

**БЕЛКООПСОЮЗ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«БЕЛОРУССКИЙ ТОРГОВО-ЭКОНОМИЧЕСКИЙ
УНИВЕРСИТЕТ ПОТРЕБИТЕЛЬСКОЙ КООПЕРАЦИИ»**

Кафедра информационно-вычислительных систем

**ТЕХНОЛОГИИ ОРГАНИЗАЦИИ,
ХРАНЕНИЯ И ОБРАБОТКИ
ДАННЫХ**

**Практикум
для студентов заочной формы обучения
экономических специальностей**

Авторы-составители: *Л. М. Ашарчук*, ст. преподаватель;
Г. Л. Костюченко, ассистент

Рецензенты: *В. Д. Левчук*, канд. техн. наук, доцент кафедры математических проблем управления Гомельского Государственного университета им. Ф. Скорины;
С. М. Мовшович, канд. техн. наук, доцент, зав. кафедрой информационно-вычислительных систем Белорусского торгово-экономического университета потребительской кооперации

Рекомендован научно-методическим советом УО «Белорусский торгово-экономический университет потребительской кооперации». Протокол № 5 от 11 июня 2002 г.

Технологии организации, хранения и обработки данных:
Т 38 Практикум для студентов заочной формы обучения экономических специальностей / Авторы-составители: *Л. М. Ашарчук*,
Г. Л. Костюченко. — Гомель: УО «Белорусский торгово-экономический университет потребительской кооперации», 2005. — 64 с.
ISBN 985-461-294-5

ББК 32.973.26-018.2

ISBN 985-461-294-5

© Авторы-составители: *Л. М. Ашарчук*,
Г. Л. Костюченко, 2005

© УО «Белорусский торгово-экономический университет потребительской кооперации», 2005

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Предметом учебной дисциплины «Технологии обработки и хранения данных» (ТОХОД) являются современные технологии проектирования и организации реляционных баз данных в среде наиболее доступной широкому кругу пользователей СУБД Access.

Данный практикум предназначен для студентов-заочников экономических специальностей в качестве пособия по изучению курса «Технологии обработки и хранения данных» и содержит материал по следующим разделам:

1. Теоретические основы разработки баз данных.
 - 1.1. Понятие и виды информационного обеспечения автоматизированных информационных систем.
 - 1.2. Основные понятия баз данных.
 - 1.3. Информационные объекты и модели данных.
 - 1.4. Системы управления базами данных.
2. Технология ведения баз данных.
 - 2.1. Реляционная база данных СУБД Access.
 - 2.2. Свойства полей и типы данных СУБД Access.
 - 2.3. Схема данных СУБД Access.
 - 2.4. Этапы проектирования и создания базы данных СУБД Access.
3. Лабораторное занятие 1.
4. Лабораторное занятие 2.
5. Вопросы для самоконтроля.

Авторы ставят своей целью в доступной форме разъяснить студентам общие вопросы использования СУБД Access и продемонстрировать порядок создания базы данных — от проектирования простейшей инфологической модели до формирования результирующих запросов и отчетов.

Практикум рекомендуется также преподавателям для проведения лабораторных занятий со студентами заочной формы обучения.

1. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ РАЗРАБОТКИ БАЗЫ ДАННЫХ

1.1. Понятие и виды информационного обеспечения автоматизированных информационных систем

Автоматизированные информационные системы (АИС) предприятий и организаций обеспечивают информационное обслуживание своих пользователей на основе полного и достоверного информационного обеспечения.

Под *информационным обеспечением* понимается совокупность решений по составу, объему, размещению и формам организации информации, циркулирующей в системе. Информационное обеспечение включает следующие компоненты:

- совокупность реквизитов и показателей экономической информации, характеризующих предметную область;
- систему классификации и кодирования экономической информации;
- систему документации и документооборота;
- совокупность информации, организованной на машинных носителях.

Информационное обеспечение АИС принято разделять на *внемашинное* и *внутримашинное*. К *внемашинному* информационному обеспечению относят сведения на таких носителях информации, которые могут восприниматься человеком непосредственно, не прибегая к использованию технических средств. Например: экономические документы, заполняемые вручную; классификаторы информации, используемые для кодирования различных номенклатур информации (товаров, поставщиков, единиц измерения и др.).

Внутримашинное информационное обеспечение формируется и используется на основе имеющегося технического и программного обеспечения АИС. Основными способами организации внутримашинного информационного обеспечения являются: файловая организация данных и формирование баз данных.

Файловая организация данных предполагает, что для решения конкретной пользовательской задачи (или комплекса взаимосвязанных задач) средствами системы программирования формируется набор информационных массивов (файлов данных). Массив или файл данных представляет собой поименованный набор однородных записей, размещенный на машинном носителе (магнитном диске). Различают несколько видов массивов: нормативно-справочные, оперативные, текущие, хранимые в совокупности с программами их организации образуют внутримашинную информационную базу автоматизированной системы.

Файловая организация данных преобладала в АИС до недавнего времени. Существенными ее недостатками являются следующие:

- затрудненный доступ к данным для пользователя, не знакомого с программированием;
- несовместимость различных форматов файлов, созданных разными системами программирования и в разное время.
- значительное дублирование информации при создании и хранении файлов;
- внесение изменений в структуру файла влечет за собой внесение изменений во все программы, использующие данный файл;
- создание незапланированных запросов и отчетов практически невозможно без написания новых программ;

Перечисленные недостатки файловой организации привели к переходу на более эффективный способ ведения внутримашинного информационного обеспечения — базы данных.

1.2. Основные понятия баз данных

База данных (БД) представляет собой совокупность взаимосвязанных, хранящихся вместе данных при наличии такой минимальной избыточности, которая допускает их эффективное использование для одного или нескольких приложений (задач). Данные и их описания (словарь данных) хранятся так, что они в значительной степени становятся независимыми от использующих их программ. Для добавления новых или модификации существующих данных, а также для поиска данных в базе данных применяется общий управляемый способ.

Перечислим основные преимущества организации баз данных:

1. Одноразовый ввод данных и их многократное использование.
2. Независимость программ от данных. Существующие программы и логические структуры данных не переделываются при внесении изменений в базу данных, поэтому затраты на программирование существенно сокращаются.
3. База данных — это основа для будущего наращивания прикладных программ. Базы данных обеспечивают возможность разработки приложений легче, быстрее, дешевле.
4. Легкость использования. Пользователи имеют простой доступ к данным; сложные функции доступа к данным осуществляет СУБД.
5. Гибкость использования. Обращение к данным или их поиск осуществляется с помощью различных методов доступа.
6. Быстрая обработка незапланированных запросов на данные. Случайные запросы на данные обрабатываются с помощью высокоуровневого языка запросов или языка генерации отчетов, а не прикладными программами.
7. Простота внесения изменений. База данных может увеличиваться и изменяться без нарушения имеющихся способов использования данных.
8. Уменьшение избыточности данных. Требования новых приложений удовлетворяются за счет существующих данных, а не путем создания новых файлов.
9. Производительность. Запросы на данные удовлетворяются с такой скоростью, которая требуется для использования данных.
10. Достоверность хранения данных. Система предотвращает наличие различных версий одних и тех же элементов данных, доступных пользователям, на различных стадиях обновления.
11. Секретность. Несанкционированный доступ к данным невозможен.
12. Защита от искажения и уничтожения. Данные должны быть защищены от сбоев, катастрофических и криминальных ситуаций, некомпетентного или злонамеренного обращения к ним лиц, которые могут ошибочно изменить их.

13. Готовность. Пользователь быстро получает данные всякий раз, когда это необходимо.

Для поддержки баз данных необходимы специальные прикладные программы — СУБД.

Система управления базой данных (СУБД) — это универсальный комплекс прикладных программ, предназначенный для создания и обслуживания баз данных, обеспечения многовариантного доступа к данным и их обработки.

База данных может быть размещена на одном компьютере (локальная база данных) или распределена между несколькими компьютерами, соединенными в единую вычислительную систему с помощью компьютерных сетей (распределенная база данных).

Локальные базы данных используются при работе с небольшими объемами данных и решении несложных задач. Назначение распределенных баз данных состоит в предоставлении эффективных форм обслуживания удаленных пользователей при работе со значительными объемами информации в условиях их географической или структурной разобщенности.

1.3. Информационные объекты и модели данных

Организация данных в базе требует предварительного моделирования, т. е. построения *информационно-логической модели данных* (ИЛМ). Назначение ИЛМ — систематизация разнообразной информации и от-

ражение ее свойств по содержанию, структуре, объему, связям. Логическая модель данных отображает конкретную предметную область. В свою очередь, *предметная область* включает информационные объекты — информационные описания реальных объектов, процессов, явлений или событий. *Информационный объект* образуется совокупностью логически взаимосвязанных реквизитов-признаков и реквизитов-оснований, представляющих качественные и количественные характеристики некоторой сущности предметной области.

Примерами информационных объектов могут быть клиенты, счета, документы, операции, товары, поставщики, студенты, преподаватели, кафедры, дисциплины и т. п. Формализованное описание информационного объекта выглядит таким образом:

КЛИЕНТЫ (КодКлиента, НаимКлиента, Адрес, РасчСчет);

ТОВАРЫ (КодТовара, НаимТовара, ЕдИзм, Цена).

При построении информационно-логической модели данных выбирается один из трех подходов моделирования: иерархический, сетевой, реляционный.

Иерархическая модель данных организует данные в виде древовидной структуры и является реализацией логических связей: родо-видовых отношений или отношений «целое-часть».

Примером иерархического представления может служить административная структура высшего учебного заведения: университет — факультет — специальность — студенческая группа. Графическим способом представления иерархической структуры является *дерево*.

Дерево представляет собой иерархию элементов, называемых узлами. Элементы дерева — список атрибутов, описывающих объекты. На самом верхнем уровне иерархической модели находится *корень дерева*, от которого расходятся *порожденные узлы*. Каждый порожденный узел имеет один *исходный*, находящийся на более высоком уровне. Узлы, не имеющие порожденных, называются *листьями*. Между исходным узлом и порожденными узлами существует отношение *Один ко многим*.

Иерархическая структура лежит в основе формирования иерархической базы данных. Графическое представление модели показано в приложении 1.

Простота понимания используемого принципа иерархии, обеспечение независимости данных, наличие промышленных СУБД, поддерживающих данную модель, являются достоинствами рассматриваемой модели. Недостатками, ограничивающими ее использование, считаются: сложность отображения связей *Многие ко многим*, сложность добавления новых и удаления устаревших объектов, возможность доступа к любому узлу только через корневой.

Сетевая модель данных также основана на графовом представлении информации, но отличается от иерархической наличием горизонтальных связей между ее элементами. Любой элемент в сетевой структуре может быть связан с любым другим элементом. В ней присутствует два вида взаимосвязей: *Один ко многим* и *Многие к одному*. Можно считать, что иерархическая модель является частным случаем сетевой модели.

Примером сетевого представления данных могут служить отношения между объектами *Студенческий коллектив, Студенческая группа, Комната в общежитии* и *Студент*.

Графическое представление модели показано на рис. 2.

База данных, описываемая сетевой моделью, состоит из нескольких областей. Каждая область состоит из записей. *Запись* — логически заверченный набор *полей*. Объединение записей в логическую структуру производится не только по областям, но и с помощью так называемых *наборов* — поименованных двухуровневых деревьев. Используя множество таких двухуровневых связей, системный аналитик может конструировать достаточно сложные структуры данных.

Основной недостаток сетевой модели состоит в ее сложности. Прикладной программист должен детально знать логическую структуру базы для правильной навигации среди различных экземпляров наборов и записей. Другим недостатком является возможная потеря независимости данных в ходе ее модификации.

В конце 60-х годов сотрудник фирмы ИБМ доктор Э. Кодд предложил термин «реляционная модель данных».

Реляционная модель данных представляется в виде совокупности двумерных таблиц особого вида — отношений, над которыми выполняются операции, формулируемые в терминах реляционной алгебры. В настоящее время эта модель получила наибольшее распространение при разработке баз данных, она практически вытеснила другие модели из процесса создания и ведения БД.

При описании реляционных моделей используется следующая терминология:

- *отношение* — множество однотипных записей (по сути, это реляционная таблица);
- *атрибут* — столбец отношения;
- *домен* — множество допустимых значений атрибута (или совокупность значений одного столбца);
- *кортеж* — множество пар «атрибут-значение», запись (или строка);
- *схема отношения* — список имен атрибутов одного отношения (последовательность названий столбцов таблицы);
- *мощность отношения* — число его кортежей.

Фундаментальные свойства отношений:

1. *Атомарность значений атрибутов*. Значения всех атрибутов являются атомарными. Среди значений домена не могут содержаться множества значений (отношения).

2. *Отсутствие кортежей-дубликатов.* Никакие два кортежа отношения не могут быть дубликатами друг друга в любой произвольно заданный момент времени.

3. *Отсутствие упорядоченности кортежей.* Данные могут храниться в произвольном порядке, однако при формулировании запросов к базе данных можно потребовать определенной сортировки результирующей таблицы в соответствии со значениями некоторых столбцов.

4. *Отсутствие упорядоченности атрибутов.* Это свойство позволяет легко модифицировать таблицы путем добавления новых или удаления существующих атрибутов.

Из второго свойства следует, что каждое отношение реляционной модели обладает хотя бы одним возможным ключом.

Первичный ключ (ключ) — это атрибут, позволяющий однозначно идентифицировать запись среди множества других записей. Если ключ состоит из значений нескольких атрибутов, то он называется *составным*, а если из одного атрибута — *простым*.

Связи между отношениями в реляционной модели устанавливаются по равенству значений одинаковых атрибутов в парах отношений.

Эти связи могут быть типа *Один к одному*, *Один ко многим* и *Многие ко многим*. Они дают возможность совместно использовать данные из разных отношений.

Прежде чем сформировать реляционную модель данных, следует провести нормализацию наборов отношений. Цель процесса нормализации состоит в том, чтобы добиться минимальной избыточности информации, хранимой в этих отношениях.

Говорят, что отношение имеет *нормальную форму*, если оно не носит характер вложений, т. е. никакое отношение не может быть определено, как член другого отношения. Недопустимо наличие в отношении двух записей с одинаковыми ключами.

Выделяется следующая последовательность нормальных форм:

- Первая нормальная форма.
- Вторая нормальная форма.
- Третья нормальная форма.
- Нормальная форма Бойса-Кодда.
- Четвертая нормальная форма.
- Пятая нормальная форма.

Требование первой нормальной формы является базовым. Оно определяет, что значения всех атрибутов должны быть атомарными.

Отношение находится во второй нормальной форме, когда находится в первой нормальной форме и каждый неключевой (любой, не входящий в состав первичного ключа) атрибут отношения полностью зависит от первичного ключа.

Отношение находится в третьей нормальной форме в том случае, если все неключевые реквизиты взаимно независимы и полностью зависят от первичного ключа. На практике третья нормальная форма схем отношений является достаточной. Приведением к ней процесс проектирования реляционной базы данных обычно заканчивается.

С логической точки зрения *реляционная модель данных* может быть представлена *множеством двумерных таблиц* различного предметного наполнения. Графическое представление реляционной модели данных дано в приложении 3.

1.4. Системы управления базами данных

Наличие информации и приведение предметной области пользователя к какой-либо модели данных еще недостаточно для реализации технологии ведения базы данных. Необходимо приобрести соответствующее техническое и программное обеспечение.

Специальные пакеты прикладных программ — *системы управления базами данных* — предназначены именно для работы с файлами баз данных.

Система управления базой данных — это универсальный комплекс прикладных программ, предназначенный для создания и обслуживания баз данных, обеспечения многовариантного доступа к данным и их обработки. Диспетчером данных, выполняющим загрузку и сохранение данных в базах, является *ядро базы данных*. Другие компоненты СУБД — трансляторы и утилиты.

Утилиты обеспечивают настройку системы, тестирование, восстановление баз данных, сбор статистики о функционировании базы данных.

Трансляторы — это программные компоненты СУБД, обеспечивающие перевод запросов пользователя, записанных на удобном ему языке, в систему исполняемых команд. Они являются посредниками между пользователем и ядром СУБД.

При помощи средств СУБД пользователь может выполнить следующее:

- создать структуру новой базы данных;
- наполнить ее содержимым;
- редактировать содержимое и отображать информацию в удобном и привычном виде;
- получать различные отчеты.

СУБД предполагает работу пользователя с базой данных в разных режимах:

- *ассистент* — с использованием разветвленного меню;
- *командный* — предполагает диалог пользователя и системы на языке команд СУБД;
- *программный* — использующий язык СУБД и позволяющий создать пользовательские программы

различной степени сложности.

Наиболее известны СУБД dBase, FoxBase, FoxPro, Clipper, Paradox, Oracle, MS SQL Server, Access.

Выбор СУБД определяется многими факторами: конфигурацией имеющихся технических средств, стоимостью приобретения и сопровождения программы, уровнем подготовленности персонала и трудностью освоения системы. Однако главным фактором является возможность работы с построенной моделью данных.

Важнейшая характеристика СУБД — тип поддерживаемой модели данных: иерархической, сетевой, реляционной.

Большинство известных СУБД для персональных компьютеров поддерживают реляционную модель.

Одним из примеров программы, реализующей реляционную базу данных, может служить СУБД Access фирмы Microsoft. Перечислим последние версии этого пакета, используемые на практике: СУБД Access 95, СУБД Access 97, СУБД Access 2000, СУБД Access 2002. Каждая последующая версия СУБД представляет собой качественно новый уровень пакета, в котором расширена функциональность, обеспечено разнообразие и удобство пользовательского интерфейса. Например, начиная с Access 97, в СУБД появились средства облегчения работы в Интернете и создания приложений для Web-страниц.

В СУБД Access реализованы все элементы Windows-технологий:

- Полная совместимость с программами пакета MS Office.
- Динамический обмен данными (DDE) с любыми приложениями WINDOWS.
- Поддержка механизма OLE — связывания и встраивания объектов (рисунков, графиков, звуковых файлов).
- Применение метода *drag and drop* (*перетащить и отпустить*).

СУБД Access может обрабатывать файлы других СУБД (Paradox, dBase, FoxPro) и файлы СУБД, поддерживающих стандарт ODBC — Oracle, MS SQL Server и др.

СУБД Access поддерживает работу многопользовательской базы данных в компьютерной сети: следит за разграничением прав доступа пользователей к базе и обеспечивает защиту данных. При этом система может функционировать в локальных сетях с архитектурой *файл-сервер* и *клиент-сервер*. СУБД Access позволяет создавать специальные копии — *реплики* общей базы данных, с которой пользователи могут одновременно работать на разных компьютерах.

В дальнейшем мы будем ориентироваться на описание СУБД Access 97, работающей под управлением операционной системы Windows 98.

2. ТЕХНОЛОГИЯ ВЕДЕНИЯ БАЗ ДАННЫХ

2.1. Реляционная база СУБД ACCESS

В базу данных СУБД Access входят разнородные объекты следующих типов:

1. *Таблицы* — главные объекты базы. В них хранятся все данные и структура таблицы (поля, их типы и свойства).

2. *Запросы* — временные результирующие таблицы, отображающие данные базовых таблиц. Запросы позволяют извлекать данные из одной (однотабличные) или нескольких таблиц (многотабличные) и представлять их пользователю в удобном виде. С помощью запросов выполняется преобразование данных по заданному алгоритму, отбор данных, их сортировка, группировка и фильтрация.

По способу формирования выделяют два типа запросов: QBE-(Query By Example) — запросы по образцу; SQL-(Structured Query Language) — запросы на языке структурированных запросов.

В рамках данного пособия рассматриваются только запросы первого типа, которые принято называть запросами на выборку или запросами выборки.

Перечислим наиболее распространенные виды QBE-запросов:

- простые запросы на выборку из одной или нескольких таблиц;
- с формированием вычисляемых полей;
- с условиями отбора;
- с группированием записей и проведением групповых операций (суммирование, нахождение среднего, минимального, максимального, количества значений в группе и др.);
- запросы с параметрами, выполнение которых начинается с отображения специальных окон, в которые пользователь вводит условия или ограничения на выборку;
- перекрестные запросы;

- активные запросы на создание новых таблиц, на модификацию данных в имеющихся таблицах.

Запросы могут создаваться на основе таблиц, запросов и запросов таблиц.

3. *Формы* — специальные средства для ввода, просмотра и корректировки данных, находящихся в таблицах. С помощью форм можно заполнять не все поля таблицы, а только нужные. Форму часто делают в виде бланка первичного документа, особенно когда ввод данных происходит с заполненных бланков. В форму вводят различные элементы управления и оформления. Формы позволяют просмотреть графические объекты, хранящиеся в полях соответствующего типа.

4. *Отчеты* — это аналоги выходных документов, формируемых средствами Access.

По свойствам и структуре отчеты похожи на формы, но предназначены для вывода информации на принтер. Для них существенно наличие группировок и различных степеней итогов, специальных элементов управления: колонтитулов, заголовков, и др. Источниками информации являются базовые таблицы, специально созданные запросы или таблицы и запросы.

5. *Макросы* состоят из последовательности внутренних команд Access и являются средством, позволяющим квалифицированному пользователю написать на специальном языке последовательность действий СУБД в конкретной ситуации.

6. *Модули* — это программы, написанные на языке программирования Visual Basic for Applications.

Все объекты СУБД Access могут размещаться в одном файле на магнитном диске.

СУБД Access предоставляет несколько средств создания каждого из основных объектов базы. К *ручным* средствам относится разработка объектов в режиме *Конструктор* (таблиц, форм, запросов, отчетов). К *автоматизированным* средствам — разработка объектов с помощью программ-мастеров, например, мастера таблиц, форм, отчетов. К *автоматическим* средствам относятся различные способы ускоренного создания простейших объектов, настроенных по умолчанию (*Автоформа, Автоотчет*).

В следующих разделах данного пособия на примере простейших приложений будут рассмотрены наиболее доступные и целесообразные способы создания объектов доступа к данным в СУБД Access.

2.2. Свойства полей и типы данных СУБД Access

Из параграфа 1.3 мы знаем, что структуру реляционной таблицы образуют столбцы и строки. Их аналогами в базе данных СУБД Access являются поля и записи. *Поля* определяют структуру базы и групповые свойства данных, записываемых в ячейки, принадлежащие каждому полю. Перечислим и разъясним эти свойства, чтобы в дальнейшем при разработке таблиц базы данных, пользователь смог правильно описать их:

1. *Имя поля*. Определяет, как следует обращаться к данным этого поля при автоматических операциях с базой. По умолчанию используется в качестве заголовков столбцов таблиц. Состоит имя поля из последовательности символов (цифры, буквы, знаки препинания) до 64. Первым символом не может быть пробел, нельзя использовать точку.

2. *Тип данных*. Определяет тип данных, которые могут размещаться в поле (*Текстовый, Числовой, Денежный, Дата/Время, Счетчик, Мемо, Поле объекта OLE, Логический*).

3. *Размер поля* — предельная длина символов, размещаемых в поле.

4. *Формат поля*. Определяет способ форматирования данных в ячейках. Например, формат поля с типом *Дата/Время* — краткий формат даты.

5. *Маска ввода*. Уточняет форму, в которой должны вводиться все значения данного поля. Например: числовое поле с форматом *целое число* может иметь маску 000, означающую, что допустимыми значениями данного поля являются числа, состоящие ровно из трех цифр. Это своего рода средство контроля ввода данных *по шаблону*.

6. *Подпись*. Определяет заголовок столбца таблицы для данного поля. Это второй идентификатор поля. Первый — *имя поля*. Например, имя поля *КТ*, а подпись — *Код товара*. Это означает, что при выводе столбца в таблице, форме или запросе на экране будет выдан заголовок столбца *Код товара*. Если подпись не задана, выводится имя поля *КТ*.

7. *Значение по умолчанию*. Значение, вводимое в ячейки поля автоматически, если оно не введено пользователем. Например, для поля *Единица измерения* задано значение *штук*. Оно будет автоматически отображаться в ячейках этого поля, пока пользователь не введет свое значение, например, *кг*.

8. *Условие на значение*. Содержит выражение для проверки правильности ввода данных. Например, для поля *КодТовара*:

>=1001 And <=5004

Это означает, что значения кода товара должны находиться в ограниченном диапазоне от 1001 до 5004. Условие на значение позволяет контролировать правильность ввода данных при заполнении таблиц.

9. *Сообщение об ошибке*. Используется, если задано свойство *Условие на значение*. Выдается автоматически при нарушении правил ввода данных в виде сообщения, например: *неправильно введен код товара*.

10. *Обязательное поле*. Определяет обязательность заполнения данного поля при наполнении базы.

11. *Пустые строки* — свойство, разрешающее ввод пустых строковых данных.

12. *Индексированное поле* принимает три значения:

- Нет;
- Да (Допускаются совпадения);
- Да (Совпадения не допускаются).

Если поле обладает свойством индекса, то операции, связанные с сортировкой и поиском записи, существенно ускоряются.

Для уникальных ключей таблицы индекс задается автоматически, совпадения в нем не допускаются.

Для других полей, по которым часто проводятся сортировка и поиск записей, процесс индексирования следует указать специально: *Да (Допускаются совпадения)*.

Поскольку в разных полях могут содержаться данные разного типа, то и свойства у полей могут различаться.

Типы данных, с которыми работает СУБД Access следующие:

• *Текстовый* предназначен для хранения обычного неформатированного текста ограниченного 256-ю символами.

• *Числовой* служит для хранения действительных чисел.

• *Денежный* используется для хранения денежных сумм.

• *Счетчик* применяется для уникальных натуральных чисел с автоматическим наращиванием. Естественное использование — для порядковой нумерации записей, автоматического формирования порядковых кодов.

• *Поле MEMO* предназначено для хранения больших объемов текста размером до 65535 символов.

• *Поле объекта OLE* используется для хранения мультимедийных, графических объектов.

• *Дата/время* применяется для хранения календарных дат и текущего времени.

• *Логический* используют для хранения логических данных. Могут принимать только два значения:

ДА или *НЕТ*.

Из списка типов данных можно выбрать *Мастер Подстановок*. Это не специальный тип данных, а объект, настройкой которого можно автоматизировать ввод данных в поле так, чтобы не вводить их вручную, а выбирать из раскрывающегося списка.

Схема данных СУБД Access

Схема данных — это графический образ многотабличной Access-базы. Схема данных наглядно отображает структуру таблиц и связи между ними, обеспечивает использование связей при обработке и *целостность* базы данных.

В схеме данных, построенной по нормализованной реляционной модели предметной области, могут быть установлены связи *Один к одному* и *Один ко многим*.

В большинстве случаев связывают ключевое поле одной таблицы с соответствующим ему полем (чаще всего имеющим то же имя), которое называют *полем внешнего ключа*, во второй таблице. Связанные поля не обязательно должны иметь одинаковые имена, но они должны иметь одинаковые типы данных и иметь совпадающие значения в поле связи. Кроме того, связываемые поля числового типа должны иметь одинаковые значения свойства *Размер поля*. Возможно устанавливать связи между полем типа *счетчик* и полем *числового* типа.

Из двух связанных таблиц одна называется *главной*, а другая — *подчиненной*. Главной является та таблица, которая участвует в связи своим ключевым полем, т. е. уникальным ключом. *Уникальный ключ*, как это следует из описания реляционной модели, — это поле, значения в котором не повторяются. Подчиненная таблица участвует в связи полем внешнего ключа, которое в ней является неключевым полем или полем составного ключа. Такие отношения определяют связь *Один ко многим*.

Связь *Один ко многим* (1:М) устанавливается в том случае, когда уникальный ключ главной таблицы связывается с неключевым полем или элементом составного ключа подчиненной таблицы.

При заполнении таблиц записи всегда вводятся сначала в главные таблицы, а затем в подчиненные.

Связь *Один к одному* (1:1) устанавливается, если в подчиненной таблице внешний ключ является уникальным ключом, т. е. поле связи совпадает с уникальным ключом обеих таблиц. Тогда главной считается та таблица, данные в которую должны вводиться в первую очередь — справочные, нормативные, плановые сведения.

В схеме данных могут устанавливаться связи-объединения, однако в данном пособии нами они не рассматриваются.

Упомянутое понятие целостности, которую призвана обеспечить *Схема данных* в процессе создания, ведения и корректировки базы данных, означает следующее:

- в подчиненную таблицу не может быть добавлена запись с несуществующим в главной таблице значением ключа связи (обеспечение целостности данных);
- изменение значений ключа связи главной таблицы должно приводить к изменению соответствующих значений в записях подчиненной таблицы (каскадное обновление связанных полей);

- в главной таблице нельзя удалить запись, если не удалены связанные с ней записи в подчиненной таблице (каскадное удаление связанных полей);

При попытке пользователя нарушить эти условия в операциях обновления или удаления данных в связанных таблицах СУБД Access выводит соответствующее сообщение и не допускает выполнения операции.

2.4. Этапы проектирования и создания базы данных СУБД Access

Жизненный цикл любого программного продукта, в том числе и базы данных состоит из:

- проектирования;
- реализации;
- эксплуатации.

Наиболее значимой в жизненном цикле является стадия проектирования. От того, насколько тщательно продумана структура базы, насколько четко определены связи между ее элементами, зависит производительность системы БД и СУБД, ее информационная насыщенность.

Процесс проектирования разбивается на три этапа:

1. Формирование концептуальной модели БД.
2. Логическое проектирование базы данных.
3. Физическое проектирование базы данных.

На этапе *концептуального проектирования* определяются информационные потребности будущих пользователей БД, проводится идентификация функциональной деятельности конкретной предметной области, идентификация объектов, осуществляющих эту функциональную деятельность. На выходе этапа имеется концептуальная модель. Например, функциональная деятельность — *Учет поставок товаров*; информационные объекты — *Товары, Поставщики, Накладные, Поставка*.

Этап логического проектирования предполагает преобразование концептуальной модели в структуру базы данных с учетом выбранной модели данных. На выходе этапа имеем логическую структуру базы данных: набор таблиц конкретной структуры и связей между таблицами.

Этап физического проектирования включает процедуры формирования объектов хранения данных, объектов доступа к данным, алгоритмов формирования различных запросов, их выдачи пользователю. Он базируется на информационно-логической модели БД, полученной на этапе логического проектирования. На выходе — файл (файлы) базы данных.

С базами данных работают две категории исполнителей. Первая категория — *проектировщики*. Их задача состоит в разработке структуры базы данных и согласовании ее с заказчиком.

Вторая категория — *пользователи* базы данных. Они получают исходную базу от проектировщиков и занимаются ее наполнением и обслуживанием.

Поскольку материал этого пособия рассчитан на пользователей, особо подчеркнем, что именно заказчик должен предоставить проектировщику *техническое задание на проектирование базы*. Для его подготовки следует досконально знать предметную область, а не компьютер или технологии ведения баз данных. Методически правильно начать работу с карандашом и листом бумаги в руках.

Техническое задание содержит следующие списки:

- исходные данные, с которыми работает заказчик;
- выходные данные, которые необходимы для управления внутри предприятия;
- выходные данные, которые заказчик должен предоставлять в другие организации.

Получив эти данные, проектировщик приступает к созданию структуры базы данных. Известны следующие методы логического проектирования: классический подход на основе правил нормализации; построение ИЛМ на основе формальных правил выделения сущностей из документов; CASE-средства проектирования; семантическое моделирование ИЛМ (например, различные разновидности ER-моделей).

Неквалифицированному пользователю довольно сложно освоить эти специальные методы, поэтому мы ограничимся более простым, интуитивно-понятным описанием процедуры проектирования БД.

Проектирование реляционной базы данных Access включает следующие работы:

1. Составление генерального списка полей.
2. Определение типов данных и свойств полей.

3. Распределение полей генерального списка по базовым таблицам. Сначала это делают по функциональному признаку для того, чтобы ввод данных в одну таблицу происходил на одном рабочем месте или в одном подразделении. Затем анализируют полученные таблицы и проводят их нормализацию, исключая избыточность и множественные повторы одних и тех же данных в разных таблицах.

4. Определение уникального ключа для каждой таблицы. Если ни одно из полей не может быть ключевым, можно создать дополнительное поле типа *Счетчик*, значения которого будут формироваться автоматически из уникальных чисел натурального ряда. Можно также сформировать составной ключ.

5. Составление и оформление информационно-логической модели БД на бумаге.

6. Конструирование таблиц средствами СУБД Access.
 7. Создание объекта *Схема данных* средствами СУБД Access.
 8. Загрузка таблиц модельными данными в следующем порядке: сначала заполнение нормативно-справочных таблиц (главных), затем оперативно-учетных данных (подчиненных таблиц).
 9. Разработка форм для быстрого и удобного ввода данных в таблицы. При этом следует создавать подчиненные формы для таблиц, связанных отношениями *Одни ко многим*, для однократного ввода данных.
 10. Конструирование запросов — аналогов выходной информации, требуемой для управления, как правило, в рамках организации.
 11. Конструирование отчетов — аналогов выходной информации, используемой для передачи за пределы предприятия (вышестоящие организации, контролирующие органы, управление статистики и др.).
 12. Конструирование объектов, предназначенных для автоматизации работы с базой — *макросов* и *модулей*.
 13. Передача исходной базы данных заказчику в эксплуатацию.
- Работы 1–5 относятся к предпроектной подготовке *информационно-логической модели* базы данных; работы 6–12 составляют этап непосредственного проектирования базы данных.
- С работы 13 начинается процесс внедрения базы данных. На практике он включает следующие процедуры:
- заполнение справочников реальной информацией;
 - ввод оперативно-учетных данных; запуск сконструированных ранее запросов и отчетов;
 - конструирование новых запросов, отчетов, макросов и модулей;
 - внесение изменений в структуру БД в связи с возможными ошибками в техническом задании и изменениями условий эксплуатации базы данных, вызванных переменами в производственно-хозяйственной деятельности предприятия-заказчика.

ЗАДАНИЯ ПО ТЕМАМ ЛАБОРАТОРНЫХ ЗАНЯТИЙ

Лабораторное занятие 1 «Проектирование и создание базы данных Таможня в среде СУБД Access»

Цель — научиться создавать структуру реляционной базы данных, вводить данные, формировать объекты доступа к данным средствами СУБД Access.

Объем — 4 часа.

Задание — разработать базу данных *Таможня* для таможенных органов.

Описание предметной области

На таможенные органы возложено взимание таможенных пошлин при перемещении товаров через таможенную границу.

Таможенная пошлина — обязательный взнос (платеж), взимаемый таможенными органами при ввозе товара на таможенную территорию или вывозе товара с этой территории.

Таможенные пошлины уплачиваются по единым ставкам.

Основой для исчисления таможенной пошлины является таможенная стоимость товара.

Таможенная стоимость товара — это цена сделки, фактически уплаченная или подлежащая уплате за товар на момент пересечения им таможенной границы. Она указывается в первичном документе под названием «счет-фактура». Таможенная стоимость может быть разной для одного и того же товара в зависимости от коммерческих условий поставки.

На таможенные службы возлагаются также функции по сбору полных и достоверных статистических сведений о ввозе и вывозе товаров в натуральном (количественном) и стоимостном выражении.

На основе информации, хранящейся в базе данных *Таможня* решается ряд прикладных задач, например следующие:

- определение сумм таможенных пошлин на ввозимые товары;
- определение сумм таможенных пошлин за отчетный период;
- сбор статистических сведений о ввозимых товарах;
- анализ ввоза товаров.

Для ввода информации в базу данных используются следующие документы:

1. Нормативно-справочный документ «Ставки таможенных пошлин на ввозимые на таможенную территорию Республики Беларусь вещи».

Ставки утверждаются постановлением Совета Министров Республики Беларусь. Макет справочника представлен на рис. 1.

Ставки таможенных пошлин

Классификация товара по Товарной номенклатуре внешнеэкономической деятельности, код товара	Наименование товара	Ставка пошлины (в процентах от таможенной стоимости за единицу товара)
--	---------------------	--

Рис. 1. Макет документа Ставки таможенных пошлин

2. Оперативно-учетный документ *Счет-фактура*. Документ предоставляется декларантом — лицом, предъявляющим таможенным органам товары, подлежащие таможенному оформлению. Макет документа *Счет-фактура* представлен на рис. 2. Для учебной базы дан сокращенный перечень реквизитов документа *Счет-фактура*.

Счет-фактура № ____ от «__» _____ 2002 г.

Наименование товара	Ед. измерения	Количество	Таможенная стоимость единицы
---------------------	---------------	------------	------------------------------

Рис. 2. Макет документа *Счет-фактура*

Задание

В ходе лабораторной работы студент должен выполнить три основных задания: спроектировать базу данных, сформировать структуру базы данных средствами СУБД Access, ввести исходные данные и выполнить их обработку посредством получения запросов и отчетов.

1. Проектирование БД:

1.1. Составление генерального списка полей и определение для каждого поля типа данных с описанием свойств.

1.2. Определение информационных объектов (таблиц).

1.3. Распределение полей генерального списка по таблицам.

1.4. Определение ключевых полей.

1.5. Указание связей и типов отношений.

2. Создание БД:

2.1. Загрузка СУБД Access.

2.2. Создание структур таблиц.

2.3. Создание Схемы данных.

3. Использование БД:

3.1. Заполнение таблиц данными.

3.2. Проектирование запросов:

- *Пошлина* — расчет пошлины на ввозимые товары;
- *1 сентября* — вывод списка товаров, провезенных в определенный день — 1 сентября;
- *Выборка за любой день* — вывод списка товаров, провезенных в любой день;
- *Статистика* — расчет количества каждого товара и суммы таможенной пошлины за каждый день анализируемого периода.

3.3. Создание формы *Счет-фактура* для ввода и редактирования данных, получаемых из документа *Счет-фактура*.

3.4. Подготовка выходного документа *Отчет за сентябрь* с помощью *Мастера Отчетов*.

Порядок выполнения задания

1. Проектирование БД

Процесс разработки структуры базы данных называется логическим проектированием. Проектировщик может использовать разные методы логического проектирования: нормализации [3], сущность-связь [15], выделения информационных объектов из документов [16].

Самым доступным, не требующим дополнительного изучения специальной литературы, является *интуитивно-логический подход* к разработке простейшей базы данных. Таблицы базы данных проектируются так, чтобы данные в них хранились с минимальной избыточностью, а их ввод был возможен на основе конкретного документа.

1.1. Определение генерального списка полей, определение типа данных и описание свойств каждого поля

На основании перечня реквизитов, входящих в документы и необходимых для решения задачи, определяется следующий генеральный список полей:

- *Код Товара*: тип данных *Текстовый* (вычисления с кодом товара не производятся), однозначно идентифицирует каждый товар;

- *Наименование Товара*: тип данных *Текстовый*;
- *Ставка*: тип данных *Числовой* (в процентах от таможенной стоимости за единицу товара);
- *№Счет-фактуры*: тип данных *Текстовый*, номер счета-фактуры может повторяться. Например, 2 января все продавцы товаров выпишут счет-фактуру № 1;
- *Дата*: тип данных *Дата/время*, краткий формат даты;
- *Ед.Изм.*: тип данных *текстовый*. В учебной базе предложен товар, измеряемый в штуках;
- *Количество*: тип данных *Числовой*;
- *Таможенная Стоимость Ед*: тип данных *Денежный*. Согласно условию задачи у одного товара в разных счет-фактурах таможенная стоимость единицы товара может быть разная.

Обратите внимание, что *Наименование Товара* перечисляется один раз, хотя он встречается в двух документах. БД не должна содержать избыточных данных, т.е. любая информация хранится один раз. Исключением являются ключевые поля, так как с помощью ключевых полей устанавливается связь между таблицами.

1.2. Определение информационных объектов (таблиц).

После определения генерального списка полей проектировщик обдумывает, сколько таблиц необходимо и какие поля включать в какие таблицы.

Информационные объекты (таблицы) для нашей задачи:

- *Товар* — таблица заполняется один раз и содержит справочную информацию; в нее заносятся сведения из документа *Ставки таможенных пошлин*;
- *Счет-фактура* — в таблице накапливается информация обо всех провозенных через границу товарах, получаемая из документа *Счет-фактура*. В задаче рассматриваются счета-фактуры только на импортируемые товары. В одном счете-фактуре каждый товар может упоминаться один раз.

Примечание. В учебной базе предложен сокращенный перечень реквизитов документа *Счет-фактура*, поэтому можно ограничиться созданием двух таблиц. В реальной задаче в заголовочной части счета-фактуры располагаются и другие реквизиты. В этом случае пришлось бы выполнить нормализацию, т.е. таблицу *Счет-фактура* разбить на две таблицы. В одной таблице следовало указать постоянные реквизиты, находящиеся в заголовочной части документа, а в другой — реквизиты из многострочной содержательной части документа. Связь между таблицами установится с помощью поля *№Счет-фактуры*.

1.3. Распределение полей генерального списка по таблицам

- *Товар* (КодТовара, НаименованиеТовара, Ставка);
- *Счет-фактура* (№Счет-фактуры, Дата, КодТовара, ЕдИзм, Количество, Таможенная СтоимостьЕд).

1.4. Ключевые поля

В таблице *Товар* ключевое поле *КодТовара*. Код товара однозначно идентифицирует товар. Значения в этом поле не повторяются.

В таблице *Счет-фактура* явного ключевого поля нет. В таком случае возможно:

- назначить, если возможно, составной ключ из двух или более полей;
- не задавать уникальный ключ;
- поручить СУБД Access назначить ключевое поле при сохранении таблицы. Это будет дополнительное поле с первоначальным именем *код* типа *счетчик*.

В нашем примере в таблице с оперативно-учетной информацией не будем задавать уникальный ключ.

1.5. Связи и тип отношений

Связь между таблицами *Товар* и *Счет-Фактура* устанавливается по полю *Код Товара*. Тип связи — *Один ко многим*. Главная таблица — *Товар*, подчиненная — *Счет-Фактура* (рис. 3).

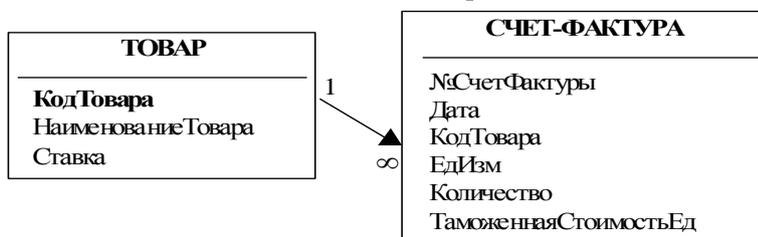


Рис. 3. Таблицы с логически завершенным перечнем полей, связью и типом отношений

Связь *Один ко многим* означает, что одна запись в таблице *Товар* может быть связана с несколькими записями в таблице *Счет-фактура*. Действительно, на практике один и тот же товар может провозиться несколько раз и быть указан в разных счет-фактурах, но в справочнике ставок каждый товар упомянут однажды.

2. Создание БД Таможня

2.1. Загрузка СУБД Access

Для запуска Microsoft Access, необходимо выполнить следующие действия:

в *Главном меню* в пункте *Программы* выберите *Microsoft Access*;

в разделе *Создание базы данных* установите переключатель *Новая база данных*, нажмите на кнопку *ОК*. Появится диалоговое окно *Файл новой базы данных*;

выберите папку (например, *STUD*) для сохранения базы;

присвойте файлу имя *Таможня* и свою фамилию (например, *Таможня-Иванов*), нажмите на кнопку *Создать*.

Появится окно *База данных*, которое не закрывается в течение всего сеанса работы с данной базой. Оно является контейнером, содержащим все объекты базы данных: *Таблицы*, *Запросы*, *Формы*, *Отчеты*, *Макросы*, *Модули*.

Примечание. Пункты, в которых предложены задания для выполнения на компьютере, промаркированы значком компьютера.

2.2. Создание структуры таблиц

Сначала следует создавать таблицы со справочной, а затем с учетной информацией.

Рекомендуемый порядок создания таблиц:

- выбрать способ создания таблицы;
- присвоить имена полям;
- задать тип данных;
- назначить свойства общие и подстановки;
- присвоить ключ;
- сохранить таблицу.

Создание структуры таблицы Товар

Порядок создания:

сделайте активной вкладку *Таблицы* и нажмите на кнопку *Создать*. В диалоговом окне *Новая таблица* выберите способ создания — *Конструктор*, нажмите на кнопку *ОК*.

Откроется окно *Конструктора таблиц*, состоящее из двух частей.

В верхней части окна для каждого поля выполняются следующие операции: вводится *Имя поля* с клавиатуры; выбирается *Тип данных* из раскрывающегося списка; вводится текст *Описания* (не обязательно).

В нижней части окна можно просмотреть и изменить *Свойства* выделенного поля.

введите имена полей с клавиатуры в соответствии с рис. 4. Обратите внимание, что имена полей написаны без пробелов и каждое слово пишется с большой буквы;

для поля *Ставка* выполните щелчок в ячейке *Тип данных*. Из раскрывающегося списка выберите *Числовой* тип данных (по умолчанию предложен *Текстовый*);

выполните щелчок по полю *КодТовара* и в *Свойствах поля* установите размер поля равный 2 (рис. 4);

для присвоения *Ключа* установите курсор в поле *КодТовара* и на панели инструментов нажмите на кнопку с изображением ключа. Слева от поля *КодТовара* должен появиться значок ключа;

для сохранения таблицы на панели инструментов выполните щелчок по кнопке *Сохранить*. В открывшемся окне диалога введите имя таблицы *Товар*, нажмите на кнопку *ОК*. Закройте окно конструктора таблицы.

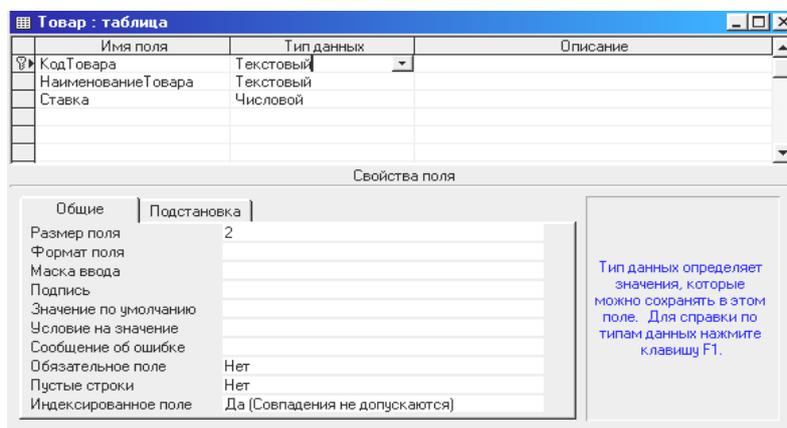


Рис. 4. Конструктор таблиц. Структура таблицы *Товар*

Порядок создания:

■ аналогично создайте структуру таблицы *Счет-фактура*. Имена полей и Типы данных предложены на рис. 5;

■ для поля *ЕдИзм* в *Свойствах поля* введите с клавиатуры *Значение по умолчанию*— *штук* (без кавычек) и подтвердите ввод щелчком в любом месте окна; Кавычки появятся после подтверждения;

■ для поля *КодТовара* в *Свойствах поля* установите *Размер поля* равный 2, как в таблице *Товар*.

Создайте свойство подстановки для поля *КодТовара*. С помощью подстановки формируются значения полей, являющихся внешними ключами (рис. 3). Данные в поле не вводятся с клавиатуры, а выбираются из раскрывающегося списка, содержащего набор возможных значений кодов товара из таблицы *Товар*.

Для этого выполните следующее:

■ установите курсор в ячейке с именем поля *КодТовара*;

■ выполните щелчок по вкладке *Подстановка* в *Свойствах поля*;

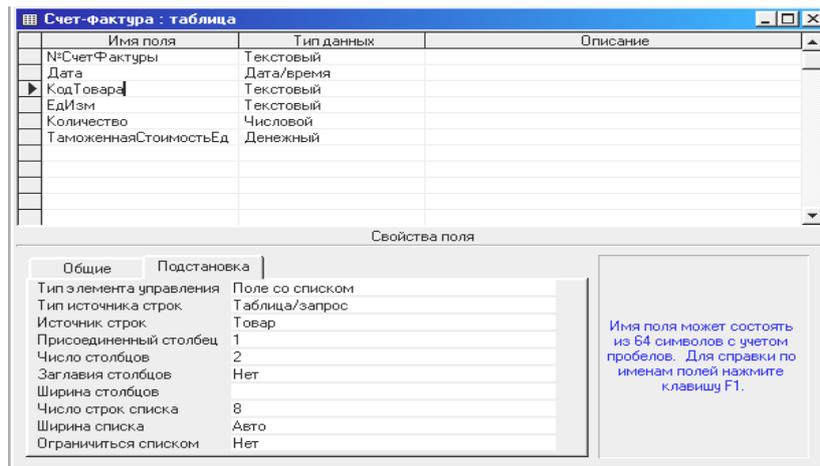


Рис.5. Структура таблицы *Счет-фактура*

■ в раскрывающихся списках *Свойств Подстановки* выберите свойства в соответствии с рис.5.;

■ сохраните таблицу с именем *Счет-фактура*. В таблице *Счет-фактура* ключевого поля нет, поэтому при сохранении на вопрос *Создать ключевое поле сейчас* нажмите на кнопку *Нет*;

■ закройте окно конструктора таблицы.

2.3. Создание Схемы данных

Для установления связей выполните следующие задания (действия):

■ нажмите на кнопку *Схема данных*  на панели инструментов. Появится окно *Схема данных* и в нем окно *Добавление таблицы*. Окно *Добавление таблицы* переместите вниз за заголовок для увеличения обзора;

■ выделите таблицу *Товар* и нажмите на кнопку *Добавить*. Аналогично добавьте таблицу *Счет фактура*;

■ закройте окно *Добавление таблицы*;

■ для установления связи в окне *Схема данных* перетащите поле *КодТовара* из главной таблицы (*Товар*) и бросьте на поле *КодТовара* в подчиненной таблице (*Счет-фактура*);

■ откроется окно диалога *Связи*, в котором установите флажки *Обеспечение целостности данных*, *Каскадное обновление связанных полей*, *Каскадное удаление связанных полей*, нажмите на кнопку *Создать*. Тип отношения будет определен как *Один ко многим* (рис. 6).

■ закройте и сохраните *Схему данных*.

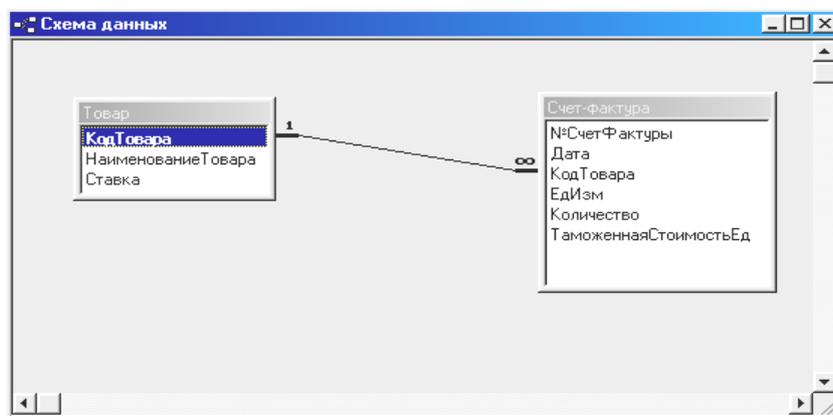


Рис. 6. Схема данных базы Таможня

3. Использование БД

3.1. Заполнение таблиц данными

Для заполнения таблиц данными выполните следующие действия:

- в окне *База данных* выделите таблицу *Товар* и нажмите на кнопку *Открыть*;
- заполните таблицу *Товар* данными в соответствии с табл. 1;

Примечание. Данный справочник содержит большое количество наименований товара, имеет длинные коды товаров. Чтобы уменьшить ввод информации с клавиатуры, для решения учебной задачи предложен сокращенный вариант справочника.

- закройте таблицу;
- аналогично заполните таблицу *Счет-фактура* данными из табл. 2. Обратите внимание, что единица измерения вводится автоматически, так как было определено свойство поля *значение по умолчанию*. *КодТовара* выбирается из раскрывающегося списка. Так реализуется свойство *подстановки* (для наглядности увеличьте ширину столбца *КодТовара*). В ячейки поля *ТаможеннаяСтоимостьЕд* вводите только число, значения дробной части и подпись денежной единицы формируются автоматически;
- закройте таблицу.

Таблица 1. Исходные данные для таблицы *Товар*

Код товара	Наименование товара	Ставка
01	Живые животные	5
06	Живые деревья	5
87	Транспортные средства	30
91	Часы	20
94	Мебель	30

Таблица 2. Исходные данные для таблицы *Счет-фактура*

№ счет-фактуры	Дата	Код товара	Единица измерения	Количество	Таможенная стоимость, ед.
123	01.09.04	87	шт.	1	2 200,00 р.
65	01.09.04	91	шт.	100	20,00 р.
43	01.09.04	87	шт.	1	2 200,00 р.
76	01.09.04	91	шт.	50	15,00 р.
12	01.09.04	87	шт.	1	3 500,00 р.
4	01.09.04	91	шт.	200	10,00 р.
54	01.09.04	87	шт.	1	2 000,00 р.
65	01.09.04	87	шт.	1	3 200,00 р.
123	02.09.04	87	шт.	1	3 200,00 р.
12	02.09.04	94	шт.	1	600,00 р.
23	02.09.04	91	шт.	30	25,00 р.
45	02.09.04	91	шт.	50	30,00 р.

3.2. Создание запросов

Создание запроса Пошлина

Назначение данного запроса — рассчитать сумму пошлины на каждый ввозимый товар. Запрос на выборку, многотабличный, с вычисляемым полем.

Вычисляемое поле создается с использованием *Построителя выражений*. В Access выражения строятся из имен полей. Рядом с именем поля указывается имя таблицы-источника. Имя вычисляемого поля отделяется от выражения двоеточием.

Выражение (формула) для расчета пошлины будет иметь вид:

*Пошлина:[Счет-фактура]![ТаможеннаяСтоимостьЕд]**
[Счет-фактура]![Количество] [Товар]![Ставка]/100*

Порядок создания запроса:

- активизируйте вкладку *Запросы* и нажмите на кнопку *Создать*;
- в окне *Новый запрос* выберите пункт *Конструктор*, нажмите на кнопку *ОК*;
- появятся окно *Конструктор запросов* и окно *Добавление таблицы*. Выделите таблицу *Счет-фактура*, нажмите на кнопку *Добавить*. Аналогично добавьте таблицу *Товар*. Закройте окно *Добавление таблицы*. На экране останется окно *Конструктор запросов* с именем *Запрос1: запрос на выборку*.

Структура окна Конструктора Запросов

Окно *Конструктора запросов* разделено на две части. В верхней части находится схема таблиц-источников, на основе которых будет создаваться запрос.

В нижней — располагается *Бланк запроса*, в котором формируются столбцы запроса. По умолчанию *Бланк запроса* состоит из строк *Поле*, *Имя таблицы*, *Сортировка*, *Вывод на экран*, *Условие отбора*, или. На пересечении строки и столбца находится ячейка. Активизация той или иной ячейки щелчком мыши вызывает появление кнопки списка, содержащего перечень применяемых опций.

Ячейки строки *Поле* предназначены для указания имени поля запроса. Обычно имя поля запроса совпадает с именем поля таблицы — источника данных. *Имя нового поля* (например, вычисляемого или полученного в результате групповых операций) указывается перед выражением и отделяется двоеточием.

В ячейках строки *Имя таблицы* выводятся имена таблиц — источников данных. Для вычисляемых полей имя таблицы не указывается.

Каждая ячейка строки *Сортировка* имеет раскрывающийся список, в котором выбирается порядок сортировки записей результата запроса (*не сортируется*, *по возрастанию*, *по убыванию*).

В ячейках строки *Вывод на экран* флажком отмечаются поля, значения которых должны быть на экране после выполнения запроса. По умолчанию предлагается показать все поля.

В ячейках строки *Условие отбора* и (или) указываются условия отбора записей. Выражениями, задающими условия отбора, могут быть значения полей с операторами сравнения и логическими операторами. Например:

>500 — значения из поля больше 500;

>=2 — значения из поля больше либо равны 2;

<> — значения из поля не равны 100;

Between #01.09.04# And #10.09.04# — все числа с 1 сентября по 10 сентября включительно;

Like «С*» OR «К*» — названия на «С» или «К».

Условия отбора задаются по одному полю или нескольким полям с операторами Or и (или) And.

Строка *Групповая операция* по умолчанию не показана, вызывается нажатием на кнопку с изображением знака суммы на панели инструментов или командой *Групповые операции* из меню *Вид*. В ячейках строки указываются порядок группирования записей, имеющих одинаковые значения в группируемых полях, и вычислительные операции над значениями полей, включенных в группу. Для расчетов наиболее часто используются статистические функции:

- Sum — сумма значений поля;
- Avg — среднее от значений поля;
- Min — наименьшее значение поля;
- Max — наибольшее значение поля;
- Count — число значений поля без учета пустых значений.

Порядок заполнения бланка запроса из окна схемы таблиц-источников:

в ячейки строки *Поле* добавьте поля: *НаименованиеТовара*, *Дата*, *Количество*, *ТаможеннаяСтоимостьЕд* и *Ставка* (рис. 7) любым из способов:

- перетащите поле из таблицы-источника и бросьте в очередную свободную ячейку строки *Поле*;
- в таблице-источнике выполните двойной щелчок по добавляемому полю.

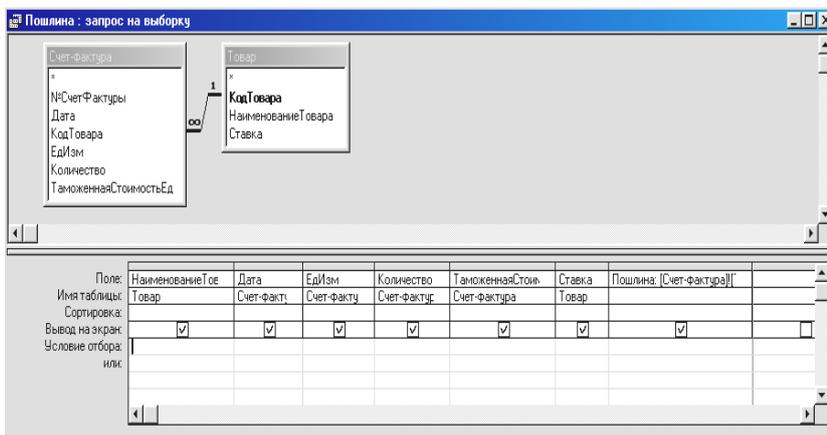


Рис. 7. Окно Конструктора запроса Пошлина

Для создания вычисляемого поля *Пошлина* выполните следующее:

- ☐ выполните щелчок кнопкой мыши в очередной свободной ячейке строки *Поле*;
- ☐ на панели инструментов нажмите на кнопку *Построить*;
- ☐ появится окно диалога *Построитель выражений* (рис. 8);
- ☐ выполните двойной щелчок кнопкой мыши по знаку плюс на папке *Таблицы*. Папка *Таблицы* развернет вложенные папки таблиц;
- ☐ выполните щелчок кнопкой мыши по папке с таблицей *Счет-фактура*. В средней области *Построителя выражений* появится список полей таблицы;
- ☐ выделите поле *ТаможеннаяСтоимостьЕд* и нажмите на кнопку *Вставить*. Поле вставится в верхнюю область *Построителя выражений*. Рядом с именем поля указывается имя таблицы-источника;
- ☐ введите знак умножения и аналогично вставьте поле *Количество*;
- ☐ введите знак умножения, вставьте поле *Ставка* из таблицы *Товар* и выполните деление на 100;
- ☐ перед формулой введите с клавиатуры имя поля *Пошлина*. Между именем поля и формулой поставьте двоеточие. Пробелы не вводятся;
- ☐ сверьте полученное вами выражение с выражением на рис. 8, нажмите на кнопку *OK*;

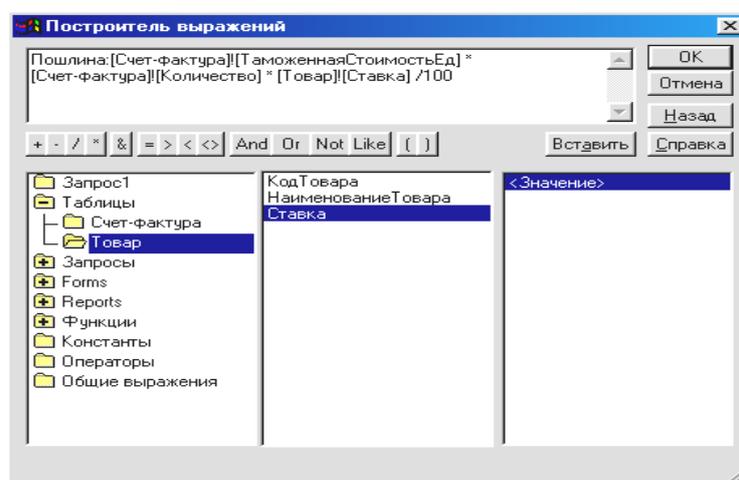


Рис. 8. Выражение для вычисляемого поля *Пошлина*

- ☐ подтвердите ввод *Выражения* в ячейку щелчком кнопки мыши в любом месте бланка запроса;
- ☐ просмотрите результат выполненных расчетов. Для этого на панели инструментов нажмите на кнопку *Представление запроса* и выберите пункт *Режим таблицы*;
- ☐ сохраните запрос под именем *Пошлина*;
- ☐ закройте запрос.

Создание запроса с именем 1 сентября

Назначение запроса — показать список товаров, провезенных в определенный день (1 сентября). Запрос на выборку с условием отбора. Значение поля *Дата 1.09.04* является критерием, вводимым в ячейку *Условие отбора* для поля *Дата*.

Порядок выполнения запроса:

- ☐ создайте новый запрос с помощью *Конструктора запросов*;
- ☐ добавьте таблицы *Товар* и *Счет-фактура*;
- ☐ в бланк запроса добавьте поля в соответствии с рис. 9;

- в ячейку условие отбора для поля *Дата* введите дату *1.09.04* и подтвердите ввод щелчком по кнопке мыши в любом месте бланка. В ячейке появится выражение *#01.09.04#* (тип данных *дата/время*);
- сохраните запрос под именем *1 сентября*;
- просмотрите результат;
- закройте запрос.

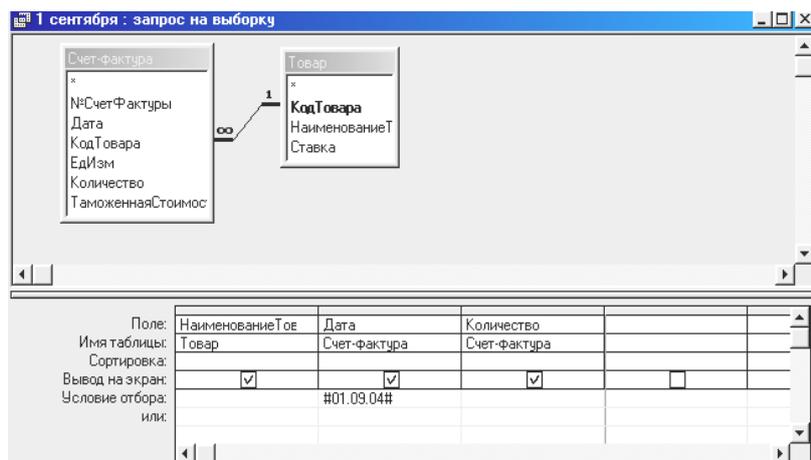


Рис. 9. Окно Конструктора запроса 1 сентября

Создание запроса Выборка за любой день

Назначение запроса — показать список товаров, провезенных в любой день, а не только 1 сентября, как это выполнялось в предыдущем запросе. Запрос *на выборку с параметром*. Значение поля *Дата* является параметром запроса и вводится с клавиатуры в специальном диалоговом окне.

Порядок создания запроса:

- создайте новый запрос с помощью *Конструктора запросов*;
- таблицы-источники — *Товар* и *Счет-фактура*;
- добавьте поля *НаименованиеТовара*, *Дата* и *Количество*;
- в ячейку *Условие отбора* для поля *Дата* введите текст приглашения в квадратных скобках [**Введите дату**];
- сохраните запрос под именем *Выборка за любой день*;
- откройте запрос. Появится окно диалога с текстовым приглашением *Введите дату*. Введите значение даты *2.09.04* и нажмите на кнопку *ОК*. Так можно просмотреть сведения за любую дату в одном запросе;
- закройте запрос.

Создание запроса Статистика

Назначение запроса — рассчитать общее количество товара и сумму таможенной пошлины за день. Запрос с сортировкой, группировкой и определением промежуточных итогов. Сортировка по полю *Дата*. Записи группируются по наименованию товара, единице измерения и дате провоза. Для поля *Пошлина* и поля *Количество* выполняется групповая операция *Sum*.

Порядок выполнения запроса:

- создайте новый запрос с помощью *Конструктора запросов*;
- таблица-источник — запрос *Пошлина*;
- добавьте поля в соответствии с рис. 10;
- по полю *Дата* выполните сортировку *по возрастанию*;
- на панели инструментов нажмите на кнопку с изображением знака суммы;
- в бланке запроса появится строка *Групповая операция*;
- выполните щелчок по ячейке *Группировка* для поля *Пошлина*. В раскрывающемся списке выберите статистическую функцию *Sum* (рис. 10);
- аналогично примените статистическую функцию *Sum* для поля *Количество*;
- сохраните запрос с именем *Статистика*;
- просмотрите результат. На базе данного запроса можно подготовить статистический отчет в количественном выражении;
- закройте запрос.

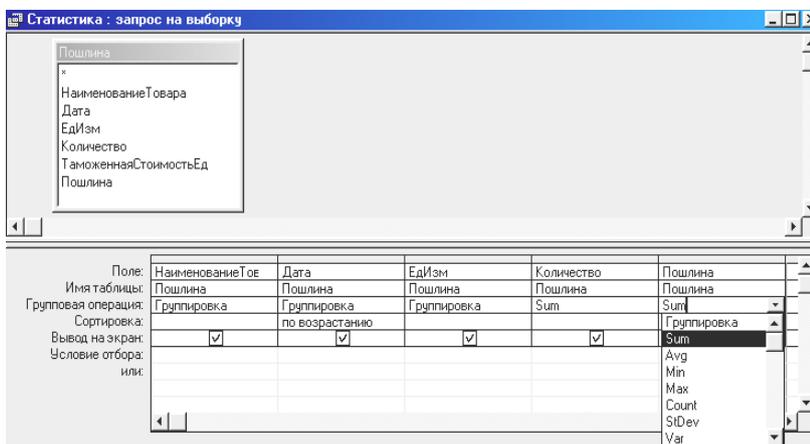


Рис. 10. Окно Конструктора запроса Статистика

3.3. Создание формы и работа с ней

Форма создается для более комфортного ввода данных, получаемых из документа *Счет-фактура*. В форме выполняется редактирование, удаление, добавление и фильтрация записей.

Порядок создания *формы* и работа с ней:

- ☐ выберите вкладку *формы*, нажмите на кнопку *Создать*;
- ☐ в окне диалога *Новая форма* выберите пункт *Автоформа: ленточная*;
- ☐ в этом же окне в раскрывающемся списке источников данных выберите таблицу *Счет-фактура*, нажмите на кнопку *OK*;
- ☐ создается мгновенная форма. Каждая запись отображается в одну строку. *Форма* включает все поля таблицы *Счет-фактура*.
- ☐ в нижней части *Автоформы* имеется набор кнопок со стрелками, позволяющими перемещаться по записям;
- ☐ в последнюю строку, помеченную звездочкой, вводятся новые записи. Введите две записи, предложенные в табл. 3;
- ☐ удалите ошибочно введенную запись из *Счет-фактуры № 43*. Для этого выделите запись, и нажмите на кнопку *Del* на клавиатуре;

Таблица 3. Записи для дополнения формы *Счет-фактура*

Номер счет-фактуры	Дата	Код товара	Единица измерения	Количество	Таможенная стоимость, ед.
6	03.09.04	87	шт.	1	2 900,00 р.
32	03.09.04	91	шт.	300	150,00 р.

- ☐ отредактируйте последнюю запись, изменив *Количество* 300 шт. на 100;
- В Microsoft Access существует понятие фильтра. С помощью фильтра проводится извлечение подмножества записей из базовой таблицы или запроса по определенному критерию. Фильтр обычно используется при работе в режиме формы или в режиме таблицы. Для фильтрации записей по критерию выполните следующее:
 - ☐ выполните щелчок по кнопке мыши в поле *КодТовара* по любой ячейке с кодом товара 87;
 - ☐ нажмите на кнопку *Фильтр по выделенному*  на панели инструментов. В форме останутся только записи с кодом товара 87;
 - ☐ отожмите утопленную кнопку *Удалить фильтр*;
 - ☐ количество записей восстановится;
 - ☐ сохраните форму под именем *Счет-фактура*;
 - ☐ закройте форму.

3.4. Создание Отчета

С помощью *Мастера отчетов* можно подготовить к выводу на печать выходной документ *Отчет за сентябрь*. В отчете подсчитывается сумма пошлины за каждый день (промежуточный итог по полю *Пошлина*) и за весь период анализа (общий итог по полю *Пошлина*).

Порядок создания отчета:

- ☐ выберите вкладку *Отчеты*, нажмите на кнопку *Создать*;

■ в диалоговом окне *Новый отчет* выделите пункт *Мастер отчетов* и выберите в качестве источника запрос *Статистика*. Нажмите на кнопку *ОК*;

■ в окне диалога *Создание отчетов* все поля поочередно переместите из списка *Доступные поля* в список *Выбранные поля*. Чтобы переместить поле, его нужно выделить и нажать кнопку со стрелкой, направленной вправо. Нажмите на кнопку *Далее*;

■ в следующем окне выполняется группировка. Выделите поле *Дата* и нажмите на кнопку со стрелкой;

■ в этом же окне нажмите на кнопку *Группировка* и измените интервал группировки, выбрав пункт *по дням*. Нажмите на кнопку *ОК*, выполните команду *Далее*;

■ в следующем окне задается сортировка. Сортировка производится не более чем по четырем полям. Выберите в первом раскрываемом списке поле *Дата*, во втором *НаименованиеТовара*. По умолчанию принят порядок сортировки *по возрастанию*;

■ в этом же окне нажмите на кнопку *Итоги*. В окне диалога *Итоги* выберите операцию *Sum* для поля *Пошлина*;

■ в группе *Показать* установите переключатель *Данные и итоги*, нажмите на кнопку *ОК*, выполните команду *Далее*.

■ в следующем окне выберите макет *Блок*, ориентацию *Альбомную* и нажмите на кнопку *Далее*;

■ в следующем окне выберите *Стиль Простой*, выполните команду *Далее*;

■ в последнем окне задайте имя отчета *Отчет за сентябрь*, нажмите на кнопку *Готово*;

■ в результате вы увидите сформированный отчет в режиме просмотра. *Мастер отчетов* сгруппировал записи по полю *Дата* и добавил итоги за каждый день и за анализируемый период. Для удобства просмотра запроса можно уменьшить масштаб. Полученный отчет представлен на рис. 11;

■ сохраните отчет;

■ полученный отчет можно вывести на печать, выполнив команду *Файл / Печать / ОК*.

Дата по дням	Дата	НаименованиеТовара	ЕдИзм
1 Сентябрь 2004 г.	01.09.04	Транспортные средства	штук
	01.09.04	Часы	штук
Итоги для 'Дата' = 01.09.04 (2 записей)			
Sum			
2 Сентябрь 2004 г.	02.09.04	Мебель	штук
	02.09.04	Транспортные средства	штук
	02.09.04	Часы	штук
Итоги для 'Дата' = 02.09.04 (3 записей)			
Sum			
3 Сентябрь 2004 г.	03.09.04	Транспортные средства	штук
	03.09.04	Часы	штук
Итоги для 'Дата' = 03.09.04 (2 записей)			
Sum			
ИТОГО			

Рис. 11. Отчет, созданный с помощью *Мастера отчетов*

Лабораторное занятие 2 «Проектирование и создание реляционной базы данных *Успеваемость студентов* в среде СУБД ACCESS»

Цель — научиться создавать структуру реляционной базы данных и объекты доступа к данным средствами СУБД Access.

Объем — 2 часа.

Задание — разработать базу данных *Успеваемость студентов*.

Описание предметной области

База данных *Успеваемость студентов* проектируется и создается для методиста деканата заочного отделения. Информационными объектами служат студенты, дисциплины, оценки.

Нормативно-справочная информация представлена справочниками студентов и дисциплин. Оперативно-учетной информацией являются сведения об оценках, регистрируемые в документе *Экзаменационная ведомость*.

Деканат выписывает ведомость, регистрирует ее в специальном журнале под уникальным номером. Для каждой группы на каждый экзамен выписываются новые экзаменационные ведомости. Для пересдачи экзаменов также выписываются отдельные ведомости. Информация об успеваемости студентов накапливается в БД в течение учебного года.

Для учебной базы предлагается сокращенный перечень реквизитов документа *Экзаменационная ведомость*. Макет документа приведен на рис. 12.

Экзаменационная ведомость № _____

Дата сдачи экзамена _____
Дисциплина _____

Фамилия	Код студента (номер зачетной книжки)	Оценка
---------	---	--------

Рис. 12. Макет документа *Экзаменационная ведомость*

На основе данных, имеющихся в базе, решается ряд регламентированных задач, например:

- получение сведений об успеваемости каждого студента;
- расчет среднего балла успеваемости;
- формирование отчета по результатам сдачи сессии.

Макет выходного документа представлен на рис. 13.

Отчет по результатам сессии

Фамилия	Информатика	Высшая математика	Микроэко- номика	Средний балл
---------	-------------	----------------------	---------------------	-----------------

Рис. 13. Макет документа *Отчет по результатам сессии*

Задания

Данная работа предполагает выполнение следующих этапов:

1. Проектирование БД.

1.1. Составление генерального списка полей и определение для каждого поля типа данных с описанием свойств.

1.2. Определение информационных объектов (таблиц) и выполнение нормализации.

1.3. Распределение полей генерального списка по таблицам.

1.4. Определение ключевых полей.

1.5. Указание связей и типов отношений.

2. Создание БД.

2.1. Загрузка СУБД Access.

2.2. Создание структур таблиц.

2.3. Установление между таблицами связей с типом отношений и обеспечением целостности данных.

3. Использование БД.

3.1. Заполнение таблиц данными.

3.2. Выполнение следующих запросов:

• *Учет успеваемости* — вывод оценок, полученных студентами по всем дисциплинам, с указанием даты сдачи экзамена.

• *Успеваемость студентки Толмачей* — вывод оценок студентки Толмачей.

• *Средний балл* — расчет среднего балла успеваемости каждого студента.

• *Результаты сессии* — вывод оценок студентов в разрезе дисциплин.

• *Отчет по результатам сессии* — вывод оценок студентов в разрезе дисциплин и среднего балла успеваемости студентов.

3.3. Создание формы *Оценки* для ввода и редактирования данных, получаемых из документа *Экзаменационная ведомость*.

3.4. Подготовка к выводу на печать выходного документа *Отчет по результатам сессии*.

Порядок выполнения задания

1. Проектирование базы данных

1.1. Определение генерального списка полей

На основании перечня реквизитов, входящих в документы и необходимых для решения задачи, определен следующий генеральный список полей:

• *КодСтудента* (номер зачетной книжки): тип данных *Текстовый* (вычисления с кодом студента не производятся), уникальный для каждого студента;

• *Фамилия*: тип данных *Текстовый*. Возможно, что одинаковую фамилию имеют несколько студентов;

- *КодВедомости*: тип данных *Текстовый*. Номер ведомости не повторяется только в течение одного учебного года. В следующем учебном году повторения могут быть. Поэтому логичнее использовать не номер ведомости, а код дисциплины, составив его из двух последних цифр года и номера ведомости. Такой код будет уникальным для каждой ведомости:

- *Дисциплина*: тип данных *Текстовый*;
- *КодДисциплины*: по условию задачи в БД должна храниться информация о перечне дисциплин. Для связи таблицы с перечнем дисциплин с другими таблицами, введем поле *КодДисциплины*, уникальный для каждой дисциплины. Тип данных *Текстовый*;
- *Дата сдачи*: тип данных *Дата/время*;
- *Оценка*: тип данных *Числовой*.

1.2. Определение Информационных объектов (таблиц) и выполнение нормализации

По условию задачи в БД должны храниться справочные сведения о студентах и дисциплинах. Поэтому создание таблиц *Студент* и *Дисциплина* очевидно.

Учетной информацией является экзаменационная оценка. Информация об оценке будет полной, если указать, кто получил оценку, по какой дисциплине, когда и в каком документе это отображено. Такая информация представлена в виде таблицы на рис. 14.

<i>КодВедомости</i>	<i>КодСтудента</i>	<i>Оценка</i>	<i>Дата</i>	<i>КодДисциплины</i>
0211	Э2170	4	03.01.04	111
0211	Э2171	5	03.01.04	111
0211	Э2172	3	03.01.04	111
0211	Э2118	3	03.01.04	111
0223	Э2170	4	07.01.04	112
0223	Э2171	4	07.01.04	112
0223	Э2172	4	07.01.04	112
0223	Э2118	2	07.01.04	112
0238	Э2170	4	15.01.04	113
0238	Э2171	3	15.01.04	113
0238	Э2172	2	15.01.04	113
0238	Э2118	4	15.01.04	113

Рис. 14. Вид таблицы с данными об оценках до выполнения нормализации

Обратите внимание, что значения полей *КодВедомости*, *Дата* и *КодДисциплины* повторяются столько раз, сколько студентов перечислено в одной ведомости. Очевидно, что таблица имеет избыточное дублирование данных. Чтобы исключить избыточность, необходимо выполнить нормализацию.

Для нормализации таблицу необходимо разбить на две (рис. 15). Связь между таблицами установится через поле *КодВедомости*.

Оценки			Ведомость		
<i>КодВедомости</i>	<i>КодСтудента</i>	<i>Оценка</i>	<i>КодВедомости</i>	<i>Дата</i>	<i>КодДисциплины</i>
0211	Э2170	4	0211	03.01.04	111
0211	Э2171	5	0223	07.01.04	112
0211	Э2172	3	0238	15.01.04	113
0211	Э2118	3			
0223	Э2170	4			
0223	Э2171	4			
0223	Э2172	4			
0223	Э2118	2			
0238	Э2170	4			
0238	Э2171	3			
0238	Э2172	2			
0238	Э2118	4			

Рис. 15. Нормализованные таблицы с данными об оценках студентов

На основании вышесказанного определились информационные объекты (таблицы): *Студент*, *Дисциплина*, *Ведомость* и *Оценки*.

Таблицы *Студент* и *Дисциплина* заполняются справочной информацией один раз, и по мере необходимости производится редактирование, добавление и удаление записей.

Таблица *Ведомость* заполняется по мере выписки экзаменационных ведомостей накануне сдачи экзаменов.

После сдачи экзамена и возврата экзаменационной ведомости в деканат в установленный внутренними правилами срок осуществляется ввод данных в таблицу *Оценки*. В таблице *Оценки* накапливаются оценки всех студентов по всем дисциплинам.

1.3. Распределение полей генерального списка по следующим таблицам:

- *Студент* (КодСтудента, Фамилия);
- *Дисциплина* (КодДисциплины, Дисциплина);
- *Ведомость* (КодВедомости, Дата, КодДисциплины);
- *Оценки* (КодВедомости, КодСтудента, Оценка).

1.4. Определение ключевых полей

В соответствии с условием задачи определяются ключевые поля:

- в таблице *Студент* ключевое поле — *КодСтудента*. Код студента однозначно идентифицирует студента. Возможно, что одну фамилию имеют несколько студентов, но коды у них разные;
- в таблице *Дисциплина* ключевое поле — *КодДисциплины*. Каждая дисциплина в справочнике дисциплин упоминается только один раз;
- в таблице *Ведомость* ключевое поле — *КодВедомости*, уникальный для каждой ведомости.
- в таблице *Оценки* простого ключевого поля нет. Возможно установление составного ключа из полей *КодВедомости* и *КодСтудента* при условии что все повторные сдачи экзаменов оформляются в новой ведомости.

1.5. Связи и тип отношений

Разработаем информационно-логическую модель базы, на основе которой будет создаваться *Схема данных*. Пара таблиц связывается по одинаковому полю. Главной является та таблица, в которой поле связи является ключом. Отношения *Один ко многим* устанавливаются, когда ключ главной таблицы связывается с неключевым полем или частью составного ключа подчиненной таблицы. Связи и тип отношений для нашей задачи показаны на рис.16.



Рис. 16. Таблицы с логически завершённым перечнем полей, связями и типом отношений

Все связи между таблицами имеют тип *Один ко многим*. Это значит, например, что одна запись в таблице *Студент* может быть связана с несколькими записями в таблице *Оценки*, т. е. в сессию студент получает оценки по нескольким дисциплинам.

Одна запись в таблице *Дисциплина* может быть связана с несколькими записями таблицы *Ведомость*, т. е. одну дисциплину сдают студенты разных групп по разным ведомостям также одну дисциплину можно сдавать несколько раз.

Одна запись в таблице *Ведомость* может быть связана с несколькими записями таблицы *Оценки*, т. е. в одной ведомости могут быть указаны результаты сдачи экзаменов несколькими студентами.

2. Создание БД *Успеваемость студентов*

2.1. Загрузка СУБД Access

Для запуска Microsoft Access, необходимо выполнить следующие действия:

- в *Главном меню* в пункте *Программы* выберите *Microsoft Access*;
- в разделе *Создание базы данных* установите переключатель *Новая база данных*;
- появится диалоговое окно *Файл новой базы данных*. Выберите папку (например *STUD*);
- присвойте файлу имя *Успеваемость* и свою фамилию, (например, *Успеваемость-Иванов*), нажмите на кнопку *Создать*.

Появится окно *База данных*, которое не закрывается в течение всего сеанса работы с данной базой. Оно является контейнером, содержащим все объекты базы данных: *Таблицы, Запросы, Формы, Отчеты, Макросы, Модули*.

2.2. Создание структур таблиц

Сначала следует создавать таблицы со справочной, а затем с учетной информацией.

Рекомендованный порядок создания таблиц:

- выбрать способ создания таблицы;

- присвоить имена полям;
- задать тип данных и выбрать свойства;
- присвоить ключ;
- сохранить таблицу.

Создание структуры таблицы Студент

Порядок создания:

- сделайте активной вкладку *Таблицы* и нажмите на кнопку *Создать*. В диалоговом окне *Новая таблица* выберите способ создания — *Конструктор*, нажмите на кнопку *ОК*;
- откроется окно *Конструктора таблиц*;
- введите имена полей с клавиатуры в соответствии с рис.17. Обратите внимание на то, что имена полей написаны без пробелов и каждое слово пишется с большой буквы;
- выполните щелчок по кнопке мыши в ячейке *Тип данных* для поля *КодСтудента*. Из раскрывающегося списка выберите тип данных *Текстовый*. По умолчанию предложен *Текстовый* тип данных.
- в *Свойствах поля* установите размер для поля *КодСтудента* — 8 (рис. 17);

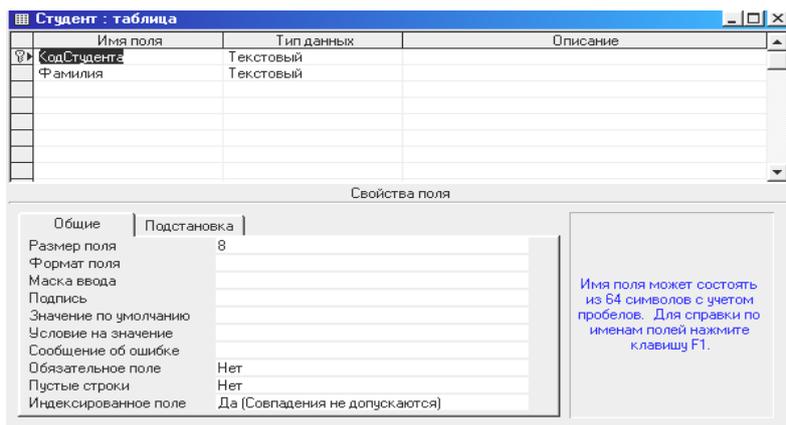


Рис. 17. Структура таблицы Студент

■ для присвоения *Ключа* выполните щелчок по кнопке мыши в ячейке *КодСтудента* и нажмите на панели инструментов на кнопку с изображением ключа. Слева от поля *КодСтудента* появится значок ключа;

■ для сохранения таблицы на панели инструментов выполните щелчок по кнопке *Сохранить*. В открывшемся окне диалога введите имя таблицы *Студент*, нажмите на кнопку *ОК*. Закройте окно *Конструктора таблиц*.

■ Аналогично создайте структуры таблиц *Дисциплина*, *Ведомость* и *Оценки*. Имена полей, Типы данных и свойства предложены в табл. 4.

■ В таблице *Оценки* ключевого поля нет. При сохранении таблицы в окне диалога появится вопрос *Создать ключевое поле сейчас*. Нажмите на кнопку *Нет*.

Таблица 4. Информация для создания структур таблиц

Имя таблицы	Имена полей	Тип данных	Свойства	
			ключевое поле	размер поля
<i>Дисциплина</i>	<i>КодДисциплины</i>	Текстовый	Ключевое поле	3
	<i>Дисциплина</i>	Текстовый		По умолчанию
<i>Ведомость</i>	<i>КодВедомости</i>	Текстовый	Ключевое поле	6
	<i>Дата</i>	Дата/время		
	<i>КодДисциплины</i>	Текстовый		3
<i>Оценки</i>	<i>КодВедомости</i>	Текстовый		6
	<i>КодСтудента</i>	Текстовый		8
	<i>Оценка</i>	Числовой		По умолчанию

2.2. Установление связей с типом отношений и обеспечением целостности данных

Для установления связей выполните следующее:

■ нажмите на кнопку *Схема данных* на панели инструментов. Появится окно *Схема данных* и в нем окно *Добавление таблицы*. Окно *Добавление таблицы* сместите вниз за заголовок для увеличения обзора;

- выделите таблицу *Студент* и нажмите на кнопку *Добавить*. Аналогично добавьте все таблицы. Закройте окно *Добавление таблицы*;
- для установления связи перетащите ключевое поле *КодСтудента* из главной таблицы (*Студент*) и бросьте на поле *КодСтудента* в подчиненной таблице (*Оценки*);
- появится окно диалога *Связи*, в котором установите флажки *Обеспечение целостности данных*, *Каскадное обновление связанных полей*, *Каскадное удаление связанных полей* (это делает невозможным случайное удаление или изменение связанных данных), нажмите на кнопку *Создать*. Тип отношения установлен *Один ко многим* (рис. 18);
- аналогично установите все остальные связи в соответствии с рис. 18;
- Закройте и сохраните окно *Схема данных*.

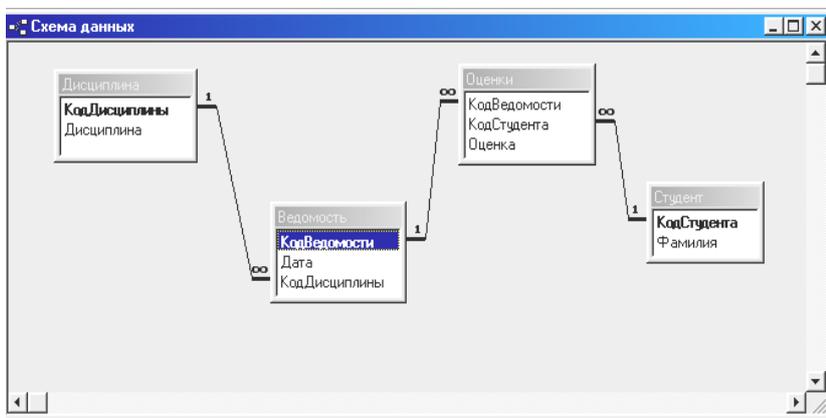


Рис. 18. Схема данных БД *Успеваемость студентов*

3. Использование базы данных

3.1. Заполнение таблиц данными

Порядок выполнения:

- в окне *База данных* выделите таблицу *Студент* и нажмите на кнопку *Открыть*;
- заполните таблицу *Студент* данными согласно табл. 5. Сохраните и закройте таблицу.

Таблица 5. Исходные данные для заполнения таблицы *Студент*

<i>КодСтудента</i>	<i>Фамилия</i>
Э2118	Голмачей
Э2170	Нестеров
Э2171	Петрова
Э2172	Ростова

- аналогично заполните данными таблицы *Дисциплина*, *Ведомость*, *Оценки* в соответствии с таблицами 6–8.

Таблица 6. Исходные данные для заполнения таблицы *Дисциплина*

<i>КодДисциплины</i>	<i>Дисциплина</i>
111	Информатика
112	Микроэкономика
113	Высшая математика

Таблица 7. Исходные данные для заполнения таблицы *Ведомость*

<i>КодВедомости</i>	<i>Дата</i>	<i>КодДисциплины</i>
0411	03.01.04	111
0423	07.01.04	112
0438	15.01.04	113

Таблица 8. Исходные данные для заполнения таблицы *Оценки*

КодВедомости	КодСтудента	Оценка
0411	Э2118	3
0411	Э2170	4
0411	Э2171	5
0411	Э2172	3
0423	Э2118	2
0423	Э2170	4
0423	Э2171	4
0423	Э2172	4
0438	Э2118	4
0438	Э2170	4
0438	Э2171	3
0438	Э2172	2

3.2. Создание запросов

Создание запроса *Учет успеваемости студентов*

В запросе показать оценки, полученные студентами по всем дисциплинам, с указанием даты. Запрос многотабличный на выборку.

Порядок создания запроса:

- ☑ активизируйте вкладку *Запросы* и нажмите на кнопку *Создать*;
- ☑ в окне *Новый запрос* выберите пункт *Конструктор*, нажмите на кнопку *ОК*;
- ☑ появится окно *Конструктора запросов* и окно *Добавление таблицы*. Добавьте все таблицы. Закройте окно *Добавление таблицы*.

На экране останется окно *Конструктора запросов* с именем *Запрос1: запрос на выборку*.

Описание окна *Конструктора* дано на страницах 30–31.

Порядок заполнения бланка запроса для создания запроса *Учет успеваемости*:

- ☑ в ячейки строки *Поле* добавьте поля *Фамилия*, *КодСтудента*, *Дата*, *Дисциплина* и *Оценки* (рис. 19). Поля добавляются методом перетаскивания или выполнением двойного щелчка по добавляемому полю в таблице-источнике;

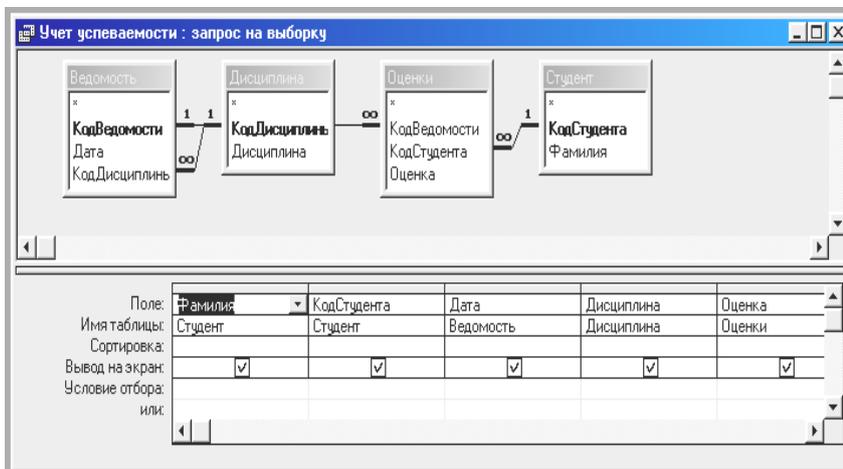


Рис. 19. Окно *Конструктора* для запроса *Учет успеваемости*

- ☑ просмотрите результат. Для этого нажмите на панели инструментов на кнопку *Представление запроса* и выберите *Режим таблицы*;
- ☑ сохраните запрос под именем *Учет успеваемости*;
- ☑ закройте запрос.

Создание запроса *Успеваемость студентки Толмачей*

В запросе требуется показать экзаменационные оценки студентки Толмачей. Это запрос с условием отбора по одному полю.

Порядок создания запроса:

- ☑ выполните команду *Запросы / Создать / Конструктор, ОК*;
- ☑ добавьте запрос *Учет успеваемости*. Окно *Добавление таблицы* закройте;
- ☑ в бланк запроса добавьте все поля из запроса-источника (рис. 20);
- ☑ в ячейку условия отбора для поля *Фамилия* введите — *Толмачей* (без кавычек) и подтвердите ввод щелчком по кнопке мыши в любом месте бланка. Кавычки появятся после подтверждения;
- ☑ просмотрите результат;
- ☑ сохраните запрос под именем *Успеваемость студентки Толмачей*;
- ☑ закройте запрос.

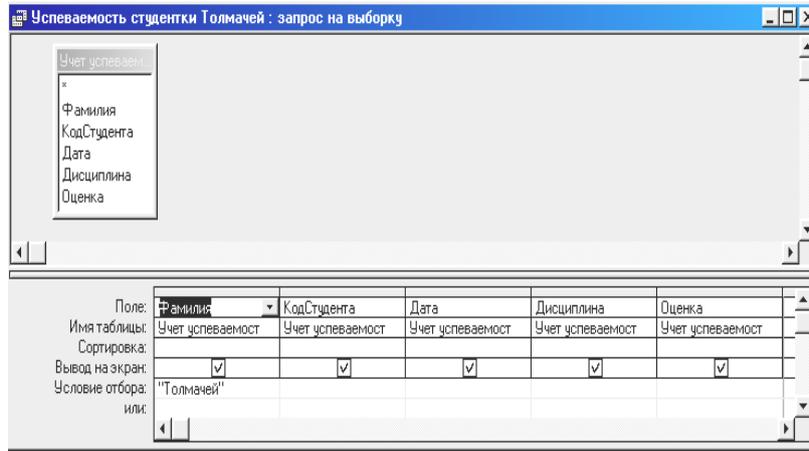


Рис. 20. Создание запроса *Успеваемость студентки Толмачей*

Создание запроса *Средний балл*

В запросе рассчитайте средний балл успеваемости каждого студента. Запрос с группировкой и использованием встроенных функций над группами записей.

Порядок создания запроса:

- ☑ выполните команду *Запросы/Создать/Конструктор, ОК*;
- ☑ добавьте таблицы *Студент* и *Оценки*. Окно *Добавление таблицы* закройте;
- ☑ в бланк запроса добавьте поля: *Фамилия* и *Оценка* в соответствии с рис. 21;
- ☑ выполните сортировку. Для этого из раскрывающегося списка поля *Фамилия* ячейки *Сортировка* выберите пункт *по возрастанию*;
- ☑ нажмите на кнопку с изображением знака суммы (Σ) на панели инструментов. В бланке запроса появится новая строка *Групповая операция*;

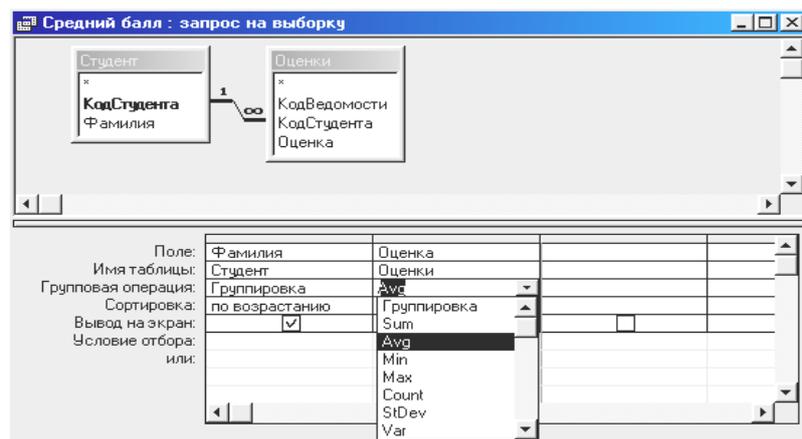


Рис. 21. Создание запроса *СреднийБалл*

- ☑ из раскрывающегося списка ячейки *Группировка* поля *Оценка* выберите пункт *Avg* (среднее от значений поля);
- ☑ просмотрите результат;
- ☑ для задания подписи столбца со средним баллом щелкните правой кнопкой мыши по полю *Avg_Оценка* и в окне *Свойства* введите *Подпись Средний балл*. В этом же окне при необходимости установите формат *Фиксированный*, *Число десятичных знаков* — *1*;
- ☑ сохраните запрос под именем *Средний балл*;
- ☑ закройте запрос.

Создание запроса Результаты сессии

В запросе показать оценки каждого студента в разрезе дисциплин. Запрос перекрестный.

В перекрестном запросе результаты группируются по двум наборам данных. Первый набор образует заголовки строк, а второй заголовки столбцов. Результаты запроса выводятся в виде таблицы.

Порядок выполнения перекрестного запроса:

- добавьте таблицы и поля в *Конструктор запросов*, как показано на рис. 22.
- в меню *Запрос* выберите пункт *Перекрестный*. В бланке запроса появятся две строки: *Групповая операция* и *Перекрестная таблица*.
- из раскрывающихся списков выберите пункты, как показано на рис. 22;

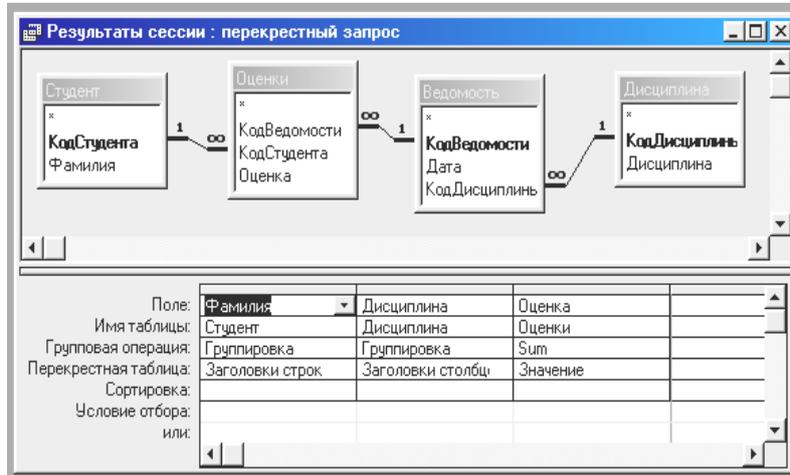


Рис. 22. Создание перекрестного запроса *Результаты сессии*

- просмотрите результат;
- сохраните запрос под именем *Результаты сессии*;
- закройте запрос.

Создание запроса Отчет по результатам сессии

В запросе показать оценки студентов и средний балл успеваемости. Запрос на выборку.

- создайте запрос самостоятельно в соответствии с рис. 23;
- обратите внимание, что запрос составлен на основе двух запросов и связь между запросами не установлена. В окне конструктора запросов установите связь без типа отношений, как показано на рис. 23;

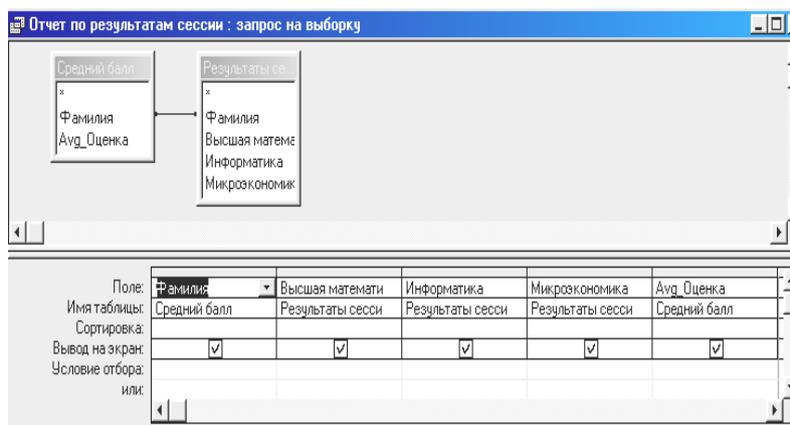


Рис. 23. Создание запроса *Отчет по результатам сессии*

- просмотрите результат и закройте запрос с сохранением.

3.3. Создание Формы

Создадим Форму для более комфортного ввода и редактирования данных, получаемых из документа *Экзаменационная ведомость*.

Порядок создания:

- ☐ выберите вкладку *Формы*, нажмите на кнопку *Создать*;
- ☐ в окне диалога *Новая форма* выберите пункт *Автоформа: ленточная*;
- ☐ в этом же окне в раскрывающемся списке источников данных выберите таблицу *Оценки*, и нажмите на кнопку *ОК*;
- ☐ создается мгновенная *форма*. Каждая запись отображается в одну строку. *Форма* включает все поля таблицы *Оценки*;
- ☐ аналогично создайте *Автоформу: ленточную* для таблицы *Ведомость*;
- ☐ в последней строке формы *Ведомость*, помеченную звездочкой, зарегистрируйте ведомость, например с номером 0456 для передачи студенткой Ростовой дисциплины *Математика* (табл. 9);
- ☐ сохраните и закройте форму.

Таблица 9. Запись для дополнения формы *Ведомость*

КодВедомости	Дата	КодДисциплины
0456	30.01.04	113

- ☐ в последнюю строку формы *Оценки* введите запись (табл. 10) о передаче экзамена студенткой Ростовской;

Таблица 10. Запись для дополнения формы *Оценки*

КодВедомости	КодСтудента	Оценка
0456	Э2172	3

- ☐ сохраните форму под именем *Оценки*;
- ☐ закройте форму.

3.3. Создание Отчета

С помощью *Автоотчета* можно подготовить к выводу на печать выходной документ *Отчет по результатам сессии*, в котором будут отображены оценки студентов в разрезе дисциплин и средний балл.

Порядок создания отчета:

- ☐ выберите вкладку *Отчеты*, нажмите на кнопку *Создать*;
- ☐ в диалоговом окне *Новый отчет* выберите пункт *Автоотчет: ленточный*, в качестве источника выберите запрос *Отчет по результатам сессии* и нажмите на кнопку *ОК*;
- ☐ создается мгновенный отчет. Сформированный отчет показан в режиме просмотра. Для удобства просмотра можно уменьшить масштаб;
- ☐ сохраните отчет под именем *Отчет по результатам сессии*;
- ☐ откройте отчет в режиме *Конструктор*;
- ☐ в верхнем колонтитуле замените надпись *Avg_Оценка* на надпись *Средний балл*;
- ☐ просмотрите результат. Вид полученного отчета приведен на рис. 24.

Фамилия	Высшая математика	Информатика	Микроэкономика	Средний балл
Нестеров	4	4	4	4
Петрова	3	5	4	4
Ростова	2	3	4	3
Толмачей	4	3	2	3

Рис. 24. Автоотчет *Отчет по результатам сессии*

- ☐ для вывода на печать откройте отчет в режиме просмотра и выполните команду *Файл/Печать/ОК*.

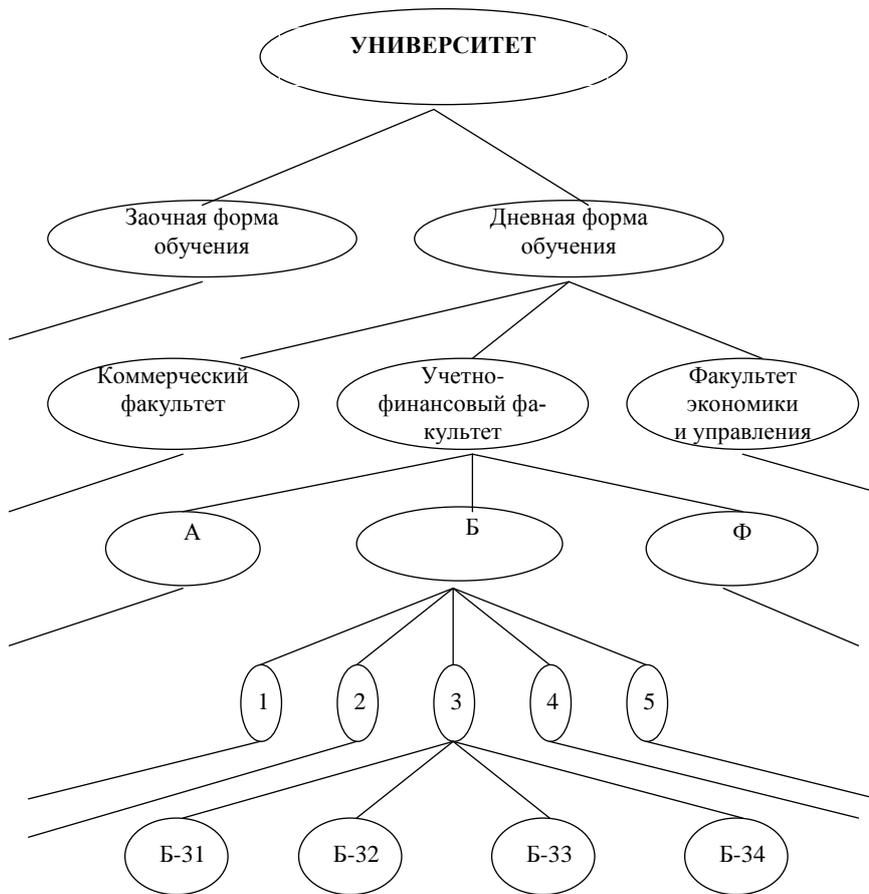
ВОПРОСЫ ДЛЯ САМОКОНТРОЛЯ

1. Назовите преимущества организации баз данных по сравнению с файловыми средами данных.
2. Дайте понятие информационного объекта. Приведите примерный перечень информационных объектов конкретной предметной области.
3. Какую процедуру следует осуществить перед созданием базы данных. Что такое информационно-логическая модель? Перечислите известные виды моделей данных.
4. Опишите сущность иерархической модели данных и приведите пример.
5. Опишите сущность сетевой модели данных и приведите пример.
6. Дайте понятие реляционной модели данных и перечислите ее основные понятия.
7. Каким базовым требованиям должно соответствовать отношение?
8. Опишите структуру реляционной таблицы-отношения. Что такое ключ? Какие виды ключей могут быть определены в реляционной модели.
9. Какие виды связей могут быть установлены между таблицами в реляционной базе данных?
10. Что такое СУБД, структура СУБД? Перечислите основные функции СУБД.
11. Дайте характеристику СУБД Access.
12. Определите понятие проектирования базы данных? Цель проектирования, участники, методы логического проектирования?
13. Опишите общий порядок создания реляционной базы данных в среде СУБД Access.
14. Перечислите объекты доступа к данным СУБД Access и определите их назначение.
15. Какие типы данных могут храниться в базе данных СУБД Access?
16. Какие свойства полей в СУБД Access обеспечивают контроль правильности ввода данных?
17. Опишите порядок конструирования таблиц в СУБД Access.
18. Раскройте назначение Схемы данных Access.
19. Опишите порядок создания Схемы данных Access.
20. Определите назначение и виды форм в СУБД Access.
21. Какие виды запросов реализуются в СУБД Access? Перечислите разновидности запросов на выборку.
22. Опишите технологию создания многотабличного запроса на выборку в СУБД Access.
23. Как получить новое поле в запросе путем вычисления из имеющихся в базе данных полей?
24. Как формируется запрос с группировкой? Какие статистические функции применяются для групповых операций?
25. Опишите назначение отчетов и порядок создания простейших отчетов в СУБД Access.

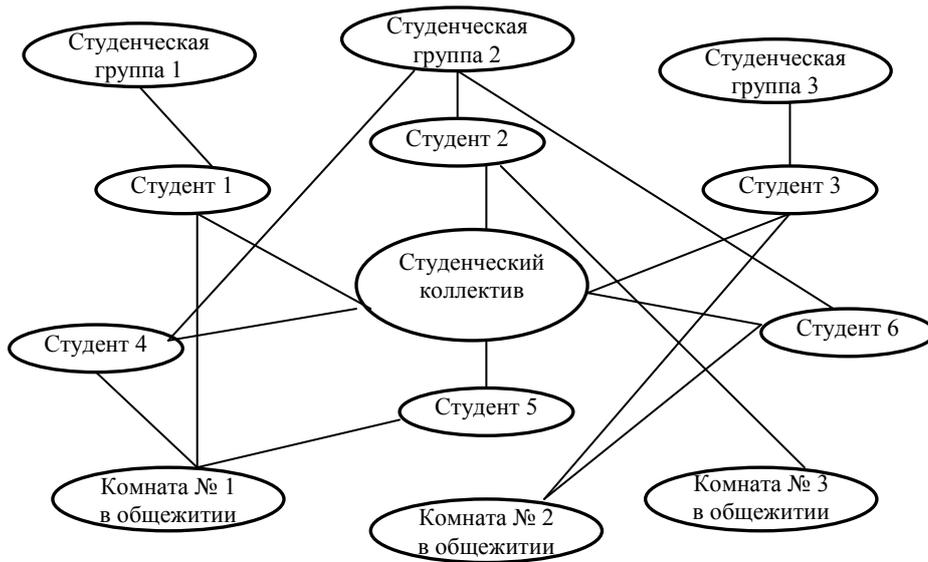
СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

1. Автоматизированные информационные технологии в экономике: Учебник / М. И. Семенов, И. Т. Трубилин, В. И. Лойко. Т. II. Барановская; Под ред. И. Т. Тубилина. — М.: Финансы и статистика, 2000. — 416 с.
2. Автоматизированные информационные технологии в экономике: Учебник / Под ред. Г. А. Титоренко. — М.: Компьютер: ЮНИТИ, 1998. — 400 с.
3. Автоматизация работы пользователя в среде СУБД Access: Пособие для студентов всех специальностей: В 2 ч. Ч. 1 / Авторы-составители: Л. М. Ашарчук, Т. В. Астапкина, И. В. Дубинина. — 2-е изд. с изм. — Гомель: Белорусский торгово-экономический университет потребительской кооперации, 2001. — 48 с.
4. Бекаревич Ю. Б., Пушкина Н. В. СУБД Access для Windows 95 в примерах. — СПб.: Санкт-Петербург, 1991. — 400 с.
5. Вейскас Дж. Эффективная работа с Microsoft Access 2000. — СПб.: Питер, 2000.
6. Винтер Р. Microsoft Access 97: Справочник. — СПб.: Питер, 1999. — 416 с.
7. Джонс Э., Джонс Дж. М. Access 97: Книга ответов. — СПб.: Питер, 1988. — 400 с.
8. Дубнов П. Ю. Access 2000. Проектирование баз данных. — М.: ДМК, 2000.
9. Золотова С. И. Практикум по Access. — М.: Финансы и статистика, 2001.
10. Игнатюк А. З. Таможенное право Республики Беларусь — Мн.: Амалфея, 2002. — 400 с.
11. Когаловский М. Р. Энциклопедия технологий баз данных / М. Р. Когаловский. — М.: Финансы и статистика, 2002.
12. Коннолли Т., Бегг К, Страчан А. Базы данных: проектирование, реализация и сопровождение. Теория и практика: Учебное пособие: Пер. с англ. — 2-е изд. — М.: ИД «Вильямс», 2000. — 1120 с.
13. Левчук Е. А., Заяц Т. А. Построение информационно-логической модели данных и ее реализация в СУБД Access: Пособие для студентов всех специальностей. — Гомель: Белорусский торгово-экономический университет, 2003. — 120 с.
14. ER-метод проектирования баз данных и его реализация в среде СУБД Access: Пособие для студентов всех специальностей / Авторы-составители: С. М. Мовшович, К. Г. Сулейманов. — Гомель: Белорусский торгово-экономический университет потребительской кооперации, 2002. — 138 с.
15. Харитонова И. А., Михеева В. Д. Microsoft Access 2000. — СПб.: БНВ-Санкт-Петербург, 2000. — 1088 с.
16. Экономическая информатика: Учебник / Под ред. В. П. Косарева, Л. В. Еремينا. — М.: Финансы и статистика, 2001. — 592 с.

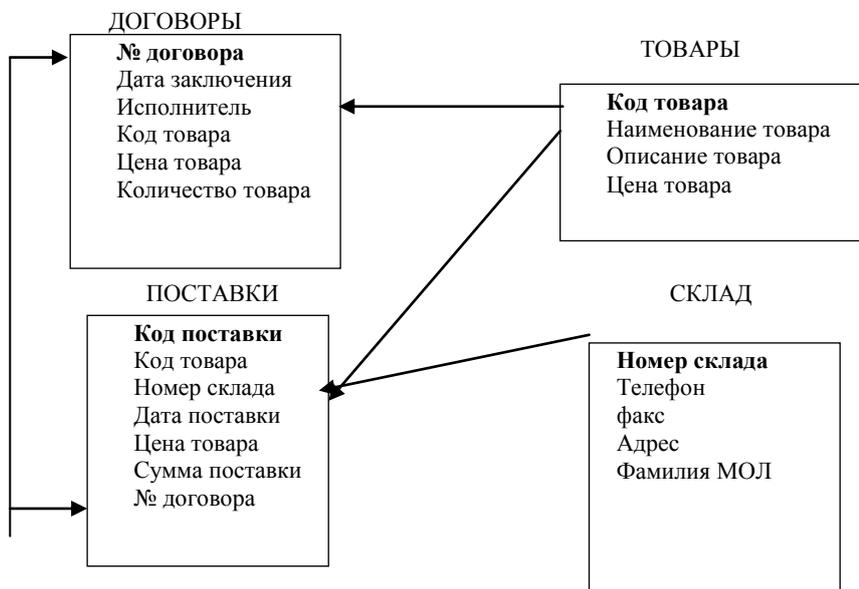
Иерархическая модель данных



Сетевая модель данных



Реляционная концептуальная схема информационной модели «Фирма — Поставщик»



СОДЕРЖАНИЕ

Пояснительная записка	3
1. Теоретические основы разработки базы данных.....	4
1.1. Понятие и виды информационного обеспечения автоматизированных информационных систем	4
1.2. Основные понятия баз данных	5
1.3. Информационные объекты и модели данных.....	7
1.4. Системы управления базами данных.....	10
2. Технология ведения баз данных.....	12
2.1. Реляционная база СУБД Access	12
2.2. Свойства полей и типы данных СУБД Access	14
2.3. Схема данных СУБД Access	16
2.4. Этапы проектирования и создания базы данных СУБД Access...	18
3. Лабораторное занятие 1 «Проектирование и создание реляционной базы данных <i>Таможня</i> с помощью СУБД Access».....	20
4. Лабораторное занятие 2 «Проектирование и создание реляционной базы данных <i>Успеваемость студентов</i> с помощью СУБД Access».....	22
Вопросы для самоконтроля	32
Список рекомендуемой литературы	59
Приложения	60

Учебное издание

ТЕХНОЛОГИИ ОРГАНИЗАЦИИ, ХРАНЕНИЯ И ОБРАБОТКИ ДАнных

Практикум
для студентов заочной формы обучения
экономических специальностей

Авторы составители: **Ашарчук** Лилия Михайловна
Костюченко Галина Леонидовна

Редактор *О. В. Ивановская*
Компьютерная верстка *И. А. Козлова*

Подписано в печать 18.02.05. Формат 60×84¹/₁₆.
Бумага тип. № 1. Гарнитура Таймс.
Усл. печ. л. 3,74. Уч.-изд. л. 3,4. Тираж 750 экз.
Заказ №

УО «Белорусский торгово-экономический
университет потребительской кооперации»
ЛИ № 02330/0056814 от 02.03.2004 г.
246029, г. Гомель, просп. Октября, 50.

Отпечатано на ризографе
УО «Белорусского торгово-экономического
университета потребительской кооперации»
246029, г. Гомель, просп. Октября, 50.