**).
2. В появившемся диалоговом окне укажите имена полей для главной и подчиненной форм и порядок их размещения в новой форме, выбрав имя таблицы из раскрывающегося списка Таблицы/Запросы. Нажмите кнопку **ДАЛЕЕ**.
3. В следующем окне выберите переключатель Подчиненные формы.
4. Далее выберите вид подчиненной формы.
5. Озаглавьте главную и подчиненную формы и нажмите кнопку Готово.

Для просмотра записей главной формы используются кнопки просмотра в нижней части окна. Выше нее выводится строка для просмотра записей подчиненной формы, которые представлены в виде таблицы.

3.3.2. Встраивание объектов в форму

Под объектами будем понимать все то, что создано не в среде Access, а в других Windows-приложениях. СУБД Access может работать с такими объектами, как, например, рисунок Paint, рисунок MS Word, документ MS Word, лист MS Excel, диаграмма MS Excel, музыкальные файлы и видеоклипы и т.д. Программа, в среде которой создан объект, называется родительской программой. Под **встраиванием** объекта понимается использование его в составе таблиц, форм и отчетов. При двойном щелчке мышью на встроенном объекте вызывается родительская программа с загруженным объектом, который можно редактировать. Процесс встраивания объектов базируется на механизме OLE (Object Linking and Embedding).

При встраивании объекта можно применить один из двух способов: внедрение и связывание.

При **внедрении** объект хранится в файле базы данных. При **связывании** объект хранится в отдельном от базы данных файле, созданном родительской программой, а в базе данных хранится только путь к этому файлу. В момент загрузки и открытия таблицы, формы или отчета, где используется связанный объект, Access связывается с этим файлом, извлекает его содержимое и вставляет в таблицу, форму или отчет. Связывание объектов рекомендуется применять в тех случаях, когда в базу данных необходимо включить объект, который можно изменять, не вызывая Access. Например, речь может идти о таблице счетов, обрабатываемой табличным процессором Excel и используемой в нескольких базах данных. Наличие связи этих баз данных с одним и тем же Excel-файлом гарантирует наличие в них последней (самой свежей) версии таблицы счетов. Преимущество связи заключается в том, что оригинальный объект можно связать с несколькими базами данных и при этом его не нужно многократно подвергать копированию и сохранению в базах данных. Но следует помнить, что измененной версией будут пользоваться все приложения, которые связаны с этим объектом.

Рассмотрим пример встраивания **внедренного** объекта в таблицу базы данных. Пусть в каком-либо каталоге хранятся файлы, содержащие фото студентов, отредактированные в графическом редакторе Paint. В таблицу **Студент** в режиме конструктора таблиц наряду с полями новое поле с типом данных **Поле объекта OLE** с именем поля **Фото** (фото студента). В режиме ввода данных в таблице **Студент** щелкнем мышью в поле **Фото** вызовем контекстное меню, выбираем **Вставить объект.** Появится окно **Вставка объекта** (рис. 2.21).

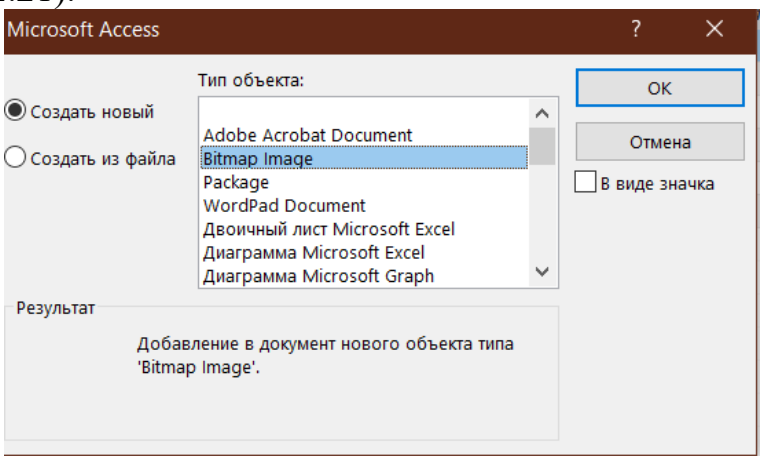

Рисунок 2.21 – Окно Вставка объекта

Если бы мы хотели сами нарисовать портреты студентов в редакторе Paint, то выбрали бы опцию *Создать новый*. В этом случае объекты будут внедренными. Но так как имеются готовые файлы, мы выбираем опцию *Создать из файла.* В списке *Тип объекта:* выбираем строку с надписью **Bitmap Image** и нажимаем кнопку *ОК*.

Опцию **Связь** не включаем, так как нам нужно встроить внедренные объекты, которые будут храниться в самой базе данных наравне со всеми другими данными. Далее нажимаем кнопку **Обзор** и в открывшемся окне

Обзор находим и вводим файл с нужным портретом. Нажимаем кнопку **ОК**. Вновь появляется окно **Вставка объекта**. В этом окне также нажимаем кнопку **ОК**.

Появляется окно таблицы **Студент** в режиме ввода данных. В первой строке таблицы в поле **Фото** появилась надпись **Bitmap Image**. Аналогичным путем вводим в таблицу **Студент** остальные файлы с фотографиями студентов. Теперь каждое изображение студента будет внедренным объектом, оно будет храниться в самой базе данных, как и все остальные данные. Исходные файлы с фотографиями теперь не нужны, их можно удалить.

Фото студента можно увидеть, если выполнить двойной щелчок мышью по надписи **Bitmap Image** в поле **Фото** в соответствующей записи таблицы **Студент** в режиме ввода данных. Однако просмотр таблицы **Студент** удобнее осуществлять в форме. Для этого создадим форму **Фото студента**. В форме будет расположено окно с фотографией студента, соответствующего записи с первым номером. Переключая номер записи, можно быстро просматривать информацию таблицы **Студент**, в том числе фото студентов.

Для этого необходимо на вкладке **Создание** в группе **Формы** выбрать кнопку **Мастер форм.**

В качестве источника данных выберем таблицу или запрос. Выберем допустимые поля, нажмем кнопку **Далее** (рис 2.22).

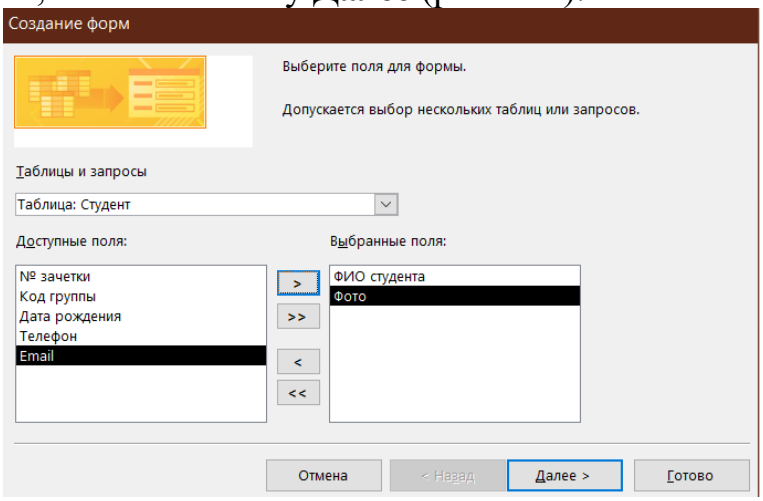

Рисунок 2.22 – Создание формы с помощью мастера форм

Выберем внешний вид формы нажимаем **Далее**. Зададим имя формы **Фото студента** далее кнопка **Готово.** Введем команду **Режим/Конструктор** и отредактируем форму (изменим название и размеры полей, изменим размеры окна для фото).

Вид отредактированной формы **Фото студента** показан на рисунке 2.23.

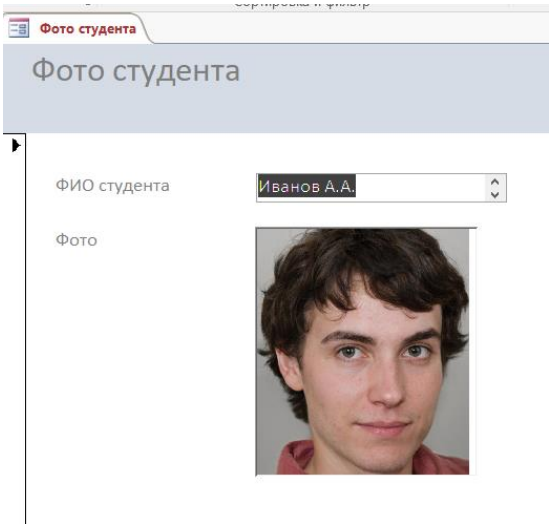

Рисунок 2.23 – Форма **Фото студента**

При создании формы в режиме конструктора для ввода поля внедренного объекта следует воспользоваться управляющим элементом **Присоединенная рамка объекта**, а в таблице его свойств в строке **Данные** указать поле **Фото** таблицы **Студент**.

3.3.3. Создание элементов формы или отчета

Как в формах, так и в отчетах помимо информации из БД можно отображать и дополнительную информацию. Окно формы может содержать следующие элементы: подписи, поля, поля со списком, списки, выключатели, переключатели, флажки и кнопки. Кроме того, форму (отчет) можно дополнить иллюстрацией (рисунком или диаграммой), текстом и линиями различного типа. Для оформления форм (отчетов) также может быть использована возможность изменения начертания, стиля и выравнивания данных, которые отображаются в полях, а также цвета символов, фона и границы.

Создание элементов окна осуществляется в режиме **Конструктора**.

Существует три основных типа элементов управления: присоединенные, свободные, вычисляемые.

Присоединенные элементы управления – элементы, связанные с полем таблицы. При вводе значения в присоединенный элемент управления поле таблицы в текущей записи автоматически обновляется. Большинство элементов управления, в том числе объекты OLE, можно присоединить к полю. Чаще всего присоединенные элементы управления содержат данные текстового типа, а также даты, числа, логические данные (Да/Нет), рисунки и поля MEMO.

Свободные элементы управления сохраняют введенную величину, не обновляя при этом поля таблицы. Их можно использовать для отображения:

- текста;
- значений, которые должны быть переданы макросам;
- линий и прямоугольников.

Кроме того, их можно использовать для хранения объектов OLE (например рисунков), которые расположены не в таблице, а в самой форме.

Свободные элементы управления называют также **переменными** или **переменными памяти.**

Вычисляемые элементы управления создают на основе выражений, например, функций или формул. Поскольку они не присоединены к полям таблицы, они не обновляют содержание полей таблицы. Этот элемент управления позволяет производить необходимые вычисления, используя данные полей таблицы, с последующим отображением в форме.

Выбор объектов – позволяет изменить указатель курсора на инструмент выбора объекта.

Мастера элементов – позволяет включать и отключать мастера по созданию элементов управления.

Надпись – предназначена для вывода на экран не изменяющегося текста, например, заголовков, подписей или пояснений. Надпись относится к свободным элементам управления, в которые нельзя вводить данные.

Поле – позволяет создать область для отображения, ввода или изменения данных. В поле можно использовать данные любого типа: текст, числа, дата/время, логические величины и МЕМО. Поля могут быть как присоединенными, так и свободными. В них можно использовать поля из таблиц или запросов, а также вычисляемые выражения, поэтому такие элементы управления называют **связанными полями**. При создании связанного поля вместе с ним одновременно образуется еще один элемент управления **– присоединенная надпись.**

Группа параметров – позволяет создать область настраиваемого размера для размещения набора флажков, переключателей или выключателей, представляющих набор альтернативных значений.

Выключатель – позволяет создать кнопку, связанную с логическим полем. Элемент может находиться в двух состояниях: **Истина** – кнопка **нажата**, **Ложь** – кнопка **отжата**.

Переключатель – предназначен для создания кнопки (называемой радиокнопкой). Ее функции аналогичны функциям выключателя. Элемент находится в двух состояниях: **Истина** – кружок **с точкой**, **Ложь** – **пустой** кружок. С кнопкой можно связать команды, например, выполняющие фильтрацию.

Флажок – предназначен для создания флажка, связанного с логическим полем. Действуют аналогично переключателям, но в отличие от них, допускают множественный выбор. Элемент может находится в двух состояниях: **Истина** – квадрат **с галочкой**, **Ложь** – **пустой** квадрат.

Поле со списком – позволяет создать составной элемент управления, объединяющий поле и раскрывающийся список значений. Для ввода значения, можно ввести значение в поле или выбрать значение в списке.

Список – позволяет создать список, допускающий прокрутку, и предназначенный для выбора значения. Позволяет отображать список значений в форме или отчете. В списках можно также отображать заголовки столбцов.

Кнопка – позволяет создать кнопку, используемую для выполнения набора макрокоманд Access или процедур VBA.

Рисунок – позволяет создать рамку, в которой в форме или отчете выводится неизменяемый рисунок. Поскольку рисунок не является объектом OLE, то после помещения рисунка в форму или отчет не допускается его изменение из Microsoft Access.

Свободная рамка объекта – позволяет создать рамку для отображения в форме или отчете объектов OLE, как правило, набор иллюстраций или диаграмму. Рамка не связана ни с каким полем таблиц базы данных.

Присоединенная рамка объекта –для отображения в форме или отчете объектов OLE, таких как набор иллюстраций или диаграммы. С присоединенной рамкой связано одно из полей таблиц. При переходе от записи к записи в форме или отчете выводятся разные объекты.

Конец страницы – позволяет создать элемент управления, указывающий принтеру начало новой страницы в печатной форме или новой страницы в отчете. Этот элемент управления не появляется в форме или запросе в режиме формы.

Вкладка – позволяет вставить элемент управления **Вкладка** для создания вложенных форм. Страницы элемента управления **Вкладка** могут содержать другие элементы управления.

Подчиненная форма/отчет – предназначена для добавления в основную форму или основной отчет подчиненной формы или подчиненного отчета соответственно. Добавляемые подчиненная форма или подчиненный отчет должны существовать.

Линия – позволяет создать прямую линию, которую можно перемещать и размеры которой можно изменять. Цвет и толщину линии можно изменить с помощью кнопок панели инструментов **Панель форматирования** или окна свойств. Используется для разделения элементов формы или отчета.

Прямоугольник – позволяет создать прямоугольник, который можно перемещать и размеры которого можно изменять. Используется для выделения элементов формы.

Дополнительные элементы – выбор этой кнопки открывает список дополнительных элементов управления ActiveX, которые можно использовать в формах и отчетах (рис. 2.24).

Рисунок 2.24 – Элементы управления

Для создания элемента управления: текста, поля, линии, прямоугольника (рамки), кнопки и др.:

1. Щелкните на соответствующей пиктограмме.
2. Укажите курсором мыши (крест с уменьшенным изображением создаваемого элемента) место для создаваемого элемента.

После того, как будет отпущена кнопка мыши для создания некоторых элементов (таких как, например, поле со списком или кнопка) Access выводит на экран Мастер. Так, после создания кнопки появляется Мастер, предлагающий выбрать тип действия, которое будет привязано к этой кнопке.

Внешний вид, структура и режимы работы отдельных управляющих элементов определяются значениями характеристик этих объектов (кнопка **Страница свойств** группа **Сервис**).

3.4. Создание отчета

Располагая базой данных можно распечатать любую таблицу, запрос или форму. Однако результаты печати не будут выглядеть профессионально, так как эти инструменты не предназначены для печати. С помощью отчета можно получить результаты в высококачественном варианте. В MS Access отчет представляет собой форму специального типа, предназначенную для вывода на печать. Но в отличие от форм отчеты не предназначены для вывода в окне и предназначены только для печати, т.е. создают не экранные, а печатные документы.

При создании отчета MS Access всегда оперирует только с одной единственной таблицей или запросом. Если необходимо объединить информацию из нескольких таблиц и (или) запросов в одном отчете, то прежде следует собрать желаемые данные в новом запросе.

Для создания отчета с помощью **Мастера отчетов**:

1. На вкладке **Создание** группа **Отчеты** выберите команду **Мастер отчетов**.
2. Укажите имя таблицы или запроса, на основе которых создаете отчет.
3. Выберите поля, данные которых будут помещены в отчет.
4. Задайте требуемый порядок сортировки полей.
5. Выберите вид макета отчета.
6. Задайте имя отчета.
7. Нажмите кнопку **Готово**.

Для создания отчета самостоятельно **без помощи Мастера Отчетов**:

1. На вкладке **Создание** группа **Отчеты** выберите команду **Конструктор отчетов**.
2. На вкладке **Конструктор** группа **Сервис** выберите команду **Добавить поля**.
3. Укажите имя таблицы, на которой должен базироваться отчет и выберите поля, данные которых будут отображаться в отчете.

Заголовок отчета – печатается только в начале отчета, используется на титульной странице.

Верхний колонтитул – печатается вверху каждой страницы.

Заголовок группы – печатается перед обработкой первой записи группы.

Область данных – печатается каждая запись таблицы или динамического набора данных запроса.

Примечание группы – печатается после обработки последней записи группы.

Нижний колонтитул – печатается внизу каждой страницы. Примечание отчета – печатается в конце отчета после обработки всех записей.

Проектирование отчета состоит в создании структуры его разделов и в размещении элементов управления внутри этих разделов, а также в задании связей между этими элементами и полями таблиц или запросов базы данных.

Отчеты предназначены для вывода информации на принтер, поэтому для расчета расположения данных на печатной странице программа MS Access должна «знать» все необходимое об особенностях принтера. Эти данные MS MS Access получает от операционной системы. Соответственно, принтер в системе должен быть установлен.

Более детально с приемами работы в MS Access можно ознакомиться, используя материалы рекомендованных источников.

Контрольные вопросы

1. Системы управления базами данных. Назначение, возможности.
2. Базы данных. Назначение. Преимущества.
3. Реляционный подход к построению модели данных.
4. Основные этапы проектирования реляционной базы данных.
5. Информационные модели реляционных баз данных.
6. Первичный и внешний ключ. Правила целостности для связанных полей
7. Примеры определения взаимосвязей между объектами в моделях Главная и подчиненная таблицы. Очередность их заполнения.
8. СУБД MS Access. Экранный интерфейс программы.
9. Технологии разработки баз данных средствами MS Access.
10. Конструктор таблиц. Типы данных.
11. Модификация структуры таблицы.
12. Нормализация таблиц базы данных.
13. Определение связей между таблицами.
14. Основные операции реляционной алгебры.
15. данных.
16. Свойства полей.
17. Связи между таблицами. Типы связей.
18. Технологии разработки таблиц базы данных
19. Целостность данных. Преимущества целостной базы данных.
20. Что такое ключевое поле? Для чего оно используется? Допустимо ли использовать несколько полей таблицы в качестве ключевого поля? Особенности ключевого поля с типом данных Счетчик?
21. Что такое поле со списком в таблице? В чем преимущество его использования?
22. Что такое свойства поля таблицы? Какие существуют свойства
23. Что такое Каскадное обновление связанных полей?
24. Что такое Каскадное удаление связанных записей?

25. Бланк запроса? Для чего он используется? Можно ли изменять межтабличные связи в бланке запроса?

26. Виды запросов. Технологии разработки запросов.

27. Источник данных для запроса? Позволяет ли запрос использовать в качестве источника данных другие запросы, несколько таблиц или запросов?

28. Режимы работы с запросом: Режим таблицы, Сводная таблица, Сводная диаграмма, Режим SQL, Конструктор.

29. Создание запросов на выборку данных, запросов с параметром

30. Создание запросов, изменяющих данные базы данных.

31. Создание итоговых запросов, вычисляемых полей

32. Формирование критериев отбора данных в зависимости от типа данных

33. Группировка и сортировка записей в отчете.

34. Из каких разделов состоит форма. Что содержит каждый раздел.

35. Конструктор форм, элементы управления. Конструктор отчетов

36. Нумерация записей в отчете.

37. Отчет. Структура отчета. Способы создания и форматирования.

38. Создание отчетной документации на основе данных базы данных

39. Создание форм ввода и управления данными.

40. Управляющая форма. Назначение. Свойство Источник записей формы.

Список использованных и рекомендованных источников

1. Кабанов В.А. Практикум Access: Учебное пособие. – Сергиев Посад, Филиал ФГБОУ ВПО «МГИУ», 2014. – 55 с.
2. Ковалева М.А. Создание баз данных в Microsoft Access. Учебно-методическое пособие – М.: Мир науки, 2019.
3. Одиночкина С.В. Разработка баз данных в Microsoft Access 2010 - СПб: НИУ ИТМО, 2012 – 83 с.
4. Селина Е.Г. Создание реляционных баз данных средствами СУБД Microsoft Access: Учеб.-метод. пособие. СПб.: Университет ИТМО, 2016 46 с.
5. Тарасов В.Л. Работа с базами данных в Access 2010. ЧАСТЬ 1: Учебно-методическое пособие. – [электронный ресурс]. – Нижний Новгород: Нижегородский государственный университет, 2014. – 126 с.
6. Тарасов В.Л. Работа с базами данных в Access 2010. ЧАСТЬ 2: Учебно-методическое пособие. – [электронный ресурс]. – Нижний Новгород: Нижегородский государственный университет, 2014. – 126 с.
7. Фомина, Е.Е. Работа с базами данных в MS Access 2010: учебное пособие / Е.Е. Фомина. Тверь: Тверской государственный технический университет, 2014 124 с.

Основные понятия MS Access

Система управления базами данных (СУБД) – программа для создания и использования баз данных.

База данных (БД) – хранилище данных некой предметной области, организованное для удобного накопления, быстрого поиска и обработки данных.

Объекты базы данных Access – таблицы, запросы, формы, отчеты, макросы и модули.

Таблица – множество строк (записей), содержащих данные, разнесенные по поименованным столбцам (полям).

Поле – столбец таблицы. Каждое поле таблицы имеет уникальное имя, характеризуется типом данных и свойствами, зависящими от этого типа.

Тип данных – характеристика поля, определяющая тип данных, который может содержать это поле. Существуют следующие типы данных: текстовый, числовой, дата/время, денежный, логический, счетчик, гиперссылка, MEMO, OLE и др.

Свойства поля – это набор характеристик, обеспечивающих дополнительные возможности управления хранением, вводом и отображением данных в поле. Перечень доступных свойств зависит от типа данных поля. Например, есть такие свойства, как: Размер поля, Обязательное поле, Условие на значение и др.

Ключевое поле (Первичный ключ) – поле с уникальными непустыми значениями, однозначно идентифицирующими каждую запись в таблице. Применяется для связи таблиц.

Ключевое поле (Первичный ключ) – одно или несколько полей (столбцов), комбинация значений которых однозначно определяет каждую запись в таблице. Первичный ключ должен иметь уникальные непустые значения. Первичный ключ используется для связывания таблицы с внешними ключами в других таблицах.

Внешний ключ – поле подчиненной таблицы, которое связано с ключевым полем главной таблицы. Если включено обеспечение целостности, то каждое значение внешнего ключа совпадает с одним из существующих значений в связанном ключевом поле. 6

Запись – строка таблицы. Каждая запись таблицы содержит сведения о каком-либо конкретном объекте предметной области.

Поле записи – пересечение строки и столбца (ячейка таблицы).

Значение поля записи – содержимое соответствующей ячейки таблицы.

Поле со списком – элемент управления, который состоит из поля, в которое можно вводить значения, и раскрывающегося списка заранее подготовленных значений, из которого можно выбрать значение и сохранить его в поле. Использование поля со списком делает более удобной работу с базой данных и снижает вероятность ошибок ввода.

Свойства подстановки поля со списком: Свойство «Источник строк» содержит имя таблицы, из которой берутся данные для поля со списком.

Свойство «Присоединенный столбец» содержит номер столбца указанной таблицы. Столбцы нумеруются слева направо, начиная с единицы.

Схема данных – графическое представление структуры базы данных, содержит списки полей таблиц и межтабличные линии связи. Позволяет устанавливать (удалять) межтабличные связи и изменять параметры связи.

Связь таблиц. Обычно в базе данных создается несколько таблиц. В одной таблице хранится информация об объектах одного типа, в другой таблице - об объектах другого типа. Если разнотипные объекты реального мира связаны между собой, то и таблицы могут быть связаны для отражения этой связи. Разнотипные объекты могут иметь общее свойство. Информация об этом свойстве хранится в столбце одной таблицы и в столбце другой таблицы. С помощью этих столбцов и осуществляется межтабличная связь.

Список полей – окно с перечнем полей таблицы. Ключевые поля выделены специальным значком.

Линия связи – линия, соединяющая ключевое поле главной таблицы и внешний ключ подчиненной таблицы. Если включено обеспечение целостности, то у концов линии появляются символы, показывающие тип связи (1 ко многим, или 1:1). Типы межтабличных связей: "один-к-одному", "один-ко-многим", "многие-ко-многим".

Связывание таблиц осуществляется способом перетаскивания первичного ключа главной таблицы на внешний ключ подчиненной таблицы в схеме данных.

Требования к связываемым полям: связываемые поля должны иметь одинаковую по смыслу информацию и одинаковый тип данных.

Главная и подчиненная таблица: если таблицы связаны связью «один-ко-многим», то таблица на стороне «один» называется главной, а на стороне «многие» – подчиненной таблицей. Первичный ключ главной таблицы связан с внешним ключом подчиненной таблицы. Если таблицы связаны связью «один-к-одному», то левая таблица в окне "Изменение связей" называется главной, а правая - подчиненной (связанной) таблицей.

Целостность данных – правила, которые СУБД автоматически соблюдает при вводе и удалении значений в связанных полях таблиц. Обеспечение целостности данных можно включить либо отключить при создании связи в схеме данных. Правила целостности:

1. первичный ключ должен содержать уникальные непустые значения,

2. внешний ключ должен содержать только те значения, которые уже имеются среди значений первичного ключа. Из этого следует:

– нельзя вводить во внешний ключ значения, которых нет в первичном ключе,

– нельзя изменять значения первичного ключа, для которых имеются совпадающие значения во внешнем ключе (если только не разрешено каскадное обновление связанных полей),

– нельзя удалять записи в главной таблице, для которых имеются подчиненные записи в подчиненной таблице (если только не разрешено каскадное удаление связанных записей).

Каскадное обновление связанных полей: для автоматического обновления значений внешнего ключа в подчиненной таблице при изменении значения первичного ключа в главной таблице, установите флажки «Обеспечение целостности данных» и «Каскадное обновление связанных полей». Для предотвращения изменений значения первичного ключа в главной таблице, если существуют связанные записи в подчиненной таблице, установите флажок «Обеспечение целостности данных» и снимите флажок «Каскадное обновление связанных полей».

Каскадное удаление связанных записей: для автоматического удаления связанных записей в подчиненной таблице при удалении записи в главной таблице установите флажки «Обеспечение целостности данных» и «Каскадное удаление связанных записей». Для предотвращения удаления записей из главной таблицы, если имеются связанные записи в подчиненной таблице, установите флажок «Обеспечение целостности данных» и снимите флажок «Каскадное удаление связанных записей».

Фильтр – набор условий, применяемых для отбора или сортировки записей.

Условие отбора в фильтре – выражение, относящееся к определенному полю таблицы, используемое для отбора записей, удовлетворяющих этому выражению.

Выражение – формула, записанная с использованием операторов, констант, функций, имен объектов базы данных. В результате вычисления формулы получается единственное значение. Правила, используемые при записи выражений:

- числа вводятся без ограничителей, например, 21;
- текст заключается в кавычки, например, «Иванов»;
- даты ограничиваются символами #, например, #10/01/99#.

Операторы, используемые в выражениях:

- арифметические: *, +, -, /,^;
- сравнения: <, <=, >, >=, =, < >;
- логические: And (И), Not (Нет), Or (Или);
- Like – для нахождения части значения поля;
- In - для определения, содержится ли элемент данных в списке значений;
- Between… And - для выбора значений из определенного интервала.

Форма – объект базы данных. Различают формы ввода-вывода и управляющие формы. Форма ввода-вывода представляет собой окно, специально разработанное для удобства ввода и просмотра информации. Управляющая форма представляет собой окно, специально разработанное для удобства работы с базой данных.

Источник данных формы – таблицы или запросы, на основе которых создается форма. Источником данных формы могут служить несколько таблиц или запросов.

Свободная форма – форма, для которой не указан источник данных. Используется для построения управляющих форм.

Элемент управления. Объект графического интерфейса пользователя (такой как поле, флажок, полоса прокрутки или кнопка), позволяющий пользователям управлять приложением. Элементы управления используются для отображения данных или параметров, для выполнения действий, либо для упрощения работы с интерфейсом пользователя.

Присоединенное поле – текстовое поле в форме, в котором отображается информация из поля текущей записи таблицы или запроса, на основе которых построена форма.

Свободное поле – текстовое поле в форме или отчете, которое не присоединено ни к какому источнику данных. В это поле пользователь может вводить свою информацию.

Запрос – объект базы данных. Запросы используются для выборки данных из таблиц, для изменения данных в таблицах, для вычислений.

Запрос на выборку – средство базы данных, позволяющее выбрать информацию из нескольких таблиц и других запросов в соответствии с условиями отбора. Кроме того, запросы на выборку позволяют производить вычисления.

Запрос на обновление – средство обновления устаревших данных в таблице.

Запрос на добавление – средство добавления записей из одной таблицу в другую. Таблицы должны иметь поля, с одинаковой по смыслу информацией.

Запрос на удаление – средство удаления из таблицы или нескольких таблиц записей, удовлетворяющих заданным условиям.

Запрос на создание таблицы – средство создания новой таблицы на основе существующих таблиц.

Перекрёстный запрос позволяет произвести вычисления в таблице базы данных и выдать результаты в виде перекрёстной таблицы. Применение перекрёстных таблиц позволяет более компактно и наглядно представить обобщенную информацию, сформированную на основе исходной таблицы базы данных.

Запрос с параметром – запрос, после запуска которого запрашивается ввод условий отбора. Чтобы создать запрос с параметром надо в условии отбора набрать в квадратных скобках произвольный поясняющий текст. Запросы с параметром создаются для удобства работы пользователя. Источник данных для запроса – таблицы и другие запросы, включенные в запрос.

Вычисляемое поле – поле, определенное в запросе для вычисления выражения для каждой записи с использованием данных из одного или нескольких полей.

Построитель выражений – средство, позволяющее автоматизировать построение сложных выражений.

Отчет – объект базы данных, обеспечивающий формирование печатных документов на основе информации из базы данных.

Приложение – база данных, обеспечивающая удобную среду общения пользователя с базой данных. Это подразумевает работу, в основном, с помощью управляющих форм, без непосредственного обращения к таблицам и запросам.

Связь между формой и запросом может быть организована так:

1. В форме создать свободное поле (или поле со списком)
2. В запросе в условии отбора сделать ссылку на это поле
3. В форме сделать кнопку для запуска этого запроса

Связь между формой и отчетом можно установить так:

1. Создать отчёт на основе запроса
2. В форме создать свободное поле (или поле со списком)
3. В запросе в условии отбора использовать ссылку на поле (или поле со списком), расположенное в форме
4. В форме создать кнопку для открытия отчета.

Учебно-методическое и информационное обеспечение

а) основная литература:

1. Баранникова, И. В. Теоретические основы автоматизированной обработки информации и управления: решение прикладных задач в MS Excel / И. В. Баранникова, Е. С. Могирева, О. Г. Харахан - Москва : МИСиС, 2018. - 58 с. - ISBN --. - Текст : электронный // ЭБС "Консультант студента" : [сайт]. - URL : https://www.studentlibrary.ru/book/misis_0008.html

2. Басыня Е.А., Операционные системы: учебно-методическое пособие / Басыня Е.А. - Новосибирск: Изд-во НГТУ, 2016. - 84 с. - ISBN 978-5-7782-3106-1 - Текст: электронный // ЭБС "Консультант студента": [сайт]. - URL: http://www.studentlibrary.ru/book/ISBN9785778231061.html

3. Воробьева Ф.И., Применение компьютерной техники в научных расчетах. MS Excel 2013: учебное пособие / Ф.И. Воробьева, Е.С. Воробьев - Казань: Издательство КНИТУ, 2018. - 152 с. - ISBN 978-5-7882-2357-5 - Текст: электронный // ЭБС "Консультант студента": [сайт]. - URL: http://www.studentlibrary.ru/book/ISBN9785788223575.html

4. Кильдишов, В. Д. Word 2019 для офисных работников : Справочник-практикум / В. Д. Кильдишов. - Москва : СОЛОН-ПРЕСС, 2020. - 140 с. - ISBN 978-5-91359-353-5. - Текст : электронный // ЭБС "Консультант студента" : [сайт]. - URL : https://www.studentlibrary.ru/book/ISBN9785913593535.html

5. Кильдишов, В. Д. Использование приложения MS Excel для моделирования различных задач / Кильдишов В. Д. - Москва : СОЛОН-ПРЕСС, 2015. - 156 с. - ISBN 978-5-91359-145-6. - Текст : электронный // ЭБС "Консультант студента" : [сайт]. - URL : https://www.studentlibrary.ru/book/ISBN9785913591456.html

6. Сергеева, А. С. Базовые навыки работы с программным обеспечением в техническом вузе. Пакет MS Office (Word, Excel, PowerPoint, Visio), Electronic Workbench, MATLAB : учебное пособие / Сергеева А. С. , Синявская А. С. - Новосибирск. : СибГУТИ, 2016. - 263 с. - Текст : электронный // ЭБС "Консультант студента" : [сайт]. - URL : https://www.studentlibrary.ru/book/SibGUTI-009.html

7. Широков, А. И. Операционные системы и среды : основные понятия теории : учеб. / А. И. Широков, Ф. Г. Кирдяшов, С. Э. Мурадханов, под ред. Е. А. Калашникова и Л. П. Рябова. - Москва : МИСиС, 2018. - 192 с. - ISBN 978-5-906953-49-0. - Текст : электронный // ЭБС "Консультант студента" : [сайт]. - URL : https://www.studentlibrary.ru/book/ISBN9785906953490.html

б) дополнительная литература:

1. Давыдов И.С., Информатика: Учебное пособие / И. С. Давыдов. - СПб: Проспект Науки, 2017. - 480 с. - ISBN 978-5-903090-19-8 - Текст: электронный // ЭБС "Консультант студента": [сайт]. - URL: http://www.studentlibrary.ru/book/PN0017.html

2. Кильдишов, В. Д. MS Excel и VBA для моделирования различных задач / Кильдишов В. Д. - Москва : СОЛОН-ПРЕСС, 2019. - 256 с. - ISBN 978-

5-91359-315-3. - Текст : электронный // ЭБС "Консультант студента" : [сайт]. - URL : https://www.studentlibrary.ru/book/ISBN9785913593153.html

3. Краткое введение в операционные системы / Сташук П. В. - Москва : ФЛИНТА, 2019. - ISBN 978-5-9765-0143-0. - Текст : электронный // ЭБС "Консультант студента" : [сайт]. - URL : https://www.studentlibrary.ru/book/ISBN9785976501430.html

4. Омельченко В.П., Информационные технологии в профессиональной деятельности / Омельченко В.П., Демидова А.А. - М.: ГЭОТАР-Медиа, 2019. - 432 с. - ISBN 978-5-9704-5035-2 - Текст: электронный // ЭБС "Консультант студента": [сайт]. - URL: http://www.studentlibrary.ru/book/ISBN9785970450352.html

5. Шандриков А.С., Информационные технологии: учеб. пособие / А.С. Шандриков - Минск: РИПО, 2017. - 443 с. - ISBN 978-985-503-694-5 - Текст: электронный // ЭБС "Консультант студента": [сайт]. - URL: http://www.studentlibrary.ru/book/ISBN9789855036945.html

Учебное издание

КОНСПЕКТ ЛЕКЦИЙ
по дисциплине
«ПРОИЗВОДСТВЕННОЕ ОБУЧЕНИЕ»
для студентов направления подготовки
Профессиональное обучение (по отраслям),
профили «Информационные технологии и системы»,
«Профессиональная психология»
(в 2-х частях). Часть 2.

С о с т а в и т е л ь:
Марина Владимировна Авершина

Печатается в авторской редакции.
Компьютерная верстка и оригинал-макет автора.

Подписано в печать _____
Формат 60x84$^{1}/_{16}$. Бумага типограф. Гарнитура Times
Печать офсетная. Усл. печ. л. _____. Уч.-изд. л. _____
Тираж 100 экз. Изд. №_____. Заказ №_____. Цена договорная.

Издательство Луганского государственного
университета имени Владимира Даля

Свидетельство о государственной регистрации издательства
МИ-СРГ ИД 000003 от 20 ноября 2015г.

Адрес издательства: 91034, г. Луганск, кв. Молодежный, 20а
Телефон: 8 (0642) 41-34-12, **факс:** 8 (0642) 41-31-60
E-mail: izdat.lguv.dal@gmail.com **http:** //izdat.dahluniver.ru/