**Методические указания и варианты заданий по учебной практике**

**Задание 1**.

Содержание.

Проектирование концептуальной модели базы данных:

1. Создание базы данных.
2. Создание запросов на языке SQL.

**ВАРИАНТЫ ЗАДАНИЙ**

1. Информационная система «Экологическая служба» содержит данные об отделениях ЭС (название отделения, город, телефон, год создания), о предприятиях, по вине которых произошли экологические катастрофы (ЭК) (название предприятия, город, отрасль промышленности (химическая, металлургическая, деревообрабатывающая, …), год создания, телефон), экологических катастрофах (предприятие, тип катастрофы (выброс окиси углерода, сброс отходов в реку, утечка отравляющих веществ, взрыв, пожар,…), дата, количество пострадавших людей, последствия (вид(гибель животных, гибель людей, отравление, радиоактивное заражение, загрязнение атмосферы,…), материальный ущерб)), сведения об участии ЭС в устранении ЭК (отделение ЭС, количество людей, принимавших участие, количество дней).
2. Для автоматизации учета шефской помощи предприятиями детским садам города необходимы следующие сведения: о детских садах (номер детсада, название, район города, тип собственности (государственный, частный, предприятия,…), год открытия, телефон, дети (ФИО, группа (младшая, средняя, старшая, подготовительная), дата рождения)), о предприятиях, оказывающих помощь (название, форма собственности (государственное, частное, ЗАО, ОАО, ООО,…), телефон, год начала работы, количество сотрудников), об оказанной помощи детским садам (вид помощи (поставка продуктов, мебели, медикаментов, ремонт, оплата экскурсии,…), стоимость, дата оказания) и о помощи отдельным детям (вид помощи (оплата лечения, денежная помощь, покупка детского питания, покупка одежды,…), стоимость, дата оказания) .
3. В информационной системе об ассортименте книжных магазинов города обрабатываются данные о магазинах (номер магазина, район города, тип собственности (государственный, частный, ОАО, ЗАО, ООО,…), год открытия), книгах (название книги, автор (ФИО, страна, дата рождения, дата смерти), издательство (название издательства, город, год создания, телефон), год издания, язык, в какие магазины поступила (цена экземпляра, количество экземпляров, дата поступления в магазин)).
4. База даннях «Кондитерские фабрики» содержит такую информацию: название фабрики, город, телефон, год введения в строй, тип собственности (ОАО, ЗАО, ООО, частная, государственная,), выпускаемая фабрикой продукция (наименование продукции, тип (карамель, ирис, шоколад, мармелад, печенье …), расфасовка (коробка, весовая, штучная,…)), поступления партий продукции в магазины (магазин (номер или название, район города, год открытия, телефон, число сотрудников), продукция, дата поступления, объем (в кг или штуках), цена единицы (кг или штуки)).
5. База данных «Зоопарки» содержит следующую информацию: название зоопарка, тип (частный, государственный, ОАО, ЗАО, ООО,…), город, год открытия, телефон, сотрудники (ФИО, дата рождения, должность, оклад), имеющиеся животные (кличка, класс (земноводные, пресмыкающиеся, птицы, млекопитающие, рыбы,…), вид (медведь, волк, слон,…), родина (указать часть света, остров, океан,….), средняя продолжительность жизни, сотрудник, под наблюдением которого находится животное, дневной рацион (количество кг), дата поступления в зоопарк, случаи оказания животным медицинской помощи (вид помощи (прививка, операция, лечение, …), стоимость, дата оказания)).
6. Информационная система о функционировании кинологических клубов города должна обрабатывать данные о клубах (название клуба, район города, год создания, телефон, вступительный взнос), сведения о собаках, которые прикреплены к клубам (кличка, порода, год рождения, стоимость щенка, хозяин (ФИО, социальное положение (служащий, учащийся, предприниматель, пенсионер, рабочий, …), дата рождения, адрес)), сведения об участии собак в соревнованиях (соревнование (название соревнования, дата проведения, взнос, количество зрителей), награда (вид награды, денежный эквивалент)).
7. Система учета кредитной деятельности банков должна содержать такие данные: название банка, тип (государственный, коммерческий, акционерный, …), год создания, уставной фонд, телефон, сведения о выданных кредитах (клиент (ФИО или название фирмы, город, телефон, номер счета), вид кредита (потребительский, под векселя, под ценные бумаги, краткосрочный, долгосрочный,…), сумма, годовой процент, дата выдачи), возврат кредитов (сумма, дата возврата).
8. Для автоматизации учета издательской деятельности необходимы сведения об издательствах (название, город, тип собственности (государственное, частное, ООО, ОАО, ЗАО,…), год образования, телефон, адрес) и об изданных книгах (название книги, автор (фамилия, страна, дата рождения, дата смерти), жанр (детектив, фантастика, исторический, приключения,…), язык издания, цена, год издания, переплет (твердый, мягкий, целлофанированный), тип издания (отдельное произведение, собрание сочинений, избранные произведения), тираж).
9. База данных «Ботанические сады» включает следующую информацию: название ботанического сада, город, тип (частный, государственный,…), год открытия, сотрудники (ФИО, должность, дата рождения, оклад), растения (номер растения, номер участка произрастания, семейство (лилейные, крестоцветные, розоцветные, пасленовые, бобовые,…), вид (тюльпан, гиацинт, лилия, яблоня, груша, роза, боярышник,…), срок жизни (количество лет), дата посадки, стоимость саженца, сотрудник, ухаживающий за растением), сведения об удобрении растений (удобрение (название, объем упаковки, стоимость упаковки), количество упаковок, дата выполнения).
10. Для системы, автоматизирующей учет наличия лекарств в аптеках города, необходимы следующие сведения: наименование лекарства, фирма- производитель (название фирмы, тип собственности (государственная, частная, акционерная,…), страна, год образования, адрес), форма выпуска лекарства (капли, таблетки, мазь,…), фармакологическая группа (сердечно-сосудистое, болеутоляющее, противовоспалительное,…), в какие аптеки поступило лекарство (сведения об аптеке (номер аптеки, тип (государственная, частная, акционерная,…), район города, телефон), дата поступления партии в аптеку, количество упаковок лекарства в партии, цена одной упаковки).

Примечание: для упрощения можно считать, что каждое лекарство производится только одной фирмой.

1. Для автоматизации учета пользователей электронной почты необходима информация о фирмах-провайдерах (название фирмы, тип собственности (государственная, частная , ЗАО, ОАО,…), адрес, телефон, год начала работы), о заключенных с ними договорах (абонент (ФИО или название предприятия, тип (частное лицо, ВУЗ, школа, предприятие,…), физический адрес, адрес электронной почты), дата подключения, стоимость подключения, стоимость пересылки 1 Mb информации) и о предоставленных провайдерами услугах (абонент, дата предоставления, объем сообщения (в Мb)).
2. Информационная система, предназначенная для учета результатов техосмотров автомобилей, должна содержать: сведения о ГАИ (район города, телефон, адрес, сотрудники (ФИО, звание, год рождения)), владельцах автомобилей (ФИО, социальное положение (предприниматель, служащий, учащийся, рабочий, пенсионер, домохозяйка,…), дата рождения, телефон, автомобили (номер, марка, год приобретения, цена)), техосмотрах (дата, стоимость, кто проводил, выявленные недостатки (вид (неотрегулированное освещение, плохие тормоза, люфт руля, отсутствие аптечки, разбитые фары, …), штраф)).

Примечание: каждый владелец автомобилей прикреплен только к одному ГАИ.

1. Для автоматизации работы администратора гостиницы необходима информация о номерах гостиницы (номер, этаж, тип (люкс, полулюкс, отдельный, номер на несколько человек,…), размер оплаты за сутки), о вселении в гостиницу (ФИО, город проживания, дата рождения, социальное положение (банковский служащий, предприниматель, рабочий, учащийся,…), дата поселения, на сколько суток), о предоставленных дополнительных услугах (название услуги (использование телевизора, холодильника, междугородний телефонный разговор, отправка телеграммы,…), цена за условную единицу (условные единицы: сутки, минута, слово,…), дата предоставления, продолжительность (количество условных единиц)).
2. Система обработки информации о деятельности страховых компаний использует сведения о компаниях (название компании, тип (частная, государственная, акционерная,…), год получения лицензии, город, в котором находится центральное отделение, телефон, филиалы (название филиала, город, адрес, телефон, количество сотрудников)) и заключенных с филиалами договорах (клиент (ФИО, дата рождения, социальное положение (служащий, предприниматель, учащийся, рабочий, пенсионер,…), телефон), вид страхования (недвижимость, имущество, от несчастного случая, к бракосочетанию, до совершеннолетия,…), сумма страхования, дата заключения договора).

Примечание: клиент может заключать договора с филиалами разных компаний.

1. Информационная система «Мясокомбинаты области» должна обрабатывать сведения о мясокомбинатах (название мясокомбината, город области, тип собственности (частный, государственный, ООО, ЗАО, ОАО,…), год начала работы, телефон, выпускаемая мясокомбинатом продукция (вид продукции (буженина, колбаса, рулет, корейка, паштет,…), название, годовой объем выпуска (в кг), цена 1 кг)) и заказах на продукцию (заказчик (название организации, город, телефон, адрес), дата заказа, объем).
2. Для учета деятельности рекламных агентств используются сведения об агентствах (название агентства, город, тип (частное, государственное, акционерное,…), год создания, расчетный счет, телефон, виды производимой в агентстве рекламной продукции (видеоклип, радиорепортаж, рекламный щит, реклама на общественном транспорте, в газете, журнале, листовка,…)) и заказах на изготовление рекламной продукции (заказчик (название предприятия или ФИО, город, телефон, адрес), вид рекламной продукции, цена экземпляра, количество экземпляров, дата заказа, срок изготовления (в днях), форма расчета (наличная, безналичная)).

Примечание: заказчик может заказывать различные виды рекламы в разных агентствах.

1. База даннях «Станции техобслуживания города» включает информацию о станциях (название станции, форма собственности (государственная, частная, ОАО, ЗАО, ООО,…), район расположения, год открытия, ремонтные услуги, оказываемые станцией (вид ремонта (замена масла, регулировка тормозной системы, регулировка электрооборудования, переборка двигателя, замена рессор,…), номинальная стоимость ремонта)), о владельцах автомобилей, пользующихся услугами станций (ФИО, автомобили (номер, марка, стоимость), социальная группа (предприниматель, банковский служащий, инженер, рабочий,…), адрес, телефон) и о произведенных на станциях ремонтах (вид ремонта, где и когда производился, реальная стоимость ремонта).
2. Для автоматизации учета работы СОБЕСов используются данные о самих учреждениях (район города, год открытия, количество сотрудников, адрес, телефон, клиенты (ФИО, дата рождения, тип (ветеран ВОВ, ветеран труда, инвалид 1 группы, инвалид 2 группы, инвалид 3 группы, пенсионер,…), размер пенсии, имеющиеся льготы (бесплатный проезд в общественном транспорте, бесплатный проезд в междугородних автобусах, в пригородном транспорте, 50% оплата квартиры, 50% оплата коммунальных услуг, бесплатные медикаменты, 50% оплата за медикаменты,…))), о предоставлении клиентам СОБЕСов помощи (вид помощи (денежная, продуктовая, предметы быта, оплата лечения, санаторная путевка,), денежный эквивалент, дата предоставления).
3. Для автоматизации деятельности органов государственной криминалистики необходимы следующие сведения о нарушениях: анкетные данные правонарушителя (ФИО, дата рождения, номер дела в картотеке, адрес последнего места проживания, судимости (дата, название суда (районный, городской, областной, верховный), статья (206- хулиганство, 93- умышленное убийство, 84 - кража государственного имущества, 140 - кража имущества граждан,…), количество лет, тип приговора (лишение свободы, ссылка, исправительные работы, штраф, оправдательный приговор,…), место заключения (колония строгого режима, колония общего режима, колония для несовершеннолетних,…), стоимость процесса)), жалобы осужденных на решение суда (дата подачи жалобы, ее направленность (несогласие со свидетельскими показаниями, несогласие с протоколом №.., несогласие с приговором,…), в какой суд подана, дата разбирательства жалобы, результат (отменить приговор, подтвердить, изменить)).
4. Для автоматизации обработки информации о работе крупнейших предприятий автомобилестроения необходимы данные об этих предприятиях (название, страна, тип собственности (государственное, частное, ЗАО, ОАО, ООО,…), год начала функционирования, телефон, факс, выпускаемые автомобили (марка, год начала выпуска, цена единицы, технические характеристики (мощность, максимальная скорость, расход топлива))) и о заказах на автомобили (заказчик (ФИО или название предприятия, страна, город, телефон), марка автомобиля, количество единиц, дата заказа).
5. Система учета поставки овощей в магазины города содержит информацию о предприятиях, производящих продукцию (название предприятия, район области, тип собственности (государственная, частная, ЗАО, ОАО, ООО,…), число сотрудников, год начала работы, телефон, производимая предприятием продукция (вид (огурцы, помидоры, картофель, капуста,…), наименование сорта)), и о закупках продукции (магазин (название или номер магазина, район города, адрес, телефон), продукция, количество килограмм, цена килограмма, дата поставки).
6. Для автоматизации деятельности горно-спасательной службы (ГСС) Донбасса необходимы данные об отделениях службы (название отделения, город, телефон, год создания, сотрудники (ФИО, должность (командир отряда, командир взвода, командир роты, боец,…), оклад, с какого года работает в отделении, дата рождения)) и шахтах (название шахты, город, максимальная глубина, площадь выработок, участки (номер, длина, тип выработки (вертикальная, горизонтальная, наклонная), тип угля (антрацит, коксующийся, энергетический, рядовой, …), год ввода в действие, чрезвычайные происшествия на участках (тип (обвал лавы, пожар, взрыв газа, затопление, …), причина (объективная, субъективная), дата, материальный ущерб, сотрудники ГСС, участвующие в устранении ЧП (сколько дней работал каждый из них)))).
7. Для автоматизации учета обеспеченности населения города хлебо - булочными изделиями необходима информация о предприятиях, выпекающих хлеб (название или номер предприятия, район города, тип собственности (государственное, частное, ООО, ОАО, ЗАО,…), год начала работы, телефон, выпускаемая предприятием продукция (название продукции (хлеб “Дарницкий”, хлеб “Ромашка”, хлеб с отрубями, бублик, булочка с изюмом, батон,…), сорт муки (белая высшего сорта, белая первого сорта, ржаная, с отрубями, ...), вес единицы)) и о поставках в магазины (магазин (номер или название магазина, район города, адрес, телефон), продукция, количество единиц, цена единицы, дата поставки).
8. База данных «Банки города» содержит сведения об отделениях банков (номер (название) отделения, банк (название банка, тип (государственный, акционерный, коммерческий,…), уставной фонд, операции с валютой (возможны или нет), сфера обслуживания (юридические лица, физические лица, те и другие), дата получения лицензии), телефон, район города, в котором находится отделение, год открытия) и выполненных операциях (клиент (ФИО или наименование организации, юридическое или физическое лицо, телефон, номер счета, дата открытия счета), вид операции (банковский перевод, инкассо, аккредитив отзывной, аккредитив безотзывной,…), сумма, дата выполнения).
9. Для учета заболеваемости детей, посещающих детские сады города, необходимы следующие сведения о детях: ФИО, группа (младшая, средняя, старшая, подготовительная), дата рождения, пол, посещаемый детский сад (номер, название, район города, тип собственности (государственный, частный, предприятия,…), год открытия), имеющиеся прививки (против каких болезней и когда сделаны), перенесенные заболевания (название заболевания, дата начала, продолжительность болезни (количество дней), стоимость лечения, полученные осложнения (новое заболевание (воспаление легких, ангина, …), стоимость лечения)).

**Методические указания к выполнению задания 1**

**ПРОЕКТИРОВАНИЕ КОНЦЕПТУАЛЬНОЙ МОДЕЛИ БАЗЫ ДАННЫХ**

**Цель работы:** изучение вопросов представления данных об объектах предметной области и их связях в рамках реляционной модели.

Задание к работе

Для предметной области, описанной в заданном варианте, выделить объекты и представить их взаимосвязанными таблицами, находящимися в третьей нормальной форме. Определить базовые таблицы и таблицы- справочники; указать для них первичные и внешние ключи.

Контрольные вопросы

1. Сформулировать следующие определения:
   * предметная область, БД, СУБД;
   * реляционная модель данных;
   * нормализованная таблица, нормализация;
   * нормальные формы таблиц (первая, вторая, третья);
   * первичный, искусственный, внешний ключи;
   * целостность данных, ссылочная целостность.
2. Виды связей между объектами предметной области.

Основой для всех современных СУБД является реляционная модель данных, базирующаяся на теории множеств и отношений (англ. *relation* – отношение). Отношение имеет простую графическую интерпретацию в виде таблицы. На логическом уровне БД представляет собой множество взаимосвязанных таблиц. Каждая строка ( *запись*, или *кортеж*) таблицы включает данные об одном объекте (например, банке, клиенте, студенте), а столбцы (*поля* или *атрибуты*) таблицы содержат различные характеристики этих объектов (например, наименование банка, фамилия и адрес клиента, номер зачетной книжки студента). Таким образом, *реляционной* называется модель данных, в которой для хранения данных об объектах предметной области и для моделирования взаимосвязей между ними используется набор таблиц.

После анализа предметной области отдельные данные объединяются в таблицы. Такая группировка не имеет единственного решения, поэтому нужно определить оптимальный набор, который обладает лучшими свойствами при включении, удалении и модификации данных, а также позволяет сделать избыточность данных и время ответов на запросы к БД минимальными.

Рассмотрим примеры проектирования концептуальной модели базы данных.

Пример 1

Для автоматизации учета работы молокозаводов области информационная система должна содержать такие данные о заводах: название завода, город, в котором находится завод, год начала функционирования, директор, телефон, цехи (название цеха, дата ввода в действие), выпускаемая продукция (тип продукции (сметана, кефир, ряженка, творог…), жирность, вес, дата выпуска, объем, цена за 1 кг.).

После первоначального анализа данных, характеризующих указанную предметную область, можно выделить объекты

ЗАВОДЫ, ПРОДУКЦИЯ.

Соответствующие им таблицы можно представить так:

ЗАВОДЫ

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Завод** | **Город** | **Начало** | **Дирек- тор** | **Тел** | **Цех** | **Дата ввода** |
| Геркулес | Донецк | 1993 | Иванов. | … | Творожный | … |
| Геркулес | Донецк | 1993 | Иванов | … | Жидких кисломолочных  продуктов |  |
| Геркулес | Донецк | 1993 | Иванов | … | Жидких пастеризованных  продуктов |  |
| Маричка | Горловка | 1999 | Петров | … | Творожный |  |
| Маричка | Горловка | 1999 | Петров | … | Жидких кисломолочных  продуктов |  |
| Маричка | Горловка | 1999 | Петров | … | Жидких пастеризованных  продуктов |  |
| Лактис | Марьинка | !998 | Власов | … | Творожный |  |
| Лактис | Марьинка | !998 | Власов | … | Жидких кисломолочных  продуктов |  |
| … | … | … | … | … | … | … |

ПРОДУКЦИЯ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Тип** | **Жирность** | **Вес** | **Дата выпуска** | **Цена единицы** |
| Молоко Творог Ряженка  …….. | …  …  …  … | ?  ?  ? | ?  ?  ? | ?  ?  ? |

Недостатки выполненного проектирования:

1. Избыточность данных (в таблице «Заводы» многократно повторяются данные о каждом заводе).
2. Неадекватность представления данных «Дата выпуска», «Вес» и «Цена единицы» в таблице «Продукция».
3. Отсутствие связи между таблицами .

Нормализуем таблицу «Заводы», разбив её на 2 таблицы: «Заводы» (в ней будут приведены сведения о заводах; первичный ключ таблицы – поле «Завод») и «Цехи» (сведения о цехах заводов; поле **«**Завод**»** - внешний ключ; первичный ключ - составной: **«**Завод + Цех»).

ЗАВОДЫ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Завод** | **Город** | **Начало** | **Директор** | **Телефон** |
| Геркулес Маричка  Лактис | Донецк Горловка  Марьинка | 1993  1999  1998 | …  …  … | …  …  … |

ЦЕХИ

|  |  |  |
| --- | --- | --- |
| **Завод** | **Цех** | **Дата ввода** |
| Геркулес Геркулес Геркулес Маричка Маричка  ….. | Творожный  Жидких кисломолочных продуктов Жидких пастеризованных продуктов Творожный  Жидких кисломолочных продуктов  ……. | …  …  …  …  …  … |

Между таблицами «Заводы» и «Цехи» существует связь “один-ко- многим” (один завод имеет несколько цехов). Так как названия цехов повторяются, то целесообразно организовывать справочник названий; в этой таблице поле «Код названия цеха**»** – искусственный ПК.

НАЗВАНИЯ ЦЕХОВ

|  |  |
| --- | --- |
| **Код названия цеха** | **Цех** |
| Т К П  …. | Творожный  Жидких кисломолочных продуктов Жидких пастеризованных продуктов  …….. |

В таблице «Заводы» введем искусственный первичный ключ, чтобы в таблице «Цехи» заменить длинное поле «Завод» коротким полем «Код завода»:

ЗАВОДЫ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Код завода** | **Завод** | **Город** | **Начало** | **Директор** | **Телефон** |
| 1  2  3  … | Геркулес Маричка Лактис  … | Донецк Горловка Марьинка  … | …  …  …  … | …  …  …  …. | …  …  …  … |

Для таблицы «Цехи» составной ПК «Завод + Цех» также нужно заменить простым ключом. Для этого введем искусственный ПК «Код цеха»:

ЦЕХИ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Код цеха** | **Код завода** | **Код названия цеха** | **Дата ввода** | **Количество** |
| 1  2  3  4  5  … | 1  1  1  2  2  … | Т К П Т К  … | …  …  …  …  …  … | …  …  …  …  …  … |

Между объектами ЦЕХИ–ПРОДУКЦИЯ существует отношение “m:n» (с одной стороны, каждый цех выпускает несколько видов продукции, с другой стороны, каждый вид продукции выпускается цехами нескольких заводов). Для представления этой связи необходимо ввести таблицу пересечения.

Разобьем таблицу «Продукция» на три: «Виды продукции», «Ассортимент» и «Выпускаемая продукция». «Код вида продукции», «Код продукции» – искусственные первичные ключи.

ВИДЫ ПРОДУКЦИИ

|  |  |
| --- | --- |
| **Код вида продукции** | **Вид** |
| 1  2  3  … | Сметана Кефир Ряженка  … |

АССОРТИМЕНТ

|  |  |  |
| --- | --- | --- |
| **Код продукции** | **Код вида продукции** | **Жирность** |
| 1 | 1 | 10 |
| 2 | 1 | 15 |
| 3 | 1 | 20 |
| 4 | 2 | 2,5 |
| 5 | 2 | 3,2 |
| 6 | 2 | 1 |

ВЫПУСКАЕМАЯ ПРОДУКЦИЯ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Код цеха** | **Код продукции** | **Дата выпуска** | **Объем** | **Цена** |
| 1  1  1  5 | 2  3  3  2 | 01.02.2013  01.02.2013  01.02.2013  02.02.2013 | 120  210  220  112 | 17.90  21.20  26.40  19.00 |

Таблица «Выпускаемая продукция» является таблицей пересечения («Код цеха» – внешний ключ*, «*Код продукции»- внешний ключ).

Естественный первичный ключ – составной: «Код цеха + Код продукции

+ Дата выпуска», поэтому введем короткий искусственный ПК.

Структура рассмотренных таблиц (все находятся в 3НФ) и связи между ними приведены на рис. 1.

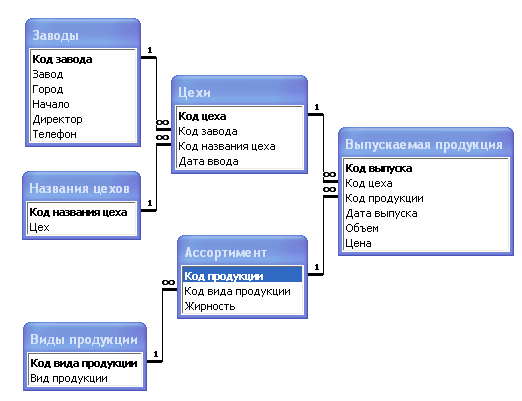


Рисунок 1 Концептуальная модель данных для БД «Продукция молокозаводов»

Пример 2.

Для учета работы посреднической фирмы, занимающейся продажей вычислительной техники, необходимы сведения о поставщиках (название фирмы, город, страна, форма собственности, телефон, адрес) и о представленных ими товарах (наименование товара, фирма-изготовитель, дата поставки партии, цена единицы, количество единиц)

Первоначально рассмотрим представление БД в виде двух таблиц.

Первая таблица – «Поставщики» - выглядит таким образом:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Фирма** | **Город** | **Страна** | **Собственность** | **Телефон** | **Адрес** |
| Альфа | Киев | Украина | частная | 8-044-212-312 | Ул.Парковая, 12 |
| Гамма | Минск | Беларусь | государственная | **…** | **…** |
| Омега | Киев | Украина | акционерная | **…** | **…** |
| **…** | **…** | **…** | **…** | **…** | **…** |

Вторая таблица – «Товары»:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Товар** | **Изготовитель** | **Дата поставки** | **Цена единицы** | **Количество единиц** |
| Принтер | Hewlett Packard | 10.01.12 | 2000.00 | 10 |
| DVD-RW | Acme | 12.01.12 | 4.00 | 100 |
| CD-RW | Memorex | 12.02.12 | 2.00 | 300 |
| CD-R | Acme | 12.04.12 | 2.00 | 100 |
| **…** | **…** | **…** | **…** | **…** |

Данное представление имеет ряд недостатков:

1. Таблицы не связаны, следовательно, нельзя выяснить, кто поставлял заданный товар, какова общая стоимость товаров, представленных поставщиком, и т.п.
2. Избыточность в полях “Город”, “Страна”, “Собственность” – возможно многократное повторение одних и тех. Же заранее известных значений.
3. Присутствует аномалия обновления: если нужно изменить название товара, необходим просмотр всех его поставок.
4. Присутствует аномалия включения: нельзя включить в БД информацию о новом товаре до тех пор, пока не будет выполнена хотя бы одна его поставка/
5. Присутствует аномалия удаления: если некоторый товар исключен из ассортимента поставщиков, то нужно удалять записи обо всех его поставках.
6. Из- за того, что таблица «Поставщики» находится в НФ2 (присутствует зависимость между неключевыми полями «Страна**»** и «Город»), возможно занесение в таблицу заведомо недостоверных данных).
7. В реальной жизни между объектами «Поставщики» и «Товары» имеет место связь «многие ко многим»: один и тот же поставщик может поставлять разные товары, один и тот же товар может поставляться разными поставщиками. Для реализации этой связи введем таблицу пересечения, содержащую сведения о поставках товаров, поскольку явно такая связь представлена быть не может.

Для устранения избыточности в полях «Город», «Страна», «Собственность» введем справочники «Страны», «Города», «Собственность» с искусственными первичными ключами. Для связи включим одноименные внешние ключи в базовые таблицы.

Чтобы устранить аномалии обновления, включения и удаления, выделим отдельную таблицу, содержащую ассортимент товаров, предлагаемых поставщиками (таблица «Каталог»).

Чтобы избежать зависимости между неключевыми полями, информацию о городах и странах будем хранить в двух связанных таблицах.

Для отражения связи между объектами «Поставщики» и «Товары», присутствующей в реальной жизни, создадим таблицу пересечений «Товары», содержащую данные о дате, объеме поставки каждого товара и цене единицы.

Для реализации связей в каждую таблицу включим искусственный первичный ключ, а в подчиненные таблицы – внешний ключ.

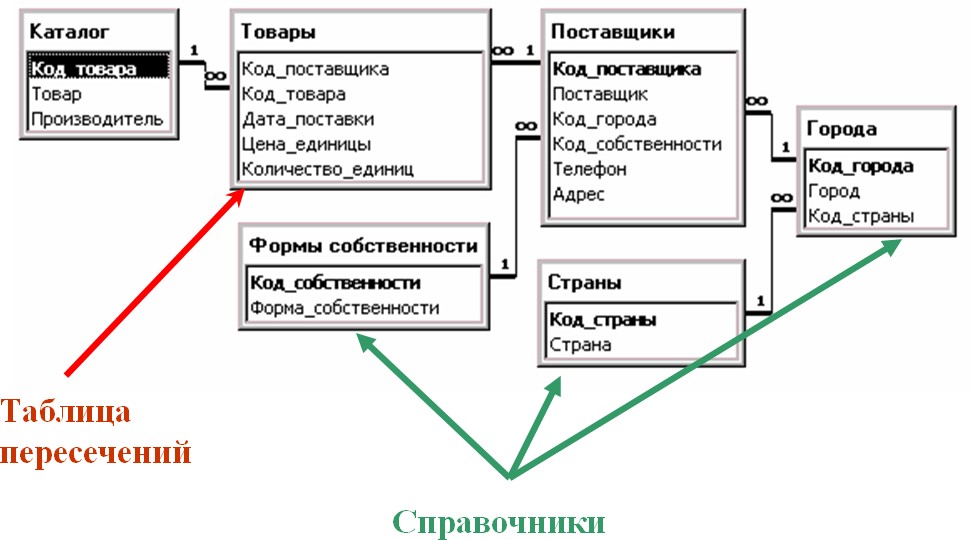
Получим набор таблиц, показанный на рис. 2.

Рисунок 2 – Концептуальная модель данных для БД учета работы посреднической фирмы

1.1. СОЗДАНИЕ БАЗЫ ДАННЫХ

**Цель работы:** освоение возможностей создания таблиц и установления связей между ними.

Задание к работе

Для спроектированной реляционной модели выполнить определение таблиц, указать типы и свойства полей (основные свойства, подстановка), создать схему данных.

Контрольные вопросы

1. Реализация ограничений целостности данных в Access.
2. Реализация связей между таблицами в Access.
3. Каскадное обновление полей, каскадное удаление записей.
4. Свойства полей (общие, подстановка).

При создании базы данных необходимо придерживаться такой последовательности:

* 1. Создание таблиц базы данных, установка свойств полей и таблиц, создание подстановок для внешних ключей. При этом в первую очередь следует создавать главные таблицы, то есть находящиеся на линии связи со стороны «один», а после них – подчиненные.
  2. Установление связей между таблицами, создание схемы данных.
  3. Заполнение таблиц данными.

Создание таблиц базы данных

В начале создадим таблицы «Каталог», «Формы собственности», «Страны», затем таблицы «Города» и «Поставщики», и в последнюю очередь – таблицу «Товары».

При создании таблицы будем использовать режим конструктора – наиболее гибкое средство.

Для создания новой таблицы нужно перейти на вкладку «Создание» и нажать кнопку «Конструктор таблиц» .

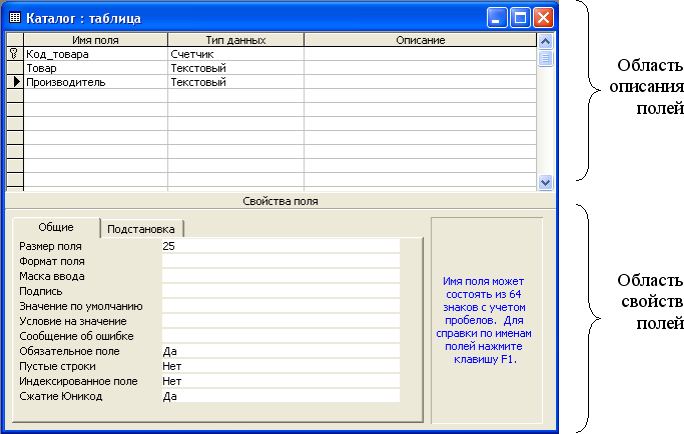
Окно конструктора таблицы показано на рис.3. В верхней части окна размещается список полей таблицы с указанием их типов данных и описаний (при необходимости). В нижней части окна приводятся свойства текущего поля. Набор свойств может быть различным для полей разных типов. Значения некоторых свойств задаются по умолчанию.

Рисунок 3 – Общий вид окна конструктора таблицы

Рассмотрим этапы создания таблицы «Страны».

Зададим имена полей создаваемой таблицы, их типы и свойства полей. Перечень типов полей приведен в приложении А.

Поле «Код\_страны» является искусственным первичным ключом таблицы. Выберем для него тип данных «Счетчик», гарантирующий уникальность значений. Чтобы указать, что поле является искусственным ключом, ….

Поле «Страна» будет содержать только алфавитную информацию, поэтому выберем для него тип данных «Текстовый». Размер поля установим равным 25 символам. Кроме этого, укажем в свойстве «Обязательное поле» значение «Да», а в свойстве «Индексированное поле» выберем значение «Да, совпадения не допускаются». Два последних свойства очень важны, так как по сути это поле является естественным первичным ключом.

Сохраним таблицу, указав при этом ее имя «Страны».

Аналогично создадим таблицы «Каталог» и «Формы собственности».

Далее перейдем к созданию таблицы «Города», которая является подчиненной по отношению к таблице «Страны». При создании поля

«Код\_страны» (внешний ключ) воспользуемся возможностью организовать ввод данных в это поле путем выбора значения из раскрывающегося списка, как это показано на рис. 4. При этом физически в поле внешнего ключа будет сохраняться числовое значение из справочника, а выводиться на экран будет соответствующее ему текстовое значение.

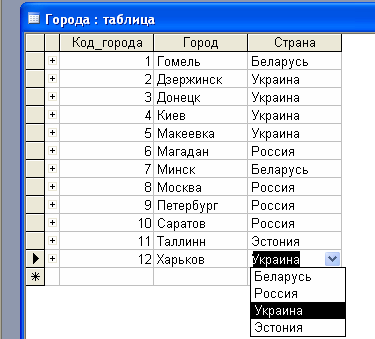


Рисунок 4 – Ввод значения в поле внешнего ключа путем выбора из раскрывающегося списка

Для типа поля внешнего ключа выберем последний элемент списка – «Мастер подстановок». В ходе создания подстановки пользователь отвечает на вопросы надстройки – мастера. Первый такой вопрос связан с источником данных для списка подстановки (см.рис.5), в нашем случае для всех подстановок используются значения из другой таблицы.

Следующим шагом будет выбор таблицы- источника строк подстановки.

В нашем случае это таблица «Страны» (см.рис.6).

|  |  |
| --- | --- |
|  |  |
| Рисунок 5 | Рисунок 6 |

После этого нужно указать поля, участвующие в подстановке. Таких поле должно быть как минимум два. Одно – это поле первичного ключа, значение которого заносится в создаваемое нами поле в результате выполнения подстановки. Второе – соответствующее текстовое значение из связанной таблицы, которое будет выводиться на экран вместо малоинформативного кода. Выполнение данного шага показано на рисунке 7.

На следующем шаге можно указать, каким образом будут упорядочены строки в списке подстановки, после чего скрыть ключевой столбец, содержащий малопонятное для пользователя искусственное значение (рис.8).

|  |  |
| --- | --- |
|  |  |
| Рисунок 7 | Рисунок 8 |

Последним шагом при создании подстановки является задание подписи для поля, однако эту операцию лучше выполнить позже, непосредственно в окне конструктора таблицы.

Аналогично создадим и остальные таблицы нашей базы данных:

«Поставщики» и «Товары».

При создании таблицы «Поставщики» воспользуемся мастером подстановок для полей «Код\_города» и «Код собственности», задав подписи полей «Город» и «Форма собственности». Для поля «Телефон» выберем тип данных «Текстовый»: хотя значения этого поля и содержат цифровую информацию, в вычислениях они участвовать не будут. Для данного поля укажем размер поля- символов и зададим маску ввода, обеспечивающую ввод только цифр и заполнение поля справа налево:

!###\-##\-##;1

Правила формирования маски ввода и символы, которые для этого используются, приведены в приложении В.

В приложении Б приведены форматы вывода значений разных типов.

При создании таблицы «Товары» обратим внимание на поля «Дата\_поставки», «Цена\_единицы» и «Количество\_единиц».

Для поля «Дата\_поставки» выберем тип «Дата/время», укажем значения свойств «Формат поля», «Маска ввода», «Подпись», «Значение по умолчанию»,

«Условие на значение», «Сообщение об ошибке», «Обязательное поле».

Для полей «Цена\_единицы» и «Количество\_единиц» укажем значения свойств «Формат поля», «Подпись», «Условие на значение», «Сообщение об ошибке», «Обязательное поле».

В таблице 1 приведено описание полей таблицы «Товары».

Таблица 1. Свойства полей таблицы «Товары».

|  |  |  |
| --- | --- | --- |
| **Имя поля** | **Тип данных** | **Свойства** |
| Код\_поставки | Счетчик | Первичный ключ |
| Код\_поставщика | Числовой | Размер поля: длинное целое |
| Код\_товара | Числовой | Размер поля: длинное целое |
| Дата\_поставки | Дата/время | Подпись: Дата поставки  Формат поля: Краткий формат даты Маска ввода: 99/99/00 Обязательное поле: Да  Значение по умолчанию: Date() Условие на значение: <=Date() Сообщение об ошибке: Неправильная  дата! |
| Цена\_единицы | Числовой | Размер поля: Одинарное с плавающей Подпись: Цена единицы  Число десятичных знаков: 2 Формат поля: Денежный Обязательное поле: Да Условие на значение: >0  Сообщение об ошибке: Цена д.б. положительной |
| Количество\_единиц | Числовой | Размер поля: длинное целое Подпись: Количество единиц Обязательное поле: Да Условие на значение: >0  Сообщение об ошибке: Количество д.б. положительным |

Создание схемы данных

После определения таблиц нужно установить связи между ними. В дальнейшем Access будет использовать эти связи для поддержания целостности данных и при создании запросов по нескольким таблицам. Связи устанавливаются в окне «Схема данных», которое вызывается кнопкой «Схема данных» на вкладке «Работа с базами данных»: .

В окне нужно выполнить следующие действия:

* щелкнуть по кнопке «Отобразить таблицу» ;
* в диалоговом окне «Добавление таблицы» по очереди выбрать необходимые таблицы, дважды щелкнув на имени каждой таблицы;
* закрыть окно «Добавление таблицы».

Чтобы установить связь между двумя таблицами – главной и подчиненной, щелкните по первичному ключу главной таблицы, перетащите и опустите его на соответствующий внешний ключ подчиненной. Когда вы отпустите кнопку мыши, Access откроет окно диалога «Изменение связей» (рис.9). В этом окне содержится информация о полях связи главной и подчиненной таблиц, параметрах обеспечения целостности данных и типе устанавливаемого отношения.

Первоначально нужно включить поддержку целостности данных с помощью опции «Обеспечение целостности данных». Правила поддержки целостности данных будут действовать при выполнении таких операций, как ввод или изменение данных (например, нельзя будет создавать в «сыне» записи, не имеющие «отца»). После этого Access позволит включить опции

«Каскадное обновление связанных полей» и «Каскадное удаление связанных записей».

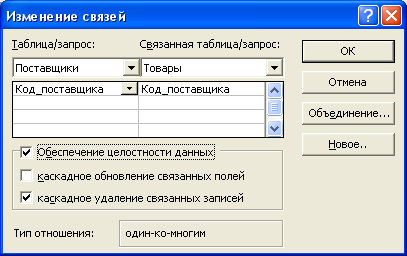


Рисунок 9 – Окно изменения связей

После выбора опций нужно установить связь, щелкнув по кнопке «ОК». После закрытия окна на линии связи возле главной таблицы будет стоять 1, а возле подчиненной – символ бесконечности.

Для редактирования существующей связи нужно дважды щелкнуть по линии связи - снова откроется окно «Изменение связей».

Частично связи создаются в результате работы мастера подстановок. Чтобы вывести эти связи на экран, нужно нажать кнопку «Все связи» . После этого, выполнив двойной щелчок на линии связи, можно изменить свойства этих связей.

На рис.10 показана схема данных, построенная для рассмотренного набора таблиц.

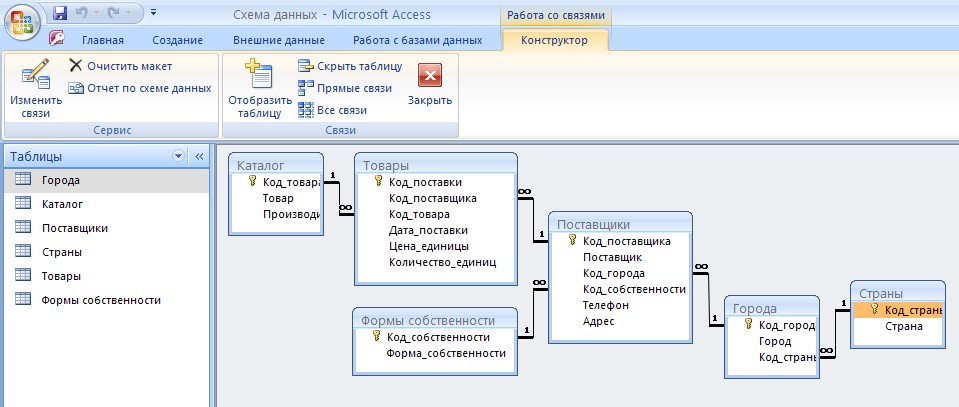


Рисунок 10 – Схема данных в окне Microsoft Access

**1.2. СОЗДАНИЕ ЗАПРОСОВ НА ЯЗЫКЕ SQL**

**Цель работы**: изучение возможностей языка SQL для создания запросов на выборку.

Задание к работе

Создать на языке SQL следующие виды запросов:

* + выборка с условием;
  + симметричное объединение с предикатом DISTINCT;
  + симметричное объединение либо итоговый запрос с предикатом TOP n;
  + итоговый запрос с условием на данные и на группы,
  + перекрестный запрос с условием,
  + подчиненный запрос (использовать операции сравнения и один из предикатов EXISTS, ANY, ALL, IN).

Требования к выполнению

1. SQL – запросы не должны совпадать с имеющимися QBЕ – запросами.
2. Условия отбора должны содержать параметры параметры.

Контрольные вопросы

* 1. Создание на SQL внешних объединений (назначение опций INNER, LEFT, RIGHT); назначение предикатов DISTINCT, ALL, TOP n, TOP n PERСENT.
  2. Создание итоговых и перекрестных запросов.
  3. Подчиненные запросы (назначение и синтаксис предикатов сравнения, EXISTS, IN, ANY, ALL).

Чтобы записать запрос на языке SQL, следует создать запрос в режим конструктора, отказаться от выбора базовых таблиц, вызвать с помощью правой кнопки «мыши» контекстное меню и выбрать пункт **SQL**.

Запрос на языке SQL, как правило, записывается с помощью одной команды. Для всех запросов, выполняющих выборку данных, включая перекрестные и итоговые запросы, эта команда является вариантом одной инструкции языка SQL – инструкции SELECT. Данная инструкция может состоять из нескольких разделов, их набор определяется самим запросом.

В **SQL** для запроса на выборку используется инструкция **SELECT**, обобщенный синтаксис которой имеет вид:

**SELECT** <список - выбора>

**FROM** <список - таблиц>

[ **WHERE** <условие – отбора - записей>]

[ **GROUP BY** <список – полей – для - группировки>] [ **HAVING** <условие – отбора - групп>]

[ **ORDER BY** <условия - сортировки>];

Здесь FROM, WHERE, GROUP BY, HAVING, ORDER BY – предложения SQL.

**FROM** определяет список таблиц, на основе которых строится запрос (обязательное предложение).

**WHERE** указывает условие, которому должны удовлетворять записи результирующего набора.

**GROUP BY** определяет, как должны быть сформированы группы для *итогового* запроса (то есть указывает список полей, по одинаковым значениям которых будет выполняться группировка записей).

**HAVING** используется для указания условия отбора тех групп, которые должны быть включены в результат запроса.

**ORDER BY** указывает условия упорядоченности строк в результирующем наборе.

<**список - выбора**> определяет, какие поля (столбцы) должен иметь результирующий набор записей (наличие <списка> в SQL – запросе обязательно).

Рассмотрим, как записываются на языке SQL различные запросы

Запросы на выборку

При записи простых запросов на выборку в инструкции SELECT могут присутствовать только предложения **FROM, WHERE, ORDER BY.** Перед списком выбора может быть записан предикат, дополнительно ограничивающий результирующий набор записей. Если в условии отбора выполняется сравнение с параметрами, то перед инструкцией SELECT указываются типы этих параметров с помощью инструкции PARAMETERS/

**Пример 1.** Вывести сведения о поставщиках и их товарах, поставленных в указанный период.

Для записи данного запроса в начале опишем два параметра. Сравниваться со значениями параметров будут значения поля Дата/Время, поэтому укажем для параметров соответствующий тип данных SQL – DateTime. Результирующий набор записей будем выводить упорядоченным по дате поставки. Инструкция, реализующая данный запрос, будет выглядеть так:

**PARAMETERS** [Начало периода] DateTime,

[Конец периода] DateTime;

**SELECT** Поставщик, Город, Товар, Дата\_поставки

**FROM** (Города **INNER JOIN** Поставщики

**ON** Города.Код\_города = Поставщики.Код\_города )

**INNER JOIN** (Каталог **INNER JOIN** Товары

**ON** Каталог.Код\_товара = Товары.Код\_товара)

**ON** Поставщики.Код\_поставщика = Товары.Код\_поставщика

**WHERE** Дата\_поставки **Between**[Начало периода] **And**

[Конец периода]

**ORDER BY** Поставщик, Товар, Дата\_поставки;

**Пример 2.** Вывести сведения о пяти поставщиках, которые в течение заданного периода поставили партии товара с наибольшей стоимостью.

Данный запрос отличается от предыдущего двумя особенностями. Во- первых, в него следует включить вычисляемое поле «Стоимость партии» - произведение значений полей «Цена\_единицы» и «Количество\_единиц». Чтобы вывести сведения о поставщиках, представивших пять партий с наибольшей стоимостью, следует упорядочить результирующий набор по убыванию значений вычисляемого поля (ORDER BY 7 DESC – сортировка набора записей по седьмому полю из списка выбора) и взять из него только 5 первых записей (это реализуется с помощью предиката TOP 5**).**

Текст запроса:

**PARAMETERS** [Начало периода] DateTime,

[Конец периода] DateTime;

**SELECT TOP 5** Поставщик, Город, Товар, Дата\_поставки, Цена\_единицы, Количество\_единиц, Цена\_единицы\*Количество\_единиц

**AS** [Стоимость партии]

**FROM** (Города **INNER JOIN** Поставщики

**ON** Города.Код\_города = Поставщики.Код\_города )

**INNER JOIN** ( Каталог **INNER JOIN** Товары

**ON** Каталог.Код\_товара = Товары.Код\_товара)

**ON** Поставщики.Код\_поставщика = Товары.Код\_поставщика

**WHERE** Дата\_поставки Between[Начало периода] **And**

[Конец периода]

**ORDER BY 7 DESC**;

**Пример 3.** Определить, какие виды товаров поставил каждый поставщик в заданном году.

В данном запросе обязательно следует использовать предикат **DISTINCT,** поскольку один и тот же товар мог поставляться одним и тем же поставщиком неоднократно.

Текст запроса:

**PARAMETERS** [Введите год] Short; **SELECT DISTINCT** Поставщик, Товар **FROM** Поставщики **INNER JOIN** (Каталог

**INNER JOIN** Товары **ON** Каталог.Код\_товара = Товары.Код\_товара) **ON** Поставщики.Код\_поставщика = Товары.Код\_поставщика

**WHERE year(**Дата\_поставки)=[Введите год]

**ORDER BY** Поставщик,Товар;

Итоговые запросы

Для итоговых запросов инструкция SELECT должна дополнительно содержать предложения **GROUP BY** и **HAVING**. Эти предложения записываются после предложения **WHERE**, перед предложением **ORDER BY**. В списке выбора такой инструкции должны быть перечислены поля группировки и вычисляемые поля, получаемые с помощью итоговых функций. В разделе **GROUP BY** повторно перечисляются поля группировки. При необходимости в разделе **HAVING** записывается условие на группы.

**Пример 4.** Определить поставщиков, которые в заданный период поставили товаров на сумму, превышающую заданную. Вывести: наименование поставщика, общую сумму за товары, количество партий товаров.

В данном запросе необходимо проверять два условия. Первое условие – дата поставки должна принадлежать заданному периоду. Это условие на данные, оно записывается в разделе **WHERE**. Второе условие – общая стоимость поставок должна превышать некоторую сумму. Это – условие на группы, оно должно быть записано в предложении **HAVING**.

Текст запроса:

**PARAMETERS** [Начало периода] DateTime,

[Конец периода] DateTime, [Укажите сумму] IEEESingle**;**

**SELECT** Поставщик,

**Sum**(Цена\_единицы\*Количество\_единиц) **AS** [Стоимость],

**Count**(Товары.Код\_поставщика) **AS** [Количество партий],

**FROM** Поставщики **INNER JOIN** Товары

**ON** Поставщики.Код\_поставщика = Товары.Код\_поставщика

**WHERE** Дата\_поставки **Between**[Начало периода] **And**

[Конец периода]

**GROUP BY** Поставщик

**HAVING Sum**(Цена\_единицы\*Количество\_единиц)>[Укажите сумму]

**ORDER BY 2 DESC**;

Перекрестные запросы

Для записи перекрестного запроса используется следующая модификация инструкции SELECT:

**PARAMETERS** список параметров;

**TRANSFORM** значение **SELECT** заголовки строк **FROM** список таблиц **WHERE** условие отбора

**GROUP BY** заголовки строк

**ORDER BY** поля сортировки

**PIVOT** заголовки столбцов in (порядок);

Выражение, содержащее итоговую функцию для получения значения в ячейках перекрестной таблицы, записывается в предложении **TRANSFORM.** Как видно, заголовки строк перечисляются дважды: в списке выбора и в предложении **GROUP BY.** Условие отбора при выполнении перекрестного запроса проверяется только для данных, не для групп. Предложения **PARAMETERS** и **WHERE** являются необязательными.

**Пример 5.** Определить для каждого поставщика количество поставленных партий товаров по месяцам задаваемого года и общее количество партий за год.

Особенность данного запроса заключается в порядке вывода на экран столбцов. Названия месяцев года, идущие в естественном порядке, не располагаются ни в алфавитном порядке, ни в порядке, обратном алфавитному. Поэтому укажем явно порядок, в котором должны располагаться столбцы перекрестного запроса, с помощью предложения **IN.**

Текст запроса:

**PARAMETERS** [Укажите год] **Short;**

**TRANSFORM Count**(Товары.Код\_поставщика) **AS** [За месяц]

**SELECT** Поставщик,

**Count**(Товары.Код\_поставщика) **AS** [За год]

**FROM** Поставщики **INNER JOIN** Товары **ON**

Поставщики.Код\_поставщика = Товары.Код\_поставщика

**WHERE Year**(Дата\_поставки) = [Укажите год]

**GROUP BY** Поставщик

**ORDER BY** Поставщик

**PIVOT Format**(Дата\_поставки, "mmm")

**IN** ("янв", "фев", "мар", "апр", "май", "июн",

"июл", "авг","сен","окт","ноя","дек");

Подчиненные запросы

Язык SQL поддерживает возможность включать в условия отбора значения, возвращаемые другим (подчиненным) запросом. В случае, если подчиненный запрос возвращает более одного значения, необходимо использовать предикат **ALL** или **ANY.**

**Пример 6.** Определить поставщиков из заданного города

В данном запросе коды поставщиков из заданного города возвращаются с помощью подчиненного запроса. Использование подчиненного запроса позволяет избежать объединения таблиц.

Текст запроса:

**PARAMETERS** [Введите город] **Text**; **SELECT** Поставщик, Телефон

**FROM** Поставщики

**WHERE** Поставщики.Код\_города **IN**

(**SELECT** Города.Код\_города

**FROM** Города

**WHERE** Город=[Введите город]);

**Пример 7.** Определить товары, у которых цена единицы меньше средней цены всех товаров.

Для проверки условия отбора в данном запросе вначале должен выполниться подчиненный запрос, определяющий среднюю цену товаров.

Текст запроса:

**SELECT** Товар**,**Дата\_поставки, Цена

**FROM** Товары

**WHERE** Цена\_единицы<

(**SELECT Avg** (Цена\_единицы)

**FROM** Товары)

**ORDER BY** 3;

**Пример 8.** Определить партии товаров, в которых цена единицы товара превышает цену указываемого товара (один и тот же товар может иметь в разных партиях различные цены).

В данном случае подчиненный запрос может возвращать множество значений, поэтому в условии отбора используется предикат **ALL.**

Текст запроса:

**PARAMETERS** [Укажите код товара] **Long**; **SELECT** Товары.Код\_товара, Товар, Цена\_единицы, Дата\_поставки

**FROM** Каталог **INNER JOIN** Товары

**ON** Каталог.Код\_товара = Товары.Код\_товара

**WHERE** Цена\_единицы > **ALL**

(**SELECT Distinct** Цена\_единицы

**FROM** Товары

**WHERE** Код\_товара = [Укажите код товара])

**ORDER BY** Код\_товара;

**Пример 9**. Найти производителей товаров, которые ни разу не поставлялись.

В данном запросе необходимо найти записи в таблице «Каталог», у которых нет связанных записей в таблице «Товары». Поскольку один и тот же производитель может выпускать различные товары, воспользуемся предикатом **DISTINCT**.

Текст запроса:

**SELECT DISTINCT** Производитель

**FROM** Каталог

**WHERE** Код\_товара <> **ANY**

(**SELECT** Код\_товара

**FROM** Товары);

**Задание 2.**

**Создание макросов в Access.**

**Задание 2** является продолжением задания 1 по автоматизации работы с базами данных. Следовательно, варианты работ по заданию 2 остаются такими же, что и для выполнения работ по заданию 1.

**Теоретические пояснения по теме задания 2.**

**Создание главной кнопочной формы**

Главная кнопочная форма создается с целью навигации по базе данных. Эта форма может использоваться в качестве главного меню БД. Элементами главной кнопочной формы являются объекты форм и отчётов.

Запросы и таблицы не являются элементами главной кнопочной формы. Поэтому для создания кнопок Запросы или Таблицы на кнопочной форме можно использовать макросы.

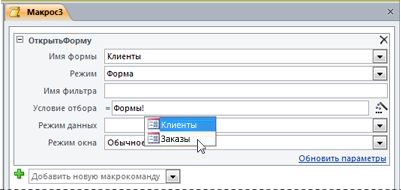
**Макрос в Access** — это инструмент, позволяющий автоматизировать задачи и добавлять функции в формы, отчеты и элементы управления. Например, добавив в форму кнопку, вы можете связать ее событие **OnClick** с макросом, включающим команды, которые должны выполняться при нажатии этой кнопки.

В Access можно рассматривать макросы как упрощенный язык программирования, код на котором создается в виде списка необходимых действий. Создавая макрос, вы выбираете каждое действие из раскрывающегося списка, а затем вводите для него необходимые сведения. С помощью макросов можно добавлять функции в формы, отчеты и элементы управления без необходимости писать код в модуле Visual Basic для приложений (VBA). В макросах доступно подмножество команд VBA, и большинство людей считают, что создать макрос легче, чем написать код VBA.

Предположим, вы хотите создать отчет непосредственно из одной из своих форм для ввода данных. Вы можете добавить в форму кнопку, а затем создать макрос, который открывает отчет. Макрос может быть либо автономным, т. е. отдельным объектом в базе данных, который связывается с событием **OnClick** кнопки, либо внедренным непосредственно в событие **OnClick**. В любом случае при нажатии кнопки макрос запускает и открывает отчет. Такие макросы обычно называют макросами пользовательского интерфейса.

Понятие макросов данных впервые появилось в Access 2010. С помощью макросов данных можно автоматизировать задачи и добавлять функции непосредственно в таблицы. Макросы данных и их действия вкладываются в определенные события таблиц, например добавление новой записи в таблицу.

Создать макрос можно с помощью конструктора макросов, который показан на приведенном ниже рисунке.



Чтобы открыть конструктор макросов пользовательского интерфейса, сделайте следующее:

* На вкладке **Создание** нажмите кнопку **Макрос**.

Чтобы открыть конструктор макросов данных, сделайте следующее:

* Откройте любую таблицу в режиме таблицы. На вкладке **Таблица** в группах **До событий** и **После событий** выберите одну из команд событий, например **До изменения** или **После вставки**.

Чтобы узнать больше о конструкторе макросов и о том, как создавать и тестировать макросы пользовательского интерфейса и данных, см. связанные статьи в разделе **См. также** в конце этого материала.

## Функции макросов в Access

В более ранних версиях Access многие популярные функции не удавалось выполнять без написания программного кода VBA. В текущих версиях Access были добавлены новые функции и действия макросов, избавляющие от необходимости в коде. Это упрощает добавление функций в базу данных и помогает защитить ее.

* **Внедренные макросы**. Вы можете внедрять макросы в любые события, предоставляемые формой, отчетом или элементом управления. Внедренный макрос не отображается в области навигации. Он становится частью формы, отчета или элемента управления, в котором он был создан. Если создать копию формы, отчета или элемента управления, содержащего встроенные макросы, то эти макросы также будут присутствовать в копии.
* **Повышенная безопасность**. Если кнопка **Показать все действия** не выделена в конструкторе макросов, то для использования доступны только те действия макросов и аргументы RunCommand, для выполнения которых не требуется статус доверенной базы данных. Макрос, созданный с использованием этих действий, будет выполняться, только если база данных находится в отключенном режиме (когда выполнение VBA запрещено). Базам данных, которые содержат действия макросов, не включенные в список доверенных (или базам данных с кодом VBA), необходимо явно предоставлять статус доверенных.
* **Обработка ошибок и устранение неполадок.** Access предоставляет действия макросов, включая **OnError** (аналогично оператору "On Error" в VBA) и **ClearMacroError**, позволяющие выполнять определенные действия, если во время работы макроса возникнет ошибка. Кроме того, действие макроса **SingleStep** позволяет переходить в режим пошагового выполнения из любой точки макроса, чтобы поэтапно просматривать его работу.
* **Временные переменные.** Три действия макросов (**SetTempVar**, **RemoveTempVar** и **RemoveAllTempVars**) позволяют создавать и использовать временные переменные в макросах. Их можно использовать в условных выражениях, чтобы управлять выполняющимися макросами, передавать данные в отчеты или формы либо из них, а также при выполнении любой другой задачи, требующей временного хранилища для значения. Эти временные переменные также доступны в VBA, поэтому с их помощью также можно передавать данные в модули VBA и из них.

## Дополнительные сведения о действиях макросов

Во время работы в конструкторе макросов вы можете получить дополнительные сведения о действии или аргументе, щелкнув его и прочитав описание в поле в правом нижнем углу окна конструктора макросов. Кроме того, с каждым действием макроса связана статья справки. Чтобы узнать больше о действии, выберите его в списке действий и нажмите клавишу F1.

**Выполнение задания.**

Сначала в окне базы данных создают макросы «Открыть Запрос» или «Открыть Таблицу» с уникальными именами, а затем в кнопочной форме создают кнопки для вызова этих макросов.

Для одной базы данных можно создать несколько кнопочных форм. Кнопки следует группировать на страницах кнопочной формы таким образом, чтобы пользователю было понятно, в каких кнопочных формах можно выполнять определенные команды (запросы, отчеты, ввода и редактирования данных). Необходимо отметить, что на подчиненных кнопочных формах должны быть помещены кнопки возврата в главную кнопочную форму.

Технология создания кнопочных форм следующая:

* **создать страницу главной кнопочной формы (ГКФ);**
* **создать необходимое количество страниц подчиненных кнопочных форм (например, формы для ввода данных, для отчетов, для запросов и т.д.);**
* **создать элементы главной кнопочной формы;**
* **создать элементы для кнопочных форм отчетов и форм ввода или изменения данных;**
* **создать макросы для запросов или для таблиц с уникальными именами;**
* **создать элементы для кнопочных форм запросов или таблиц.**

Пример главной кнопочной формы представлен на рисунке 11.



Рисунок 11 – Структура главной кнопочной формы

Для редактирования главной кнопочной формы используется конструктор форм, он представлен на рисунке 12.

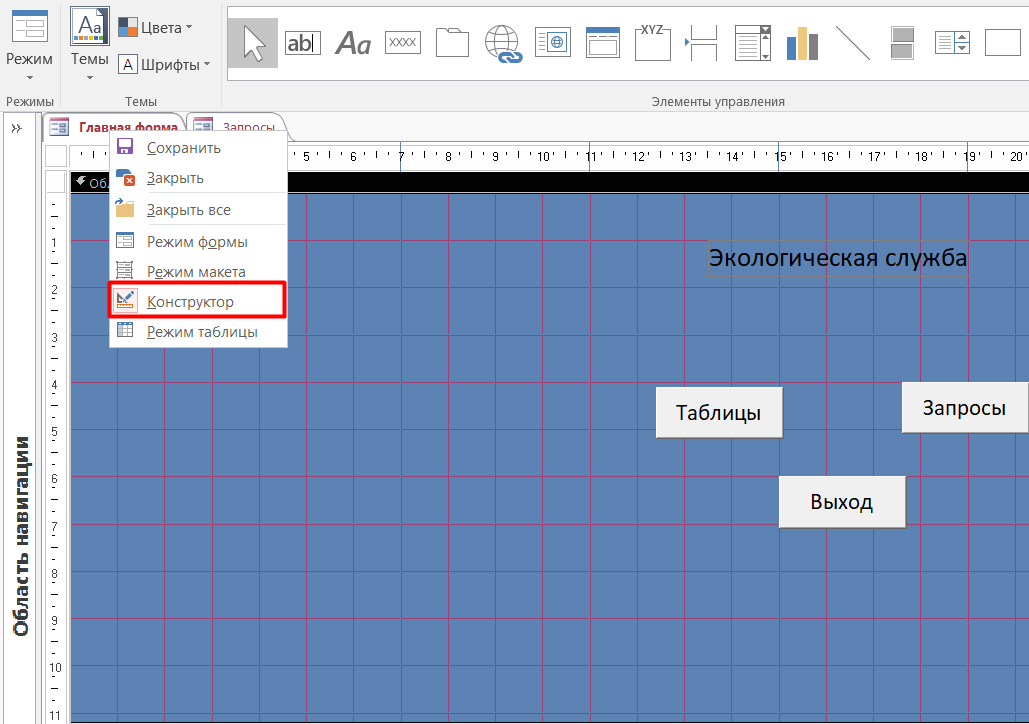


Рисунок 12 – Конструктор форм

Пример подчинённой формы представлен на рисунке 13.

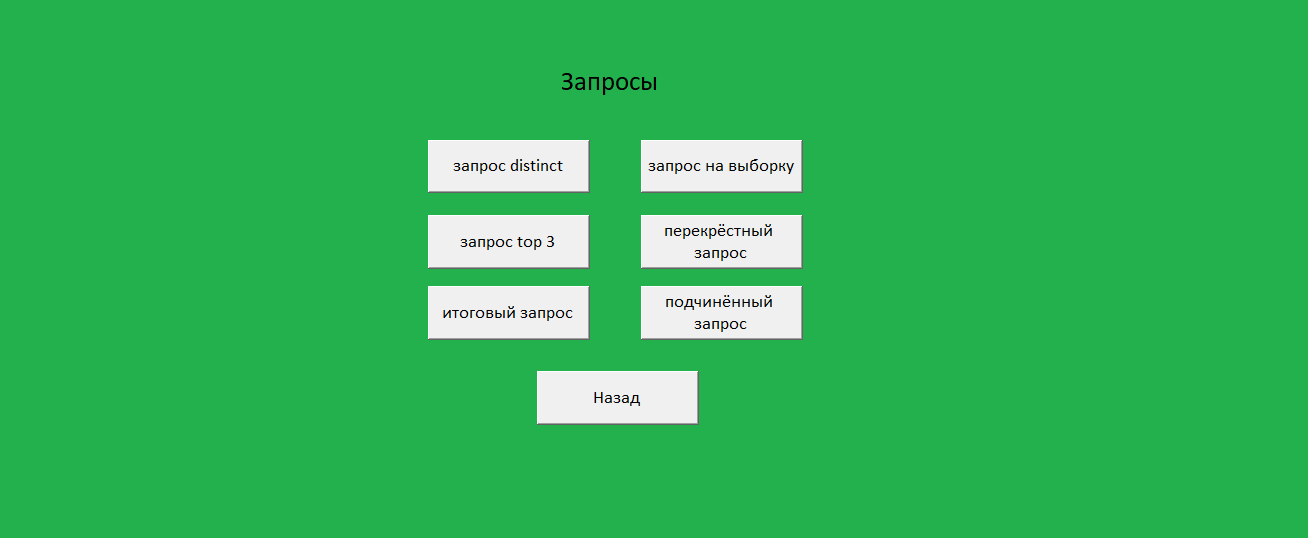


Рисунок 13 – Подчинённая форма

**Результатом выполнения практических заданий 1 и 2 являются:**

* реализованные и заполненные таблицы;
* схема данных;
* по одному запросу каждого вида:
  + выборка с условием;
  + симметричное объединение с предикатом DISTINCT;
  + симметричное объединение либо итоговый запрос с предикатом TOP n;
  + итоговый запрос с условием на данные и на группы,
  + перекрестный запрос с условием,
  + подчиненный запрос (использовать операции сравнения и один из предикатов EXISTS, ANY, ALL, IN).
* Главная кнопочная форма и две подчинённых формы, реализуемые при помощи макросов.

**Задание 3.**

**Решение оптимизационных задач в Excel.**

По заданию 3 выполняются те же номера вариантов, что и по заданиям 1 и 2.

Задание 3 посвящено решению оптимизационных задач в Excel. В качестве реализации таких задач рассматривается задача линейного раскроя, когда решение этой задачи реализуется в 2 этапа:

1-й этап: генерация всех возможных карт раскроя.

2-й этап: на построенном множестве раскройных карт находится оптимальный план раскроя в двух постановках.

**Описание задачи линейного раскроя.**

Рассмотрим процесс получения оптимального раскройного плана в задачах линейного раскроя. Разберём вариант постановки задачи с критерием на минимум использования материала (как известно, альтернативным вариантом в этих задачах является постановка с критерием на максимум выпуска продукции).

**Постановка задачи**. Для раскроя имеется линейный материал определённой длины (или один типоразмер, или несколько) в неограниченном количестве. Таким материалом могут быть доски, трубы, ковровые дорожки и прочее. Этот материал необходимо раскроить на **m** линейных заготовок в запланированных количествах **Вi**, где **i** индекс заготовки, который изменяется от 1 до **m,** израсходовав при этом минимальное количество имеющегося линейного материала. Предполагается, что материал имеется в достаточном для раскроя количестве.

На **первом этапе** производим раскрой, в результате чего получаем различные варианты раскроя линейного материала, которые называются раскройными картами. Допустим, что результатом первого этапа решения задачи получено **n** раскройных карт.

На **втором этапе** строим математическую модель оптимального раскройного план с учётом полученных раскройных карт.

Индекс раскройным картам присвоим **j**, т.е. **j** изменяется от **1** до **n**. Величину отхода по **j**-й карте обозначим **сj**. Количество **i**-йзаготовки, расположенной на **j**-й карте обозначим **аij**. Интенсивность (т.е. количество раз) использования **j**-й карты в раскройном плане обозначим **хj** (именно эти данные является искомым по решению задачи). Тогда, умножив **аij** на **хj**, получим количество **i**-йзаготовки, реализуемой по плану на **j**-й карте. Если эти произведения просуммировать по всем раскройным картам, то получаем плановое количество **i**-йзаготовки по данной задаче, т.е. **Вi**,

В качестве целевой функции в математической модели можно взять сумму интенсивностей использования раскройных карт, и этот показатель в соответствие с постановкой задачи необходимо минимизировать. Математически это можно представить следующим образом:

 (1)

Но для данного варианта целевой функции не указывается приоритет попадания и, соответственно, интенсивность использования **j**-й карты в раскройном плане, где **j** изменяется от 1 до **n**, так как все интенсивности **хj** имеют один и тот же весовой коэффициент, равный единице. А это значит, что мы не учитываем коэффициент раскроя любой из раскройных карт, т.е. нам безразлично какую величину отхода имеет раскройная карта. Понятно, что от такого варианта целевой функции раскройный план будет недостаточно оптимальным, так как в нём, без учёта приоритетов раскройных карт, могут быть задействованы раскройные карты с большой величиной отхода.

Поэтому, для усовершенствования постановки задачи, нам необходимо определить весовые коэффициенты для раскройных карт, которые бы определили приоритет попадания раскройных карт с меньшим отходом в раскройный план. И достаточно очевидно, что в качестве таких коэффициентов могут быть записаны величины **сj**, величины отхода по **j**-й карте, т.е. чем меньше отход у раскройной карты, тем быстрее целевая функция достигнет своего минимума именно при использовании данной карты.

Следовательно, вариант записи целевой функции по формуле (1) правильнее записать в следующей модификации:

 (2)

или в матричном виде: , (2')

где с - это вектор-строка остатков раскроя,  
х - это вектор-столбец интенсивностей использования раскройных карт

Область ограничений для определения минимума целевой функции определим следующим образом.

Первый вариант постановки:  (3)

или в матричном виде: , (3')

где А – это матрица элементов аij,   
B - это вектор-столбец плановых заданий.

В формулах (3) и (3') определены условия выполнения планового задания. Для поиска оптимального раскройного плана этот вариант ограничения является достаточно жёстким, и на решение задачи требуется значительно больше времени по сравнению со вторым вариантом ограничения, где, исходя из производственной ситуации, допускается некоторое перевыполнение планового задания в виде допустимого вектора перевыполнения плана . Запись второго варианта постановки по ограничения запишем следующим образом:

 (4)

или в матричном виде:  (4')

где B**'** - это вектор-столбец перевыполнения плановых заданий

Помимо ограничений (3), (3') или (4), (4'), необходимо чтобы интенсивности **хj** были неотрицательные и целочисленные, т.е.:

 (5)

Итак, мы имеем два варианта моделей оптимального раскройного плана. **Первый вариант** определяется формулами (2) или (2'); (3) или (3'); и (5), который соответствует выполнению планового задания. **Второй вариант** определяется формулами (2) или (2'); (4) или (4'); и (5), который соответствует возможному перевыполнению планового задания, и который имеет лучшие варианты по реализации оптимальности.

Примеры реализации этих моделей представим с помощью функции "Поиск решения" в системе Excel.

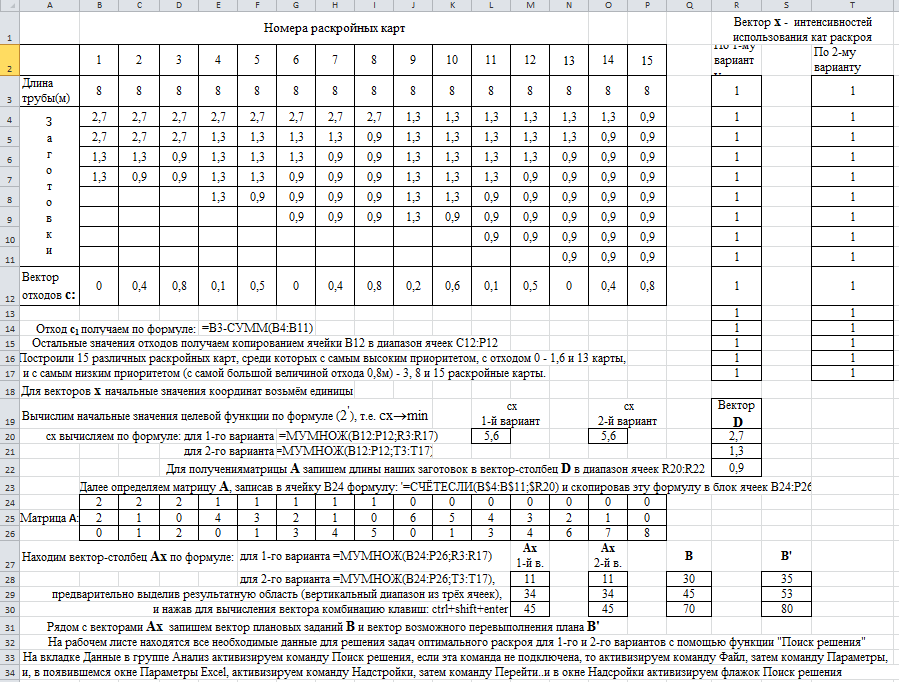
Рассмотрим следующий пример для получения оптимального раскройного плана:

Пример. На складе имеются трубы длиной 8 м в достаточном количестве. Их необходимо разрезать на куски длиной: 2,7м, 1,3м и 0,9м в плановых количествах соответственно длинам: 30, 45 и 70 штук, израсходовав минимальное количество труб. Возможен вариант некоторого перевыполнения плана, когда в результате решения план может быть перевыполнен, но значения перевыполнения не должны быть больше, чем 35, 53, 80 штук в соответствии с указанными длинами заготовок.

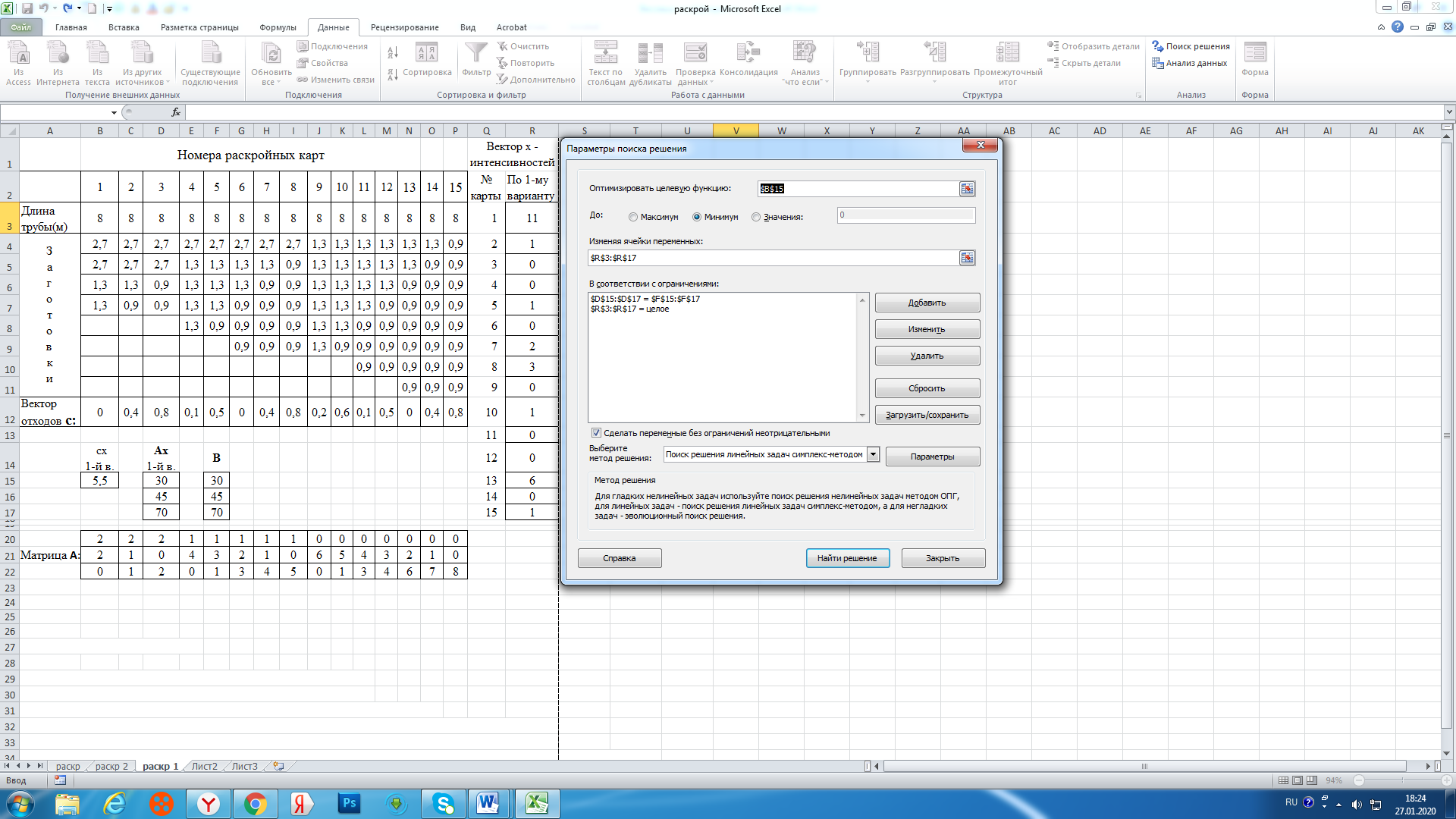
Решение.

На **первом этапе** производим раскройные карты для получения вектора отходов **с**, количества раскройных карт **n** и матрицы **А** элементов **аij**. Из условия задачи нам известно: **m=3** - количество различных заготовок, их плановое количество - вектор **В(30, 45, 70)** и вектор возможного перевыполнения плана **В'(35, 53, 80).** Для создания раскройных карт указаны длины исходного материала: 8м и длины заготовок: 2,7м, 1,3м и 0,9м. Обозначим для дальнейших рассуждений длины заготовок через вектор **D**. Для удобства реализацию раскроя будем производить на рабочем листе системы Excel. При создании раскройных карт будем придерживаться следующего алгоритма (или порядка действий): сначала кроим исходный материал на более крупные заготовки, а затем получаемые остатки докраиваем более мелкими по порядку уменьшения размеров заготовками.

Процедура реализации этого этапа раскроя представлена на следующей странице (стр. 32), а реализация **второго этапа** представлена на последующих двух страницах (стр. 33- первый вариант постановки и стр. 34 - второй вариант постановки).

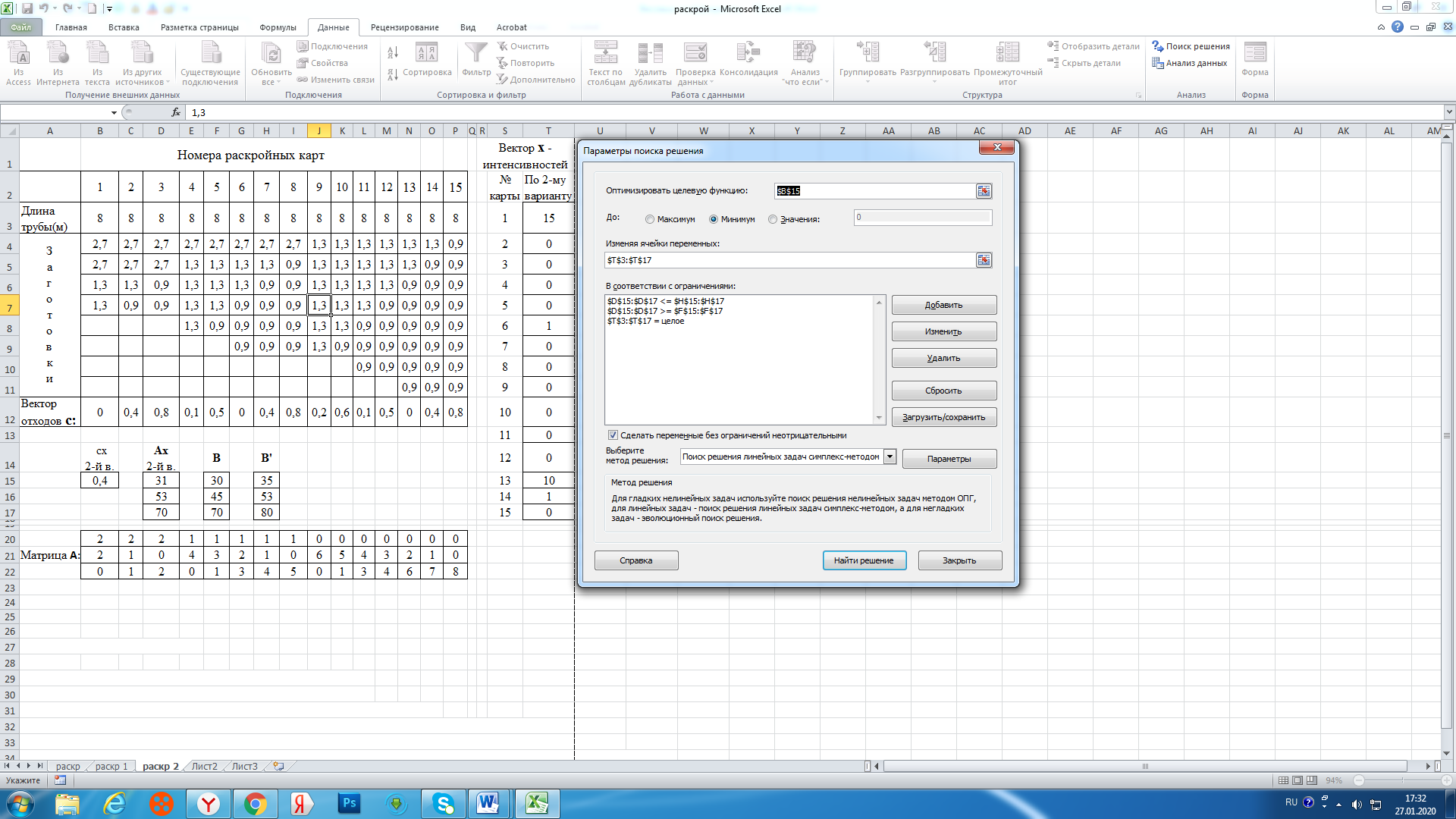


Рассмотрим решение по первому варианту: см. настройку окна Параметры поиска решения и результаты решения на рабочем листе.



1. время расчёта ≈ 20 минут; 2. Минимальное значение целевой функции 5,5 м; 3. в раскройном плане задействовано 8 видов раскройных карт: карта №1 – 11 раз (остаток 0), карта №2 – 1 раз (остаток 0,4), карта №5 – 1 раз (остаток 0,5), карта №7 – 2 раза (остаток 0,4) карта №8 – 3 раза (остаток 0,8), карта №10 – 1 раз (остаток 0,6), карта №13 – 6 раз (остаток 0), карта №15 – 1 раз (остаток 0,8). Этот раскройный план технологически является сложным для реализации, так как: необходимо 8 раз переналаживать оборудование; карты №2, №5, №10 и №15 для реализации раскроя используются только по одному разу и в раскрое приходится использовать карты №8 и №15 с самым большим отходом. Далее увидим, что вторая модель – более предпочтительна!

Рассмотрим решение по второму варианту: см. настройку окна Параметры поиска решения и результаты решения на рабочем листе.



1. время расчёта – несколько секунд; 2. значение целевой функции улучшилось (0,4); 3. План заготовок перевыполнен по первой заготовке на одну, по второй – на 8 заготовок, что допустимо по модели; 4. в раскройном плане задействовано только 4 вида раскройных карты с хорошими остатками: карта №1 – 15 раз (остаток 0), карта №6 – 1 раз (остаток 0), карта №13 – 10 раз (остаток 0), карта №14 – 1 раз (остаток 0,4). Здесь отрицательным технологическим аспектом является то, что карты №6 и №14 для реализации раскроя используются только по одному разу (а это дополнительное время на перенастройку оборудования)

**Задания по вариантам.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № варианта | Длина исходного линейного материала(м) | Размеры заготовок (м)  (вектор **D**) | | | | Плановое количество заготовок (вектор **В**) | | | | Возможное перевыполнение (вектор **В'**) | | | |
| 1-й | 2-й | 3-й | 4-й | 1-й | 2-й | 3-й | 4-й | 1-й | 2-й | 3-й | 4-й |
|  | 10 | 2,7 | 2,1 | 1,5 | 0,9 | 40 | 45 | 50 | 55 | 44 | 50 | 60 | 68 |
|  | 10 | 2,8 | 2,3 | 1,7 | 1,1 | 44 | 51 | 62 | 70 | 50 | 60 | 70 | 78 |
|  | 10 | 2,9 | 2,2 | 1,6 | 1,2 | 50 | 55 | 65 | 70 | 55 | 60 | 72 | 81 |
|  | 10 | 3 | 2,3 | 1,7 | 1,2 | 55 | 65 | 75 | 85 | 60 | 70 | 85 | 94 |
|  | 10 | 2,6 | 1,9 | 1,4 | 0,9 | 50 | 60 | 74 | 88 | 56 | 70 | 82 | 96 |
|  | 10 | 3,2 | 2,4 | 1,7 | 1,1 | 42 | 50 | 58 | 70 | 50 | 62 | 66 | 82 |
|  | 10 | 3,1 | 2,5 | 1,6 | 0,8 | 40 | 52 | 61 | 74 | 48 | 60 | 78 | 88 |
|  | 10,5 | 2,7 | 2,1 | 1,5 | 0,9 | 40 | 45 | 50 | 55 | 44 | 50 | 60 | 68 |
|  | 10,5 | 2,8 | 2,3 | 1,7 | 1,1 | 44 | 51 | 62 | 70 | 50 | 60 | 70 | 78 |
|  | 10,5 | 2,9 | 2,2 | 1,6 | 1,2 | 50 | 55 | 65 | 70 | 55 | 60 | 72 | 81 |
|  | 10,5 | 3 | 2,3 | 1,7 | 1,2 | 55 | 65 | 75 | 85 | 60 | 70 | 85 | 94 |
|  | 10,5 | 2,6 | 1,9 | 1,4 | 0,9 | 50 | 60 | 74 | 88 | 56 | 70 | 82 | 96 |
|  | 10,5 | 3,2 | 2,4 | 1,7 | 1,1 | 42 | 50 | 58 | 70 | 50 | 62 | 66 | 82 |
|  | 10,5 | 3,1 | 2,5 | 1,6 | 0,8 | 40 | 52 | 61 | 74 | 48 | 60 | 78 | 88 |
|  | 11 | 2,7 | 2,1 | 1,5 | 0,9 | 40 | 45 | 50 | 55 | 44 | 50 | 60 | 68 |
|  | 11 | 2,8 | 2,3 | 1,7 | 1,1 | 44 | 51 | 62 | 70 | 50 | 60 | 70 | 78 |
|  | 11 | 2,9 | 2,2 | 1,6 | 1,2 | 50 | 55 | 65 | 70 | 55 | 60 | 72 | 81 |
|  | 11 | 3 | 2,3 | 1,7 | 1,2 | 55 | 65 | 75 | 85 | 60 | 70 | 85 | 94 |
|  | 11 | 2,6 | 1,9 | 1,4 | 0,9 | 50 | 60 | 74 | 88 | 56 | 70 | 82 | 96 |
|  | 11 | 3,2 | 2,4 | 1,7 | 1,1 | 42 | 50 | 58 | 70 | 50 | 62 | 66 | 82 |
|  | 11 | 3,1 | 2,5 | 1,6 | 0,8 | 40 | 52 | 61 | 74 | 48 | 60 | 78 | 88 |
|  | 11,5 | 2,8 | 2,3 | 1,7 | 1,1 | 44 | 51 | 62 | 70 | 50 | 60 | 70 | 78 |
|  | 11,5 | 2,9 | 2,2 | 1,6 | 1,2 | 50 | 55 | 65 | 70 | 55 | 60 | 72 | 81 |
|  | 11,5 | 3 | 2,3 | 1,7 | 1,2 | 55 | 65 | 75 | 85 | 60 | 70 | 85 | 94 |
|  | 11,5 | 2,6 | 1,9 | 1,4 | 0,9 | 50 | 60 | 74 | 88 | 56 | 70 | 82 | 96 |

Отчёт по заданию 3 должен быть представлен, как реализация примера на страницах 32-34 этого документа, с кратким описанием выводов по результатам реализации двух вариантов постановок задачи оптимального линейного раскроя.